NuMicro® Family

Arm® Cortex®-M23-based Microcontroller

M2L31 Series
Technical Reference Manual

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro® microcontroller and microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

TABLE OF CONTENTS

1 GENERAL DESCRIPTION 35
2 FEATURES 37
2.1 M2L31 Series Features 37
3 PARTS INFORMATION 50
3.1 M2L31 Series Package Type 50
3.2 M2L31 Series Naming Rule 51
3.3 M2L31 Series Selection Guide 52
3.4 M2L31xxDAE/M2L31xx4AE Features Comparison Table 53
4 PIN CONFIGURATION 54
4.1 Pin Configuration 54
4.2 M2L31 Series Pin Diagram 54
4.2.1 QFN33-Pin Diagram 54
4.2.2 QFN48-Pin Diagram 54
4.2.3 LQFP48-Pin Diagram 55
4.2.4 LQFP64-Pin Diagram 55
4.2.5 LQFP128-Pin Diagram 55
4.3 M2L31 Series Multi-function Diagram 57
4.3.1 QFN33-Pin Multi-function Diagram 57
4.3.2 QFN48-Pin Multi-function Diagram 63
4.3.3 LQFP48-Pin Multi-function Diagram 80
4.3.4 LQFP64-Pin Multi-function Diagram 97
4.3.5 LQFP128-Pin Multi-function Diagram 113
4.4 Pin Mapping Table 125
4.4.1 M2L31 Series Mapping Table 125
4.5 Pin Functional Description 129
4.5.1 M2L31 Series Summary Function Pin Description 129
5 BLOCK DIAGRAM 140
5.1 M2L31 Series Block Diagram 140
6 FUNCTIONAL DESCRIPTION 141
6.1 Arm Cortex-M23 Core 141
6.2 System Manager 143
6.2.1 Overview 143
6.2.2 System Reset 143
6.2.3 System Power Distribution 149
6.2.4 Power Modes and Wake-up Sources 151
6.2.5 Power Modes and Power Level Transition 160

6.2.6 System Memory Map 161
6.2.7 SRAM Memory Orginization 164
6.2.8 Low Power SRAM Memory Orginization 168
6.2.9 Bus Matrix 169
6.2.10 HIRC Auto Trim 170
6.2.11 Register Lock Control 171
6.2.12 Deep Power Down Wakeup Pin 174
6.2.13 Wake-up Timer (WKTMR) and Wake-up IO (WKIO) Function 174
6.2.14 Low Power Domain System Clock 174
6.2.15 Low Power Domain Control Register Map 175
6.2.16 Low Power Domain Control Register Description 176
6.2.17 Register Map 188
6.2.18 Register Description 191
6.2.19 System Timer (SysTick) 295
6.2.20 System Timer Control Register Map 296
6.2.21 Nested Vectored Interrupt Controller (NVIC) 300
6.2.22 System Control Register 342
6.3 Clock Controller 352
6.3.1 Overview 352
6.3.2 Clock Generator 355
6.3.3 Internal Multiple Speed RC Oscillator (MIRC) 356
6.3.4 System Clock 357
6.3.5 Clock Fail Detection 358
6.3.6 Peripherals Clock 359
6.3.7 Power-down Mode Clock 359
6.3.8 Clock Output 359
6.3.9 USB Clock Source 361
6.3.10 Register Map 362
6.3.11 Register Description 364
6.4 True Random Number Generator (TRNG) 429
6.4.1 Overview 429
6.4.2 Features 429
6.4.3 Block Diagram 429
6.4.4 Basic Configuration 429
6.4.5 Functional Description 429
6.4.6 Register Map 433
6.4.7 Register Description 434
6.5 RRAM Memory Controller (RMC) 440
6.5.1 Overview 440
6.5.2 Features 440
6.5.3 Block Diagram 441
6.5.4 Functional Description 443

6.5.5 Register Map 485
6.5.6 Register Description 486
6.6 General Purpose I/O (GPIO) 508
6.6.1 Overview 508
6.6.2 Features 508
6.6.3 Block Diagram 509
6.6.4 Basic Configuration 509
6.6.5 Functional Description 509
6.6.6 Register Map 514
6.6.7 Register Description 519
6.7 Low Power General Purpose I/O (LPGPIO) 540
6.7.1 Overview 540
6.7.2 Features 540
6.7.3 Block Diagram 541
6.7.4 Basic Configuration 541
6.7.5 Functional Description 541
6.7.6 Register Map 543
6.7.7 Register Description 544
6.8 PDMA Controller (PDMA) 549
6.8.1 Overview 549
6.8.2 Features 549
6.8.3 Block Diagram 549
6.8.4 Basic Configuration 549
6.8.5 Functional Description 550
6.8.6 Register Map 556
6.8.7 Register Description 558
6.9 Low Power PDMA Controller (LPPDMA) 589
6.9.1 Overview 589
6.9.2 Features 589
6.9.3 Block Diagram 589
6.9.4 Basic Configuration 589
6.9.5 Functional Description 590
6.9.6 Register Map 595
6.9.7 Register Description 596
6.10 External Bus Interface (EBI) 618
6.10.1 Overview 618
6.10.2 Features 618
6.10.3 Block Diagram 619
6.10.4 Basic Configuration 619
6.10.5 Functional Description 619
6.10.6 Register Map 629

6.10.7 Register Description 630
6.11 Timer Controller (TMR) 634
6.11.1 Overview 634
6.11.2 Features 634
6.11.3 Block Diagram 636
6.11.4 Basic Configuration 639
6.11.5 Timer Functional Description 640
6.11.6 PWM Functional Description 647
6.11.7 Register Map 658
6.11.8 Register Description 663
6.12 Low Power Timer Controller (LPTMR) 698
6.12.1 Overview 698
6.12.2 Features 698
6.12.3 Block Diagram 700
6.12.4 Basic Configuration 703
6.12.5 Low Power Timer Functional Description 703
6.12.6 PWM Functional Description 711
6.12.7 Register Map 723
6.12.8 Register Description 726
6.13 Tick Timer Controller (TTMR) 757
6.13.1 Overview 757
6.13.2 Features 757
6.13.3 Block Diagram 758
6.13.4 Basic Configuration 760
6.13.5 Functional Description 760
6.13.6 Register Map 764
6.13.7 Register Description 765
6.14 PWM Generator and Capture Timer (PWM) 771
6.14.1 Overview 771
6.14.2 Features 771
6.14.3 Block Diagram 772
6.14.4 Basic Configuration 776
6.14.5 Functional Description 777
6.14.6 Register Map 804
6.14.7 Register Description 809
6.15 EPWM Generator and Capture Timer (EPWM) 869
6.15.1 Overview 869
6.15.2 Features 869
6.15.3 Block Diagram 870
6.15.4 Basic Configuration 875
6.15.5 Functional Description 876

6.15.6 Register Map 916
6.15.7 Register Description 924
6.16 Enhanced Quadrature Encoder Interface (EQEI) 1007
6.16.1 Overview 1007
6.16.2 Features 1007
6.16.3 Block Diagram 1007
6.16.4 Basic Configuration 1009
6.16.5 Functional Description 1009
6.16.6 Register Map 1024
6.16.7 Register Description 1025
6.17 Enhanced Input Capture Timer (ECAP) 1039
6.17.1 Overview 1039
6.17.2 Features 1039
6.17.3 Block Diagram 1039
6.17.4 Basic Configuration 1040
6.17.5 Functional Description 1041
6.17.6 Register Map 1046
6.17.7 Register Description 1047
6.18 Watchdog Timer (WDT) 1057
6.18.1 Overview 1057
6.18.2 Features 1057
6.18.3 Block Diagram 1057
6.18.4 Basic Configuration 1058
6.18.5 Functional Description 1058
6.18.6 Register Map 1061
6.18.7 Register Description 1062
6.19 Window Watchdog Timer (WWDT) 1066
6.19.1 Overview 1066
6.19.2 Features 1066
6.19.3 Block Diagram 1066
6.19.4 Basic Configuration 1067
6.19.5 Functional Description 1067
6.19.6 Register Map 1071
6.19.7 Register Description 1072
6.20 Real Time Clock (RTC) 1077
6.20.1 Overview 1077
6.20.2 Features 1077
6.20.3 Block Diagram 1078
6.20.4 Basic Configuration 1079
6.20.5 Functional Description 1079
6.20.6 Register Map 1085

6.20.7 Register Description 1087
6.21 I²C Serial Interface Controller (I²C) 1118
6.21.1 Overview 1118
6.21.2 Features 1118
6.21.3 Block Diagram 1119
6.21.4 Basic Configuration 1119
6.21.5 Functional Description 1120
6.21.6 Register Map 1146
6.21.7 Register Description 1146
6.22 Low Power I²C Serial Interface Controller (LPI2C) 1171
6.22.1 Overview 1171
6.22.2 Features 1171
6.22.3 Block Diagram 1172
6.22.4 Basic Configuration 1172
6.22.5 Functional Description 1173
6.22.6 Register Map 1190
6.22.7 Register Description 1191
6.23 USCI - Universal Serial Control Interface Controller (USCI) 1207
6.23.1 Overview 1207
6.23.2 Features 1207
6.23.3 Block Diagram 1207
6.23.4 Functional Description 1207
6.24 USCI – UART Mode 1218
6.24.1 Overview 1218
6.24.2 Features 1218
6.24.3 Block Diagram 1218
6.24.4 Basic Configuration 1218
6.24.5 Functional Description 1219
6.24.6 Register Map 1228
6.24.7 Register Description 1229
6.25 USCI - SPI Mode 1251
6.25.1 Overview 1251
6.25.2 Features 1251
6.25.3 Block Diagram 1252
6.25.4 Basic Configuration 1252
6.25.5 Functional Description 1253
6.25.6 Register Map 1265
6.25.7 Register Description 1266
6.26 USCI - I²C Mode 1288
6.26.1 Overview 1288
6.26.2 Features 1288

6.26.3 Block Diagram 1289
6.26.4 Basic Configuration 1289
6.26.5 Functional Description 1290
6.26.6 Register Map 1308
6.26.7 Register Description 1309
6.27 UART Interface Controller (UART) 1327
6.27.1 Overview 1327
6.27.2 Features 1327
6.27.3 Block Diagram 1328
6.27.4 Basic Configuration 1331
6.27.5 Functional Description 1333
6.27.6 Register Map 1356
6.27.7 Register Description 1359
6.28 Low Power UART Interface Controller (LPUART) 1396
6.28.1 Overview 1396
6.28.2 Features 1396
6.28.3 Block Diagram 1397
6.28.4 Basic Configuration 1399
6.28.5 Functional Description 1400
6.28.6 Register Map 1417
6.28.7 Register Description 1419
6.29 Serial Peripheral Interface (SPI) 1452
6.29.1 Overview 1452
6.29.2 Features 1452
6.29.3 Block Diagram 1452
6.29.4 Basic Configuration 1453
6.29.5 Functional Description 1454
6.29.6 Timing Diagram 1472
6.29.7 Programming Examples 1474
6.29.8 Register Map 1477
6.29.9 Register Description 1478
6.30 Low Power Serial Peripheral Interface (LPSPI) 1501
6.30.1 Overview 1501
6.30.2 Features 1501
6.30.3 Block Diagram 1501
6.30.4 Basic Configuration 1502
6.30.5 Functional Description 1503
6.30.6 Timing Diagram 1519
6.30.7 Programming Examples 1521
6.30.8 Register Map 1524
6.30.9 Register Description 1525
6.31 Quad Serial Peripheral Interface (QSPI) 1543

6.31.1 Overview 1543
6.31.2 Features 1543
6.31.3 Block Diagram 1543
6.31.4 Basic Configuration 1544
6.31.5 Functional Description 1545
6.31.6 Timing Diagram 1562
6.31.7 Programming Examples 1564
6.31.8 Register Map 1567
6.31.9 Register Description 1568
6.32 USB 2.0 Full-Speed Device Controller (USBD) 1583
6.32.1 Overview 1583
6.32.2 Features 1583
6.32.3 Block Diagram 1584
6.32.4 Basic Configuration 1584
6.32.5 Functional Description 1584
6.32.6 Register Map 1589
6.32.7 Register Description 1593
6.33 USB 1.1 Host Controller (USBH) 1622
6.33.1 Overview 1622
6.33.2 Features 1622
6.33.3 Block Diagram 1623
6.33.4 Basic Configuration 1624
6.33.5 Functional Description 1624
6.33.6 Register Map 1626
6.33.7 Register Description 1627
6.34 USB On-The-Go (OTG) 1658
6.34.1 Overview 1658
6.34.2 Features 1658
6.34.3 Block Diagram 1658
6.34.4 Basic Configuration 1659
6.34.5 Functional Description 1659
6.34.6 Register Map 1663
6.34.7 Register Description 1664
6.35 USB Type C Power Delivery Controller (UTCPD) 1673
6.35.1 Overview 1673
6.35.2 Features 1673
6.35.3 Block Diagram 1675
6.35.4 Basic Configuration 1675
6.35.5 Functional Description 1675
6.35.6 Register Map 1684
6.35.7 Register Description 1687
6.36 Controller Area Network with Flexible DataRate (CAN FD) 1763

6.36.1 Overview 1763
6.36.2 Features 1763
6.36.3 Block Diagram 1763
6.36.4 Basic Configuration 1765
6.36.5 Functional Description 1765
6.36.6 Register Map 1794
6.36.7 Register Description 1796
6.37 Key Store (KS) 1848
6.37.1 Overview 1848
6.37.2 Features 1848
6.37.3 Block Diagram 1848
6.37.4 Basic Configuration 1849
6.37.5 Functional Description 1849
6.37.6 Register Map 1853
6.37.7 Register Description 1854
6.38 Cryptographic Accelerator (CRYPTO) 1860
6.38.1 Overview 1860
6.38.2 Features 1860
6.38.3 Block Diagram 1860
6.38.4 Basic Configuration 1861
6.38.5 Functional Description 1861
6.38.6 Register Map 1871
6.38.7 Register Description 1873
6.39 Touch Key (TK) 1892
6.39.1 Overview 1892
6.39.2 Features 1892
6.39.3 Block Diagram 1892
6.39.4 Basic Configuration 1892
6.39.5 Functional Description 1893
6.39.6 Register Map 1898
6.39.7 Register Description 1900
6.40 CRC Controller (CRC) 1966
6.40.1 Overview 1966
6.40.2 Features 1966
6.40.3 Block Diagram 1966
6.40.4 Basic Configuration 1966
6.40.5 Functional Description 1967
6.40.6 Register Map 1970
6.40.7 Register Description 1971
6.41 Analog Comparator Controller (ACMP) 1981
6.41.1 Overview 1981

6.41.2 Features 1981
6.41.3 Block Diagram 1983
6.41.4 Basic Configuration 1983
6.41.5 Functional Description 1984
6.41.6 Register Map 1990
6.41.7 Register Description 1991
6.42 Enhanced 12-bit Analog-to-Digital Converter (EADC) 2009
6.42.1 Overview 2009
6.42.2 Features 2009
6.42.3 Block Diagram 2010
6.42.4 Basic Configuration 2010
6.42.5 Functional Description 2011
6.42.6 Register Map 2024
6.42.7 Register Description 2029
6.43 Low Power Analog-to-Digital Converter (LPADC) 2076
6.43.1 Overview 2076
6.43.2 Features 2076
6.43.3 Block Diagram 2077
6.43.4 Basic Configuration 2077
6.43.5 Functional Description 2078
6.43.6 Register Map 2090
6.43.7 Register Description 2092
6.44 Digital to Analog Converter (DAC) 2116
6.44.1 Overview 2116
6.44.2 Features 2116
6.44.3 Block Diagram 2117
6.44.4 Basic Configuration 2117
6.44.5 Functional Description 2118
6.44.6 Register Map 2123
6.44.7 Register Description 2124
6.45 OP Amplifier (OPA) 2139
6.45.1 Overview 2139
6.45.2 Features 2139
6.45.3 Block Diagram 2140
6.45.4 Basic Configuration 2141
6.45.5 Functional Description 2141
6.45.6 Register Map 2143
6.45.7 Register Description 2144
6.46 Peripherals Interconnection 2158
6.46.1 Overview 2158
6.46.2 Peripherals Interconnect Matrix Table 2158

6.46.3 Functional Description 2159
7 APPLICATION CIRCUIT 2166
7.1 Power Supply Scheme 2166
7.2 Peripheral Application Scheme 2167
8 ELECTRICAL CHARACTERISTIC 2168
9 ABBREVIATIONS 2169
9.1 Abbreviations 2169
10 PACKAGE DIMENSIONS 2171
10.1 QFN 33L (5x5x0.8 mm, Pitch 0.5 mm, Lead Length 0.4 mm) 2171
10.2 QFN 48L (5x5x0.8 mm, Pitch 0.35 mm, Lead Length 0.4 mm) 2172
10.3 LQFP 48L (7x7x1.4 mm Footprint 2.0 mm) 2173
10.4 LQFP 64L (7x7x1.4 mm Footprint 2.0 mm) 2174
10.5 LQFP 128L (14x14x1.4 mm Footprint 2.0 mm) 2175
11 REVISION HISTORY 2176

LIST OF FIGURES

Figure 4.2-1 QFN33-Pin Diagram 54
Figure 4.2-2 QFN48-Pin Diagram 55
Figure 4.2-3 LQFP48-Pin Diagram 55
Figure 4.2-4 LQFP64-Pin Diagram 55
Figure 4.2-5 LQFP128-Pin Diagram 56
Figure 4.3-1 M2L31ZE4AE Multi-function Pin Diagram 57
Figure 4.3-2 M2L31ZD4AE Multi-function Pin Diagram 60
Figure 4.3-3 M2L31YG4AE Multi-function Pin Diagram 63
Figure 4.3-4 M2L31YE4AE Multi-function Pin Diagram 66
Figure 4.3-5 M2L31YD4AE Multi-function Pin Diagram 69
Figure 4.3-6 M2L31YIDAE Multi-function Pin Diagram 72
Figure 4.3-7 M2L31YGDAE Multi-function Pin Diagram 76
Figure 4.3-8 M2L31LE4AE Multi-function Pin Diagram 80
Figure 4.3-9 M2L31LG4AE Multi-function Pin Diagram 83
Figure 4.3-10 M2L31LD4AE Multi-function Pin Diagram 86
Figure 4.3-11 M2L31LIDAE Multi-function Pin Diagram 89
Figure 4.3-12 M2L31LGDAE Multi-function Pin Diagram 93
Figure 4.3-13 M2L31SG4AE Multi-function Pin Diagram 97
Figure 4.3-14 M2L31SE4AE Multi-function Pin Diagram 101
Figure 4.3-15 M2L31SIDAE Multi-function Pin Diagram 105
Figure 4.3-16 M2L31SGDAE Multi-function Pin Diagram 109
Figure 4.3-17 M2L31KIDAE Multi-function Pin Diagram 113
Figure 4.3-18 M2L31KGDAE Multi-function Pin Diagram 119
Figure 5.1-1 M2L31 Series Block Diagram 140
Figure 6.1-1 Cortex-M23 Block Diagram 141
Figure 6.2-1 System Reset Sources 144
Figure 6.2-2 nRESET Reset Waveform 146
Figure 6.2-3 Power-on Reset (POR) Waveform 147
Figure 6.2-4 Low Voltage Reset (LVR) Waveform 147
Figure 6.2-5 Brown-out Detector (BOD) Waveform 148
Figure 6.2-6 NuMicro M2L31 Power Distribution Diagram 150
Figure 6.2-7 Power Mode State Machine 153
Figure 6.2-8 NuMicro M2L31 Power Mode Transition 160
Figure 6.2-9 SRAM Block Diagram 164
Figure 6.2-10 SRAM Memory Organization 165
Figure 6.2-11 SRAM Marco Organization 166

Figure 6.2-12 LPSRAM Block Diagram 168
Figure 6.2-13 NuMicro M2L31 Bus Matrix Diagram 169
Figure 6.2-14 Low Power Domain System Clock Block Diagrams 174
Figure 6.3-1 Clock Generator Global View Diagram 353
Figure 6.3-2 Low Power Domain Clock Generator Global View Diagram 354
Figure 6.3-3 Clock Generator Block Diagram 355
Figure 6.3-4 System Clock Block Diagrams 357
Figure 6.3-5 SysTick Clock Control Block Diagram 357
Figure 6.3-6 HXT Stop Protect Procedure 358
Figure 6.3-7 LXT Stop Protect Procedure 359
Figure 6.3-8 Clock Output Block Diagram 360
Figure 6.3-9 USB Clock Block Diagram 361
Figure 6.4-1 True Random Number Generator Block Diagram 429
Figure 6.5-1 RRAM Memory Controller Block Diagram 441
Figure 6.5-2 Data RRAM Shared with APROM Examples 444
Figure 6.5-3 Address Operation Model 445
Figure 6.5-4 New XOM Setting Flow 452
Figure 6.5-5 OTP Memory Map 465
Figure 6.5-6 RRAM Memory Map 466
Figure 6.5-7 System Memory Map with IAP Mode 467
Figure 6.5-8 LDROM with IAP Mode 468
Figure 6.5-9 APROM with IAP Mode 468
Figure 6.5-10 ISP Procedure Example 471
Figure 6.5-11 ISP 32-bit Single Word Programming Procedure 472
Figure 6.5-12 ISP 32-bit Word Line Programming Procedure 473
Figure 6.5-13 RRAM CRC32 Checksum Calculation 474
Figure 6.5-14 Lock Effect Table with Two Protections for ICE/ ICP/Writer 484
Figure 6.6-1 GPIO Controller Block Diagram 509
Figure 6.6-2 Push-Pull Output 510
Figure 6.6-3 Open-Drain Output 510
Figure 6.6-4 Quasi-Bidirectional I/O Mode 511
Figure 6.6-5 GPIO Rising Edge Trigger Interrupt 512
Figure 6.6-6 GPIO Falling Edge Trigger Interrupt 512
Figure 6.7-1 LPGPIO Controller Block Diagram 541
Figure 6.7-2 Push-Pull Output 542
Figure 6.8-1 PDMA Controller Block Diagram 549
Figure 6.8-2 Descriptor Table Entry Structure 550

Figure 6.8-3 Basic Mode Finite State Machine 551
Figure 6.8-4 Descriptor Table Link List Structure 552
Figure 6.8-5 Scatter-gather Mode Finite State Machine 553
Figure 6.8-6 Example of Single Transfer Type and Burst Transfer Type in Basic Mode 554
Figure 6.8-7 Example of PDMA Channel 0 Time-out Counter Operation 555
Figure 6.9-1 LPPDMA Controller Block Diagram 589
Figure 6.9-2 Descriptor Table Entry Structure 590
Figure 6.9-3 Basic Mode Finite State Machine 591
Figure 6.9-4 Descriptor Table Link List Structure 592
Figure 6.9-5 Scatter-gather Mode Finite State Machine 592
Figure 6.9-6 Example of Single Transfer Type and Burst Transfer Type in Basic Mode 593
Figure 6.10-1 EBI Block Diagram 619
Figure 6.10-2 Connection of 16-bit EBI Data Width with 16-bit Device 620
Figure 6.10-3 Connection of 8-bit EBI Data Width with 8-bit Device 621
Figure 6.10-4 Connection of 16-bit EBI Data Width with 16-bit Device in Separate mode 621
Figure 6.10-5 Connection of 8-bit EBI Data Width with 8-bit Device in Separate mode 622
Figure 6.10-6 Timing Control Waveform for 16-bit Data Width 623
Figure 6.10-7 Timing Control Waveform for 8-bit Data Width 624
Figure 6.10-8 Timing Control Waveform for Byte Write in 16-bit Data Mode 625
Figure 6.10-9 Timing Control Waveform for Insert Idle Cycle 626
Figure 6.10-10 Timing Control Waveform for 16-bit Data Width for Separate Mode 627
Figure 6.10-11 Timing Control Waveform for Continuous Data Access Mode 628
Figure 6.11-1 Timer Controller Block Diagram 636
Figure 6.11-2 Clock Source of Timer Controller 637
Figure 6.11-3 PWM Generator Overview Block Diagram 638
Figure 6.11-4 PWM Architecture Diagram 639
Figure 6.11-5 Continuous Counting Mode 641
Figure 6.11-6 External Capture Mode 642
Figure 6.11-7 Reset Counter Mode 643
Figure 6.11-8 Internal Timer Trigger 644
Figure 6.11-9 Capture interrupt trigger PDMA 645
Figure 6.11-10 Inter-Timer Trigger Capture Timing 646
Figure 6.11-11 Timer PWM Block Diagram 647
Figure 6.11-12 PWM Prescale Waveform in Up Count Type 648
Figure 6.11-13 PWM Up Count Type 648
Figure 6.11-14 PWM Comparator Events in Up Count Type 649
Figure 6.11-15 Period Loading Mode with Up Count Type 650

Figure 6.11-16 PWM Pulse Generation in Up Count Type 651
Figure 6.11-17 PWM 0% to 100% Duty Cycle in Up Count Type 651
Figure 6.11-18 PWM Output PWMx_OUT Control 652
Figure 6.11-19 PWMx_OUT Polarity Control 652
Figure 6.11-20 PWM Interrupt Architecture Diagram 653
Figure 6.11-21 PWM Wake-up Architecture Diagram 654
Figure 6.11-22 PWM Trigger Block Diagram 654
Figure 6.11-23 PWM Accumulator Stop Mode Waveform 657
Figure 6.12-1 Low Power Timer Controller Block Diagram 700
Figure 6.12-2 Clock Source of Low Power Timer Controller 701
Figure 6.12-3 PWM Generator Overview Block Diagram 702
Figure 6.12-4 PWM Architecture Diagram 703
Figure 6.12-5 Continuous Counting Mode 705
Figure 6.12-6 External Capture Mode 706
Figure 6.12-7 Reset Counter Mode 707
Figure 6.12-8 Internal Low Power Timer Trigger 708
Figure 6.12-9 Capture interrupt trigger LPPDMA 709
Figure 6.12-10 Low Power Timer PWM Block Diagram 711
Figure 6.12-11 PWM Prescale Waveform in Up Count Type 712
Figure 6.12-12 PWM Up Count Type 712
Figure 6.12-13 PWM Comparator Events in Up Count Type 713
Figure 6.12-14 Period Loading Mode with Up Count Type 714
Figure 6.12-15 PWM Pulse Generation in Up Count Type 715
Figure 6.12-16 PWM 0% to 100% Duty Cycle in Up Count Type 715
Figure 6.12-17 PWM Output PWMx_OUT Control 716
Figure 6.12-18 PWMx_OUT Polarity Control 716
Figure 6.12-19 PWM Interrupt Architecture Diagram 717
Figure 6.12-20 PWM Wake-up Architecture Diagram 718
Figure 6.12-21 PWM Trigger Block Diagram 718
Figure 6.12-22 PWM Accumulator Stop Mode Waveform 721
Figure 6.13-1 Tick Timer Controller Block Diagram 758
Figure 6.13-2 Clock Source of Tick Timer Controller 759
Figure 6.13-3 Continuous Counting Mode 761
Figure 6.13-4 Internal Tick Timer Trigger 762
Figure 6.14-1 PWM Generator Overview Block Diagram 772
Figure 6.14-2 PWM System Clock Source Control 773
Figure 6.14-3 PWM Clock Source Control 774

Figure 6.14-4 PWM Independent Mode Architecture Diagram 775
Figure 6.14-5 PWM Complementary Mode Architecture Diagram 776
Figure 6.14-6 PWM0_CH0 Prescaler Waveform in Up Counter Type 777
Figure 6.14-7 PWMx Counter Waveform when Setting clear counter 777
Figure 6.14-8 PWM Up Counter Type 778
Figure 6.14-9 PWM Down Counter Type 778
Figure 6.14-10 PWM Up-Down Counter Type 779
Figure 6.14-11 PWM Compared point Events in Up-Down Counter Type 780
Figure 6.14-12 PWM Double Buffering Illustration 781
Figure 6.14-13 Period Loading in Up-Count Mode 782
Figure 6.14-14 Immediately Loading in Up-Count Mode 783
Figure 6.14-15 Center Loading in Up-Down-Count Mode 784
Figure 6.14-16 PWM Pulse Generation 785
Figure 6.14-17 PWM 0% to 100% Pulse Generation 785
Figure 6.14-18 PWM Independent Mode Waveform 786
Figure 6.14-19 PWM Complementary Mode Waveform 787
Figure 6.14-20 PWMx_CH0 Output Control in Independent Mode 787
Figure 6.14-21 PWMx_CH0 and PWMx_CH1 Output Control in Complementary Mode 788
Figure 6.14-22 Dead-Time Insertion 789
Figure 6.14-23 Illustration of Mask Control Waveform 789
Figure 6.14-24 Brake Noise Filter Block Diagram 790
Figure 6.14-25 Brake Block Diagram for PWMx_CH0 and PWMx_CH1 Pair 791
Figure 6.14-26 Edge Detector Waveform for PWMx_CH0 and PWMx_CH1 Pair 792
Figure 6.14-27 Level Detector Waveform for PWMx_CH0 and PWMx_CH1 Pair 792
Figure 6.14-28 Brake Source Block Diagram 793
Figure 6.14-29 Brake System Fail Block Diagram 794
Figure 6.14-30 Initial State and Polarity Control with Rising Edge Dead-Time Insertion 794
Figure 6.14-31 PWMx_CH0 Accumulate Interrupt Waveform 795
Figure 6.14-32 PWM_CH0 and PWM_CH1 Pair Interrupt Architecture Diagram 796
Figure 6.14-33 PWMx_CH0 and PWMx_CH1 Pair Trigger EADC Block Diagram 797
Figure 6.14-34 PWM Trigger LPADC Block Diagram 797
Figure 6.14-35 PWM Trigger EADC/LPADC in Up-Down Counter Type Timing Waveform 798
Figure 6.14-36 PWM_CH0 and PWM_CH1 Pair Trigger DAC Block Diagram 798
Figure 6.14-37 PWM_CH0 Capture Block Diagram 799
Figure 6.14-38 Capture Operation Waveform 800
Figure 6.14-39 Capture Trigger PDMA Function Architecture Diagram for PWM_CH0 and PWM_CH1 Pair 801
Figure 6.14-40 Capture PDMA Operation Waveform of Channel 0 802

Figure 6.14-41 Accumulator Trigger PDMA Function Architecture Diagram 802
Figure 6.14-42 PWM Accumulator Stop Mode Waveform 803
Figure 6.15-1 EPWM Generator Overview Block Diagram 871
Figure 6.15-2 EPWM Clock Source Control 872
Figure 6.15-3 EPWM Clock Source Control 873
Figure 6.15-4 EPWM Independent Mode Architecture Diagram 874
Figure 6.15-5 EPWM Complementary Mode Architecture Diagram 875
Figure 6.15-6 EPWM_CH0 Prescaler Waveform in Up Counter Type 876
Figure 6.15-7 EPWM Counter Waveform when Setting Clear Counter 876
Figure 6.15-8 EPWM Up Counter Type 877
Figure 6.15-9 EPWM Down Counter Type 877
Figure 6.15-10 EPWM Up-Down Counter Type 878
Figure 6.15-11 EPWM Compared point Events in Up-Down Counter Type 879
Figure 6.15-12 EPWM Double Buffering Illustration 880
Figure 6.15-13 Period Loading in Up-Count Mode 881
Figure 6.15-14 Immediately Loading in Up-Count Mode 882
Figure 6.15-15 Window Loading in Up-Count Mode 883
Figure 6.15-16 Center Loading in Up-Down-Count Mode 884
Figure 6.15-17 EPWM One-shot Mode Output Waveform 885
Figure 6.15-18 EPWM Pulse Generation 886
Figure 6.15-19 Software Event Output Force 886
Figure 6.15-20 EPWM 0% to 100% Pulse Generation 887
Figure 6.15-21 EPWM Independent Mode Waveform 888
Figure 6.15-22 EPWM Complementary Mode Waveform 888
Figure 6.15-23 EPWM Group Function Waveform 889
Figure 6.15-24 EPWM SYNC_IN and SYNC_OUT Signal Block Diagram 890
Figure 6.15-25 EPWM SYNC_IN Noise Filter Block Diagram 891
Figure 6.15-26 EPWM Counter Synchronous Function Block Diagram 892
Figure 6.15-27 EPWM Synchronous Function with Synchronize source from SYNC_IN Signal 893
Figure 6.15-28 EPWMx_CH0 Output Control in Independent Mode 893
Figure 6.15-29 EPWMx_CH0 and EPWMx_CH1 Output Control in Complementary Mode 894
Figure 6.15-30 Dead-Time Insertion 895
Figure 6.15-31 Dead-Time Independent Control 895
Figure 6.15-32 Illustration of Mask Control Waveform 896
Figure 6.15-33 Brake Noise Filter Block Diagram 896
Figure 6.15-34 Brake Block Diagram for EPWMx_CH0 and EPWMx_CH1 Pair 897
Figure 6.15-35 Edge Detector Waveform for EPWMx_CH0 and EPWMx_CH1 Pair 898

Figure 6.15-36 Level Detector Waveform for EPWMx_CH0 and EPWMx_CH1 Pair 898
Figure 6.15-37 Brake Source Block Diagram 899
Figure 6.15-38 Brake System Fail Block Diagram 900
Figure 6.15-39 EPWM LEB Function Waveform 900
Figure 6.15-40 Initial State and Polarity Control with Rising Edge Dead-Time Insertion 901
Figure 6.15-41 EPWMx_CH0 Accumulate Interrupt Waveform 902
Figure 6.15-42 EPWMx_CH0 and EPWMx_CH1 Pair Interrupt Architecture Diagram 903
Figure 6.15-43 Fault Detect Function Interrupt Architecture Diagram 904
Figure 6.15-44 EPWMx_CH0 and EPWMx_CH1 Pair Trigger EADC Events 905
Figure 6.15-45 EPWM Trigger LPADC Block Diagram 905
Figure 6.15-46 EPWMx_CH0 Trigger EADC/LPADC Block Diagram 906
Figure 6.15-47 EPWM Trigger EADC/LPADC in Up-Down Counter Type Timing Waveform 906
Figure 6.15-48 EPWM_CH0 and EPWM_CH1 Pair Trigger DAC Block Diagram 907
Figure 6.15-49 EPWM_CH0 Capture Block Diagram 908
Figure 6.15-50 Capture Operation Waveform 909
Figure 6.15-51 Capture Trigger PDMA Function Architecture Diagram for EPWM_CH0 and EPWM_CH1 Pair 910
Figure 6.15-52 Capture PDMA Operation Waveform of Channel 0 911
Figure 6.15-53 Accumulator Trigger PDMA Function Architecture Diagram 912
Figure 6.15-54 EPWM Accumulator Stop Mode Waveform 913
Figure 6.15-55 Fault Detect Function Architecture 914
Figure 6.15-56 EPWM Mask Function Waveform 915
Figure 6.15-57 EPWM Deglitch Function Waveform 915
Figure 6.16-1 EQEI Block Diagram 1008
Figure 6.16-2 EQEI Decoder Unit 1008
Figure 6.16-3 EQEI Clock Source Control 1009
Figure 6.16-4 Noise Filter 1010
Figure 6.16-5 Noise Filter Sampling Clock Selection 1010
Figure 6.16-6 QEA/QEB/IDX Timing Requirement Through Noise Filter 1010
Figure 6.16-7 X4 Counting Mode 1012
Figure 6.16-8 X2 Counting Mode 1013
Figure 6.16-9 Compare Operation 1014
Figure 6.16-10 Phase Counting Mode, Mode1 1015
Figure 6.16-11 Phase Counting Mode, Mode2 1015
Figure 6.16-12 X1 Phase Counting Mode, Mode1, Rising Trigger 1015
Figure 6.16-13 X1 Phase Counting Mode, Mode1, Falling Trigger 1016
Figure 6.16-14 X2 Phase Counting Mode, Mode1 1016
Figure 6.16-15 X1 Phase Counting Mode, Mode2, Rising Trigger 1016

Figure 6.16-16 X1 Phase Counting Mode, Mode2, Falling Trigger 1017
Figure 6.16-17 X2 Phase Counting Mode, Mode2..... 1017
Figure 6.16-18 X2 Direction Count Mode 1018
Figure 6.16-19 x2 Direction Up Count Mode 1018
Figure 6.16-20 X2 Direction Down Count Mode 1018
Figure 6.16-21 EQEI Swap Function for QA and QB..... 1019
Figure 6.16-22 EQEI Unit Timer Counter Control Flow Chart 1020
Figure 6.16-23 EQEI_CNT Reload/Reset Control 1020
Figure 6.16-24 Trigger Control of Capturing EQEI Counter 1021
Figure 6.16-25 Capture and Latch EQEI Counter..... 1022
Figure 6.16-26 Enhanced Quadrature Encoder Interface Interrupt Architecture Diagram 1023
Figure 6.16-27 EQEI Phase Variation for CHA and CHB 1023
Figure 6.17-1 Input Capture Timer/Counter Architecture 1039
Figure 6.17-2 Input Capture Timer/Counter Clock Source Control..... 1041
Figure 6.17-3 Noise Filter Sampling Clock Selection..... 1042
Figure 6.17-4 Input Capture Timer/Counter Function Block 1043
Figure 6.17-5 Input Capture Timer/Counter Interrupt Architecture Diagram 1045
Figure 6.18-1 Watchdog Timer Clock Control..... 1057
Figure 6.18-2 Watchdog Timer Block Diagram..... 1058
Figure 6.18-3 Watchdog Timer Time-out Interval and Reset Period Timing 1060
Figure 6.19-1 WWDT Clock Control..... 1066
Figure 6.19-2 WWDT Block Diagram..... 1066
Figure 6.19-3 WWDT Reset and Reload Behavior 1068
Figure 6.19-4 WWDT Reload Counter When CNTDAT > CMPDAT 1069
Figure 6.19-5 WWDT Reload Counter When CNTDAT <= CMPDAT 1069
Figure 6.19-6 WWDT Interrupt and Reset Signals 1070
Figure 6.20-1 RTC Block Diagram..... 1078
Figure 6.20-2 Backup I/O Control Diagram..... 1084
Figure 6.21-1 I^{2}C Controller Block Diagram..... 1119
Figure 6.21-2 I^{2}C Bus Timing..... 1120
Figure 6.21-3 I^{2}C Protocol..... 1121
Figure 6.21-4 START and STOP Conditions..... 1121
Figure 6.21-5 Bit Transfer on the I^{2}C Bus..... 1122
Figure 6.21-6 Acknowledge on the I^{2}C Bus 1122
Figure 6.21-7 Master Transmits Data to Slave by 7-bit 1123
Figure 6.21-8 Master Reads Data from Slave by 7-bit..... 1123
Figure 6.21-9 Master Transmits Data to Slave by 10-bit 1123

Figure 6.21-10 Master Reads Data from Slave by 10-bit 1124
Figure 6.21-11 Control I²C Bus according to the Current I²C Status 1124
Figure 6.21-12 Master Transmitter Mode Control Flow 1125
Figure 6.21-13 Master Receiver Mode Control Flow 1126
Figure 6.21-14 Slave Mode Control Flow 1127
Figure 6.21-15 GC Mode 1128
Figure 6.21-16 Arbitration Lost 1129
Figure 6.21-17 Bus Management Packet Protocol Diagram Element Key 1131
Figure 6.21-187-bit Addressable Device to Host Communication 1132
Figure 6.21-197-bit Addressable Device Responds to an ARA 1132
Figure 6.21-20 Bus Management ALERT Function 1133
Figure 6.21-21 Bus Management Time Out Timing 1134
Figure 6.21-22 Bus Clock Low Time Out Timing 1134
Figure 6.21-23 Two-level Buffer Mode Slave Read Control 1136
Figure 6.21-24 Two-level Buffer Mode Slave Write Control 1137
Figure 6.21-25 Two-level Buffer Mode Slave Read Limitation 1138
Figure 6.21-26 Setup Time Wrong Adjustment 1139
Figure 6.21-27 Hold Time Wrong Adjustment 1139
Figure 6.21-28 I²C Data Shifting Direction 1140
Figure 6.21-29 I²C Time-out Count Block Diagram 1142
Figure 6.21-30 I²C Wake-Up Related Signals Waveform 1143
Figure 6.21-31 EEPROM Random Read 1145
Figure 6.21-32 Protocol of EEPROM Random Read 1145
Figure 6.22-1 LPI2C Controller Block Diagram 1172
Figure 6.22-2 I²C Bus Timing 1173
Figure 6.22-3 I²C Protocol 1173
Figure 6.22-4 START and STOP Conditions 1174
Figure 6.22-5 Bit Transfer on the I²C Bus 1175
Figure 6.22-6 Acknowledge on the I²C Bus 1175
Figure 6.22-7 Master Transmits Data to Slave by 7-bit 1175
Figure 6.22-8 Master Reads Data from Slave by 7-bit 1176
Figure 6.22-9 Control I²C Bus according to the Current LPI2C Status 1176
Figure 6.22-10 Master Transmitter Mode Control Flow 1177
Figure 6.22-11 Master Receiver Mode Control Flow 1178
Figure 6.22-12 Slave Mode Control Flow 1179
Figure 6.22-13 GC Mode 1180
Figure 6.22-14 Setup Time Wrong Adjustment 1182

Figure 6.22-15 Hold Time Wrong Adjustment 1182
Figure 6.22-16 LPI2C Data Shifting Direction 1183
Figure 6.22-17 LPI2C Time-out Count Block Diagram 1185
Figure 6.22-18 LPI2C Wake-Up Related Signals Waveform 1185
Figure 6.22-19 EEPROM Random Read 1186
Figure 6.22-20 Protocol of EEPROM Random Read 1187
Figure 6.22-21 LPI2C Automatic Operation Block Diagram 1188
Figure 6.23-1 USCI Block Diagram 1207
Figure 6.23-2 Input Conditioning for USCIx_DAT[1:0] and USCIx_CTL[1:0] 1208
Figure 6.23-3 Input Conditioning for USCIx_CLK 1209
Figure 6.23-4 Block Diagram of Data Buffering 1210
Figure 6.23-5 Data Access Structure 1211
Figure 6.23-6 Transmit Data Path 1211
Figure 6.23-7 Receive Data Path 1212
Figure 6.23-8 Protocol-Relative Clock Generator 1213
Figure 6.23-9 Basic Clock Divider Counter 1214
Figure 6.23-10 Block of Timing Measurement Counter 1214
Figure 6.23-11 Sample Time Counter 1215
Figure 6.23-12 Event and Interrupt Structure 1216
Figure 6.24-1 USCI-UART Mode Block Diagram 1218
Figure 6.24-2 UART Signal Connection for Full-Duplex Communication 1219
Figure 6.24-3 UART Standard Frame Format 1220
Figure 6.24-4 UART Bit Timing (Data Sample Time) 1222
Figure 6.24-5 UART Auto Baud Rate Control 1224
Figure 6.24-6 Incoming Data Wake-Up 1225
Figure 6.24-7 nCTS Wake-Up Case 1 1225
Figure 6.24-8 nCTS Wake-Up Case 2 1225
Figure 6.25-1 SPI Master Mode Application Block Diagram 1251
Figure 6.25-2 SPI Slave Mode Application Block Diagram 1251
Figure 6.25-3 USCI SPI Mode Block Diagram 1252
Figure 6.25-4 4-Wire Full-Duplex SPI Communication Signals (Master Mode) 1253
Figure 6.25-5 4-Wire Full-Duplex SPI Communication Signals (Slave Mode) 1254
Figure 6.25-6 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x0) 1255
Figure 6.25-7 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x1) 1255
Figure 6.25-8 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x2) 1256
Figure 6.25-9 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x3) 1256
Figure 6.25-10 16-bit Data Length in One Word Transaction with MSB First Format 1257

Figure 6.25-11 Word Suspend Interval between Two Transaction Words 1257
Figure 6.25-12 Auto Slave Select (SUSPITV ≧ 0x3) 1258
Figure 6.25-13 Auto Slave Select (SUSPITV < 0x3) 1259
Figure 6.25-14 One Output Data Channel Half-duplex (SPI Master Mode) 1260
Figure 6.25-15 One Input Data Channel Half-duplex (SPI Master Mode) 1260
Figure 6.25-16 SPI Timing in Master Mode 1262
Figure 6.25-17 SPI Timing in Master Mode (Alternate Phase of Serial Bus Clock) 1262
Figure 6.25-18 SPI Timing in Slave Mode 1263
Figure 6.25-19 SPI Timing in Slave Mode (Alternate Phase of Serial Bus Clock) 1263
Figure 6.26-1 I²C Bus Timing 1288
Figure 6.26-2 USCI I²C Mode Block Diagram 1289
Figure 6.26-3 I²C Protocol 1290
Figure 6.26-4 START and STOP Conditions 1290
Figure 6.26-5 Bit Transfer on the I²C Bus 1291
Figure 6.26-6 Acknowledge on the I²C Bus 1292
Figure 6.26-7 Arbitration Lost 1293
Figure 6.26-8 Control I²C Bus according to Current I²C Status 1295
Figure 6.26-9 Master Transmits Data to Slave with a 7-bit Address 1295
Figure 6.26-10 Master Reads Data from Slave with a 7-bit Address 1295
Figure 6.26-11 Master Transmits Data to Slave by 10-bit Address 1296
Figure 6.26-12 Master Reads Data from Slave by 10-bit Address 1296
Figure 6.26-13 Master Transmitter Mode Control Flow with 7-bit Address 1297
Figure 6.26-14 Master Receiver Mode Control Flow with 7-bit Address 1298
Figure 6.26-15 Master Transmitter Mode Control Flow with 10-bit Address 1299
Figure 6.26-16 Master Receiver Mode Control Flow with 10-bit Address 1300
Figure 6.26-17 Save Mode Control Flow with 7-bit Address 1301
Figure 6.26-18 Save Mode Control Flow with 10-bit Address 1302
Figure 6.26-19 GC Mode with 7-bit Address 1303
Figure 6.26-20 Setup Time Wrong Adjustment 1305
Figure 6.26-21 Hold Time Wrong Adjustment 1305
Figure 6.26-22 I²C Time-out Count Block Diagram 1306
Figure 6.26-23 EEPROM Random Read 1307
Figure 6.26-24 Protocol of EEPROM Random Read 1307
Figure 6.27-1 UART Clock Control Diagram 1329
Figure 6.27-2 UART Block Diagram 1330
Figure 6.27-3 Auto-Baud Rate Measurement 1336
Figure 6.27-4 Transmit Delay Time Operation 1337

Figure 6.27-5 UART nCTS Wake-up Case1 1337
Figure 6.27-6 UART nCTS Wake-up Case2 1338
Figure 6.27-7 UART Data Wake-up 1338
Figure 6.27-8 UART Received Data FIFO reached threshold wake-up 1339
Figure 6.27-9 UART RS-485 AAD Mode Address Match Wake-up 1339
Figure 6.27-10 UART Received Data FIFO threshold time-out wake-up 1340
Figure 6.27-11 Auto-Flow Control Block Diagram 1343
Figure 6.27-12 UART nCTS Auto-Flow Control Enabled 1344
Figure 6.27-13 UART nRTS Auto-Flow Control Enabled 1344
Figure 6.27-14 UART nRTS Auto-Flow with Software Control 1345
Figure 6.27-15 IrDA Control Block Diagram 1345
Figure 6.27-16 IrDA TX/RX Timing Diagram 1346
Figure 6.27-17 Structure of LIN Frame 1347
Figure 6.27-18 Structure of LIN Byte 1347
Figure 6.27-19 Break Detection in LIN Mode 1349
Figure 6.27-20 LIN Frame ID and Parity Format 1349
Figure 6.27-21 LIN Sync Field Measurement 1351
Figure 6.27-22 UART_BAUD Update Sequence in AR mode if SLVDUEN is 1 1352
Figure 6.27-23 UART_BAUD Update Sequence in AR mode if SLVDUEN is 0 1352
Figure 6.27-24 RS-485 nRTS Driving Level in Auto Direction Mode 1355
Figure 6.27-25 RS-485 nRTS Driving Level with Software Control 1355
Figure 6.27-26 Structure of RS-485 Frame 1356
Figure 6.28-1 LPUART Clock Control Diagram 1398
Figure 6.28-2 LPUART Block Diagram 1398
Figure 6.28-3 Auto-Baud Rate Measurement 1403
Figure 6.28-4 Transmit Delay Time Operation 1404
Figure 6.28-5 LPUART nCTS Wake-up Case1 1404
Figure 6.28-6 LPUART nCTS Wake-up Case2 1405
Figure 6.28-7 LPUART Data Wake-up 1405
Figure 6.28-8 LPUART Received Data FIFO reached threshold wake-up 1406
Figure 6.28-9 LPUART RS-485 AAD Mode Address Match Wake-up 1406
Figure 6.28-10 LPUART Received Data FIFO threshold time-out wake-up 1407
Figure 6.28-11 Auto-Flow Control Block Diagram 1410
Figure 6.28-12 LPUART nCTS Auto-Flow Control Enabled 1410
Figure 6.28-13 LPUART nRTS Auto-Flow Control Enabled 1411
Figure 6.28-14 LPUART nRTS Auto-Flow with Software Control 1411
Figure 6.28-15 RS-485 nRTS Driving Level in Auto Direction Mode 1413

Figure 6.28-16 RS-485 nRTS Driving Level with Software Control 1413
Figure 6.28-17 Structure of RS-485 Frame 1414
Figure 6.28-18 LPUART Auto-operation mode TX state Block Diagram 1415
Figure 6.28-19 LPUART Auto-operation mode RX state Block Diagram 1416
Figure 6.29-1 SPI Block Diagram 1453
Figure 6.29-2 SPI Peripheral Clock 1455
Figure 6.29-3 SPI Full-Duplex Master Mode Application Block Diagram 1455
Figure 6.29-4 SPI Full-Duplex Slave Mode Application Block Diagram 1456
Figure 6.29-5 32-bit in One Transaction 1457
Figure 6.29-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2) 1458
Figure 6.29-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3) 1458
Figure 6.29-8 Byte Reorder Function 1458
Figure 6.29-9 Timing Waveform for Byte Suspend 1459
Figure 6.29-10 SPI Half-Duplex Master Mode Application Block Diagram 1459
Figure 6.29-11 SPI Half-Duplex Slave Mode Application Block Diagram 1459
Figure 6.29-12 FIFO Threshold Comparator 1461
Figure 6.29-13 Transmit FIFO Buffer Example for 4~16 bits of data length 1462
Figure 6.29-14 Transmit FIFO Buffer Example for 17~32 bits of data length 1462
Figure 6.29-15 Receive FIFO Buffer Example for 16 bits of Data Length 1463
Figure 6.29-16 Receive FIFO Buffer Example for 32 bits of Data Length 1464
Figure 6.29-17 TX Underflow Event and Slave Under Run Event 1464
Figure 6.29-18 TX Underflow Event (SPI Slave 3-Wire Mode Enabled) 1465
Figure 6.29-19 Slave Mode Bit Count Error and Effective Bit Number of Uncompleted RX data 1466
Figure 6.29-20 I²S Data Format Timing Diagram 1468
Figure 6.29-21 MSB Justified Data Format Timing Diagram 1468
Figure 6.29-22 PCM Mode A Timing Diagram 1468
Figure 6.29-23 PCM Mode B Timing Diagram 1469
Figure 6.29-24 FIFO Contents for Various I²S Modes 1470
Figure 6.29-25 SPI Timing in Master Mode 1473
Figure 6.29-26 SPI Timing in Master Mode (Alternate Phase of SPIx_CLK) 1473
Figure 6.29-27 SPI Timing in Slave Mode 1474
Figure 6.29-28 SPI Timing in Slave Mode (Alternate Phase of SPIx_CLK) 1474
Figure 6.30-1 LPSPI Block Diagram 1502
Figure 6.30-2 LPSPI Peripheral Clock 1503
Figure 6.30-3 LPSPI Full-Duplex Master Mode Application Block Diagram 1504
Figure 6.30-4 LPSPI Full-Duplex Slave Mode Application Block Diagram 1504
Figure 6.30-5 32-bit in One Transaction 1505

Figure 6.30-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2) 1506
Figure 6.30-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3) 1506
Figure 6.30-8 Byte Reorder Function 1507
Figure 6.30-9 Timing Waveform for Byte Suspend 1507
Figure 6.30-10 SPI Half-Duplex Master Mode Application Block Diagram 1508
Figure 6.30-11 SPI Half-Duplex Slave Mode Application Block Diagram 1508
Figure 6.30-12 FIFO Threshold Comparator 1510
Figure 6.30-13 Transmit FIFO Buffer Example for 4~16 bits of data length 1510
Figure 6.30-14 Transmit FIFO Buffer Example for 17~32 bits of data length 1511
Figure 6.30-15 Receive FIFO Buffer Example for 16 bits of Data Length 1512
Figure 6.30-16 Receive FIFO Buffer Example for 32 bits of Data Length 1512
Figure 6.30-17 TX Underflow Event and Slave Under Run Event 1513
Figure 6.30-18 TX Underflow Event (SPI Slave 3-Wire Mode Enabled) 1513
Figure 6.30-19 Slave Mode Bit Count Error and Effective Bit Number of Uncompleted RX data 1514
Figure 6.30-20 LPSPI Auto-operation Mode Block Diagram 1516
Figure 6.30-21 LPSPI Auto Full-Duplex Master Mode when AUTOSS set to 0 1518
Figure 6.30-22 LPSPI Auto Half-Duplex Master Mode when AUTOSS set to 0 1518
Figure 6.30-23 LPSPI Auto Full-Duplex Master Mode when AUTOSS set to 1 1518
Figure 6.30-24 LPSPI Auto Half-Duplex Master Mode when AUTOSS set to 1 1519
Figure 6.30-25 LPSPI Timing in Master Mode 1519
Figure 6.30-26 LPSPI Timing in Master Mode (Alternate Phase of LPSPI_CLK) 1520
Figure 6.30-27 LPSPI Timing in Slave Mode 1520
Figure 6.30-28 LPSPI Timing in Slave Mode (Alternate Phase of LPSPI_CLK) 1521
Figure 6.31-1 QSPI Block Diagram 1544
Figure 6.31-2 QSPI Peripheral Clock 1545
Figure 6.31-3 QSPI Full-Duplex Master Mode Application Block Diagram 1546
Figure 6.31-4 QSPI Full-Duplex Slave Mode Application Block Diagram 1546
Figure 6.31-5 32-bit in One Transaction 1547
Figure 6.31-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2) 1548
Figure 6.31-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3) 1548
Figure 6.31-8 Byte Reorder Function 1549
Figure 6.31-9 Timing Waveform for Byte Suspend 1549
Figure 6.31-10 QSPI Half-Duplex Master Mode Application Block Diagram 1550
Figure 6.31-11 QSPI Half-Duplex Slave Mode Application Block Diagram 1550
Figure 6.31-12 Two-bit Transfer Mode System Architecture 1552
Figure 6.31-13 Two-bit Transfer Mode Timing (Master Mode) 1552
Figure 6.31-14 Bit Sequence of Dual Output Mode 1553

Figure 6.31-15 Bit Sequence of Dual Input Mode 1553
Figure 6.31-16 Bit Sequence of Quad Output Mode 1554
Figure 6.31-17 Bit Sequence of Quad Input Mode 1555
Figure 6.31-18 Timing Diagram of QSPI Output Data for Transmit Double Transfer Rate Mode 1555
Figure 6.31-19 FIFO Threshold Comparator 1556
Figure 6.31-20 Transmit FIFO Buffer Example 1557
Figure 6.31-21 Receive FIFO Buffer Example 1558
Figure 6.31-22 TX Underflow Event and Slave Under Run Event 1558
Figure 6.31-23 Two-bit Transfer Mode FIFO Buffer Example 1559
Figure 6.31-24 TX Underflow Event (QSPI Slave 3-Wire Mode Enabled) 1559
Figure 6.31-25 Slave Mode Bit Count Error and Effective Bit Number of Uncompleted RX Data 1560
Figure 6.31-26 Slave Time-out Event 1560
Figure 6.31-27 QSPI Timing in Master Mode 1563
Figure 6.31-28 QSPI Timing in Master Mode (Alternate Phase of QSPIx_CLK) 1563
Figure 6.31-29 QSPI Timing in Slave Mode 1564
Figure 6.31-30 QSPI Timing in Slave Mode (Alternate Phase of QSPIx_CLK) 1564
Figure 6.32-1 USB Block Diagram 1584
Figure 6.32-2 NEVWK Interrupt Operation Flow 1586
Figure 6.32-3 Endpoint SRAM Structure 1587
Figure 6.32-4 Setup Transaction Followed by Data IN Transaction 1587
Figure 6.32-5 Data Out Transfer 1588
Figure 6.33-1 USB 1.1 Host Controller Block Diagram 1623
Figure 6.34-1 USB OTG Block Diagram 1658
Figure 6.34-2 USB Device Mode 1660
Figure 6.34-3 USB Host Mode 1660
Figure 6.35-1 UTCPD System Block Diagram 1675
Figure 6.35-2 V_{BUS} Detection Block Diagram (Implement in design) 1677
Figure 6.35-3 UTCPD Interrupt 1679
Figure 6.35-4 CC State Change Interrupt 1679
Figure 6.35-5 Power Status Change Interrupt 1680
Figure 6.35-6 Fault Status Interrupt 1680
Figure 6.35-7 Vendor Status Interrupt 1680
Figure 6.35-8 Discharge Time Diagram 1682
Figure 6.36-1 CAN FD Controller Block Diagram 1765
Figure 6.36-2 Transceiver Delay Measurement 1769
Figure 6.36-3 Pin Controller in Bus Monitor Mode 1770

Figure 6.36-4 Pin Control in Loop Back Modes 1772
Figure 6.36-5 Standard Message ID Filter Path 1775
Figure 6.36-6 Extended Message ID Filter Path 1776
Figure 6.36-7 Rx FIFO Status 1777
Figure 6.36-8 Rx FIFO Overflow Handling 1778
Figure 6.36-9 Example of Mixed Configuration Dedicated Tx Buffers / Tx FIFO 1782
Figure 6.36-10 Example of mixed Configuration Dedicated Tx Buffers / Tx Queue 1782
Figure 6.36-11 Message RAM configuration 1784
Figure 6.36-12 Bit Timing 1792
Figure 6.37-1 Key Store Block Diagram 1848
Figure 6.38-1 Cryptographic Accelerator Block Diagram 1860
Figure 6.38-2 PRNG Function Diagram 1862
Figure 6.38-3 Electronic Codebook Mode 1864
Figure 6.38-4 Cipher Block Chaining Mode 1865
Figure 6.38-5 Cipher Feedback Mode 1866
Figure 6.38-6 Output Feedback Mode 1867
Figure 6.38-7 Counter Mode 1868
Figure 6.38-8 CBC-CS1 Encryption 1868
Figure 6.38-9 CBC-CS1 Decryption 1869
Figure 6.39-1 Touch Key Block Diagram 1892
Figure 6.39-2 Touch Key Sensing Method 1894
Figure 6.39-3 Finger Touch Detection Method 1895
Figure 6.39-4 Touch Key Controller Interrupt Modes for Threshold Control 1896
Figure 6.40-1 CRC Generator Block Diagram 1966
Figure 6.40-2 CHECKSUM Bit Order Reverse Functional Block 1967
Figure 6.40-3 Write Data Bit Order Reverse Functional Block 1968
Figure 6.41-1 Analog Comparator 0/1 Block Diagram 1983
Figure 6.41-2 Analog Comparator 2 Block Diagram 1983
Figure 6.41-3 Comparator Hysteresis Function 1984
Figure 6.41-4 Window Latch Mode 1985
Figure 6.41-5 Example of Filter Function 1985
Figure 6.41-6 Comparator Controller Interrupt 1986
Figure 6.41-7 Comparator Controller Wakeup 1986
Figure 6.41-8 Comparator0 Reference Voltage Block Diagram 1987
Figure 6.41-9 Example of Window Compare Mode 1988
Figure 6.41-10 Example of Window Compare Mode 1989
Figure 6.42-1 EADC Converter Block Diagram 2010

Figure 6.42-2 Sample Module 0~3 Block Diagram 2011
Figure 6.42-3 Sample Module 4~26 Block Diagram 2012
Figure 6.42-4 Sample Module 27~30 Block Diagram 2012
Figure 6.42-5 EADC Clock Control 2013
Figure 6.42-6 Example EADC Conversion Timing Diagram, n=0~30 2014
Figure 6.42-7 Sample Module Conversion Priority Arbitrator Diagram 2015
Figure 6.42-8 Specific Sample Module EADC EOC Signal for ADINT0~3 Interrupt 2016
Figure 6.42-9 EPWM-triggered EADC Start Conversion 2017
Figure 6.42-10 External triggered EADC Start Conversion 2017
Figure 6.42-11 Conversion Start Delay Timing Diagram 2018
Figure 6.42-12 EADC0_ST De-bounce Timing Diagram 2018
Figure 6.42-13 EADC Extend Sampling Timing Diagram 2019
Figure 6.42-14 EADC Conversion Result Monitor Logics Diagram 2021
Figure 6.42-15 Input of Fully-Differential EADC 2021
Figure 6.42-16 EADC Controller Interrupts 2023
Figure 6.42-17 Calibration Mode with 16 Times Average 2023
Figure 6.43-1 LPADC Controller Block Diagram 2077
Figure 6.43-2 LPADC Peripheral Clock Control 2078
Figure 6.43-3 Single Mode Conversion Timing Diagram 2079
Figure 6.43-4 Single-Cycle Scan Mode on Enabled Channels Timing Diagram 2080
Figure 6.43-5 Continuous Scan Mode on Enabled Channels Timing Diagram 2081
Figure 6.43-6 Calibration Mode 2082
Figure 6.43-7 A/D Conversion Result Monitor Logic Diagram 2084
Figure 6.43-8 LPADC Interrupt Diagram 2085
Figure 6.43-9 Input of Fully-Differential LPADC 2086
Figure 6.43-10 LPADC Auto-operation Mode Block Diagram 2087
Figure 6.43-11 Clock Enable/Disable Control for Data Collector 2088
Figure 6.43-12 Clock Enable/Disable Control for Threshold Detector 2088
Figure 6.43-13 LPADC Wake-up Diagram 2089
Figure 6.43-14 Conversion Result Mapping Diagram of ADC Single-end Input 2094
Figure 6.43-15 Conversion Result Mapping Diagram of ADC Differential Input 2094
Figure 6.44-1 Digital-to-Analog Converter Block Diagram 2117
Figure 6.44-2 Data Holding Register Format 2118
Figure 6.44-3 DAC Conversion Started by Software Write Trigger 2119
Figure 6.44-4 DAC Conversion Started by Hardware Trigger Event 2119
Figure 6.44-5 DAC0 and DAC1 Group and Ungroup Update Example 2120
Figure 6.44-6 DAC PDMA Under-Run Condition Example 2121

Figure 6.44-7 DAC Continuous Conversion with Software PDMA Mode 2121
Figure 6.44-8 DAC Interrupt Source 2122
Figure 6.45-1 OP Amplifier Block Diagram 2140
Figure 6.45-2 OP Amplifier Interrupt and Wake-up..... 2142

LIST OF TABLE

Table 2.1-1 NuMicro M2L31 Series Key Features Support Table 36
Table 4.3-1 M2L31ZE4AE Multi-function Pin Table 59
Table 4.3-2 M2L31ZD4AE Multi-function Pin Table 62
Table 4.3-3 M2L31YG4AE Multi-function Pin Table 65
Table 4.3-4 M2L31YE4AE Multi-function Pin Table 68
Table 4.3-5 M2L31YD4AE Multi-function Pin Table 71
Table 4.3-6 M2L31YIDAE Multi-function Pin Table 75
Table 4.3-7 M2L31YGDAE Multi-function Pin Table 79
Table 4.3-8 M2L31LE4AE Multi-function Pin Table 82
Table 4.3-9 M2L31LG4AE Multi-function Pin Table 85
Table 4.3-10 M2L31LD4AE Multi-function Pin Table 88
Table 4.3-11 M2L31LIDAE Multi-function Pin Table 92
Table 4.3-12 M2L31LGDAE Multi-function Pin Table 96
Table 4.3-13 M2L31SG4AE Multi-function Pin Table 100
Table 4.3-14 M2L31SE4AE Multi-function Pin Table 104
Table 4.3-15 M2L31SIDAE Multi-function Pin Table 108
Table 4.3-16 M2L31SGDAE Multi-function Pin Table 112
Table 4.3-17 M2L31KIDAE Multi-function Pin Table 118
Table 4.3-18 M2L31KGDAE Multi-function Pin Table 124
Table 6.2-1 Reset Value of Registers 146
Table 6.2-2 Power Mode Table 151
Table 6.2-3 AHB/APB Bus maximum speed under different Power Modes 152
Table 6.2-4 Power Mode Difference Table 152
Table 6.2-5 Power Mode Definition Table 153
Table 6.2-6 Clocks in Power Modes 154
Table 6.2-7 Re-Entering Power-down Mode Condition 158
Table 6.2-8 Function relationship with Cold-boot and warm-boot 159
Table 6.2-9 Address Space Assignments for On-Chip Controllers 163
Table 6.2-10 SRAM Power Mode Behavior 167
Table 6.2-11 Protected Registers 173
Table 6.2-12 Exception Model 301
Table 6.2-13 Interrupt Number Table 305
Table 6.2-14 Priority Grouping 347
Table 6.3-1 Symbol definition of PLL Output Frequency formula 388
Table 6.5-1 RMC Features Comparison Table at Different Chip 440
Table 6.5-2 Dual-bank Block Address Range 443

Table 6.5-3 Single-bank Block Address Range 443
Table 6.5-4 ISP Command List 470
Table 6.5-5 APROM Write Protect Level 476
Table 6.5-6 Lock Effect Table with Two Protections for CPU 484
Table 6.6-1 De-Bounce Function Setting Table 513
Table 6.8-1 Channel Priority Table 551
Table 6.9-1 Channel Priority Table 590
Table 6.10-1 EBI Address Mapping 619
Table 6.10-2 Timing Control Parameter 623
Table 6.11-1 PWM Output Level 651
Table 6.12-1 PWM Output Level 715
Table 6.14-1 PWM Clock Source Control Registers Setting Table 773
Table 6.14-2 PWM Pulse Generation Event Priority for Up-Counter 785
Table 6.14-3 PWM Pulse Generation Event Priority for Down-Counter 786
Table 6.14-4 PWM Pulse Generation Event Priority for Up-Down-Counter 786
Table 6.15-1 EPWM Clock Source Control Registers Setting Table 872
Table 6.15-2 EPWM Pulse Generation Event Priority for Up-Counter 887
Table 6.15-3 EPWM Pulse Generation Event Priority for Down-Counter 887
Table 6.15-4 EPWM Pulse Generation Event Priority for Up-Down-Counter 887
Table 6.16-1 Direction of Counting 1014
Table 6.16-2 Swap Function Table 1019
Table 6.17-1 Typical Case of Noise Filter Settings 1041
Table 6.18-1 User Configuration Setting Table 1058
Table 6.18-2 Watchdog Timer Time-out Interval Period Selection 1059
Table 6.19-1 WWDT Prescale Value Selection 1067
Table 6.19-2 CMPDAT Setting Limitation 1070
Table 6.20-1 RTC Read/Write Enable 1080
Table 6.20-212/24 Hour Time Scale Selection 1081
Table 6.20-3 Register Value After Powered On 1083
Table 6.21-1 Reserved SMBus Address 1130
Table 6.21-2 Relationship between I^2C Baud Rate and PCLK 1139
Table 6.21-3 I^2C Status Code Description 1141
Table 6.22-1 Relationship between LPI2C Baud Rate and PCLK 1181
Table 6.22-2 LPI2C Status Code Description 1184
Table 6.23-1 Input Signals for Different Protocols 1208
Table 6.23-2 Output Signals for Different Protocols 1209
Table 6.23-3 Data Transfer Events and Interrupt Handling 1216

Table 6.23-4 Protocol-specific Events and Interrupt Handling 1216
Table 6.24-1 Input Signals for UART Protocol 1220
Table 6.24-2 Output Signals for Different Protocol 1220
Table 6.24-3 Baud Rate Relationship 1223
Table 6.25-1 Serial Bus Clock Configuration 1254
Table 6.26-1 Relationship between I²C Baud Rate and PCLK 1305
Table 6.27-1 M2L31 Series UART Features 1328
Table 6.27-2 UART Interrupt 1331
Table 6.27-3 UART Interface Controller Pin 1333
Table 6.27-4 UART controller Baud Rate Equation Table 1333
Table 6.27-5 UART controller Baud Rate Parameter Setting Example Table 1334
Table 6.27-6 UART controller Baud Rate Register Setting Example Table 1334
Table 6.27-7 Baud Rate Compensation Example Table 1 1335
Table 6.27-8 Baud Rate Compensation Example Table 2 1335
Table 6.27-9 UART controller Interrupt Source and Flag List 1342
Table 6.27-10 UART Line Control of Word and Stop Length Setting 1342
Table 6.27-11 UART Line Control of PARITY bit Setting 1343
Table 6.27-12 LIN Header Selection in Master Mode 1348
Table 6.28-1 M2L31 Series LPUART Features 1397
Table 6.28-2 LPUART Interrupt 1399
Table 6.28-3 LPUART Interface Controller Pin 1400
Table 6.28-4 LPUART controller Baud Rate Equation Table 1400
Table 6.28-5 LPUART controller Baud Rate Parameter Setting Example Table 1401
Table 6.28-6 LPUART controller Baud Rate Register Setting Example Table 1401
Table 6.28-7 Baud Rate Compensation Example Table 1 1402
Table 6.28-8 Baud Rate Compensation Example Table 2 1402
Table 6.28-9 LPUART controller Interrupt Source and Flag List 1408
Table 6.28-10 LPUART Line Control of Word and Stop Length Setting 1409
Table 6.28-11 LPUART Line Control of PARITY Bit Setting 1409
Table 6.29-1 SPI/I²S Interface Controller Pin Description (SPI0~SPI3) 1454
Table 6.29-2 Dummy Data Number for I²S / PCM Master Mode and Monaural Mode 1471
Table 6.29-3 Dummy Data Number for I²S / PCM Master Mode and Stereo Mode 1471
Table 6.29-4 Dummy Data Number for I²S Slave Mode and Monaural Mode 1471
Table 6.29-5 Dummy Data Number for PCM Slave Mode and Monaural Mode 1472
Table 6.29-6 Dummy Data Number for I²S Slave Mode and Stereo Mode 1472
Table 6.29-7 Dummy Data Number for PCM Slave Mode and Stereo Mode 1472
Table 6.30-1 LPSPI Interface Controller Pin Description (LPSPI0) 1503

Table 6.35-1 Abbreviation 1677
Table 6.35-2 Reset Conditions 1678
Table 6.36-1 DLC coding in CAN FD format 1768
Table 6.36-2 Rx Buffer / FIFO Element Size 1778
Table 6.36-3 Example Filter Configuration for Rx Buffers 1779
Table 6.36-4 Possible Configuration for Frame Transmission 1780
Table 6.36-5 Tx Buffer / FIFO / Queue Element Size 1781
Table 6.36-6 Rx Buffer and FIFO Element 1785
Table 6.36-7 Rx Buffer and FIFO Element Description 1787
Table 6.36-8 Tx Buffer and FIFO Element 1787
Table 6.36-9 Tx Buffer Element Description 1788
Table 6.36-10 Tx Event FIFO Element 1789
Table 6.36-11 Tx Event FIFO Element Description 1789
Table 6.36-12 Standard Message ID Filter Element 1790
Table 6.36-13 Standard Message ID Filter Element Description 1791
Table 6.36-14 Extended Message ID Filter Element 1791
Table 6.36-15 Extended Message ID Filter Element Description 1792
Table 6.37-1 Detailed Information for OTP 1850
Table 6.37-2 Information of OTP Key Storage 1850
Table 6.37-3 CPU and Crypto Engine Access Key for All Situations 1852
Table 6.38-1 Cryptographic Features Comparison Table at Different Chip 1860
Table 6.38-2 Each Engine Error Conditions and Error Flag 1861
Table 6.38-3 DMA Enable Bit Table 1861
Table 6.38-4 DMA Cascade Bit Table 1862
Table 6.41-1 Wake-up Support Table 1987
Table 6.41-2 Truth Table of Window Compare Logic 1988
Table 6.42-1 Setting of Accumulation and Conversion Result Precision 2020
Table 6.42-2 Setting of Averaging and Conversion Result Precision 2020
Table 6.42-3 EADC Differential Model Channel Selection 2022
Table 6.43-1 LPADC Differential Model Channel Selection 2086
Table 6.46-1 Peripherals Interconnect Matrix Table 2158
Table 9.1-1 List of Abbreviations 2170

1 GENERAL DESCRIPTION

The NuMicro® M2L31 series is based on Arm Cortex-M23 core at Armv8-M architecture with a single-cycle hardware multiplier/divider. It runs up to 72 MHz and features 64 to 512 Kbytes ReRAM, 40 to 168 Kbytes SRAM, 1.71V to 3.6V operating voltage, -40°C to 105°C wide operating temperature, a variety of packages choices, and excellent high immunity characteristics by 4 KV ESD HBM and 4.4 KV EFT. The dual bank design of 512 Kbytes ReRAM supports firmware update through the Over-The-Air (FOTA) process. The M2L31 series supports Execute-Only Memory (XOM) function to protect confidential program code information from being stolen in the run-time.

ReRAM (Resistive Random-Access Memory)

A type of non-volatile memory achieves digital data storage by altering the resistance state of its components through the application of an external voltage. It boasts three major characteristics: fast read/write speeds, low power consumption, and superior durability, making it hailed as the next-generation embedded universal memory. Unlike embedded Flash memory, ReRAM doesn't require a page erase operation before writing, resulting in faster write speeds and a more straightforward and speedy operation similar to EEPROM. Furthermore, ReRAM consumes less energy for storing each bit compared to what is needed in Flash memory. Additionally, because each storage unit can be individually set or reset, it offers greater endurance than Flash memory.

Enhanced Analog Features

The M2L31 series provides up to three Programmable Gain Amplifiers (PGA) to amplify the small voltage changed by amplification factors of 1, 2, 4, 8, 16, and 32, three comparators in conjunction with PWM to ensure cycle-by-cycle over current limiting for increased application safety and robustness, one 24-ch high-speed 3.6 MSPS 12-bit SAR ADC, up to two 12-bit 1 MSPS DAC to provide precise voltage reference to other analog device, and up to 24-ch 16-bit PWM for precise modulation and optimized for controlling half bridge or H-bridge motor drive circuits. It is eligible for motor control, industrial automation, home appliance.

USB 2.0 Type-C Power Delivery 3.0 Controller

The M2L31 series includes one controller compliant with USB Type-C Rev. 2.1 and USB Power Delivery Rev. 3.0 specifications. The controller featuring USB Type-C pull-up and pull-down resistors, Dead battery and fast role swap supported, and flexible Power Delivery along with data transmission over a single cable.

Low-Power Technology

The M2L31 series provides low power consumption in Normal Run mode 60 μA/MHz at 72 MHz, 33 μA/MHz in idle run mode. It supports 6 core power-down levels, including Fast Wake-up Power-down, Normal Power-down, Low Leakage Power-down, Ultra Low Leakage Power-down, Standby Power-down, and Deep Power-down mode. Beside LIRC/LXT can be operated in Low Leakage Power-down mode, the MIRC can also work for operation of high precision sensors by shorten stable time of clock; in Ultra Low Leakage Power-down mode the M2L31 series provides Auto-operation mode to process the sensor data through low power serial interface with low power DMA and 8KB standalone low power SRAM without interrupting CPU. The current consumption of Standby Power-down modes with RAM retention is 3.2uA, and Deep Power-down mode is less than 0.5 μA.

Enhanced Security Levels

The M2L31 series supports eXecute-Only-Memory (XOM) to protect the intelligent property of developers. It is also equipped with rich functions to improve system security. The Secure Bootloader supports system-boot feature to protect cerificated firmware from being replaced with malware in the upgrade processing. The hardware crypto accelerators, including AES, PRNG, and TRNG, support encryption and decryption operations to offload the main processor's computing power and ensure data transmission in secure. Beside the configureration words and register setting for write protection, the M2L31 series is also equipped with one hardware PIN Lock function to conduct dual function protect protection. One of the major challenges for IoT devices that are connected to cloud services is security, so the IoT devices must meet some security requirements to secure assets from being stolen or modified by an attacker.

Rich peripherals for comprehensive product application scenarios

The M2L31 series provides plenty of peripherals such as Timers, Watchdog Timers, RTC with independent V_{BAT} pin, up to 16 channels of PDMA, up to 16 independent capacitive touch key sensing function with single-scan or programmable periodic key-scans modes, up to 2 sets of CAN FD controller with internal 12Kbytes SRAM, USB 2.0 FS OTG Host and device controllers, up to 8 sets of UART, up to 4 sets of I2C, up to 4 sets of SPI/I2S,1 set of QSPI, and 2 sets of Universal Serial Control Interfaces (USCI); moreover, it supports enhanced control interfaces such as Voltage adjustable interface (VAI), External Bus Interface (EBI), eCAP and QEI interface to detect the motor spinning position.

Supported packages include WLCSP 25 (2.5 mm x 2.5 mm), QFN32 (5 mm x 5 mm), LQFP48 (7 mm x 7 mm), QFN 48 (5 mm x 5 mm), WLCSP 49 (3.0 mm x 3.0 mm), LQFP64 (7 mm x 7 mm), and LQFP128 (14 mm x 14 mm).

For the development, Nuvoton provides the NuMaker evaluation board and Nuvoton Nu-Link debugger. The 3rd Party IDE such as Keil MDK, IAR EWARM, Eclipse IDE with GNU GCC compilers are also supported.

USCI*: supports UART, SPI or I^2C

Product Line	UART	I^2C	SPI/I^2S	USCI*	USB OTG	CAN FD	Timer	PDMA	PGA	ADC	DAC	ACMP	PWM
M2L31	8	4	4	2	1	2	4	16	3	24	2	3	24

Table 2.1-1 NuMicro M2L31 Series Key Features Support Table

The M2L31 series is suitable for a wide range of applications such as:

- Motor Control
- PC Peripherals
- Industrial Automatic Related Products
- Battery Management System

2 FEATURES

2.1 M2L31 Series Features

Core and System	
Arm Cortex-M23 without TrustZone	• Arm Cortex-M23 core, running up to 72 MHz when V_{DD} = 1.71V ~ 3.6V • Built-in PMSAv8 Memory Protection Unit (MPU) • Built-in Nested Vectored Interrupt Controller (NVIC) • 32-bit Single-cycle hardware multiplier and 32-bit 17-cycle hardware divider • 24-bit system tick timer • Supports Programmable and maskable interrupt • Supports Low Power Sleep mode by WFI and WFE instructions • Supports single cycle I/O access • Supports XOM feature with 4 regions
Low power mode and current	• Low Power mode: – Idle mode • Power-down mode: – Normal Power-down mode (NPD) – Standby Power-down mode (SPD) – Deep Power-down mode (DPD)
Wake-up source and wakeup time	• EINT, USCI, RTC, WDT, I^2C, Timer, UART, BOD, LVR, POR, GPIO, USBD, USBH, USBOTP, ACMP, LPTMR, TTMR, OPA, LPUART, LPI^2C, LPSPI, LPPDMA, LPADC, TouchKey, UPTPD, Debug interface, WKTMR, WKIO, Reset pin from NPD0~NPD2 • RTC, WDT, BOD, LVR, POR, ACMP, LPTMR, TTMR, OPA, LPUART, LPI^2C, LPSPI, LPPDMA, LPADC, Debug interface, WKTMR, WKIO, Reset pin from NPD3~NPD4 • RTC, BOD, LVR, POR, ACMP, WKTMR, WKIO, Reset pin from NPD5, SPD0~SPD2 • RTC, WKTMR, POR, Wake-up pins, Tamper from DPD0~DPD1
Power supply and low voltage detect	• Built-in LDO for wide operating voltage from 1.71V to 3.6V • Core power voltage: 1.1V • Brown-out detector – With 7 levels: 3.0V/2.8V/2.6V/2.4V/2.2V/2.0V/1.8V/1.6V – Supports Brown-out Interrupt and Reset option • Low Voltage Reset (LVR) with 1.5V threshold voltage level

Cyclic Redundancy Calculation Unit	• Supports four common polynomials CRC-CCITT, CRC-8, CRC-16, and CRC-32 • Programmable order reverse setting for input data and CRC checksum • Programmable 1's complement setting for input data and CRC checksum • Supports 8-/16-/32-bit of data width • Programmable seed value • 8-bit write mode: 1-AHB clock cycle operation • 16-bit write mode: 2-AHB clock cycle operation • 32-bit write mode: 4-AHB clock cycle operation • Supports using PDMA to write data to perform CRC operation
Voltage Adjustable Interface	• Supports up to 6 VAI pins • User Configurable 1.71V ~ 3.6V I/O Interface with a dedicated power input (V_{DDIO}) • Supports CAN FD, UART, QSPI, SPI, I^2C, LPUART, LPSPI and LPI2C interface
Security	• 96-bit Unique ID (UID) • 128-bit Unique Customer ID (UCID)
Memories	
Boot Loader	• Factory pre-loaded mask ROM for secure boot procedure and ISP procedure • Uses SHA-256 and ECDSA-P256 to validate data in APROM and LDROM • NuMicro ISP Programming Tool for firmware upgrade via UART, CAN FD, SPI, I^2C and high speed USB device
ReRAM	• Up to 512 KB on-chip Application ROM (APROM) • Dual bank 512 Kbytes APROM for Over-The-Air (OTA) upgrade • Four eXecute-Only-Memory (XOM) regions for code protection • Embedded with 4 Kbytes cache, with performance at zero wait cycle in continuous address read access • 8 Kbytes on-chip user-defined loader (LDROM) • 3 Kbytes One Time Programmable (OTP) ROM for data security • 32-bit ReRAM programming function. • Supports In-System-Programming (ISP), In-Application-Programming (IAP) update embedded ReRAM memory • Supports CRC-32 checksum calculation function

	• Hardware external read protection of whole ReRAM memory by Security Lock Bit • 2-wired ICP ReRAM updating through SWD interface
SRAM	• Up to 168 KB embedded SRAM – 40 Kbytes SRAM located in bank 0 with programmable size of SPD retention The total 40 Kbytes supports hardware parity check; Exception (NMI) generated upon a parity check error – 128 Kbytes SRAM located in bank 1 with SPD retention • Supports byte-, half-word- and word-access • Supports PDMA mode
Peripheral DMA (PDMA)	• Up to 16 independent configurable channels for automatic data transfer between memories and peripherals • Channel 0, 1 support time-out function • Basic and Scatter-Gather Transfer modes • Each channel supports circular buffer management using Scatter-Gather Transfer mode • Two types of priorities modes: Fixed-priority and Round-robin modes • Transfer data width of 8, 16, and 32 bits • Single and burst transfer type • Source and destination address can be increment or fixed • PDMA transfer count up to 65536 • Request source form software, SPI/I^2S, UART, USCI, EADC, DAC, PWM capture event or TIMER
Clocks	
External Clock Source	• Built-in 4~32 MHz external high speed crystal oscillator (HXT) for precise timing operation • Built-in 32.768 kHz external low speed crystal oscillator (LXT) for RTC function and low-power system operation • Supports clock failure detection for high/low speed external crystal oscillator • HXT clock frequency accuracy detector
Internal Clock Source	• Built-in 12 MHz internal high speed RC oscillator (HIRC) for system operation • Built-in 48 MHz internal high speed RC oscillator (HIRC48M) for system operation • Built-in 1/2/4/8 MHz internal high speed RC oscillator (MIRC) for system operation • Built-in 32 kHz internal low speed RC oscillator (LIRC) for

	Watchdog Timer and wake-up operation. • Supports one PLL up to 144 MHz for high performance system operation, sourced from HIRC and HXT
Timers	
32-bit Timer	**TIMER mode** • 4 sets of 32-bit timers with 24-bit up counters and 8-bit prescale counters • Independent clock source for each timer • One-shot, Periodic, Toggle and Continuous Counting operation modes • Event counting function to count the event from external pin • Input capture function to capture or reset counter value • External capture pin event for interval measurement • External capture pin event to reset 24-bit up counter • Chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated • Timer interrupt flag or external capture interrupt flag to trigger EPWM, PWM, EADC, LPADC, DAC and PDMA • Internal capture triggered source from ACMP output • Inter-Timer trigger capture mode **PWM mode** • 16-bit compare register and period register • Double buffer for period register and compare register • Supports inverse in PWM output • PWM interrupt wake-up from system Power-down mode
Enhanced PWM (EPWM)	• Twelve 16-bit counters with 12-bit clock prescale for twelve 144 MHz PWM output channels • Up to 12 independent input capture channels with 16-bit resolution counter • Supports dead time with maximum divided 12-bit prescale • Up, down or up-down PWM counter type • Supports complementary mode for 3 complementary paired PWM output channels • Synchronous function for phase control • Counter synchronous start function • Brake function with auto recovery mechanism • Mask function and tri-state output for each PWM channel • Trigger EADC or DAC to start conversion immediately

	• Trigger EADC to start conversion after a short delay • Supports External Pin Trigger function
PWM	• Supports maximum clock frequency up to 72 MHz • Up to two PWM modules; each module provides 6 output channels. • Supports independent mode for PWM output/Capture input channel • Supports complementary mode for 3 complementary paired PWM output channels: – Dead-time insertion with 12-bit resolution • Two compared values during one period • Supports 12-bit prescaler from 1 to 4096 • Supports 16-bit resolution PWM counter: – Up, down or up/down counter operation type • Supports mask function and tri-state enable for each PWM pin • Supports brake function: – Brake source from pin and system safety events (clock failed, Brown-out detection and CPU lockup) – Noise filter for brake source from pin – Edge detect brake source to control brake state until brake interrupt cleared – Level detect brake source to auto recover function after brake condition removed • Supports interrupt on the following events: • PWM counter match 0, period value or compared value • Brake condition happened • Supports trigger ADC on the following events: • PWM counter match 0, period value or compared value • Capture Function Features: – Up to 12 capture input channels with 16-bit resolution – Supports rising or falling capture condition – Supports input rising/falling capture interrupt – Supports rising/falling capture with counter reload option • Supports PDMA transfer function for all PWM channels
Watchdog	• 20-bit free running up counter for WDT time-out interval • Clock sources from LIRC (default), HCLK/2048 or LXT • 9 selectable time-out periods

	• Able to wake up from Power-down or Idle mode • Interrupt or reset selectable on watchdog time-out • Selectable WDT reset delay period, including 1026, 130, 18 or 3 WDT_CLK reset delay period • Force WDT enabled after chip power on or reset • WDT time-out wake-up function only if WDT clock source is selected as LIRC or LXT
Window Watchdog	• Clock sources from HCLK/2048 (default) or LIRC • Window set by 6-bit down counter with 11-bit prescaler • WWDT counter suspends in Idle/Power-down mode • Supports Interrupt
RTC	• Supports external power pin V_{BAT} • Software compensation by setting frequency compensate register (FCR),compensated clock accuracy reaches ±5ppm within 5 seconds • RTC counter (second, minute, hour) and calendar counter (day, month, year) • Alarm registers (second, minute, hour, day, month, year) • Selectable 12-hour or 24-hour mode • Automatic leap year recognition • Day of the Week counter • Daylight Saving Time software control • Periodic time tick interrupt with 8 period options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 or 1 second • 1 Hz clock output for RTC calibration • Wake-up from idle mode and Power-down mode • 32 kHz oscillator gain control • RTC Time Tick and Alarm Match interrupt
Tamper	• 20 bytes spare registers and 3 tamper pins to clear the content of these spare registers • Selectable spare register erase function • Supports Timestamp function
Analog Interfaces	
Enhanced Analog-to-Digital Converter (EADC)	• Up to one set of 12-bit, 31-ch 3.42 MSPS SAR EADC with up to 24 single-ended input channels or 12 differential input pairs; 10-bit accuracy is guaranteed • Seven internal channels for V_{BAT}, AV_{DD}, OPA0/1/2 output, band-

	gap VBG input and Temperature sensor input • Supports external V_{REF} pin or internal reference voltage • Supports Power-down mode • Supports calibration capability • Conversion can be triggered by software, external pin, Timer 0~3 overflow pulse or EPWM/PWM • Configurable EADC sampling time • Up to 31 sample modules • Double data buffers for sample module 0~3 • PDMA operation • Supports Averageing mode and Oversampling mode, where Oversampling mode provides up to 16-bit precision • Supports early interrupt with delay counter feature
Digital-to-Analog Converter (DAC)	• Up to one 12-bit 1 MSPS voltage type DAC • Analog output voltage: 0~V_{REF} (AV_{DD}) • Supports 8-bit and 12-bit mode • Rail to rail settle time 6us • Reference voltage selects from internal reference voltage, AV_{DD} or V_{REF} pin • Max. output voltage AV_{DD} -0.2V in buffer mode • Conversion started by software enable, Timer interrupt flag(TIF) or PDMA trigger • Voltage output buffer mode and bypass voltage output buffer mode • Supports PDMA mode
Analog Comparator (ACMP)	• Up to three rail-to-rail analog comparators • 4 multiplexed I/O pins at positive node • Negative node: – One I/O pin – Band-gap (VBG) – DAC0 output – Comparator Reference Voltage (CRV) • Programmable propagation speed and low power consumption • Interrupts generated when compare results change (Interrupt event condition programmable) • Supports Power-down Wake-up • Supports triggers for break events and cycle-by-cycle control for

	PWM • Supports window compare mode and window latch mode • Supports programmable hysteresis window: – 0 mV, 20 mV or 40 mV • Supports offset calibration
OPA	• Analog input voltage: 0~AV_{DD} • Up to three operational amplifier • Supports to use schmitt trigger buffer output for simple comparator function • Supports schmitt trigger buffer output interrupts
Internal Reference Voltage	• Internal reference voltage select: 1.6V, 2.0V, 2.5V, 3.0V for EADC, DAC and CRV (comparator reference voltage) reference voltage
Communication Interfaces	
UART	• Supports up to 8 UARTs: UART0~7 • UART baud rate clock from LXT(32.768 kHz) with 9600bps in Power-down mode • Baud rate up to 10 Mbps • Full-duplex asynchronous communications • Supports one-wire half-duplex communications • Separates receive and transmit 16/16 bytes FIFO • Programmable receiver buffer trigger level • Hardware auto-flow control (CTS and RTS) • IrDA (SIR) function: – Supports 3/16 bit duration for normal mode • RS-485 9-bit mode and direction control • UART0/1 supports LIN function: – LIN master/slave mode – Programmable break generation function for transmitter – Break detection function for receiver • Programmable baud-rate generator up to 1/16 system clock • 8-bit receiver FIFO time-out detection function • Programmable transmitting data delay time between the last stop and the next start bit • Auto-Baud Rate measurement and baud rate compensation function • Break error, frame error, parity error and receive/transmit FIFO

	overflow detection function • Supports RS-485 mode: – RS-485 9-bit mode – Hardware or software enables to program nRTS pin to control RS-485 transmission direction – nCTS, incoming data, Received Data FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function in Power-down mode. – Hardware or software enables to program nRTS pin to control RS-485 transmission direction • Fully programmable serial-interface: – Programmable number of data bit, 5-, 6-, 7-, 8- bit character – Programmable parity bit, even, odd, no parity or stick parity bit generation and detection – Programmable stop bit, 1, 1.5, or 2 stop bit generation • Supports PDMA mode
Quad SPI	• Supports one set of Quad SPI with Master/Slave mode • Master mode up to 36 MHz (V_{DD} = 2.7V ~ 3.6V) • Slave mode up to 18 MHz (V_{DD} = 2.7V ~ 3.6V) • Supports Dual and Quad I/O Transfer mode • Supports one data channel half-duplex transfer • Supports double data rate mode (Master TX DIR Mode Only) • Supports receive-only mode • Configurable bit length of a transfer word from 8 to 32-bit • Provides separate 8-level depth transmit and receive FIFO buffers • Supports MSB first or LSB first transfer sequence • Supports the byte reorder function • Supports Byte or Word Suspend mode • Supports 3-wired, no slave select signal, bi-direction interface • PDMA operation
I^2C	• Up to 4 sets of I^2C devices • Master/Slave mode • Bidirectional data transfer between masters and slaves • Multi-master bus (no central master) • 7-bit and 10-bit addressing mode • Standard mode (100 kbps), Fast mode (400 kbps) and Fast mode plus (1 Mbps)

	• Arbitration between simultaneously transmitting masters without corruption of serial data on the bus • Serial clock synchronization allows devices with different bit rates to communicate via one serial bus • Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer • Supports 14-bit time-out counter requesting the I^2C interrupt if the I^2C bus hangs up and timer-out counter overflows • Programmable clocks allow versatile rate control • Multiple address recognition (four slave address with mask option) • Supports setup/hold time programmable • Supports SMBus and PMBus • Multi-address Power-down wake-up function • Supports PDMA transfer
SPI/I^2S	• Up to four sets of SPI/I2S controllers with Master/Slave mode • Provides separate 4-level of 32-bit (or 8-level of 16-bit) transmit and receive FIFO buffers **SPI** • Master mode up to 36 MHz (V_{DD} = 2.7V ~ 3.6V) • Slave mode up to 18 MHz (V_{DD} = 2.7V ~ 3.6V) • Configurable bit length of a transfer word from 4 to 32-bit • MSB first or LSB first transfer sequence • Byte reorder function • Supports Byte or Word Suspend mode • Supports one data channel half-duplex transfer • Supports receive-only mode **I^2S** • Supports mono and stereo audio data with 8-, 16-, 24- and 32-bit audio data sizes • Supports PCM mode A, PCM mode B, I2S and MSB justified data format • PDMA operation
Universal Serial Control Interface (USCI)	• Up to 2 sets of USCI: USCI0 and USCI1 • Supports UART, SPI and I^2C function • Single byte TX and RX buffer mode **USCI_UART** • One transmit buffer and two receive buffer for data payload

	• Hardware auto flow control function and programmable flow control trigger level • Programmable baud-rate generator • Supports 9-bit data transfer • Baud rate detection by built-in capture event of baud rate generator • Supports Wake-up function (Data and nCTS Wakeup Only) • Supports PDMA transfer **USCI_SPI** • Master or Slave mode operation • Configurable bit length of a transfer word from 4 to 16-bit • One transmit buffer and two receive buffer for data payload • MSB first or LSB first transfer sequence • Word suspend function • Supports PDMA transfer • Supports 3-wire, no slave select signal, bi-direction interface • Wake-up function: input slave select transition • Supports one data channel half-duplex transfer **USCI_I2C** • Full master and slave device capability • 7-bit/10-bit addressing mode • Communication in Standard mode (100 kbps), Fast mode (up to 400 kbps) and Fast mode plus (1 Mbps) • Multi-master bus • One transmit buffer and two receive buffer for data payload • 10-bit bus time out capability • Supports Bus monitor mode • Wake-up by data toggle or address match in Power-down mode • Multiple address recognition • Setup/hold time programmable
External Bus Interface (EBI)	• Supports up to three memory banks • Supports dedicated external chip select pin with polarity control for each bank • Accessible space up to 1 Mbytes for each bank • Byte write in 16-bit data width mode • Address/Data multiplexed and separate mode • Timing parameters individual adjustment for each memory block

	• Supports LCD interface i80 mode • Supports Continuous Data Access mode • Supports PDMA mode
Controller Area Network with Feasibility Data Rate (CAN FD)	• Up to four sets of CAN FD controllers • Compliant with CAN protocol version 2.0 part A, B and ISO 11898-1: 2015 • Compliant with CAN FD version 1.0 with up to 64 data bytes supported • Supports CAN Error logging, AUTOSAR and SAE J1938 • Built-in 1.5K word (32-bit) Message SRAM for each CAN FD controller
GPIO	• Supports four I/O modes: Quasi bi-direction, Push-Pull output, Open-Drain output and Input only with high impendence mode • Selectable TTL/Schmitt trigger input • Configured as interrupt source with edge/level trigger setting • Supports independent pull-up/pull-down control • Supports software selectable slew rate control • Supports 5V-tolerance function except analog I/O • Configurable I/O mode of all pins after reset default to Quasi-bidirection mode or input mode
Control Interfaces	
Enhanced Quadrature Encoder Interface (EQEI)	• Two EQEI phase inputs (EQEI_A, EQEI_B) and one Index input (EQEI_INDEX) • Supports 2/4 times free-counting mode and 2/4 compare-counting mode • Supports encoder pulse width measurement mode with ECAP • Supports swap function for input signals (EQEI_A, EQEI_B) • Supports for detecting the occurrence of phase error • Supports one times index signal reset function
Enhanced Capture (ECAP)	• Up to four sets of Enhanced input Capture units • Supports three input channels with independent capture counter hold register • 24-bit Input Capture up-counting timer/counter supports captured events reset and/or reload capture counter • Supports rising edge, falling edge and both edge detector options with noise filter in front of input ports • Supports compare-match function
Advanced Connectivity	

USB 2.0 Full Speed with on-chip transceiver	• USB 2.0 Full Speed OTG (On-The-Go) • On-chip USB 2.0 full speed OTG transceiver • Compliant with USB OTG Supplement 2.0 • Configurable as host-only, device-only, ID-dependent or OTG device **USB 2.0 Full Speed Host Controller** • Compliant with USB Revision 1.1 Specification • Compatible with OHCI (Open Host Controller Interface) Revision 1.0 • Supports full-speed (12Mbps) and low-speed (1.5Mbps) USB devices • Supports Control, Bulk, Interrupt, Isochronous and Split transfers • Integrated a port routing logic to route full/low speed device to OHCI controller • Supports an integrated Root Hub • Supports port power control and port overcurrent detection • Built-in DMA **USB 2.0 Full Speed Device Controller** • Compliant with USB Revision 2.0 Specification • Supports suspend function when no bus activity exists for 3 ms • 25 configurable endpoints for configurable Isochronous, Bulk, Interrupt and Control transfer types • 1.5 Kbytes configurable RAM for endpoint buffer • Remote wake-up capability • Start of Frame (SOF) locked clock pulse generation for crystal-less feature • Supports double buffer function

3 PARTS INFORMATION

3.1 M2L31 Series Package Type

Package is Halogen-free, RoHS-compliant and TSCA-compliant.

Part No.	WLCSP25 (2.5x2.5mm)	WLCSP49 (3x3mm)	QFN33 (5x5mm)	QFN48 (5x5mm)	LQFP48 (7x7mm)	LQFP64 (7x7mm)	LQFP128 (14x14mm)
M2L31	M2L31XD4AE	M2L31CIDAE M2L31CGDAE	M2L31ZE4AE M2L31ZD4AE	M2L31YIDAE M2L31YGDAE M2L31YG4AE M2L31YE4AE M2L31YD4AE	M2L31LIDAE M2L31LGDAE M2L31LG4AE M2L31LE4AE M2L31LD4AE	M2L31SIDAE M2L31SGDAE M2L31SG4AE M2L31SE4AE	M2L31KIDAE M2L31KGDAE

3.2 M2L31 Series Naming Rule

M2	L31	K	I	D	A	E
Core	**Line**	**Package**	**ReRAM**	**SRAM**	**Reserve**	**Temperature**
Cortex-M23	L31: Low Power line	K: LQFP128 (14x14 mm) S: LQFP64 (7x7 mm) L: LQFP48 (7x7 mm) Y: QFN48 (5x5 mm) Z: QFN33 (5x5 mm) C: WLCSP49 X: WLCSP25	D: 64 KB E: 128 KB G: 256 KB I: 512 KB	4: 40 KB D: 168 KB	A	E:-40°C ~ 105°C

3.3 M2L31 Series Selection Guide

PART NO.		M2L31XD4AE	M2L31ZD4AE	M2L31ZE4AE	M2L31YD4AE	M2L31YE4AE	M2L31YG4AE	M2L31YGDAE	M2L31YIDAE	M2L31LD4AE	M2L31LE4AE	M2L31LG4AE	M2L31LGDAE	M2L31LIDAE	M2L31CIDAE	M2L31CGDAE	M2L31SE4AE	M2L31SG4AE	M2L31SGDAE	M2L31SIDAE	M2L31KGDAE	M2L31KIDAE
ReRAM (KB)		64	64	128	64	128	256	256	512	64	128	256	256	512	512	256	128	256	256	512	256	512
SRAM (KB)		40	40	40	40	40	40	168	168	40	40	40	168	168	168	168	40	40	168	168	168	168
Low Power SRAM (KB)		4	4	4	4	4	4	8	8	4	4	4	8	8	8	8	4	4	8	8	8	8
LDROM (KB)		8																				
32-bit Timer 16-bit PWM		4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
WDT/WWDT		√																				
LXT		√																				
RTC		√																				
16-bit EPWM		12																				
16-bit PWM		-	-	-	-	-	-	12	12	-	-	-	12	12	12	12	-	-	12	12	12	12
Connectivity	USCI*	-	1	1	1	1	1	2	2	1	1	1	2	2	2	2	1	1	2	2	2	2
	UART	5	6	6	6	6	6	8	8	6	6	6	8	8	8	8	6	6	8	8	8	8
	QSPI	-	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	SPI/I²S	2	2	2	2	2	2	4	4	2	2	2	4	4	4	4	2	2	4	4	4	4
	I²C	2	2	2	2	2	2	4	4	2	2	2	4	4	4	4	2	2	4	4	4	4
	CAN FD	1	1	1	1	1	1	2	2	1	1	1	2	2	2	2	1	1	2	2	2	2
	USB-C PD	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√
	VAI	-	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√
USB 2.0 FS OTG		√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√
LCD Parallel Data (External Bus Interface)		-	-	-	-	-	-	√	√	-	-	-	√	√	√	√	-	-	√	√	√	√
PDMA		10	10	10	10	10	10	16	16	10	10	10	16	16	16	16	10	10	16	16	16	16
EQEI		1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
ECAP		1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
CRC		√																				
Crypto		AES-256																				
TRNG		-	-	-	-	-	-	√	√	-	-	-	√	√	√	√	-	-	√	√	√	√
PRNG		-	-	-	-	-	-	√	√	-	-	-	√	√	√	√	-	-	√	√	√	√
PLL(MHz)		√																				
ISO-60730		√																				
Touchkey (ch)		-	-	-	-	-	-	11	11	-	-	-	11	11	11	11	-	-	16	16	17	17
ACMP		2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
OP Amplifier		-	2	2	2	2	2	3	3	2	2	2	3	3	3	3	2	2	3	3	3	3
12-bit 1MSPS DAC		1	1	1	1	1	1	2	2	1	1	1	2	2	2	2	1	1	2	2	2	2
12-bit ADC		6	10		16					16					16		20				24	
GPIO		18	26		41					41					40		53				109	
Package		WLCSP25	QFN32		QFN48					LQFP48					WLCSP49		LQFP64				LQFP 128	

USCI*: supports UART, SPI or I²C

3.4 M2L31xxDAE/M2L31xx4AE Features Comparison Table

Section	Sub-section	M2L31xxDAE	M2L31xx4AE
RMC Features	Dual-bank RRAM macro	●	

4 PIN CONFIGURATION

4.1 Pin Configuration

Users can find pin configuration information in the Multi-function Pin diagram sections or by using NuTool - PinConfigure. The NuTool - PinConfigure contains all NuMicro Family chip series with all part number, and helps users configure GPIO multi-function correctly and handily.

4.2 M2L31 Series Pin Diagram

4.2.1 QFN33-Pin Diagram

Corresponding Part Number: M2L31ZE4AE, M2L31ZD4AE

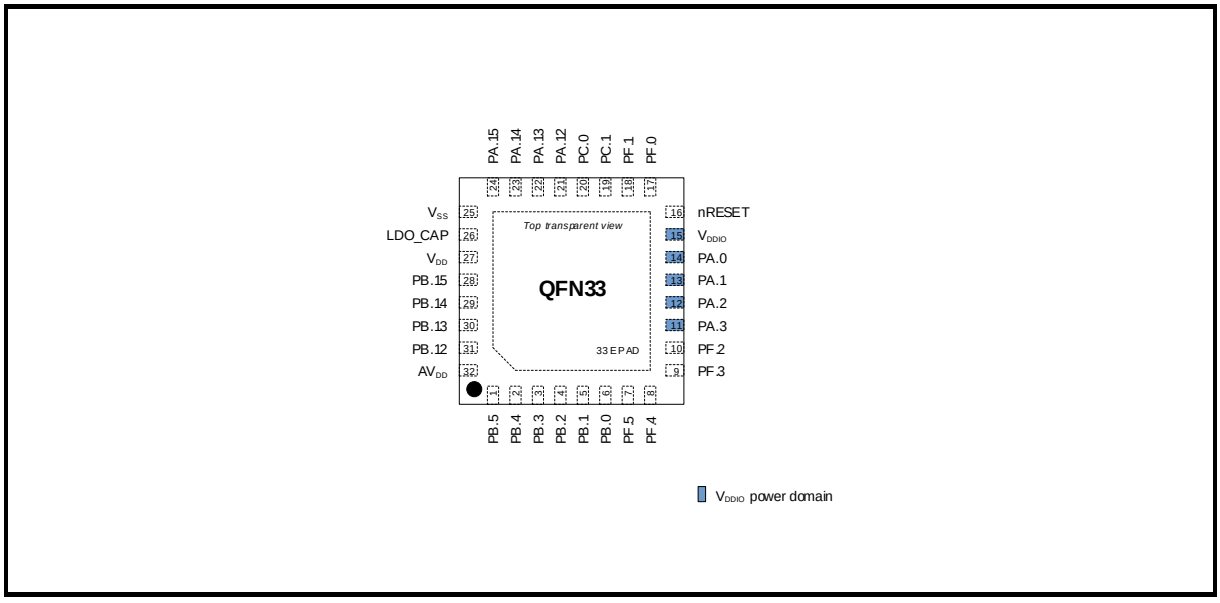


Figure 4.2-1 QFN33-Pin Diagram

4.2.2 QFN48-Pin Diagram

Corresponding Part Number: M2L31YG4AE, M2L31YE4AE, M2L31YD4AE, M2L31YIDAE, M2L31YGDAE

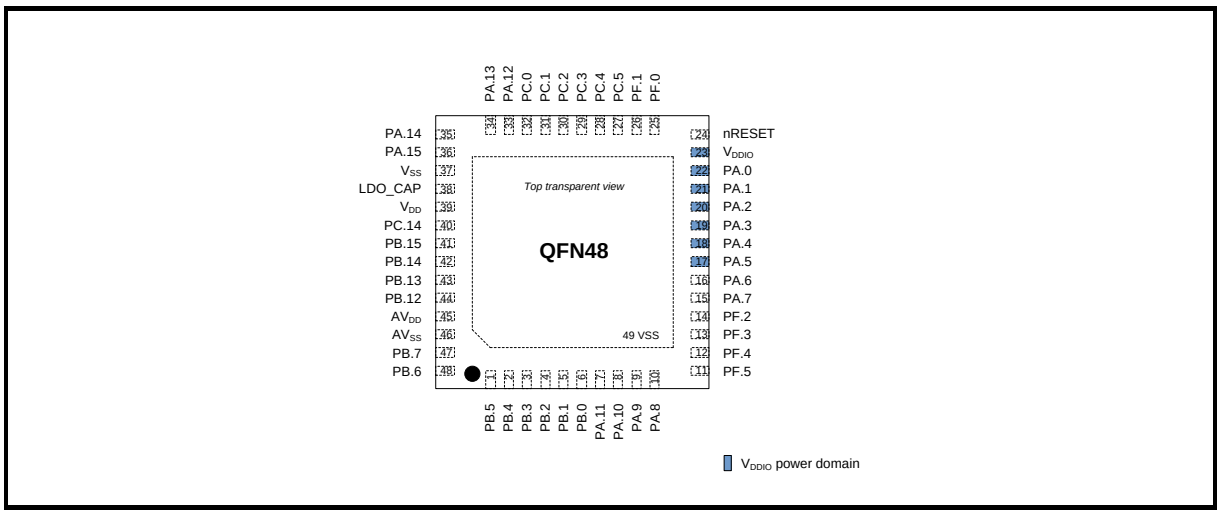


Figure 4.2-2 QFN48-Pin Diagram

4.2.3 LQFP48-Pin Diagram

Corresponding Part Number: M2L31LE4AE, M2L31LG4AE, M2L31LD4AE, M2L31LIDAE, M2L31LGDAE

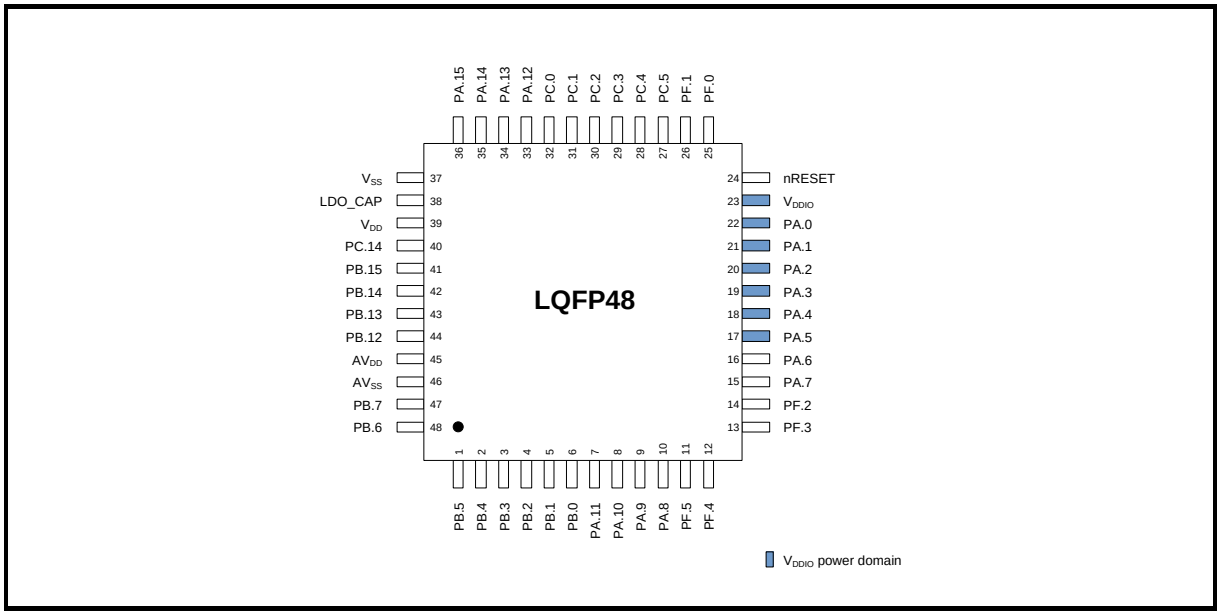


Figure 4.2-3 LQFP48-Pin Diagram

4.2.4 LQFP64-Pin Diagram

Corresponding Part Number: M2L31SG4AE, M2L31SE4AE, M2L31SIDAE, M2L31SGDAE

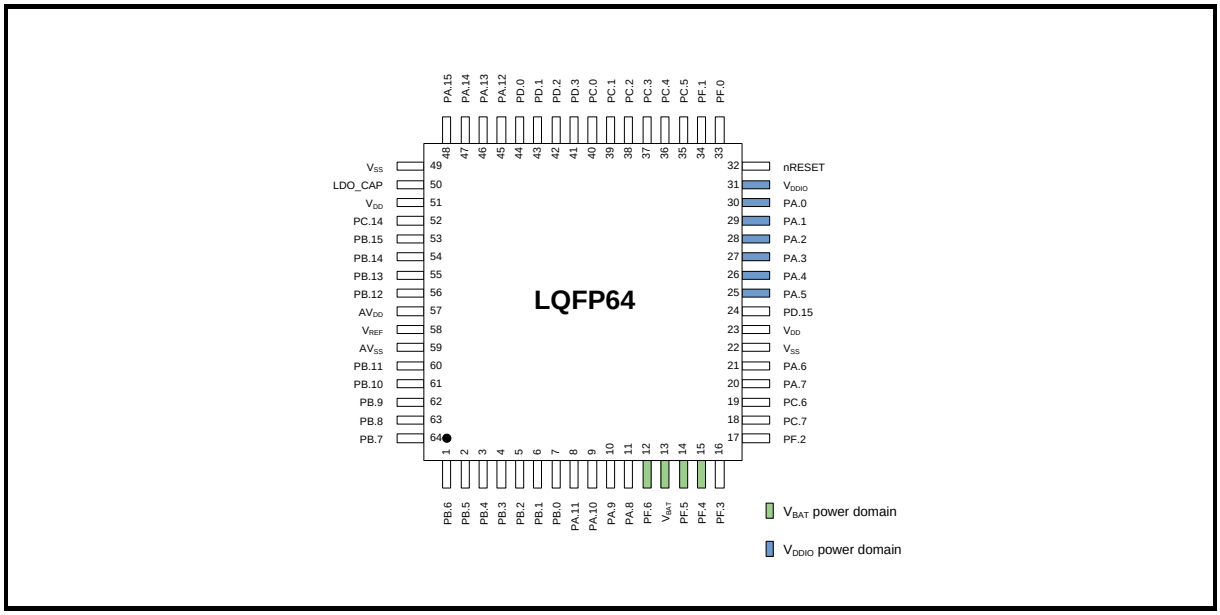


Figure 4.2-4 LQFP64-Pin Diagram

4.2.5 LQFP128-Pin Diagram

Corresponding Part Number: M2L31KIDAE, M2L31KGDAE

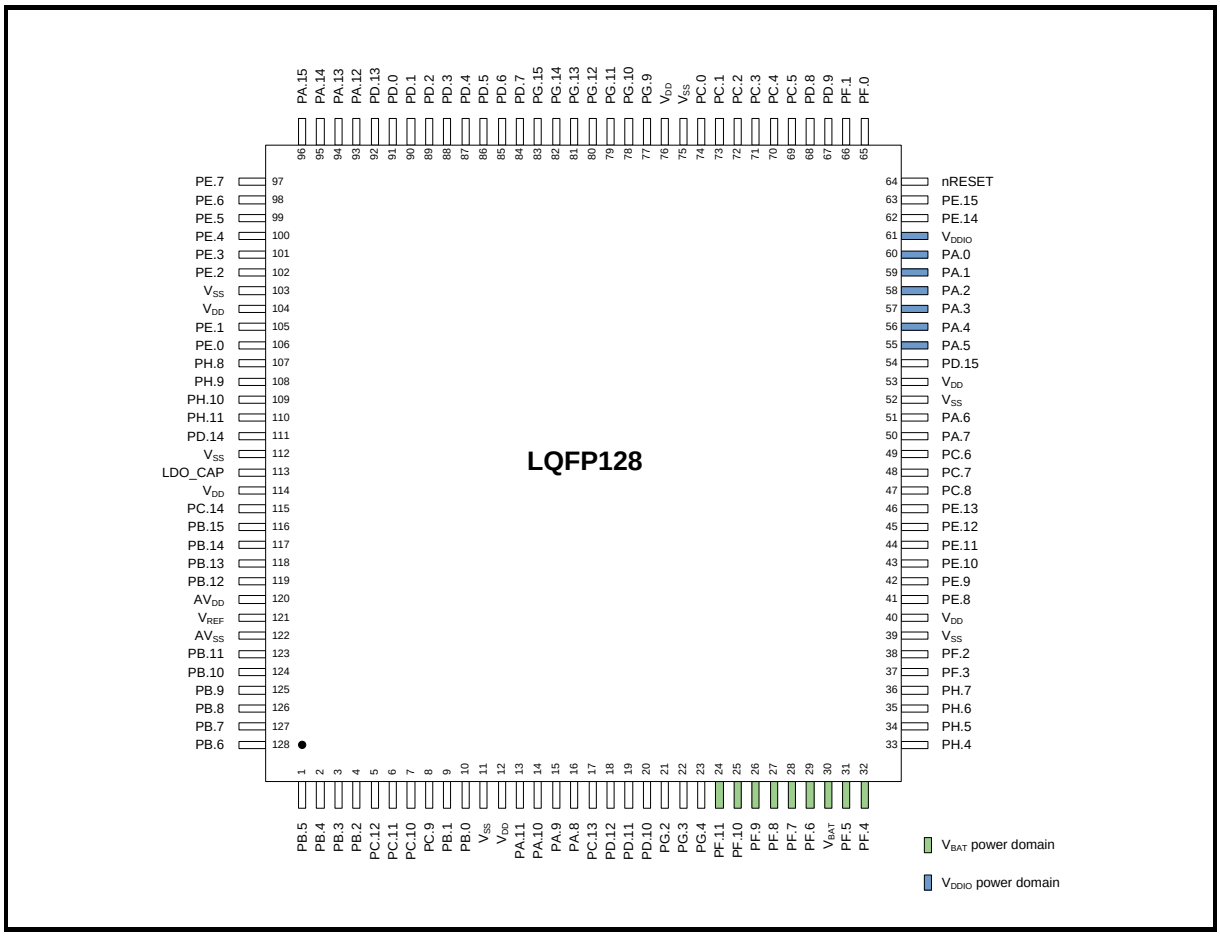


Figure 4.2-5 LQFP128-Pin Diagram

4.3 M2L31 Series Multi-function Diagram

4.3.1 QFN33-Pin Multi-function Diagram

Corresponding Part Number: M2L31ZE4AE, M2L31ZD4AE

4.3.1.1 M2L31ZE4AE

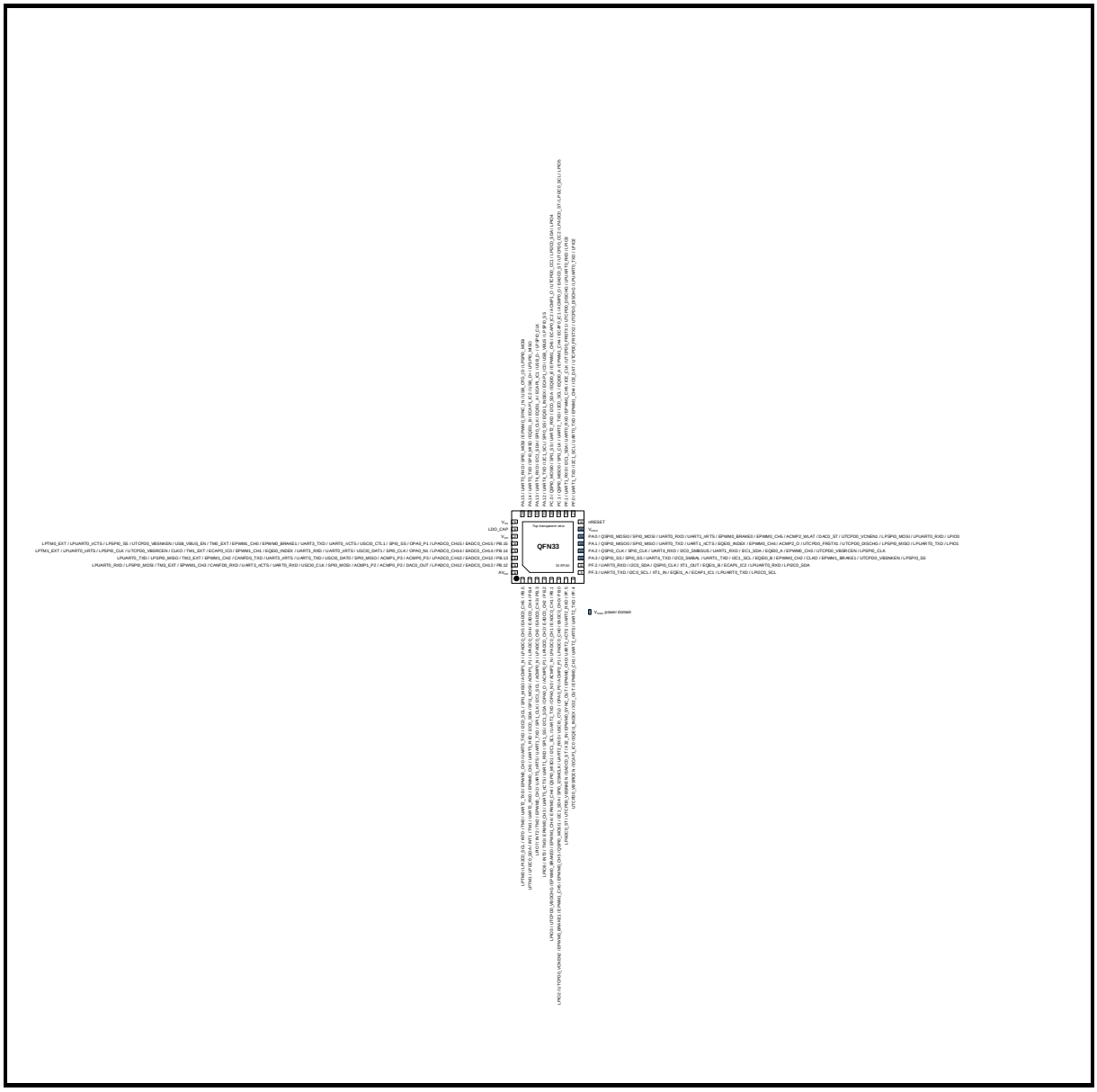


Figure 4.3-1 M2L31ZE4AE Multi-function Pin Diagram

Pin	Type	M2L31ZE4AE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / SPI1_MISO / I2C0_SCL / UART5_TXD / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1

Pin	Type	M2L31ZE4AE Pin Function
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / EPWM0_CH2 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / UART2_TXD / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
8	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
9	I/O	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
10	I/O	PF.2 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
11	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / EQEI0_B / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / UTCPD0_VBSNKEN / LPSPI0_SS
12	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / EPWM0_CH3 / UTCPD0_VBSRCEN / LPSPI0_CLK
13	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / EQEI0_INDEX / EPWM0_CH4 / ACMP2_O / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
14	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / EPWM0_BRAKE0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
15	P	V_{DDIO}
16	I	nRESET
17	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
18	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / EPWM1_CH5 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
19	I/O	PC.1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPADC0_ST / LPI2C0_SCL / LPIO5
20	I/O	PC.0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
21	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI0_SS / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
22	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI0_CLK / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
23	I/O	PA.14 / UART0_TXD / SPI0_MISO / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
24	I/O	PA.15 / UART0_RXD / SPI0_MOSI / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
25	P	V_{SS}
26	A	LDO_CAP
27	P	V_{DD}
28	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN /

Pin	Type	M2L31ZE4AE Pin Function
		UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
29	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
30	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
31	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
32	P	AV_{DD}
33	P	V_{SS}

Table 4.3-1 M2L31ZE4AE Multi-function Pin Table

4.3.1.2 M2L31ZD4AE

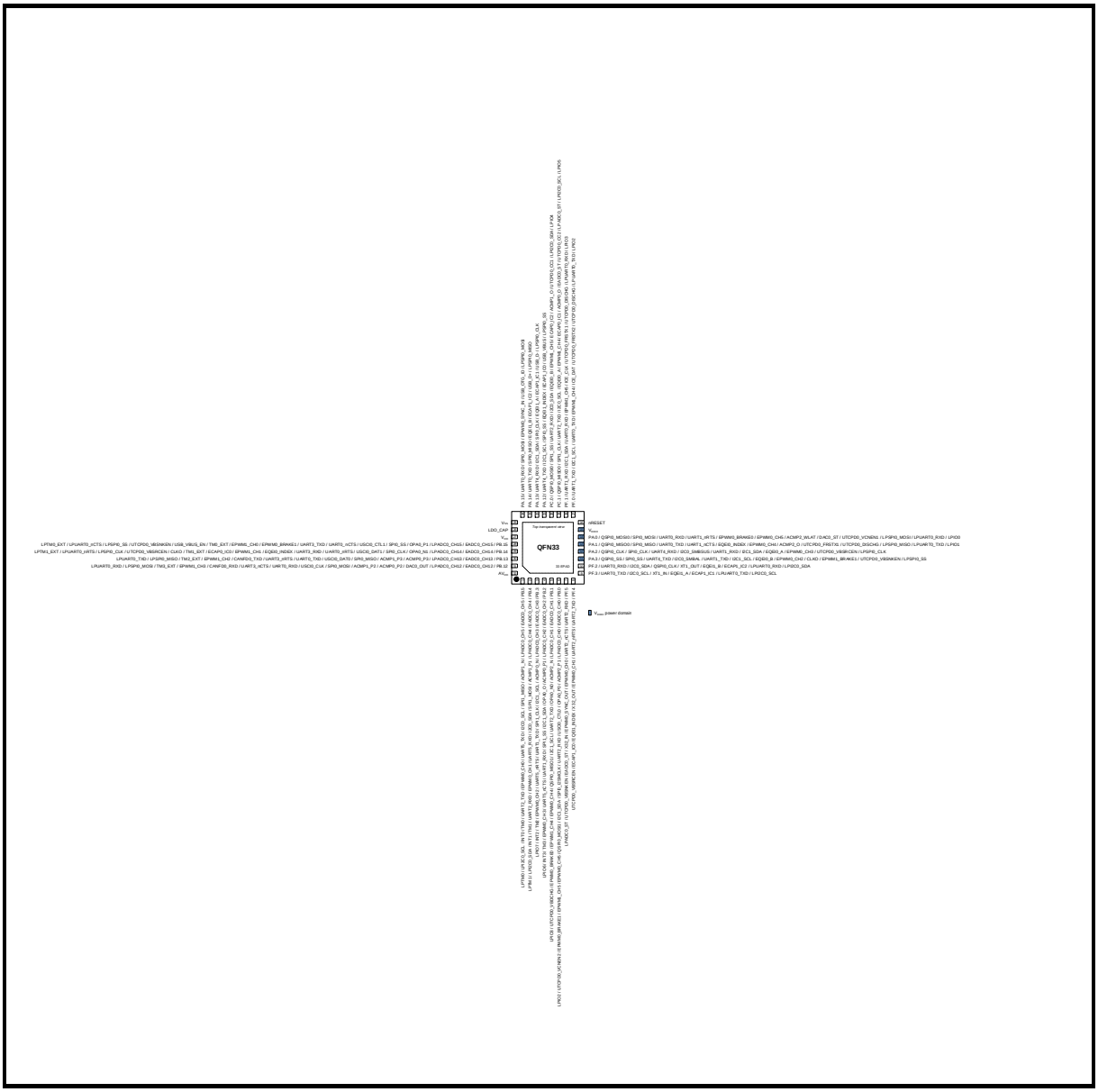


Figure 4.3-2 M2L31ZD4AE Multi-function Pin Diagram

Pin	Type	M2L31ZD4AE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / SPI1_MISO / I2C0_SCL / UART5_TXD / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / EPWM0_CH2 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / UART2_TXD / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3

Pin	Type	M2L31ZD4AE Pin Function
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
8	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
9	I/O	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
10	I/O	PF.2 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
11	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / EQEI0_B / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / UTCPD0_VBSNKEN / LPSPI0_SS
12	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / EPWM0_CH3 / UTCPD0_VBSRCEN / LPSPI0_CLK
13	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / EQEI0_INDEX / EPWM0_CH4 / ACMP2_O / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
14	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / EPWM0_BRAKE0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
15	P	V_{DDIO}
16	I	nRESET
17	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
18	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / EPWM1_CH5 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
19	I/O	PC.1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPADC0_ST / LPI2C0_SCL / LPIO5
20	I/O	PC.0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
21	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI0_SS / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
22	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI0_CLK / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
23	I/O	PA.14 / UART0_TXD / SPI0_MISO / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
24	I/O	PA.15 / UART0_RXD / SPI0_MOSI / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
25	P	V_{SS}
26	A	LDO_CAP
27	P	V_{DD}
28	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
29	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
30	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / CANFD0_TXD / EPWM1_CH2 / TM2_EXT /

Pin	Type	M2L31ZD4AE Pin Function
		LPSPI0_MISO / LPUART0_TXD
31	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
32	P	AV_{DD}
33	P	V_{SS}

Table 4.3-2 M2L31ZD4AE Multi-function Pin Table

4.3.2 QFN48-Pin Multi-function Diagram

Corresponding Part Number: M2L31YG4AE, M2L31YE4AE, M2L31YD4AE, M2L31YIDAE, M2L31YGDAE

4.3.2.1 M2L31YG4AE

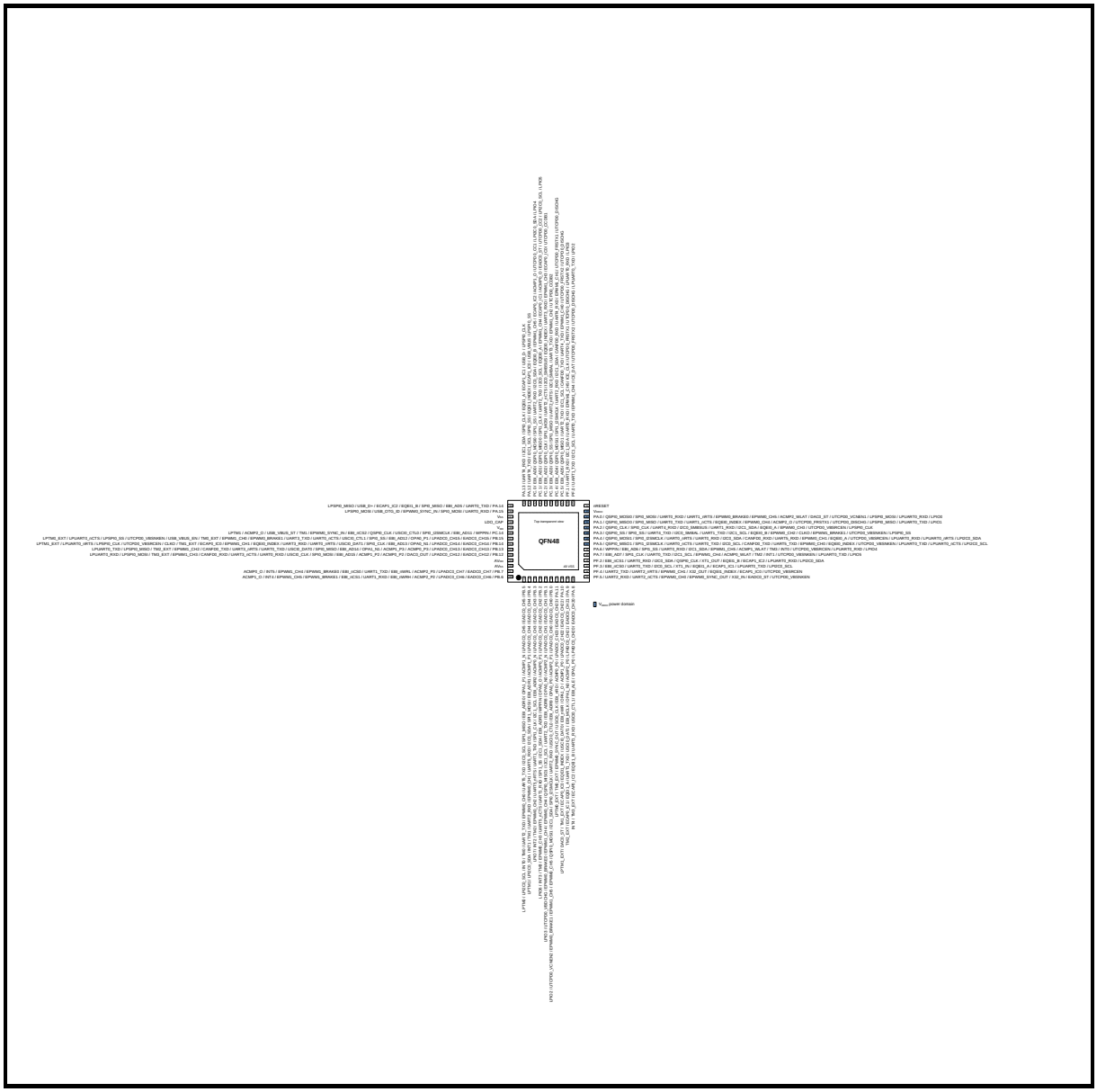


Figure 4.3-3 M2L31YG4AE Multi-function Pin Diagram

Pin	Type	M2L31YG4AE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / EPWM0_CH2 / TM2 / INT2 / LPIO7

Pin	Type	M2L31YG4AE Pin Function
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / EBI_nRD / USCI0_CLK / EPWM0_SYNC_OUT / TM0_EXT / LPTM0_EXT
8	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / USCI0_DAT0 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
9	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / EQEI1_A / ECAP0_IC1 / TM2_EXT
10	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / USCI0_CTL1 / UART1_RXD / EQEI1_B / ECAP0_IC2 / TM3_EXT / INT4
11	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
12	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
13	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
14	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
15	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / ACMP0_WLAT / TM2 / INT1 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
16	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / ACMP1_WLAT / TM3 / INT0 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
17	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / EPWM0_CH0 / EQEI0_INDEX / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
18	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / EPWM0_CH1 / EQEI0_A / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
19	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / EQEI0_B / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / UTCPD0_VBSNKEN / LPSPI0_SS
20	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / EPWM0_CH3 / UTCPD0_VBSRCEN / LPSPI0_CLK
21	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / EQEI0_INDEX / EPWM0_CH4 / ACMP2_O / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
22	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / EPWM0_BRAKE0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
23	P	V_{DDIO}
24	I	nRESET
25	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2

Pin	Type	M2L31YG4AE Pin Function
26	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / EPWM1_CH5 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
27	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
28	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
29	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / UTCPD0_CCDB2
30	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / UTCPD0_CCDB1
31	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
32	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
33	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI0_SS / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
34	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI0_CLK / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
35	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI0_MISO / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
36	I/O	PA.15 / UART0_RXD / SPI0_MOSI / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
37	P	V_{SS}
38	A	LDO_CAP
39	P	V_{DD}
40	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
41	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
42	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
43	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
44	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
45	P	AV_{DD}
46	P	AV_{SS}
47	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / EBI_nWRL / UART1_TXD / EBI_nCS0 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / ACMP0_O
48	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / EBI_nWRH / UART1_RXD / EBI_nCS1 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / ACMP1_O

Table 4.3-3 M2L31YG4AE Multi-function Pin Table

4.3.2.2 M2L31YE4AE

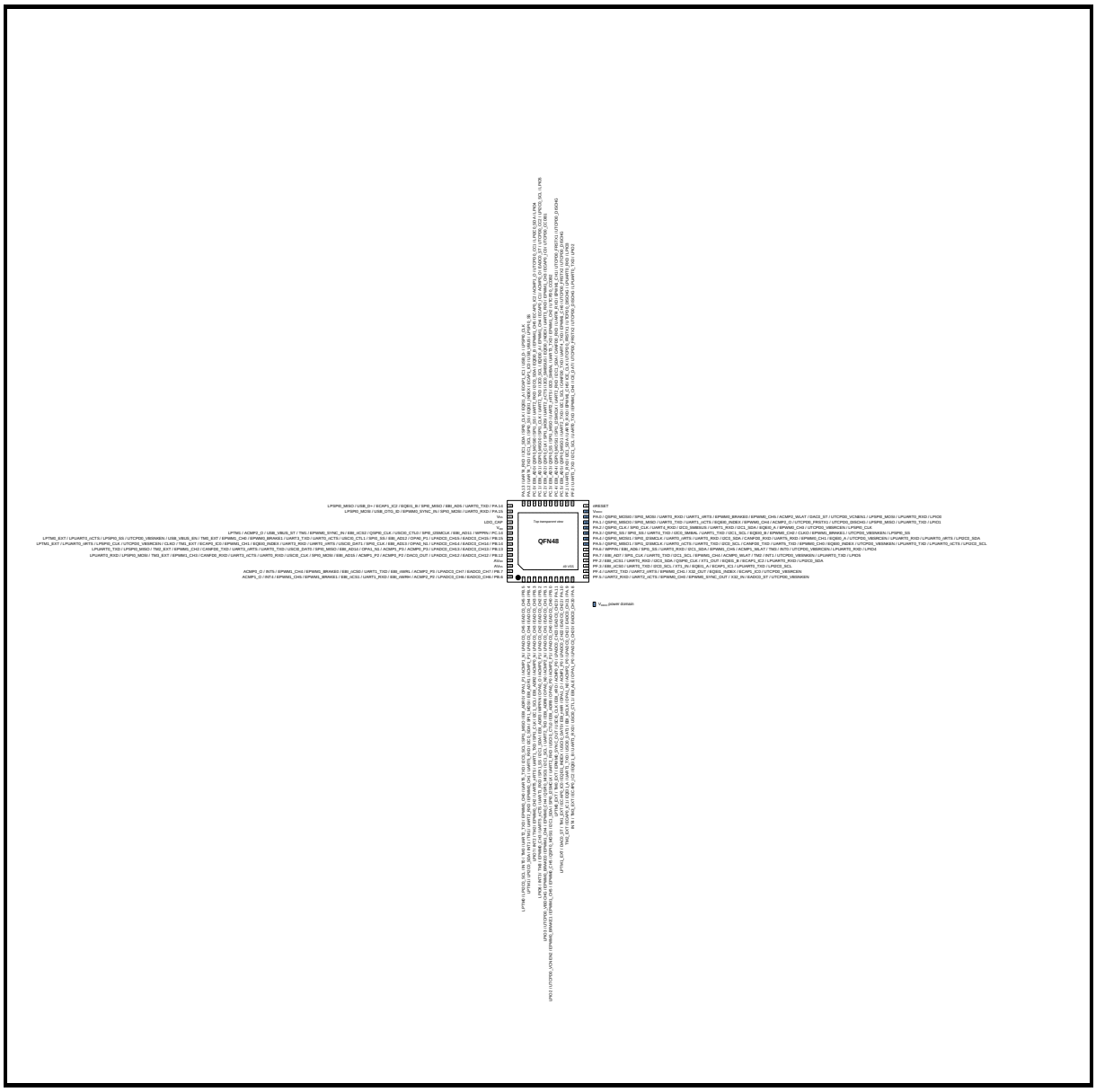


Figure 4.3-4 M2L31YE4AE Multi-function Pin Diagram

Pin	Type	M2L31YE4AE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / EPWM0_CH2 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / EPWM0_CH3 / TM3 / INT3 / LPIO6

Pin	Type	M2L31YE4AE Pin Function
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / EBI_nRD / USCI0_CLK / EPWM0_SYNC_OUT / TM0_EXT / LPTM0_EXT
8	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / USCI0_DAT0 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
9	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / EQEI1_A / ECAP0_IC1 / TM2_EXT
10	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / USCI0_CTL1 / UART1_RXD / EQEI1_B / ECAP0_IC2 / TM3_EXT / INT4
11	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
12	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
13	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
14	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
15	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / ACMP0_WLAT / TM2 / INT1 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
16	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / ACMP1_WLAT / TM3 / INT0 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
17	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / EPWM0_CH0 / EQEI0_INDEX / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
18	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / EPWM0_CH1 / EQEI0_A / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
19	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / EQEI0_B / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / UTCPD0_VBSNKEN / LPSPI0_SS
20	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / EPWM0_CH3 / UTCPD0_VBSRCEN / LPSPI0_CLK
21	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / EQEI0_INDEX / EPWM0_CH4 / ACMP2_O / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
22	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / EPWM0_BRAKE0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
23	P	V_{DDIO}
24	I	nRESET
25	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
26	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / EPWM1_CH5 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3

Pin	Type	M2L31YE4AE Pin Function
27	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
28	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
29	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / UTCPD0_CCDB2
30	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / UTCPD0_CCDB1
31	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
32	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
33	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI0_SS / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
34	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI0_CLK / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
35	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI0_MISO / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
36	I/O	PA.15 / UART0_RXD / SPI0_MOSI / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
37	P	V_{SS}
38	A	LDO_CAP
39	P	V_{DD}
40	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
41	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
42	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
43	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
44	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
45	P	AV_{DD}
46	P	AV_{SS}
47	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / EBI_nWRL / UART1_TXD / EBI_nCS0 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / ACMP0_O
48	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / EBI_nWRH / UART1_RXD / EBI_nCS1 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / ACMP1_O

Table 4.3-4 M2L31YE4AE Multi-function Pin Table

4.3.2.3 M2L31YD4AE

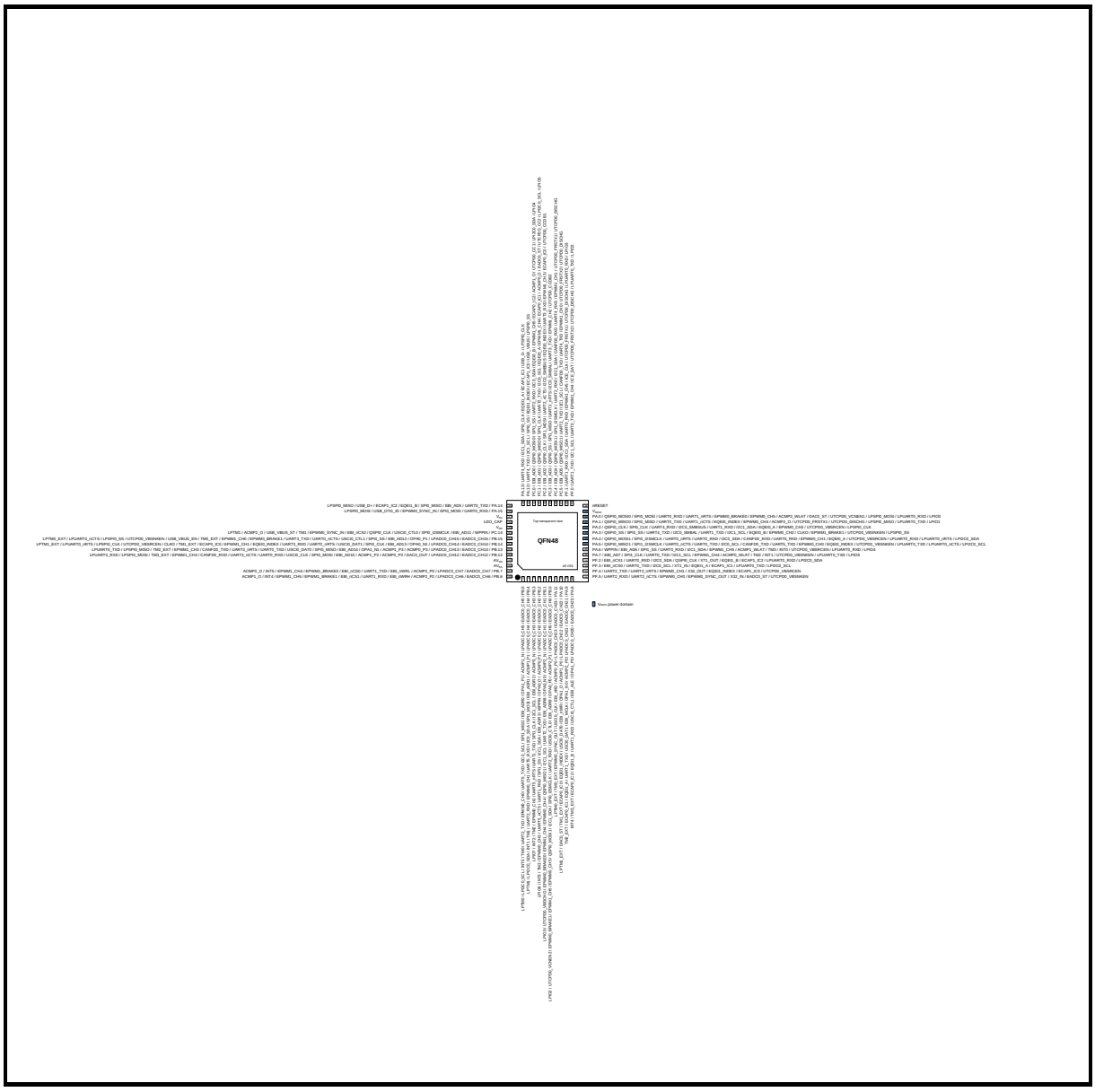


Figure 4.3-5 M2L31YD4AE Multi-function Pin Diagram

Pin	Type	M2L31YD4AE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / EPWM0_CH2 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3

Pin	Type	M2L31YD4AE Pin Function
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / EBI_nRD / USCI0_CLK / EPWM0_SYNC_OUT / TM0_EXT / LPTM0_EXT
8	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / USCI0_DAT0 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
9	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / EQEI1_A / ECAP0_IC1 / TM2_EXT
10	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / USCI0_CTL1 / UART1_RXD / EQEI1_B / ECAP0_IC2 / TM3_EXT / INT4
11	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
12	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
13	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
14	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
15	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / ACMP0_WLAT / TM2 / INT1 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
16	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / ACMP1_WLAT / TM3 / INT0 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
17	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / EPWM0_CH0 / EQEI0_INDEX / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
18	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / EPWM0_CH1 / EQEI0_A / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
19	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / EQEI0_B / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / UTCPD0_VBSNKEN / LPSPI0_SS
20	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / EPWM0_CH3 / UTCPD0_VBSRCEN / LPSPI0_CLK
21	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / EQEI0_INDEX / EPWM0_CH4 / ACMP2_O / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
22	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / EPWM0_BRAKE0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
23	P	V_{DDIO}
24	I	nRESET
25	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
26	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / EPWM1_CH5 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
27	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / UTCPD0_FRSTX2 / UTCPD0_DISCHG

Pin	Type	M2L31YD4AE Pin Function
28	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
29	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / UTCPD0_CCDB2
30	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / UTCPD0_CCDB1
31	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
32	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
33	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI0_SS / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
34	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI0_CLK / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
35	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI0_MISO / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
36	I/O	PA.15 / UART0_RXD / SPI0_MOSI / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
37	P	V_{SS}
38	A	LDO_CAP
39	P	V_{DD}
40	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
41	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
42	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
43	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
44	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
45	P	AV_{DD}
46	P	AV_{SS}
47	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / EBI_nWRL / UART1_TXD / EBI_nCS0 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / ACMP0_O
48	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / EBI_nWRH / UART1_RXD / EBI_nCS1 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / ACMP1_O

Table 4.3-5 M2L31YD4AE Multi-function Pin Table

4.3.2.4 M2L31YIDAE

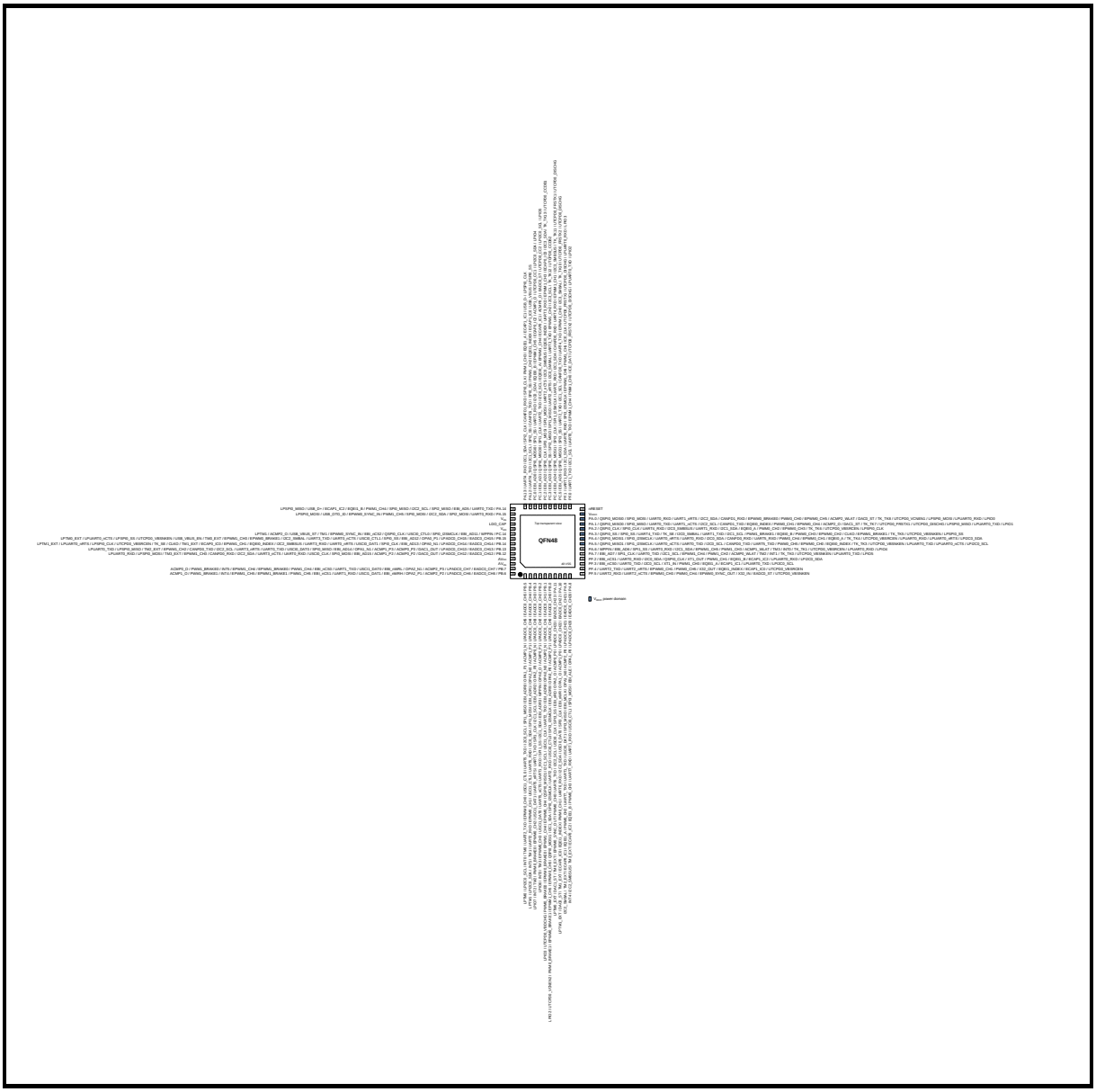


Figure 4.3-6 M2L31YIDAE Multi-function Pin Diagram

Pin	Type	M2L31YIDAE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / USCI1_CTL0 / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / OPA2_N0 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / USCI1_CTL1 / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / OPA2_P0 / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / USCI1_DAT1 / EPWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / USCI1_DAT0 / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / PWM0_BRAKE0 /

Pin	Type	M2L31YIDAE Pin Function
		UTCPD0_VBDCHG / LPIO3
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / SPI3_I2SMCLK / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / PWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / OPA2_O / EBI_nRD / SPI3_SS / USCI0_CLK / I2C2_SCL / UART6_TXD / PWM0_CH0 / EPWM0_SYNC_OUT / TM0_EXT / DAC1_ST / LPTM0_EXT
8	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / SPI3_CLK / USCI0_DAT0 / I2C2_SDA / UART6_RXD / PWM0_CH1 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
9	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / SPI3_MISO / USCI0_DAT1 / UART1_TXD / UART7_TXD / PWM0_CH2 / EQEI1_A / ECAP0_IC1 / TM2_EXT / I2C2_SMBAL
10	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / SPI3_MOSI / USCI0_CTL1 / UART1_RXD / UART7_RXD / PWM0_CH3 / EQEI1_B / ECAP0_IC2 / TM3_EXT / I2C2_SMBSUS / INT4
11	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / PWM0_CH4 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
12	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / PWM0_CH5 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
13	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / PWM1_CH0 / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
14	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / PWM1_CH1 / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
15	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / PWM1_CH2 / ACMP0_WLAT / TM2 / INT1 / TK_TK0 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
16	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / PWM1_CH3 / ACMP1_WLAT / TM3 / INT0 / TK_TK1 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
17	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / PWM0_CH5 / EPWM0_CH0 / EQEI0_INDEX / TK_TK3 / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
18	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / PWM0_CH4 / EPWM0_CH1 / EQEI0_A / TK_TK4 / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
19	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / TK_SE / I2C0_SMBAL / UART1_TXD / I2C1_SCL / PWM1_BRAKE1 / EQEI0_B / PWM0_CH3 / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / TK_TK5 / UTCPD0_VBSNKEN / LPSPI0_SS
20	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / PWM0_CH2 / EPWM0_CH3 / TK_TK6 / UTCPD0_VBSRCEN / LPSPI0_CLK
21	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / I2C2_SCL / CANFD1_TXD / EQEI0_INDEX / PWM0_CH1 / EPWM0_CH4 / ACMP2_O / DAC1_ST / TK_TK7 / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
22	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / I2C2_SDA / CANFD1_RXD / EPWM0_BRAKE0 / PWM0_CH0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / TK_TK8 / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
23	P	V_{DDIO}
24	I	nRESET
25	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / PWM1_CH0 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2

Pin	Type	M2L31YIDAE Pin Function
26	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / SPI3_I2SMCLK / EPWM1_CH5 / PWM1_CH1 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
27	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / SPI3_SS / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / I2C3_SMBAL / TK_TK10 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
28	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI3_CLK / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / I2C3_SMBSUS / TK_TK11 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
29	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI3_MISO / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / I2C3_SCL / TK_TK12 / UTCPD0_CCDB2
30	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI3_MOSI / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / I2C3_SDA / TK_TK13 / UTCPD0_CCDB1
31	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
32	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
33	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI2_SS / CANFD1_TXD / SPI0_SS / PWM1_CH2 / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
34	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI2_CLK / CANFD1_RXD / SPI0_CLK / PWM1_CH3 / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
35	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI2_MISO / I2C2_SCL / SPI0_MISO / PWM1_CH4 / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
36	I/O	PA.15 / UART0_RXD / SPI2_MOSI / I2C2_SDA / SPI0_MOSI / PWM1_CH5 / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
37	P	V_{SS}
38	A	LDO_CAP
39	P	V_{DD}
40	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
41	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / I2C2_SMBAL / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
42	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / I2C2_SMBSUS / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / TK_SE / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
43	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / DAC1_OUT / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / I2C2_SCL / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
44	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / I2C2_SDA / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
45	P	AV_{DD}
46	P	AV_{SS}
47	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / OPA2_N1 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / PWM1_CH4 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / PWM1_BRAKE0 / ACMP0_O

Pin	Type	M2L31YIDAE Pin Function
48	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / OPA2_P1 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / PWM1_CH5 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / PWM1_BRAKE1 / ACMP1_O

Table 4.3-6 M2L31YIDAE Multi-function Pin Table

4.3.2.5 M2L31YGDAE

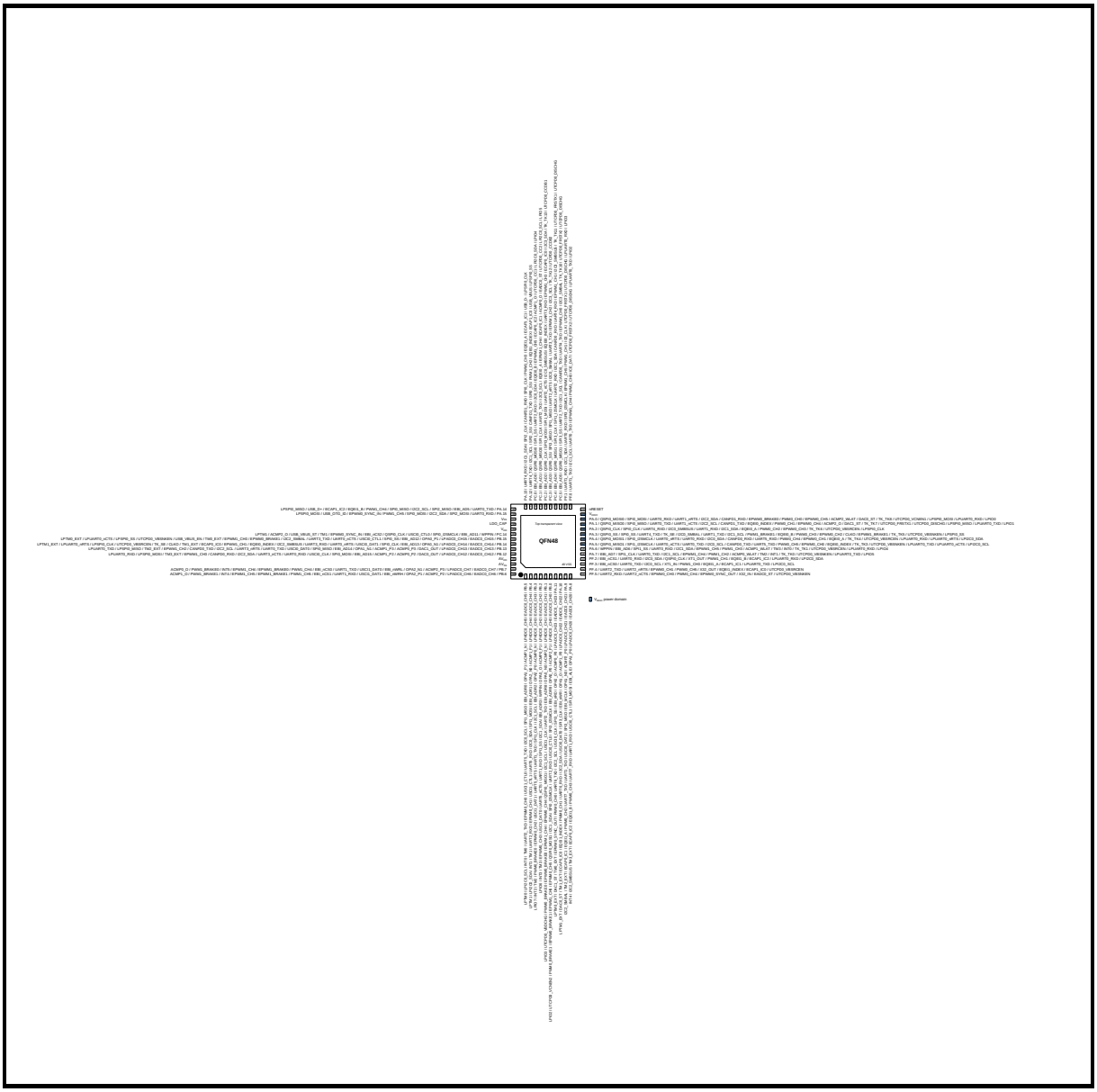


Figure 4.3-7 M2L31YGDAE Multi-function Pin Diagram

Pin	Type	M2L31YGDAE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / USCI1_CTL0 / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / OPA2_N0 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / USCI1_CTL1 / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / OPA2_P0 / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / USCI1_DAT1 / EPWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / USCI1_DAT0 / EPWM0_CH3 / TM3 / INT3 / LPIO6

Pin	Type	M2L31YGDAE Pin Function
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / PWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / SPI3_I2SMCLK / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / PWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / OPA2_O / EBI_nRD / SPI3_SS / USCI0_CLK / I2C2_SCL / UART6_TXD / PWM0_CH0 / EPWM0_SYNC_OUT / TM0_EXT / DAC1_ST / LPTM0_EXT
8	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / SPI3_CLK / USCI0_DAT0 / I2C2_SDA / UART6_RXD / PWM0_CH1 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
9	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / SPI3_MISO / USCI0_DAT1 / UART1_TXD / UART7_TXD / PWM0_CH2 / EQEI1_A / ECAP0_IC1 / TM2_EXT / I2C2_SMBAL
10	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / SPI3_MOSI / USCI0_CTL1 / UART1_RXD / UART7_RXD / PWM0_CH3 / EQEI1_B / ECAP0_IC2 / TM3_EXT / I2C2_SMBSUS / INT4
11	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / PWM0_CH4 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
12	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / PWM0_CH5 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
13	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / PWM1_CH0 / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
14	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / PWM1_CH1 / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
15	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / PWM1_CH2 / ACMP0_WLAT / TM2 / INT1 / TK_TK0 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
16	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / PWM1_CH3 / ACMP1_WLAT / TM3 / INT0 / TK_TK1 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
17	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / PWM0_CH5 / EPWM0_CH0 / EQEI0_INDEX / TK_TK3 / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
18	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / PWM0_CH4 / EPWM0_CH1 / EQEI0_A / TK_TK4 / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
19	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / TK_SE / I2C0_SMBAL / UART1_TXD / I2C1_SCL / PWM1_BRAKE1 / EQEI0_B / PWM0_CH3 / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / TK_TK5 / UTCPD0_VBSNKEN / LPSPI0_SS
20	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / PWM0_CH2 / EPWM0_CH3 / TK_TK6 / UTCPD0_VBSRCEN / LPSPI0_CLK
21	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / I2C2_SCL / CANFD1_TXD / EQEI0_INDEX / PWM0_CH1 / EPWM0_CH4 / ACMP2_O / DAC1_ST / TK_TK7 / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
22	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / I2C2_SDA / CANFD1_RXD / EPWM0_BRAKE0 / PWM0_CH0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / TK_TK8 / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
23	P	V_{DDIO}
24	I	nRESET

Pin	Type	M2L31YGDAE Pin Function
25	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / PWM1_CH0 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
26	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / SPI3_I2SMCLK / EPWM1_CH5 / PWM1_CH1 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
27	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / SPI3_SS / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / I2C3_SMBAL / TK_TK10 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
28	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI3_CLK / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / I2C3_SMBSUS / TK_TK11 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
29	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI3_MISO / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / I2C3_SCL / TK_TK12 / UTCPD0_CCDB2
30	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI3_MOSI / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / I2C3_SDA / TK_TK13 / UTCPD0_CCDB1
31	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
32	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
33	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI2_SS / CANFD1_TXD / SPI0_SS / PWM1_CH2 / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
34	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI2_CLK / CANFD1_RXD / SPI0_CLK / PWM1_CH3 / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
35	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI2_MISO / I2C2_SCL / SPI0_MISO / PWM1_CH4 / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
36	I/O	PA.15 / UART0_RXD / SPI2_MOSI / I2C2_SDA / SPI0_MOSI / PWM1_CH5 / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
37	P	V_{SS}
38	A	LDO_CAP
39	P	V_{DD}
40	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
41	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / I2C2_SMBAL / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
42	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / I2C2_SMBSUS / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / TK_SE / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
43	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / DAC1_OUT / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / I2C2_SCL / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
44	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / I2C2_SDA / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
45	P	AV_{DD}
46	P	AV_{SS}
47	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / OPA2_N1 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / PWM1_CH4 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / PWM1_BRAKE0

Pin	Type	M2L31YGDAE Pin Function
		/ ACMP0_O
48	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / OPA2_P1 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / PWM1_CH5 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / PWM1_BRAKE1 / ACMP1_O

Table 4.3-7 M2L31YGDAE Multi-function Pin Table

4.3.3 LQFP48-Pin Multi-function Diagram

Corresponding Part Number: M2L31LE4AE, M2L31LG4AE, M2L31LD4AE, M2L31LIDAE, M2L31LGDAE

4.3.3.1 M2L31LE4AE

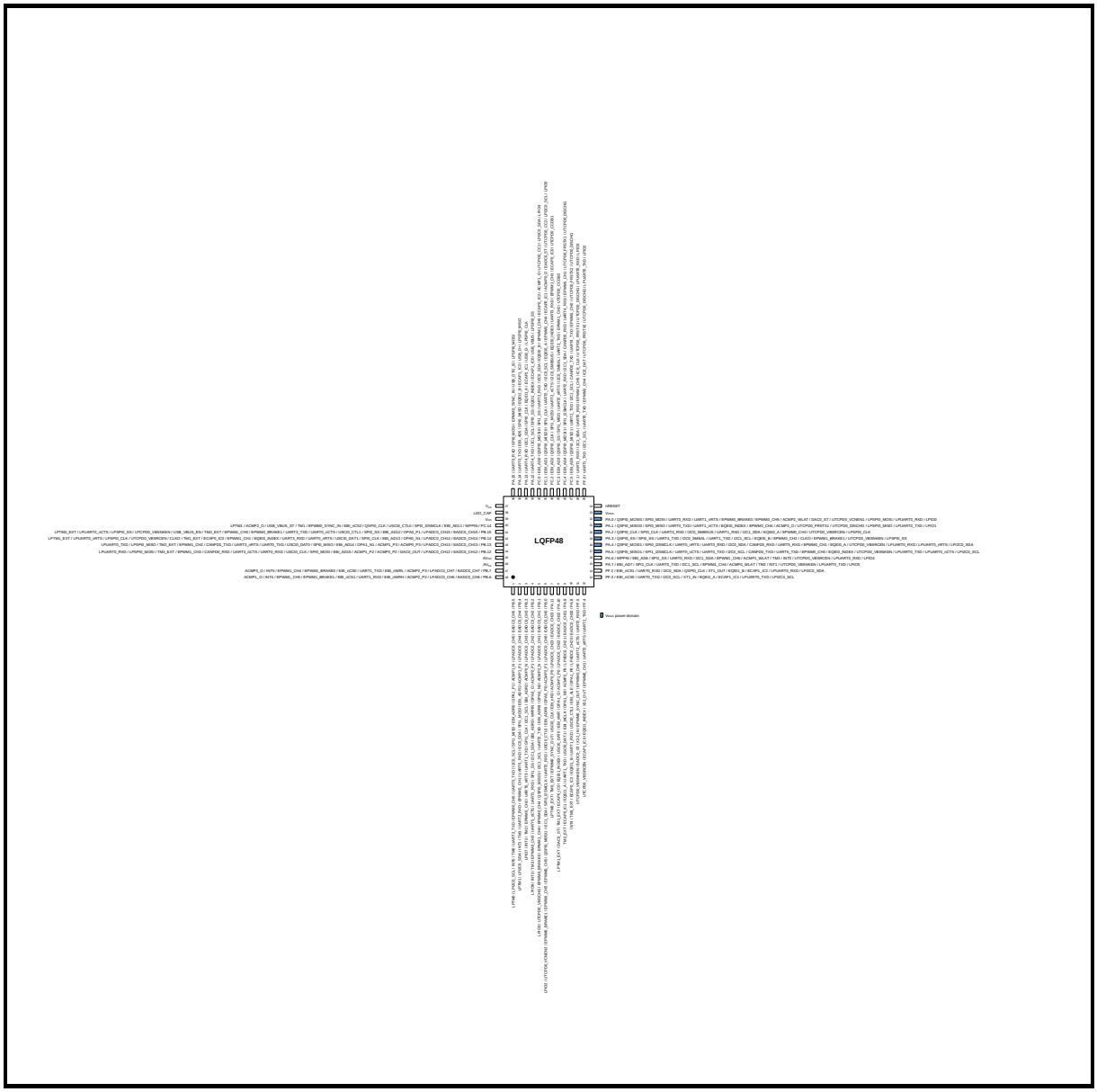


Figure 4.3-8 M2L31LE4AE Multi-function Pin Diagram

Pin	Type	M2L31LE4AE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / EPWM0_CH2 / TM2 / INT2 / LPIO7

Pin	Type	M2L31LE4AE Pin Function
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / EBI_nRD / USCI0_CLK / EPWM0_SYNC_OUT / TM0_EXT / LPTM0_EXT
8	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / USCI0_DAT0 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
9	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / EQEI1_A / ECAP0_IC1 / TM2_EXT
10	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / USCI0_CTL1 / UART1_RXD / EQEI1_B / ECAP0_IC2 / TM3_EXT / INT4
11	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
12	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
13	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
14	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
15	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / ACMP0_WLAT / TM2 / INT1 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
16	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / ACMP1_WLAT / TM3 / INT0 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
17	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / EPWM0_CH0 / EQEI0_INDEX / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
18	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / EPWM0_CH1 / EQEI0_A / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
19	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / EQEI0_B / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / UTCPD0_VBSNKEN / LPSPI0_SS
20	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / EPWM0_CH3 / UTCPD0_VBSRCEN / LPSPI0_CLK
21	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / EQEI0_INDEX / EPWM0_CH4 / ACMP2_O / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
22	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / EPWM0_BRAKE0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
23	P	V_{DDIO}
24	I	nRESET
25	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2

Pin	Type	M2L31LE4AE Pin Function
26	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / EPWM1_CH5 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
27	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
28	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
29	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / UTCPD0_CCDB2
30	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / UTCPD0_CCDB1
31	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
32	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
33	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI0_SS / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
34	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI0_CLK / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
35	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI0_MISO / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
36	I/O	PA.15 / UART0_RXD / SPI0_MOSI / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
37	P	V_{SS}
38	A	LDO_CAP
39	P	V_{DD}
40	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
41	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
42	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
43	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
44	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
45	P	AV_{DD}
46	P	AV_{SS}
47	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / EBI_nWRL / UART1_TXD / EBI_nCS0 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / ACMP0_O
48	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / EBI_nWRH / UART1_RXD / EBI_nCS1 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / ACMP1_O

Table 4.3-8 M2L31LE4AE Multi-function Pin Table

4.3.3.2 M2L31LG4AE

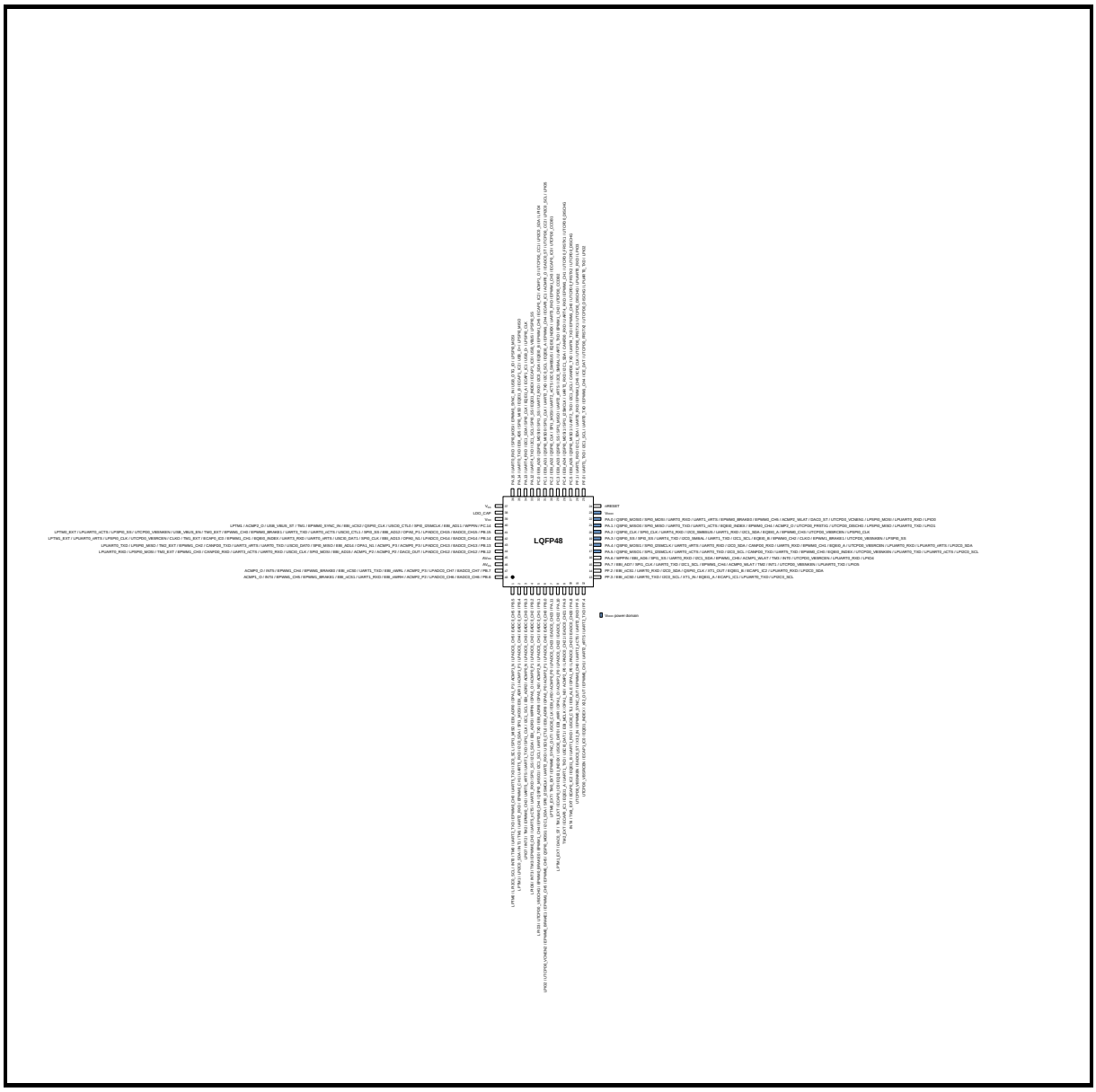


Figure 4.3-9 M2L31LG4AE Multi-function Pin Diagram

Pin	Type	M2L31LG4AE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / EPWM0_CH2 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / EPWM0_CH3 / TM3 / INT3 / LPIO6

Pin	Type	M2L31LG4AE Pin Function
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / EBI_nRD / USCI0_CLK / EPWM0_SYNC_OUT / TM0_EXT / LPTM0_EXT
8	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / USCI0_DAT0 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
9	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / EQEI1_A / ECAP0_IC1 / TM2_EXT
10	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / USCI0_CTL1 / UART1_RXD / EQEI1_B / ECAP0_IC2 / TM3_EXT / INT4
11	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
12	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
13	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
14	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
15	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / ACMP0_WLAT / TM2 / INT1 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
16	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / ACMP1_WLAT / TM3 / INT0 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
17	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / EPWM0_CH0 / EQEI0_INDEX / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
18	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / EPWM0_CH1 / EQEI0_A / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
19	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / EQEI0_B / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / UTCPD0_VBSNKEN / LPSPI0_SS
20	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / EPWM0_CH3 / UTCPD0_VBSRCEN / LPSPI0_CLK
21	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / EQEI0_INDEX / EPWM0_CH4 / ACMP2_O / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
22	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / EPWM0_BRAKE0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
23	P	V_{DDIO}
24	I	nRESET
25	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
26	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / EPWM1_CH5 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3

Pin	Type	M2L31LG4AE Pin Function
27	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
28	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
29	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / UTCPD0_CCDB2
30	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / UTCPD0_CCDB1
31	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
32	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
33	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI0_SS / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
34	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI0_CLK / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
35	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI0_MISO / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
36	I/O	PA.15 / UART0_RXD / SPI0_MOSI / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
37	P	V_{SS}
38	A	LDO_CAP
39	P	V_{DD}
40	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
41	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
42	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
43	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
44	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
45	P	AV_{DD}
46	P	AV_{SS}
47	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / EBI_nWRL / UART1_TXD / EBI_nCS0 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / ACMP0_O
48	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / EBI_nWRH / UART1_RXD / EBI_nCS1 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / ACMP1_O

Table 4.3-9 M2L31LG4AE Multi-function Pin Table

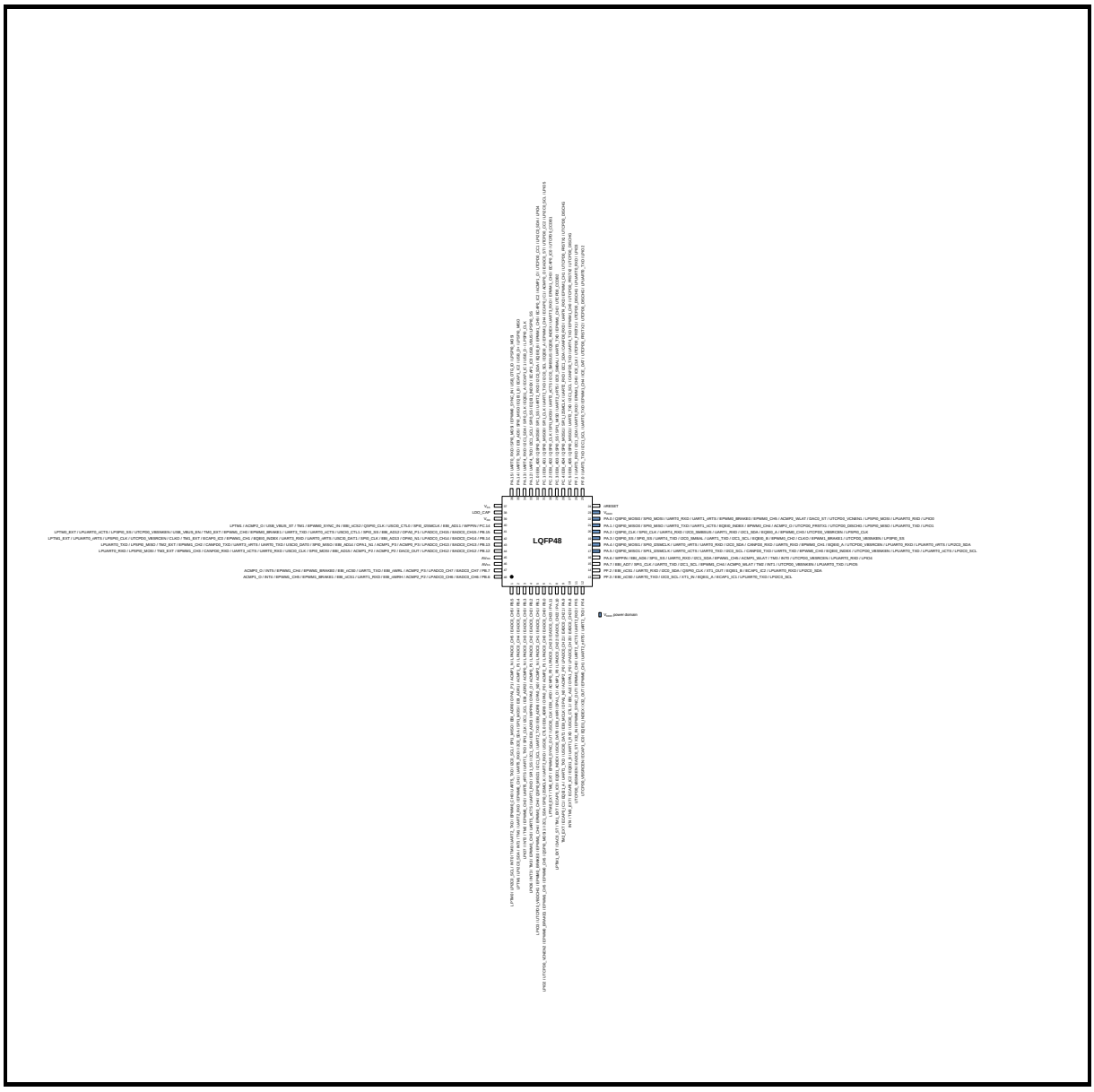


Figure 4.3-10 M2L31LD4AE Multi-function Pin Diagram

Pin	Type	M2L31LD4AE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / EPWM0_CH2 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / USCI0_CTL0 /

Pin	Type	M2L31LD4AE Pin Function
		UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / EBI_nRD / USCI0_CLK / EPWM0_SYNC_OUT / TM0_EXT / LPTM0_EXT
8	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / USCI0_DAT0 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
9	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / EQEI1_A / ECAP0_IC1 / TM2_EXT
10	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / USCI0_CTL1 / UART1_RXD / EQEI1_B / ECAP0_IC2 / TM3_EXT / INT4
11	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
12	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
13	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
14	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
15	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / ACMP0_WLAT / TM2 / INT1 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
16	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / ACMP1_WLAT / TM3 / INT0 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
17	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / EPWM0_CH0 / EQEI0_INDEX / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
18	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / EPWM0_CH1 / EQEI0_A / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
19	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / EQEI0_B / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / UTCPD0_VBSNKEN / LPSPI0_SS
20	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / EPWM0_CH3 / UTCPD0_VBSRCEN / LPSPI0_CLK
21	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / EQEI0_INDEX / EPWM0_CH4 / ACMP2_O / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
22	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / EPWM0_BRAKE0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
23	P	V_{DDIO}
24	I	nRESET
25	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
26	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / EPWM1_CH5 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
27	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
28	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / UTCPD0_FRSTX1 / UTCPD0_DISCHG

Pin	Type	M2L31LD4AE Pin Function
29	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / UTCPD0_CCDB2
30	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / UTCPD0_CCDB1
31	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
32	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
33	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI0_SS / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
34	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI0_CLK / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
35	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI0_MISO / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
36	I/O	PA.15 / UART0_RXD / SPI0_MOSI / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
37	P	V_{SS}
38	A	LDO_CAP
39	P	V_{DD}
40	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
41	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
42	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
43	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
44	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
45	P	AV_{DD}
46	P	AV_{SS}
47	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / EBI_nWRL / UART1_TXD / EBI_nCS0 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / ACMP0_O
48	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / EBI_nWRH / UART1_RXD / EBI_nCS1 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / ACMP1_O

Table 4.3-10 M2L31LD4AE Multi-function Pin Table

4.3.3.3 M2L31LIDAE

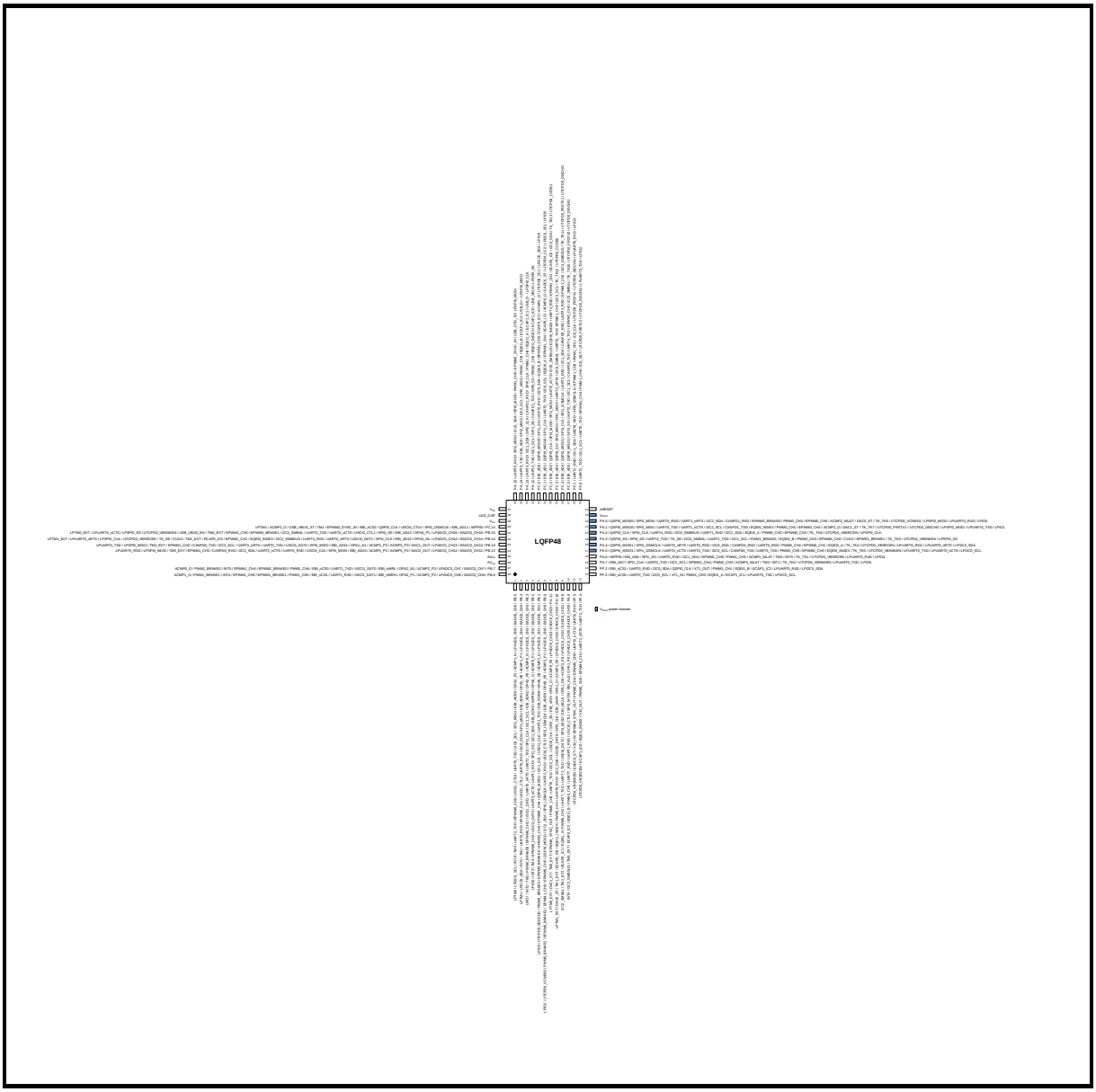


Figure 4.3-11 M2L31LIDAE Multi-function Pin Diagram

Pin	Type	M2L31LIDAE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / USCI1_CTL0 / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / OPA2_N0 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / USCI1_CTL1 / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / OPA2_P0 / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / USCI1_DAT1 / EPWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / USCI1_DAT0 / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / PWM0_BRAKE0 /

Pin	Type	M2L31LIDAE Pin Function
		UTCPD0_VBDCHG / LPIO3
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / SPI3_I2SMCLK / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / PWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / OPA2_O / EBI_nRD / SPI3_SS / USCI0_CLK / I2C2_SCL / UART6_TXD / PWM0_CH0 / EPWM0_SYNC_OUT / TM0_EXT / DAC1_ST / LPTM0_EXT
8	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / SPI3_CLK / USCI0_DAT0 / I2C2_SDA / UART6_RXD / PWM0_CH1 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
9	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / SPI3_MISO / USCI0_DAT1 / UART1_TXD / UART7_TXD / PWM0_CH2 / EQEI1_A / ECAP0_IC1 / TM2_EXT / I2C2_SMBAL
10	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / SPI3_MOSI / USCI0_CTL1 / UART1_RXD / UART7_RXD / PWM0_CH3 / EQEI1_B / ECAP0_IC2 / TM3_EXT / I2C2_SMBSUS / INT4
11	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / PWM0_CH4 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
12	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / PWM0_CH5 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
13	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / PWM1_CH0 / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
14	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / PWM1_CH1 / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
15	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / PWM1_CH2 / ACMP0_WLAT / TM2 / INT1 / TK_TK0 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
16	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / PWM1_CH3 / ACMP1_WLAT / TM3 / INT0 / TK_TK1 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
17	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / PWM0_CH5 / EPWM0_CH0 / EQEI0_INDEX / TK_TK3 / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
18	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / PWM0_CH4 / EPWM0_CH1 / EQEI0_A / TK_TK4 / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
19	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / TK_SE / I2C0_SMBAL / UART1_TXD / I2C1_SCL / PWM1_BRAKE1 / EQEI0_B / PWM0_CH3 / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / TK_TK5 / UTCPD0_VBSNKEN / LPSPI0_SS
20	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / PWM0_CH2 / EPWM0_CH3 / TK_TK6 / UTCPD0_VBSRCEN / LPSPI0_CLK
21	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / I2C2_SCL / CANFD1_TXD / EQEI0_INDEX / PWM0_CH1 / EPWM0_CH4 / ACMP2_O / DAC1_ST / TK_TK7 / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
22	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / I2C2_SDA / CANFD1_RXD / EPWM0_BRAKE0 / PWM0_CH0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / TK_TK8 / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
23	P	V_{DDIO}
24	I	nRESET
25	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / PWM1_CH0 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2

Pin	Type	M2L31LIDAE Pin Function
26	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / SPI3_I2SMCLK / EPWM1_CH5 / PWM1_CH1 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
27	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / SPI3_SS / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / I2C3_SMBAL / TK_TK10 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
28	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI3_CLK / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / I2C3_SMBSUS / TK_TK11 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
29	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI3_MISO / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / I2C3_SCL / TK_TK12 / UTCPD0_CCDB2
30	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI3_MOSI / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / I2C3_SDA / TK_TK13 / UTCPD0_CCDB1
31	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
32	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
33	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI2_SS / CANFD1_TXD / SPI0_SS / PWM1_CH2 / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
34	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI2_CLK / CANFD1_RXD / SPI0_CLK / PWM1_CH3 / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
35	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI2_MISO / I2C2_SCL / SPI0_MISO / PWM1_CH4 / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
36	I/O	PA.15 / UART0_RXD / SPI2_MOSI / I2C2_SDA / SPI0_MOSI / PWM1_CH5 / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
37	P	V_{SS}
38	A	LDO_CAP
39	P	V_{DD}
40	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
41	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / I2C2_SMBAL / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
42	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / I2C2_SMBSUS / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / TK_SE / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
43	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / DAC1_OUT / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / I2C2_SCL / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
44	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / I2C2_SDA / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
45	P	AV_{DD}
46	P	AV_{SS}
47	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / OPA2_N1 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / PWM1_CH4 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / PWM1_BRAKE0 / ACMP0_O

Pin	Type	M2L31LIDAE Pin Function
48	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / OPA2_P1 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / PWM1_CH5 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / PWM1_BRAKE1 / ACMP1_O

Table 4.3-11 M2L31LIDAE Multi-function Pin Table

4.3.3.4 M2L31LGDAE

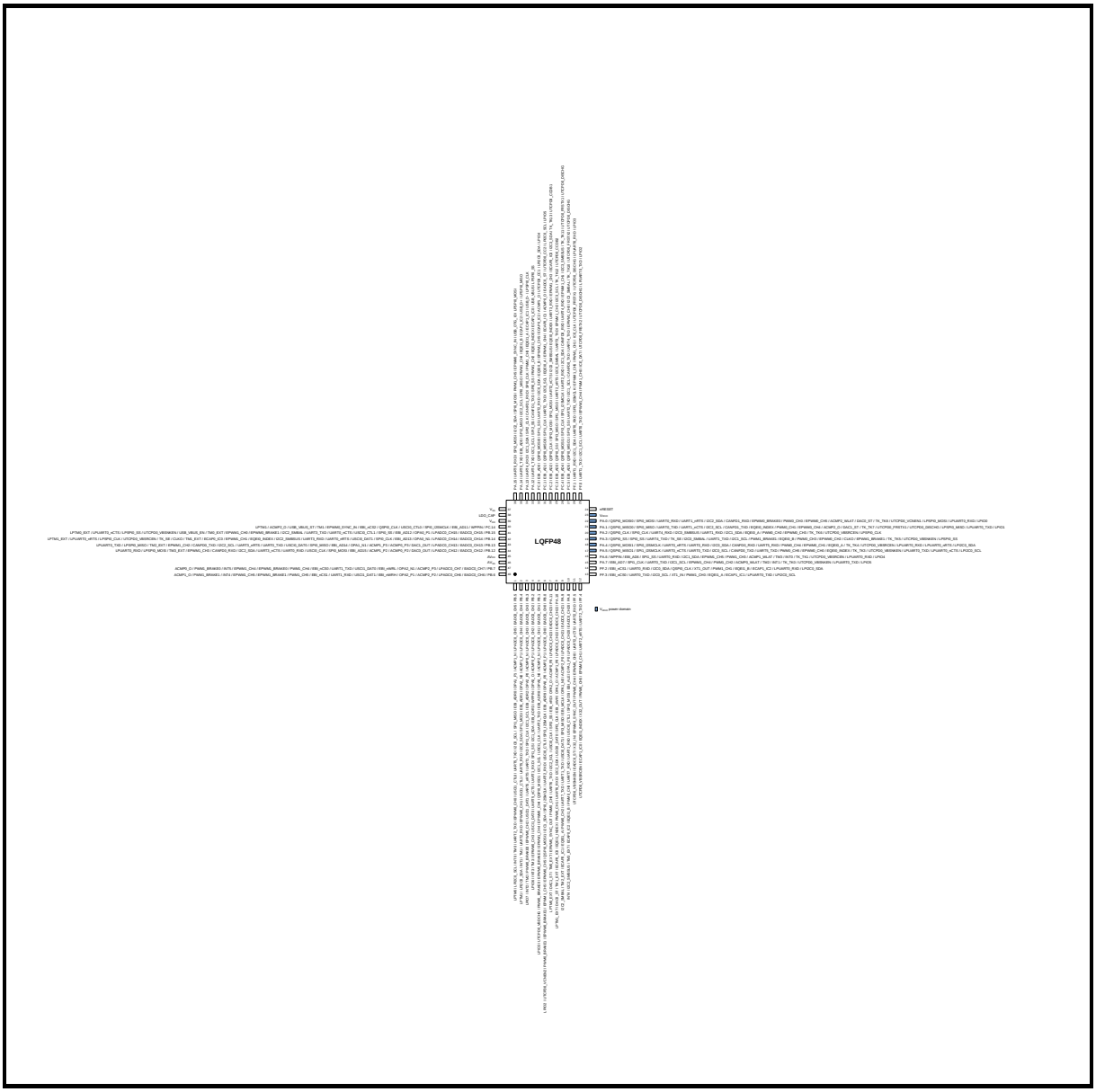


Figure 4.3-12 M2L31LGDAE Multi-function Pin Diagram

Pin	Type	M2L31LGDAE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / USCI1_CTL0 / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / OPA2_N0 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / USCI1_CTL1 / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / OPA2_P0 / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / USCI1_DAT1 / EPWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / USCI1_DAT0 / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / PWM0_BRAKE0 /

Pin	Type	M2L31LGDAE Pin Function
		UTCPD0_VBDCHG / LPIO3
6	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / SPI3_I2SMCLK / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / PWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
7	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / OPA2_O / EBI_nRD / SPI3_SS / USCI0_CLK / I2C2_SCL / UART6_TXD / PWM0_CH0 / EPWM0_SYNC_OUT / TM0_EXT / DAC1_ST / LPTM0_EXT
8	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / SPI3_CLK / USCI0_DAT0 / I2C2_SDA / UART6_RXD / PWM0_CH1 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
9	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / SPI3_MISO / USCI0_DAT1 / UART1_TXD / UART7_TXD / PWM0_CH2 / EQEI1_A / ECAP0_IC1 / TM2_EXT / I2C2_SMBAL
10	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / SPI3_MOSI / USCI0_CTL1 / UART1_RXD / UART7_RXD / PWM0_CH3 / EQEI1_B / ECAP0_IC2 / TM3_EXT / I2C2_SMBSUS / INT4
11	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / PWM0_CH4 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
12	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / PWM0_CH5 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
13	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / PWM1_CH0 / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
14	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / PWM1_CH1 / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
15	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / PWM1_CH2 / ACMP0_WLAT / TM2 / INT1 / TK_TK0 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
16	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / PWM1_CH3 / ACMP1_WLAT / TM3 / INT0 / TK_TK1 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
17	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / PWM0_CH5 / EPWM0_CH0 / EQEI0_INDEX / TK_TK3 / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
18	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / PWM0_CH4 / EPWM0_CH1 / EQEI0_A / TK_TK4 / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
19	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / TK_SE / I2C0_SMBAL / UART1_TXD / I2C1_SCL / PWM1_BRAKE1 / EQEI0_B / PWM0_CH3 / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / TK_TK5 / UTCPD0_VBSNKEN / LPSPI0_SS
20	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / PWM0_CH2 / EPWM0_CH3 / TK_TK6 / UTCPD0_VBSRCEN / LPSPI0_CLK
21	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / I2C2_SCL / CANFD1_TXD / EQEI0_INDEX / PWM0_CH1 / EPWM0_CH4 / ACMP2_O / DAC1_ST / TK_TK7 / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
22	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / I2C2_SDA / CANFD1_RXD / EPWM0_BRAKE0 / PWM0_CH0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / TK_TK8 / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
23	P	V_{DDIO}
24	I	nRESET
25	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / PWM1_CH0 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2

Pin	Type	M2L31LGDAE Pin Function
26	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / SPI3_I2SMCLK / EPWM1_CH5 / PWM1_CH1 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
27	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / SPI3_SS / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / I2C3_SMBAL / TK_TK10 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
28	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI3_CLK / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / I2C3_SMBSUS / TK_TK11 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
29	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI3_MISO / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / I2C3_SCL / TK_TK12 / UTCPD0_CCDB2
30	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI3_MOSI / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / I2C3_SDA / TK_TK13 / UTCPD0_CCDB1
31	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
32	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
33	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI2_SS / CANFD1_TXD / SPI0_SS / PWM1_CH2 / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
34	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI2_CLK / CANFD1_RXD / SPI0_CLK / PWM1_CH3 / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
35	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI2_MISO / I2C2_SCL / SPI0_MISO / PWM1_CH4 / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
36	I/O	PA.15 / UART0_RXD / SPI2_MOSI / I2C2_SDA / SPI0_MOSI / PWM1_CH5 / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
37	P	V_{SS}
38	A	LDO_CAP
39	P	V_{DD}
40	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
41	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / I2C2_SMBAL / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
42	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / I2C2_SMBSUS / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / TK_SE / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
43	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / DAC1_OUT / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / I2C2_SCL / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
44	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / I2C2_SDA / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
45	P	AV_{DD}
46	P	AV_{SS}
47	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / OPA2_N1 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / PWM1_CH4 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / PWM1_BRAKE0 / ACMP0_O

Pin	Type	M2L31LGDAE Pin Function
48	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / OPA2_P1 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / PWM1_CH5 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / PWM1_BRAKE1 / ACMP1_O

Table 4.3-12 M2L31LGDAE Multi-function Pin Table

4.3.4 LQFP64-Pin Multi-function Diagram

Corresponding Part Number: M2L31SG4AE, M2L31SE4AE, M2L31SIDAE, M2L31SGDAE

4.3.4.1 M2L31SG4AE

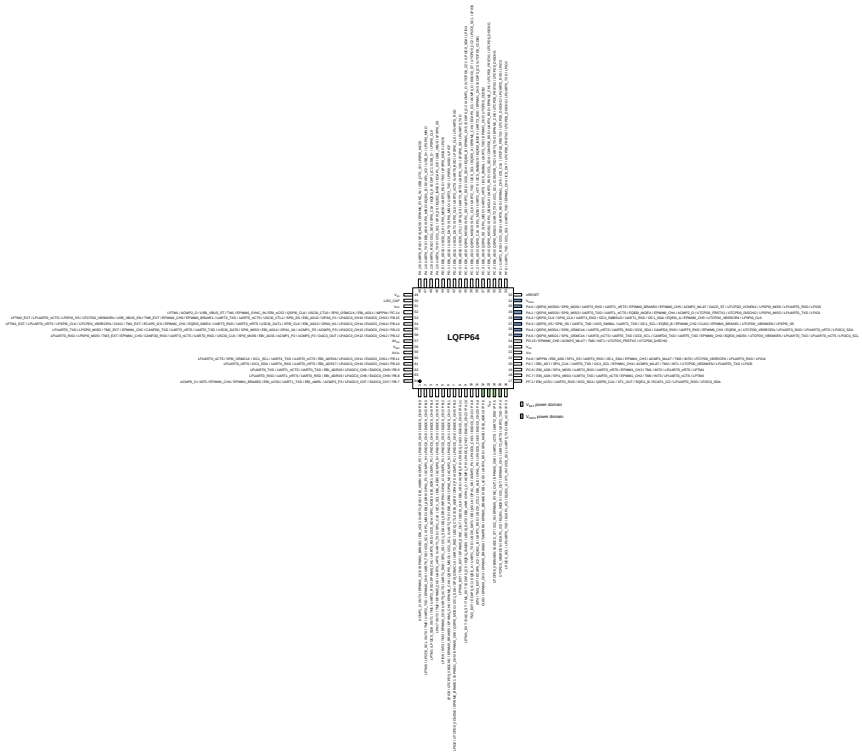


Figure 4.3-13 M2L31SG4AE Multi-function Pin Diagram

Pin	Type	M2L31SG4AE Pin Function
1	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / EBI_nWRH / UART1_RXD / EBI_nCS1 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / ACMP1_O
2	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
3	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1

Pin	Type	M2L31SG4AE Pin Function
4	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / EPWM0_CH2 / TM2 / INT2 / LPIO7
5	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / EPWM0_CH3 / TM3 / INT3 / LPIO6
6	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
7	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
8	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / EBI_nRD / USCI0_CLK / EPWM0_SYNC_OUT / TM0_EXT / LPTM0_EXT
9	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / USCI0_DAT0 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
10	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / EQEI1_A / ECAP0_IC1 / TM2_EXT
11	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / USCI0_CTL1 / UART1_RXD / EQEI1_B / ECAP0_IC2 / TM3_EXT / INT4
12	I/O	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0 / EPWM1_BRAKE0 / TAMPER0 / EPWM0_BRAKE0 / EPWM0_CH4 / CLKO
13	P	V_{BAT}
14	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
15	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
16	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
17	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
18	I/O	PC.7 / EBI_AD9 / SPI1_MISO / UART4_TXD / UART0_nCTS / EPWM1_CH2 / TM0 / INT3 / LPUART0_nCTS / LPTM0
19	I/O	PC.6 / EBI_AD8 / SPI1_MOSI / UART4_RXD / UART0_nRTS / EPWM1_CH3 / TM1 / INT2 / LPUART0_nRTS / LPTM1
20	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / ACMP0_WLAT / TM2 / INT1 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
21	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / ACMP1_WLAT / TM3 / INT0 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
22	P	V_{SS}
23	P	V_{DD}
24	I/O	PD.15 / EPWM0_CH5 / ACMP2_WLAT / TM3 / INT1 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
25	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / EPWM0_CH0 / EQEI0_INDEX / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
26	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / EPWM0_CH1 / EQEI0_A / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA

Pin	Type	M2L31SG4AE Pin Function
27	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / EQEI0_B / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / UTCPD0_VBSNKEN / LPSPI0_SS
28	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / EPWM0_CH3 / UTCPD0_VBSRCEN / LPSPI0_CLK
29	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / EQEI0_INDEX / EPWM0_CH4 / ACMP2_O / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
30	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / EPWM0_BRAKE0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
31	P	V_{DDIO}
32	I	nRESET
33	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
34	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / EPWM1_CH5 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
35	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
36	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
37	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / UTCPD0_CCDB2
38	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / UTCPD0_CCDB1
39	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
40	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
41	I/O	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / UART0_TXD / LPSPI0_SS / LPUART0_TXD
42	I/O	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD / LPSPI0_CLK / LPUART0_RXD
43	I/O	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD / LPSPI0_MISO / LPIO7
44	I/O	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2 / LPSPI0_MOSI / LPIO6
45	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI0_SS / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
46	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI0_CLK / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
47	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI0_MISO / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
48	I/O	PA.15 / UART0_RXD / SPI0_MOSI / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
49	P	V_{SS}
50	A	LDO_CAP
51	P	V_{DD}
52	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
53	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 /

Pin	Type	M2L31SG4AE Pin Function
		UART0_nCTS / UART3_TXD / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
54	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
55	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
56	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
57	P	AV_{DD}
58	A	V_{REF}
59	P	AV_{SS}
60	I/O	PB.11 / EADC0_CH11 / LPADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / LPUART0_nCTS
61	I/O	PB.10 / EADC0_CH10 / LPADC0_CH10 / EBI_ADR17 / UART0_nRTS / UART4_RXD / I2C1_SDA / LPUART0_nRTS
62	I/O	PB.9 / EADC0_CH9 / LPADC0_CH9 / EBI_ADR18 / UART0_TXD / UART1_nCTS / LPUART0_TXD
63	I/O	PB.8 / EADC0_CH8 / LPADC0_CH8 / EBI_ADR19 / UART0_RXD / UART1_nRTS / LPUART0_RXD
64	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / EBI_nWRL / UART1_TXD / EBI_nCS0 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / ACMP0_O

Table 4.3-13 M2L31SG4AE Multi-function Pin Table

4.3.4.2 M2L31SE4AE

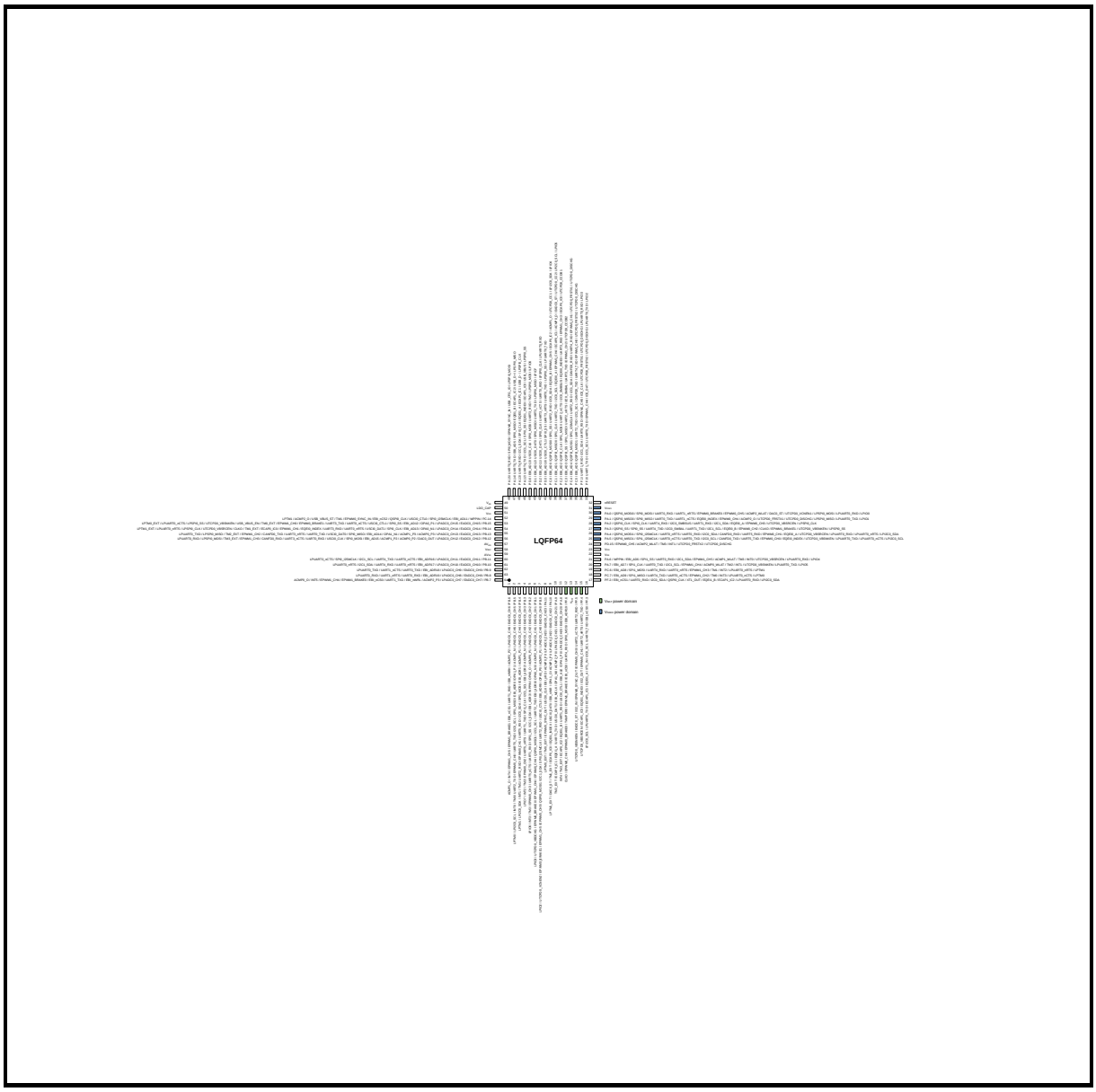


Figure 4.3-14 M2L31SE4AE Multi-function Pin Diagram

Pin	Type	M2L31SE4AE Pin Function
1	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / EBI_nWRH / UART1_RXD / EBI_nCS1 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / ACMP1_O
2	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
3	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
4	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / EPWM0_CH2 / TM2 / INT2 / LPIO7

Pin	Type	M2L31SE4AE Pin Function
5	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / EPWM0_CH3 / TM3 / INT3 / LPIO6
6	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
7	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
8	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / EBI_nRD / USCI0_CLK / EPWM0_SYNC_OUT / TM0_EXT / LPTM0_EXT
9	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / USCI0_DAT0 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
10	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / EQEI1_A / ECAP0_IC1 / TM2_EXT
11	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / USCI0_CTL1 / UART1_RXD / EQEI1_B / ECAP0_IC2 / TM3_EXT / INT4
12	I/O	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0 / EPWM1_BRAKE0 / TAMPER0 / EPWM0_BRAKE0 / EPWM0_CH4 / CLKO
13	P	V_{BAT}
14	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
15	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
16	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
17	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
18	I/O	PC.7 / EBI_AD9 / SPI1_MISO / UART4_TXD / UART0_nCTS / EPWM1_CH2 / TM0 / INT3 / LPUART0_nCTS / LPTM0
19	I/O	PC.6 / EBI_AD8 / SPI1_MOSI / UART4_RXD / UART0_nRTS / EPWM1_CH3 / TM1 / INT2 / LPUART0_nRTS / LPTM1
20	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / ACMP0_WLAT / TM2 / INT1 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
21	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / ACMP1_WLAT / TM3 / INT0 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
22	P	V_{SS}
23	P	V_{DD}
24	I/O	PD.15 / EPWM0_CH5 / ACMP2_WLAT / TM3 / INT1 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
25	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / EPWM0_CH0 / EQEI0_INDEX / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
26	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / EPWM0_CH1 / EQEI0_A / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
27	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / EQEI0_B / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / UTCPD0_VBSNKEN / LPSPI0_SS

Pin	Type	M2L31SE4AE Pin Function
28	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / EPWM0_CH3 / UTCPD0_VBSRCEN / LPSPI0_CLK
29	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / EQEI0_INDEX / EPWM0_CH4 / ACMP2_O / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
30	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / EPWM0_BRAKE0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
31	P	V_{DDIO}
32	I	nRESET
33	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
34	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / EPWM1_CH5 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
35	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
36	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
37	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / UTCPD0_CCDB2
38	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / UTCPD0_CCDB1
39	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
40	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
41	I/O	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / UART0_TXD / LPSPI0_SS / LPUART0_TXD
42	I/O	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD / LPSPI0_CLK / LPUART0_RXD
43	I/O	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD / LPSPI0_MISO / LPIO7
44	I/O	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2 / LPSPI0_MOSI / LPIO6
45	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI0_SS / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
46	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI0_CLK / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
47	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI0_MISO / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
48	I/O	PA.15 / UART0_RXD / SPI0_MOSI / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
49	P	V_{SS}
50	A	LDO_CAP
51	P	V_{DD}
52	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
53	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT

Pin	Type	M2L31SE4AE Pin Function
54	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
55	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
56	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
57	P	AV_{DD}
58	A	V_{REF}
59	P	AV_{SS}
60	I/O	PB.11 / EADC0_CH11 / LPADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / LPUART0_nCTS
61	I/O	PB.10 / EADC0_CH10 / LPADC0_CH10 / EBI_ADR17 / UART0_nRTS / UART4_RXD / I2C1_SDA / LPUART0_nRTS
62	I/O	PB.9 / EADC0_CH9 / LPADC0_CH9 / EBI_ADR18 / UART0_TXD / UART1_nCTS / LPUART0_TXD
63	I/O	PB.8 / EADC0_CH8 / LPADC0_CH8 / EBI_ADR19 / UART0_RXD / UART1_nRTS / LPUART0_RXD
64	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / EBI_nWRL / UART1_TXD / EBI_nCS0 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / ACMP0_O

Table 4.3-14 M2L31SE4AE Multi-function Pin Table

4.3.4.3 M2L31SIDAE

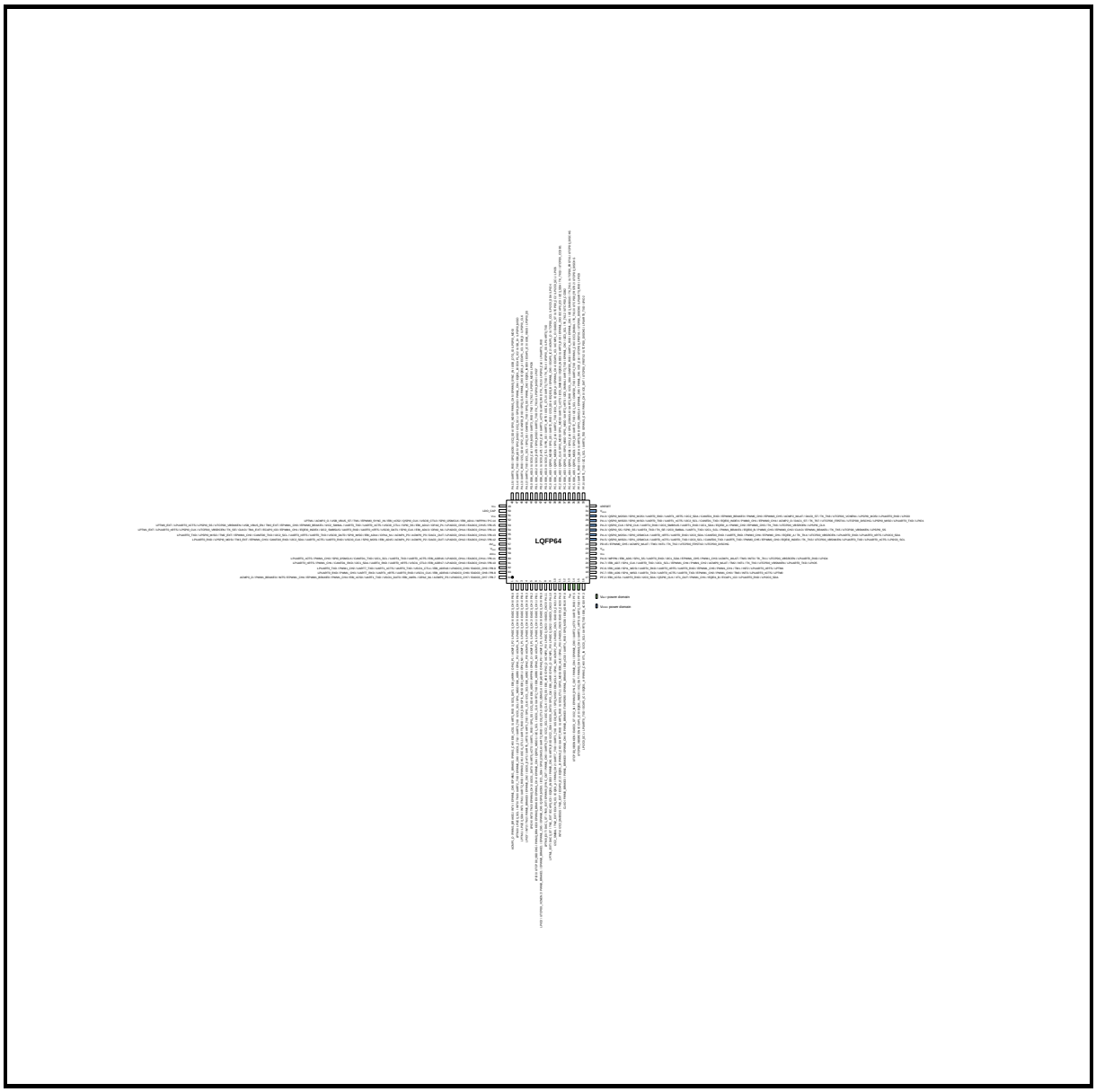


Figure 4.3-15 M2L31SIDAE Multi-function Pin Diagram

Pin	Type	M2L31SIDAE Pin Function
1	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / OPA2_P1 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / PWM1_CH5 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / PWM1_BRAKE1 / ACMP1_O
2	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / USCI1_CTL0 / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
3	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / OPA2_N0 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / USCI1_CTL1 / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
4	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / OPA2_P0 / EBI_ADR2 / I2C1_SCL / SPI1_CLK /

Pin	Type	M2L31SIDAE Pin Function
		UART1_TXD / UART5_nRTS / USCI1_DAT1 / EPWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2 / LPIO7
5	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / USCI1_DAT0 / EPWM0_CH3 / TM3 / INT3 / LPIO6
6	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / PWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
7	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / SPI3_I2SMCLK / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / PWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
8	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / OPA2_O / EBI_nRD / SPI3_SS / USCI0_CLK / I2C2_SCL / UART6_TXD / PWM0_CH0 / EPWM0_SYNC_OUT / TM0_EXT / DAC1_ST / LPTM0_EXT
9	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / SPI3_CLK / USCI0_DAT0 / I2C2_SDA / UART6_RXD / PWM0_CH1 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
10	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / SPI3_MISO / USCI0_DAT1 / UART1_TXD / UART7_TXD / PWM0_CH2 / EQEI1_A / ECAP0_IC1 / TM2_EXT / I2C2_SMBAL
11	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / SPI3_MOSI / USCI0_CTL1 / UART1_RXD / UART7_RXD / PWM0_CH3 / EQEI1_B / ECAP0_IC2 / TM3_EXT / I2C2_SMBSUS / INT4
12	I/O	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0 / EPWM1_BRAKE0 / TAMPER0 / EPWM0_BRAKE0 / EPWM0_CH4 / PWM1_BRAKE0 / PWM0_BRAKE0 / CLKO
13	P	V_{BAT}
14	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / PWM0_CH4 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
15	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / PWM0_CH5 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
16	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / PWM1_CH0 / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
17	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / PWM1_CH1 / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
18	I/O	PC.7 / EBI_AD9 / SPI1_MISO / UART4_TXD / UART0_nCTS / UART6_TXD / EPWM1_CH2 / PWM1_CH0 / TM0 / INT3 / LPUART0_nCTS / LPTM0
19	I/O	PC.6 / EBI_AD8 / SPI1_MOSI / UART4_RXD / UART0_nRTS / UART6_RXD / EPWM1_CH3 / PWM1_CH1 / TM1 / INT2 / LPUART0_nRTS / LPTM1
20	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / PWM1_CH2 / ACMP0_WLAT / TM2 / INT1 / TK_TK0 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
21	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / PWM1_CH3 / ACMP1_WLAT / TM3 / INT0 / TK_TK1 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
22	P	V_{SS}
23	P	V_{DD}
24	I/O	PD.15 / EPWM0_CH5 / ACMP2_WLAT / TM3 / INT1 / TK_TK2 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
25	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / PWM0_CH5 / EPWM0_CH0 / EQEI0_INDEX / TK_TK3 / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL

Pin	Type	M2L31SIDAE Pin Function
26	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / PWM0_CH4 / EPWM0_CH1 / EQEI0_A / TK_TK4 / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
27	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / TK_SE / I2C0_SMBAL / UART1_TXD / I2C1_SCL / PWM1_BRAKE1 / EQEI0_B / PWM0_CH3 / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / TK_TK5 / UTCPD0_VBSNKEN / LPSPI0_SS
28	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / PWM0_CH2 / EPWM0_CH3 / TK_TK6 / UTCPD0_VBSRCEN / LPSPI0_CLK
29	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / I2C2_SCL / CANFD1_TXD / EQEI0_INDEX / PWM0_CH1 / EPWM0_CH4 / ACMP2_O / DAC1_ST / TK_TK7 / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
30	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / I2C2_SDA / CANFD1_RXD / EPWM0_BRAKE0 / PWM0_CH0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / TK_TK8 / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
31	P	V_{DDIO}
32	I	nRESET
33	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / PWM1_CH0 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
34	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / SPI3_I2SMCLK / EPWM1_CH5 / PWM1_CH1 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
35	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / SPI3_SS / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / I2C3_SMBAL / TK_TK10 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
36	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI3_CLK / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / I2C3_SMBSUS / TK_TK11 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
37	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI3_MISO / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / I2C3_SCL / TK_TK12 / UTCPD0_CCDB2
38	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI3_MOSI / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / I2C3_SDA / TK_TK13 / UTCPD0_CCDB1
39	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
40	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
41	I/O	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD / TK_TK14 / LPSPI0_SS / LPUART0_TXD
42	I/O	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD / TK_TK15 / LPSPI0_CLK / LPUART0_RXD
43	I/O	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD / TK_TK16 / LPSPI0_MISO / LPIO7
44	I/O	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2 / TK_TK17 / LPSPI0_MOSI / LPIO6
45	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI2_SS / CANFD1_TXD / SPI0_SS / PWM1_CH2 / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
46	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI2_CLK / CANFD1_RXD / SPI0_CLK / PWM1_CH3 / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
47	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI2_MISO / I2C2_SCL / SPI0_MISO / PWM1_CH4 / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO

Pin	Type	M2L31SIDAE Pin Function
48	I/O	PA.15 / UART0_RXD / SPI2_MOSI / I2C2_SDA / SPI0_MOSI / PWM1_CH5 / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
49	P	V_{SS}
50	A	LDO_CAP
51	P	V_{DD}
52	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
53	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / I2C2_SMBAL / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
54	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / I2C2_SMBSUS / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / TK_SE / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
55	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / DAC1_OUT / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / I2C2_SCL / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
56	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / I2C2_SDA / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
57	P	AV_{DD}
58	A	V_{REF}
59	P	AV_{SS}
60	I/O	PB.11 / EADC0_CH11 / LPADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / CANFD1_TXD / SPI0_I2SMCLK / PWM1_CH0 / LPUART0_nCTS
61	I/O	PB.10 / EADC0_CH10 / LPADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / CANFD1_RXD / PWM1_CH1 / LPUART0_nRTS
62	I/O	PB.9 / EADC0_CH9 / LPADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / UART7_TXD / PWM1_CH2 / LPUART0_TXD
63	I/O	PB.8 / EADC0_CH8 / LPADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / UART7_RXD / PWM1_CH3 / LPUART0_RXD
64	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / OPA2_N1 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / PWM1_CH4 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / PWM1_BRAKE0 / ACMP0_O

Table 4.3-15 M2L31SIDAE Multi-function Pin Table

4.3.4.4 M2L31SGDAE

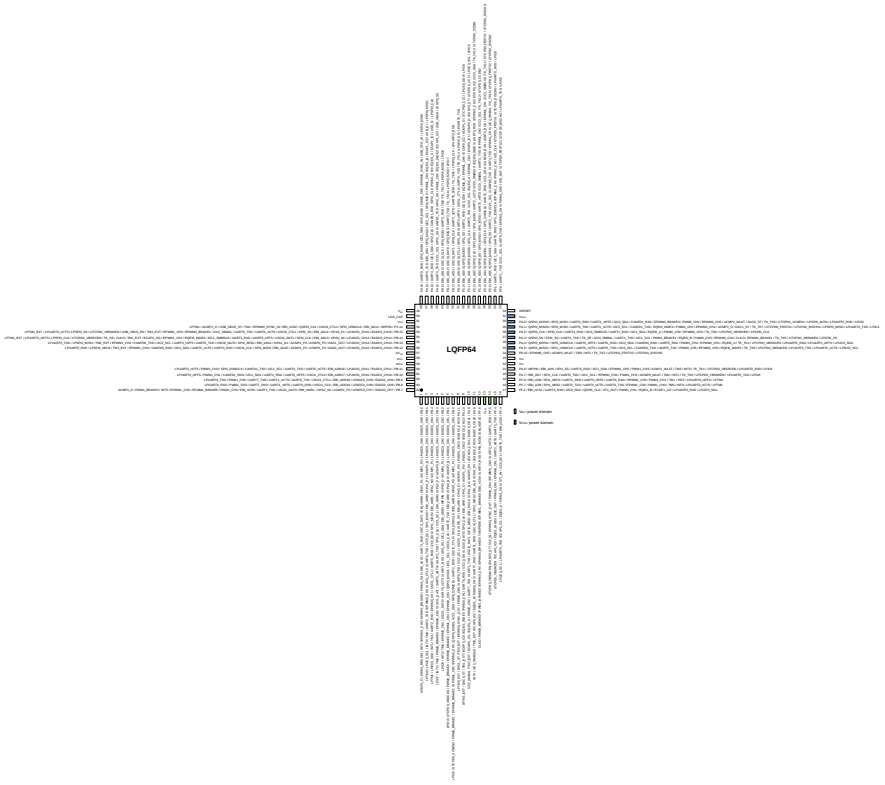


Figure 4.3-16 M2L31SGDAE Multi-function Pin Diagram

Pin	Type	M2L31SGDAE Pin Function
1	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / OPA2_P1 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / PWM1_CH5 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / PWM1_BRAKE1 / ACMP1_O
2	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / USCI1_CTL0 / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
3	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / OPA2_N0 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / USCI1_CTL1 / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
4	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / OPA2_P0 / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / USCI1_DAT1 / EPWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2 / LPIO7
5	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA /

Pin	Type	M2L31SGDAE Pin Function
		SPI1_SS / UART1_RXD / UART5_nCTS / USCI1_DAT0 / EPWM0_CH3 / TM3 / INT3 / LPIO6
6	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / PWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
7	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / SPI3_I2SMCLK / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / PWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
8	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / OPA2_O / EBI_nRD / SPI3_SS / USCI0_CLK / I2C2_SCL / UART6_TXD / PWM0_CH0 / EPWM0_SYNC_OUT / TM0_EXT / DAC1_ST / LPTM0_EXT
9	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / SPI3_CLK / USCI0_DAT0 / I2C2_SDA / UART6_RXD / PWM0_CH1 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
10	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / SPI3_MISO / USCI0_DAT1 / UART1_TXD / UART7_TXD / PWM0_CH2 / EQEI1_A / ECAP0_IC1 / TM2_EXT / I2C2_SMBAL
11	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / SPI3_MOSI / USCI0_CTL1 / UART1_RXD / UART7_RXD / PWM0_CH3 / EQEI1_B / ECAP0_IC2 / TM3_EXT / I2C2_SMBSUS / INT4
12	I/O	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0 / EPWM1_BRAKE0 / TAMPER0 / EPWM0_BRAKE0 / EPWM0_CH4 / PWM1_BRAKE0 / PWM0_BRAKE0 / CLKO
13	P	V_{BAT}
14	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / PWM0_CH4 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
15	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / PWM0_CH5 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
16	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / PWM1_CH0 / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
17	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / PWM1_CH1 / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
18	I/O	PC.7 / EBI_AD9 / SPI1_MISO / UART4_TXD / UART0_nCTS / UART6_TXD / EPWM1_CH2 / PWM1_CH0 / TM0 / INT3 / LPUART0_nCTS / LPTM0
19	I/O	PC.6 / EBI_AD8 / SPI1_MOSI / UART4_RXD / UART0_nRTS / UART6_RXD / EPWM1_CH3 / PWM1_CH1 / TM1 / INT2 / LPUART0_nRTS / LPTM1
20	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / PWM1_CH2 / ACMP0_WLAT / TM2 / INT1 / TK_TK0 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
21	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / PWM1_CH3 / ACMP1_WLAT / TM3 / INT0 / TK_TK1 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
22	P	V_{SS}
23	P	V_{DD}
24	I/O	PD.15 / EPWM0_CH5 / ACMP2_WLAT / TM3 / INT1 / TK_TK2 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
25	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / PWM0_CH5 / EPWM0_CH0 / EQEI0_INDEX / TK_TK3 / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
26	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / PWM0_CH4 / EPWM0_CH1 / EQEI0_A / TK_TK4 / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA

Pin	Type	M2L31SGDAE Pin Function
27	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / TK_SE / I2C0_SMBAL / UART1_TXD / I2C1_SCL / PWM1_BRAKE1 / EQEI0_B / PWM0_CH3 / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / TK_TK5 / UTCPD0_VBSNKEN / LPSPI0_SS
28	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / PWM0_CH2 / EPWM0_CH3 / TK_TK6 / UTCPD0_VBSRCEN / LPSPI0_CLK
29	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / I2C2_SCL / CANFD1_TXD / EQEI0_INDEX / PWM0_CH1 / EPWM0_CH4 / ACMP2_O / DAC1_ST / TK_TK7 / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
30	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / I2C2_SDA / CANFD1_RXD / EPWM0_BRAKE0 / PWM0_CH0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / TK_TK8 / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
31	P	V_{DDIO}
32	I	nRESET
33	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / PWM1_CH0 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
34	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / SPI3_I2SMCLK / EPWM1_CH5 / PWM1_CH1 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
35	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / SPI3_SS / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / I2C3_SMBAL / TK_TK10 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
36	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI3_CLK / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / I2C3_SMBSUS / TK_TK11 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
37	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI3_MISO / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / I2C3_SCL / TK_TK12 / UTCPD0_CCDB2
38	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI3_MOSI / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / I2C3_SDA / TK_TK13 / UTCPD0_CCDB1
39	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
40	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
41	I/O	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD / TK_TK14 / LPSPI0_SS / LPUART0_TXD
42	I/O	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD / TK_TK15 / LPSPI0_CLK / LPUART0_RXD
43	I/O	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD / TK_TK16 / LPSPI0_MISO / LPIO7
44	I/O	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2 / TK_TK17 / LPSPI0_MOSI / LPIO6
45	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI2_SS / CANFD1_TXD / SPI0_SS / PWM1_CH2 / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
46	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI2_CLK / CANFD1_RXD / SPI0_CLK / PWM1_CH3 / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
47	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI2_MISO / I2C2_SCL / SPI0_MISO / PWM1_CH4 / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
48	I/O	PA.15 / UART0_RXD / SPI2_MOSI / I2C2_SDA / SPI0_MOSI / PWM1_CH5 / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
49	P	V_{SS}

Pin	Type	M2L31SGDAE Pin Function
50	A	LDO_CAP
51	P	V_{DD}
52	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
53	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / I2C2_SMBAL / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
54	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / I2C2_SMBSUS / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / TK_SE / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
55	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / DAC1_OUT / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / I2C2_SCL / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
56	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / I2C2_SDA / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
57	P	AV_{DD}
58	A	V_{REF}
59	P	AV_{SS}
60	I/O	PB.11 / EADC0_CH11 / LPADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / CANFD1_TXD / SPI0_I2SMCLK / PWM1_CH0 / LPUART0_nCTS
61	I/O	PB.10 / EADC0_CH10 / LPADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / CANFD1_RXD / PWM1_CH1 / LPUART0_nRTS
62	I/O	PB.9 / EADC0_CH9 / LPADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / UART7_TXD / PWM1_CH2 / LPUART0_TXD
63	I/O	PB.8 / EADC0_CH8 / LPADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / UART7_RXD / PWM1_CH3 / LPUART0_RXD
64	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / OPA2_N1 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / PWM1_CH4 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / PWM1_BRAKE0 / ACMP0_O

Table 4.3-16 M2L31SGDAE Multi-function Pin Table

4.3.5 LQFP128-Pin Multi-function Diagram

Corresponding Part Number: M2L31KIDAE, M2L31KGDAE

4.3.5.1 M2L31KIDAE

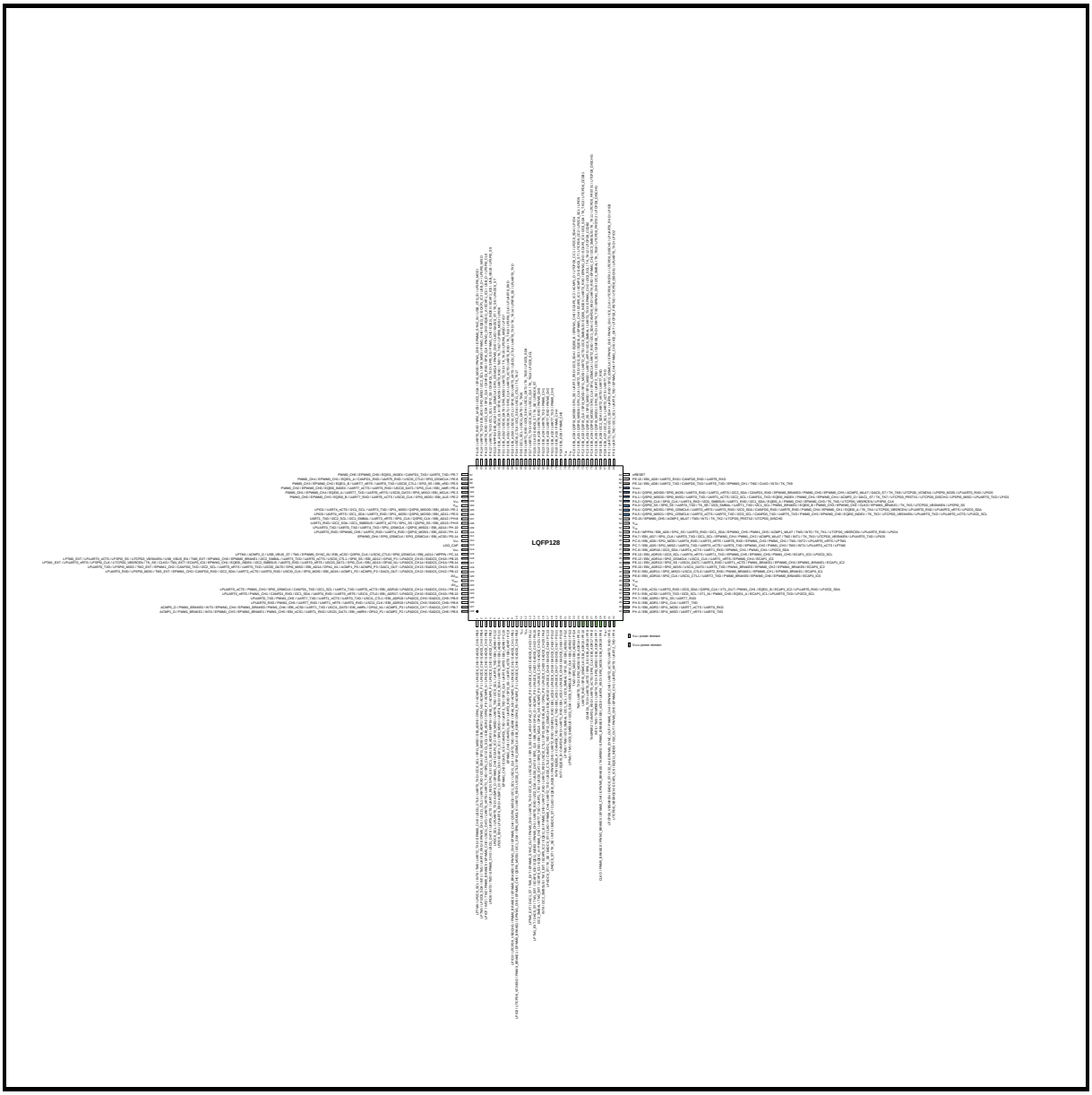


Figure 4.3-17 M2L31KIDAE Multi-function Pin Diagram

Pin	Type	M2L31KIDAE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / USCI1_CTL0 / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / OPA2_N0 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / USCI1_CTL1 / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / OPA2_P0 / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / USCI1_DAT1 / EPWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2 / LPIO7

Pin	Type	M2L31KIDAE Pin Function
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / USCI1_DAT0 / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PC.12 / EBI_ADR4 / UART0_TXD / I2C0_SCL / UART6_TXD / SPI3_MISO / ECAP1_IC2 / EPWM1_CH0 / ACMP0_O / LPUART0_TXD / LPI2C0_SCL
6	I/O	PC.11 / EBI_ADR5 / UART0_RXD / I2C0_SDA / UART6_RXD / SPI3_MOSI / ECAP1_IC1 / EPWM1_CH1 / ACMP1_O / LPUART0_RXD / LPI2C0_SDA
7	I/O	PC.10 / EBI_ADR6 / UART6_nRTS / SPI3_CLK / UART3_TXD / CANFD1_TXD / ECAP1_IC0 / EPWM1_CH2
8	I/O	PC.9 / EBI_ADR7 / UART6_nCTS / SPI3_SS / UART3_RXD / CANFD1_RXD / EPWM1_CH3
9	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / PWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
10	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / SPI3_I2SMCLK / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / PWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
11	P	V_{SS}
12	P	V_{DD}
13	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / OPA2_O / EBI_nRD / SPI3_SS / USCI0_CLK / I2C2_SCL / UART6_TXD / PWM0_CH0 / EPWM0_SYNC_OUT / TM0_EXT / DAC1_ST / LPTM0_EXT
14	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / SPI3_CLK / USCI0_DAT0 / I2C2_SDA / UART6_RXD / PWM0_CH1 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
15	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / SPI3_MISO / USCI0_DAT1 / UART1_TXD / UART7_TXD / PWM0_CH2 / EQEI1_A / ECAP0_IC1 / TM2_EXT / I2C2_SMBAL
16	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / SPI3_MOSI / USCI0_CTL1 / UART1_RXD / UART7_RXD / PWM0_CH3 / EQEI1_B / ECAP0_IC2 / TM3_EXT / I2C2_SMBSUS / INT4
17	I/O	PC.13 / EADC0_CH19 / LPADC0_CH19 / EBI_ADR10 / SPI2_I2SMCLK / CANFD1_TXD / USCI0_CTL0 / UART2_TXD / PWM0_CH4 / CLKO / EADC0_ST / TK_SE / LPADC0_ST
18	I/O	PD.12 / EADC0_CH18 / LPADC0_CH18 / EBI_nCS0 / CANFD1_RXD / UART2_RXD / PWM0_CH5 / EQEI0_INDEX / CLKO / EADC0_ST / INT5 / TK_SE / LPADC0_ST
19	I/O	PD.11 / EADC0_CH17 / LPADC0_CH17 / EBI_nCS1 / UART1_TXD / CANFD0_TXD / EQEI0_A / INT6
20	I/O	PD.10 / EADC0_CH16 / LPADC0_CH16 / EBI_nCS2 / UART1_RXD / CANFD0_RXD / EQEI0_B / INT7
21	I/O	PG.2 / EBI_ADR11 / SPI2_SS / I2C0_SMBAL / I2C1_SCL / I2C3_SMBAL / TM0 / LPTM0
22	I/O	PG.3 / EBI_ADR12 / SPI2_CLK / I2C0_SMBSUS / I2C1_SDA / I2C3_SMBSUS / TM1 / LPTM1
23	I/O	PG.4 / EBI_ADR13 / SPI2_MISO / TM2
24	I/O	PF.11 / EBI_ADR14 / SPI2_MOSI / UART5_TXD / TM3
25	I/O	PF.10 / EBI_ADR15 / SPI0_I2SMCLK / UART5_RXD
26	I/O	PF.9 / EBI_ADR16 / SPI0_SS / UART5_nRTS / CANFD1_TXD
27	I/O	PF.8 / EBI_ADR17 / SPI0_CLK / UART5_nCTS / CANFD1_RXD / TAMPER2
28	I/O	PF.7 / EBI_ADR18 / SPI0_MISO / UART4_TXD / TAMPER1 / TM3 / INT5

Pin	Type	M2L31KIDAE Pin Function
29	I/O	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0 / EPWM1_BRAKE0 / TAMPER0 / EPWM0_BRAKE0 / EPWM0_CH4 / PWM1_BRAKE0 / PWM0_BRAKE0 / CLKO
30	P	V_{BAT}
31	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / PWM0_CH4 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
32	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / PWM0_CH5 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
33	I/O	PH.4 / EBI_ADR3 / SPI1_MISO / UART7_nRTS / UART6_TXD
34	I/O	PH.5 / EBI_ADR2 / SPI1_MOSI / UART7_nCTS / UART6_RXD
35	I/O	PH.6 / EBI_ADR1 / SPI1_CLK / UART7_TXD
36	I/O	PH.7 / EBI_ADR0 / SPI1_SS / UART7_RXD
37	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / PWM1_CH0 / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
38	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / PWM1_CH1 / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
39	P	V_{SS}
40	P	V_{DD}
41	I/O	PE.8 / EBI_ADR10 / SPI2_CLK / USCI1_CTL1 / UART2_TXD / PWM0_BRAKE0 / EPWM0_CH0 / EPWM0_BRAKE0 / ECAP0_IC0
42	I/O	PE.9 / EBI_ADR11 / SPI2_MISO / USCI1_CTL0 / UART2_RXD / PWM0_BRAKE1 / EPWM0_CH1 / EPWM0_BRAKE1 / ECAP0_IC1
43	I/O	PE.10 / EBI_ADR12 / SPI2_MOSI / USCI1_DAT0 / UART3_TXD / PWM1_BRAKE0 / EPWM0_CH2 / EPWM1_BRAKE0 / ECAP0_IC2
44	I/O	PE.11 / EBI_ADR13 / SPI2_SS / USCI1_DAT1 / UART3_RXD / UART1_nCTS / PWM1_BRAKE1 / EPWM0_CH3 / EPWM1_BRAKE1 / ECAP1_IC2
45	I/O	PE.12 / EBI_ADR14 / SPI2_I2SMCLK / USCI1_CLK / UART1_nRTS / EPWM0_CH4 / ECAP1_IC1
46	I/O	PE.13 / EBI_ADR15 / I2C0_SCL / UART4_nRTS / UART1_TXD / EPWM0_CH5 / EPWM1_CH0 / PWM1_CH5 / ECAP1_IC0 / LPI2C0_SCL
47	I/O	PC.8 / EBI_ADR16 / I2C0_SDA / UART4_nCTS / UART1_RXD / EPWM1_CH1 / PWM1_CH4 / LPI2C0_SDA
48	I/O	PC.7 / EBI_AD9 / SPI1_MISO / UART4_TXD / UART0_nCTS / UART6_TXD / EPWM1_CH2 / PWM1_CH0 / TM0 / INT3 / LPUART0_nCTS / LPTM0
49	I/O	PC.6 / EBI_AD8 / SPI1_MOSI / UART4_RXD / UART0_nRTS / UART6_RXD / EPWM1_CH3 / PWM1_CH1 / TM1 / INT2 / LPUART0_nRTS / LPTM1
50	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / PWM1_CH2 / ACMP0_WLAT / TM2 / INT1 / TK_TK0 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
51	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / PWM1_CH3 / ACMP1_WLAT / TM3 / INT0 / TK_TK1 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
52	P	V_{SS}
53	P	V_{DD}
54	I/O	PD.15 / EPWM0_CH5 / ACMP2_WLAT / TM3 / INT1 / TK_TK2 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
55	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD /

Pin	Type	M2L31KIDAE Pin Function
		UART5_TXD / PWM0_CH5 / EPWM0_CH0 / EQEI0_INDEX / TK_TK3 / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
56	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / PWM0_CH4 / EPWM0_CH1 / EQEI0_A / TK_TK4 / UTCPD0_VBSRCEN / LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
57	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / TK_SE / I2C0_SMBAL / UART1_TXD / I2C1_SCL / PWM1_BRAKE1 / EQEI0_B / PWM0_CH3 / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / TK_TK5 / UTCPD0_VBSNKEN / LPSPI0_SS
58	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / PWM0_CH2 / EPWM0_CH3 / TK_TK6 / UTCPD0_VBSRCEN / LPSPI0_CLK
59	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / I2C2_SCL / CANFD1_TXD / EQEI0_INDEX / PWM0_CH1 / EPWM0_CH4 / ACMP2_O / DAC1_ST / TK_TK7 / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
60	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / I2C2_SDA / CANFD1_RXD / EPWM0_BRAKE0 / PWM0_CH0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / TK_TK8 / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
61	P	V_{DDIO}
62	I/O	PE.14 / EBI_AD8 / UART2_TXD / CANFD0_TXD / UART6_TXD / EPWM0_CH1 / TM2 / CLKO / INT4 / TK_TK9
63	I/O	PE.15 / EBI_AD9 / UART2_RXD / CANFD0_RXD / UART6_RXD
64	I	nRESET
65	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / PWM1_CH0 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
66	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / SPI3_I2SMCLK / EPWM1_CH5 / PWM1_CH1 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
67	I/O	PD.9 / EBI_AD7 / I2C2_SCL / UART2_nCTS / UART7_TXD
68	I/O	PD.8 / EBI_AD6 / I2C2_SDA / UART2_nRTS / UART7_RXD
69	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / SPI3_SS / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / I2C3_SMBAL / TK_TK10 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
70	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI3_CLK / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / I2C3_SMBSUS / TK_TK11 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
71	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI3_MISO / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / I2C3_SCL / TK_TK12 / UTCPD0_CCDB2
72	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI3_MOSI / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / I2C3_SDA / TK_TK13 / UTCPD0_CCDB1
73	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
74	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
75	P	V_{SS}
76	P	V_{DD}
77	I/O	PG.9 / EBI_AD0 / PWM0_CH5
78	I/O	PG.10 / EBI_AD1 / PWM0_CH4

Pin	Type	M2L31KIDAE Pin Function
79	I/O	PG.11 / EBI_AD2 / UART7_TXD / PWM0_CH3
80	I/O	PG.12 / EBI_AD3 / UART7_RXD / PWM0_CH2
81	I/O	PG.13 / EBI_AD4 / UART6_TXD / PWM0_CH1
82	I/O	PG.14 / EBI_AD5 / UART6_RXD / PWM0_CH0
83	I/O	PG.15 / CLKO / EADC0_ST / TK_SE / LPADC0_ST
84	I/O	PD.7 / UART1_TXD / I2C0_SCL / USCI1_CLK / TK_TK14 / LPI2C0_SCL
85	I/O	PD.6 / UART1_RXD / I2C0_SDA / USCI1_DAT1 / TK_TK15 / LPI2C0_SDA
86	I/O	PD.5 / I2C1_SCL / USCI1_DAT0 / TK_TK16
87	I/O	PD.4 / USCI0_CTL0 / I2C1_SDA / USCI1_CTL1 / TK_TK17
88	I/O	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD / TK_TK14 / LPSPI0_SS / LPUART0_TXD
89	I/O	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD / TK_TK15 / LPSPI0_CLK / LPUART0_RXD
90	I/O	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD / TK_TK16 / LPSPI0_MISO / LPIO7
91	I/O	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2 / TK_TK17 / LPSPI0_MOSI / LPIO6
92	I/O	PD.13 / WPPIN / EBI_AD10 / SPI0_I2SMCLK / SPI1_I2SMCLK / PWM0_CH0 / CLKO / EADC0_ST / TK_SE / LPADC0_ST
93	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI2_SS / CANFD1_TXD / SPI0_SS / PWM1_CH2 / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
94	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI2_CLK / CANFD1_RXD / SPI0_CLK / PWM1_CH3 / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
95	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI2_MISO / I2C2_SCL / SPI0_MISO / PWM1_CH4 / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
96	I/O	PA.15 / UART0_RXD / SPI2_MOSI / I2C2_SDA / SPI0_MOSI / PWM1_CH5 / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
97	I/O	PE.7 / UART5_TXD / CANFD1_TXD / EQEI1_INDEX / EPWM0_CH0 / PWM0_CH5
98	I/O	PE.6 / SPI3_I2SMCLK / USCI0_CTL0 / UART5_RXD / CANFD1_RXD / EQEI1_A / EPWM0_CH1 / PWM0_CH4
99	I/O	PE.5 / EBI_nRD / SPI3_SS / USCI0_CTL1 / UART6_TXD / UART7_nRTS / EQEI1_B / EPWM0_CH2 / PWM0_CH3
100	I/O	PE.4 / EBI_nWR / SPI3_CLK / USCI0_DAT1 / UART6_RXD / UART7_nCTS / EQEI0_INDEX / EPWM0_CH3 / PWM0_CH2
101	I/O	PE.3 / EBI_MCLK / SPI3_MISO / USCI0_DAT0 / UART6_nRTS / UART7_TXD / EQEI0_A / EPWM0_CH4 / PWM0_CH1
102	I/O	PE.2 / EBI_ALE / SPI3_MOSI / USCI0_CLK / UART6_nCTS / UART7_RXD / EQEI0_B / EPWM0_CH5 / PWM0_CH0
103	P	V_{SS}
104	P	V_{DD}
105	I/O	PE.1 / EBI_AD10 / QSPI0_MISO0 / SPI1_MISO / UART3_TXD / I2C1_SCL / UART4_nCTS / LPIO1
106	I/O	PE.0 / EBI_AD11 / QSPI0_MOSI0 / SPI1_MOSI / UART3_RXD / I2C1_SDA / UART4_nRTS / LPIO0
107	I/O	PH.8 / EBI_AD12 / QSPI0_CLK / SPI1_CLK / UART3_nRTS / I2C1_SMBAL / I2C2_SCL /

Pin	Type	M2L31KIDAE Pin Function
		UART1_TXD
108	I/O	PH.9 / EBI_AD13 / QSPI0_SS / SPI1_SS / UART3_nCTS / I2C1_SMBSUS / I2C2_SDA / UART1_RXD
109	I/O	PH.10 / EBI_AD14 / QSPI0_MISO1 / SPI1_I2SMCLK / UART4_TXD / UART0_TXD / LPUART0_TXD
110	I/O	PH.11 / EBI_AD15 / QSPI0_MOSI1 / UART4_RXD / UART0_RXD / EPWM0_CH5 / LPUART0_RXD
111	I/O	PD.14 / EBI_nCS0 / SPI3_I2SMCLK / SPI0_I2SMCLK / EPWM0_CH4
112	P	V_{SS}
113	A	LDO_CAP
114	P	V_{DD}
115	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
116	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / I2C2_SMBAL / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
117	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / I2C2_SMBSUS / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / TK_SE / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
118	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / DAC1_OUT / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / I2C2_SCL / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
119	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / I2C2_SDA / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
120	P	AV_{DD}
121	A	V_{REF}
122	P	AV_{SS}
123	I/O	PB.11 / EADC0_CH11 / LPADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / CANFD1_TXD / SPI0_I2SMCLK / PWM1_CH0 / LPUART0_nCTS
124	I/O	PB.10 / EADC0_CH10 / LPADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / CANFD1_RXD / PWM1_CH1 / LPUART0_nRTS
125	I/O	PB.9 / EADC0_CH9 / LPADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / UART7_TXD / PWM1_CH2 / LPUART0_TXD
126	I/O	PB.8 / EADC0_CH8 / LPADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / UART7_RXD / PWM1_CH3 / LPUART0_RXD
127	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / OPA2_N1 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / PWM1_CH4 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / PWM1_BRAKE0 / ACMP0_O
128	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / OPA2_P1 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / PWM1_CH5 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / PWM1_BRAKE1 / ACMP1_O

Table 4.3-17 M2L31KIDAE Multi-function Pin Table

4.3.5.2 M2L31KGDAE

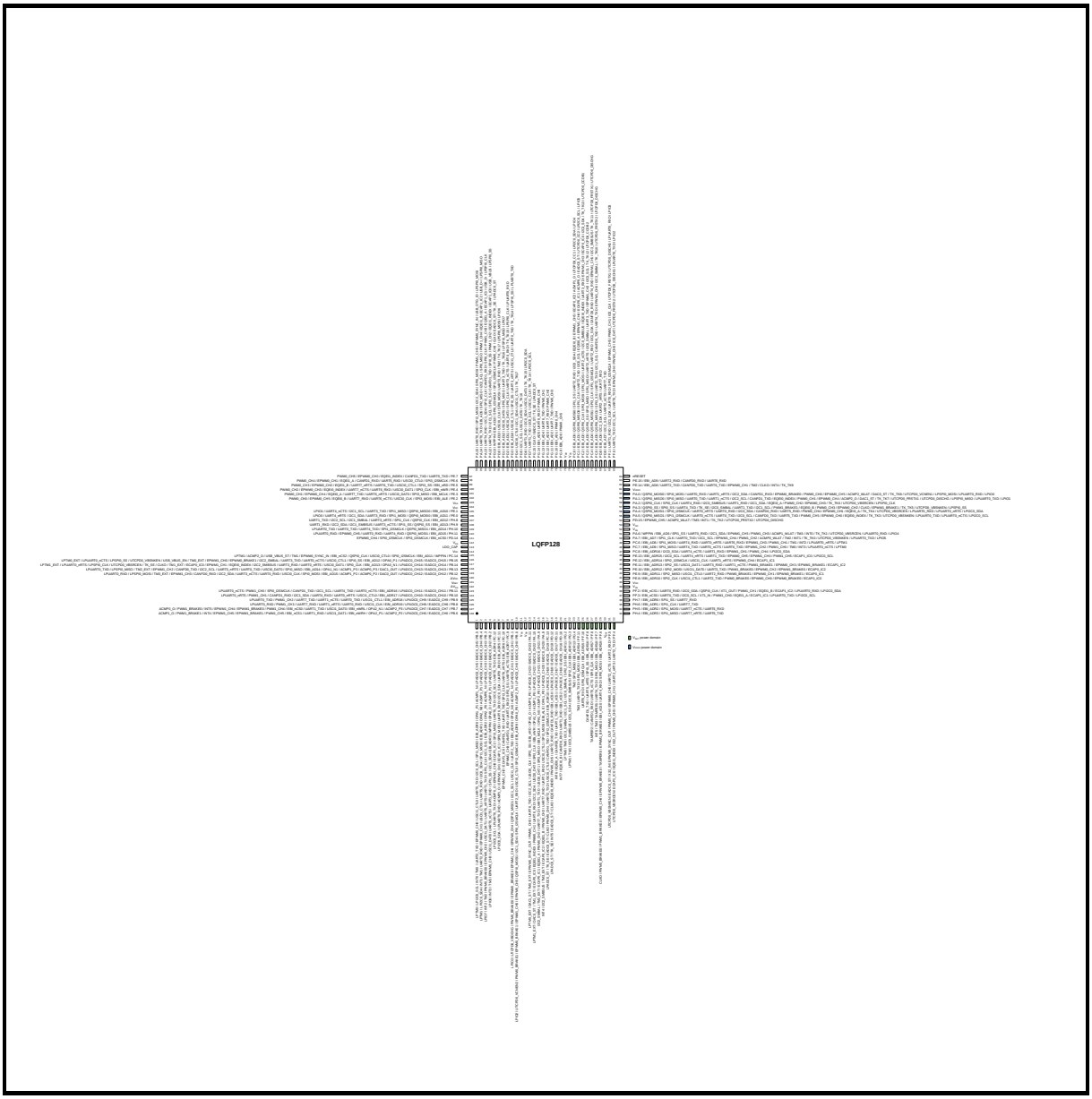


Figure 4.3-18 M2L31KGDAE Multi-function Pin Diagram

Pin	Type	M2L31KGDAE Pin Function
1	I/O	PB.5 / EADC0_CH5 / LPADC0_CH5 / ACMP1_N / OPA1_P1 / EBI_ADR0 / SPI1_MISO / I2C0_SCL / UART5_TXD / USCI1_CTL0 / EPWM0_CH0 / UART2_TXD / TM0 / INT0 / LPI2C0_SCL / LPTM0
2	I/O	PB.4 / EADC0_CH4 / LPADC0_CH4 / ACMP1_P1 / OPA2_N0 / EBI_ADR1 / SPI1_MOSI / I2C0_SDA / UART5_RXD / USCI1_CTL1 / EPWM0_CH1 / UART2_RXD / TM1 / INT1 / LPI2C0_SDA / LPTM1
3	I/O	PB.3 / EADC0_CH3 / LPADC0_CH3 / ACMP0_N / OPA2_P0 / EBI_ADR2 / I2C1_SCL / SPI1_CLK / UART1_TXD / UART5_nRTS / USCI1_DAT1 / EPWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2 / LPIO7
4	I/O	PB.2 / EADC0_CH2 / LPADC0_CH2 / ACMP0_P1 / OPA0_O / WPPIN / EBI_ADR3 / I2C1_SDA / SPI1_SS / UART1_RXD / UART5_nCTS / USCI1_DAT0 / EPWM0_CH3 / TM3 / INT3 / LPIO6
5	I/O	PC.12 / EBI_ADR4 / UART0_TXD / I2C0_SCL / UART6_TXD / SPI3_MISO / ECAP1_IC2 /

Pin	Type	M2L31KGDAE Pin Function
		EPWM1_CH0 / ACMP0_O / LPUART0_TXD / LPI2C0_SCL
6	I/O	PC.11 / EBI_ADR5 / UART0_RXD / I2C0_SDA / UART6_RXD / SPI3_MOSI / ECAP1_IC1 / EPWM1_CH1 / ACMP1_O / LPUART0_RXD / LPI2C0_SDA
7	I/O	PC.10 / EBI_ADR6 / UART6_nRTS / SPI3_CLK / UART3_TXD / CANFD1_TXD / ECAP1_IC0 / EPWM1_CH2
8	I/O	PC.9 / EBI_ADR7 / UART6_nCTS / SPI3_SS / UART3_RXD / CANFD1_RXD / EPWM1_CH3
9	I/O	PB.1 / EADC0_CH1 / LPADC0_CH1 / ACMP2_N / OPA0_N0 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / EPWM0_CH4 / EPWM1_CH4 / EPWM0_BRAKE0 / PWM0_BRAKE0 / UTCPD0_VBDCHG / LPIO3
10	I/O	PB.0 / EADC0_CH0 / LPADC0_CH0 / ACMP2_P1 / OPA0_P0 / EBI_ADR9 / SPI3_I2SMCLK / USCI0_CTL0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / EPWM0_CH5 / EPWM1_CH5 / EPWM0_BRAKE1 / PWM0_BRAKE1 / UTCPD0_VCNEN2 / LPIO2
11	P	V_{SS}
12	P	V_{DD}
13	I/O	PA.11 / EADC0_CH23 / LPADC0_CH23 / ACMP0_P0 / OPA2_O / EBI_nRD / SPI3_SS / USCI0_CLK / I2C2_SCL / UART6_TXD / PWM0_CH0 / EPWM0_SYNC_OUT / TM0_EXT / DAC1_ST / LPTM0_EXT
14	I/O	PA.10 / EADC0_CH22 / LPADC0_CH22 / ACMP1_P0 / OPA1_O / EBI_nWR / SPI3_CLK / USCI0_DAT0 / I2C2_SDA / UART6_RXD / PWM0_CH1 / EQEI1_INDEX / ECAP0_IC0 / TM1_EXT / DAC0_ST / LPTM1_EXT
15	I/O	PA.9 / EADC0_CH21 / LPADC0_CH21 / ACMP2_P0 / OPA1_N0 / EBI_MCLK / SPI3_MISO / USCI0_DAT1 / UART1_TXD / UART7_TXD / PWM0_CH2 / EQEI1_A / ECAP0_IC1 / TM2_EXT / I2C2_SMBAL
16	I/O	PA.8 / EADC0_CH20 / LPADC0_CH20 / OPA1_P0 / EBI_ALE / SPI3_MOSI / USCI0_CTL1 / UART1_RXD / UART7_RXD / PWM0_CH3 / EQEI1_B / ECAP0_IC2 / TM3_EXT / I2C2_SMBSUS / INT4
17	I/O	PC.13 / EADC0_CH19 / LPADC0_CH19 / EBI_ADR10 / SPI2_I2SMCLK / CANFD1_TXD / USCI0_CTL0 / UART2_TXD / PWM0_CH4 / CLKO / EADC0_ST / TK_SE / LPADC0_ST
18	I/O	PD.12 / EADC0_CH18 / LPADC0_CH18 / EBI_nCS0 / CANFD1_RXD / UART2_RXD / PWM0_CH5 / EQEI0_INDEX / CLKO / EADC0_ST / INT5 / TK_SE / LPADC0_ST
19	I/O	PD.11 / EADC0_CH17 / LPADC0_CH17 / EBI_nCS1 / UART1_TXD / CANFD0_TXD / EQEI0_A / INT6
20	I/O	PD.10 / EADC0_CH16 / LPADC0_CH16 / EBI_nCS2 / UART1_RXD / CANFD0_RXD / EQEI0_B / INT7
21	I/O	PG.2 / EBI_ADR11 / SPI2_SS / I2C0_SMBAL / I2C1_SCL / I2C3_SMBAL / TM0 / LPTM0
22	I/O	PG.3 / EBI_ADR12 / SPI2_CLK / I2C0_SMBSUS / I2C1_SDA / I2C3_SMBSUS / TM1 / LPTM1
23	I/O	PG.4 / EBI_ADR13 / SPI2_MISO / TM2
24	I/O	PF.11 / EBI_ADR14 / SPI2_MOSI / UART5_TXD / TM3
25	I/O	PF.10 / EBI_ADR15 / SPI0_I2SMCLK / UART5_RXD
26	I/O	PF.9 / EBI_ADR16 / SPI0_SS / UART5_nRTS / CANFD1_TXD
27	I/O	PF.8 / EBI_ADR17 / SPI0_CLK / UART5_nCTS / CANFD1_RXD / TAMPER2
28	I/O	PF.7 / EBI_ADR18 / SPI0_MISO / UART4_TXD / TAMPER1 / TM3 / INT5
29	I/O	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0 / EPWM1_BRAKE0 / TAMPER0 / EPWM0_BRAKE0 / EPWM0_CH4 / PWM1_BRAKE0 / PWM0_BRAKE0 / CLKO
30	P	V_{BAT}

Pin	Type	M2L31KGDAE Pin Function
31	I/O	PF.5 / UART2_RXD / UART2_nCTS / EPWM0_CH0 / PWM0_CH4 / EPWM0_SYNC_OUT / X32_IN / EADC0_ST / UTCPD0_VBSNKEN
32	I/O	PF.4 / UART2_TXD / UART2_nRTS / EPWM0_CH1 / PWM0_CH5 / X32_OUT / EQEI1_INDEX / ECAP1_IC0 / UTCPD0_VBSRCEN
33	I/O	PH.4 / EBI_ADR3 / SPI1_MISO / UART7_nRTS / UART6_TXD
34	I/O	PH.5 / EBI_ADR2 / SPI1_MOSI / UART7_nCTS / UART6_RXD
35	I/O	PH.6 / EBI_ADR1 / SPI1_CLK / UART7_TXD
36	I/O	PH.7 / EBI_ADR0 / SPI1_SS / UART7_RXD
37	I/O	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / PWM1_CH0 / EQEI1_A / ECAP1_IC1 / LPUART0_TXD / LPI2C0_SCL
38	I/O	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / PWM1_CH1 / EQEI1_B / ECAP1_IC2 / LPUART0_RXD / LPI2C0_SDA
39	P	V_{SS}
40	P	V_{DD}
41	I/O	PE.8 / EBI_ADR10 / SPI2_CLK / USCI1_CTL1 / UART2_TXD / PWM0_BRAKE0 / EPWM0_CH0 / EPWM0_BRAKE0 / ECAP0_IC0
42	I/O	PE.9 / EBI_ADR11 / SPI2_MISO / USCI1_CTL0 / UART2_RXD / PWM0_BRAKE1 / EPWM0_CH1 / EPWM0_BRAKE1 / ECAP0_IC1
43	I/O	PE.10 / EBI_ADR12 / SPI2_MOSI / USCI1_DAT0 / UART3_TXD / PWM1_BRAKE0 / EPWM0_CH2 / EPWM1_BRAKE0 / ECAP0_IC2
44	I/O	PE.11 / EBI_ADR13 / SPI2_SS / USCI1_DAT1 / UART3_RXD / UART1_nCTS / PWM1_BRAKE1 / EPWM0_CH3 / EPWM1_BRAKE1 / ECAP1_IC2
45	I/O	PE.12 / EBI_ADR14 / SPI2_I2SMCLK / USCI1_CLK / UART1_nRTS / EPWM0_CH4 / ECAP1_IC1
46	I/O	PE.13 / EBI_ADR15 / I2C0_SCL / UART4_nRTS / UART1_TXD / EPWM0_CH5 / EPWM1_CH0 / PWM1_CH5 / ECAP1_IC0 / LPI2C0_SCL
47	I/O	PC.8 / EBI_ADR16 / I2C0_SDA / UART4_nCTS / UART1_RXD / EPWM1_CH1 / PWM1_CH4 / LPI2C0_SDA
48	I/O	PC.7 / EBI_AD9 / SPI1_MISO / UART4_TXD / UART0_nCTS / UART6_TXD / EPWM1_CH2 / PWM1_CH0 / TM0 / INT3 / LPUART0_nCTS / LPTM0
49	I/O	PC.6 / EBI_AD8 / SPI1_MOSI / UART4_RXD / UART0_nRTS / UART6_RXD / EPWM1_CH3 / PWM1_CH1 / TM1 / INT2 / LPUART0_nRTS / LPTM1
50	I/O	PA.7 / EBI_AD7 / SPI1_CLK / UART0_TXD / I2C1_SCL / EPWM1_CH4 / PWM1_CH2 / ACMP0_WLAT / TM2 / INT1 / TK_TK0 / UTCPD0_VBSNKEN / LPUART0_TXD / LPIO5
51	I/O	PA.6 / WPPIN / EBI_AD6 / SPI1_SS / UART0_RXD / I2C1_SDA / EPWM1_CH5 / PWM1_CH3 / ACMP1_WLAT / TM3 / INT0 / TK_TK1 / UTCPD0_VBSRCEN / LPUART0_RXD / LPIO4
52	P	V_{SS}
53	P	V_{DD}
54	I/O	PD.15 / EPWM0_CH5 / ACMP2_WLAT / TM3 / INT1 / TK_TK2 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
55	I/O	PA.5 / QSPI0_MISO1 / SPI1_I2SMCLK / UART0_nCTS / UART0_TXD / I2C0_SCL / CANFD0_TXD / UART5_TXD / PWM0_CH5 / EPWM0_CH0 / EQEI0_INDEX / TK_TK3 / UTCPD0_VBSNKEN / LPUART0_TXD / LPUART0_nCTS / LPI2C0_SCL
56	I/O	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / CANFD0_RXD / UART5_RXD / PWM0_CH4 / EPWM0_CH1 / EQEI0_A / TK_TK4 / UTCPD0_VBSRCEN /

Pin	Type	M2L31KGDAE Pin Function
		LPUART0_RXD / LPUART0_nRTS / LPI2C0_SDA
57	I/O	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / TK_SE / I2C0_SMBAL / UART1_TXD / I2C1_SCL / PWM1_BRAKE1 / EQEI0_B / PWM0_CH3 / EPWM0_CH2 / CLKO / EPWM1_BRAKE1 / TK_TK5 / UTCPD0_VBSNKEN / LPSPI0_SS
58	I/O	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / EQEI0_A / PWM0_CH2 / EPWM0_CH3 / TK_TK6 / UTCPD0_VBSRCEN / LPSPI0_CLK
59	I/O	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / I2C2_SCL / CANFD1_TXD / EQEI0_INDEX / PWM0_CH1 / EPWM0_CH4 / ACMP2_O / DAC1_ST / TK_TK7 / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPSPI0_MISO / LPUART0_TXD / LPIO1
60	I/O	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / I2C2_SDA / CANFD1_RXD / EPWM0_BRAKE0 / PWM0_CH0 / EPWM0_CH5 / ACMP2_WLAT / DAC0_ST / TK_TK8 / UTCPD0_VCNEN1 / LPSPI0_MOSI / LPUART0_RXD / LPIO0
61	P	V_{DDIO}
62	I/O	PE.14 / EBI_AD8 / UART2_TXD / CANFD0_TXD / UART6_TXD / EPWM0_CH1 / TM2 / CLKO / INT4 / TK_TK9
63	I/O	PE.15 / EBI_AD9 / UART2_RXD / CANFD0_RXD / UART6_RXD
64	I	nRESET
65	I/O	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / EPWM1_CH4 / PWM1_CH0 / ICE_DAT / UTCPD0_FRSTX2 / UTCPD0_DISCHG / LPUART0_TXD / LPIO2
66	I/O	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / SPI3_I2SMCLK / EPWM1_CH5 / PWM1_CH1 / ICE_CLK / UTCPD0_FRSTX1 / UTCPD0_DISCHG / LPUART0_RXD / LPIO3
67	I/O	PD.9 / EBI_AD7 / I2C2_SCL / UART2_nCTS / UART7_TXD
68	I/O	PD.8 / EBI_AD6 / I2C2_SDA / UART2_nRTS / UART7_RXD
69	I/O	PC.5 / EBI_AD5 / QSPI0_MISO1 / SPI3_SS / UART2_TXD / I2C1_SCL / CANFD0_TXD / UART4_TXD / EPWM1_CH0 / I2C3_SMBAL / TK_TK10 / UTCPD0_FRSTX2 / UTCPD0_DISCHG
70	I/O	PC.4 / EBI_AD4 / QSPI0_MOSI1 / SPI3_CLK / SPI1_I2SMCLK / UART2_RXD / I2C1_SDA / CANFD0_RXD / UART4_RXD / EPWM1_CH1 / I2C3_SMBSUS / TK_TK11 / UTCPD0_FRSTX1 / UTCPD0_DISCHG
71	I/O	PC.3 / EBI_AD3 / QSPI0_SS / SPI3_MISO / SPI1_MISO / UART2_nRTS / I2C0_SMBAL / UART3_TXD / EPWM1_CH2 / I2C3_SCL / TK_TK12 / UTCPD0_CCDB2
72	I/O	PC.2 / EBI_AD2 / QSPI0_CLK / SPI3_MOSI / SPI1_MOSI / UART2_nCTS / I2C0_SMBSUS / EQEI0_INDEX / UART3_RXD / EPWM1_CH3 / ECAP0_IC0 / I2C3_SDA / TK_TK13 / UTCPD0_CCDB1
73	I/O	PC.1 / EBI_AD1 / QSPI0_MISO0 / SPI1_CLK / UART2_TXD / I2C0_SCL / EQEI0_A / EPWM1_CH4 / ECAP0_IC1 / ACMP0_O / EADC0_ST / UTCPD0_CC2 / LPI2C0_SCL / LPIO5
74	I/O	PC.0 / EBI_AD0 / QSPI0_MOSI0 / SPI1_SS / UART2_RXD / I2C0_SDA / EQEI0_B / EPWM1_CH5 / ECAP0_IC2 / ACMP1_O / UTCPD0_CC1 / LPI2C0_SDA / LPIO4
75	P	V_{SS}
76	P	V_{DD}
77	I/O	PG.9 / EBI_AD0 / PWM0_CH5
78	I/O	PG.10 / EBI_AD1 / PWM0_CH4
79	I/O	PG.11 / EBI_AD2 / UART7_TXD / PWM0_CH3
80	I/O	PG.12 / EBI_AD3 / UART7_RXD / PWM0_CH2
81	I/O	PG.13 / EBI_AD4 / UART6_TXD / PWM0_CH1

Pin	Type	M2L31KGDAE Pin Function
82	I/O	PG.14 / EBI_AD5 / UART6_RXD / PWM0_CH0
83	I/O	PG.15 / CLKO / EADC0_ST / TK_SE / LPADC0_ST
84	I/O	PD.7 / UART1_TXD / I2C0_SCL / USCI1_CLK / TK_TK14 / LPI2C0_SCL
85	I/O	PD.6 / UART1_RXD / I2C0_SDA / USCI1_DAT1 / TK_TK15 / LPI2C0_SDA
86	I/O	PD.5 / I2C1_SCL / USCI1_DAT0 / TK_TK16
87	I/O	PD.4 / USCI0_CTL0 / I2C1_SDA / USCI1_CTL1 / TK_TK17
88	I/O	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD / TK_TK14 / LPSPI0_SS / LPUART0_TXD
89	I/O	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD / TK_TK15 / LPSPI0_CLK / LPUART0_RXD
90	I/O	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD / TK_TK16 / LPSPI0_MISO / LPIO7
91	I/O	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2 / TK_TK17 / LPSPI0_MOSI / LPIO6
92	I/O	PD.13 / WPPIN / EBI_AD10 / SPI0_I2SMCLK / SPI1_I2SMCLK / PWM0_CH0 / CLKO / EADC0_ST / TK_SE / LPADC0_ST
93	I/O	PA.12 / UART4_TXD / I2C1_SCL / SPI2_SS / CANFD1_TXD / SPI0_SS / PWM1_CH2 / EQEI1_INDEX / ECAP1_IC0 / USB_VBUS / LPSPI0_SS
94	I/O	PA.13 / UART4_RXD / I2C1_SDA / SPI2_CLK / CANFD1_RXD / SPI0_CLK / PWM1_CH3 / EQEI1_A / ECAP1_IC1 / USB_D- / LPSPI0_CLK
95	I/O	PA.14 / UART0_TXD / EBI_AD5 / SPI2_MISO / I2C2_SCL / SPI0_MISO / PWM1_CH4 / EQEI1_B / ECAP1_IC2 / USB_D+ / LPSPI0_MISO
96	I/O	PA.15 / UART0_RXD / SPI2_MOSI / I2C2_SDA / SPI0_MOSI / PWM1_CH5 / EPWM0_SYNC_IN / USB_OTG_ID / LPSPI0_MOSI
97	I/O	PE.7 / UART5_TXD / CANFD1_TXD / EQEI1_INDEX / EPWM0_CH0 / PWM0_CH5
98	I/O	PE.6 / SPI3_I2SMCLK / USCI0_CTL0 / UART5_RXD / CANFD1_RXD / EQEI1_A / EPWM0_CH1 / PWM0_CH4
99	I/O	PE.5 / EBI_nRD / SPI3_SS / USCI0_CTL1 / UART6_TXD / UART7_nRTS / EQEI1_B / EPWM0_CH2 / PWM0_CH3
100	I/O	PE.4 / EBI_nWR / SPI3_CLK / USCI0_DAT1 / UART6_RXD / UART7_nCTS / EQEI0_INDEX / EPWM0_CH3 / PWM0_CH2
101	I/O	PE.3 / EBI_MCLK / SPI3_MISO / USCI0_DAT0 / UART6_nRTS / UART7_TXD / EQEI0_A / EPWM0_CH4 / PWM0_CH1
102	I/O	PE.2 / EBI_ALE / SPI3_MOSI / USCI0_CLK / UART6_nCTS / UART7_RXD / EQEI0_B / EPWM0_CH5 / PWM0_CH0
103	P	V_{SS}
104	P	V_{DD}
105	I/O	PE.1 / EBI_AD10 / QSPI0_MISO0 / SPI1_MISO / UART3_TXD / I2C1_SCL / UART4_nCTS / LPIO1
106	I/O	PE.0 / EBI_AD11 / QSPI0_MOSI0 / SPI1_MOSI / UART3_RXD / I2C1_SDA / UART4_nRTS / LPIO0
107	I/O	PH.8 / EBI_AD12 / QSPI0_CLK / SPI1_CLK / UART3_nRTS / I2C1_SMBAL / I2C2_SCL / UART1_TXD
108	I/O	PH.9 / EBI_AD13 / QSPI0_SS / SPI1_SS / UART3_nCTS / I2C1_SMBSUS / I2C2_SDA / UART1_RXD
109	I/O	PH.10 / EBI_AD14 / QSPI0_MISO1 / SPI1_I2SMCLK / UART4_TXD / UART0_TXD / LPUART0_TXD

Pin	Type	M2L31KGDAE Pin Function
110	I/O	PH.11 / EBI_AD15 / QSPI0_MOSI1 / UART4_RXD / UART0_RXD / EPWM0_CH5 / LPUART0_RXD
111	I/O	PD.14 / EBI_nCS0 / SPI3_I2SMCLK / SPI0_I2SMCLK / EPWM0_CH4
112	P	V_{SS}
113	A	LDO_CAP
114	P	V_{DD}
115	I/O	PC.14 / WPPIN / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / EBI_nCS2 / EPWM0_SYNC_IN / TM1 / USB_VBUS_ST / ACMP2_O / LPTM1
116	I/O	PB.15 / EADC0_CH15 / LPADC0_CH15 / OPA0_P1 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / I2C2_SMBAL / EPWM0_BRAKE1 / EPWM1_CH0 / TM0_EXT / USB_VBUS_EN / UTCPD0_VBSNKEN / LPSPI0_SS / LPUART0_nCTS / LPTM0_EXT
117	I/O	PB.14 / EADC0_CH14 / LPADC0_CH14 / OPA0_N1 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / I2C2_SMBSUS / EQEI0_INDEX / EPWM1_CH1 / ECAP0_IC0 / TM1_EXT / CLKO / TK_SE / UTCPD0_VBSRCEN / LPSPI0_CLK / LPUART0_nRTS / LPTM1_EXT
118	I/O	PB.13 / EADC0_CH13 / LPADC0_CH13 / DAC1_OUT / ACMP0_P3 / ACMP1_P3 / OPA1_N1 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / I2C2_SCL / CANFD0_TXD / EPWM1_CH2 / TM2_EXT / LPSPI0_MISO / LPUART0_TXD
119	I/O	PB.12 / EADC0_CH12 / LPADC0_CH12 / DAC0_OUT / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / I2C2_SDA / CANFD0_RXD / EPWM1_CH3 / TM3_EXT / LPSPI0_MOSI / LPUART0_RXD
120	P	AV_{DD}
121	A	V_{REF}
122	P	AV_{SS}
123	I/O	PB.11 / EADC0_CH11 / LPADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / CANFD1_TXD / SPI0_I2SMCLK / PWM1_CH0 / LPUART0_nCTS
124	I/O	PB.10 / EADC0_CH10 / LPADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / CANFD1_RXD / PWM1_CH1 / LPUART0_nRTS
125	I/O	PB.9 / EADC0_CH9 / LPADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / UART7_TXD / PWM1_CH2 / LPUART0_TXD
126	I/O	PB.8 / EADC0_CH8 / LPADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / UART7_RXD / PWM1_CH3 / LPUART0_RXD
127	I/O	PB.7 / EADC0_CH7 / LPADC0_CH7 / ACMP2_P3 / OPA2_N1 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / PWM1_CH4 / EPWM1_BRAKE0 / EPWM1_CH4 / INT5 / PWM1_BRAKE0 / ACMP0_O
128	I/O	PB.6 / EADC0_CH6 / LPADC0_CH6 / ACMP2_P2 / OPA2_P1 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / PWM1_CH5 / EPWM1_BRAKE1 / EPWM1_CH5 / INT4 / PWM1_BRAKE1 / ACMP1_O

Table 4.3-18 M2L31KGDAE Multi-function Pin Table

4.4 Pin Mapping Table

4.4.1 M2L31 Series Mapping Table

	M231 Series				
Pin Name	M2L31ZxAE	M2L31YxAE	M2L31LxAE	M2L31SxAE	M2L31KxAE
PB.5	1	1	1	2	1
PB.4	2	2	2	3	2
PB.3	3	3	3	4	3
PB.2	4	4	4	5	4
PC.12					5
PC.11					6
PC.10					7
PC.9					8
PB.1	5	5	5	6	9
PB.0	6	6	6	7	10
V_{SS}					11
V_{DD}					12
PA.11		7	7	8	13
PA.10		8	8	9	14
PA.9		9	9	10	15
PA.8		10	10	11	16
PC.13					17
PD.12					18
PD.11					19
PD.10					20
PG.2					21
PG.3					22
PG.4					23
PF.11					24
PF.10					25
PF.9					26
PF.8					27
PF.7					28
PF.6				12	29
V_{BAT}				13	30

PF.5	7	11	11	14	31
PF.4	8	12	12	15	32
PH.4					33
PH.5					34
PH.6					35
PH.7					36
PF.3	9	13	13	16	37
PF.2	10	14	14	17	38
V_{SS}					39
V_{DD}					40
PE.8					41
PE.9					42
PE.10					43
PE.11					44
PE.12					45
PE.13					46
PC.8					47
PC.7				18	48
PC.6				19	49
PA.7		15	15	20	50
PA.6		16	16	21	51
V_{SS}				22	52
V_{DD}				23	53
PD.15				24	54
PA.5		17	17	25	55
PA.4		18	18	26	56
PA.3	11	19	19	27	57
PA.2	12	20	20	28	58
PA.1	13	21	21	29	59
PA.0	14	22	22	30	60
V_{DDIO}	15	23	23	31	61
PE.14					62
PE.15					63
nRESET	16	24	24	32	64
PF.0	17	25	25	33	65

PF.1	18	26	26	34	66
PD.9					67
PD.8					68
PC.5		27	27	35	69
PC.4		28	28	36	70
PC.3		29	29	37	71
PC.2		30	30	38	72
PC.1	19	31	31	39	73
PC.0	20	32	32	40	74
V_{SS}					75
V_{DD}					76
PG.9					77
PG.10					78
PG.11					79
PG.12					80
PG.13					81
PG.14					82
PG.15					83
PD.7					84
PD.6					85
PD.5					86
PD.4					87
PD.3				41	88
PD.2				42	89
PD.1				43	90
PD.0				44	91
PD.13					92
PA.12	21	33	33	45	93
PA.13	22	34	34	46	94
PA.14	23	35	35	47	95
PA.15	24	36	36	48	96
PE.7					97
PE.6					98
PE.5					99
PE.4					100

PE.3					101
PE.2					102
V_{SS}	25	37	37	49	103
V_{DD}					104
PE.1					105
PE.0					106
PH.8					107
PH.9					108
PH.10					109
PH.11					110
PD.14					111
V_{SS}					112
LDO_CAP	26	38	38	50	113
V_{DD}	27	39	39	51	114
PC.14		40	40	52	115
PB.15	28	41	41	53	116
PB.14	29	42	42	54	117
PB.13	30	43	43	55	118
PB.12	31	44	44	56	119
AV_{DD}	32	45	45	57	120
V_{REF}				58	121
AV_{SS}		46	46	59	122
PB.11				60	123
PB.10				61	124
PB.9				62	125
PB.8				63	126
PB.7		47	47	64	127
PB.6		48	48	1	128
V_{SS}	33	49			

4.5 Pin Functional Description

4.5.1 M2L31 Series Summary Function Pin Description

Group	Pin Name	Type	Description
ACMP0	ACMP0_N	A	Analog comparator 0 negative input pin.
	ACMP0_O	O	Analog comparator 0 output pin.
	ACMP0_P0	A	Analog comparator 0 positive input 0 pin.
	ACMP0_P1	A	Analog comparator 0 positive input 1 pin.
	ACMP0_P2	A	Analog comparator 0 positive input 2 pin.
	ACMP0_P3	A	Analog comparator 0 positive input 3 pin.
	ACMP0_WLAT	I	Analog comparator 0 window latch input pin
ACMP1	ACMP1_N	A	Analog comparator 1 negative input pin.
	ACMP1_O	O	Analog comparator 1 output pin.
	ACMP1_P0	A	Analog comparator 1 positive input 0 pin.
	ACMP1_P1	A	Analog comparator 1 positive input 1 pin.
	ACMP1_P2	A	Analog comparator 1 positive input 2 pin.
	ACMP1_P3	A	Analog comparator 1 positive input 3 pin.
	ACMP1_WLAT	I	Analog comparator 1 window latch input pin
ACMP2	ACMP2_N	A	Analog comparator 2 negative input pin.
	ACMP2_O	O	Analog comparator 2 output pin.
	ACMP2_P0	A	Analog comparator 2 positive input 0 pin.
	ACMP2_P1	A	Analog comparator 2 positive input 1 pin.
	ACMP2_P2	A	Analog comparator 2 positive input 2 pin.
	ACMP2_P3	A	Analog comparator 2 positive input 3 pin.
	ACMP2_WLAT	I	Analog comparator 2 window latch input pin
CANFD0	CANFD0_RXD	I	CANF bus receiver input.
	CANFD0_TXD	O	CANF bus transmitter output.
CANFD1	CANFD1_RXD	I	CANF bus receiver input.
	CANFD1_TXD	O	CANF bus transmitter output.
CLKO	CLKO	O	Clock Out
DAC0	DAC0_OUT	A	DAC0 channel analog output.
	DAC0_ST	I	DAC0 external trigger input.
DAC1	DAC1_OUT	A	DAC1 channel analog output.
	DAC1_ST	I	DAC1 external trigger input.
EADC0	EADC0_CH0	A	EADC0 channel 0 analog input.

Group	Pin Name	Type	Description
	EADC0_CH1	A	EADC0 channel 1 analog input.
	EADC0_CH2	A	EADC0 channel 2 analog input.
	EADC0_CH3	A	EADC0 channel 3 analog input.
	EADC0_CH4	A	EADC0 channel 4 analog input.
	EADC0_CH5	A	EADC0 channel 5 analog input.
	EADC0_CH6	A	EADC0 channel 6 analog input.
	EADC0_CH7	A	EADC0 channel 7 analog input.
	EADC0_CH8	A	EADC0 channel 8 analog input.
	EADC0_CH9	A	EADC0 channel 9 analog input.
	EADC0_CH10	A	EADC0 channel 10 analog input.
	EADC0_CH11	A	EADC0 channel 11 analog input.
	EADC0_CH12	A	EADC0 channel 12 analog input.
	EADC0_CH13	A	EADC0 channel 13 analog input.
	EADC0_CH14	A	EADC0 channel 14 analog input.
	EADC0_CH15	A	EADC0 channel 15 analog input.
	EADC0_CH16	A	EADC0 channel 16 analog input.
	EADC0_CH17	A	EADC0 channel 17 analog input.
	EADC0_CH18	A	EADC0 channel 18 analog input.
	EADC0_CH19	A	EADC0 channel 19 analog input.
	EADC0_CH20	A	EADC0 channel 20 analog input.
	EADC0_CH21	A	EADC0 channel 21 analog input.
	EADC0_CH22	A	EADC0 channel 22 analog input.
	EADC0_CH23	A	EADC0 channel 23 analog input.
	EADC0_ST	I	EADC0 external trigger input.
EBI	EBI_AD0	I/O	EBI address/data bus bit 0.
	EBI_AD1	I/O	EBI address/data bus bit 1.
	EBI_AD2	I/O	EBI address/data bus bit 2.
	EBI_AD3	I/O	EBI address/data bus bit 3.
	EBI_AD4	I/O	EBI address/data bus bit 4.
	EBI_AD5	I/O	EBI address/data bus bit 5.
	EBI_AD6	I/O	EBI address/data bus bit 6.
	EBI_AD7	I/O	EBI address/data bus bit 7.
	EBI_AD8	I/O	EBI address/data bus bit 8.
	EBI_AD9	I/O	EBI address/data bus bit 9.

Group	Pin Name	Type	Description
	EBI_AD10	I/O	EBI address/data bus bit 10.
	EBI_AD11	I/O	EBI address/data bus bit 11.
	EBI_AD12	I/O	EBI address/data bus bit 12.
	EBI_AD13	I/O	EBI address/data bus bit 13.
	EBI_AD14	I/O	EBI address/data bus bit 14.
	EBI_AD15	I/O	EBI address/data bus bit 15.
	EBI_ADR0	O	EBI address bus bit 0.
	EBI_ADR1	O	EBI address bus bit 1.
	EBI_ADR2	O	EBI address bus bit 2.
	EBI_ADR3	O	EBI address bus bit 3.
	EBI_ADR4	O	EBI address bus bit 4.
	EBI_ADR5	O	EBI address bus bit 5.
	EBI_ADR6	O	EBI address bus bit 6.
	EBI_ADR7	O	EBI address bus bit 7.
	EBI_ADR8	O	EBI address bus bit 8.
	EBI_ADR9	O	EBI address bus bit 9.
	EBI_ADR10	O	EBI address bus bit 10.
	EBI_ADR11	O	EBI address bus bit 11.
	EBI_ADR12	O	EBI address bus bit 12.
	EBI_ADR13	O	EBI address bus bit 13.
	EBI_ADR14	O	EBI address bus bit 14.
	EBI_ADR15	O	EBI address bus bit 15.
	EBI_ADR16	O	EBI address bus bit 16.
	EBI_ADR17	O	EBI address bus bit 17.
	EBI_ADR18	O	EBI address bus bit 18.
	EBI_ADR19	O	EBI address bus bit 19.
	EBI_ALE	O	EBI address latch enable output pin.
	EBI_MCLK	O	EBI external clock output pin.
	EBI_nCS0	O	EBI chip select 0 output pin.
	EBI_nCS1	O	EBI chip select 1 output pin.
	EBI_nCS2	O	EBI chip select 2 output pin.
	EBI_nRD	O	EBI read enable output pin.
	EBI_nWR	O	EBI write enable output pin.
	EBI_nWRH	O	EBI high byte write enable output pin

Group	Pin Name	Type	Description
	EBI_nWRL	O	EBI low byte write enable output pin.
ECAP0	ECAP0_IC0	I	Enhanced capture unit 0 input 0 pin.
	ECAP0_IC1	I	Enhanced capture unit 0 input 1 pin.
	ECAP0_IC2	I	Enhanced capture unit 0 input 2 pin.
ECAP1	ECAP1_IC0	I	Enhanced capture unit 1 input 0 pin.
	ECAP1_IC1	I	Enhanced capture unit 1 input 1 pin.
	ECAP1_IC2	I	Enhanced capture unit 1 input 2 pin.
EPWM0	EPWM0_BRAKE0	I	EPWM0 Brake 0 input pin.
	EPWM0_BRAKE1	I	EPWM0 Brake 1 input pin.
	EPWM0_CH0	I/O	EPWM0 channel 0 output/capture input.
	EPWM0_CH1	I/O	EPWM0 channel 1 output/capture input.
	EPWM0_CH2	I/O	EPWM0 channel 2 output/capture input.
	EPWM0_CH3	I/O	EPWM0 channel 3 output/capture input.
	EPWM0_CH4	I/O	EPWM0 channel 4 output/capture input.
	EPWM0_CH5	I/O	EPWM0 channel 5 output/capture input.
	EPWM0_SYNC_IN	I	EPWM0 counter synchronous trigger input pin.
	EPWM0_SYNC_OUT	O	EPWM0 counter synchronous trigger output pin.
EPWM1	EPWM1_BRAKE0	I	EPWM1 Brake 0 input pin.
	EPWM1_BRAKE1	I	EPWM1 Brake 1 input pin.
	EPWM1_CH0	I/O	EPWM1 channel 0 output/capture input.
	EPWM1_CH1	I/O	EPWM1 channel 1 output/capture input.
	EPWM1_CH2	I/O	EPWM1 channel 2 output/capture input.
	EPWM1_CH3	I/O	EPWM1 channel 3 output/capture input.
	EPWM1_CH4	I/O	EPWM1 channel 4 output/capture input.
	EPWM1_CH5	I/O	EPWM1 channel 5 output/capture input.
EQEI0	EQEI0_A	I	EQEI0 phase A input.
	EQEI0_B	I	EQEI0 phase B input.
	EQEI0_INDEX	I	EQEI0 index input.
EQEI1	EQEI1_A	I	EQEI1 phase A input.
	EQEI1_B	I	EQEI1 phase B input.
	EQEI1_INDEX	I	EQEI1 index input.
I2C0	I2C0_SCL	I/O	I2C0 clock pin.
	I2C0_SDA	I/O	I2C0 data input/output pin.

Group	Pin Name	Type	Description
	I2C0_SMBAL	O	I2C0 SMBus SMBALTER pin
	I2C0_SMBSUS	O	I2C0 SMBus SMBSUS pin (PMBus CONTROL pin)
I2C1	I2C1_SCL	I/O	I2C1 clock pin.
	I2C1_SDA	I/O	I2C1 data input/output pin.
	I2C1_SMBAL	O	I2C1 SMBus SMBALTER pin
	I2C1_SMBSUS	O	I2C1 SMBus SMBSUS pin (PMBus CONTROL pin)
I2C2	I2C2_SCL	I/O	I2C2 clock pin.
	I2C2_SDA	I/O	I2C2 data input/output pin.
	I2C2_SMBAL	O	I2C2 SMBus SMBALTER pin
	I2C2_SMBSUS	O	I2C2 SMBus SMBSUS pin (PMBus CONTROL pin)
I2C3	I2C3_SCL	I/O	I2C3 clock pin.
	I2C3_SDA	I/O	I2C3 data input/output pin.
	I2C3_SMBAL	O	I2C3 SMBus SMBALTER pin
	I2C3_SMBSUS	O	I2C3 SMBus SMBSUS pin (PMBus CONTROL pin)
ICE	ICE_CLK	I	Serial wired debugger clock pin. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_CLK pin.
	ICE_DAT	I/O	Serial wired debugger data pin. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_DAT pin.
INT0	INT0	I	External interrupt 0 input pin.
INT1	INT1	I	External interrupt 1 input pin.
INT2	INT2	I	External interrupt 2 input pin.
INT3	INT3	I	External interrupt 3 input pin.
INT4	INT4	I	External interrupt 4 input pin.
INT5	INT5	I	External interrupt 5 input pin.
INT6	INT6	I	External interrupt 6 input pin.
INT7	INT7	I	External interrupt 7 input pin.
LPADC0	LPADC0_CH0	A	Low Power ADC0 channel 0 analog input.
	LPADC0_CH1	A	Low Power ADC0 channel 1 analog input.
	LPADC0_CH2	A	Low Power ADC0 channel 2 analog input.
	LPADC0_CH3	A	Low Power ADC0 channel 3 analog input.
	LPADC0_CH4	A	Low Power ADC0 channel 4 analog input.
	LPADC0_CH5	A	Low Power ADC0 channel 5 analog input.
	LPADC0_CH6	A	Low Power ADC0 channel 6 analog input.
	LPADC0_CH7	A	Low Power ADC0 channel 7 analog input.

Group	Pin Name	Type	Description
	LPADC0_CH8	A	Low Power ADC0 channel 8 analog input.
	LPADC0_CH9	A	Low Power ADC0 channel 9 analog input.
	LPADC0_CH10	A	Low Power ADC0 channel 10 analog input.
	LPADC0_CH11	A	Low Power ADC0 channel 11 analog input.
	LPADC0_CH12	A	Low Power ADC0 channel 12 analog input.
	LPADC0_CH13	A	Low Power ADC0 channel 13 analog input.
	LPADC0_CH14	A	Low Power ADC0 channel 14 analog input.
	LPADC0_CH15	A	Low Power ADC0 channel 15 analog input.
	LPADC0_ST	I	Low Power ADC0 external trigger input pin.
LPI2C0	LPI2C0_SCL	I/O	Low Power I2C0 clock pin.
	LPI2C0_SDA	I/O	Low Power I2C0 data input/output pin.
LPIO0	LPIO0	I	Low Power I/O 0 input pin.
LPIO1	LPIO1	I	Low Power I/O 1 input pin.
LPIO2	LPIO2	I	Low Power I/O 2 input pin.
LPIO3	LPIO3	I	Low Power I/O 3 input pin.
LPIO4	LPIO4	I	Low Power I/O 4 input pin.
LPIO5	LPIO5	I	Low Power I/O 5 input pin.
LPIO6	LPIO6	I	Low Power I/O 6 input pin.
LPIO7	LPIO7	I	Low Power I/O 7 input pin.
LPSPI0	LPSPI0_CLK	I/O	Low Power SPI0 serial clock pin.
	LPSPI0_MISO	I/O	Low Power SPI0 MISO (Master In, Slave Out) pin.
	LPSPI0_MOSI	I/O	Low Power SPI0 MOSI (Master Out, Slave In) pin.
	LPSPI0_SS	I/O	Low Power SPI0 slave select pin.
LPTM0	LPTM0	I/O	Low Power Timer0 event counter input/toggle output pin.
	LPTM0_EXT	I/O	Low Power Timer0 external capture input/toggle output pin.
LPTM1	LPTM1	I/O	Low Power Timer1 event counter input/toggle output pin.
	LPTM1_EXT	I/O	Low Power Timer1 external capture input/toggle output pin.
LPUART0	LPUART0_RXD	I	Low Power UART0 data receiver input pin.
	LPUART0_TXD	O	Low Power UART0 data transmitter output pin.
	LPUART0_nCTS	I	Low Power UART0 clear to Send input pin.
	LPUART0_nRTS	O	Low Power UART0 request to Send output pin.
OPA0	OPA0_N0	A	Operational amplifier 0 negative 0 input pin.
	OPA0_N1	A	Operational amplifier 0 negative 1 input pin.
	OPA0_O	A	Operational amplifier 0 output pin.

Group	Pin Name	Type	Description
	OPA0_P0	A	Operational amplifier 0 positive 0 input pin.
	OPA0_P1	A	Operational amplifier 0 positive 1 input pin.
OPA1	OPA1_N0	A	Operational amplifier 1 negative 0 input pin.
	OPA1_N1	A	Operational amplifier 1 negative 1 input pin.
	OPA1_O	A	Operational amplifier 1 output pin.
	OPA1_P0	A	Operational amplifier 1 positive 0 input pin.
	OPA1_P1	A	Operational amplifier 1 positive 1 input pin.
OPA2	OPA2_N0	A	Operational amplifier 2 negative 0 input pin.
	OPA2_N1	A	Operational amplifier 2 negative 1 input pin.
	OPA2_O	A	Operational amplifier 2 output pin.
	OPA2_P0	A	Operational amplifier 2 positive 0 input pin.
	OPA2_P1	A	Operational amplifier 2 positive 1 input pin.
PWM0	PWM0_BRAKE0	I	PWM0 Brake 0 input pin.
	PWM0_BRAKE1	I	PWM0 Brake 1 input pin.
	PWM0_CH0	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH1	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH2	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH3	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH4	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH5	I/O	PWM0 channel 5 output/capture input.
PWM1	PWM1_BRAKE0	I	PWM1 Brake 0 input pin.
	PWM1_BRAKE1	I	PWM1 Brake 1 input pin.
	PWM1_CH0	I/O	PWM1 channel 0 output/capture input.
	PWM1_CH1	I/O	PWM1 channel 1 output/capture input.
	PWM1_CH2	I/O	PWM1 channel 2 output/capture input.
	PWM1_CH3	I/O	PWM1 channel 3 output/capture input.
	PWM1_CH4	I/O	PWM1 channel 4 output/capture input.
	PWM1_CH5	I/O	PWM1 channel 5 output/capture input.
QSPI0	QSPI0_CLK	I/O	Quad SPI0 serial clock pin.
	QSPI0_MISO0	I/O	Quad SPI0 MISO0 (Master In, Slave Out) pin.
	QSPI0_MISO1	I/O	Quad SPI0 MISO1 (Master In, Slave Out) pin.
	QSPI0_MOSI0	I/O	Quad SPI0 MOSI0 (Master Out, Slave In) pin.
	QSPI0_MOSI1	I/O	Quad SPI0 MOSI1 (Master Out, Slave In) pin.
	QSPI0_SS	I/O	Quad SPI0 slave select pin.

Group	Pin Name	Type	Description
SPI0	SPI0_CLK	I/O	SPI0 serial clock pin.
	SPI0_I2SMCLK	I/O	SPI0 I2S master clock output pin
	SPI0_MISO	I/O	SPI0 MISO (Master In, Slave Out) pin.
	SPI0_MOSI	I/O	SPI0 MOSI (Master Out, Slave In) pin.
	SPI0_SS	I/O	SPI0 slave select pin.
SPI1	SPI1_CLK	I/O	SPI1 serial clock pin.
	SPI1_I2SMCLK	I/O	SPI1 I2S master clock output pin
	SPI1_MISO	I/O	SPI1 MISO (Master In, Slave Out) pin.
	SPI1_MOSI	I/O	SPI1 MOSI (Master Out, Slave In) pin.
	SPI1_SS	I/O	SPI1 slave select pin.
SPI2	SPI2_CLK	I/O	SPI2 serial clock pin.
	SPI2_I2SMCLK	I/O	SPI2 I2S master clock output pin
	SPI2_MISO	I/O	SPI2 MISO (Master In, Slave Out) pin.
	SPI2_MOSI	I/O	SPI2 MOSI (Master Out, Slave In) pin.
	SPI2_SS	I/O	SPI2 slave select pin.
SPI3	SPI3_CLK	I/O	SPI3 serial clock pin.
	SPI3_I2SMCLK	I/O	SPI3 I2S master clock output pin
	SPI3_MISO	I/O	SPI3 MISO (Master In, Slave Out) pin.
	SPI3_MOSI	I/O	SPI3 MOSI (Master Out, Slave In) pin.
	SPI3_SS	I/O	SPI3 slave select pin.
TAMPER0	TAMPER0	I/O	TAMPER detector loop pin 0.
TAMPER1	TAMPER1	I/O	TAMPER detector loop pin 1.
TAMPER2	TAMPER2	I/O	TAMPER detector loop pin 2.
TK	TK_SE	I/O	Touch key (shielding electrode)
	TK_TK0	I/O	Touch key 0
	TK_TK1	I/O	Touch key 1
	TK_TK2	I/O	Touch key 2
	TK_TK3	I/O	Touch key 3
	TK_TK4	I/O	Touch key 4
	TK_TK5	I/O	Touch key 5
	TK_TK6	I/O	Touch key 6
	TK_TK7	I/O	Touch key 7
	TK_TK8	I/O	Touch key 8
	TK_TK9	I/O	Touch key 9

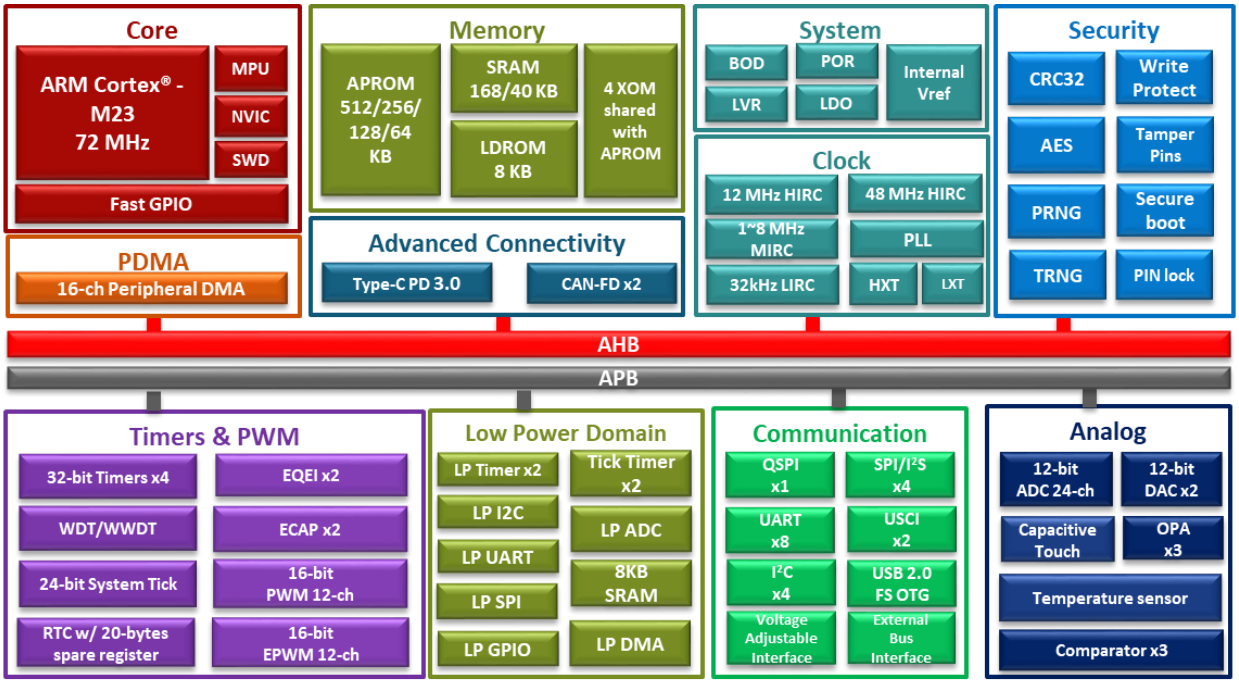
Group	Pin Name	Type	Description
	TK_TK10	I/O	Touch key 10
	TK_TK11	I/O	Touch key 11
	TK_TK12	I/O	Touch key 12
	TK_TK13	I/O	Touch key 13
	TK_TK14	I/O	Touch key 14
	TK_TK15	I/O	Touch key 15
	TK_TK16	I/O	Touch key 16
	TK_TK17	I/O	Touch key 17
TM0	TM0	I/O	Timer0 event counter input/toggle output pin.
	TM0_EXT	I/O	Timer0 external capture input/toggle output pin.
TM1	TM1	I/O	Timer1 event counter input/toggle output pin.
	TM1_EXT	I/O	Timer1 external capture input/toggle output pin.
TM2	TM2	I/O	Timer2 event counter input/toggle output pin.
	TM2_EXT	I/O	Timer2 external capture input/toggle output pin.
TM3	TM3	I/O	Timer3 event counter input/toggle output pin.
	TM3_EXT	I/O	Timer3 external capture input/toggle output pin.
UART0	UART0_RXD	I	UART0 data receiver input pin.
	UART0_TXD	O	UART0 data transmitter output pin.
	UART0_nCTS	I	UART0 clear to Send input pin.
	UART0_nRTS	O	UART0 request to Send output pin.
UART1	UART1_RXD	I	UART1 data receiver input pin.
	UART1_TXD	O	UART1 data transmitter output pin.
	UART1_nCTS	I	UART1 clear to Send input pin.
	UART1_nRTS	O	UART1 request to Send output pin.
UART2	UART2_RXD	I	UART2 data receiver input pin.
	UART2_TXD	O	UART2 data transmitter output pin.
	UART2_nCTS	I	UART2 clear to Send input pin.
	UART2_nRTS	O	UART2 request to Send output pin.
UART3	UART3_RXD	I	UART3 data receiver input pin.
	UART3_TXD	O	UART3 data transmitter output pin.
	UART3_nCTS	I	UART3 clear to Send input pin.
	UART3_nRTS	O	UART3 request to Send output pin.
UART4	UART4_RXD	I	UART4 data receiver input pin.
	UART4_TXD	O	UART4 data transmitter output pin.

Group	Pin Name	Type	Description
	UART4_nCTS	I	UART4 clear to Send input pin.
	UART4_nRTS	O	UART4 request to Send output pin.
UART5	UART5_RXD	I	UART5 data receiver input pin.
	UART5_TXD	O	UART5 data transmitter output pin.
	UART5_nCTS	I	UART5 clear to Send input pin.
	UART5_nRTS	O	UART5 request to Send output pin.
UART6	UART6_RXD	I	UART6 data receiver input pin.
	UART6_TXD	O	UART6 data transmitter output pin.
	UART6_nCTS	I	UART6 clear to Send input pin.
	UART6_nRTS	O	UART6 request to Send output pin.
UART7	UART7_RXD	I	UART7 data receiver input pin.
	UART7_TXD	O	UART7 data transmitter output pin.
	UART7_nCTS	I	UART7 clear to Send input pin.
	UART7_nRTS	O	UART7 request to Send output pin.
USB	USB_D+	A	USB differential signal D+.
	USB_D-	A	USB differential signal D-.
	USB_OTG_ID	I	USB_ identification.
	USB_VBUS	P	Power supply from USB host or HUB.
	USB_VBUS_EN	O	USB external VBUS regulator enable pin.
	USB_VBUS_ST	I	USB external VBUS regulator status pin.
USCI0	USCI0_CLK	I/O	USCI0 clock pin.
	USCI0_CTL0	I/O	USCI0 control 0 pin.
	USCI0_CTL1	I/O	USCI0 control 1 pin.
	USCI0_DAT0	I/O	USCI0 data 0 pin.
	USCI0_DAT1	I/O	USCI0 data 1 pin.
USCI1	USCI1_CLK	I/O	USCI1 clock pin.
	USCI1_CTL0	I/O	USCI1 control 0 pin.
	USCI1_CTL1	I/O	USCI1 control 1 pin.
	USCI1_DAT0	I/O	USCI1 data 0 pin.
	USCI1_DAT1	I/O	USCI1 data 1 pin.
UTCPD0	UTCPD0_CC1	A	UTCPD0 USB Type-C Configuration Channel pin 1
	UTCPD0_CC2	A	UTCPD0 USB Type-C Configuration Channel pin 2
	UTCPD0_CCDB1	A	UTCPD0 USB Type-C Configuration Channel pin D
	UTCPD0_CCDB2	A	UTCPD0 USB Type-C Configuration Channel pin D

Group	Pin Name	Type	Description
	UTCPD0_DISCHG	O	UTCPD0 VCONN Discharge Enable output pin
	UTCPD0_FRSTX1	O	UTCPD0 CC1 Fast Role Swap Transmit output pin
	UTCPD0_FRSTX2	O	UTCPD0 CC2 Fast Role Swap Transmit output pin
	UTCPD0_VBDCHG	O	UTCPD0 VBUS Discharge Enable output pin
	UTCPD0_VBSNKEN	O	UTCPD0 VBUS Sink Enable output pin
	UTCPD0_VBSRCEN	O	UTCPD0 VBUS Source Enable output pin
	UTCPD0_VCNEN1	O	UTCPD0 CC1 VCONN Enable output pin
	UTCPD0_VCNEN2	O	UTCPD0 CC2 VCONN Enable output pin
X32	X32_IN	I	External 32.768 kHz crystal input pin.
	X32_OUT	O	External 32.768 kHz crystal output pin.
XT1	XT1_IN	I	External 4~24 MHz (high speed) crystal input pin.
	XT1_OUT	O	External 4~24 MHz (high speed) crystal output pin.
Power	AV_{DD}	P	Power supply for internal analog circuit.
	AV_{SS}	P	Ground pin for analog circuit.
	LDO_CAP	A	LDO output pin. **Note:** This pin needs to be connected with a capacitor whose value can be found in General operating conditions table in Datasheet.
	V_{BAT}	P	Power supply by batteries for RTC.
	V_{DD}	P	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
	V_{DDIO}	P	Power supply for PA.0~PA.5.
	V_{REF}	A	ADC reference voltage input. **Note:** This pin needs to be connected with a 1uF capacitor.
	V_{SS}	P	Ground pin for digital circuit.
	nRESET	I	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state. **Note:** It is recommended to use 10 kΩ pull-up resistor and 10 uF capacitor on nRESET pin.

5 BLOCK DIAGRAM

5.1 M2L31 Series Block Diagram



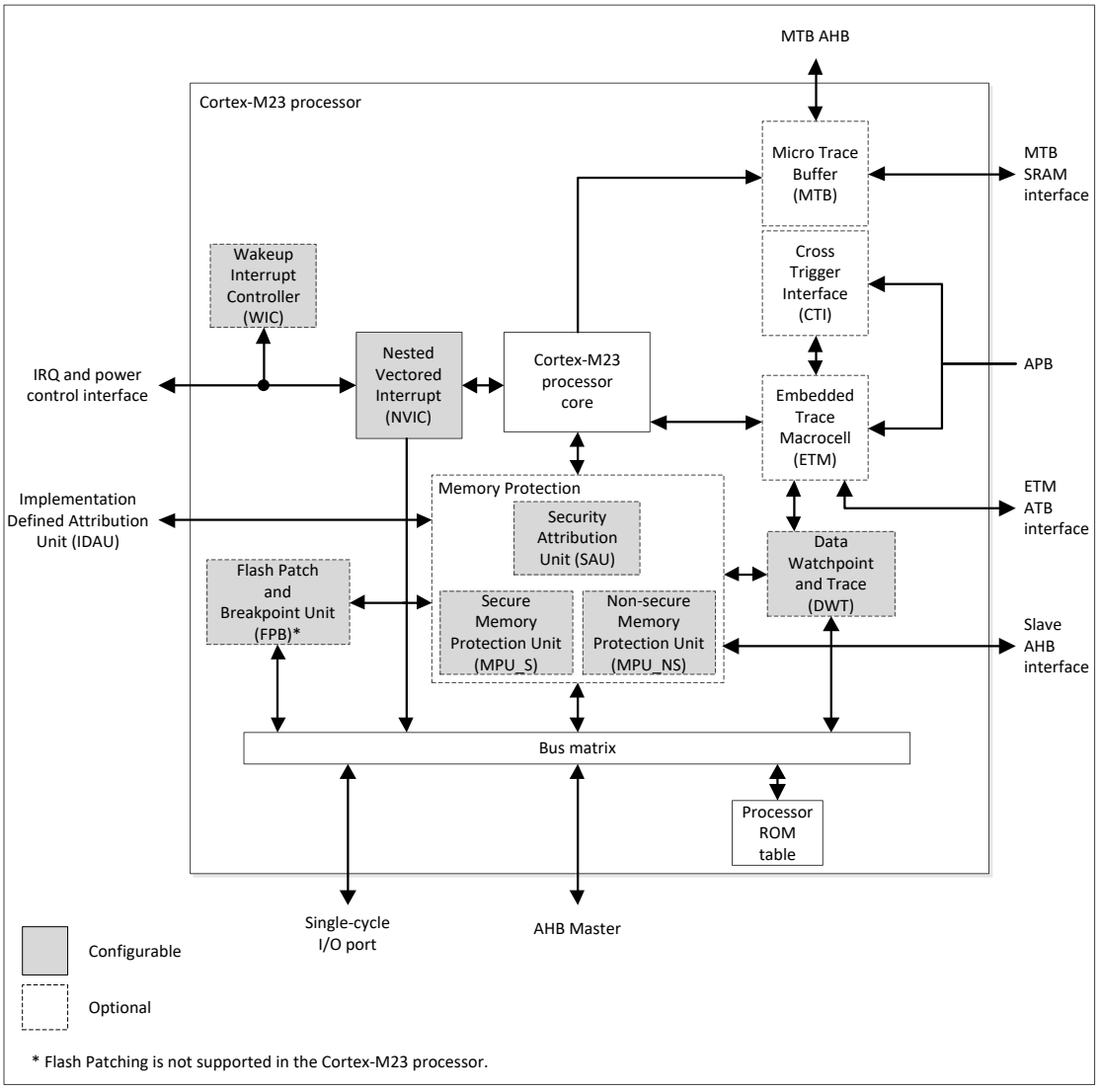
Please refer to the selection guide in section 4.2 for detailed information.

Figure 5.1-1 M2L31 Series Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 Arm Cortex-M23 Core

The Cortex-M23 processor is a low gate count, two-stage, and highly energy efficient 32-bit RISC processor, which has an AMBA AHB5 interface supporting Arm® TrustZone technology, a debug access port supporting serial wire debug and single-cycle I/O ports. It has an NVIC component and MPU for memory-protection functionality. The processor also supports Security Extension. The NuMicro M2L31 is embedded with Cortex-M23 processor.Figure 6.1-1 shows the functional controller of the processor.



Note: The M2L31 does not support security attribution unit, ETM, CTI and MTB function.

Figure 6.1-1 Cortex-M23 Block Diagram

Cortex-M23 processor features:

- Armv8-M Baseline architecture.
- Armv8-M Baseline Thumb-2 instruction set that combines high code density with 32-bit performance.
- Support for single-cycle I/O access.
- Power control optimization of system components.
- Integrated sleep modes for low power consumption.
- Optimized code fetching for reduced ReRAM and ROM power consumption.
- A 32-bit Single cycle Hardware multiplier.
- A 32-bit Hardware divider.
- Deterministic, high-performance interrupt handling for time-critical applications.
- Deterministic instruction cycle timing.
- Support for system level debug authentication.
- Support for Arm® Debug Interface Architecture ADIv5.1 Serial Wire Debug (SWD).
- ETM for instruction trace.
- Separated privileged and unprivileged modes.
- Security Extension supporting a Secure and a Non-secure state.
- Protected Memory System Architecture (PMSAv8) Memory Protection Units (MPUs) for both Secure and Non-secure states.
- Security Attribution Unit (SAU).
- SysTick timers for both Secure and Non-secure states.
- A Nested Vectored Interrupt Controller (NVIC) closely integrated with the processor with up to 240 interrupts.

6.2 System Manager

6.2.1 Overview

System management includes the following sections:

- System Reset
- System Power Distribution
- SRAM Memory Orginization
- System Timer (SysTick)
- Nested Vectored Interrupt Controller (NVIC)
- System Control register

6.2.2 System Reset

The system reset can be issued by one of the events listed below. These reset event flags can be read from SYS_RSTSTS register to determine the reset source. Hardware reset sources are from peripheral signals. Software reset can trigger reset through setting control registers.

- Hardware Reset Sources
  - Power-on Reset
  - Low level on the nRESET pin
  - Watchdog Time-out Reset and Window Watchdog Reset (WDT/WWDT Reset)
  - Low Voltage Reset (LVR)
  - Brown-out Detector Reset (BOD Reset)
  - CPU Lockup Reset
- Software Reset Sources
  - CHIP Reset will reset whole chip by writing 1 to CHIPRST (SYS_IPRST0[0])
  - MCU Reset to reboot but keeping the booting setting from APROM or LDROM by writing 1 to SYSRESETREQ (AIRCR[2])
  - CPU Reset for Cortex-M23 core only by writing 1 to CPURST (SYS_IPRST0[1])

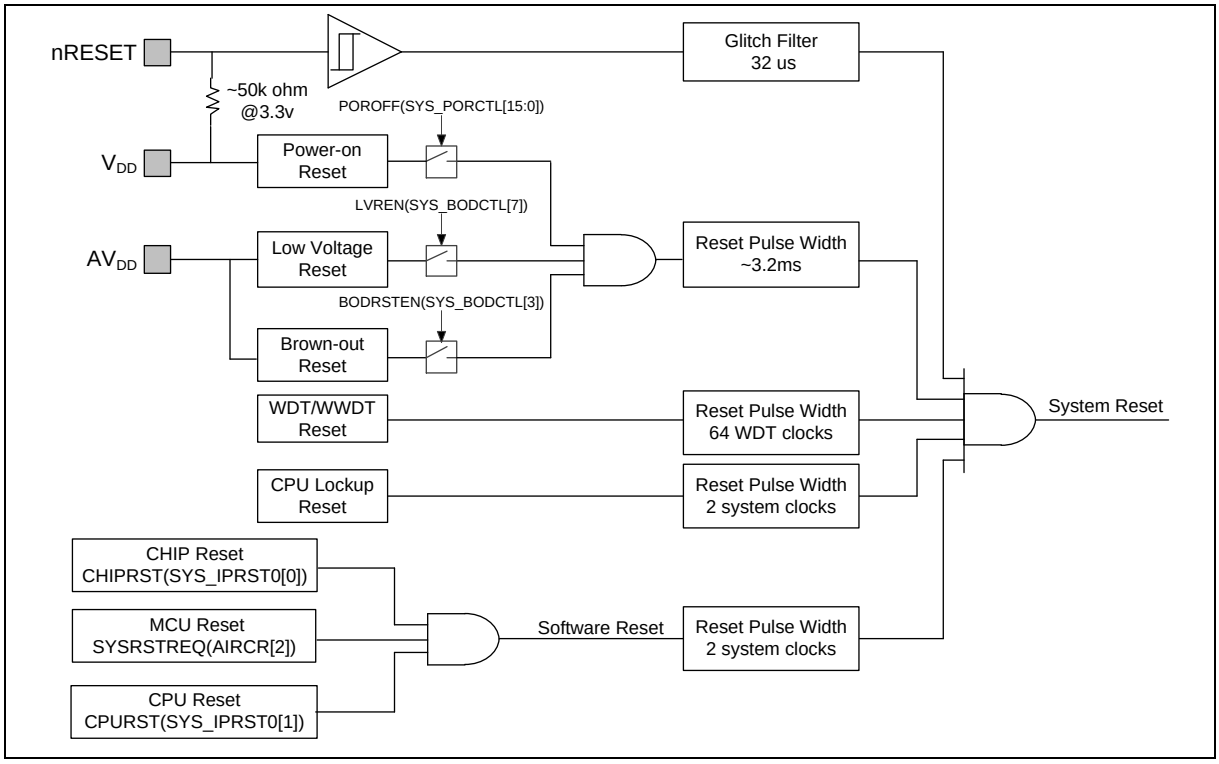


Figure 6.2-1 System Reset Sources

There are a total of 9 reset sources in the NuMicro family. In general, CPU reset is used to reset Cortex-M23 only; the other reset sources will reset Cortex-M23 and all peripherals. However, there are small differences between each reset source and they are listed in Table 6.2-1.

Reset Sources Register	POR	nRESET	WDT	LVR	BOD	Lockup	CHIP	MCU	CPU
SYS_RSTSTS	Bit 0 = 1	Bit 1 = 1	Bit 2 = 1	Bit 3 = 1	Bit 4 = 1	Bit 8 = 1	Bit 0 = 1	Bit 5 = 1	Bit 7 = 1
CHIPRST (SYS_IPRST0[0])	0x0	-	-	-	-	-	-	-	-
BODEN (SYS_BODCTL[0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-
BODVL (SYS_BODCTL[2:1])									
BODRSTEN (SYS_BODCTL[3])									
LXTEN (CLK_PWRCTL[1])	0x0	-	-	-	-	-	-	-	-
WDTCKEN (CLK_APBCLK0[0])	0x1	-	0x1	-	-	-	0x1	-	-
WDTSEL (CLK_CLKSEL1[1:0])	0x3	0x3	-	-	-	-	-	-	-
HXTSTB	0x0	-	-	-	-	-	-	-	-

(CLK_STATUS[0])									
LXTSTB (CLK_STATUS[1])	0x0	-	-	-	-	-	-	-	-
PLLSTB (CLK_STATUS[2])	0x0	-	-	-	-	-	-	-	-
HIRCSTB (CLK_STATUS[4])	0x0	-	-	-	-	-	-	-	-
CLKSFAIL (CLK_STATUS[7])	0x0	0x0	-	-	-	-	-	-	-
RSTEN (WDT_CTL[1]) WDTEN (WDT_CTL[7])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
WDT_CTL except bit 1 and bit 7.	0x0700	0x0700	0x0700	0x0700	0x0700	-	0x0700	-	-
WDT_ALTCTL	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_RLDCNT	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_CTL	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	-	0x3F0800	-	-
WWDT_STATUS	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_CNT	0x3F	0x3F	0x3F	0x3F	0x3F	-	0x3F	-	-
BS (RMC_ISPCTL[1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
DFBA (RMC_DFBA[31:0])	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	-	Reload from CONFIG1	-	-
CBS (RMC_ISPSTS[2])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
VECMAP (RMC_ISPSTS[23:9])	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	-	Reload base on CONFIG0	-	-
APPROENn (RMC_APWPROT0[n]) N=0, 1, …, 31	Reload base on CONFIG8	Reload base on CONFIG8	Reload base on CONFIG8	Reload base on CONFIG8	Reload base on CONFIG8	-	Reload base on CONFIG8	-	-
APPROENn (RMC_APWPROT1[n]) N=0, 1, …, 31	Reload base on CONFIG9	Reload base on CONFIG9	Reload base on CONFIG9	Reload base on CONFIG9	Reload base on CONFIG9	-	Reload base on CONFIG9	-	-
RMC_APWPKEEP	Reload base on CONFIG10	Reload base on CONFIG10	Reload base on CONFIG10	Reload base on CONFIG10	Reload base on CONFIG10	Reload base on CONFIG10	Reload base on CONFIG10	Reload base on CONFIG10	Reload base on CONFIG10
SCACT (RMC_SCACT[0])	0x0	0x0	0x0	0x0	0x0	-	0x0	-	-

Other Peripheral Registers	Reset Value	-
RMC Registers	Reset Value	
Note: '-' means that the value of register keeps original setting.		

Table 6.2-1 Reset Value of Registers

6.2.2.1 nRESET Reset

The nRESET reset means to generate a reset signal by pulling low nRESET pin, which is an asynchronous reset input pin and can be used to reset system at any time. When the nRESET voltage is lower than 0.3 V_{DD} and the state keeps longer than 32 us (glitch filter), chip will be reset. The nRESET reset will control the chip in reset state until the nRESET voltage rises above 0.7 V_{DD} and the state keeps longer than 32 us (glitch filter). The PINRF(SYS_RSTSTS[1]) will be set to 1 if the previous reset source is nRESET reset. Figure 6.2-2 shows the nRESET reset waveform.

Note: Please refer to the datasheet for T_{RP} value.

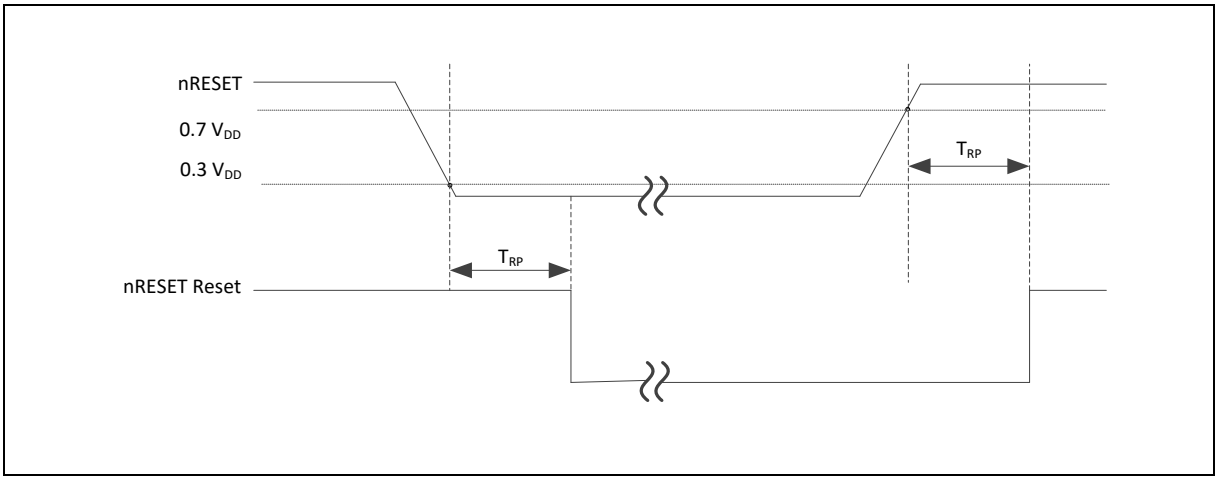


Figure 6.2-2 nRESET Reset Waveform

6.2.2.2 Power-on Reset (POR)

The Power-on reset (POR) is used to generate a stable system reset signal and forces the system to be reset when power-on to avoid unexpected behavior of MCU. When applying the power to MCU, the POR module will detect the rising voltage and generate reset signal to system until the voltage is ready for MCU operation. At POR reset, the PORF(SYS_RSTSTS[0]) will be set to 1 to indicate there is a POR reset event. The PORF(SYS_RSTSTS[0]) bit can be cleared by writing 1 to it. Figure 6.2-3 shows the power-on reset waveform.

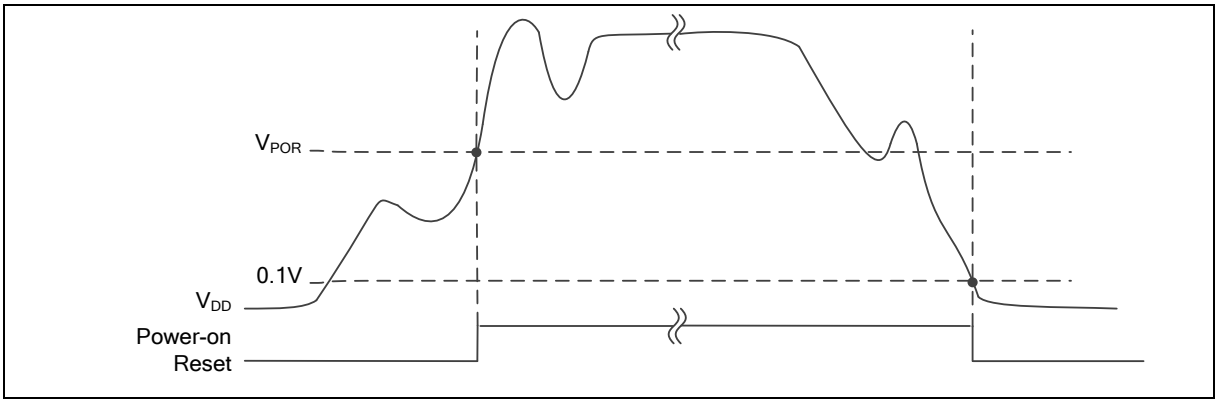


Figure 6.2-3 Power-on Reset (POR) Waveform

6.2.2.3 Low Voltage Reset (LVR)

If the Low Voltage Reset function is enabled by setting the Low Voltage Reset Enable Bit LVREN (SYS_BODCTL[7]) to 1, after 100us delay, LVR detection circuit will be stable and the LVR function will be active. Then LVR function will detect AV_{DD} during system operation. When the AV_{DD} voltage is lower than V_{LVR} and the state keeps longer than De-glitch time set by LVRDGSEL (SYS_BODCTL[14:12]), chip will be reset. The LVR reset will control the chip in reset state until the AV_{DD} voltage rises above V_{LVR} and the state keeps longer than De-glitch time set by LVRDGSEL (SYS_BODCTL[14:12]). The default setting of Low Voltage Reset is enabled without De-glitch function. Figure 6.2-4 shows the Low Voltage Reset waveform.

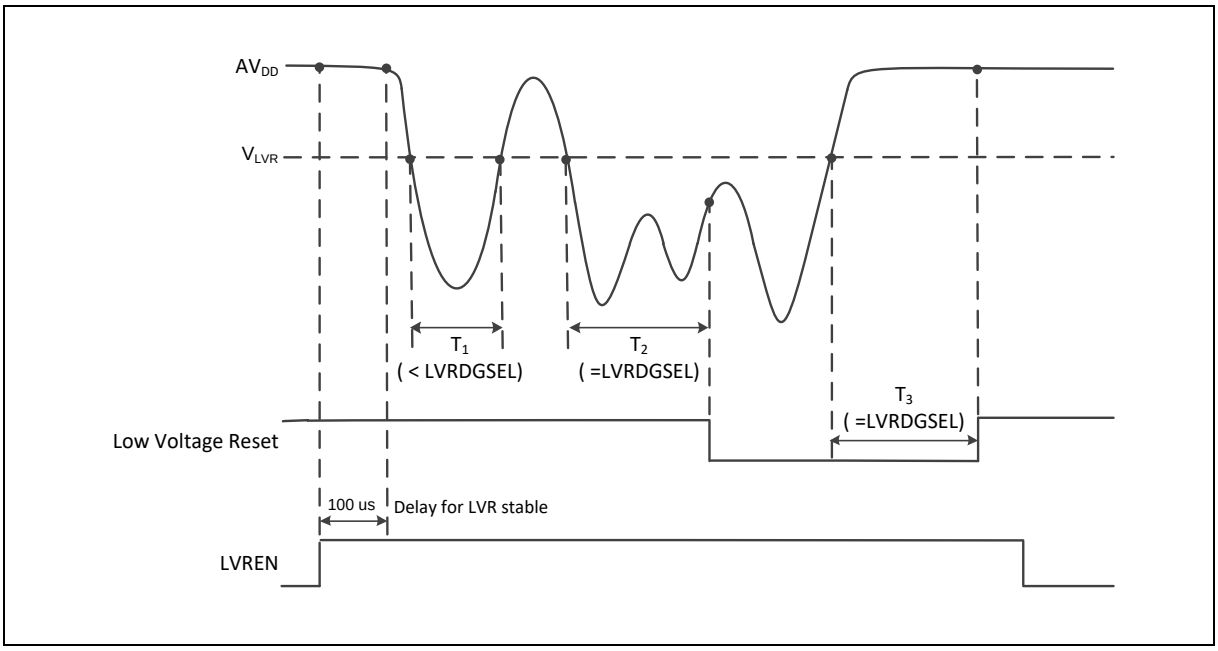


Figure 6.2-4 Low Voltage Reset (LVR) Waveform

6.2.2.4 Brown-out Detector Reset (BOD Reset)

If the Brown-out Detector (BOD) function is enabled by setting the Brown-out Detector Enable Bit BODEN (SYS_BODCTL[0]), Brown-out Detector function will detect AV_{DD} during system operation. When the AV_{DD} voltage is lower than V_{BOD} which is decided by BODEN and BODVL (SYS_BODCTL[18:16]) and the state keeps longer than De-glitch time set by BODDGSEL (SYS_BODCTL[10:8]), chip will be reset. The BOD reset will control the chip in reset state until the AV_{DD} voltage rises above V_{BOD} and the state keeps longer than De-glitch time set by BODDGSEL. The default

value of BODEN, BODVL and BODRSTEN (SYS_BODCTL[3]) is set by RRAM controller user configuration register CBODEN (CONFIG0 [19]), CBOV (CONFIG0 [23:21]) and CBORST(CONFIG0[20]) respectively. User can determine the initial BOD setting by setting the CONFIG0 register. Figure 6.2-5 shows the Brown-out Detector waveform.

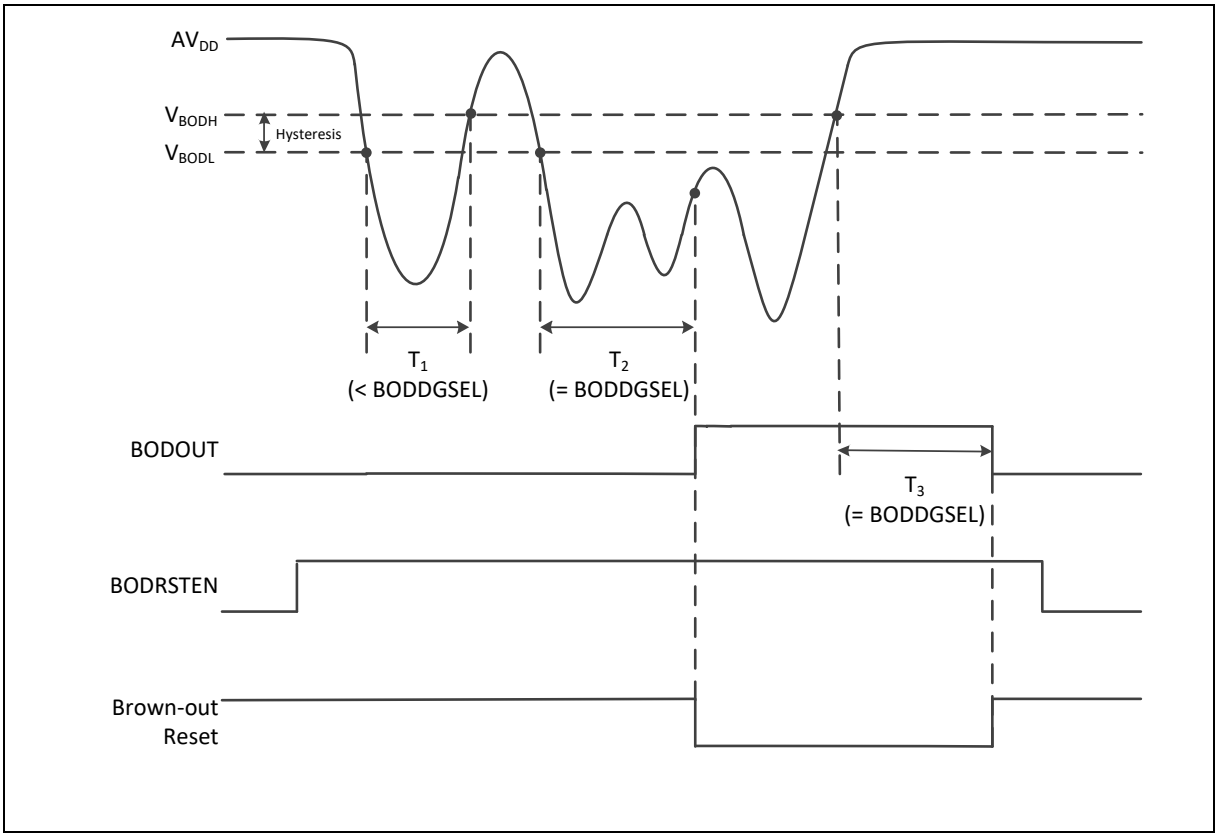


Figure 6.2-5 Brown-out Detector (BOD) Waveform

6.2.2.5 Watchdog Timer Reset (WDT)

In most industrial applications, system reliability is very important. To automatically recover the MCU from failure status is one way to improve system reliability. The watchdog timer(WDT) is widely used to check if the system works fine. If the MCU is crashed or out of control, it may cause the watchdog time-out. User may decide to enable system reset during watchdog time-out to recover the system and take action for the system crash/out-of-control after reset.

Software can check if the reset is caused by watchdog time-out to indicate the previous reset is a watchdog reset and handle the failure of MCU after watchdog time-out reset by checking WDTRF(SYS_RSTSTS[2]).

6.2.2.6 CPU Lockup Reset

CPU enters lockup status after CPU produces hardfault at hardfault handler and chip gives immediate indication of seriously errant kernel software. This is the result of the CPU being locked because of an unrecoverable exception following the activation of the processor's built in system state protection hardware. When chip enters debug mode, the CPU lockup reset will be ignored.

6.2.2.7 CPU Reset, CHIP Reset and MCU Reset

The CPU Reset means only Cortex-M23 core is reset and all other peripherals remain the same status after CPU reset. User can set the CPURST(SYS_IPRST0[1]) to 1 to assert the CPU Reset signal.

The CHIP Reset is same with Power-on Reset. The CPU and all peripherals are reset and BS(RMC_ISPCTL[1]) bit is automatically reloaded from CONFIG0 setting. User can set the

CHIPRST(SYS_IPRST0[0]) to 1 to assert the CHIP Reset signal.

The MCU Reset is similar with CHIP Reset. The difference is that BS(RMC_ISPCTL[1]) will not be reloaded from CONFIG0 setting and keep its original software setting for booting from APROM or LDROM. User can set the SYSRESETREQ(AIRCR[2]) to 1 to assert the MCU Reset.

6.2.3 System Power Distribution

In this chip, power distribution is divided into four segments:

- Analog power from AV_{DD} and AV_{SS} provides the power for analog components operation.
- Digital power from V_{DD} and V_{SS} supplies the power to the internal regulator for digital operation and I/O pins.
- I/O power from V_{DDIO} supplies the power for PA.0 ~ PA.5
- RTC power from V_{BAT} provides the power for RTC and 80 bytes backup registers.

The output of internal voltage regulators, LDO_CAP, require an external capacitor which should be located close to the corresponding pin. Analog power (AV_{DD}) should be the same voltage level of the digital power (V_{DD}). Figure 6.2-6 shows the NuMicro M2L31 power distribution. In Figure 6.2-6, the block of Low Power Digital Logic includes LPPDMA, LPGPIO, LPADC, LPI2C, LPSPI, LPUART, LPTMR, TTMR, WDT and OPA peripherals.

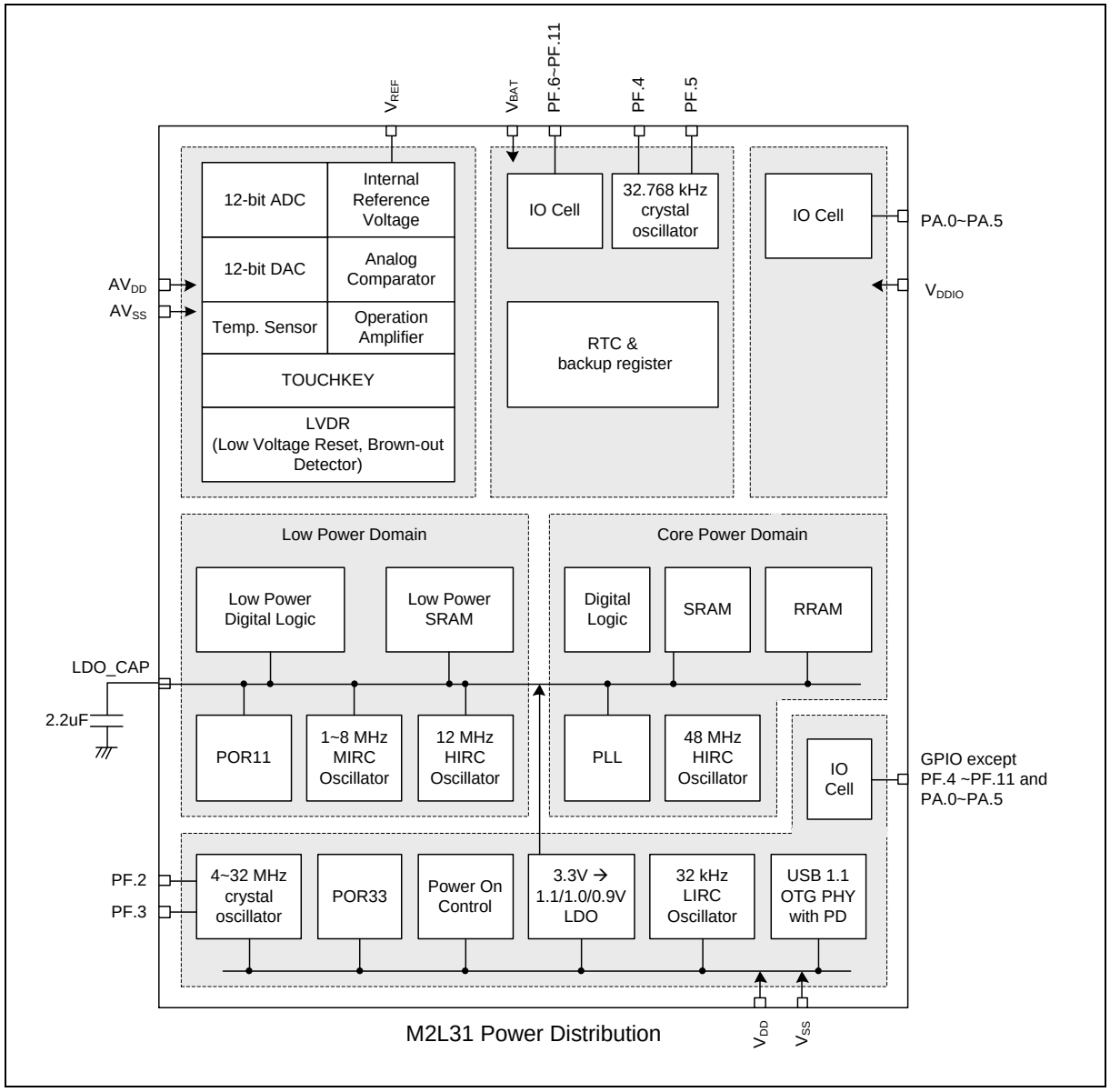


Figure 6.2-6 NuMicro M2L31 Power Distribution Diagram

6.2.4 Power Modes and Wake-up Sources

The NuMicro M2L31 series has power manager unit to support several operating modes for saving power. Table 6.2-2 lists all power mode at NuMicro M2L31 series.

Mode	CPU Status	LDO_CAP (V)	Clock Status
PL1 run mode	Normal Run	1.1	All clocks can be disabled by control register.
Idle mode	Sleep mode	1.1	Only CPU clock is disabled.
Normal Power-down mode 0 (NPD0)	Deep Sleep mode	1.1	Most clocks are disabled except HIRC/MIRC/LIRC/LXT. RTC, TMR, UART and low power domain peripherals could be active if HIRC/MIRC/LIRC/LXT are selected as clock sources.
Normal Power-down mode 1 (NPD1)	Deep Sleep mode	1.1	Most clocks are disabled except HIRC/MIRC/LIRC/LXT. RTC, TMR, UART and low power domain peripherals could be active if HIRC/MIRC/LIRC/LXT are selected as clock sources.
Normal Power-down mode 2 (NPD2)	Deep Sleep mode	0.9	Most clocks are disabled except HIRC/MIRC/LIRC/LXT. RTC, TMR, UART and low power domain peripherals could be active if HIRC/MIRC/LIRC/LXT are selected as clock sources.
Normal Power-down mode 3 (NPD3)	Power off	1.1	Most clocks are disabled except HIRC/MIRC/LIRC/LXT. RTC and low power domain peripherals could be active if HIRC/MIRC/LIRC/LXT are selected as clock sources.
Normal Power-down mode 4 (NPD4)	Power off	0.9	Most clocks are disabled except HIRC/MIRC/LIRC/LXT. RTC and low power domain peripherals could be active if HIRC/MIRC/LIRC/LXT are selected as clock sources.
Normal Power-down mode 5 (NPD5)	Power off	0.7	LXT is enabled for RTC function and LIRC is enabled for wake-up timer usage.
Standby Power-down mode 0 (SPD0)	Power off	1.1	LXT is enabled for RTC function and LIRC is enabled for wake-up timer usage.
Standby Power-down mode 1 (SPD1)	Power off	0.9	LXT is enabled for RTC function and LIRC is enabled for wake-up timer usage.
Standby Power-down mode 2 (SPD2)	Power off	0.7	LXT is enabled for RTC function and LIRC is enabled for wake-up timer usage.
Deep Power-down mode 0 (DPD0)	Power off	Floating	LXT is enabled for RTC function and LIRC is enabled for wake-up timer usage.
Deep Power-down mode 1 (DPD1)	Power off	Floating	LXT is enabled for RTC function and LIRC is enabled for wake-up timer usage.

Table 6.2-2 Power Mode Table

Mode	Core Power Domain AHB/APB Bus Maximum Speed (MHz (HCLK0/PCLK0/PCLK1)	Low Power Domain AHB/APB Bus Maximum Speed (MHz) (HCLK1/PCLK2)
PL1 run mode	72	24
NPD0 mode	Off	12

NPD1 mode	Off	12
NPD2 mode	Off	4
NPD3 mode	Off	12
NPD4 mode	Off	4
NPD5 mode	Off	Off

Table 6.2-3 AHB/APB Bus maximum speed under different Power Modes

Table 6.2-3 shows the maximum speed of AHB/APB bus in Core Power and Low Power domain, different power mode.

There are different power mode entry setting. For each power mode, they have different entry setting and leaving condition. Table 6.2-4 shows the entry setting for each power mode. When chip power-on, chip is running at normal mode. User can enter each mode by setting SLEEPDEEP (SCR[2]), PDEN (CLK_PWRCTL[7]) and PDMSEL (CLK_PMUCTL[2:0]) and execute WFI instruction.

Register/Instruction Mode	SLEEPDEEP (SCR[2])	PDEN (CLK_PWRCTL[7])	PDMSEL (CLK_PMUCTL[2:0])	CPU Run WFI Instruction
Normal mode	0	0	0	NO
Idle mode	0	0	0	YES
Normal Power-down mode 0 (NPD0)	1	1	0	YES
Normal Power-down mode 1 (NPD1)	1	1	1	YES
Normal Power-down mode 2 (NPD2)	1	1	2	YES
Normal Power-down mode 3 (NPD3)	1	1	3	YES
Normal Power-down mode 4 (NPD4)	1	1	4	YES
Normal Power-down mode 5 (NPD5)	1	1	5	YES
Standby Power-down mode 0 (SPD0)	1	1	8	YES
Standby Power-down mode 1 (SPD1)	1	1	9	YES
Standby Power-down mode 2 (SPD2)	1	1	10	YES
Deep Power-down mode 0 (DPD0)	1	1	12	YES
Deep Power-down mode 1 (DPD1)	1	1	13	YES

Table 6.2-4 Power Mode Difference Table

There are several wake-up sources in Idle mode and Power-down mode. Table 6.2-5 lists the available clocks for each power mode. When chip wake-up from standby Power-down mode 0/1/2 (SPD0/1/2) or deep Power-down mode 0/1 (DPD0/1), the CPU will be reset and start running.

Power Mode	Normal Mode	Idle Mode	Power-Down Mode
Definition	CPU is in active state	CPU is in sleep state	CPU is in sleep state and all clocks stop except LXT and LIRC. SRAM content retended.
Entry Condition	Chip is in normal mode after system reset released	CPU executes WFI instruction.	CPU sets sleep mode enable and power down enable and executes WFI instruction.
Wake-up Sources	N/A	All interrupts	RTC, WDT, I²C, Timer, UART, BOD, GPIO, EINT, USCI, USBD, ACMP and BOD.
Available Clocks	All	All except CPU clock	LXT and LIRC
After Wake-up	N/A	CPU back to normal mode	CPU back to normal mode

Table 6.2-5 Power Mode Definition Table

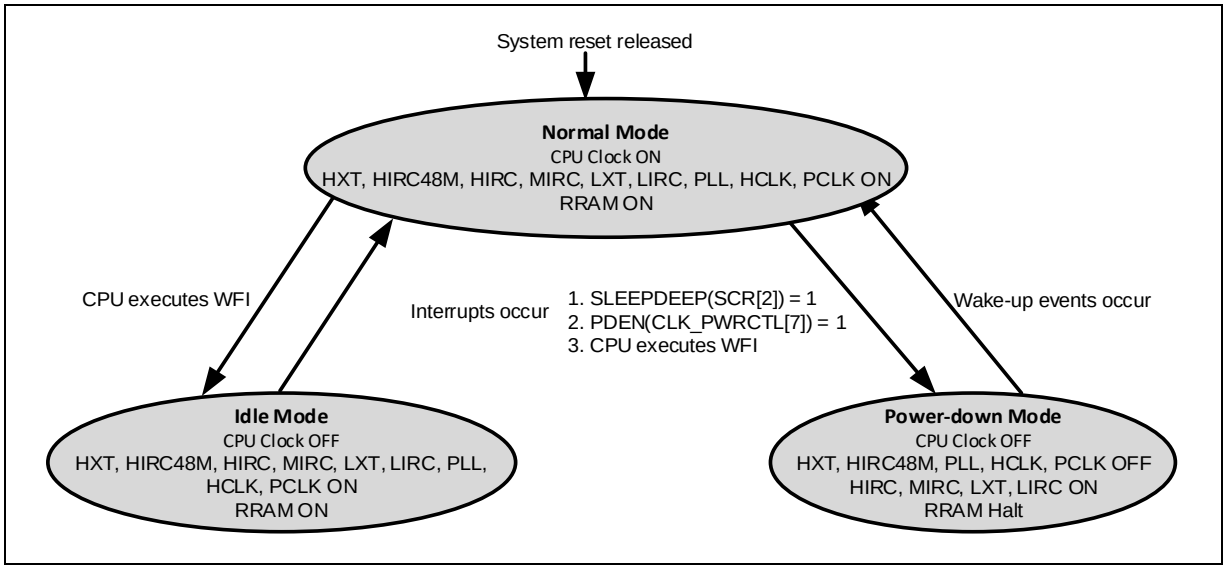


Figure 6.2-7 Power Mode State Machine

Clocks	Idle Mode	NPD0, NPD1,NPD2	NPD3, NPD4	NPD5, SPD0~2	DPD0, DPD1
HXT	ON/OFF	Halt	Halt	Halt	Halt
HIRC48M	ON/OFF	Halt	Halt	Halt	Halt
HIRC	ON/OFF	ON/OFF[1]	ON/OFF[1]	Halt	Halt
MIRC	ON/OFF	ON/OFF[1]	ON/OFF[1]	Halt	Halt
LXT	ON/OFF	ON/OFF[1]	ON/OFF[1]	ON/OFF[1]	ON/OFF[1]
LIRC	ON/OFF	ON/OFF[1]	ON/OFF[1]	ON/OFF[1]	ON/OFF[1],[6]
PLL	ON/OFF	Halt	Halt	Halt	Halt
HCLK0/PCLK0/PCLK1	ON	Halt	Halt	Halt	Halt
HCLK1/PCLK2	ON/OFF	ON/OFF	ON/OFF	Halt	Halt
CPU	Halt	Halt	Halt	Halt	Halt
SRAM	ON	Halt	Halt	Halt	Halt
LPSRAM	ON/OFF	ON/OFF	ON/OFF	Halt	Halt
RRAM	ON	Halt	Halt	Halt	Halt
TIMER	ON	ON/OFF[2]	Halt	Halt	Halt
WDT	ON	ON/OFF[3]	ON/OFF[3]	Halt	Halt
RTC	ON	ON/OFF[4]	ON/OFF[4]	ON/OFF[4]	ON/OFF[4]
UART	ON	ON/OFF[5]	Halt	Halt	Halt
Low power domain peripherals	ON	ON/OFF[7]	ON/OFF[7]	Halt	Halt
Others	ON	Halt	Halt	Halt	Halt

Table 6.2-6 Clocks in Power Modes

Note:

1. HIRC, MIRC, LXT, LIRC ON or OFF depends on SOFTWARE setting in normal mode.
2. If TIMER clock source is selected as LIRC/LXT and LIRC/LXT is enabled.
3. If WDT clock source is selected as LIRC and LIRC is enabled.
4. If RTC clock source is selected as LXT and LXT is enabled.
5. If UART clock source is selected as LXT and LXT is enabled.
6. If timer wake up function is disabled, LIRC will be disabled automatically when chip enter DPD0~1 mode for power saving.
7. If low power domain peripherals clock sources are selected as HIRC/MIRC/LIRC/LXT and HIRC/MIRC/LIRC/LXT are enabled.

Wake-up/Reset-Wake-up sources in each power-down mode :

After chip enters power down, the following wake-up sources can wake up chip to normal mode.

Table 6.2-7 lists the condition about how to enter Power-down mode again for each peripheral. User needs to wait this condition before setting PDEN(CLK_PWRCTL[7]) and execute WFI to enter Power-down mode.

Wake-Up/ Reset-Wake-up Source	Wake-Up/ Reset-Wake-up Condition	Power-Down Mode				Re-Entering Power-Down Mode Condition
		NPD0/ NPD1/ NPD2	NPD3/ NPD4	NPD5/ SPD0/ SPD1/ SPD2	DPD0/ DPD1	
RST pin (Reset-Wake-up)	RST pin Reset 3x us debounce	V	-	-	-	After software writes 1 to clear PINF (SYS_RSTSTS[1])
	RST pin Reset 300ns debounce	-	V	V	V	After software writes 1 (CLK_PMUSTS[31]) to clear RSTWK (CLK_PMUSTS[15]) when NPD3 or NPD4 or NPD5 or SPD0~2 or DPD0~1 mode is entered.
RTC	Wakeup by RTC alarm	V	V	V	V	After software writes 1 (CLK_PMUSTS[31]) to clear RTCWK (CLK_PMUSTS[2]) when DPD0~1 or SPD0~2 mode is entered.
	Wakeup by RTC tick time	V	V	V	V	After software writes 1 (CLK_PMUSTS[31]) to clear RTCWK (CLK_PMUSTS[2]) when DPD0~1 or SPD0~2 mode is entered.
	Wakeup by tamper event	V	V	V	V	After software writes 1 (CLK_PMUSTS[31]) to clear RTCWK (CLK_PMUSTS[2]) when DPD0~1 or SPD0~2 mode is entered.
WKTMR	Wakeup by wake-up timer time-out	V	V	V	V	After software writes 1 (CLK_PMUSTS[31]) to clear TMRWK (CLK_PMUSTS[1]) when power down mode is entered.
BOD (Wake-up/ Reset-Wake-up)	Brown-Out Detector Reset / Interrupt	V	-	-	-	After software writes 1 to clear BODIF (SYS_BODCTL[4]).
	Brown-Out Detector Reset / Wake-up	-	V	V	-	After software writes 1 (CLK_PMUSTS[31]) to clear BODWK (CLK_PMUSTS[13]) when SPD0~2 mode is entered.
LVR (Reset-Wake-up)	LVR Reset	V	-	-	-	After software writes 1 to clear LVRF (SYS_RSTSTS[3])
		-	V	V	-	After software writes 1 (CLK_PMUSTS[31]) to clear LVRWK (CLK_PMUSTS[12]) when SPD0~2 mode is entered.
POR (Reset-Wake-up)	POR Reset	V	V	V	V	After software writes 1 to clear PORF (SYS_RSTSTS[0])
ACMP	Comparator Power-Down Wake-Up Interrupt	V	V	-	-	After software writes 1 to clear WKIF0 (ACMP_STATUS[8]) and WKIF1 (ACMP_STATUS[9]) and WKIF2 (ACMP_STATUS2[8])
	ACMPO status change	-	-	V	-	After software writes 1 (CLK_PMUSTS[31]) to clear ACMPWK (CLK_PMUSTS[18:16]) when SPD0~2 mode is entered.
OPA	Digital output change state	V	V	-	-	After software writes 1 to clear OPDOWKFx (OPA_STATUS [10:8]) and OPDOIFx

						(OPA_STATUS [6:4]).
GPIO	GPIO Interrupt	V	-	-	-	After software write 1 to clear the Px_INTSRC[n] bit.
WKIOx (PA~PD)	rising or falling edge event, 64-pin	V	V	V	-	After software writes 1 (CLK_PMUSTS[31]) to clear GPxWKx (CLK_PMUSTS[27:24], CLK_PMUSTS[11:8) when NPD0~5/SPD0~2 mode is entered.
WKPINx (PC.0/PB.0/PB.2 /PB.12/PF.6/PA.12)	rising or falling edge event, 6-pins	-	-	-	V	After software writes 1 (CLK_PMUSTS[31]) to clear WKPINx (CLK_PMUSTS[6:3] and CLK_PMUSTS[0]) when DPD0~1 mode is entered.
EINT	GPIO Interrupt	V	-	-	-	After software write 1 to clear the Px_INTSRC[n] bit.
WDT (Wake-up/ Reset-Wake-up)	WDT Wake-up with Interrupt/ Reset-Wake-up	V	V	-	-	After software writes 1 to clear WKF (WDT_CTL[5]) (Write Protect).
TIMER	Timer Interrupt	V	-	-	-	After software writes 1 to clear TWKF (TIMERx_INTSTS[1]) and TIF (TIMERx_INTSTS[0]).
	PWM PIF Interrupt	V				After software writes 1 to clear PWMINTWKF (TIMERx_PWMSTATUS[8]) and PIF (TIMERx_PWMINTSTS0[1]).
	PWM CMPUIF Interrupt	V				After software writes 1 to clear PWMINTWKF (TIMERx_PWMSTATUS[8]) and CMPUIF (TIMERx_PWMINTSTS0[2]).
	PWM Interrupt Accumulator	V				After software writes 1 to clear PWMINTWKF (TIMERx_PWMSTATUS[8]) and IFAIF (TIMERx_PWMAINTSTS[0]).
LPTIMER	Timer Interrupt	V	V	-	-	After software writes 1 to clear TWKF (LPTMRx_INTSTS[1]) and TIF (LPTMRx_INTSTS[0]).
	PWM PIF Interrupt	V	V			After software writes 1 to clear PWMINTWKF (LPTMRx_PWMSTATUS[8]) and PIF (LPTMRx_PWMINTSTS0[1]).
	PWM CMPUIF Interrupt	V	V			After software writes 1 to clear PWMINTWKF (LPTMRx_PWMSTATUS[8]) and CMPUIF (LPTMRx_PWMINTSTS0[2]).
	PWM Interrupt Accumulator	V	V			After software writes 1 to clear PWMINTWKF (LPTMRx_PWMSTATUS[8]) and IFAIF (LPTMRx_PWMAINTSTS[0]).
TTMR	Timer Time-out Interrupt	V	V	-	-	After software writes 1 to clear TWKF (TTMRx_INTSTS[1]) and TIF (TTMRx_INTSTS[0]).
UART	nCTS wake-up	V	-	-	-	After software writes 1 to clear CTSWKF (UARTx_WKSTS[0]).
	RX Data wake-up	V	-	-	-	After software writes 1 to clear DATWKF (UARTx_WKSTS[1]).
	Received FIFO Threshold Wake-up	V	-	-	-	After software writes 1 to clear RFRTWKF (UARTx_WKSTS[2]).
	RS-485 AAD Mode Wake-up	V	-	-	-	After software writes 1 to clear RS485WKF (UARTx_WKSTS[3]).
	Received FIFO Threshold Time-out	V	-	-	-	After software writes 1 to clear TOUTWKF (UARTx_WKSTS[4]).

	Wake-up					
LPUART	nCTS wake-up	V	V	-	-	After software writes 1 to clear CTSWKF (LPUARTx_WKSTS[0]).
	RX Data wake-up	V	V	-	-	After software writes 1 to clear DATWKF (LPUARTx_WKSTS[1]).
	Received FIFO Threshold Wake-up	V	V	-	-	After software writes 1 to clear RFRTWKF (LPUARTx_WKSTS[2]).
	RS-485 AAD Mode Wake-up	V	V	-	-	After software writes 1 to clear RS485WKF (LPUARTx_WKSTS[3]).
	Received FIFO Threshold Time-out Wake-up	V	V	-	-	After software writes 1 to clear TOUTWKF (LPUARTx_WKSTS[4]).
	Bus idle time-out	V	V	-	-	After software writes 1 to clear AOTOWKF (LPUARTx_AUTOSTS[0]).
USCI UART	CTS Toggle	V	-	-	-	After software writes 1 to clear WKF (UUART_WKSTS[0]).
	Data Toggle	V	-	-	-	After software writes 1 to clear WKF (UUART_WKSTS[0]).
USCI I^2C	Data toggle	V	-	-	-	After software writes 1 to clear WKF (UI2C_WKSTS[0]).
	Address match	V	-	-	-	After software writes 1 to clear WKAKDONE (UI2C_PROTSTS[16], then writes 1 to clear WKF (UI2C_WKSTS[0]).
USCI SPI	SS Toggle	V	-	-	-	After software writes 1 to clear WKF (USPI_WKSTS[0]).
I^2C	Address match wake-up	V	-	-	-	After software writes 1 to clear WKAKDONE (I2C_WKSTS[1]). Then software writes 1 to clear WKIF(I2C_WKSTS[0]).
LPI^2C	Address match wake-up	V	V	-	-	After software writes 1 to clear WKAKDONE (LPI2C_WKSTS[1]). Then software writes 1 to clear WKIF(LPI2C_WKSTS[0]).
	Master: Transfer data count match TX/RX count	V	V	-	-	After software writes 1 to clear RXWKF (LPI2C_AUTOSTS[1]) or TXWKF (LPI2C_AUTOSTS[0])
	Master: Receive NACK in address phase (address + r/w)	V	V	-	-	After software writes 1 to clear NACKWKF (LPI2C_AUTOSTS[2])
LPSPI	Master: TCNT[7:0] count match wake-up	V	V	-	-	Master: after software write 1 to clear CNTWKF (LPSPI_AUTOSTS[3])
	Slave: SS active wake-up	V	V	-	-	Slave: after software write 1 to clear SSWKF (LPSPI_AUTOSTS[1])
TOUCHKEY	TK Sensing Result Data Match Threshold	V	-	-	-	After software writes 1 to clear SCIF (TK_STA[1]). Then software writes 1 to clear TKIFx (TK_STA[6]~TK_STA[24] and TK_STA1[0]).
UTCPD	Force VBUS off event(from EINT0~5)	V	-	-	-	After software writes 1 to clear FOFFVB (UTCPD_FUTSTS[6]). Then software writes 1 to clear FUTIS (UTCPD_IS[9]
	Connection detected	V	-	-	-	After software writes 1 to clear CCSCHIS (UTCPD_IS[0])
USBOTG	ID pin state be change	V	-	-	-	After software writes 1 to set WKEN(OTG_CTL[5]).

USBH	USE remote wakeup (K-state detected)	V	-	-	-	After USBH engine clock active
	USB connector plug-in/Plug-out (D+/D- state changed)	V	-	-	-	After CCS (HcRhPortStatus1[0]) is 1 for Plug-in while after CCS (HcRhPortStatus1[0]) is 0 for Plug-out.
	Overcurrent MFP pin input state change	V	-	-	-	After POCI (HcRhPortStatus1[3]) is 1.
USBD	Plug-in Wakeup	V	-	-	-	After software writes 1 to clear BUSIF (USBD_INTSTS[0]).
	Resume Wakeup	V	-	-	-	After software writes 1 to clear BUSIF (USBD_INTSTS[0]).
LPPDMA	Transfer Done	V	V	-	-	After software writes 1 to clear WKF (LPPDMAx_INTSTS [3]) and TDIF (LPPDMAx_TDSTS).
	Address don't alignment	V	V	-	-	After software writes 1 to clear WKF (LPPDMAx_INTSTS [3]) and ALIGNIF (LPPDMAx_ALIGN) .
	Bus response error	V	V	-	-	After software writes 1 to clear WKF (LPPDMAx_INTSTS [3]) and ABTIF(LPPDMAx_ABTSTS).
LPADC	ADC Conversion End	V	V	-	-	After software writes 1 to clear ADWKF (LPADC_AUTOSTS[0]).
	Conversion Result Monitor	V	V	-	-	After software writes 1 to clear CMP0WKF (LPADC_AUTOSTS[1]) and CMP1WKF (LPADC_AUTOSTS [2]).
Debug interface	Wakeup by ICE	V	-	-	-	-

Table 6.2-7 Re-Entering Power-down Mode Condition

The NuMicro M2L31 series has two startup methods: Cold Boot and Warm Boot. Cold Boot is the process of re-excuting security boot, and Warm Boot does not re-excute security boot. Table 6.2-8 lists the difference and functional relationships between Cold-boot and Warm-boot.

	Normal-Run/ Power-down mode	**Wake-up/ Resrt-Wake-up Source**	**BootFlag (Re-Security-Boot/ BootFlag value)**	**Vector Map (Function/ Setting)**	**GPIO (I/O-State/ control-by)**	**VTOR (Function/ Setting)**	
Cold Boot	DPDx	POR, nRESET	Yes/ 1	No/ Reset	Tri-state/ NA	No/ Reset	
	SPDx	POR, nRESET, LVR, BOD Reset	Yes/ 1	No/ Reset	Tri-state/ NA	No/ Reset	
	NPDx	POR, nRESET, LVR, BOD Reset, WDT reset	Yes/ 1	No/ Reset	Tri-state/ NA	No/ Reset	
	Normal Run	POR, nRESET, LVR, BOD Reset, WDT reset	Yes/ 1	No/ Reset	Tri-state/ NA	No/ Reset	
	Normal Run	SYS_IPRST0[0], AIRCR[2]	Yes/ 1	No/ Reset	Tri-state/ NA	No/ Reset	
Warm Boot	DPDx	RTC, WKTMR, WKPIN	No/ 1	Yes/ Keep	Keep/ GPIO_HOLD (At CLK_IOPDCTL[8] =1)	No/ Reset	
	SPDx	RTC, WKTMR, WKIO, ACMP, BOD wakeup	No/ 1	Yes/ Keep	Keep/ GPIO_HOLD	Yes/ Reset	
	NPDx	Reference Wake-up source table	No/ 1	Yes/ Keep	Keep/ IP of GPIO	Yes/ Keep	
	Normal Run	SYS_IPRST0[1]	No/ 1	Yes/ Keep	Keep/ IP of GPIO	Yes/ Keep	

Table 6.2-8 Function relationship with Cold-boot and warm-boot

6.2.5 Power Modes and Power Level Transition

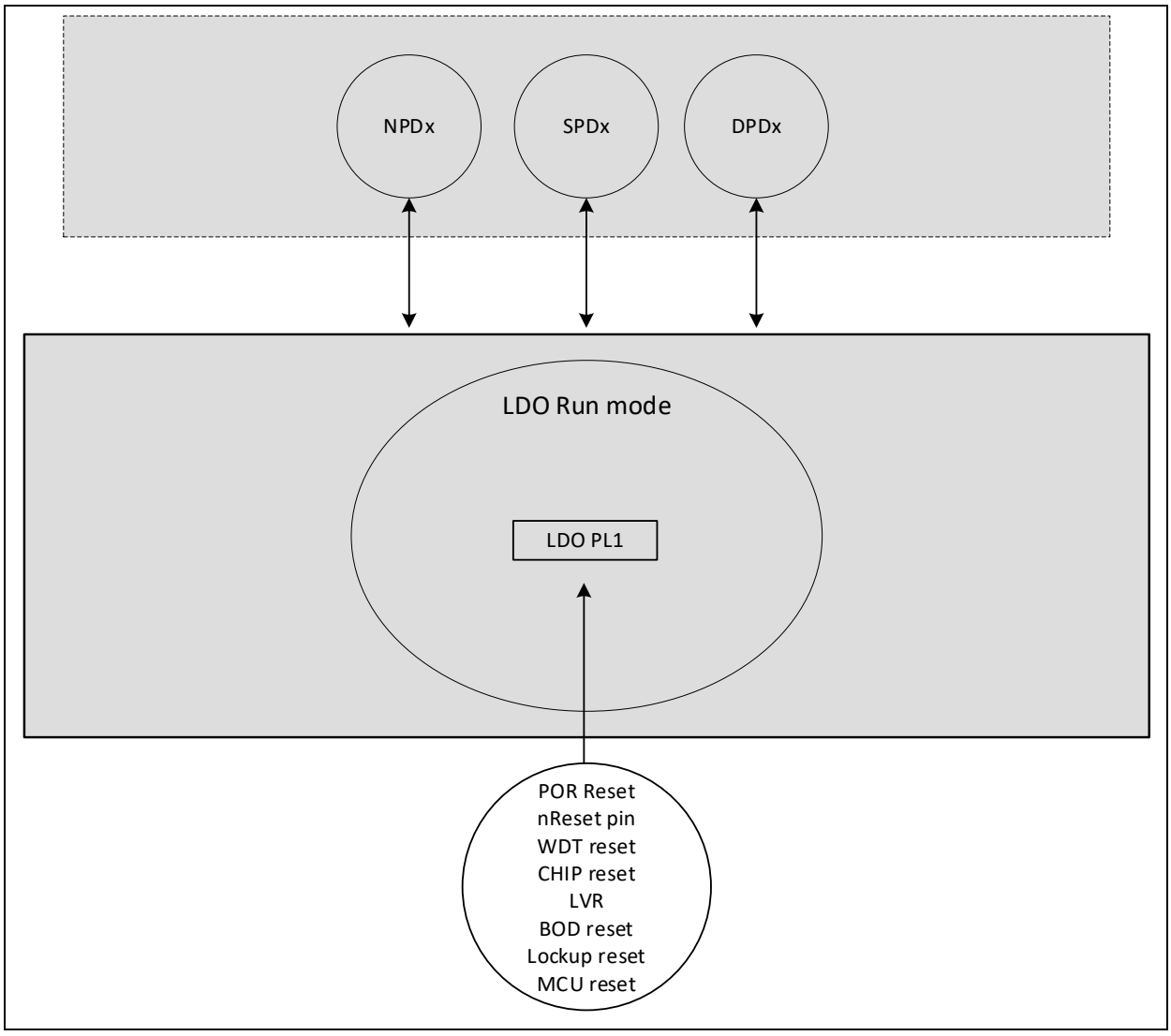


Figure 6.2-8 NuMicro M2L31 Power Mode Transition

6.2.6 System Memory Map

The NuMicro M2L31 series provides 4G-byte addressing space. The memory locations assigned to each on-chip controllers are shown in Table 6.2-9. The detailed register definition, memory space, and programming will be described in the following sections for each on-chip peripheral. The NuMicro M2L31 series only supports little-endian data format.

Address Space	Token	Controllers
RRAM and SRAM Memory Space		
0x0000_0000 – 0x0007_FFFF	RRAM_BA	RRAM Memory Space (512 Kbytes)
0x2000_0000 – 0x2000_9FFF	SRAM0_BA	SRAM Memory Space (40 Kbytes)
0x2000_A000 – 0x2002_9FFF	SRAM1_BA	SRAM Memory Space (128 Kbytes)
0x2800_0000 – 0x2800_1FFF	LPSRAM_BA	Low power SRAM Memory Space (8 Kbytes)
0x6000_0000 – 0x6FFF_FFFF	EXTMEM_BA	External Memory Space (256 Mbytes)
Peripheral Controllers Space (0x4000_0000 – 0x400F_FFFF)		
0x4000_0000 – 0x4000_01FF 0x4000_0500 – 0x4000_0FFF	SYS_BA	System Control Registers
0x4000_0200 – 0x4000_02FF	CLK_BA	Clock Control Registers
0x4000_0300 – 0x4000_03FF	NMI_BA	NMI Control Registers
0x4000_4000 – 0x4000_4FFF	GPIO_BA	GPIO Control Registers
0x4000_8000 – 0x4000_8FFF	PDMA0_BA	Peripheral DMA0 Control Registers
0x4000_9000 – 0x4000_9FFF	USBH_BA	USB Host Control Registers
0x4000_C000 – 0x4000_CFFF	RMC_BA	RRAM Memory Control Registers
0x4001_0000 – 0x4001_0FFF	EBI_BA	External Bus Interface Control Registers
0x4002_0000 – 0x4002_1FFF	CANFD0_BA	CANFD0 Control Registers
0x4002_4000 – 0x4002_5FFF	CANFD1_BA	CANFD1 Control Registers
0x4003_1000 – 0x4003_1FFF	CRC_BA	CRC Generator Registers
0x4003_2000 – 0x4003_4FFF	CRYP_BA	Cryptographic Accelerator Registers
0x4003_5000 – 0x4003_5FFF	KS_BA	Key Store Registers
0x4003_8000 – 0x4003_8FFF	LPSCC_BA	Low Power System and Clock Control Registers
0x4003_9000 – 0x4003_9FFF	LPPDMA0_BA	Low Power Peripheral DMA0 Control Registers
0x4003_A000 – 0x4003_AFFF	LPGPIO_BA	Low Power GPIO Control Registers
APB Controllers Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4004_0000 – 0x4004_0FFF	WDT_BA	Watchdog Timer Control Registers
0x4004_1000 – 0x4004_1FFF	RTC_BA	Real Time Clock (RTC) Control Register
0x4004_3000 – 0x4004_3FFF	EADC0_BA	Enhanced Analog-Digital-Converter 0 (EADC0) Control Registers
0x4004_5000 – 0x4004_5FFF	ACMP01_BA	Analog Comparator 0/ 1 Control Registers
0x4004_6000 – 0x4004_6FFF	OPA_BA	OP Amplifier Control Registers
0x4004_7000 – 0x4004_7FFF	DAC_BA	DAC Control Registers

0x4004_D000 – 0x4004_DFFF	OTG_BA	OTG Control Registers
0x4005_0000 – 0x4005_0FFF	TMR01_BA	Timer0/Timer1 Control Registers
0x4005_1000 – 0x4005_1FFF	TMR23_BA	Timer2/Timer3 Control Registers
0x4005_8000 – 0x4005_8FFF	EPWM0_BA	EPWM0 Control Registers
0x4005_9000 – 0x4005_9FFF	EPWM1_BA	EPWM1 Control Registers
0x4005_C000 – 0x4005_CFFF	PWM0_BA	PWM0 Control Registers
0x4005_D000 – 0x4005_DFFF	PWM1_BA	PWM1 Control Registers
0x4006_0000 – 0x4006_0FFF	QSPI0_BA	QSPI0 Control Registers
0x4006_1000 – 0x4006_1FFF	SPI0_BA	SPI0 Control Registers
0x4006_2000 – 0x4006_2FFF	SPI1_BA	SPI1 Control Registers
0x4006_3000 – 0x4006_3FFF	SPI2_BA	SPI2 Control Registers
0x4006_4000 – 0x4006_4FFF	SPI3_BA	SPI3 Control Registers
0x4007_0000 – 0x4007_0FFF	UART0_BA	UART0 Control Registers
0x4007_1000 – 0x4007_1FFF	UART1_BA	UART1 Control Registers
0x4007_2000 – 0x4007_2FFF	UART2_BA	UART2 Control Registers
0x4007_3000 – 0x4007_3FFF	UART3_BA	UART3 Control Registers
0x4007_4000 – 0x4007_4FFF	UART4_BA	UART4 Control Registers
0x4007_5000 – 0x4007_5FFF	UART5_BA	UART5 Control Registers
0x4007_6000 – 0x4007_6FFF	UART6_BA	UART6 Control Registers
0x4007_7000 – 0x4007_7FFF	UART7_BA	UART7 Control Registers
0x4008_0000 – 0x4008_0FFF	I2C0_BA	I2C0 Control Registers
0x4008_1000 – 0x4008_1FFF	I2C1_BA	I2C1 Control Registers
0x4008_2000 – 0x4008_2FFF	I2C2_BA	I2C2 Control Registers
0x4008_3000 – 0x4008_3FFF	I2C3_BA	I2C3 Control Registers
0x4009_6000 – 0x4009_6FFF	WWDT_BA	WWDT Control Registers
0x400B_0000 – 0x400B_0FFF	EQEI0_BA	EQEI0 Control Registers
0x400B_1000 – 0x400B_1FFF	EQEI1_BA	EQEI1 Control Registers
0x400B_4000 – 0x400B_4FFF	ECAP0_BA	ECAP0 Control Registers
0x400B_5000 – 0x400B_5FFF	ECAP1_BA	ECAP1 Control Registers
0x400B_9000 – 0x400B_9FFF	TRNG_BA	TRNG Control Registers
0x400C_0000 – 0x400C_0FFF	USBD_BA	USB Device Control Register
0x400C_4000 – 0x400C_4FFF	TK_BA	Touchkey Control Register
0x400C_6000 – 0x400C_6FFF	UTCPD_BA	UTCPD Control Register
0x400C_9000 – 0x400C_9FFF	ACMP2_BA	Analog Comparator 2 Control Registers
0x400D_0000 – 0x400D_0FFF	USCI0_BA	USCI0 Control Registers

0x400D_1000 – 0x400D_1FFF	USCI1_BA	USCI1 Control Registers
0x400E_0000 – 0x400E_0FFF	LPUART0_BA	LPUART0 Control Registers
0x400E_1000 – 0x400E_1FFF	LPSPI0_BA	LPSPI0 Control Registers
0x400E_2000 – 0x400E_2FFF	LPI2C0_BA	LPI2C0 Control Registers
0x400E_3000 – 0x400E_3FFF	LPADC0_BA	LPADC0 Control Registers
0x400E_4000 – 0x400E_4FFF	LPTMR01_BA	LPTMR01 Control Registers
0x400E_5000 – 0x400E_5FFF	TTMR01_BA	TTMR01 Control Registers
System Controllers Space (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SCS_BA	System Timer Control Registers
0xE000_E100 – 0xE000_ECFF	SCS_BA	External Interrupt Controller Control Registers
0xE000_ED00 – 0xE000_ED8F	SCS_BA	System Control Registers

Table 6.2-9 Address Space Assignments for On-Chip Controllers

6.2.7 SRAM Memory Orginization

The M2L31 supports embedded SRAM up to 168 Kbytes size and the SRAM organization is separated to two banks: SRAM bank0 and SRAM bank1. The first bank has 40 Kbytes address space, the second bank has 128 Kbyte address space. These two banks address space can be accessed simultaneously. The SRAM bank0 supports parity error check to make sure chip operating more stable.

- Supports up to 168 Kbytes SRAM
- Supports byte / half word / word write
- Supports parity error check function for SRAM bank0
- Supports oversize response error

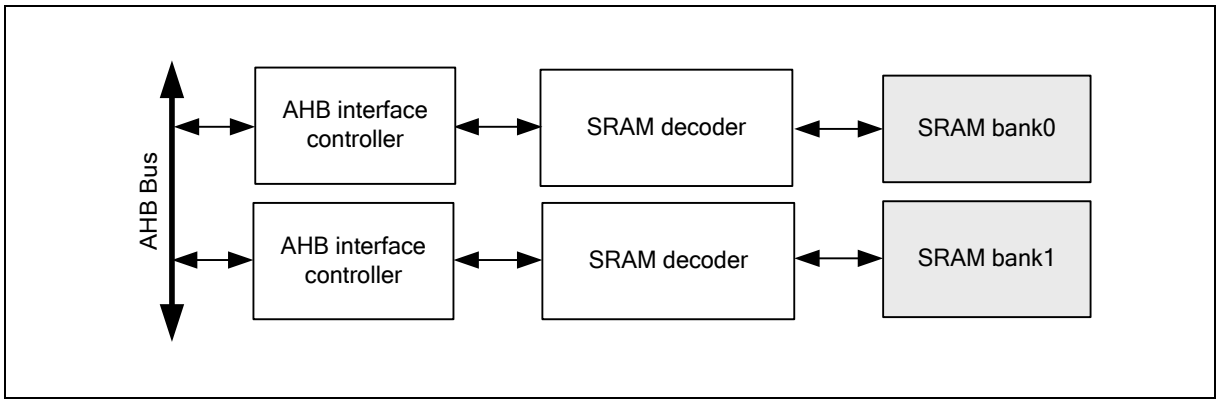


Figure 6.2-9 SRAM Block Diagram

Figure 6.2-9 shows the SRAM organization of M2L31. There are two SRAM banks in M2L31. The bank0 is addressed to 40 Kbytes and the bank1 is addressed to 128 Kbytes. The bank0 address space is from 0x2000_0000 to 0x2000_9FFF. The bank1 address space is from 0x2000_A000 to 0x2002_9FFF. The address between 0x2002_A000 to 0x3FFF_FFFF is illegal memory space and chip will enter hardfault if CPU accesses these illegal memory addresses.

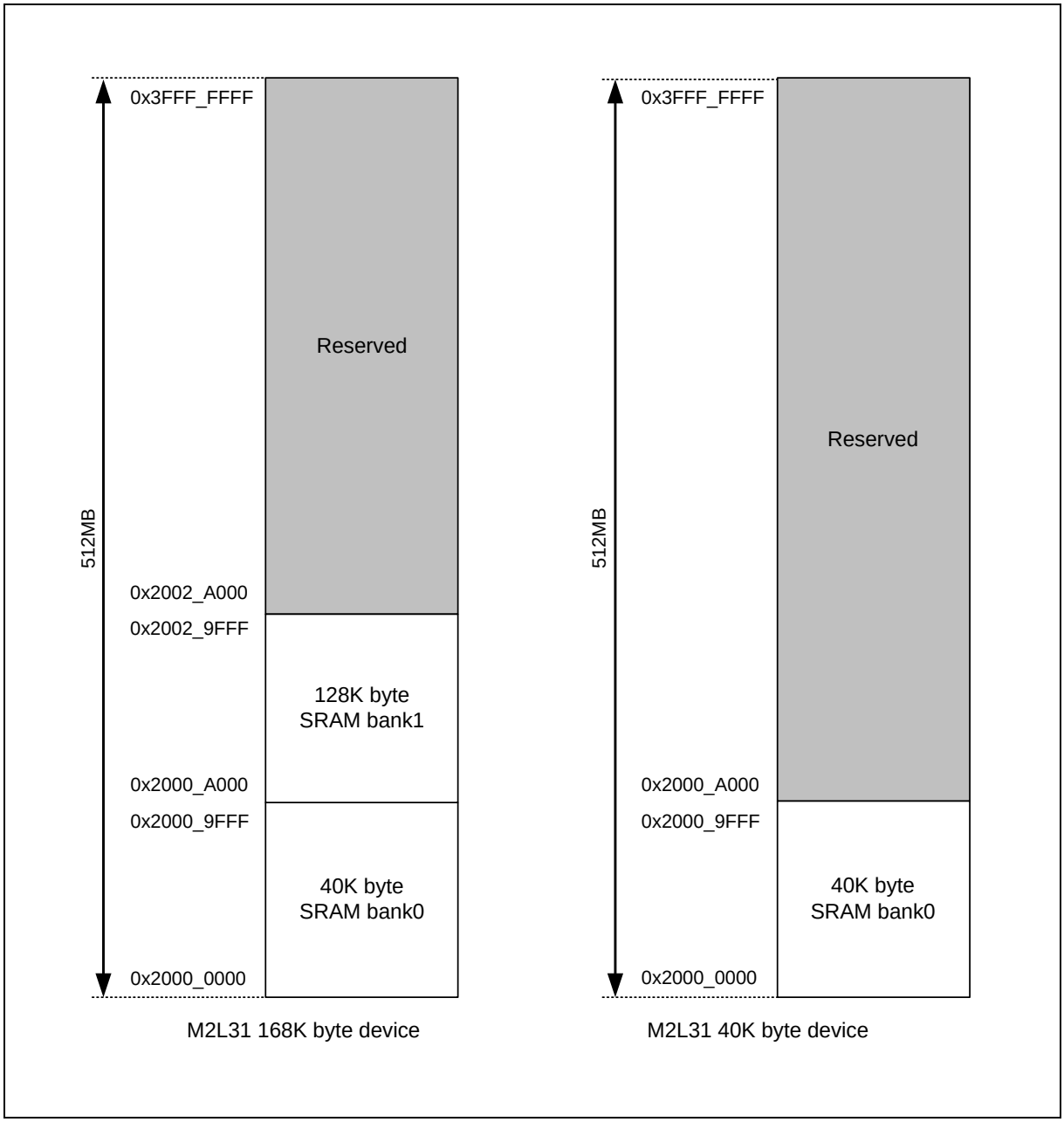


Figure 6.2-10 SRAM Memory Organization

6.2.7.1 SRAM Parity Error Check

SRAM bank0 has byte parity error check function. Users must initialize SRAM before CPU accessing SRAM bank0. The parity error check is excuting while SRAM being read. As parity error occured, the PERRIF (SYS_SRAM_STATUS[0]) will be asserted to 1 and the SYS_SRAM_ERRADDR register will recode the address with parity error. Chip will enter interrupt when SRAM parity error occurred if PERRIEN (SYS_SRAM_INTCTL[0]) is set to 1. When SRAM parity error occured, chip will stop detecting SRAM parity error until user writes 1 to clear the PERRIF(SYS_SRAM_STATUS[0]) bit.

6.2.7.2 SRAM Power Control

The SRAM bank0 and bank1 have marco retention and shut down function. Each SRAM marco can be configured to retention or shut down mode independently by SRAMxPMn(SYS_SRAMPC0 and SYS_SRAMPC1, x=0-1 n=0-7). Figure 6.2-11 shows the SRAM marco number in bank0 and bank1.

When chip wake-up from Power-down mode, the SRAM marcos wake up in the order of marco number, from SRAM marco0 to SRAM marco6.

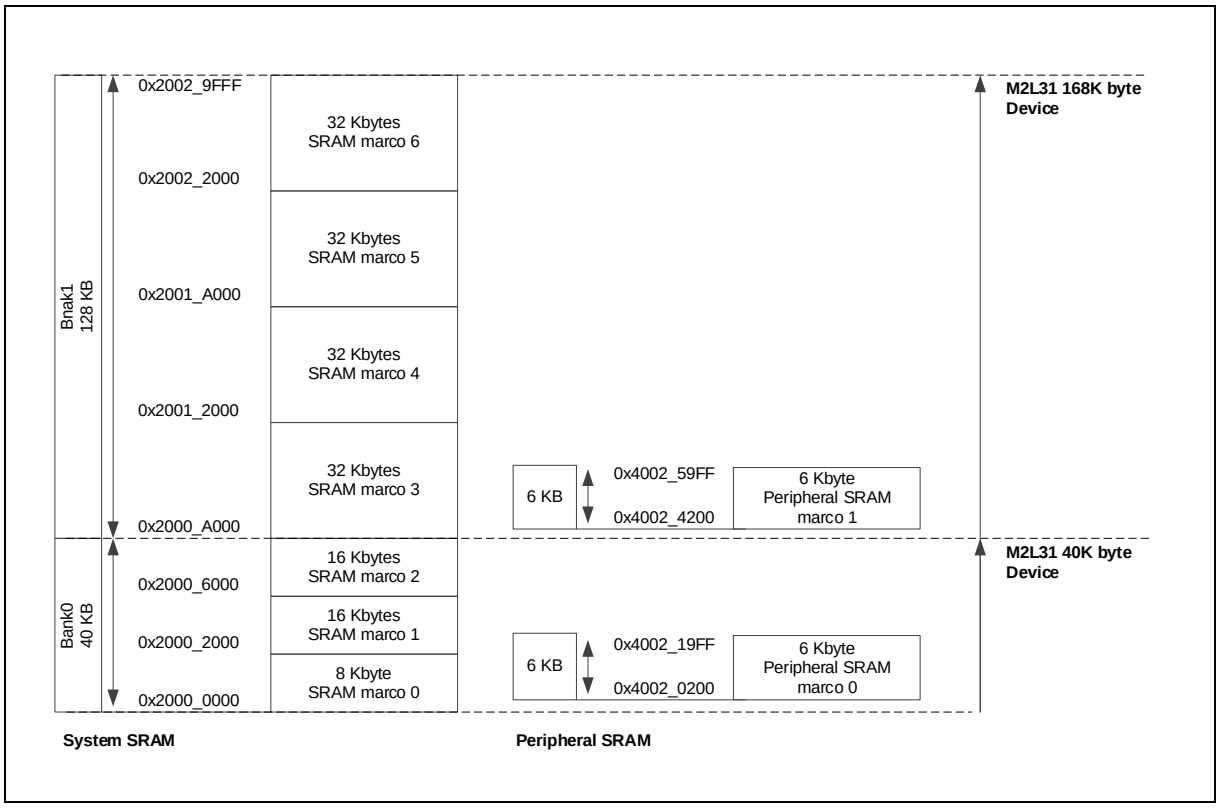


Figure 6.2-11 SRAM Marco Organization

For system SRAM (bank0 and bank1), if the SRAM power mode is set to normal mode, it automatically changes to retention mode when system enters NPD0~5/SPD0~2 Power-down mode, and then changes back to normal mode after wake-up. System SRAM power mode do not change when system enters NPD0 Power-down mode. When system enters DPD0~1 Power-down mode, system SRAM is always operating in shut down mode and reset to normal mode after wake-up.

For other peripheral SRAM, when system enters NPD0~5/SPD0~2 Power-down mode, if peripheral SRAM power mode is set to normal mode, the peripheral SRAM power mode automatically changes to retention mode, and then changes back to normal mode after wake-up, too.

But if entering SPD0~2 Power-down mode, peripheral SRAM resets to default power mode after wake-up.

When system enters DPD0~1 Power-down mode, peripheral SRAM is always operating in shut down mode and reset to default power mode after wake-up. Peripheral SRAM power mode does not change when system enters NPD0 Power-down mode.

SRAM	Power-Down Mode	SRAM Power Mode Before and After Wake-Up
System SRAM bank0/1	NPD0~2	1.If SRAM power mode is set to normal mode, it changes to retention mode after entering Power-down Mode, and changes back to normal mode after wake-up. 2.If SRAM power mode is set to retention or shut down mode, it keeps power mode setting.
	NPD3~5	1.If SRAM power mode is set to normal mode and is selected within retention size, it changes to retention mode after entering Power-down Mode, and changes back to normal mode after wake-up.

		2.If SRAM power mode is set to normal mode and is not selected within retention size, it changes to shut down mode after entering Power-down Mode, and changes back to normal mode after wake-up. 3.If SRAM power mode is set to retention or shut down mode, it keeps power mode setting.
	SPD0~2	1.If SRAM power mode is set to normal mode and is selected within retention size, it changes to retention mode after entering Power-down Mode, and changes back to normal mode after wake-up. 2.If SRAM power mode is set to normal mode and is not selected within retention size, it changes to shut down mode after entering Power-down Mode, and changes back to normal mode after wake-up. 3.If SRAM power mode is set to retention or shut down mode, it keeps power mode setting after entering Power-down Mode, and changes back to normal mode after wake-up.
	DPD0~1	No matter what SRAM power mode is set, SRAM will change to shut down mode after entering Power-down Mode, and changes back to normal mode after wake-up.
Peripheral SRAM bank0/1	NPD0~2	SRAM keep normal mode before/after entring Power-down Mode and wake-up.
	NPD3~5, SPD0~2, DPD0~1	SRAM will enter shut down mode after entering Power-down Mode, and changes back to normal mode after wake-up.

Table 6.2-10 SRAM Power Mode Behavior

6.2.8 Low Power SRAM Memory Orginization

The M2L31 supports embedded low power SRAM up to 8 Kbytes size. The low power SRAM located in low power domain and operating under HCLK1 clock speed. The low power SRAM can co-work with LPPDMA and the IP which in low power domain to do data transfer. The CPU and PDMA also can access low power SRAM, and it takes four HCLK0 clock and four HCLK1 clock for write data. For read data, it takse three HCLK0 clock and four HCLK1 clock.

- Supports up to 8 Kbytes SRAM
- Supports byte / half word / word write
- Supports oversize response error

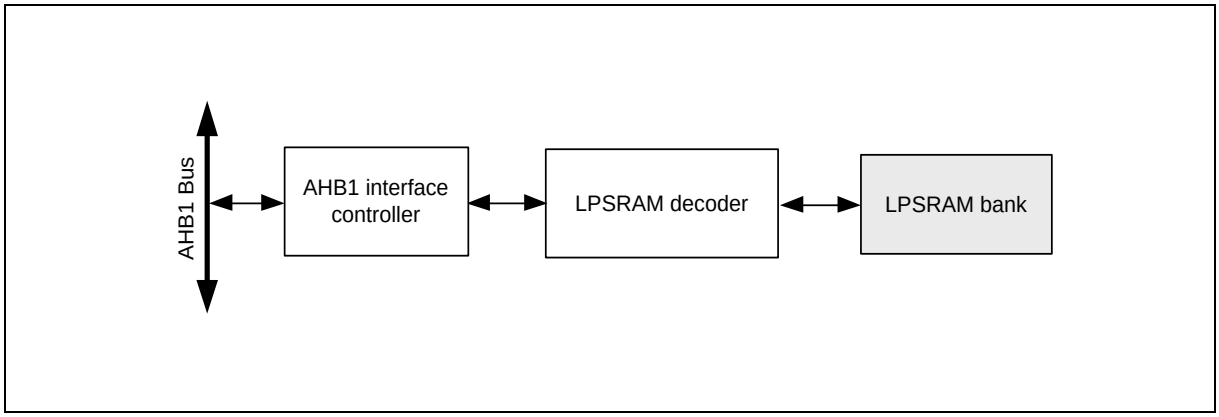


Figure 6.2-12 LPSRAM Block Diagram

6.2.9 Bus Matrix

The M2L31 supports Bus Matrix to manage the access arbitration between masters. The access arbitration can be selected by INTACTEN (SYS_AHBMCTL[0]) to use round-robin algorithm or set Cortex-M23 CPU as the highest bus priority.

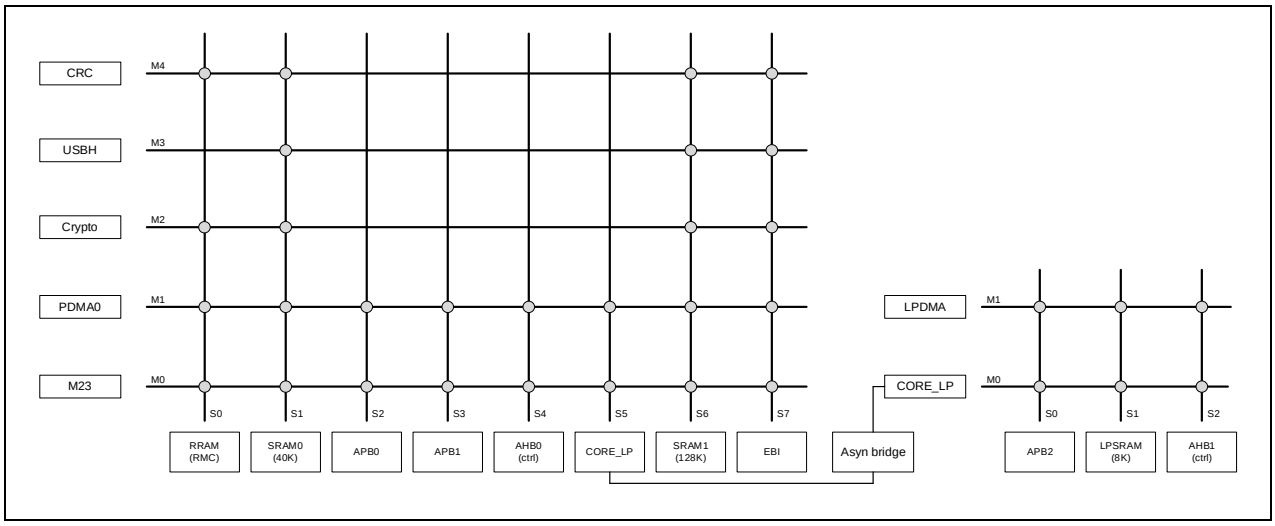


Figure 6.2-13 NuMicro M2L31 Bus Matrix Diagram

Note: The RRAM, SRAM0/1, LPSRAM, CRC and EBI transfers must be aligned to the address boundry which equal to the transfer size (Word/HalfWord/Byte). Other AHB and APB slaves must be word-aligned to the address boundry with transfer size in Word.

6.2.10 HIRC Auto Trim

This chip supports auto-trim function: the HIRC trim (12 MHz RC oscillator), HIRC trim (48 MHz RC oscillator) and MIRC trim (1 MHz RC oscillator) according to the accurate LXT (32.768 kHz crystal oscillator) or internal USB synchronous mode, automatically gets accurate HIRC output frequency, 0.25 % /0.5% /0.75% /1% selected by ACCURSEL (SYS_HIRCTCTL[3:2]/ SYS_IRCTCTL[3:2]/ SYS_MIRCTCTL[3:2]) deviation within all temperature ranges.

For some application needs an accurate 12 MHz clock. In such case, if neither using PLL as the system clock source nor soldering 32.768 kHz crystal in system, user has to set REFCKSEL (SYS_IRCTCTL[10] reference clock selection) to “1”, set FREQSEL (SYS_IRCTCTL[1:0] trim frequency selection) to “01”, and the auto-trim function will be enabled. Interrupt status bit FREQLOCK (SYS_IRCTISTS[8] HIRC frequency lock status) “1” indicates the HIRC output frequency is accurate within the deviation ACCURSEL(SYS_IRCTCTL[3:2]) set.

For some application needs an accurate 48 MHz clock. In such case, if neither using PLL as the system clock source nor soldering 32.768 kHz crystal in system, user has to set REFCKSEL (SYS_HIRCTCTL[10] reference clock selection) to “1”, set FREQSEL (SYS_HIRCTCTL[1:0] trim frequency selection) to “10”, and the auto-trim function will be enabled. Interrupt status bit FREQLOCK (SYS_HIRCTSTS[8] HIRC frequency lock status) “1” indicates the HIRC output frequency is accurate within the deviation ACCURSEL(SYS_HIRCTCTL[3:2]) set.

For some application needs an accurate 1 MHz clock. In such case, if neither using PLL as the system clock source nor soldering 32.768 kHz crystal in system, user has to set REFCKSEL (SYS_MIRCTCTL[10] reference clock selection) to “1”, set FREQSEL (SYS_MIRCTCTL[1:0] trim frequency selection) to “10”, and the auto-trim function will be enabled. Interrupt status bit FREQLOCK (SYS_MIRCTSTS[8] HIRC frequency lock status) “1” indicates the HIRC output frequency is accurate within the deviation ACCURSEL(SYS_MIRCTCTL[3:2]) set.

6.2.11 Register Lock Control

Some of the system control registers need to be protected to avoid inadvertent write and disturb the chip operation. These system control registers are protected after the power-on reset till user to disable register protection. For user to program these protected registers, a register protection disable sequence needs to be followed by a special programming. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register SYS_REGLCTL address at 0x4000_0100 continuously. Any different data value, different sequence or any other write to other address during these three data writing will abort the whole sequence.

Protected Register	Location
SYS_IPRST0	Address 0x4000_0008
SYS_BODCTL	address 0x4000_0018
SYS_VREFCTL	address 0x4000_0028
SYS_USBPHY	address 0x4000_002C
SYS_SRAM_BISTCTL	address 0x4000_00D0
SYS_PORDISAN	address 0x4000_01EC
SYS_PLCTL	address 0x4000_01F8
CLK_PWRCTL	address 0x4000_0200
CLK_APBCLK0	address 0x4000_0208
CLK_CLKSEL0	address 0x4000_0210
CLK_CLKSEL1	address 0x4000_0214
CLK_PLLCTL	address 0x4000_0240
CLK_PMUCTL	address 0x4000_0290
NMIEN	address 0x4000_0300
AHBMCTL	address 0x4000_0400
RMC_FTCTL	address 0x4000_5018
RMC_ICPCMD	address 0x4000_501C
RMC_ISPCTL	address 0x4000_C000
RMC_ISPTRG	address 0x4000_C010
RMC_ISPSTS	address 0x4000_C040
RMC_CYCCTL	address 0x4000_C04C
RMC_KPKEYTRG	address 0x4000_C05C
RMC_KPKEYSTS	address 0x4000_C060
WDT_CTL	address 0x4004_0000
WDT_ALTCTL	address 0x4004_0004
TIMER0_CTL	address 0x4005_0000
TIMER1_CTL	address 0x4005_0100
TIMER2_CTL	address 0x4005_1000
TIMER3_CTL	address 0x4005_1100

TIMER0_PWMCTL	address 0x4005_0040
TIMER1_PWMCTL	address 0x4005_0140
TIMER2_PWMCTL	address 0x4005_1040
TIMER3_PWMCTL	address 0x4005_1140
TIMER0_PWMDTCTL	address 0x4005_0058
TIMER1_PWMDTCTL	address 0x4005_0158
TIMER2_PWMDTCTL	address 0x4005_1058
TIMER3_PWMDTCTL	address 0x4005_1158
TIMER0_PWMBRKCTL	address 0x4005_0070
TIMER1_PWMBRKCTL	address 0x4005_0170
TIMER2_PWMBRKCTL	address 0x4005_1070
TIMER3_PWMBRKCTL	address 0x4005_1170
TIMER0_PWMSWBRK	address 0x4005_007C
TIMER1_PWMSWBRK	address 0x4005_017C
TIMER2_PWMSWBRK	address 0x4005_107C
TIMER3_PWMSWBRK	address 0x4005_117C
TIMER0_PWMINTEN1	address 0x4005_0084
TIMER1_PWMINTEN1	address 0x4005_0184
TIMER2_PWMINTEN1	address 0x4005_1084
TIMER3_PWMINTEN1	address 0x4005_1184
TIMER0_PWMINTSTS1	address 0x4005_008C
TIMER1_PWMINTSTS1	address 0x4005_018C
TIMER2_PWMINTSTS1	address 0x4005_108C
TIMER3_PWMINTSTS1	address 0x4005_118C
EPWM_CTL0	address 0x4005_8000/0x4005_9000
EPWM_CTL1	address 0x4005_8000/0x4005_9000
EPWM_DTCTL0_1	address 0x4005_8070/0x4005_9070
EPWM_DTCTL2_3	address 0x4005_8074/0x4005_9074
EPWM_DTCTL4_5	address 0x4005_8078/0x4005_9078
EPWM_BRKCTL0_1	address 0x4005_80C8/0x4005_90C8
EPWM_BRKCTL2_3	address 0x4005_80CC/0x4005_90CC
EPWM_BRKCTL4_5	address 0x4005_80D0/0x4005_90D0
EPWM_SWBRK	address 0x4005_80DC/0x4005_90DC
EPWM_INTEN1	address 0x4005_80E4/0x4005_90E4
EPWM_INTSTS1	address 0x4005_80EC/0x4005_90EC

BPWM_CTL0	address 0x4005_A000/0x4005_B000
SYST_VAL	address 0xE000_E018

Table 6.2-11 Protected Registers

6.2.12 Deep Power Down Wakeup Pin

When chip enter deep Power-down mode (DPD0/DPD1), the chip can be wake-up by six wake-up pins. These pins are WKPIN0 (PC.0), WKPIN1 (PB.0), WKPIN2 (PB.2), WKPIN3 (PB.12), WKPIN4 (PF.6) and WKPIN5 (PA.12). User can set WKPINEN0 (CLK_PMUWKCTL[17:16]) to select WKPIN0 trigger edge or disable WKPIN0 wake-up function. In the same way, user can set WKPINEN1 (CLK_PMUWKCTL[19:18]) to select WKPIN1 trigger edge or disable WKPIN1 wake-up function, set WKPINEN2 (CLK_PMUWKCTL[21:20]) to select WKPIN2 trigger edge or disable WKPIN2 wake-up function, set WKPINEN3 (CLK_PMUWKCTL[23:22]) to select WKPIN3 trigger edge or disable WKPIN3 wake-up function, set WKPINEN4 (CLK_PMUWKCTL[25:24]) to select WKPIN4 trigger edge or disable WKPIN4 wake-up function and set WKPINEN5 (CLK_PMUWKCTL[27:26]) to select WKPIN5 trigger edge or disable WKPIN5 wake-up function.

6.2.13 Wake-up Timer (WKTMR) and Wake-up IO (WKIO) Function

The WKTMR function can be enabled by WKTMREN (CLK_PMUWKCTL[0]), WKTMRMOD (CLK_PMUWKCTL[1]) and WKTMRIS (CLK_PMUWKCTL[11:8]). WKTMR can wake-up chip from NPD0~5/SPD0~2/DPD0~1 modes.

When WKTMR wake-up event occurred in NPD0~5 modes, TMRWK (CLK_PMUSTS[1]) will be set, and if WKTMRIE (CLK_PMUINTC[0]) is set to 1, WKTMRIF (CLK_PMUINTS[0]) will be set and ETI interrupt will be triggered. When WKTMR wake-up event occurred in SPD0~2/DPD0~1 modes, chip will be reset and re-run. Users can check wakeup flag from TMRWK (CLK_PMUSTS[1]).

The WKIO function can be enabled by CLK_PAPWCTL, CLK_PBPWCTL, CLK_PCPWCTL, CLK_PDPWCTL and CLK_PWDBCTL. WKIO can wakeup chip from NPD0~5/SPD0~2 modes.

When WKIO event occurred in NPD0~5 modes, GPXWKx(CLK_PMUSTS[27:24], CLK_PMUSTS[11:8]) will be set, and if any of WKIO interrupt enable (CLK_PMUINTC[15:8]) is set to 1, WKIO interrupt flag (CLK_PMUINTS[15:8]) will be set and ETI interrupt will be triggered. When WKIO wake-up event occurred in SPD0~2 modes, chip will be reset and re-run. Users can check wakeup flag from GPXWKx (CLK_PMUSTS[27:24], CLK_PMUSTS[11:8]).

6.2.14 Low Power Domain System Clock

The low power domain system clock include HCLK1 and PCLK2. HCLK1 has five clock sources, which can be selected by HCLK1SEL (CLK_CLKSEL0[7:6]). PCLK2 is generated from HCLK1, can be HCLK1 divided by 1/2/4/8/16. The block diagram is shown in Figure 6.2-14.

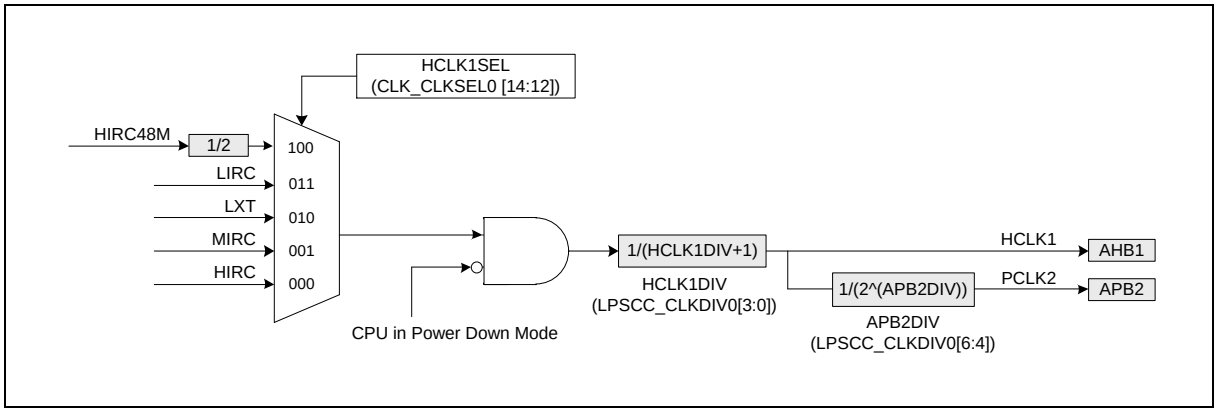


Figure 6.2-14 Low Power Domain System Clock Block Diagrams

6.2.15 Low Power Domain Control Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
LPSCC Base Address: **LPSCC_BA = 0x4003_8000**				
LPSCC_IPRST0	LPSCC_BA+0x04	R/W	Peripheral Reset Control Register 0	0x0000_0000
LPSCC_CLKEN0	LPSCC_BA+0x40	R/W	Peripheral Clock Enable Control Register 0	0x0000_0000
LPSCC_CLKKEEP0	LPSCC_BA+0x44	R/W	Peripheral Clock Keep Control Register 0	0x0000_0000
LPSCC_CLKSEL0	LPSCC_BA+0x50	R/W	Peripheral Clock Source Select Control Register 0	0x0000_0000
LPSCC_CLKDIV0	LPSCC_BA+0x60	R/W	Peripheral Clock Divider Number Register 0	0x0000_0000
LPSCC_CLKMCTL	LPSCC_BA+0x80	R/W	Clock Monitor Control Register	0x0000_0000
LPSCC_SRAMCTL	LPSCC_BA+0xA0	R/W	Low Power SRAM Control Register	0x0000_0010
LPSCC_IOHCTL	LPSCC_BA+0x100	R/W	I/O Hold Control Register	0x0000_0000

6.2.16 Low Power Domain Control Register Description

Peripheral Reset Control Register 0 (LPSCC_IPRST0)

Register	Offset	R/W	Description	Reset Value
LPSCC_IPRST0	LPSCC_BA+0x04	R/W	Peripheral Reset Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**OPARST**	**Reserved**		**LPADC0RST**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TTMR1RST	**TTMR0RST**	**LPTMR1RST**	**LPTMR0RST**	**LPUART0RST**	**LPI2C0RST**	**LPSPI0RST**	**WDTRST**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**LPSRAMRST**	**LPGPIORST**	**LPPDMA0RST**

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**OPARST**	**OP Amplifier Controller Reset** 0 = OPA controller normal operation. 1 = OPA controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[26:25]	**Reserved**	Reserved.
[24]	**LPADC0RST**	**LPADC0 Controller Reset** 0 = LPADC0 controller normal operation. 1 = LPADC0 controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[23]	**TTMR1RST**	**TTMR1 Controller Reset** 0 = TTMR1 controller normal operation. 1 = TTMR1 controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[22]	**TTMR0RST**	**TTMR0 Controller Reset** 0 = TTMR0 controller normal operation. 1 = TTMR0 controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[21]	**LPTMR1RST**	**LPTMR1 Controller Reset** 0 = LPTMR1 controller normal operation. 1 = LPTMR1 controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[20]	**LPTMR0RST**	**LPTMR0 Controller Reset** 0 = LPTMR0 controller normal operation.

		1 = LPTMR0 controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[19]	**LPUART0RST**	**LPUART0 Controller Reset** 0 = LPUART0 controller normal operation. 1 = LPUART0 controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[18]	**LPI2C0RST**	**LPI2C0 Controller Reset** 0 = LPI2C0 controller normal operation. 1 = LPI2C0 controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[17]	**LPSPI0RST**	**LPSPI0 Controller Reset** 0 = LPSPI0 controller normal operation. 1 = LPSPI0 controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[16]	**WDTRST**	**WDT Controller Reset** 0 = WDT controller normal operation. 1 = WDT controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[15:3]	**Reserved**	Reserved.
[2]	**LPSRAMRST**	**LPSRAM Controller Reset** 0 = LPSRAM controller normal operation. 1 = LPSRAM controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1]	**LPGPIORST**	**LPGPIO Controller Reset** 0 = LPGPIO controller normal operation. 1 = LPGPIO controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**LPPDMA0RST**	**LPPDMA0 Controller Reset** 0 = LPPDMA0 controller normal operation. 1 = LPPDMA0 controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Peripheral Clock Enable Control Register 0 (LPSCC_CLKEN0)

The bits in this register are used to enable/disable clock for low power domain peripherial clock.

Register	Offset	R/W	Description	Reset Value
LPSCC_CLKEN0	LPSCC_BA+0x40	R/W	Peripheral Clock Enable Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				OPACKEN	Reserved		LPADC0CKEN
23	22	21	20	19	18	17	16
TTMR1CKEN	TTMR0CKEN	LPTMR1CKEN	LPTMR0CKEN	LPUART0CKEN	LPI2C0CKEN	LPSPI0CKEN	WDTCKEN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					LPSRAMCKEN	LPGPIOCKEN	LPPDMA0CKEN

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**OPACKEN**	**OP Amplifier Clock Enable Bit** 0 = OPA clock Disabled. 1 = OPA clock Enabled.
[26:25]	**Reserved**	Reserved.
[24]	**LPADC0CKEN**	**LPADC0 Clock Enable Bit** 0 = LPADC0 clock Disabled. 1 = LPADC0 clock Enabled.
[23]	**TTMR1CKEN**	**TTMR1 Clock Enable Bit** 0 = TTMR1 clock Disabled. 1 = TTMR1 clock Enabled.
[22]	**TTMR0CKEN**	**TTMR0 Clock Enable Bit** 0 = TTMR0 clock Disabled. 1 = TTMR0 clock Enabled.
[21]	**LPTMR1CKEN**	**LPTMR1 Clock Enable Bit** 0 = LPTMR1 clock Disabled. 1 = LPTMR1 clock Enabled.
[20]	**LPTMR0CKEN**	**LPTMR0 Clock Enable Bit** 0 = LPTMR0 clock Disabled. 1 = LPTMR0 clock Enabled.
[19]	**LPUART0CKEN**	**LPUART0 Clock Enable Bit** 0 = LPUART0 clock Disabled. 1 = LPUART0 clock Enabled.
[18]	**LPI2C0CKEN**	**LPI2C0 Clock Enable Bit**

		0 = LPI2C0 clock Disabled. 1 = LPI2C0 clock Enabled.
[17]	**LPSPI0CKEN**	**LPSPI0 Clock Enable Bit** 0 = LPSPI0 clock Disabled. 1 = LPSPI0 clock Enabled.
[16]	**WDTCKEN**	**WDT Clock Enable Bit** 0 = WDT clock Disabled. 1 = WDT clock Enabled.
[15:3]	**Reserved**	Reserved.
[2]	**LPSRAMCKEN**	**LPSRAM Clock Enable Bit** 0 = LPSRAM clock Disabled. 1 = LPSRAM clock Enabled.
[1]	**LPGPIOCKEN**	**LPGPIO Clock Enable Bit** 0 = LPGPIO clock Disabled. 1 = LPGPIO clock Enabled.
[0]	**LPPDMA0CKEN**	**LPPDMA0 Clock Enable Bit** 0 = LPPDMA0 clock Disabled. 1 = LPPDMA0 clock Enabled.

Peripheral Clock Keep Control Register 0 (LPSCC_CLKKEEP0)

The bits in this register are used to enable/disable clock for low power domain peripherial clock.

Register	Offset	R/W	Description	Reset Value
LPSCC_CLKKEEP0	LPSCC_BA+0x44	R/W	Peripheral Clock Keep Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				OPAKEEP	Reserved		LPADC0KEEP
23	22	21	20	19	18	17	16
TTMR1KEEP	TTMR0KEEP	LPTMR1KEEP	LPTMR0KEEP	LPUART0KEEP	LPI2C0KEEP	LPSPI0KEEP	WDTKEEP
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					LPSRAMKEEP	LPGPIOKEEP	LPPDMA0KEEP

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**OPAKEEP**	**OP Amplifier Clock Keep Bit** 0 = OPA clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = OPA clock Enabled when entering NPD0~5/SPD0~2 mode.
[26:25]	**Reserved**	Reserved.
[24]	**LPADC0KEEP**	**LPADC0 Clock Keep Bit** 0 = LPADC0 clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = LPADC0 clock Enabled when entering NPD0~5/SPD0~2 mode.
[23]	**TTMR1KEEP**	**TTMR1 Clock Keep Bit** 0 = TTMR1 clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = TTMR1 clock Enabled when entering NPD0~5/SPD0~2 mode.
[22]	**TTMR0KEEP**	**TTMR0 Clock Keep Bit** 0 = TTMR0 clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = TTMR0 clock Enabled when entering NPD0~5/SPD0~2 mode.
[21]	**LPTMR1KEEP**	**LPTMR1 Clock Keep Bit** 0 = LPTMR1 clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = LPTMR1 clock Enabled when entering NPD0~5/SPD0~2 mode.
[20]	**LPTMR0KEEP**	**LPTMR0 Clock Keep Bit** 0 = LPTMR0 clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = LPTMR0 clock Enabled when entering NPD0~5/SPD0~2 mode.
[19]	**LPUART0KEEP**	**LPUART0 Clock Keep Bit** 0 = LPUART0 clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = LPUART0 clock Enabled when entering NPD0~5/SPD0~2 mode.
[18]	**LPI2C0KEEP**	**LPI2C0 Clock Keep Bit**

		0 = LPI2C0 clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = LPI2C0 clock Enabled when entering NPD0~5/SPD0~2 mode.
[17]	**LPSPI0KEEP**	**LPSPI0 Clock Keep Bit** 0 = LPSPI0 clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = LPSPI0 clock Enabled when entering NPD0~5/SPD0~2 mode.
[16]	**WDTKEEP**	**WDT Clock Keep Bit** 0 = WDT clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = WDT clock Enabled when entering NPD0~5/SPD0~2 mode.
[15:3]	**Reserved**	Reserved.
[2]	**LPSRAMKEEP**	**LPSRAM Clock Keep Bit** 0 = LPSRAM clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = LPSRAM clock Enabled when entering NPD0~5/SPD0~2 mode.
[1]	**LPGPIOKEEP**	**LPGPIO Clock Keep Bit** 0 = LPGPIO clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = LPGPIO clock Enabled when entering NPD0~5/SPD0~2 mode.
[0]	**LPPDMA0KEEP**	**LPPDMA0 Clock Keep Bit** 0 = LPPDMA0 clock Disabled when entering NPD0~5/SPD0~2 mode. 1 = LPPDMA0 clock Enabled when entering NPD0~5/SPD0~2 mode.

Peripheral Clock Source Select Control Register 0 (LPSCC_CLKSEL0)

Before clock switching, the related clock sources (pre-select and new-select) must be turned on.

Register	Offset	R/W	Description	Reset Value
LPSCC_CLKSEL0	LPSCC_BA+0x50	R/W	Peripheral Clock Source Select Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						WDTSEL	
23	22	21	20	19	18	17	16
Reserved						LPADC0SEL	
15	14	13	12	11	10	9	8
Reserved	LPTMR1SEL			Reserved	LPTMR0SEL		
7	6	5	4	3	2	1	0
TTMR1SEL		TTMR0SEL		LPSPI0SEL		LPUART0SEL	

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:24]	**WDTSEL**	**WDT Clock Source Selection** 00 = Clock source from 32 kHz internal low speed RC oscillator (LIRC). 01 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 10 = Clock source from HCLK1/2048. 11 = Reserved. **Note:** The WDTSEL cannot be changed when WDT is operating. User should make WDT disable before changing WDTSEL, and reset WDT after changing WDTSEL. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[23:18]	**Reserved**	Reserved.
[17:16]	**LPADC0SEL**	**LPADC0 Clock Source Selection** 00 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 01 = Clock source from 1~24 MHz internal middle speed RC oscillator (MIRC). 10 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 11 = Clock source from PCLK2. **Note:** The LPADC0SEL cannot be changed when LPADC0 is operating. User should make LPADC0 disable before changing LPADC0SEL, and reset LPADC0 after changing LPADC0SEL.
[15]	**Reserved**	Reserved.
[14:12]	**LPTMR1SEL**	**LPTMR1 Clock Source Selection** 000 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 001 = Clock source from 1~24 MHz internal middle speed RC oscillator (MIRC). 010 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 011 = Clock source from 32 kHz internal low speed RC oscillator (LIRC). 100 = Clock source from external clock LPTM1 pin. Others = Reserved. **Note:** The LPTMR1SEL cannot be changed when LPTMR1 is operating. User should

		make LPTMR1 disable before changing LPTMR1SEL, and reset LPTMR1 after changing LPTMR1SEL.
[11]	**Reserved**	Reserved.
[10:8]	**LPTMR0SEL**	**LPTMR0 Clock Source Selection** 000 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 001 = Clock source from 1~24 MHz internal middle speed RC oscillator (MIRC). 010 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 011 = Clock source from 32 kHz internal low speed RC oscillator (LIRC). 100 = Clock source from external clock LPTM0 pin. Others = Reserved. **Note:** The LPTMR0SEL cannot be changed when LPTMR0 is operating. User should make LPTMR0 disable before changing LPTMR0SEL, and reset LPTMR0 after changing LPTMR0SEL.
[7:6]	**TTMR1SEL**	**TTMR1 Clock Source Selection** 00 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 01 = Clock source from 1~24 MHz internal middle speed RC oscillator (MIRC). 10 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 11 = Clock source from 32 kHz internal low speed RC oscillator (LIRC). **Note:** The TTMR1SEL cannot be changed when TTMR1 is operating. User should make TTMR1 disable before changing TTMR1SEL, and reset TTMR1 after changing TTMR1SEL.
[5:4]	**TTMR0SEL**	**TTMR0 Clock Source Selection** 00 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 01 = Clock source from 1~24 MHz internal middle speed RC oscillator (MIRC). 10 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 11 = Clock source from 32 kHz internal low speed RC oscillator (LIRC). **Note:** The TTMR0SEL cannot be changed when TTMR0 is operating. User should make TTMR0 disable before changing TTMR0SEL, and reset TTMR0 after changing TTMR0SEL.
[3:2]	**LPSPI0SEL**	**LPSPI0 Clock Source Selection** 00 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 01 = Clock source from 1~24 MHz internal middle speed RC oscillator (MIRC). 10 = Reserved. 11 = Reserved. **Note:** The LPSPI0SEL cannot be changed when LPSPI0 is operating. User should make LPSPI0 disable before changing LPSPI0SEL, and reset LPSPI0 after changing LPSPI0SEL.
[1:0]	**LPUART0SEL**	**LPUART0 Clock Source Selection** 00 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 01 = Clock source from 1~24 MHz internal middle speed RC oscillator (MIRC). 10 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 11 = Reserved. **Note:** The LPUART0SEL cannot be changed when LPUART0 is operating. User should make LPUART0 disable before changing LPUART0SEL, and reset LPUART0 after changing LPUART0SEL.

Peripheral Clock Divider Number Register 0 (LPSCC_CLKDIV0)

Register	Offset	R/W	Description	Reset Value
LPSCC_CLKDIV0	LPSCC_BA+0x60	R/W	Peripheral Clock Divider Number Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**LPADC0DIV**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**LPUART0DIV**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**APB2DIV**			**HCLK1DIV**			

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**LPADC0DIV**	**LPADC0 Clock Divide Number from LPADC0 Clock Source** LPADC0 clock frequency = (LPADC0 clock source frequency) / (LPADC0DIV + 1).
[15:12]	**Reserved**	Reserved.
[11:8]	**LPUART0DIV**	**LPUART0 Clock Divide Number from LPUART0 Clock Source** LPUART0 clock frequency = (LPUART0 clock source frequency) / (LPUART0DIV + 1).
[7]	**Reserved**	Reserved.
[6:4]	**APB2DIV**	**APB2 Clock Divider** APB2 clock can be divided from HCLK1. 000 = PCLK2 frequency is HCLK1. 001 = PCLK2 frequency is 1/2 HCLK1. 010 = PCLK2 frequency is 1/4 HCLK1. 011 = PCLK2 frequency is 1/8 HCLK1. 100 = PCLK2 frequency is 1/16 HCLK1. Others = Reserved.
[3:0]	**HCLK1DIV**	**HCLK1 Clock Divide Number from HCLK1 Clock Source** HCLK1 clock frequency = (HCLK1 clock source frequency) / (HCLK1DIV + 1).

Clock Monitor Control Register (LPSCC_CLKMCTL)

Register	Offset	R/W	Description	Reset Value
LPSCC_CLKMCTL	LPSCC_BA+0x80	R/W	Clock Monitor Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						AOCMEN2	AOCMEN1
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						CLKM1SEL	
7	6	5	4	3	2	1	0
Reserved						CLKM0SEL	

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**AOCMEN2**	**Auto-operation Clock Monitor Enable 2** 0 = Auto-operation clock monitor function is Disabled. 1 = Auto-operation clock monitor function is Enabled. If clock monitor source (selected by CLKM0SEL) is active, PF.2 will output high. If clock monitor source (selected by CLKM1SEL) is active, PF.3 will output high. **Note:** When crystal is connected to chip, DO NOT set this bit to 1, it may damage chip IO.
[24]	**AOCMEN1**	**Auto-operation Clock Monitor Enable 1** 0 = Auto-operation clock monitor function is Disabled. 1 = Auto-operation clock monitor function is Enabled. If clock monitor source (selected by CLKM0SEL) is active, PF.0 will output high. If clock monitor source (selected by CLKM1SEL) is active, PF.1 will output high.
[23:10]	**Reserved**	Reserved.
[9:8]	**CLKM1SEL**	**Clock Monitor 1 Source Select** 00 = Clock monitor source is HCLK1. 01 = Clock monitor source is HIRC. 10 = Clock monitor source is MIRC. Others = Reserved.
[7:2]	**Reserved**	Reserved.
[1:0]	**CLKM0SEL**	**Clock Monitor 0 Source Select** 00 = Clock monitor source is HCLK1. 01 = Clock monitor source is HIRC. 10 = Clock monitor source is MIRC. Others = Reserved.

Low Power SRAM Control Register (LPSCC_SRAMCTL)

Register	Offset	R/W	Description	Reset Value
LPSCC_SRAMCTL	LPSCC_BA+0xA0	R/W	Low Power SRAM Control Register	0x0000_0010

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		LPSRAMDRL	LPSRAMDR	Reserved			LPSRAMEN

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**LPSRAMDRL**	**LPSRAM low power Retention at idle Enabled** 0 = LPSRAM retention at idle only (default). 1 = LPSRAM retention at idle with low power mode. **Note**: This control works when LPSRAMDR=1.
[4]	**LPSRAMDR**	**LPSRAM Dynamic Retention Enable Bit** 0 = LPSRAM dynamic retention function is disabled. LPSRAM is in active mode when chip in normal run/idle/NPDx mode. 1 = LPSRAM dynamic retention function is enabled when chip is in normal run/idle/NPDx mode. **Note 1**: If this bit is 1, LPSRAM will be auto changed to active mode when LPPDMA accesses LPSRAM. **Note 2:** This bit does not work if LSRETSEL(CLK_PMUCTL[16]) is 0 when chip in NPD3/NPD4/NPD5 mode.
[0]	**LPSRAMEN**	**Force LPSRAM Enable Bit** 0 = LPSRAM is controlled by LPSRAMDR(LPSCC_SRAMCTL[4]) and LSRETSEL(CLK_PMUCTL[16]). 1 = Force LPSRAM to active mode when chip is in normal run/idle/NPDx mode.

I/O Hold Control Register (LPSCC_IOHCTL)

Register	Offset	R/W	Description	Reset Value
LPSCC_IOHCTL	LPSCC_BA+0x100	R/W	I/O Hold Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SCHDIS**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SCHDIS**	**I/O Schmitt-Trigger Disable Bit** When chip enter NPD3/NPD4/NPD5, the I/O will be hold and the input mode is default at Schmitt-Trigger mode. Setting this bit can change all I/O input mode to TTL mode. 0 = I/O hold with Schmitt-Trigger iput mode. 1 = I/O hold with TTL input mode.

6.2.17 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
SYS Base Address: **SYS_BA = 0x4000_0000**				
SYS_PDID	SYS_BA+0x00	R	Part Device Identification Number Register	0xXXXX_XXXX [1]
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Status Register	0x0000_0043
SYS_IPRST0	SYS_BA+0x08	R/W	Peripheral Reset Control Register 0	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	Peripheral Reset Control Register 1	0x0000_0000
SYS_IPRST2	SYS_BA+0x10	R/W	Peripheral Reset Control Register 2	0x0000_0000
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-out Detector Control Register	0x000X_838X
SYS_IVSCTL	SYS_BA+0x1C	R/W	Internal Voltage Source Control Register	0x0000_0000
SYS_IPRST3	SYS_BA+0x20	R/W	Peripheral Reset Control Register 3	0x0000_0000
SYS_VREFCTL	SYS_BA+0x28	R/W	V_{REF} Control Register	0x00xx_0000
SYS_USBPHY	SYS_BA+0x2C	R/W	USB PHY Control Register	0x0003_0007
SYS_UTCPDCTL	SYS_BA+0x30	R/W	UTCPD Control Register	0x0000_0000
SYS_GPA_MFOS	SYS_BA+0x80	R/W	GPIOA Multiple Function Output Select Register	0x0000_0000
SYS_GPB_MFOS	SYS_BA+0x84	R/W	GPIOB Multiple Function Output Select Register	0x0000_0000
SYS_GPC_MFOS	SYS_BA+0x88	R/W	GPIOC Multiple Function Output Select Register	0x0000_0000
SYS_GPD_MFOS	SYS_BA+0x8C	R/W	GPIOD Multiple Function Output Select Register	0x0000_0000
SYS_GPE_MFOS	SYS_BA+0x90	R/W	GPIOE Multiple Function Output Select Register	0x0000_0000
SYS_GPF_MFOS	SYS_BA+0x94	R/W	GPIOF Multiple Function Output Select Register	0x0000_0000
SYS_GPG_MFOS	SYS_BA+0x98	R/W	GPIOG Multiple Function Output Select Register	0x0000_0000
SYS_GPH_MFOS	SYS_BA+0x9C	R/W	GPIOH Multiple Function Output Select Register	0x0000_0000
SYS_MIRCTCTL	SYS_BA+0xB0	R/W	MIRC1M Trim Control Register	0x0008_0000
SYS_MIRCTIEN	SYS_BA+0xB4	R/W	MIRC1M Trim Interrupt Enable Register	0x0000_0000
SYS_MIRCTISTS	SYS_BA+0xB8	R/W	MIRC1M Trim Interrupt Status Register	0x0000_0000
SYS_SRAM_INTCTL	SYS_BA+0xC0	R/W	System SRAM Interrupt Enable Control Register	0x0000_0000
SYS_SRAM_STATUS	SYS_BA+0xC4	R/W	System SRAM Parity Error Status Register	0x0000_0000
SYS_SRAM_ERRADDR	SYS_BA+0xC8	R	System SRAM Parity Check Error Address Register	0x0000_0000
SYS_SRAM_BISTCTL	SYS_BA+0xD0	R/W	System SRAM BIST Test Control Register	0x0000_0000
SYS_SRAM_BISTSTS	SYS_BA+0xD4	R	System SRAM BIST Test Status Register	0x00xx_00xx

SYS_SRAMPC0	SYS_BA+0xDC	R/W	SRAM Power Mode Control Register 0	0x0000_0000
SYS_HIRCTCTL	SYS_BA+0xE4	R/W	HIRC48M Trim Control Register	0x0008_0000
SYS_HIRCTIEN	SYS_BA+0xE8	R/W	HIRC48M Trim Interrupt Enable Register	0x0000_0000
SYS_HIRCTISTS	SYS_BA+0xEC	R/W	HIRC48M Trim Interrupt Status Register	0x0000_0000
SYS_IRCTCTL	SYS_BA+0xF0	R/W	HIRC Trim Control Register	0x0008_0000
SYS_IRCTIEN	SYS_BA+0xF4	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000
SYS_IRCTISTS	SYS_BA+0xF8	R/W	HIRC Trim Interrupt Status Register	0x0000_0000
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock Control Register	0x0000_0000
SYS_PMLDOCTL	SYS_BA+0x168	R/W	MEGPM LDO Control Register	0xX001_0107
SYS_PORDISAN	SYS_BA+0x1EC	R/W	Analog POR Disable Control Register	0x0000_0000
SYS_CSERVER	SYS_BA+0x1F4	R	Chip Series Version Register	0x0000_00XX
SYS_PLCTL	SYS_BA+0x1F8	R/W	Power Level Control Register	0x0000_0001
SYS_PLSTS	SYS_BA+0x1FC	R/W	Power Level Status Register	0x0000_0XXX
SYS_INIVTOR	SYS_BA+0x310	R/W	Initial VTOR Control Register	0x0000_0000
SYS_GPA_MFP0	SYS_BA+0x500	R/W	GPIOA Multiple Function Control Register 0	0x0000_0000
SYS_GPA_MFP1	SYS_BA+0x504	R/W	GPIOA Multiple Function Control Register 1	0x0000_0000
SYS_GPA_MFP2	SYS_BA+0x508	R/W	GPIOA Multiple Function Control Register 2	0x0000_0000
SYS_GPA_MFP3	SYS_BA+0x50C	R/W	GPIOA Multiple Function Control Register 3	0x0000_0000
SYS_GPB_MFP0	SYS_BA+0x510	R/W	GPIOB Multiple Function Control Register 0	0x0000_0000
SYS_GPB_MFP1	SYS_BA+0x514	R/W	GPIOB Multiple Function Control Register 1	0x0000_0000
SYS_GPB_MFP2	SYS_BA+0x518	R/W	GPIOB Multiple Function Control Register 2	0x0000_0000
SYS_GPB_MFP3	SYS_BA+0x51C	R/W	GPIOB Multiple Function Control Register 3	0x0000_0000
SYS_GPC_MFP0	SYS_BA+0x520	R/W	GPIOC Multiple Function Control Register 0	0x0000_0000
SYS_GPC_MFP1	SYS_BA+0x524	R/W	GPIOC Multiple Function Control Register 1	0x0000_0000
SYS_GPC_MFP2	SYS_BA+0x528	R/W	GPIOC Multiple Function Control Register 2	0x0000_0000
SYS_GPC_MFP3	SYS_BA+0x52C	R/W	GPIOC Multiple Function Control Register 3	0x0000_0000
SYS_GPD_MFP0	SYS_BA+0x530	R/W	GPIOD Multiple Function Control Register 0	0x0000_0000
SYS_GPD_MFP1	SYS_BA+0x534	R/W	GPIOD Multiple Function Control Register 1	0x0000_0000
SYS_GPD_MFP2	SYS_BA+0x538	R/W	GPIOD Multiple Function Control Register 2	0x0000_0000
SYS_GPD_MFP3	SYS_BA+0x53C	R/W	GPIOD Multiple Function Control Register 3	0x0000_0000
SYS_GPE_MFP0	SYS_BA+0x540	R/W	GPIOE Multiple Function Control Register 0	0x0000_0000

SYS_GPE_MFP1	SYS_BA+0x544	R/W	GPIOE Multiple Function Control Register 1	0x0000_0000
SYS_GPE_MFP2	SYS_BA+0x548	R/W	GPIOE Multiple Function Control Register 2	0x0000_0000
SYS_GPE_MFP3	SYS_BA+0x54C	R/W	GPIOE Multiple Function Control Register 3	0x0000_0000
SYS_GPF_MFP0	SYS_BA+0x550	R/W	GPIOF Multiple Function Control Register 0	0x0000_0E0E
SYS_GPF_MFP1	SYS_BA+0x554	R/W	GPIOF Multiple Function Control Register 1	0x0000_0000
SYS_GPF_MFP2	SYS_BA+0x558	R/W	GPIOF Multiple Function Control Register 2	0x0000_0000
SYS_GPG_MFP0	SYS_BA+0x560	R/W	GPIOG Multiple Function Control Register 0	0x0000_0000
SYS_GPG_MFP1	SYS_BA+0x564	R/W	GPIOG Multiple Function Control Register 1	0x0000_0000
SYS_GPG_MFP2	SYS_BA+0x568	R/W	GPIOG Multiple Function Control Register 2	0x0000_0000
SYS_GPG_MFP3	SYS_BA+0x56C	R/W	GPIOG Multiple Function Control Register 3	0x0000_0000
SYS_GPH_MFP1	SYS_BA+0x574	R/W	GPIOH Multiple Function Control Register 1	0x0000_0000
SYS_GPH_MFP2	SYS_BA+0x578	R/W	GPIOH Multiple Function Control Register 2	0x0000_0000

6.2.18 Register Description

Part Device Identification Number Register (SYS_PDID)

Register	Offset	R/W	Description	Reset Value
SYS_PDID	SYS_BA+0x00	R	Part Device Identification Number Register	0xXXXX_XXXX [1]

[1] Every part number has a unique default reset value.

31	30	29	28	27	26	25	24
PDID							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PDID							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PDID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PDID							

Bits	Description	
[31:0]	**PDID**	**Part Device Identification Number (Read Only)** This register reflects device part number code. Software can read this register to identify which device is used.

System Reset Status Register (SYS_RSTSTS)

This register provides specific information for software to identify this chip's reset source from last operation.

Register	Offset	R/W	Description	Reset Value
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Status Register	0x0000_0043

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							CPULKRF
7	6	5	4	3	2	1	0
CPURF	HRESETRF	SYSRF	BODRF	LVRF	WDTRF	PINRF	PORF

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CPULKRF**	**CPU Lockup Reset Flag** 0 = No reset from CPU lockup happened. 1 = The Cortex-M23 lockup happened and chip is reset. **Note 1:** Write 1 to clear this bit to 0. **Note 2:** When CPU lockup happened under ICE is connected, this flag will set to 1 but chip will not reset.
[7]	**CPURF**	**CPU Reset Flag** The CPU reset flag is set by hardware if software writes CPURST (SYS_IPRST0[1]) 1 to reset Cortex-M23 core and RRAM Memory Controller (RMC). 0 = No reset from CPU. 1 = The Cortex-M23 core and RMC are reset by software setting CPURST to 1. **Note:** Write 1 to clear this bit to 0.
[6]	**HRESETRF**	**HRESET Reset Flag** The HRESET reset flag is set by the "Reset Signal" from the HRESET. 0 = No reset from HRESET. 1 = Reset from HRESET. **Note:** Write 1 to clear this bit to 0.
[5]	**SYSRF**	**System Reset Flag** The system reset flag is set by the "Reset Signal" from the Cortex-M23 core to indicate the previous reset source. 0 = No reset from Cortex-M23. 1 = The Cortex-M23 had issued the reset signal to reset the system by writing 1 to the bit SYSRESETREQ(AIRCR[2], Application Interrupt and Reset Control Register, address = 0xE000ED0C) in system control registers of Cortex-M23 core. **Note:** Write 1 to clear this bit to 0.

Bits	Description	
[4]	**BODRF**	**BOD Reset Flag** The BOD reset flag is set by the “Reset Signal” from the Brown-Out Detector to indicate the previous reset source. 0 = No reset from BOD. 1 = The BOD had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[3]	**LVRF**	**LVR Reset Flag** The LVR reset flag is set by the “Reset Signal” from the Low Voltage Reset Controller to indicate the previous reset source. 0 = No reset from LVR. 1 = LVR controller had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[2]	**WDTRF**	**WDT Reset Flag** The WDT reset flag is set by the “Reset Signal” from the Watchdog Timer or Window Watchdog Timer to indicate the previous reset source. 0 = No reset from watchdog timer or window watchdog timer. 1 = The watchdog timer or window watchdog timer had issued the reset signal to reset the system. **Note 1:** Write 1 to clear this bit to 0. **Note 2:** Watchdog Timer register RSTF(WDT_CTL[2]) bit is set if the system has been reset by WDT time-out reset. Window Watchdog Timer register WWDTRF(WWDT_STATUS[1]) bit is set if the system has been reset by WWDT time-out reset.
[1]	**PINRF**	**NRESET Pin Reset Flag** The nRESET pin reset flag is set by the “Reset Signal” from the nRESET Pin to indicate the previous reset source. 0 = No reset from nRESET pin. 1 = Pin nRESET had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[0]	**PORF**	**POR Reset Flag** The POR reset flag is set by the “Reset Signal” from the Power-on Reset (POR) Controller or bit CHIPRST (SYS_IPRST0[0]) to indicate the previous reset source. 0 = No reset from POR or CHIPRST. 1 = Power-on Reset (POR) or CHIPRST had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.

Peripheral Reset Control Register 0 (SYS_IPRST0)

Register	Offset	R/W	Description	Reset Value
SYS_IPRST0	SYS_BA+0x08	R/W	Peripheral Reset Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		CANFD1RST	CANFD0RST	Reserved			
15	14	13	12	11	10	9	8
Reserved			CRPTRST	Reserved			
7	6	5	4	3	2	1	0
CRCRST	Reserved		USBHRST	EBIRST	PDMA0RST	CPURST	CHIPRST

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21]	**CANFD1RST**	**CANFD1 Controller Reset** Setting this bit to 1 will generate a reset signal to the CANFD1 controller. User needs to set this bit to 0 to release from the reset state. 0 = CANFD1 controller normal operation. 1 = CANFD1 controller reset.
[20]	**CANFD0RST**	**CANFD0 Controller Reset** Setting this bit to 1 will generate a reset signal to the CANFD0 controller. User needs to set this bit to 0 to release from the reset state. 0 = CANFD0 controller normal operation. 1 = CANFD0 controller reset.
[19:13]	**Reserved**	Reserved.
[12]	**CRPTRST**	**CRYPTO Controller Reset (Write Protect)** Setting this bit to 1 will generate a reset signal to the CRYPTO controller. User needs to set this bit to 0 to release from the reset state. 0 = CRYPTO controller normal operation. 1 = CRYPTO controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[11:8]	**Reserved**	Reserved.
[7]	**CRCRST**	**CRC Calculation Controller Reset (Write Protect)** Set this bit to 1 will generate a reset signal to the CRC calculation controller. User needs to set this bit to 0 to release from the reset state. 0 = CRC calculation controller normal operation. 1 = CRC calculation controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6:5]	**Reserved**	Reserved.
[4]	**USBHRST**	**USBH Controller Reset (Write Protect)**

		Set this bit to 1 will generate a reset signal to the USBH. User needs to set this bit to 0 to release from the reset state. 0 = USBH controller normal operation. 1 = USBH controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**EBIRST**	**EBI Controller Reset (Write Protect)** Set this bit to 1 will generate a reset signal to the EBI. User needs to set this bit to 0 to release from the reset state. 0 = EBI controller normal operation. 1 = EBI controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**PDMA0RST**	**PDMA0 Controller Reset (Write Protect)** Setting this bit to 1 will generate a reset signal to the PDMA0. User needs to set this bit to 0 to release from reset state. 0 = PDMA0 controller normal operation. 1 = PDMA0 controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1]	**CPURST**	**Processor Core One-shot Reset (Write Protect)** Setting this bit will only reset the processor core and RRAM Memory Controller(RMC), and this bit will automatically return to 0 after the 2 clock cycles. 0 = Processor core normal operation. 1 = Processor core one-shot reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**CHIPRST**	**Chip One-shot Reset (Write Protect)** Setting this bit will reset the whole chip, including Processor core and all peripherals, and this bit will automatically return to 0 after the 2 clock cycles. The CHIPRST is same as the POR reset, all the chip controllers is reset and the chip setting from RRAM are also reload. About the difference between CHIPRST and SYSRESETREQ(AIRCR[2]), please refer to section 7.2.2 0 = Chip normal operation. 1 = Chip one-shot reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Peripheral Reset Control Register 1 (SYS_IPRST1)

Setting these bits 1 will generate asynchronous reset signals to the corresponding module controller. Users need to set these bits to 0 to release corresponding module controller from reset state.

Register	Offset	R/W	Description	Reset Value
SYS_IPRST1	SYS_BA+0x0C	R/W	Peripheral Reset Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
TRNGRST	Reserved		EADC0RST	USBDRST	OTGRST	Reserved	
23	22	21	20	19	18	17	16
UART7RST	UART6RST	UART5RST	UART4RST	UART3RST	UART2RST	UART1RST	UART0RST
15	14	13	12	11	10	9	8
SPI2RST	SPI1RST	SPI0RST	QSPI0RST	I2C3RST	I2C2RST	I2C1RST	I2C0RST
7	6	5	4	3	2	1	0
ACMP01RST	Reserved	TMR3RST	TMR2RST	TMR1RST	TMR0RST	GPIORST	Reserved

Bits	Description	
[31]	**TRNGRST**	**TRNG Controller Reset** 0 = TRNG controller normal operation. 1 = TRNG controller reset.
[30:29]	**Reserved**	Reserved.
[28]	**EADC0RST**	**EADC0 Controller Reset** 0 = EADC0 controller normal operation. 1 = EADC0 controller reset.
[27]	**USBDRST**	**USBD Controller Reset** 0 = USBD controller normal operation. 1 = USBD controller reset.
[26]	**OTGRST**	**OTG Controller Reset** 0 = OTG controller normal operation. 1 = OTG controller reset.
[25:24]	**Reserved**	Reserved.
[23]	**UART7RST**	**UART7 Controller Reset** 0 = UART7 controller normal operation. 1 = UART7 controller reset.
[22]	**UART6RST**	**UART6 Controller Reset** 0 = UART6 controller normal operation. 1 = UART6 controller reset.
[21]	**UART5RST**	**UART5 Controller Reset** 0 = UART5 controller normal operation. 1 = UART5 controller reset.

[20]	**UART4RST**	**UART4 Controller Reset** 0 = UART4 controller normal operation. 1 = UART4 controller reset.
[19]	**UART3RST**	**UART3 Controller Reset** 0 = UART3 controller normal operation. 1 = UART3 controller reset.
[18]	**UART2RST**	**UART2 Controller Reset** 0 = UART2 controller normal operation. 1 = UART2 controller reset.
[17]	**UART1RST**	**UART1 Controller Reset** 0 = UART1 controller normal operation. 1 = UART1 controller reset.
[16]	**UART0RST**	**UART0 Controller Reset** 0 = UART0 controller normal operation. 1 = UART0 controller reset.
[15]	**SPI2RST**	**SPI2 Controller Reset** 0 = SPI2 controller normal operation. 1 = SPI2 controller reset.
[14]	**SPI1RST**	**SPI1 Controller Reset** 0 = SPI1 controller normal operation. 1 = SPI1 controller reset.
[13]	**SPI0RST**	**SPI0 Controller Reset** 0 = SPI0 controller normal operation. 1 = SPI0 controller reset.
[12]	**QSPI0RST**	**Qual SPI0 Controller Reset** 0 = Qual SPI0 controller normal operation. 1 = Qual SPI0 controller reset.
[11]	**I2C3RST**	**I2C3 Controller Reset** 0 = I2C3 controller normal operation. 1 = I2C3 controller reset.
[10]	**I2C2RST**	**I2C2 Controller Reset** 0 = I2C2 controller normal operation. 1 = I2C2 controller reset.`.
[9]	**I2C1RST**	**I2C1 Controller Reset** 0 = I2C1 controller normal operation. 1 = I2C1 controller reset.
[8]	**I2C0RST**	**I2C0 Controller Reset** 0 = I2C0 controller normal operation. 1 = I2C0 controller reset.
[7]	**ACMP01RST**	**ACMP01 Controller Reset** 0 = ACMP01 controller normal operation. 1 = ACMP01 controller reset.
[6]	**Reserved**	Reserved.

[5]	**TMR3RST**	**Timer3 Controller Reset** 0 = Timer3 controller normal operation. 1 = Timer3 controller reset.
[4]	**TMR2RST**	**Timer2 Controller Reset** 0 = Timer2 controller normal operation. 1 = Timer2 controller reset.
[3]	**TMR1RST**	**Timer1 Controller Reset** 0 = Timer1 controller normal operation. 1 = Timer1 controller reset.
[2]	**TMR0RST**	**Timer0 Controller Reset** 0 = Timer0 controller normal operation. 1 = Timer0 controller reset.
[1]	**GPIORST**	**GPIO Controller Reset** 0 = GPIO controller normal operation. 1 = GPIO controller reset.
[0]	**Reserved**	Reserved.

Peripheral Reset Control Register 2 (SYS_IPRST2)

Setting these bits to 1 will generate asynchronous reset signals to the corresponding module controller. Users need to set these bits to 0 to release corresponding module controller from reset state.

Register	Offset	R/W	Description	Reset Value
SYS_IPRST2	SYS_BA+0x10	R/W	Peripheral Reset Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				ECAP1RST	ECAP0RST	TKRST	Reserved
23	22	21	20	19	18	17	16
EQEI1RST	EQEI0RST	Reserved				EPWM1RST	EPWM0RST
15	14	13	12	11	10	9	8
Reserved			DACRST	WWDTRST	Reserved	USCI1RST	USCI0RST
7	6	5	4	3	2	1	0
Reserved	SPI3RST	Reserved					

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**ECAP1RST**	**ECAP1 Controller Reset** 0 = ECAP1 controller normal operation. 1 = ECAP1 controller reset.
[26]	**ECAP0RST**	**ECAP0 Controller Reset** 0 = ECAP0 controller normal operation. 1 = ECAP0 controller reset.
[25]	**TKRST**	**TK Controller Reset** 0 = TK controller normal operation. 1 = TK controller reset.
[24]	**Reserved**	Reserved.
[23]	**EQEI1RST**	**EQEI1 Controller Reset** 0 = EQEI1 controller normal operation. 1 = EQEI1 controller reset.
[22]	**EQEI0RST**	**EQEI0 Controller Reset** 0 = EQEI0 controller normal operation. 1 = EQEI0 controller reset.
[21:18]	**Reserved**	Reserved.
[17]	**EPWM1RST**	**EPWM1 Controller Reset** 0 = EPWM1 controller normal operation. 1 = EPWM1 controller reset.
[16]	**EPWM0RST**	**EPWM0 Controller Reset** 0 = EPWM0 controller normal operation.

		1 = EPWM0 controller reset.
[15:13]	**Reserved**	Reserved.
[12]	**DACRST**	**DAC Controller Reset** 0 = DAC controller normal operation. 1 = DAC controller reset.
[11]	**WWDTRST**	**WWDT Controller Reset** 0 = WWDT controller normal operation. 1 = WWDT controller reset.
[10]	**Reserved**	Reserved.
[9]	**USCI1RST**	**USCI1 Controller Reset** 0 = USCI1 controller normal operation. 1 = USCI1 controller reset.
[8]	**USCI0RST**	**USCI0 Controller Reset** 0 = USCI0 controller normal operation. 1 = USCI0 controller reset.
[7]	**Reserved**	Reserved.
[6]	**SPI3RST**	**SPI3 Controller Reset** 0 = SPI3 controller normal operation. 1 = SPI3 controller reset.
[5:0]	**Reserved**	Reserved.

Brown-out Detector Control Register (SYS_BODCTL)

Partial of the SYS_BODCTL control registers values are initiated by the RRAM configuration and partial bits are write-protected bit.

Register	Offset	R/W	Description	Reset Value
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-out Detector Control Register	0x000X_838X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				BODVL			
15	14	13	12	11	10	9	8
LVRRDY	LVRDGSEL			Reserved	BODDGSEL		
7	6	5	4	3	2	1	0
LVREN	BODOUT	BODLPM	BODIF	BODRSTEN	Reserved		BODEN

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**BODVL**	**Brown-out Detector Threshold Voltage Selection (Write Protect)** The default value is set by RRAM controller user configuration register CBOV ({1'b1, CONFIG0 [23:21]}). 0000 = Brown-Out Detector threshold voltage is 1.5V. 1000 = Brown-Out Detector threshold voltage is 1.6V. 1001 = Brown-Out Detector threshold voltage is 1.8V. 1010 = Brown-Out Detector threshold voltage is 2.0V. 1011 = Brown-Out Detector threshold voltage is 2.2V. 1100 = Brown-Out Detector threshold voltage is 2.4V. 1101 = Brown-Out Detector threshold voltage is 2.6V. 1110 = Brown-Out Detector threshold voltage is 2.8V. 1111 = Brown-Out Detector threshold voltage is 3.0V. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[15]	**LVRRDY**	**LVR Enable Ready Flag (Read Only)** 0 = LVR disabled and not ready. 1 = LVR enabled and ready.

Bits	Description	
[14:12]	**LVRDGSEL**	**LVR Output De-glitch Time Select (Write Protect)** 000 = Without de-glitch function. 001 = 4 system clock (HCLK). 010 = 8 system clock (HCLK). 011 = 16 system clock (HCLK). 100 = 32 system clock (HCLK). 101 = 64 system clock (HCLK). 110 = 128 system clock (HCLK). 111 = 256 system clock (HCLK). **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** In NPD3/NPD4/NPD5/SPD0~2 mode, LVR output is sampled by RC10K clock.
[11]	**Reserved**	Reserved.
[10:8]	**BODDGSEL**	**Brown-out Detector Output De-glitch Time Select (Write Protect)** 000 = BOD output is sampled by RC10K clock. 001 = 3~4 system clock (HCLK). 010 = 7~8 system clock (HCLK). 011 = 15~16 system clock (HCLK). 100 = 31~32 system clock (HCLK). 101 = 63~64 system clock (HCLK). 110 = 127~128 system clock (HCLK). 111 = 255~256 system clock (HCLK). **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** In NPD3/NPD4/NPD5/SPD0~2 mode, BOD output is sampled by RC10K clock.
[7]	**LVREN**	**Low Voltage Reset Enable Bit (Write Protect)** The LVR function resets the chip when the input power voltage is lower than LVR circuit setting. LVR function is enabled by default. 0 = Low Voltage Reset function Disabled. 1 = Low Voltage Reset function Enabled. **Note 1:** After enabling the bit, the LVR function will be active with 100us delay for LVR output stable (default). **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6]	**BODOUT**	**Brown-out Detector Output Status** 0 = Brown-out Detector output status is 0. It means the detected voltage is higher than BODVL setting or BODEN is 0. 1 = Brown-out Detector output status is 1. It means the detected voltage is lower than BODVL setting. If the BODEN is 0, BOD function disabled, this bit always responds 0000.
[5]	**BODLPM**	**Brown-out Detector Low Power Mode (Write Protect)** 0 = BOD operate in normal mode (default). 1 = BOD Low Power mode Enabled. **Note 1:** The BOD consumes about 100uA in normal mode, the low power mode can reduce the current to about 1/10 but slow the BOD response. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.

Bits	Description	
[4]	**BODIF**	**Brown-out Detector Interrupt Flag** 0 = Brown-out Detector does not detect any voltage draft at AV_{DD} down through or up through the voltage of BODVL setting. 1 = When Brown-out Detector detects the AV_{DD} is dropped down through the voltage of BODVL setting or the AV_{DD} is raised up through the voltage of BODVL setting, this bit is set to 1 and the brown-out interrupt is requested if brown-out interrupt is enabled. **Note:** Write 1 to clear this bit to 0.
[3]	**BODRSTEN**	**Brown-out Reset Enable Bit (Write Protect)** The default value is set by RRAM controller user configuration register CBORST(CONFIG0[20]) bit. 0 = Brown-out “INTERRUPT” function Enabled. 1 = Brown-out “RESET” function Enabled. **Note 1:** While the Brown-out Detector function is enabled (BODEN high) and BOD reset function is enabled (BODRSTEN high), BOD will assert a signal to reset chip when the detected voltage is lower than the threshold (BODOUT high). While the BOD function is enabled (BODEN high) and BOD interrupt function is enabled (BODRSTEN low), BOD will assert an interrupt if BODOUT is high. BOD interrupt will keep till to the BODEN set to 0. BOD interrupt can be blocked by disabling the NVIC BOD interrupt or disabling BOD function (set BODEN low). **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2:1]	**Reserved**	Reserved.
[0]	**BODEN**	**Brown-out Detector Enable Bit (Write Protect)** The default value is set by RRAM controller user configuration register CBODEN (CONFIG0 [19]). 0 = Brown-out Detector function Disabled. 1 = Brown-out Detector function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Internal Voltage Source Control Register (SYS_IVSCTL)

Register	Offset	R/W	Description	Reset Value
SYS_IVSCTL	SYS_BA+0x1C	R/W	Internal Voltage Source Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ADCCSEL	Reserved				AVDDDIV4EN	VBATUGEN	VTEMPEN

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**ADCCSEL**	**ADC Controller Select Bit** This bit is used to select ADC controller. 0 = EADC0 controlled is active. 1 = LPADC0 controller is active.
[6:3]	**Reserved**	Reserved.
[2]	**AVDDDIV4EN**	**AV_{DD} divide 4 Enable Bit** This bit is used to enable/disable AV_{DD} divide 4 function. 0 = AV_{DD} divide 4 function Disabled (default). 1 = AV_{DD} divide 4 function Enabled. **Note:** After this bit is set to 1, the value of AV_{DD} divide 4 output voltage can be obtained from ADC conversion result
[1]	**VBATUGEN**	**V_{BAT} Unity Gain Buffer Enable Bit** This bit is used to enable/disable V_{BAT} unity gain buffer function. 0 = V_{BAT} unity gain buffer function Disabled (default). 1 = V_{BAT} unity gain buffer function Enabled. **Note:** After this bit is set to 1, the value of V_{BAT} unity gain buffer output voltage can be obtained from ADC conversion result
[0]	**VTEMPEN**	**Temperature Sensor Enable Bit** This bit is used to enable/disable temperature sensor function. 0 = Temperature sensor function Disabled (default). 1 = Temperature sensor function Enabled. **Note:** Set this bit after setting VTEMPSEL(SYS_IVSCTL[4])=1.

Peripheral Reset Control Register 3 (SYS_IPRST3)

Setting these bits to 1 will generate asynchronous reset signals to the corresponding module controller. Users need to set these bits to 0 to release corresponding module controller from reset state.

Register	Offset	R/W	Description	Reset Value
SYS_IPRST3	SYS_BA+0x20	R/W	Peripheral Reset Control Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
UTCPD0RST	Reserved					PWM1RST	PWM0RST
7	6	5	4	3	2	1	0
ACMP2RST	Reserved						

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**UTCPD0RST**	**UTCPD0 Controller Reset** 0 = UTCPD0 controller normal operation. 1 = UTCPD0 controller reset.
[14:10]	**Reserved**	Reserved.
[9]	**PWM1RST**	**PWM1 Controller Reset** 0 = PWM1 controller normal operation. 1 = PWM1 controller reset.
[8]	**PWM0RST**	**PWM0 Controller Reset** 0 = PWM0 controller normal operation. 1 = PWM0 controller reset.
[7]	**ACMP2RST**	**ACMP2 Controller Reset** 0 = ACMP2 controller normal operation. 1 = ACMP2 controller reset.
[6:0]	**Reserved**	Reserved.

V_{REF} Control Register (SYS_VREFCTL)

Register	Offset	R/W	Description	Reset Value
SYS_VREFCTL	SYS_BA+0x28	R/W	V_{REF} Control Register	0x00xx_0000

31	30	29	28	27	26	25	24
Reserved							VBGFEN
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PRELOAD_SEL		Reserved	VREFCTL				

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**VBGFEN**	**Chip Internal Voltage Band-gap Force Enable Bit (Write Only)** 0 = Chip internal voltage band-gap controlled by EADC/ACMP/USB PHY if source selected. 1 = Chip internal voltage band-gap force enable. **Note :** This bit must be enabled if VBG voltage is measured by LPADC.
[23:8]	**Reserved**	Reserved.
[7:6]	**PRELOAD_SEL**	**Pre-load Timing Selection (Write Protect)** 00 = pre-load time is 60us for 0.1uF Capacitor. 01 = pre-load time is 310us for 1uF Capacitor. 10 = pre-load time is 1270us for 4.7uF Capacitor. 11 = pre-load time is 2650us for 10uF Capacitor.
[5]	**Reserved**	Reserved.
[4:0]	**VREFCTL**	**V_{REF} Control Bits (Write Protect)** 00000 = INT_VREF is disabled and V_{REF} is from external pin. 00011 = V_{REF} is internal 1.6V and V_{REF} pin must connect to external capacitor. 00111 = V_{REF} is internal 2.0V and V_{REF} pin must connect to external capacitor. 01011 = V_{REF} is internal 2.5V and V_{REF} pin must connect to external capacitor. 01111 = V_{REF} is internal 3.0V and V_{REF} pin must connect to external capacitor. Others = Reserved. **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** VREFCTL must be selected as 00000 if package doesn't include V_{REF} pin.

UTCPD Control Register (SYS_UTCPDCTL)

Register	Offset	R/W	Description	Reset Value
SYS_UTCPDCTL	SYS_BA+0x30	R/W	UTCPD Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			UDVBDETS	Reserved	PD0VBDSS		
7	6	5	4	3	2	1	0
Reserved						POREN0	IOMODE

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**UDVBDETS**	**UDC11 V_{BUS} Detect Source Select** 0 = UDC11 V_{BUS} detect source from OTGPHY. 1 = UDC11 V_{BUS} detect source from VBDETSW0. **Note:** Before use UDC11 function, UDVBDETS should be set and cannot change during UDC11 operating.
[11]	**Reserved**	Reserved.
[10:8]	**PD0VBDSS**	**UTCPD0 V_{BUS} Detect Source Select** UTCPD0 controller need a V_{BUS} detect result to note if V_{BUS} is connected. For SPD0~2/NPD3/NPD4/NPD5 usage, ACMP can be another voltage detect method to detect V_{BUS} pulg in or out. This bit field is used to select UTCPD0 V_{BUS} detect source. And the selected result is VBDETSW0 000 = UTCPD0 V_{BUS} detect source from UTCPD0 PHY. 001 = UTCPD0 V_{BUS} detect source from ACMP0 output. 010 = UTCPD0 V_{BUS} detect source from ACMP1 output. 011 = UTCPD0 V_{BUS} detect source from ACMP2 output. Others = Reserved. **Note:** Before use UTCPD0 function, PDVBDETS should be set and cannot change during UTCPD0 operating.
[7:2]	**Reserved**	Reserved.
[1]	**POREN0**	**UTCPD0 Power-on Enable Bit** 0 = UTCPD0 PHY in reset mode. 1 = UTCPD0 PHY in normal mode. **Note:** user should set POREN0 to 1 after IOMODE is setting down.
[0]	**IOMODE**	**UTCPD0 as I/O mode** This bit is used to define UTCPD0 CCx and CCDBx function 0 = Pin as UTCPD0 CCx and CCDBx function. 1 = Pin as general I/O function.

GPIO A-H Multiple Function Output Select Register (SYS_GPx_MFOS)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFOS	SYS_BA+0x80	R/W	GPIOA Multiple Function Output Select Register	0x0000_0000
SYS_GPB_MFOS	SYS_BA+0x84	R/W	GPIOB Multiple Function Output Select Register	0x0000_0000
SYS_GPC_MFOS	SYS_BA+0x88	R/W	GPIOC Multiple Function Output Select Register	0x0000_0000
SYS_GPD_MFOS	SYS_BA+0x8C	R/W	GPIOD Multiple Function Output Select Register	0x0000_0000
SYS_GPE_MFOS	SYS_BA+0x90	R/W	GPIOE Multiple Function Output Select Register	0x0000_0000
SYS_GPF_MFOS	SYS_BA+0x94	R/W	GPIOF Multiple Function Output Select Register	0x0000_0000
SYS_GPG_MFOS	SYS_BA+0x98	R/W	GPIOG Multiple Function Output Select Register	0x0000_0000
SYS_GPH_MFOS	SYS_BA+0x9C	R/W	GPIOH Multiple Function Output Select Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MFOS							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFOS							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**MFOS**	**GPIOA-H Pin[n] Multiple Function Pin Output Mode Select** This bit used to select multiple function pin output mode type for Px.n pin 0 = Multiple function pin output mode type is Push-pull mode. 1 = Multiple function pin output mode type is Open-drain mode. **Note:** For more information about Px.n, please refer to the “PIN CONFIGURATION” chapter.

MIRC1M Trim Control Register (SYS_MIRCTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_MIRCTCTL	SYS_BA+0xB0	R/W	MIRC1M Trim Control Register	0x0008_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			BOUNDARY				
15	14	13	12	11	10	9	8
Reserved					REFCKSEL	BOUNDEN	CESTOPEN
7	6	5	4	3	2	1	0
RETRYCNT		LOOPSEL		ACCURSEL		FREQSEL	

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20:16]	**BOUNDARY**	**Boundary Selection** Fill the boundary range from 0x1 to 0x31, 0x0 is reserved. **Note:** This field is effective only when the BOUNDEN(SYS_MIRCTRIMCTL[9]) is enabled.
[15:11]	**Reserved**	Reserved.
[10]	**REFCKSEL**	**Reference Clock Selection** 0 = MIRC trim reference clock is from LXT (32.768 kHz). 1 = Reserved. **Note:** MIRC trim reference clock is 20 kHz in test mode.
[9]	**BOUNDEN**	**Boundary Enable Bit** 0 = Boundary function Disabled. 1 = Boundary function Enabled.
[8]	**CESTOPEN**	**Clock Error Stop Enable Bit** 0 = The trim operation is keep going if clock is inaccuracy. 1 = The trim operation is stopped if clock is inaccuracy.
[7:6]	**RETRYCNT**	**Trim Value Update Limitation Count** This field defines that how many times the auto trim circuit will try to update the MIRC trim value before the frequency of MIRC locked. Once the MIRC locked, the internal trim value update counter will be reset. If the trim value update counter reached this limitation value and frequency of MIRC still doesn't lock, the auto trim operation will be disabled and FREQSEL will be cleared to 00. 00 = Trim retry count limitation is 64 loops. 01 = Trim retry count limitation is 128 loops. 10 = Trim retry count limitation is 256 loops. 11 = Trim retry count limitation is 512 loops.
[5:4]	**LOOPSEL**	**Trim Calculation Loop Selection** This field defines that trim value calculation is based on how many reference clocks.

		00 = Trim value calculation is based on average difference in 32 clocks of reference clock. 01 = Trim value calculation is based on average difference in 64 clocks of reference clock. 10 = Trim value calculation is based on average difference in 96 clocks of reference clock. 11 = Trim value calculation is based on average difference in 128 clocks of reference clock. **Note:** For example, if LOOPSEL is set as 00, auto trim circuit will calculate trim value based on the average frequency difference in 4 clocks of reference clock.
[3:2]	**ACCURSEL**	**Trim Accuracy Selection** This field indicates the target frequency accuracy of 1 MHz internal high speed RC oscillator (MIRC) auto trim. 00 = Accuracy is +-0.25% deviation within all temperature ranges. 01 = Accuracy is +-0.50% deviation within all temperature ranges. 10 = Accuracy is +-0.75% deviation within all temperature ranges. 11 = Accuracy is +-1% deviation within all temperature ranges.
[1:0]	**FREQSEL**	**Trim Frequency Selection** This field indicates the target frequency of 1 MHz internal high speed RC oscillator (MIRC) auto trim. During auto trim operation, if clock error detected with CESTOPEN is set to 1 or trim retry limitation count reached, this field will be cleared to 00 automatically. 00 = Disable MIRC auto trim function. 01 = Enable MIRC auto trim function and trim MIRC to 1 MHz. 10 = Reserved. 11 = Reserved.

MIRC1M Trim Interrupt Enable Register (SYS_MIRCTIEN)

Register	Offset	R/W	Description	Reset Value
SYS_MIRCTIEN	SYS_BA+0xB4	R/W	MIRC1M Trim Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CLKEIEN	TFAILIEN	Reserved

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CLKEIEN**	**Clock Error Interrupt Enable Bit** This bit controls if CPU would get an interrupt while clock is inaccuracy during auto trim operation. If this bit is set to1, and CLKERRIF(SYS_MIRCTISTS[2]) is set during auto trim operation, an interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = Disable CLKERRIF(SYS_MIRCTISTS[2]) status to trigger an interrupt to CPU. 1 = Enable CLKERRIF(SYS_MIRCTISTS[2]) status to trigger an interrupt to CPU.
[1]	**TFAILIEN**	**Trim Failure Interrupt Enable Bit** This bit controls if an interrupt will be triggered while MIRC trim value update limitation count reached and MIRC frequency still not locked on target frequency set by FREQSEL(SYS_MIRCTCTL[1:0]). If this bit is high and TFAILIF(SYS_MIRCTISTS[1]) is set during auto trim operation, an interrupt will be triggered to notify that MIRC trim value update limitation count was reached. 0 = Disable TFAILIF(SYS_MIRCTISTS[1]) status to trigger an interrupt to CPU. 1 = Enable TFAILIF(SYS_MIRCTISTS[1]) status to trigger an interrupt to CPU.
[0]	**Reserved**	Reserved.

MIRC1M Trim Interrupt Status Register (SYS_MIRCTISTS)

Register	Offset	R/W	Description	Reset Value
SYS_MIRCTISTS	SYS_BA+0xB8	R/W	MIRC1M Trim Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				OVBDIF	CLKERRIF	TFAILIF	FREQLOCK

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**OVBDIF**	**Over Boundary Status** When the over boundary function is set, if there occurs the over boundary condition, this flag will be set. 0 = Over boundary coundition did not occur. 1 = Over boundary coundition occurred. **Note :** Write 1 to clear this flag.
[2]	**CLKERRIF**	**Clock Error Interrupt Status** When the frequency of 32.768 kHz external low speed crystal oscillator (LXT) or 1 MHz internal high speed RC oscillator (MIRC) is shift larger to unreasonable value, this bit will be set and to be an indicate that clock frequency is inaccuracy Once this bit is set to 1, the auto trim operation stopped and FREQSEL(SYS_MIRCTCL[1:0]) will be cleared to 00 by hardware automatically if CESTOPEN(SYS_MIRCTCTL[8]) is set to 1. If this bit is set and CLKEIEN(SYS_MIRCTIEN[2]) is high, an interrupt will be triggered to notify the clock frequency is inaccuracy. Write 1 to clear this to 0. 0 = Clock frequency is accuracy. 1 = Clock frequency is inaccuracy.
[1]	**TFAILIF**	**Trim Failure Interrupt Status** This bit indicates that MIRC trim value update limitation count reached and the MIRC clock frequency still doesn't be locked. Once this bit is set, the auto trim operation stopped and FREQSEL(SYS_MIRCTCTL[1:0]) will be cleared to 00 by hardware automatically. If this bit is set and TFAILIEN(SYS_MIRCTIEN[1]) is high, an interrupt will be triggered to notify that MIRC trim value update limitation count was reached. Write 1 to clear this to 0. 0 = Trim value update limitation count does not reach. 1 = Trim value update limitation count reached and MIRC frequency still not locked.
[0]	**FREQLOCK**	**MIRC Frequency Lock Status** This bit indicates the MIRC frequency is locked. This is a status bit and doesn't trigger any interrupt Write 1 to clear this to 0. This bit will be set automatically, if the frequency is lock and the

		RC_TRIM is enabled. 0 = The internal high-speed oscillator frequency doesn't lock at 1 MHz yet. 1 = The internal high-speed oscillator frequency locked at 1 MHz.

System SRAM Parity Error Interrupt Enable Control Register (SYS_SRAM_INTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_INTCTL	SYS_BA+0xC0	R/W	System SRAM Interrupt Enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							PERRIEN

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PERRIEN**	**SRAM Parity Check Error Interrupt Enable Bit** 0 = SRAM parity check error interrupt Disabled. 1 = SRAM parity check error interrupt Enabled.

System SRAM Parity Check Status Register (SYS_SRAM_STATUS)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_STATUS	SYS_BA+0xC4	R/W	System SRAM Parity Error Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							PERRIF

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PERRIF**	**SRAM Parity Check Error Flag** This bit indicates the System SRAM parity error occurred. Write 1 to clear this to 0. 0 = No System SRAM parity error. 1 = System SRAM parity error occur.

System SRAM Parity Error Address Register (SYS_SRAM_ERRADDR)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_ERRADDR	SYS_BA+0xC8	R	System SRAM Parity Check Error Address Register	0x0000_0000

31	30	29	28	27	26	25	24
ERRADDR							
23	22	21	20	19	18	17	16
ERRADDR							
15	14	13	12	11	10	9	8
ERRADDR							
7	6	5	4	3	2	1	0
ERRADDR							

Bits	Description	
[31:0]	ERRADDR	**System SRAM Parity Error Address** This register shows system SRAM parity error byte address.

System SRAM BIST Test Control Register (SYS_SRAM_BISTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_BISTCTL	SYS_BA+0xD0	R/W	System SRAM BIST Test Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				LPSRBIST	Reserved		
7	6	5	4	3	2	1	0
Reserved			USBBIST	CANFDBIST	CRBIST	SRBIST1	SRBIST0

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**LPSRBIST**	**Low Power SRAM BIST Enable Bit (Write Protect)** This bit enables BIST test for LPSRAM. 0 = system LPSRAM BIST Disabled. 1 = system LPSRAM BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[10:5]	**Reserved**	Reserved.
[4]	**USBBIST**	**USB BIST Enable Bit (Write Protect)** This bit enables BIST test for USB RAM. 0 = system USB BIST Disabled. 1 = system USB BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**CANFDBIST**	**CANFDx BIST Enable Bit (Write Protect)** This bit enables BIST test for CANFDx RAM. 0 = system CANFDx BIST Disabled. 1 = system CANFDx BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**CRBIST**	**CACHE BIST Enable Bit (Write Protect)** This bit enables BIST test for CACHE RAM. 0 = system CACHE BIST Disabled. 1 = system CACHE BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1]	**SRBIST1**	**SRAM Bank1 BIST Enable Bit (Write Protect)** This bit enables BIST test for SRAM bank1. 0 = system SRAM bank1 BIST Disabled. 1 = system SRAM bank1 BIST Enabled.

		Note: This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**SRBIST0**	**SRAM Bank0 BIST Enable Bit (Write Protect)** This bit enables BIST test for SRAM bank0. 0 = system SRAM bank0 BIST Disabled. 1 = system SRAM bank0 BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

System SRAM BIST Test Status Register (SYS_SRAM_BISTSTS)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_BISTSTS	SYS_BA+0xD4	R	System SRAM BIST Test Status Register	0x00xx_00xx

31	30	29	28	27	26	25	24
Reserved				LPSRBEND			
23	22	21	20	19	18	17	16
Reserved			USBBEND	CANBEND	CRBEND	SRBEND1	SRBEND0
15	14	13	12	11	10	9	8
Reserved				LPSRBEF			
7	6	5	4	3	2	1	0
Reserved			USBBEF	CANBEF	CRBISTEF	SRBISTEF1	SRBISTEF0

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**LPSRBEND**	**Low Power SRAM BIST Test Finish** 0 = LPSRAM BIST is active. 1 = LPSRAM BIST test finish.
[26:21]	**Reserved**	Reserved.
[20]	**USBBEND**	**USB SRAM BIST Test Finish** 0 = USB SRAM BIST is active. 1 = USB SRAM BIST test finish.
[19]	**CANBEND**	**CAN SRAM BIST Test Finish** 0 = CAN SRAM BIST is active. 1 = CAN SRAM BIST test finish.
[18]	**CRBEND**	**CACHE SRAM BIST Test Finish** 0 = System CACHE RAM BIST is active. 1 = System CACHE RAM BIST test finish.
[17]	**SRBEND1**	**2nd SRAM BIST Test Finish** 0 = 2nd system SRAM BIST is active. 1 = 2nd system SRAM BIST finish.
[16]	**SRBEND0**	**1st SRAM BIST Test Finish** 0 = 1st system SRAM BIST active. 1 =1st system SRAM BIST finish.
[15:12]	**Reserved**	Reserved.
[11]	**LPSRBEF**	**Low Power SRAM BIST Fail Flag** 0 = LPSRAM BIST test pass. 1 = LPSRAM BIST test fail.
[10:5]	**Reserved**	Reserved.

[4]	**USBBEF**	**USB SRAM BIST Fail Flag** 0 = USB SRAM BIST test pass. 1 = USB SRAM BIST test fail.
[3]	**CANBEF**	**CAN SRAM BIST Fail Flag** 0 = CAN SRAM BIST test pass. 1 = CAN SRAM BIST test fail.
[2]	**CRBISTEF**	**CACHE SRAM BIST Fail Flag** 0 = System CACHE RAM BIST test pass. 1 = System CACHE RAM BIST test fail.
[1]	**SRBISTEF1**	**2nd System SRAM BIST Fail Flag** 0 = 2nd system SRAM BIST test pass. 1 = 2nd system SRAM BIST test fail.
[0]	**SRBISTEF0**	**1st System SRAM BIST Fail Flag** 0 = 1st system SRAM BIST test pass. 1 = 1st system SRAM BIST test fail.

SRAM Power Mode Control Register 0 (SYS_SRAMPC0)

Register	Offset	R/W	Description	Reset Value
SYS_SRAMPC0	SYS_BA+0xDC	R/W	SRAM Power Mode Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
PCBUSY	Reserved				SRAM6PM		
23	22	21	20	19	18	17	16
Reserved	SRAM5PM			Reserved	SRAM4PM		
15	14	13	12	11	10	9	8
Reserved	SRAM3PM			Reserved	SRAM2PM		
7	6	5	4	3	2	1	0
Reserved	SRAM1PM			Reserved	SRAM0PM		

Bits	Description	
[31]	**PCBUSY**	**Power Changing Busy Flag (Read Only)** This bit indicate SRAM power changing. 0 = SRAM power change finish. 1 = SRAM power changing.
[30:27]	**Reserved**	Reserved.
[26:24]	**SRAM6PM**	**SRAM Group 6 Power Mode Select (Write Protect)** This field can control SRAM group 6 in bank2 (8k) power mode for range 0x2800_0000 - 0x2800_1FFF. 000 = Normal mode. 010 = Retention mode. 011 = Shut down mode (No Retention). Other = Reserved. **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** Mode change need to wait PCBUSY=0. **Note 3:** Only support from normal mode to each Power-down mode and each Power-down mode to normal mode, not mode change between Power-down mode. **Note 4:** Power saving priority first. If CLK_PMUCTL setting SRAM at retention mode but SYS_SRAMPC0 setting to shut mode, SRAM go to shut down mode. **Note 5:** Beforce setting SRAM6PM, set LPSRAMDR(LPSCC_SRAMCTL[4])=0 firstly.
[23]	**Reserved**	Reserved.
[22:20]	**SRAM5PM**	**SRAM Group 5 Power Mode Select (Write Protect)** This field can control SRAM group 5 in bank1 (64k) power mode for range 0x2001_A000 - 0x2002_9FFF. 000 = Normal mode. 010 = Retention mode. 011 = Shut down mode (No Retention). Other = Reserved.

		Note 1: These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** Mode change need to wait PCBUSY=0. **Note 3:** Only support from normal mode to each Power-down mode and each Power-down mode to normal mode, not mode change between Power-down mode. **Note 4:** Power saving priority first. If CLK_PMUCTL setting SRAM at retention mode but SYS_SRAMPC0 setting to shut mode, SRAM go to shut down mode.
[19]	**Reserved**	Reserved.
[18:16]	**SRAM4PM**	**SRAM Group 4 Power Mode Select (Write Protect)** This field can control SRAM group 4 in bank1 (32k) power mode for range 0x2001_2000 - 0x2001_9FFF. 000 = Normal mode. 010 = Retention mode. 011 = Shut down mode (No Retention). Other = Reserved. **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** Mode change need to wait PCBUSY=0. **Note 3:** Only support from normal mode to each Power-down mode and each Power-down mode to normal mode, not mode change between Power-down mode. **Note 4:** Power saving priority first. If CLK_PMUCTL setting SRAM at retention mode but SYS_SRAMPC0 setting to shut mode, SRAM go to shut down mode.
[15]	**Reserved**	Reserved.
[14:12]	**SRAM3PM**	**SRAM Group 3 Power Mode Select (Write Protect)** This field can control SRAM group 3 in bank1 (32k) power mode for range 0x2000_A000 - 0x2001_1FFF. 000 = Normal mode. 010 = Retention mode. 011 = Shut down mode (No Retention). Other = Reserved. **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** Mode change need to wait PCBUSY=0. **Note 3:** Only support from normal mode to each Power-down mode and each Power-down mode to normal mode, not mode change between Power-down mode. **Note 4:** Power saving priority first. If CLK_PMUCTL setting SRAM at retention mode but SYS_SRAMPC0 setting to shut mode, SRAM go to shut down mode.
[11]	**Reserved**	Reserved.
[10:8]	**SRAM2PM**	**SRAM Group 2 Power Mode Select (Write Protect)** This field can control SRAM group2 in bank0 (16k) power mode for range 0x2000_6000 - 0x2000_9FFF. 000 = Normal mode. 010 = Retention mode. 011 = Shut down mode (No Retention). Other = Reserved. **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** Mode change need to wait PCBUSY=0. **Note 3:** Only support from normal mode to each Power-down mode and each Power-down mode to normal mode, not mode change between Power-down mode. **Note 4:** Power saving priority first. If CLK_PMUCTL setting SRAM at retention mode but SYS_SRAMPC0 setting to shut mode, SRAM go to shut down mode.
[7]	**Reserved**	Reserved.

[6:4]	**SRAM1PM**	**SRAM Group 1 Power Mode Select (Write Protect)** This field can control SRAM group1 in bank0 (16k) power mode for range 0x2000_2000 - 0x2000_5FFF. 000 = Normal mode. 010 = Retention mode. 011 = Shut down mode (No Retention). Other = Reserved. **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** Mode change need to wait PCBUSY=0. **Note 3:** Only support from normal mode to each Power-down mode and each Power-down mode to normal mode, not mode change between Power-down mode. **Note 4:** Power saving priority first. If CLK_PMUCTL setting SRAM at retention mode but SYS_SRAMPC0 setting to shut mode, SRAM go to shut down mode.
[3]	**Reserved**	Reserved.
[2:0]	**SRAM0PM**	**SRAM Group 0 Power Mode Select (Write Protect)** This field can control SRAM group0 in bank0 (8k) power mode for range 0x2000_0000 - 0x2000_1FFF. 000 = Normal mode. 010 = Retention mode. 011 = Shut down mode (No Retention). Other = Reserved. **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** Mode change need to wait PCBUSY=0. **Note 3:** Only support from normal mode to each Power-down mode and each Power-down mode to normal mode, not mode change between Power-down mode. **Note 4:** Power saving priority first. If CLK_PMUCTL setting SRAM at retention mode but SYS_SRAMPC0 setting to shut mode, SRAM go to shut down mode.

HIRC48M Trim Control Register (SYS_HIRCTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_HIRCTCTL	SYS_BA+0xE4	R/W	HIRC48M Trim Control Register	0x0008_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			BOUNDARY				
15	14	13	12	11	10	9	8
Reserved					REFCKSEL	BOUNDEN	CESTOPEN
7	6	5	4	3	2	1	0
RETRYCNT		LOOPSEL		ACCURSEL		FREQSEL	

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20:16]	**BOUNDARY**	**Boundary Selection** Fill the boundary range from 0x1 to 0x31, 0x0 is reserved. **Note:** This field is effective only when the BOUNDEN(SYS_HIRCTRIMCTL[9]) is enabled.
[15:11]	**Reserved**	Reserved.
[10]	**REFCKSEL**	**Reference Clock Selection** 0 = HIRC trim reference clock is from LXT (32.768 kHz). 1 = HIRC trim reference clock is from internal USB synchronous mode. **Note:** HIRC trim reference clock is 20 kHz in test mode.
[9]	**BOUNDEN**	**Boundary Enable Bit** 0 = Boundary function Disabled. 1 = Boundary function Enabled.
[8]	**CESTOPEN**	**Clock Error Stop Enable Bit** 0 = The trim operation is keep going if clock is inaccuracy. 1 = The trim operation is stopped if clock is inaccuracy.
[7:6]	**RETRYCNT**	**Trim Value Update Limitation Count** This field defines that how many times the auto trim circuit will try to update the HIRC trim value before the frequency of HIRC locked. Once the HIRC locked, the internal trim value update counter will be reset. If the trim value update counter reached this limitation value and frequency of HIRC still doesn't lock, the auto trim operation will be disabled and FREQSEL will be cleared to 00. 00 = Trim retry count limitation is 64 loops. 01 = Trim retry count limitation is 128 loops. 10 = Trim retry count limitation is 256 loops. 11 = Trim retry count limitation is 512 loops.
[5:4]	**LOOPSEL**	**Trim Calculation Loop Selection** This field defines that trim value calculation is based on how many reference clocks.

		00 = Trim value calculation is based on average difference in 4 clocks of reference clock. 01 = Trim value calculation is based on average difference in 8 clocks of reference clock. 10 = Trim value calculation is based on average difference in 16 clocks of reference clock. 11 = Trim value calculation is based on average difference in 32 clocks of reference clock. **Note:** For example, if LOOPSEL is set as 00, auto trim circuit will calculate trim value based on the average frequency difference in 4 clocks of reference clock.
[3:2]	**ACCURSEL**	**Trim Accuracy Selection** This field indicates the target frequency accuracy of 48 MHz internal high speed RC oscillator (HIRC) auto trim. 00 = Accuracy is +-0.25% deviation within all temperature ranges. 01 = Accuracy is +-0.50% deviation within all temperature ranges. 10 = Accuracy is +-0.75% deviation within all temperature ranges. 11 = Accuracy is +-1% deviation within all temperature ranges.
[1:0]	**FREQSEL**	**Trim Frequency Selection** This field indicates the target frequency of 48 MHz internal high speed RC oscillator (HIRC) auto trim. During auto trim operation, if clock error detected with CESTOPEN is set to 1 or trim retry limitation count reached, this field will be cleared to 00 automatically. 00 = Disable HIRC auto trim function. 01 = Enable HIRC auto trim function and trim HIRC to 48 MHz. 10 = Reserved. 11 = Reserved.

HIRC48M Trim Interrupt Enable Register (SYS_HIRCTIEN)

Register	Offset	R/W	Description	Reset Value
SYS_HIRCTIEN	SYS_BA+0xE8	R/W	HIRC48M Trim Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CLKEIEN	TFAILIEN	Reserved

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CLKEIEN**	**Clock Error Interrupt Enable Bit** This bit controls if CPU would get an interrupt while clock is inaccuracy during auto trim operation. If this bit is set to1, and CLKERRIF(SYS_IRCTISTS[2]) is set during auto trim operation, an interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = Disable CLKERRIF(SYS_IRCTISTS[2]) status to trigger an interrupt to CPU. 1 = Enable CLKERRIF(SYS_IRCTISTS[2]) status to trigger an interrupt to CPU.
[1]	**TFAILIEN**	**Trim Failure Interrupt Enable Bit** This bit controls if an interrupt will be triggered while HIRC trim value update limitation count reached and HIRC frequency still not locked on target frequency set by FREQSEL(SYS_IRCTCTL[1:0]). If this bit is high and TFAILIF(SYS_IRCTISTS[1]) is set during auto trim operation, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. 0 = Disable TFAILIF(SYS_IRCTISTS[1]) status to trigger an interrupt to CPU. 1 = Enable TFAILIF(SYS_IRCTISTS[1]) status to trigger an interrupt to CPU.
[0]	**Reserved**	Reserved.

HIRC48M Trim Interrupt Status Register (SYS_HIRCTISTS)

Register	Offset	R/W	Description	Reset Value
SYS_HIRCTISTS	SYS_BA+0xEC	R/W	HIRC48M Trim Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				OVBDIF	CLKERRIF	TFAILIF	FREQLOCK

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**OVBDIF**	**Over Boundary Status** When the over boundary function is set, if there occurs the over boundary condition, this flag will be set. 0 = Over boundary coundition did not occur. 1 = Over boundary coundition occurred. **Note:** Write 1 to clear this flag.
[2]	**CLKERRIF**	**Clock Error Interrupt Status** When the frequency of 32.768 kHz external low speed crystal oscillator (LXT) or 48 MHz internal high speed RC oscillator (HIRC) is shift larger to unreasonable value, this bit will be set and to be an indicate that clock frequency is inaccuracy Once this bit is set to 1, the auto trim operation stopped and FREQSEL(SYS_IRCTCL[1:0]) will be cleared to 00 by hardware automatically if CESTOPEN(SYS_IRCTCTL[8]) is set to 1. If this bit is set and CLKEIEN(SYS_IRCTIEN[2]) is high, an interrupt will be triggered to notify the clock frequency is inaccuracy. Write 1 to clear this to 0. 0 = Clock frequency is accuracy. 1 = Clock frequency is inaccuracy.
[1]	**TFAILIF**	**Trim Failure Interrupt Status** This bit indicates that HIRC trim value update limitation count reached and the HIRC clock frequency still doesn't be locked. Once this bit is set, the auto trim operation stopped and FREQSEL(SYS_IRCTCTL[1:0]) will be cleared to 00 by hardware automatically. If this bit is set and TFAILIEN(SYS_IRCTIEN[1]) is high, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. Write 1 to clear this to 0. 0 = Trim value update limitation count does not reach. 1 = Trim value update limitation count reached and HIRC frequency still not locked.
[0]	**FREQLOCK**	**HIRC Frequency Lock Status** This bit indicates the HIRC frequency is locked. This is a status bit and doesn't trigger any interrupt Write 1 to clear this to 0. This bit will be set automatically, if the frequency is lock and the

		RC_TRIM is enabled. 0 = The internal high-speed oscillator frequency doesn't lock at 48 MHz yet. 1 = The internal high-speed oscillator frequency locked at 48 MHz.

HIRC Trim Control Register (SYS_IRCTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTCTL	SYS_BA+0xF0	R/W	HIRC Trim Control Register	0x0008_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			BOUNDARY				
15	14	13	12	11	10	9	8
Reserved					REFCKSEL	BOUNDEN	CESTOPEN
7	6	5	4	3	2	1	0
RETRYCNT		LOOPSEL		ACCURSEL		FREQSEL	

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20:16]	**BOUNDARY**	**Boundary Selection** Fill the boundary range from 0x1 to 0x31, 0x0 is reserved. **Note:** This field is effective only when the BOUNDEN(SYS_HIRCTRIMCTL[9]) is enabled.
[15:11]	**Reserved**	Reserved.
[10]	**REFCKSEL**	**Reference Clock Selection** 0 = HIRC trim reference clock is from LXT (32.768 kHz). 1 = HIRC trim reference clock is from internal USB synchronous mode. **Note:** HIRC trim reference clock is 20 kHz in test mode.
[9]	**BOUNDEN**	**Boundary Enable Bit** 0 = Boundary function Disabled. 1 = Boundary function Enabled.
[8]	**CESTOPEN**	**Clock Error Stop Enable Bit** 0 = The trim operation is keep going if clock is inaccuracy. 1 = The trim operation is stopped if clock is inaccuracy.
[7:6]	**RETRYCNT**	**Trim Value Update Limitation Count** This field defines that how many times the auto trim circuit will try to update the HIRC trim value before the frequency of HIRC locked. Once the HIRC locked, the internal trim value update counter will be reset. If the trim value update counter reached this limitation value and frequency of HIRC still doesn't lock, the auto trim operation will be disabled and FREQSEL will be cleared to 00. 00 = Trim retry count limitation is 64 loops. 01 = Trim retry count limitation is 128 loops. 10 = Trim retry count limitation is 256 loops. 11 = Trim retry count limitation is 512 loops.
[5:4]	**LOOPSEL**	**Trim Calculation Loop Selection** This field defines that trim value calculation is based on how many reference clocks.

		00 = Trim value calculation is based on average difference in 4 clocks of reference clock. 01 = Trim value calculation is based on average difference in 8 clocks of reference clock. 10 = Trim value calculation is based on average difference in 16 clocks of reference clock. 11 = Trim value calculation is based on average difference in 32 clocks of reference clock. **Note:** For example, if LOOPSEL is set as 00, auto trim circuit will calculate trim value based on the average frequency difference in 4 clocks of reference clock.
[3:2]	**ACCURSEL**	**Trim Accuracy Selection** This field indicates the target frequency accuracy of 12 MHz internal high speed RC oscillator (IRC) auto trim. 00 = Accuracy is +-0.25% deviation within all temperature ranges. 01 = Accuracy is +-0.50% deviation within all temperature ranges. 10 = Accuracy is +-0.75% deviation within all temperature ranges. 11 = Accuracy is +-1% deviation within all temperature ranges.
[1:0]	**FREQSEL**	**Trim Frequency Selection** This field indicates the target frequency of 12 MHz internal high speed RC oscillator (HIRC) auto trim. During auto trim operation, if clock error detected with CESTOPEN is set to 1 or trim retry limitation count reached, this field will be cleared to 00 automatically. 00 = Disable HIRC auto trim function. 01 = Enable HIRC auto trim function and trim HIRC to 12 MHz. 10 = Reserved. 11 = Reserved.

HIRC Trim Interrupt Enable Register (SYS_IRCTIEN)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTIEN	SYS_BA+0xF4	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CLKEIEN	TFAILIEN	Reserved

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CLKEIEN**	**Clock Error Interrupt Enable Bit** This bit controls if CPU would get an interrupt while clock is inaccuracy during auto trim operation. If this bit is set to1, and CLKERRIF(SYS_IRCTISTS[2]) is set during auto trim operation, an interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = Disable CLKERRIF(SYS_IRCTISTS[2]) status to trigger an interrupt to CPU. 1 = Enable CLKERRIF(SYS_IRCTISTS[2]) status to trigger an interrupt to CPU.
[1]	**TFAILIEN**	**Trim Failure Interrupt Enable Bit** This bit controls if an interrupt will be triggered while HIRC trim value update limitation count reached and HIRC frequency still not locked on target frequency set by FREQSEL(SYS_IRCTCTL[1:0]). If this bit is high and TFAILIF(SYS_IRCTISTS[1]) is set during auto trim operation, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. 0 = Disable TFAILIF(SYS_IRCTISTS[1]) status to trigger an interrupt to CPU. 1 = Enable TFAILIF(SYS_IRCTISTS[1]) status to trigger an interrupt to CPU.
[0]	**Reserved**	Reserved.

HIRC Trim Interrupt Status Register (SYS_IRCTISTS)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTISTS	SYS_BA+0xF8	R/W	HIRC Trim Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				OVBDIF	CLKERRIF	TFAILIF	FREQLOCK

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**OVBDIF**	**Over Boundary Status** When the over boundary function is set, if there occurs the over boundary condition, this flag will be set. 0 = Over boundary coundition did not occur. 1 = Over boundary coundition occurred. **Note:** Write 1 to clear this flag.
[2]	**CLKERRIF**	**Clock Error Interrupt Status** When the frequency of 32.768 kHz external low speed crystal oscillator (LXT) or 12 MHz internal high speed RC oscillator (HIRC) is shift larger to unreasonable value, this bit will be set and to be an indicate that clock frequency is inaccuracy Once this bit is set to 1, the auto trim operation stopped and FREQSEL(SYS_IRCTCL[1:0]) will be cleared to 00 by hardware automatically if CESTOPEN(SYS_IRCTCTL[8]) is set to 1. If this bit is set and CLKEIEN(SYS_IRCTIEN[2]) is high, an interrupt will be triggered to notify the clock frequency is inaccuracy. Write 1 to clear this to 0. 0 = Clock frequency is accuracy. 1 = Clock frequency is inaccuracy.
[1]	**TFAILIF**	**Trim Failure Interrupt Status** This bit indicates that HIRC trim value update limitation count reached and the HIRC clock frequency still doesn't be locked. Once this bit is set, the auto trim operation stopped and FREQSEL(SYS_IRCTCTL[1:0]) will be cleared to 00 by hardware automatically. If this bit is set and TFAILIEN(SYS_IRCTIEN[1]) is high, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. Write 1 to clear this to 0. 0 = Trim value update limitation count does not reach. 1 = Trim value update limitation count reached and HIRC frequency still not locked.
[0]	**FREQLOCK**	**HIRC Frequency Lock Status** This bit indicates the HIRC frequency is locked. This is a status bit and doesn't trigger any interrupt Write 1 to clear this to 0. This bit will be set automatically, if the frequency is lock and the

		RC_TRIM is enabled. 0 = The internal high-speed oscillator frequency doesn't lock at 12 MHz yet. 1 = The internal high-speed oscillator frequency locked at 12 MHz.

Register Lock Control Register (SYS_REGLCTL)

Some of the system control registers need to be protected to avoid inadvertent write and disturb the chip operation. These system control registers are protected after the power-on reset till user to disable register protection. For user to program these protected registers, a register protection disable sequence needs to be followed by a special programming. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register SYS_REGLCTL address at 0x4000_0100 continuously. Any different data value, different sequence or any other write to other address during these three data writing will abort the whole sequence.

After the protection is disabled, user can check the protection disable bit at address 0x4000_0100 bit0, 1 is protection disable, and 0 is protection enable. Then user can update the target protected register value and then write any data to the address “0x4000_0100” to enable register protection.

This register is written to disable/enable register protection and read for the REGLCTL status.

Register	Offset	R/W	Description	Reset Value
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REGLCTL							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**REGLCTL**	**Register Lock Control Code (Write Only)** Some registers have write-protection function. Writing these registers have to disable the protected function by writing the sequence value “59h”, “16h”, “88h” to this field. After this sequence is completed, the REGLCTL bit will be set to 1 and write-protection registers can be normal write. **REGLCTL[0]** **Register Lock Control Disable Index (Read Only)** 0 = Write-protection Enabled for writing protected registers. Any write to the protected register is ignored. 1 = Write-protection Disabled for writing protected registers.

HIRC 48M Trim Value Register (SYS_RCADJ48M)

Register	Offset	R/W	Description	Reset Value
SYS_RCADJ48M	SYS_BA+0x10C	R/W	HIRC 48M Trim Value Register	0x0000_0XXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							RCADJ
7	6	5	4	3	2	1	0
RCADJ							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**RCADJ**	**RC48M Trim Value (Write Protect)** This field reflects the RC48M trim value. Software can update RC48m trim value by writing this field. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Low Speed Internal Oscillator Trim Code Register (SYS_LIRCT)

Register	Offset	R/W	Description	Reset Value
SYS_LIRCT	SYS_BA+0x138	R/W	Low Speed Internal Oscillator Trim Code Register	0x0003_0XXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						LIRCTTRIM	
15	14	13	12	11	10	9	8
Reserved							LIRCTRIM
7	6	5	4	3	2	1	0
LIRCTRIM							

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**LIRCTTRIM**	**Low Speed Internal Oscillator Temperature Trim Code Value** LIRC temperature trim Defult is 2b11
[15:9]	**Reserved**	Reserved.
[8:0]	**LIRCTRIM**	**Low Speed Internal Oscillator Trim Code Value** This field reflects factory low speed internal oscillator adjust code. (for internal use only)

MEGPM LDO Control Register (SYS_PMLDOCTL)

Register	Offset	R/W	Description	Reset Value
SYS_PMLDOCTL	SYS_BA+0x168	R/W	MEGPM LDO Control Register	0xX001_0107

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		LDO_OPCS	Reserved				
7	6	5	4	3	2	1	0
Reserved						LDO_CS	

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**LDO_OPCS**	**LDO OP current option** For LDO IQ current better setting during CHIP in NPDx/SPDx power down mode, Setting LDO_CS = 10 and LDO_OPCS = 0 before enter SPDx power down mode. Setting LDO_CS = 11 and LDO_OPCS = 1 before enter NPDx power down mode. **Note: R**ead after write to keep other bits value in SYS_PMLDOCTL. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note:** This field only reset by POR reset.
[12:2]	**Reserved**	Reserved.
[1:0]	**LDO_CS**	**LDO bias current option,** For LDO IQ current better setting during CHIP in NPDx/SPDx power down mode, Setting LDO_CS = 10 and LDO_OPCS = 0 before enter SPDx power down mode. Setting LDO_CS = 11 and LDO_OPCS = 1 before enter NPDx power down mode. **Note: R**ead after write to keep other bits value in SYS_PMLDOCTL. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note:** This field only reset by POR reset.

HIRC Test Mode Control Register (SYS_HIRCTMCTL)

Register	Offset	R/W	Description	Reset Value
SYS_HIRCTMCTL	SYS_BA+0x188	R/W	HIRC Test Mode Control Register	0x0XXX_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	OLDO_VOLTAGE			FTRIMOS		ICOMP	
7	6	5	4	3	2	1	0
CTAT				PTAT			

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**OLDO_TRIM**	**HIRC OLDO Voltage Trim Value** This bit field is used to adjust HIRC LDO voltage Default value is 100 **Note:** After chip power-on, this bit is defined by ROMMAP22[14:12]
[11:10]	**FTRIMOS**	**HIRC Frequency Trim Offset Selection** This bit field is used to select HIRC frequency center trim offset level. Default value is 11 Frequency Shift up : 01 Frequency Shift down : 10 **Note:** After chip power-on, this bit is defined by ROMMAP22[9:8]
[9:8]	**ICOMP**	**HIRC Compare Current Selection** This bit field is used to select HIRC compare current. Default value is 01 **Note:** After chip power-on, this bit is defined by ROMMAP22[9:8]
[7:4]	**CTAT**	**HIRC Nagtive T Temperature Parameter Control** This bit field is used to control negaitive tc for HIRC frequency adjusting Default value is 1111 **Note:** After chip power-on, this bit is defined by ROMMAP22[7:4]
[3:0]	**PTAT**	**HIRC Postive Temperature Parameter Control** This bit field is used to control positive tc for HIRC frequency adjusting Default value is 1111 **Note:** After chip power-on, this bit is defined by ROMMAP22[3:0]

Analog POR Disable Control Register (SYS_PORDISAN)

Register	Offset	R/W	Description	Reset Value
SYS_PORDISAN	SYS_BA+0x1EC	R/W	Analog POR Disable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
POROFFAN							
7	6	5	4	3	2	1	0
POROFFAN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**POROFFAN**	**Power-on Reset Enable Bit (Write Protect)** After powered on, User can turn off internal analog POR circuit to save power by writing 0x5AA5 to this field. The analog POR circuit will be active again when this field is set to another value or chip is reset by other reset source, including: nRESET, Watchdog, LVR reset, BOD reset, ICE reset command and the software-chip reset function. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Chip Series Version Register (SYS_CSERVER)

Register	Offset	R/W	Description	Reset Value
SYS_CSERVER	SYS_BA+0x1F4	R	Chip Series Version Register	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
VERSION							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**VERSION**	**Chip Series Version** These bits indicate the series version of chip. 00 = M2L31xxDAE. 01 = M2L31xx4AE. Others = Reserved.

Power Level Control Register (SYS_PLCTL)

Register	Offset	R/W	Description	Reset Value
SYS_PLCTL	SYS_BA+0x1F8	R/W	Power Level Control Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					PLSEL		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**PLSEL**	**Power Level Select (Write Protect)** These bits indicate the status of power level. 001 = Power level is PL1. Others = Reserved. **Note** : Write ignore when wtire reserved setting. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Power Level Status Register (SYS_PLSTS)

Register	Offset	R/W	Description	Reset Value
SYS_PLSTS	SYS_BA+0x1FC	R/W	Power Level Status Register	0x0000_0XXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					PLSTATUS		
7	6	5	4	3	2	1	0
Reserved							PLCBUSY

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**PLSTATUS**	**Power Level Status (Read Only)** This bit indicates the status of power level. 001 = Power level is PL1. Others = Reserved.
[7:1]	**Reserved**	Reserved.
[0]	**PLCBUSY**	**Power Level Change Busy Bit (Read Only)** This bit is set by hardware when power level is changing. After power level change is completed, this bit will be cleared automatically by hardware. 0 = Core voltage change is completed. 1 = Core voltage change is ongoing.

Initial VTOR Control Register (SYS_INIVTOR)

Register	Offset	R/W	Description	Reset Value
SYS_INIVTOR	SYS_BA+0x310	R/W	Initial VTOR Control Register	0x0000_0000

31	30	29	28	27	26	25	24
INIVTOR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
INIVTOR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
INIVTOR						**Reserved**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:10]	**INIVTOR**	**Initial VTOR Control Register** This is the register to set the address of vector table after CPU reseted or chip waked up from SPD0~2 mode. The value will be loaded to Vector Table Offset Register, which is at the address 0xE000ED08, when CPU reseted or chip wake up from SPD0~2 mode. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[9:0]	**Reserved**	Reserved.

GPIOA Multiple Function Control Register 0 (SYS_GPA_MFP0)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP0	SYS_BA+0x500	R/W	GPIOA Multiple Function Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PA3MFP				
23	22	21	20	19	18	17	16
Reserved			PA2MFP				
15	14	13	12	11	10	9	8
Reserved			PA1MFP				
7	6	5	4	3	2	1	0
Reserved			PA0MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PA3MFP**	**PA.3 Multi-function Pin Selection** 03 = QSPI0_SS 04 = SPI0_SS 05 = UART4_TXD 06 = TK_SE 07 = I2C0_SMBAL 08 = UART1_TXD 09 = I2C1_SCL 10 = PWM1_BRAKE1 11 = EQEI0_B 12 = PWM0_CH3 13 = EPWM0_CH2 14 = CLKO 15 = EPWM1_BRAKE1 16 = TK_TK5 17 = UTCPD0_VBSNKEN 20 = LPSPI0_SS
[23:21]	**Reserved**	Reserved.
[20:16]	**PA2MFP**	**PA.2 Multi-function Pin Selection** 03 = QSPI0_CLK 04 = SPI0_CLK 05 = UART4_RXD 07 = I2C0_SMBSUS 08 = UART1_RXD 09 = I2C1_SDA 11 = EQEI0_A 12 = PWM0_CH2

Bits	Description	
		13 = EPWM0_CH3 16 = TK_TK6 17 = UTCPD0_VBSRCEN 20 = LPSPI0_CLK
[15:13]	**Reserved**	Reserved.
[12:8]	**PA1MFP**	**PA.1 Multi-function Pin Selection** 03 = QSPI0_MISO0 04 = SPI0_MISO 07 = UART0_TXD 08 = UART1_nCTS 09 = I2C2_SCL 10 = CANFD1_TXD 11 = EQEI0_INDEX 12 = PWM0_CH1 13 = EPWM0_CH4 14 = ACMP2_O 15 = DAC1_ST 16 = TK_TK7 17 = UTCPD0_FRSTX1 18 = UTCPD0_DISCHG 20 = LPSPI0_MISO 21 = LPUART0_TXD 23 = LPIO1
[7:5]	**Reserved**	Reserved.
[4:0]	**PA0MFP**	**PA.0 Multi-function Pin Selection** 03 = QSPI0_MOSI0 04 = SPI0_MOSI 07 = UART0_RXD 08 = UART1_nRTS 09 = I2C2_SDA 10 = CANFD1_RXD 11 = EPWM0_BRAKE0 12 = PWM0_CH0 13 = EPWM0_CH5 14 = ACMP2_WLAT 15 = DAC0_ST 16 = TK_TK8 17 = UTCPD0_VCNEN1 20 = LPSPI0_MOSI 21 = LPUART0_RXD 23 = LPIO0

GPIOA Multiple Function Control Register 1 (SYS_GPA_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP1	SYS_BA+0x504	R/W	GPIOA Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PA7MFP				
23	22	21	20	19	18	17	16
Reserved			PA6MFP				
15	14	13	12	11	10	9	8
Reserved			PA5MFP				
7	6	5	4	3	2	1	0
Reserved			PA4MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PA7MFP**	**PA.7 Multi-function Pin Selection** 02 = EBI_AD7 04 = SPI1_CLK 07 = UART0_TXD 08 = I2C1_SCL 11 = EPWM1_CH4 12 = PWM1_CH2 13 = ACMP0_WLAT 14 = TM2 15 = INT1 16 = TK_TK0 17 = UTCPD0_VBSNKEN 21 = LPUART0_TXD 23 = LPIO5
[23:21]	**Reserved**	Reserved.
[20:16]	**PA6MFP**	**PA.6 Multi-function Pin Selection** 02 = EBI_AD6 04 = SPI1_SS 07 = UART0_RXD 08 = I2C1_SDA 11 = EPWM1_CH5 12 = PWM1_CH3 13 = ACMP1_WLAT 14 = TM3 15 = INT0 16 = TK_TK1 17 = UTCPD0_VBSRCEN

Bits	Description	
		21 = LPUART0_RXD 23 = LPIO4
[15:13]	**Reserved**	Reserved.
[12:8]	**PA5MFP**	**PA.5 Multi-function Pin Selection** 03 = QSPI0_MISO1 04 = SPI1_I2SMCLK 07 = UART0_nCTS 08 = UART0_TXD 09 = I2C0_SCL 10 = CANFD0_TXD 11 = UART5_TXD 12 = PWM0_CH5 13 = EPWM0_CH0 14 = EQEI0_INDEX 16 = TK_TK3 17 = UTCPD0_VBSNKEN 20 = LPUART0_TXD 21 = LPUART0_nCTS 22 = LPI2C0_SCL
[7:5]	**Reserved**	Reserved.
[4:0]	**PA4MFP**	**PA.4 Multi-function Pin Selection** 03 = QSPI0_MOSI1 04 = SPI0_I2SMCLK 07 = UART0_nRTS 08 = UART0_RXD 09 = I2C0_SDA 10 = CANFD0_RXD 11 = UART5_RXD 12 = PWM0_CH4 13 = EPWM0_CH1 14 = EQEI0_A 16 = TK_TK4 17 = UTCPD0_VBSRCEN 20 = LPUART0_RXD 21 = LPUART0_nRTS 22 = LPI2C0_SDA

GPIOA Multiple Function Control Register 2 (SYS_GPA_MFP2)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP2	SYS_BA+0x508	R/W	GPIOA Multiple Function Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PA11MFP				
23	22	21	20	19	18	17	16
Reserved			PA10MFP				
15	14	13	12	11	10	9	8
Reserved			PA9MFP				
7	6	5	4	3	2	1	0
Reserved			PA8MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PA11MFP**	**PA.11 Multi-function Pin Selection** 01 = EADC0_CH23, ACMP0_P0, OPA2_O 02 = EBI_nRD 05 = SPI3_SS 06 = USCI0_CLK 07 = I2C2_SCL 08 = UART6_TXD 09 = PWM0_CH0 10 = EPWM0_SYNC_OUT 13 = TM0_EXT 14 = DAC1_ST 23 = LPTM0_EXT
[23:21]	**Reserved**	Reserved.
[20:16]	**PA10MFP**	**PA.10 Multi-function Pin Selection** 01 = EADC0_CH22, ACMP1_P0, OPA1_O 02 = EBI_nWR 05 = SPI3_CLK 06 = USCI0_DAT0 07 = I2C2_SDA 08 = UART6_RXD 09 = PWM0_CH1 10 = EQEI1_INDEX 11 = ECAP0_IC0 13 = TM1_EXT 14 = DAC0_ST 23 = LPTM1_EXT

Bits	Description	
[15:13]	**Reserved**	Reserved.
[12:8]	**PA9MFP**	**PA.9 Multi-function Pin Selection** 01 = EADC0_CH21, ACMP2_P0, OPA1_N0 02 = EBI_MCLK 05 = SPI3_MISO 06 = USCI0_DAT1 07 = UART1_TXD 08 = UART7_TXD 09 = PWM0_CH2 10 = EQEI1_A 11 = ECAP0_IC1 13 = TM2_EXT 14 = I2C2_SMBAL
[7:5]	**Reserved**	Reserved.
[4:0]	**PA8MFP**	**PA.8 Multi-function Pin Selection** 01 = EADC0_CH20, OPA1_P0 02 = EBI_ALE 05 = SPI3_MOSI 06 = USCI0_CTL1 07 = UART1_RXD 08 = UART7_RXD 09 = PWM0_CH3 10 = EQEI1_B 11 = ECAP0_IC2 13 = TM3_EXT 14 = I2C2_SMBSUS 15 = INT4

GPIOA Multiple Function Control Register 3 (SYS_GPA_MFP3)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP3	SYS_BA+0x50C	R/W	GPIOA Multiple Function Control Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PA15MFP				
23	22	21	20	19	18	17	16
Reserved			PA14MFP				
15	14	13	12	11	10	9	8
Reserved			PA13MFP				
7	6	5	4	3	2	1	0
Reserved			PA12MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PA15MFP**	**PA.15 Multi-function Pin Selection** 03 = UART0_RXD 05 = SPI2_MOSI 06 = I2C2_SDA 08 = SPI0_MOSI 11 = PWM1_CH5 12 = EPWM0_SYNC_IN 14 = USB_OTG_ID 20 = LPSPI0_MOSI
[23:21]	**Reserved**	Reserved.
[20:16]	**PA14MFP**	**PA.14 Multi-function Pin Selection** 03 = UART0_TXD 04 = EBI_AD5 05 = SPI2_MISO 06 = I2C2_SCL 08 = SPI0_MISO 11 = PWM1_CH4 12 = EQEI1_B 13 = ECAP1_IC2 14 = USB_D+ 20 = LPSPI0_MISO
[15:13]	**Reserved**	Reserved.
[12:8]	**PA13MFP**	**PA.13 Multi-function Pin Selection** 03 = UART4_RXD 04 = I2C1_SDA 05 = SPI2_CLK

Bits	Description	
		06 = CANFD1_RXD 08 = SPI0_CLK 11 = PWM1_CH3 12 = EQEI1_A 13 = ECAP1_IC1 14 = USB_D- 20 = LPSPI0_CLK
[7:5]	**Reserved**	Reserved.
[4:0]	**PA12MFP**	**PA.12 Multi-function Pin Selection** 03 = UART4_TXD 04 = I2C1_SCL 05 = SPI2_SS 06 = CANFD1_TXD 08 = SPI0_SS 11 = PWM1_CH2 12 = EQEI1_INDEX 13 = ECAP1_IC0 14 = USB_VBUS 20 = LPSPI0_SS

GPIOB Multiple Function Control Register 0 (SYS_GPB_MFP0)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFP0	SYS_BA+0x510	R/W	GPIOB Multiple Function Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PB3MFP				
23	22	21	20	19	18	17	16
Reserved			PB2MFP				
15	14	13	12	11	10	9	8
Reserved			PB1MFP				
7	6	5	4	3	2	1	0
Reserved			PB0MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PB3MFP**	**PB.3 Multi-function Pin Selection** 01 = EADC0_CH3, LPADC0_CH3, ACMP0_N, OPA2_P0 02 = EBI_ADR2 04 = I2C1_SCL 05 = SPI1_CLK 06 = UART1_TXD 07 = UART5_nRTS 08 = USCI1_DAT1 11 = EPWM0_CH2 13 = PWM0_BRAKE0 14 = TM2 15 = INT2 23 = LPIO7
[23:21]	**Reserved**	Reserved.
[20:16]	**PB2MFP**	**PB.2 Multi-function Pin Selection** 01 = EADC0_CH2, LPADC0_CH2, ACMP0_P1, OPA0_O 02 = EBI_ADR3 04 = I2C1_SDA 05 = SPI1_SS 06 = UART1_RXD 07 = UART5_nCTS 08 = USCI1_DAT0 11 = EPWM0_CH3 14 = TM3 15 = INT3 23 = LPIO6

Bits	Description	
[15:13]	**Reserved**	Reserved.
[12:8]	**PB1MFP**	**PB.1 Multi-function Pin Selection** 01 = EADC0_CH1, LPADC0_CH1, ACMP2_N, OPA0_N0 02 = EBI_ADR8 07 = UART2_TXD 08 = USCI1_CLK 09 = I2C1_SCL 10 = QSPI0_MISO1 11 = EPWM0_CH4 12 = EPWM1_CH4 13 = EPWM0_BRAKE0 14 = PWM0_BRAKE0 17 = UTCPD0_VBDCHG 23 = LPIO3
[7:5]	**Reserved**	Reserved.
[4:0]	**PB0MFP**	**PB.0 Multi-function Pin Selection** 01 = EADC0_CH0, LPADC0_CH0, ACMP2_P1, OPA0_P0 02 = EBI_ADR9 05 = SPI3_I2SMCLK 06 = USCI0_CTL0 07 = UART2_RXD 08 = SPI0_I2SMCLK 09 = I2C1_SDA 10 = QSPI0_MOSI1 11 = EPWM0_CH5 12 = EPWM1_CH5 13 = EPWM0_BRAKE1 14 = PWM0_BRAKE1 17 = UTCPD0_VCNEN2 23 = LPIO2

GPIOB Multiple Function Control Register 1 (SYS_GPB_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFP1	SYS_BA+0x514	R/W	GPIOB Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PB7MFP				
23	22	21	20	19	18	17	16
Reserved			PB6MFP				
15	14	13	12	11	10	9	8
Reserved			PB5MFP				
7	6	5	4	3	2	1	0
Reserved			PB4MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PB7MFP**	**PB.7 Multi-function Pin Selection** 01 = EADC0_CH7, LPADC0_CH7, ACMP2_P3, OPA2_N1 02 = EBI_nWRL 04 = USCI1_DAT0 06 = UART1_TXD 08 = EBI_nCS0 10 = PWM1_CH4 11 = EPWM1_BRAKE0 12 = EPWM1_CH4 13 = INT5 14 = PWM1_BRAKE0 15 = ACMP0_O
[23:21]	**Reserved**	Reserved.
[20:16]	**PB6MFP**	**PB.6 Multi-function Pin Selection** 01 = EADC0_CH6, LPADC0_CH6, ACMP2_P2, OPA2_P1 02 = EBI_nWRH 04 = USCI1_DAT1 06 = UART1_RXD 08 = EBI_nCS1 10 = PWM1_CH5 11 = EPWM1_BRAKE1 12 = EPWM1_CH5 13 = INT4 14 = PWM1_BRAKE1 15 = ACMP1_O
[15:13]	**Reserved**	Reserved.

Bits	Description	
[12:8]	**PB5MFP**	**PB.5 Multi-function Pin Selection** 01 = EADC0_CH5, LPADC0_CH5, ACMP1_N, OPA1_P1 02 = EBI_ADR0 05 = SPI1_MISO 06 = I2C0_SCL 07 = UART5_TXD 08 = USCI1_CTL0 11 = EPWM0_CH0 13 = UART2_TXD 14 = TM0 15 = INT0 22 = LPI2C0_SCL 23 = LPTM0
[7:5]	**Reserved**	Reserved.
[4:0]	**PB4MFP**	**PB.4 Multi-function Pin Selection** 01 = EADC0_CH4, LPADC0_CH4, ACMP1_P1, OPA2_N0 02 = EBI_ADR1 05 = SPI1_MOSI 06 = I2C0_SDA 07 = UART5_RXD 08 = USCI1_CTL1 11 = EPWM0_CH1 13 = UART2_RXD 14 = TM1 15 = INT1 22 = LPI2C0_SDA 23 = LPTM1

GPIOB Multiple Function Control Register 2 (SYS_GPB_MFP2)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFP2	SYS_BA+0x518	R/W	GPIOB Multiple Function Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PB11MFP				
23	22	21	20	19	18	17	16
Reserved			PB10MFP				
15	14	13	12	11	10	9	8
Reserved			PB9MFP				
7	6	5	4	3	2	1	0
Reserved			PB8MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PB11MFP**	**PB.11 Multi-function Pin Selection** 01 = EADC0_CH11, LPADC0_CH11 02 = EBI_ADR16 05 = UART0_nCTS 06 = UART4_TXD 07 = I2C1_SCL 08 = CANFD1_TXD 09 = SPI0_I2SMCLK 10 = PWM1_CH0 21 = LPUART0_nCTS
[23:21]	**Reserved**	Reserved.
[20:16]	**PB10MFP**	**PB.10 Multi-function Pin Selection** 01 = EADC0_CH10, LPADC0_CH10 02 = EBI_ADR17 04 = USCI1_CTL0 05 = UART0_nRTS 06 = UART4_RXD 07 = I2C1_SDA 08 = CANFD1_RXD 10 = PWM1_CH1 21 = LPUART0_nRTS
[15:13]	**Reserved**	Reserved.
[12:8]	**PB9MFP**	**PB.9 Multi-function Pin Selection** 01 = EADC0_CH9, LPADC0_CH9 02 = EBI_ADR18 04 = USCI1_CTL1

Bits	Description	
		05 = UART0_TXD 06 = UART1_nCTS 08 = UART7_TXD 10 = PWM1_CH2 21 = LPUART0_TXD
[7:5]	**Reserved**	Reserved.
[4:0]	**PB8MFP**	**PB.8 Multi-function Pin Selection** 01 = EADC0_CH8, LPADC0_CH8 02 = EBI_ADR19 04 = USCI1_CLK 05 = UART0_RXD 06 = UART1_nRTS 08 = UART7_RXD 10 = PWM1_CH3 21 = LPUART0_RXD

GPIOB Multiple Function Control Register 3 (SYS_GPB_MFP3)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFP3	SYS_BA+0x51C	R/W	GPIOB Multiple Function Control Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PB15MFP				
23	22	21	20	19	18	17	16
Reserved			PB14MFP				
15	14	13	12	11	10	9	8
Reserved			PB13MFP				
7	6	5	4	3	2	1	0
Reserved			PB12MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PB15MFP**	**PB.15 Multi-function Pin Selection** 01 = EADC0_CH15, LPADC0_CH15, OPA0_P1 02 = EBI_AD12 04 = SPI0_SS 05 = USCI0_CTL1 06 = UART0_nCTS 07 = UART3_TXD 08 = I2C2_SMBAL 10 = EPWM0_BRAKE1 11 = EPWM1_CH0 13 = TM0_EXT 14 = USB_VBUS_EN 17 = UTCPD0_VBSNKEN 20 = LPSPI0_SS 21 = LPUART0_nCTS 23 = LPTM0_EXT
[23:21]	**Reserved**	Reserved.
[20:16]	**PB14MFP**	**PB.14 Multi-function Pin Selection** 01 = EADC0_CH14, LPADC0_CH14, OPA0_N1 02 = EBI_AD13 04 = SPI0_CLK 05 = USCI0_DAT1 06 = UART0_nRTS 07 = UART3_RXD 08 = I2C2_SMBSUS 10 = EQEI0_INDEX 11 = EPWM1_CH1

Bits	Description	
		12 = ECAP0_IC0 13 = TM1_EXT 14 = CLKO 16 = TK_SE 17 = UTCPD0_VBSRCEN 20 = LPSPI0_CLK 21 = LPUART0_nRTS 23 = LPTM1_EXT
[15:13]	**Reserved**	Reserved.
[12:8]	**PB13MFP**	**PB.13 Multi-function Pin Selection** 01 = EADC0_CH13, LPADC0_CH13, DAC1_OUT, ACMP0_P3, ACMP1_P3, OPA1_N1 02 = EBI_AD14 04 = SPI0_MISO 05 = USCI0_DAT0 06 = UART0_TXD 07 = UART3_nRTS 08 = I2C2_SCL 10 = CANFD0_TXD 11 = EPWM1_CH2 13 = TM2_EXT 20 = LPSPI0_MISO 21 = LPUART0_TXD
[7:5]	**Reserved**	Reserved.
[4:0]	**PB12MFP**	**PB.12 Multi-function Pin Selection** 01 = EADC0_CH12, LPADC0_CH12, DAC0_OUT, ACMP0_P2, ACMP1_P2 02 = EBI_AD15 04 = SPI0_MOSI 05 = USCI0_CLK 06 = UART0_RXD 07 = UART3_nCTS 08 = I2C2_SDA 10 = CANFD0_RXD 11 = EPWM1_CH3 13 = TM3_EXT 20 = LPSPI0_MOSI 21 = LPUART0_RXD

GPIOC Multiple Function Control Register 0 (SYS_GPC_MFP0)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFP0	SYS_BA+0x520	R/W	GPIOC Multiple Function Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PC3MFP				
23	22	21	20	19	18	17	16
Reserved			PC2MFP				
15	14	13	12	11	10	9	8
Reserved			PC1MFP				
7	6	5	4	3	2	1	0
Reserved			PC0MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PC3MFP**	**PC.3 Multi-function Pin Selection** 02 = EBI_AD3 04 = QSPI0_SS 06 = SPI3_MISO 07 = SPI1_MISO 08 = UART2_nRTS 09 = I2C0_SMBAL 11 = UART3_TXD 12 = EPWM1_CH2 15 = I2C3_SCL 16 = TK_TK12 17 = UTCPD0_CCDB2
[23:21]	**Reserved**	Reserved.
[20:16]	**PC2MFP**	**PC.2 Multi-function Pin Selection** 02 = EBI_AD2 04 = QSPI0_CLK 06 = SPI3_MOSI 07 = SPI1_MOSI 08 = UART2_nCTS 09 = I2C0_SMBSUS 10 = EQEI0_INDEX 11 = UART3_RXD 12 = EPWM1_CH3 13 = ECAP0_IC0 15 = I2C3_SDA 16 = TK_TK13 17 = UTCPD0_CCDB1

Bits	Description	
[15:13]	**Reserved**	Reserved.
[12:8]	**PC1MFP**	**PC.1 Multi-function Pin Selection** 02 = EBI_AD1 04 = QSPI0_MISO0 07 = SPI1_CLK 08 = UART2_TXD 09 = I2C0_SCL 10 = EQEI0_A 12 = EPWM1_CH4 13 = ECAP0_IC1 14 = ACMP0_O 15 = EADC0_ST 17 = UTCPD0_CC2 22 = LPI2C0_SCL 23 = LPIO5
[7:5]	**Reserved**	Reserved.
[4:0]	**PC0MFP**	**PC.0 Multi-function Pin Selection** 02 = EBI_AD0 04 = QSPI0_MOSI0 07 = SPI1_SS 08 = UART2_RXD 09 = I2C0_SDA 10 = EQEI0_B 12 = EPWM1_CH5 13 = ECAP0_IC2 14 = ACMP1_O 17 = UTCPD0_CC1 22 = LPI2C0_SDA 23 = LPIO4

GPIOC Multiple Function Control Register 1 (SYS_GPC_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFP1	SYS_BA+0x524	R/W	GPIOC Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PC7MFP				
23	22	21	20	19	18	17	16
Reserved			PC6MFP				
15	14	13	12	11	10	9	8
Reserved			PC5MFP				
7	6	5	4	3	2	1	0
Reserved			PC4MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PC7MFP**	**PC.7 Multi-function Pin Selection** 02 = EBI_AD9 04 = SPI1_MISO 05 = UART4_TXD 07 = UART0_nCTS 09 = UART6_TXD 11 = EPWM1_CH2 12 = PWM1_CH0 14 = TM0 15 = INT3 21 = LPUART0_nCTS 23 = LPTM0
[23:21]	**Reserved**	Reserved.
[20:16]	**PC6MFP**	**PC.6 Multi-function Pin Selection** 02 = EBI_AD8 04 = SPI1_MOSI 05 = UART4_RXD 07 = UART0_nRTS 09 = UART6_RXD 11 = EPWM1_CH3 12 = PWM1_CH1 14 = TM1 15 = INT2 21 = LPUART0_nRTS 23 = LPTM1
[15:13]	**Reserved**	Reserved.

Bits	Description	
[12:8]	**PC5MFP**	**PC.5 Multi-function Pin Selection** 02 = EBI_AD5 04 = QSPI0_MISO1 06 = SPI3_SS 08 = UART2_TXD 09 = I2C1_SCL 10 = CANFD0_TXD 11 = UART4_TXD 12 = EPWM1_CH0 15 = I2C3_SMBAL 16 = TK_TK10 17 = UTCPD0_FRSTX2 18 = UTCPD0_DISCHG
[7:5]	**Reserved**	Reserved.
[4:0]	**PC4MFP**	**PC.4 Multi-function Pin Selection** 02 = EBI_AD4 04 = QSPI0_MOSI1 06 = SPI3_CLK 07 = SPI1_I2SMCLK 08 = UART2_RXD 09 = I2C1_SDA 10 = CANFD0_RXD 11 = UART4_RXD 12 = EPWM1_CH1 15 = I2C3_SMBSUS 16 = TK_TK11 17 = UTCPD0_FRSTX1 18 = UTCPD0_DISCHG

GPIOC Multiple Function Control Register 2 (SYS_GPC_MFP2)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFP2	SYS_BA+0x528	R/W	GPIOC Multiple Function Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PC11MFP				
23	22	21	20	19	18	17	16
Reserved			PC10MFP				
15	14	13	12	11	10	9	8
Reserved			PC9MFP				
7	6	5	4	3	2	1	0
Reserved			PC8MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PC11MFP**	**PC.11 Multi-function Pin Selection** 02 = EBI_ADR5 03 = UART0_RXD 04 = I2C0_SDA 05 = UART6_RXD 06 = SPI3_MOSI 11 = ECAP1_IC1 12 = EPWM1_CH1 14 = ACMP1_O 21 = LPUART0_RXD 22 = LPI2C0_SDA
[23:21]	**Reserved**	Reserved.
[20:16]	**PC10MFP**	**PC.10 Multi-function Pin Selection** 02 = EBI_ADR6 05 = UART6_nRTS 06 = SPI3_CLK 07 = UART3_TXD 09 = CANFD1_TXD 11 = ECAP1_IC0 12 = EPWM1_CH2
[15:13]	**Reserved**	Reserved.
[12:8]	**PC9MFP**	**PC.9 Multi-function Pin Selection** 02 = EBI_ADR7 05 = UART6_nCTS 06 = SPI3_SS 07 = UART3_RXD

Bits	Description	
		09 = CANFD1_RXD 12 = EPWM1_CH3
[7:5]	**Reserved**	Reserved.
[4:0]	**PC8MFP**	**PC.8 Multi-function Pin Selection** 02 = EBI_ADR16 04 = I2C0_SDA 05 = UART4_nCTS 08 = UART1_RXD 11 = EPWM1_CH1 12 = PWM1_CH4 22 = LPI2C0_SDA

GPIOC Multiple Function Control Register 3 (SYS_GPC_MFP3)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFP3	SYS_BA+0x52C	R/W	GPIOC Multiple Function Control Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			PC14MFP				
15	14	13	12	11	10	9	8
Reserved			PC13MFP				
7	6	5	4	3	2	1	0
Reserved			PC12MFP				

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20:16]	**PC14MFP**	**PC.14 Multi-function Pin Selection** 02 = EBI_AD11 04 = SPI0_I2SMCLK 05 = USCI0_CTL0 06 = QSPI0_CLK 08 = EBI_nCS2 11 = EPWM0_SYNC_IN 13 = TM1 14 = USB_VBUS_ST 15 = ACMP2_O 23 = LPTM1
[15:13]	**Reserved**	Reserved.
[12:8]	**PC13MFP**	**PC.13 Multi-function Pin Selection** 01 = EADC0_CH19 02 = EBI_ADR10 04 = SPI2_I2SMCLK 05 = CANFD1_TXD 06 = USCI0_CTL0 07 = UART2_TXD 09 = PWM0_CH4 13 = CLKO 14 = EADC0_ST 16 = TK_SE 23 = LPADC0_ST
[7:5]	**Reserved**	Reserved.
[4:0]	**PC12MFP**	**PC.12 Multi-function Pin Selection**

Bits	Description	
		02 = EBI_ADR4 03 = UART0_TXD 04 = I2C0_SCL 05 = UART6_TXD 06 = SPI3_MISO 11 = ECAP1_IC2 12 = EPWM1_CH0 14 = ACMP0_O 21 = LPUART0_TXD 22 = LPI2C0_SCL

GPIOD Multiple Function Control Register 0 (SYS_GPD_MFP0)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFP0	SYS_BA+0x530	R/W	GPIOD Multiple Function Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PD3MFP				
23	22	21	20	19	18	17	16
Reserved			PD2MFP				
15	14	13	12	11	10	9	8
Reserved			PD1MFP				
7	6	5	4	3	2	1	0
Reserved			PD0MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PD3MFP**	**PD.3 Multi-function Pin Selection** 02 = EBI_AD10 03 = USCI0_CTL1 04 = SPI0_SS 05 = UART3_nRTS 06 = USCI1_CTL0 09 = UART0_TXD 16 = TK_TK14 20 = LPSPI0_SS 21 = LPUART0_TXD
[23:21]	**Reserved**	Reserved.
[20:16]	**PD2MFP**	**PD.2 Multi-function Pin Selection** 02 = EBI_AD11 03 = USCI0_DAT1 04 = SPI0_CLK 05 = UART3_nCTS 09 = UART0_RXD 16 = TK_TK15 20 = LPSPI0_CLK 21 = LPUART0_RXD
[15:13]	**Reserved**	Reserved.
[12:8]	**PD1MFP**	**PD.1 Multi-function Pin Selection** 02 = EBI_AD12 03 = USCI0_DAT0 04 = SPI0_MISO 05 = UART3_TXD

Bits	Description	
		16 = TK_TK16 20 = LPSPI0_MISO 23 = LPIO7
[7:5]	**Reserved**	Reserved.
[4:0]	**PD0MFP**	**PD.0 Multi-function Pin Selection** 02 = EBI_AD13 03 = USCI0_CLK 04 = SPI0_MOSI 05 = UART3_RXD 14 = TM2 16 = TK_TK17 20 = LPSPI0_MOSI 23 = LPIO6

GPIOD Multiple Function Control Register 1 (SYS_GPD_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFP1	SYS_BA+0x534	R/W	GPIOD Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PD7MFP				
23	22	21	20	19	18	17	16
Reserved			PD6MFP				
15	14	13	12	11	10	9	8
Reserved			PD5MFP				
7	6	5	4	3	2	1	0
Reserved			PD4MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PD7MFP**	**PD.7 Multi-function Pin Selection** 03 = UART1_TXD 04 = I2C0_SCL 06 = USCI1_CLK 16 = TK_TK14 22 = LPI2C0_SCL
[23:21]	**Reserved**	Reserved.
[20:16]	**PD6MFP**	**PD.6 Multi-function Pin Selection** 03 = UART1_RXD 04 = I2C0_SDA 06 = USCI1_DAT1 16 = TK_TK15 22 = LPI2C0_SDA
[15:13]	**Reserved**	Reserved.
[12:8]	**PD5MFP**	**PD.5 Multi-function Pin Selection** 04 = I2C1_SCL 06 = USCI1_DAT0 16 = TK_TK16
[7:5]	**Reserved**	Reserved.
[4:0]	**PD4MFP**	**PD.4 Multi-function Pin Selection** 03 = USCI0_CTL0 04 = I2C1_SDA 06 = USCI1_CTL1 16 = TK_TK17

GPIOD Multiple Function Control Register 2 (SYS_GPD_MFP2)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFP2	SYS_BA+0x538	R/W	GPIOD Multiple Function Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PD11MFP				
23	22	21	20	19	18	17	16
Reserved			PD10MFP				
15	14	13	12	11	10	9	8
Reserved			PD9MFP				
7	6	5	4	3	2	1	0
Reserved			PD8MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PD11MFP**	**PD.11 Multi-function Pin Selection** 01 = EADC0_CH17 02 = EBI_nCS1 03 = UART1_TXD 04 = CANFD0_TXD 10 = EQEI0_A 15 = INT6
[23:21]	**Reserved**	Reserved.
[20:16]	**PD10MFP**	**PD.10 Multi-function Pin Selection** 01 = EADC0_CH16 02 = EBI_nCS2 03 = UART1_RXD 04 = CANFD0_RXD 10 = EQEI0_B 15 = INT7
[15:13]	**Reserved**	Reserved.
[12:8]	**PD9MFP**	**PD.9 Multi-function Pin Selection** 02 = EBI_AD7 03 = I2C2_SCL 04 = UART2_nCTS 05 = UART7_TXD
[7:5]	**Reserved**	Reserved.
[4:0]	**PD8MFP**	**PD.8 Multi-function Pin Selection** 02 = EBI_AD6 03 = I2C2_SDA

Bits	Description	
		04 = UART2_nRTS 05 = UART7_RXD

GPIOD Multiple Function Control Register 3 (SYS_GPD_MFP3)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFP3	SYS_BA+0x53C	R/W	GPIOD Multiple Function Control Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PD15MFP				
23	22	21	20	19	18	17	16
Reserved			PD14MFP				
15	14	13	12	11	10	9	8
Reserved			PD13MFP				
7	6	5	4	3	2	1	0
Reserved			PD12MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PD15MFP**	**PD.15 Multi-function Pin Selection** 12 = EPWM0_CH5 13 = ACMP2_WLAT 14 = TM3 15 = INT1 16 = TK_TK2 17 = UTCPD0_FRSTX2 18 = UTCPD0_DISCHG
[23:21]	**Reserved**	Reserved.
[20:16]	**PD14MFP**	**PD.14 Multi-function Pin Selection** 02 = EBI_nCS0 03 = SPI3_I2SMCLK 05 = SPI0_I2SMCLK 11 = EPWM0_CH4
[15:13]	**Reserved**	Reserved.
[12:8]	**PD13MFP**	**PD.13 Multi-function Pin Selection** 02 = EBI_AD10 04 = SPI0_I2SMCLK 05 = SPI1_I2SMCLK 11 = PWM0_CH0 14 = CLKO 15 = EADC0_ST 16 = TK_SE 23 = LPADC0_ST
[7:5]	**Reserved**	Reserved.

Bits	Description	
[4:0]	**PD12MFP**	**PD.12 Multi-function Pin Selection** 01 = EADC0_CH18 02 = EBI_nCS0 05 = CANFD1_RXD 07 = UART2_RXD 09 = PWM0_CH5 10 = EQEI0_INDEX 13 = CLKO 14 = EADC0_ST 15 = INT5 16 = TK_SE 23 = LPADC0_ST

GPIOE Multiple Function Control Register 0 (SYS_GPE_MFP0)

Register	Offset	R/W	Description	Reset Value
SYS_GPE_MFP0	SYS_BA+0x540	R/W	GPIOE Multiple Function Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PE3MFP				
23	22	21	20	19	18	17	16
Reserved			PE2MFP				
15	14	13	12	11	10	9	8
Reserved			PE1MFP				
7	6	5	4	3	2	1	0
Reserved			PE0MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PE3MFP**	**PE.3 Multi-function Pin Selection** 02 = EBI_MCLK 05 = SPI3_MISO 07 = USCI0_DAT0 08 = UART6_nRTS 09 = UART7_TXD 11 = EQEI0_A 12 = EPWM0_CH4 13 = PWM0_CH1
[23:21]	**Reserved**	Reserved.
[20:16]	**PE2MFP**	**PE.2 Multi-function Pin Selection** 02 = EBI_ALE 05 = SPI3_MOSI 07 = USCI0_CLK 08 = UART6_nCTS 09 = UART7_RXD 11 = EQEI0_B 12 = EPWM0_CH5 13 = PWM0_CH0
[15:13]	**Reserved**	Reserved.
[12:8]	**PE1MFP**	**PE.1 Multi-function Pin Selection** 02 = EBI_AD10 03 = QSPI0_MISO0 06 = SPI1_MISO 07 = UART3_TXD 08 = I2C1_SCL

Bits	Description	
		09 = UART4_nCTS 23 = LPIO1
[7:5]	**Reserved**	Reserved.
[4:0]	**PE0MFP**	**PE.0 Multi-function Pin Selection** 02 = EBI_AD11 03 = QSPI0_MOSI0 06 = SPI1_MOSI 07 = UART3_RXD 08 = I2C1_SDA 09 = UART4_nRTS 23 = LPIO0

GPIOE Multiple Function Control Register 1 (SYS_GPE_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPE_MFP1	SYS_BA+0x544	R/W	GPIOE Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PE7MFP				
23	22	21	20	19	18	17	16
Reserved			PE6MFP				
15	14	13	12	11	10	9	8
Reserved			PE5MFP				
7	6	5	4	3	2	1	0
Reserved			PE4MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PE7MFP**	**PE.7 Multi-function Pin Selection** 08 = UART5_TXD 09 = CANFD1_TXD 11 = EQEI1_INDEX 12 = EPWM0_CH0 13 = PWM0_CH5
[23:21]	**Reserved**	Reserved.
[20:16]	**PE6MFP**	**PE.6 Multi-function Pin Selection** 05 = SPI3_I2SMCLK 07 = USCI0_CTL0 08 = UART5_RXD 09 = CANFD1_RXD 11 = EQEI1_A 12 = EPWM0_CH1 13 = PWM0_CH4
[15:13]	**Reserved**	Reserved.
[12:8]	**PE5MFP**	**PE.5 Multi-function Pin Selection** 02 = EBI_nRD 05 = SPI3_SS 07 = USCI0_CTL1 08 = UART6_TXD 09 = UART7_nRTS 11 = EQEI1_B 12 = EPWM0_CH2 13 = PWM0_CH3

Bits	Description	
[7:5]	**Reserved**	Reserved.
[4:0]	**PE4MFP**	**PE.4 Multi-function Pin Selection** 02 = EBI_nWR 05 = SPI3_CLK 07 = USCI0_DAT1 08 = UART6_RXD 09 = UART7_nCTS 11 = EQEI0_INDEX 12 = EPWM0_CH3 13 = PWM0_CH2

GPIOE Multiple Function Control Register 2 (SYS_GPE_MFP2)

Register	Offset	R/W	Description	Reset Value
SYS_GPE_MFP2	SYS_BA+0x548	R/W	GPIOE Multiple Function Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PE11MFP				
23	22	21	20	19	18	17	16
Reserved			PE10MFP				
15	14	13	12	11	10	9	8
Reserved			PE9MFP				
7	6	5	4	3	2	1	0
Reserved			PE8MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PE11MFP**	**PE.11 Multi-function Pin Selection** 02 = EBI_ADR13 05 = SPI2_SS 06 = USCI1_DAT1 07 = UART3_RXD 08 = UART1_nCTS 09 = PWM1_BRAKE1 10 = EPWM0_CH3 11 = EPWM1_BRAKE1 13 = ECAP1_IC2
[23:21]	**Reserved**	Reserved.
[20:16]	**PE10MFP**	**PE.10 Multi-function Pin Selection** 02 = EBI_ADR12 05 = SPI2_MOSI 06 = USCI1_DAT0 07 = UART3_TXD 09 = PWM1_BRAKE0 10 = EPWM0_CH2 11 = EPWM1_BRAKE0 12 = ECAP0_IC2
[15:13]	**Reserved**	Reserved.
[12:8]	**PE9MFP**	**PE.9 Multi-function Pin Selection** 02 = EBI_ADR11 05 = SPI2_MISO 06 = USCI1_CTL0 07 = UART2_RXD

Bits	Description	
		09 = PWM0_BRAKE1 10 = EPWM0_CH1 11 = EPWM0_BRAKE1 12 = ECAP0_IC1
[7:5]	**Reserved**	Reserved.
[4:0]	**PE8MFP**	**PE.8 Multi-function Pin Selection** 02 = EBI_ADR10 05 = SPI2_CLK 06 = USCI1_CTL1 07 = UART2_TXD 09 = PWM0_BRAKE0 10 = EPWM0_CH0 11 = EPWM0_BRAKE0 12 = ECAP0_IC0

GPIOE Multiple Function Control Register 3 (SYS_GPE_MFP3)

Register	Offset	R/W	Description	Reset Value
SYS_GPE_MFP3	SYS_BA+0x54C	R/W	GPIOE Multiple Function Control Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PE15MFP				
23	22	21	20	19	18	17	16
Reserved			PE14MFP				
15	14	13	12	11	10	9	8
Reserved			PE13MFP				
7	6	5	4	3	2	1	0
Reserved			PE12MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PE15MFP**	**PE.15 Multi-function Pin Selection** 02 = EBI_AD9 03 = UART2_RXD 04 = CANFD0_RXD 06 = UART6_RXD
[23:21]	**Reserved**	Reserved.
[20:16]	**PE14MFP**	**PE.14 Multi-function Pin Selection** 02 = EBI_AD8 03 = UART2_TXD 04 = CANFD0_TXD 06 = UART6_TXD 12 = EPWM0_CH1 13 = TM2 14 = CLKO 15 = INT4 16 = TK_TK9
[15:13]	**Reserved**	Reserved.
[12:8]	**PE13MFP**	**PE.13 Multi-function Pin Selection** 02 = EBI_ADR15 04 = I2C0_SCL 05 = UART4_nRTS 08 = UART1_TXD 10 = EPWM0_CH5 11 = EPWM1_CH0 12 = PWM1_CH5 13 = ECAP1_IC0

Bits	Description	
		22 = LPI2C0_SCL
[7:5]	**Reserved**	Reserved.
[4:0]	**PE12MFP**	**PE.12 Multi-function Pin Selection** 02 = EBI_ADR14 05 = SPI2_I2SMCLK 06 = USCI1_CLK 08 = UART1_nRTS 10 = EPWM0_CH4 13 = ECAP1_IC1

GPIOF Multiple Function Control Register 0 (SYS_GPF_MFP0)

Register	Offset	R/W	Description	Reset Value
SYS_GPF_MFP0	SYS_BA+0x550	R/W	GPIOF Multiple Function Control Register 0	0x0000_0E0E

31	30	29	28	27	26	25	24
Reserved			PF3MFP				
23	22	21	20	19	18	17	16
Reserved			PF2MFP				
15	14	13	12	11	10	9	8
Reserved			PF1MFP				
7	6	5	4	3	2	1	0
Reserved			PF0MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PF3MFP**	**PF.3 Multi-function Pin Selection** 02 = EBI_nCS0 03 = UART0_TXD 04 = I2C0_SCL 10 = XT1_IN 11 = PWM1_CH0 12 = EQEI1_A 13 = ECAP1_IC1 21 = LPUART0_TXD 22 = LPI2C0_SCL
[23:21]	**Reserved**	Reserved.
[20:16]	**PF2MFP**	**PF.2 Multi-function Pin Selection** 02 = EBI_nCS1 03 = UART0_RXD 04 = I2C0_SDA 05 = QSPI0_CLK 10 = XT1_OUT 11 = PWM1_CH1 12 = EQEI1_B 13 = ECAP1_IC2 21 = LPUART0_RXD 22 = LPI2C0_SDA
[15:13]	**Reserved**	Reserved.
[12:8]	**PF1MFP**	**PF.1 Multi-function Pin Selection** 02 = UART1_RXD 03 = I2C1_SDA

Bits	Description	
		04 = UART0_RXD 06 = SPI3_I2SMCLK 11 = EPWM1_CH5 12 = PWM1_CH1 14 = ICE_CLK 17 = UTCPD0_FRSTX1 18 = UTCPD0_DISCHG 21 = LPUART0_RXD 23 = LPIO3
[7:5]	**Reserved**	Reserved.
[4:0]	**PF0MFP**	**PF.0 Multi-function Pin Selection** 02 = UART1_TXD 03 = I2C1_SCL 04 = UART0_TXD 11 = EPWM1_CH4 12 = PWM1_CH0 14 = ICE_DAT 17 = UTCPD0_FRSTX2 18 = UTCPD0_DISCHG 21 = LPUART0_TXD 23 = LPIO2

GPIOF Multiple Function Control Register 1 (SYS_GPF_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPF_MFP1	SYS_BA+0x554	R/W	GPIOF Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PF7MFP				
23	22	21	20	19	18	17	16
Reserved			PF6MFP				
15	14	13	12	11	10	9	8
Reserved			PF5MFP				
7	6	5	4	3	2	1	0
Reserved			PF4MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PF7MFP**	**PF.7 Multi-function Pin Selection** 02 = EBI_ADR18 05 = SPI0_MISO 06 = UART4_TXD 10 = TAMPER1 14 = TM3 15 = INT5 20 = LPSPI0_MISO
[23:21]	**Reserved**	Reserved.
[20:16]	**PF6MFP**	**PF.6 Multi-function Pin Selection** 02 = EBI_ADR19 05 = SPI0_MOSI 06 = UART4_RXD 07 = EBI_nCS0 09 = EPWM1_BRAKE0 10 = TAMPER0 11 = EPWM0_BRAKE0 12 = EPWM0_CH4 13 = PWM1_BRAKE0 14 = PWM0_BRAKE0 15 = CLKO 20 = LPSPI0_MOSI
[15:13]	**Reserved**	Reserved.
[12:8]	**PF5MFP**	**PF.5 Multi-function Pin Selection** 02 = UART2_RXD 04 = UART2_nCTS

Bits	Description	
		07 = EPWM0_CH0 08 = PWM0_CH4 09 = EPWM0_SYNC_OUT 10 = X32_IN 11 = EADC0_ST 17 = UTCPD0_VBSNKEN
[7:5]	**Reserved**	Reserved.
[4:0]	**PF4MFP**	**PF.4 Multi-function Pin Selection** 02 = UART2_TXD 04 = UART2_nRTS 07 = EPWM0_CH1 08 = PWM0_CH5 10 = X32_OUT 12 = EQEI1_INDEX 13 = ECAP1_IC0 17 = UTCPD0_VBSRCEN

GPIOF Multiple Function Control Register 2 (SYS_GPF_MFP2)

Register	Offset	R/W	Description	Reset Value
SYS_GPF_MFP2	SYS_BA+0x558	R/W	GPIOF Multiple Function Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PF11MFP				
23	22	21	20	19	18	17	16
Reserved			PF10MFP				
15	14	13	12	11	10	9	8
Reserved			PF9MFP				
7	6	5	4	3	2	1	0
Reserved			PF8MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PF11MFP**	**PF.11 Multi-function Pin Selection** 02 = EBI_ADR14 03 = SPI2_MOSI 06 = UART5_TXD 13 = TM3
[23:21]	**Reserved**	Reserved.
[20:16]	**PF10MFP**	**PF.10 Multi-function Pin Selection** 02 = EBI_ADR15 05 = SPI0_I2SMCLK 06 = UART5_RXD
[15:13]	**Reserved**	Reserved.
[12:8]	**PF9MFP**	**PF.9 Multi-function Pin Selection** 02 = EBI_ADR16 05 = SPI0_SS 06 = UART5_nRTS 08 = CANFD1_TXD 20 = LPSPI0_SS
[7:5]	**Reserved**	Reserved.
[4:0]	**PF8MFP**	**PF.8 Multi-function Pin Selection** 02 = EBI_ADR17 05 = SPI0_CLK 06 = UART5_nCTS 08 = CANFD1_RXD 10 = TAMPER2 20 = LPSPI0_CLK

GPIOG Multiple Function Control Register 0 (SYS_GPG_MFP0)

Register	Offset	R/W	Description	Reset Value
SYS_GPG_MFP0	SYS_BA+0x560	R/W	GPIOG Multiple Function Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PG3MFP				
23	22	21	20	19	18	17	16
Reserved			PG2MFP				
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PG3MFP**	**PG.3 Multi-function Pin Selection** 02 = EBI_ADR12 03 = SPI2_CLK 04 = I2C0_SMBSUS 05 = I2C1_SDA 09 = I2C3_SMBSUS 13 = TM1 23 = LPTM1
[23:21]	**Reserved**	Reserved.
[20:16]	**PG2MFP**	**PG.2 Multi-function Pin Selection** 02 = EBI_ADR11 03 = SPI2_SS 04 = I2C0_SMBAL 05 = I2C1_SCL 09 = I2C3_SMBAL 13 = TM0 23 = LPTM0
[15:0]	**Reserved**	Reserved.

GPIOG Multiple Function Control Register 1 (SYS_GPG_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPG_MFP1	SYS_BA+0x564	R/W	GPIOG Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			PG4MFP				

Bits	Description	
[31:5]	Reserved	Reserved.
[4:0]	PG4MFP	**PG.4 Multi-function Pin Selection** 02 = EBI_ADR13 03 = SPI2_MISO 13 = TM2

GPIOG Multiple Function Control Register 2 (SYS_GPG_MFP2)

Register	Offset	R/W	Description	Reset Value
SYS_GPG_MFP2	SYS_BA+0x568	R/W	GPIOG Multiple Function Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PG11MFP				
23	22	21	20	19	18	17	16
Reserved			PG10MFP				
15	14	13	12	11	10	9	8
Reserved			PG9MFP				
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PG11MFP**	**PG.11 Multi-function Pin Selection** 02 = EBI_AD2 06 = UART7_TXD 12 = PWM0_CH3
[23:21]	**Reserved**	Reserved.
[20:16]	**PG10MFP**	**PG.10 Multi-function Pin Selection** 02 = EBI_AD1 12 = PWM0_CH4
[15:13]	**Reserved**	Reserved.
[12:8]	**PG9MFP**	**PG.9 Multi-function Pin Selection** 02 = EBI_AD0 12 = PWM0_CH5
[7:0]	**Reserved**	Reserved.

GPIOG Multiple Function Control Register 3 (SYS_GPG_MFP3)

Register	Offset	R/W	Description	Reset Value
SYS_GPG_MFP3	SYS_BA+0x56C	R/W	GPIOG Multiple Function Control Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PG15MFP				
23	22	21	20	19	18	17	16
Reserved			PG14MFP				
15	14	13	12	11	10	9	8
Reserved			PG13MFP				
7	6	5	4	3	2	1	0
Reserved			PG12MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PG15MFP**	**PG.15 Multi-function Pin Selection** 14 = CLKO 15 = EADC0_ST 16 = TK_SE 23 = LPADC0_ST
[23:21]	**Reserved**	Reserved.
[20:16]	**PG14MFP**	**PG.14 Multi-function Pin Selection** 02 = EBI_AD5 06 = UART6_RXD 12 = PWM0_CH0
[15:13]	**Reserved**	Reserved.
[12:8]	**PG13MFP**	**PG.13 Multi-function Pin Selection** 02 = EBI_AD4 06 = UART6_TXD 12 = PWM0_CH1
[7:5]	**Reserved**	Reserved.
[4:0]	**PG12MFP**	**PG.12 Multi-function Pin Selection** 02 = EBI_AD3 06 = UART7_RXD 12 = PWM0_CH2

GPIOH Multiple Function Control Register 1 (SYS_GPH_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPH_MFP1	SYS_BA+0x574	R/W	GPIOH Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PH7MFP				
23	22	21	20	19	18	17	16
Reserved			PH6MFP				
15	14	13	12	11	10	9	8
Reserved			PH5MFP				
7	6	5	4	3	2	1	0
Reserved			PH4MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PH7MFP**	**PH.7 Multi-function Pin Selection** 02 = EBI_ADR0 03 = SPI1_SS 04 = UART7_RXD
[23:21]	**Reserved**	Reserved.
[20:16]	**PH6MFP**	**PH.6 Multi-function Pin Selection** 02 = EBI_ADR1 03 = SPI1_CLK 04 = UART7_TXD
[15:13]	**Reserved**	Reserved.
[12:8]	**PH5MFP**	**PH.5 Multi-function Pin Selection** 02 = EBI_ADR2 03 = SPI1_MOSI 04 = UART7_nCTS 05 = UART6_RXD
[7:5]	**Reserved**	Reserved.
[4:0]	**PH4MFP**	**PH.4 Multi-function Pin Selection** 02 = EBI_ADR3 03 = SPI1_MISO 04 = UART7_nRTS 05 = UART6_TXD

GPIOH Multiple Function Control Register 2 (SYS_GPH_MFP2)

Register	Offset	R/W	Description	Reset Value
SYS_GPH_MFP2	SYS_BA+0x578	R/W	GPIOH Multiple Function Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			PH11MFP				
23	22	21	20	19	18	17	16
Reserved			PH10MFP				
15	14	13	12	11	10	9	8
Reserved			PH9MFP				
7	6	5	4	3	2	1	0
Reserved			PH8MFP				

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**PH11MFP**	**PH.11 Multi-function Pin Selection** 02 = EBI_AD15 03 = QSPI0_MOSI1 07 = UART4_RXD 08 = UART0_RXD 11 = EPWM0_CH5 21 = LPUART0_RXD
[23:21]	**Reserved**	Reserved.
[20:16]	**PH10MFP**	**PH.10 Multi-function Pin Selection** 02 = EBI_AD14 03 = QSPI0_MISO1 06 = SPI1_I2SMCLK 07 = UART4_TXD 08 = UART0_TXD 21 = LPUART0_TXD
[15:13]	**Reserved**	Reserved.
[12:8]	**PH9MFP**	**PH.9 Multi-function Pin Selection** 02 = EBI_AD13 03 = QSPI0_SS 06 = SPI1_SS 07 = UART3_nCTS 08 = I2C1_SMBSUS 09 = I2C2_SDA 10 = UART1_RXD
[7:5]	**Reserved**	Reserved.

Bits	Description	
[4:0]	**PH8MFP**	**PH.8 Multi-function Pin Selection** 02 = EBI_AD12 03 = QSPI0_CLK 06 = SPI1_CLK 07 = UART3_nRTS 08 = I2C1_SMBAL 09 = I2C2_SCL 10 = UART1_TXD

6.2.19 System Timer (SysTick)

The Cortex-M23 includes an integrated system timer, SysTick, which provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used as a Real Time Operating System (RTOS) tick timer or as a simple counter.

When system timer is enabled, it will count down from the value in the SysTick Current Value Register (SYST_VAL) to zero, and reload (wrap) to the value in the SysTick Reload Value Register (SYST_LOAD) on the next clock cycle, and then decrement on subsequent clocks. When the counter transitions to zero, the COUNTFLAG status bit is set. The COUNTFLAG bit clears on reads.

The SYST_VAL value is UNKNOWN on reset. Software should write to the register to clear it to zero before enabling the feature. This ensures the timer will count from the SYST_LOAD value rather than an arbitrary value when it is enabled.

If the SYST_LOAD is zero, the timer will be maintained with a current value of zero after it is reloaded with this value. This mechanism can be used to disable the feature independently from the timer enable bit.

For more detailed information, please refer to the "*Arm Cortex-M23 Technical Reference Manual*" and "*Arm v8-M Architecture Reference Manual*".

6.2.20 System Timer Control Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
SYST Base Address: **SCS_BA = 0xE000_E000**				
SYST_CTL	SCS_BA+0x10	R/W	SysTick Control and Status Register	0x0000_0000
SYST_LOAD	SCS_BA+0x14	R/W	SysTick Reload Value Register	0xXXXX_XXXX
SYST_VAL	SCS_BA+0x18	R/W	SysTick Current Value Register	0xXXXX_XXXX

6.2.20.1 System Timer Control Register Description

SysTick Control and Status Register (SYST_CTL)

Register	Offset	R/W	Description	Reset Value
SYST_CTL	SCS_BA+0x10	R/W	SysTick Control and Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**COUNTFLAG**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**CLKSRC**	**TICKINT**	**ENABLE**

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**COUNTFLAG**	**System Tick Counter Flag** Returns 1 if timer counted to 0 since last time this register was read. COUNTFLAG is set by a count transition from 1 to 0. COUNTFLAG is cleared on read or by a write to the Current Value register.
[15:3]	**Reserved**	Reserved.
[2]	**CLKSRC**	**System Tick Clock Source Selection** 0 = Clock source is the (optional) external reference clock. 1 = Core clock used for SysTick.
[1]	**TICKINT**	**System Tick Interrupt Enabled** 0 = Counting down to 0 does not cause the SysTick exception to be pended. Software can use COUNTFLAG to determine if a count to zero has occurred. 1 = Counting down to 0 will cause the SysTick exception to be pended. Clearing the SysTick current value register by a register write in software will not cause SysTick to be pended.
[0]	**ENABLE**	**System Tick Counter Enabled** 0 = Counter Disabled. 1 = Counter will operate in a multi-shot manner.

SysTick Reload Value Register (SYST_LOAD)

Register	Offset	R/W	Description	Reset Value
SYST_LOAD	SCS_BA+0x14	R/W	SysTick Reload Value Register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RELOAD							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RELOAD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RELOAD							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**RELOAD**	**System Tick Reload Value** Value to load into the Current Value register when the counter reaches 0.

SysTick Current Value Register (SYST_VAL)

Register	Offset	R/W	Description	Reset Value
SYST_VAL	SCS_BA+0x18	R/W	SysTick Current Value Register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CURRENT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CURRENT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CURRENT							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CURRENT**	**System Tick Current Value** Current counter value. This is the value of the counter at the time it is sampled. The counter does not provide read-modify-Write Protect. The register is write-clear. A software write of any value will clear the register to 0.

6.2.21 Nested Vectored Interrupt Controller (NVIC)

The NVIC and the processor core interface are closely coupled to enable low latency interrupt processing and efficient processing of late arriving interrupts. The NVIC maintains knowledge of the stacked, or nested, interrupts to enable tail-chaining of interrupts. You can only fully access the NVIC from privileged mode, but you can cause interrupts to enter a pending state in user mode if you enable the Configuration and Control Register. Any other user mode access causes a bus fault. You can access all NVIC registers using byte, halfword, and word accesses unless otherwise stated. NVIC registers are located within the SCS (System Control Space). All NVIC registers and system debug registers are little-endian regardless of the endianness state of the processor.

The NVIC supports:

- An implementation-defined number of interrupts, in the range 1-240 interrupts.
- A programmable priority level of 0-3 for each interrupt; a higher level corresponds to a lower priority, so level 0 is the highest interrupt priority.
- Level and pulse detection of interrupt signals.
- Dynamic reprioritization of interrupts.
- Grouping of priority values into group priority and subpriority fields.
- Interrupt tail-chaining.
- An external Non Maskable Interrupt (NMI)
- WIC with Ultra-low Power Sleep mode support

The processor automatically stacks its state on exception entry and unstacks this state on exception exit, with no instruction overhead. This provides low latency exception handling.

6.2.21.1 Exception Model and System Interrupt Map

Table 6.2-12 lists the exception model supported by M2L31 Series. Software can set 16 levels of priority on some of these exceptions as well as on all interrupts. The highest user-configurable priority is denoted as “0x00” and the lowest priority is denoted as “0xF0” (The 4-LSB always 0). The default priority of all the user-configurable interrupts is “0x00”. Note that priority “0” is treated as the fourth priority on the system, after three system exceptions “Reset”, “NMI” and “Hard Fault”.

When any interrupts is accepted, the processor will automatically fetch the starting address of the interrupt service routine (ISR) from a vector table in memory. On system reset, the vector table is fixed at address 0x00000000. Privileged software can write to the VTOR to relocate the vector table start address to a different memory location, in the range 0x00000080 to 0x3FFFFF80,

The vector table contains the initialization value for the stack pointer on reset, and the entry point addresses for all exception handlers. The vector number on previous page defines the order of entries in the vector table associated with exception handler entry as illustrated in previous section.

Exception Type	Vector Number	Vector Address	Priority
Reset	1	0x00000004	-3
NMI	2	0x00000008	-2
Hard Fault	3	0x0000000C	-1
Memory Manager Fault	4	0x00000010	Configurable
Bus Fault	5	0x00000014	Configurable
Usage Fault	6	0x00000018	Configurable
Reserved	7 ~ 10		Reserved

SVCall	11	0x0000002C	Configurable
Reserved	12~13		Reserved
PendSV	14	0x00000038	Configurable
SysTick	15	0x0000003C	Configurable
Interrupt (IRQ0 ~ IRQ)	16 ~ 159	0x00000000 + (Vector Number)*4	Configurable

Table 6.2-12 Exception Model

Vector Number	Interrupt Number (Bit In Interrupt Registers)	Interrupt Name	Interrupt Description
0 ~ 15	-	-	System exceptions
16	0	BODOUT	Brown-Out low voltage detected interrupt
17	1	IRC_INT	IRC TRIM interrupt
18	2	PWRWU_INT	Clock controller interrupt for chip wake-up from power-down state
19	3	SRAM_PERR	SRAM parity check error interrupt
20	4	CLKFAIL	Clock fail detected interrupt
21	5	RMC_INT	RRAM Memory Controller interrupt
22	6	RTC_INT	Real time clock interrupt
23	7	TAMPER_INT	Backup register tamper interrupt
24	8	WDT_INT	Watchdog Timer interrupt
25	9	WWDT_INT	Window Watchdog Timer interrupt
26	10	EINT0	External interrupt from INT0 pins
27	11	EINT1	External interrupt from INT1 pins
28	12	EINT2	External interrupt from INT2 pin
29	13	EINT3	External interrupt from INT3 pin
30	14	EINT4	External interrupt from INT4 pin
31	15	EINT5	External interrupt from INT5 pin
32	16	GPA_INT	External interrupt from PA[15:0] pin
33	17	GPB_INT	External interrupt from PB[15:0] pin
34	18	GPC_INT	External interrupt from PC[15:0] pin
35	19	GPD_INT	External interrupt from PD[15:0] pin
36	20	GPE_INT	External interrupt from PE[15:0] pin
37	21	GPF_INT	External interrupt from PF[15:0] pin
38	22	QSPI0_INT	QSPI0 interrupt
39	23	SPI0_INT	SPI0 interrupt

40	24	EBRAKE0_INT	EPWM0 brake interrupt
41	25	EPWM0_P0_INT	EPWM0 pair 0 interrupt
42	26	EPWM0_P1_INT	EPWM0 pair 1 interrupt
43	27	EPWM0_P2_INT	EPWM0 pair 2 interrupt
44	28	EBRAKE1_INT	EPWM1 brake interrupt
45	29	EPWM1_P0_INT	EPWM1 pair 0 interrupt
46	30	EPWM1_P1_INT	EPWM1 pair 1 interrupt
47	31	EPWM1_P2_INT	EPWM1 pair 2 interrupt
48	32	TMR0_INT	Timer 0 interrupt
49	33	TMR1_INT	Timer 1 interrupt
50	34	TMR2_INT	Timer 2 interrupt
51	35	TMR3_INT	Timer 3 interrupt
52	36	UART0_INT	UART0 interrupt
53	37	UART1_INT	UART1 interrupt
54	38	I2C0_INT	I2C0 interrupt
55	39	I2C1_INT	I2C1 interrupt
56	40	PDMA0_INT	PDMA0 interrupt
57	41	DAC_INT	DAC interrupt
58	42	EADC0_INT0	EADC0 interrupt source 0
59	43	EADC0_INT1	EADC0 interrupt source 1
60	44	ACMP01_INT	ACMP0 and ACMP1 interrupt
61	45	ACMP2_INT	ACMP2 interrupt
62	46	EADC0_INT2	EADC0 interrupt source 2
63	47	EADC0_INT3	EADC0 interrupt source 3
64	48	UART2_INT	UART2 interrupt
65	49	UART3_INT	UART3 interrupt
66	50	Reserved	Reserved
67	51	SPI1_INT	SPI1 interrupt
68	52	SPI2_INT	SPI2 interrupt
69	53	USBD_INT	USB device interrupt
70	54	USBH_INT	USB host interrupt
71	55	USBOTG_INT	USB OTG interrupt
72	56	ETI_INT	External Trigger Interface interrupt
73	57	CRC0_INT	CRC0 interrupt
74	58	Reserved	Reserved

75	59	Reserved	Reserved
76	60	Reserved	Reserved
77	61	Reserved	Reserved
78	62	SPI3_INT	SPI3 interrupt
79	63	TK_INT	Touchkey interrupt
80	64	Reserved	Reserved
81	65	Reserved	Reserved
82	66	Reserved	Reserved
83	67	Reserved	Reserved
84	68	Reserved	Reserved
85	69	Reserved	Reserved
86	70	OPA_INT	Analog OPA interrupt
87	71	CRYPTO	Crypto interrupt
88	72	GPG_INT	External interrupt from PG[15:0] pin
89	73	EINT6	External interrupt from INT6 pin
90	74	UART4_INT	UART4 interrupt
91	75	UART5_INT	UART5 interrupt
92	76	USCI0_INT	USCI0 interrupt
93	77	USCI1_INT	USCI1 interrupt
94	78	Reserved	Reserved
95	79	Reserved	Reserved
96	80	Reserved	Reserved
97	81	Reserved	Reserved
98	82	I2C2_INT	I2C2 interrupt
99	83	I2C3_INT	I2C3 interrupt
100	84	EQEI0_INT	EQEI0 interrupt
101	85	EQEI1_INT	EQEI1 interrupt
102	86	ECAP0_INT	ECAP0 interrupt
103	87	ECAP1_INT	ECAP1 interrupt
104	88	GPH_INT	External interrupt from PH[15:0] pin
105	89	EINT7	External interrupt from INT7 pin
106	90	Reserved	Reserved
107	91	Reserved	Reserved
108	92	Reserved	Reserved
109	93	Reserved	Reserved

110	94	Reserved	Reserved
111	95	Reserved	Reserved
112	96	Reserved	Reserved
113	97	Reserved	Reserved
114	98	LPPDMA0_INT	LPPDMA0 interrupt
115	99	Reserved	Reserved
116	100	Reserved	Reserved
117	101	TRNG_INT	TRNG interrupt
118	102	UART6_INT	UART6 interrupt
119	103	UART7_INT	UART7 interrupt
120	104	Reserved	Reserved
121	105	Reserved	Reserved
122	106	Reserved	Reserved
123	107	Reserved	Reserved
124	108	UTCPD_INT	UTCPD interrupt
125	109	Reserved	Reserved
126	110	Reserved	Reserved
127	111	Reserved	Reserved
128	112	CANFD00	CANFD00 interrupt
129	113	CANFD01	CANFD01 interrupt
130	114	CANFD10	CANFD10 interrupt
131	115	CANFD11	CANFD11 interrupt
132	116	Reserved	Reserved
133	117	Reserved	Reserved
134	118	Reserved	Reserved
135	119	Reserved	Reserved
136	120	Reserved	Reserved
137	121	Reserved	Reserved
138	122	Reserved	Reserved
139	123	Reserved	Reserved
140	124	Reserved	Reserved
141	125	Reserved	Reserved
142	126	Reserved	Reserved
143	127	Reserved	Reserved
144	128	BRAKE0_INT	PWM0 brake interrupt

145	129	PWM0_P0_INT	PWM0 pair 0 interrupt
146	130	PWM0_P1_INT	PWM0 pair 1 interrupt
147	131	PWM0_P2_INT	PWM0 pair 2 interrupt
148	132	BRAKE1_INT	PWM1 brake interrupt
149	133	PWM1_P0_INT	PWM1 pair 0 interrupt
150	134	PWM1_P1_INT	PWM1 pair 1 interrupt
151	135	PWM1_P2_INT	PWM1 pair 2 interrupt
152	136	LPADC0_INT	LPADC0 interrupt
153	137	LPUART0_INT	LPUART0 interrupt
154	138	LPI2C0_INT	LPI2C0 interrupt
155	139	LPSPI0_INT	LPSPI0 interrupt
156	140	LPTMR0_INT	LPTMR0 interrupt
157	141	LPTMR1_INT	LPTMR1 interrupt
158	142	TTMR0_INT	TTMR0 interrupt
159	143	TTMR1_INT	TTMR1 interrupt

Table 6.2-13 Interrupt Number Table

6.2.21.2 Operation Description

NVIC interrupts can be enabled and disabled by writing to their corresponding Interrupt Set-Enable or Interrupt Clear-Enable register bit-field. The registers use a write-1-to-enable and write-1-to-clear policy, both registers reading back the current enabled state of the corresponding interrupts. When an interrupt is disabled, interrupt assertion will cause the interrupt to become Pending, however, the interrupt will not activate. If an interrupt is Active when it is disabled, it remains in its Active state until cleared by reset or an exception return. Clearing the enable bit prevents new activations of the associated interrupt.

NVIC interrupts can be pended/un-pended using a complementary pair of registers to those used to enable/disable the interrupts, named the Set-Pending Register and Clear-Pending Register respectively. The registers use a write-1-to-enable and write-1-to-clear policy, both registers reading back the current pended state of the corresponding interrupts. The Clear-Pending Register has no effect on the execution status of an Active interrupt.

NVIC interrupts are prioritized by updating an 8-bit field within a 32-bit register (each register supporting four interrupts).

The general registers associated with the NVIC are all accessible from a block of memory in the System Control Space and will be described in next section.

6.2.21.3 NVIC Control Registers

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
NVIC Base Address: **NVIC_BA = 0xE000_E100**				
NVIC_ISER0	NVIC_BA+0x00	R/W	IRQ0 ~ IRQ143 Set-enable Control Register	0x0000_0000
NVIC_ISER1	NVIC_BA+0x04	R/W	IRQ0 ~ IRQ143 Set-enable Control Register	0x0000_0000
NVIC_ISER2	NVIC_BA+0x08	R/W	IRQ0 ~ IRQ143 Set-enable Control Register	0x0000_0000
NVIC_ISER3	NVIC_BA+0x0C	R/W	IRQ0 ~ IRQ143 Set-enable Control Register	0x0000_0000
NVIC_ISER4	NVIC_BA+0x10	R/W	IRQ0 ~ IRQ143 Set-enable Control Register	0x0000_0000
NVIC_ICER0	NVIC_BA+0x80	R/W	IRQ0 ~ IRQ143 Clear-enable Control Register	0x0000_0000
NVIC_ICER1	NVIC_BA+0x84	R/W	IRQ0 ~ IRQ143 Clear-enable Control Register	0x0000_0000
NVIC_ICER2	NVIC_BA+0x88	R/W	IRQ0 ~ IRQ143 Clear-enable Control Register	0x0000_0000
NVIC_ICER3	NVIC_BA+0x8C	R/W	IRQ0 ~ IRQ143 Clear-enable Control Register	0x0000_0000
NVIC_ICER4	NVIC_BA+0x90	R/W	IRQ0 ~ IRQ143 Clear-enable Control Register	0x0000_0000
NVIC_ISPR0	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ143 Set-pending Control Register	0x0000_0000
NVIC_ISPR1	NVIC_BA+0x104	R/W	IRQ0 ~ IRQ143 Set-pending Control Register	0x0000_0000
NVIC_ISPR2	NVIC_BA+0x108	R/W	IRQ0 ~ IRQ143 Set-pending Control Register	0x0000_0000
NVIC_ISPR3	NVIC_BA+0x10C	R/W	IRQ0 ~ IRQ143 Set-pending Control Register	0x0000_0000
NVIC_ISPR4	NVIC_BA+0x110	R/W	IRQ0 ~ IRQ143 Set-pending Control Register	0x0000_0000
NVIC_ICPR0	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ143 Clear-pending Control Register	0x0000_0000
NVIC_ICPR1	NVIC_BA+0x184	R/W	IRQ0 ~ IRQ143 Clear-pending Control Register	0x0000_0000
NVIC_ICPR2	NVIC_BA+0x188	R/W	IRQ0 ~ IRQ143 Clear-pending Control Register	0x0000_0000
NVIC_ICPR3	NVIC_BA+0x18C	R/W	IRQ0 ~ IRQ143 Clear-pending Control Register	0x0000_0000
NVIC_ICPR4	NVIC_BA+0x190	R/W	IRQ0 ~ IRQ143 Clear-pending Control Register	0x0000_0000
NVIC_IABR0	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ143 Active Bit Register	0x0000_0000
NVIC_IABR1	NVIC_BA+0x204	R/W	IRQ0 ~ IRQ143 Active Bit Register	0x0000_0000
NVIC_IABR2	NVIC_BA+0x208	R/W	IRQ0 ~ IRQ143 Active Bit Register	0x0000_0000
NVIC_IABR3	NVIC_BA+0x20C	R/W	IRQ0 ~ IRQ143 Active Bit Register	0x0000_0000
NVIC_IABR4	NVIC_BA+0x210	R/W	IRQ0 ~ IRQ143 Active Bit Register	0x0000_0000
NVIC_IPRn **n=0,1..35**	NVIC_BA+0x300 +0x4*n	R/W	IRQ0 ~ IRQ143 Priority Control Register	0x0000_0000

STIR	NVIC_BA+0xE00	R/W	Software Trigger Interrupt Registers	0x0000_0000

IRQ0 ~ IRQ143 Set-enable Control Register (NVIC_ISER0)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER0	NVIC_BA+0x00	R/W	IRQ0 ~ IRQ143 Set-enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SETENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETENA							

Bits	Description	
[31:0]	**SETENA**	**Interrupt Set Enable Bit** The NVIC_ISER0-NVIC_ISER4 registers enable interrupts, and show which interrupts are enabled Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ143 Set-enable Control Register (NVIC_ISER1)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER1	NVIC_BA+0x04	R/W	IRQ0 ~ IRQ143 Set-enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SETENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETENA							

Bits	Description	
[31:0]	**SETENA**	**Interrupt Set Enable Bit** The NVIC_ISER0-NVIC_ISER4 registers enable interrupts, and show which interrupts are enabled Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ143 Set-enable Control Register (NVIC_ISER2)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER2	NVIC_BA+0x08	R/W	IRQ0 ~ IRQ143 Set-enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SETENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETENA							

Bits	Description	
[31:0]	**SETENA**	**Interrupt Set Enable Bit** The NVIC_ISER0-NVIC_ISER4 registers enable interrupts, and show which interrupts are enabled Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ143 Set-enable Control Register (NVIC_ISER3)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER3	NVIC_BA+0x0C	R/W	IRQ0 ~ IRQ143 Set-enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SETENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETENA							

Bits	Description	
[31:0]	**SETENA**	**Interrupt Set Enable Bit** The NVIC_ISER0-NVIC_ISER4 registers enable interrupts, and show which interrupts are enabled Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ143 Set-enable Control Register (NVIC_ISER4)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER4	NVIC_BA+0x10	R/W	IRQ0 ~ IRQ143 Set-enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SETENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETENA							

Bits	Description	
[31:0]	**SETENA**	**Interrupt Set Enable Bit** The NVIC_ISER0-NVIC_ISER4 registers enable interrupts, and show which interrupts are enabled Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ143 Clear-enable Control Register (NVIC_ICER0)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER0	NVIC_BA+0x80	R/W	IRQ0 ~ IRQ143 Clear-enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CALENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALENA							

Bits	Description	
[31:0]	**CALENA**	**Interrupt Clear Enable Bit** The NVIC_ICER0-NVIC_ICER4 registers disable interrupts, and show which interrupts are enabled. Write Operation: 0 = No effect. 1 = Interrupt Disabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ143 Clear-enable Control Register (NVIC_ICER1)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER1	NVIC_BA+0x84	R/W	IRQ0 ~ IRQ143 Clear-enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CALENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALENA							

Bits	Description	
[31:0]	**CALENA**	**Interrupt Clear Enable Bit** The NVIC_ICER0-NVIC_ICER4 registers disable interrupts, and show which interrupts are enabled. Write Operation: 0 = No effect. 1 = Interrupt Disabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ143 Clear-enable Control Register (NVIC_ICER2)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER2	NVIC_BA+0x88	R/W	IRQ0 ~ IRQ143 Clear-enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CALENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALENA							

Bits	Description	
[31:0]	**CALENA**	**Interrupt Clear Enable Bit** The NVIC_ICER0-NVIC_ICER4 registers disable interrupts, and show which interrupts are enabled. Write Operation: 0 = No effect. 1 = Interrupt Disabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ143 Clear-enable Control Register (NVIC_ICER3)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER3	NVIC_BA+0x8C	R/W	IRQ0 ~ IRQ143 Clear-enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CALENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALENA							

Bits	Description	
[31:0]	**CALENA**	**Interrupt Clear Enable Bit** The NVIC_ICER0-NVIC_ICER4 registers disable interrupts, and show which interrupts are enabled. Write Operation: 0 = No effect. 1 = Interrupt Disabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ143 Clear-enable Control Register (NVIC_ICER4)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER4	NVIC_BA+0x90	R/W	IRQ0 ~ IRQ143 Clear-enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CALENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALENA							

Bits	Description	
[31:0]	**CALENA**	**Interrupt Clear Enable Bit** The NVIC_ICER0-NVIC_ICER4 registers disable interrupts, and show which interrupts are enabled. Write Operation: 0 = No effect. 1 = Interrupt Disabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ143 Set-pending Control Register (NVIC_ISPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR0	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ143 Set-pending Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SETPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETPEND							

Bits	Description	
[31:0]	**SETPEND**	**Interrupt Set-pending** The NVIC_ISPR0-NVIC_ISPR4 registers force interrupts into the pending state, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ143 Set-pending Control Register (NVIC_ISPR1)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR1	NVIC_BA+0x104	R/W	IRQ0 ~ IRQ143 Set-pending Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SETPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETPEND							

Bits	Description	
[31:0]	**SETPEND**	**Interrupt Set-pending** The NVIC_ISPR0-NVIC_ISPR4 registers force interrupts into the pending state, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ143 Set-pending Control Register (NVIC_ISPR2)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR2	NVIC_BA+0x108	R/W	IRQ0 ~ IRQ143 Set-pending Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SETPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETPEND							

Bits	Description	
[31:0]	**SETPEND**	**Interrupt Set-pending** The NVIC_ISPR0-NVIC_ISPR4 registers force interrupts into the pending state, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ143 Set-pending Control Register (NVIC_ISPR3)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR3	NVIC_BA+0x10C	R/W	IRQ0 ~ IRQ143 Set-pending Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SETPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETPEND							

Bits	Description	
[31:0]	**SETPEND**	**Interrupt Set-pending** The NVIC_ISPR0-NVIC_ISPR4 registers force interrupts into the pending state, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ143 Set-pending Control Register (NVIC_ISPR4)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR4	NVIC_BA+0x110	R/W	IRQ0 ~ IRQ143 Set-pending Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SETPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETPEND							

Bits	Description	
[31:0]	**SETPEND**	**Interrupt Set-pending** The NVIC_ISPR0-NVIC_ISPR4 registers force interrupts into the pending state, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ143 Clear-pending Control Register (NVIC_ICPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR0	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ143 Clear-pending Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CALPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALPEND							

Bits	Description	
[31:0]	**CALPEND**	**Interrupt Clear-pending** The NVIC_ICPR0-NVIC_ICPR4 registers remove the pending state from interrupts, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Removes pending state an interrupt. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ143 Clear-pending Control Register (NVIC_ICPR1)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR1	NVIC_BA+0x184	R/W	IRQ0 ~ IRQ143 Clear-pending Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CALPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALPEND							

Bits	Description	
[31:0]	**CALPEND**	**Interrupt Clear-pending** The NVIC_ICPR0-NVIC_ICPR4 registers remove the pending state from interrupts, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Removes pending state an interrupt. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ143 Clear-pending Control Register (NVIC_ICPR2)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR2	NVIC_BA+0x188	R/W	IRQ0 ~ IRQ143 Clear-pending Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CALPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALPEND							

Bits	Description	
[31:0]	**CALPEND**	**Interrupt Clear-pending** The NVIC_ICPR0-NVIC_ICPR4 registers remove the pending state from interrupts, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Removes pending state an interrupt. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ143 Clear-pending Control Register (NVIC_ICPR3)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR3	NVIC_BA+0x18C	R/W	IRQ0 ~ IRQ143 Clear-pending Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CALPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALPEND							

Bits	Description	
[31:0]	**CALPEND**	**Interrupt Clear-pending** The NVIC_ICPR0-NVIC_ICPR4 registers remove the pending state from interrupts, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Removes pending state an interrupt. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ143 Clear-pending Control Register (NVIC_ICPR4)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR4	NVIC_BA+0x190	R/W	IRQ0 ~ IRQ143 Clear-pending Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CALPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALPEND							

Bits	Description	
[31:0]	**CALPEND**	**Interrupt Clear-pending** The NVIC_ICPR0-NVIC_ICPR4 registers remove the pending state from interrupts, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Removes pending state an interrupt. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ143 Active Bit Register (NVIC_IABR0)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR0	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ143 Active Bit Register	0x0000_0000

31	30	29	28	27	26	25	24
ACTIVE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ACTIVE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACTIVE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACTIVE							

Bits	Description	
[31:0]	**ACTIVE**	**Interrupt Active Flags** The NVIC_IABR0-NVIC_IABR4 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ0 ~ IRQ143 Active Bit Register (NVIC_IABR1)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR1	NVIC_BA+0x204	R/W	IRQ0 ~ IRQ143 Active Bit Register	0x0000_0000

31	30	29	28	27	26	25	24
ACTIVE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ACTIVE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACTIVE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACTIVE							

Bits	Description	
[31:0]	**ACTIVE**	**Interrupt Active Flags** The NVIC_IABR0-NVIC_IABR4 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ0 ~ IRQ143 Active Bit Register (NVIC_IABR2)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR2	NVIC_BA+0x208	R/W	IRQ0 ~ IRQ143 Active Bit Register	0x0000_0000

31	30	29	28	27	26	25	24
ACTIVE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ACTIVE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACTIVE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACTIVE							

Bits	Description	
[31:0]	**ACTIVE**	**Interrupt Active Flags** The NVIC_IABR0-NVIC_IABR4 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ0 ~ IRQ143 Active Bit Register (NVIC_IABR3)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR3	NVIC_BA+0x20C	R/W	IRQ0 ~ IRQ143 Active Bit Register	0x0000_0000

31	30	29	28	27	26	25	24
ACTIVE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ACTIVE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACTIVE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACTIVE							

Bits	Description	
[31:0]	**ACTIVE**	**Interrupt Active Flags** The NVIC_IABR0-NVIC_IABR4 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ0 ~ IRQ143 Active Bit Register (NVIC_IABR4)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR4	NVIC_BA+0x210	R/W	IRQ0 ~ IRQ143 Active Bit Register	0x0000_0000

31	30	29	28	27	26	25	24
ACTIVE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ACTIVE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACTIVE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACTIVE							

Bits	Description	
[31:0]	**ACTIVE**	**Interrupt Active Flags** The NVIC_IABR0-NVIC_IABR4 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ0 ~ IRQ143 Interrupt Priority Register (NVIC_IPRn)

Register	Offset	R/W	Description	Reset Value
NVIC_IPRn **n=0,1..35**	NVIC_BA+0x300 +0x4*n	R/W	IRQ0 ~ IRQ143 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_4n_3		Reserved					
23	22	21	20	19	18	17	16
PRI_4n_2		Reserved					
15	14	13	12	11	10	9	8
PRI_4n_1		Reserved					
7	6	5	4	3	2	1	0
PRI_4n_0		Reserved					

Bits	Description	
[31:30]	**PRI_4n_3**	**Priority of IRQ_4n+3** “0” denotes the highest priority and “3” denotes the lowest priority
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_4n_2**	**Priority of IRQ_4n+2** “0” denotes the highest priority and “3” denotes the lowest priority
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_4n_1**	**Priority of IRQ_4n+1** “0” denotes the highest priority and “3” denotes the lowest priority
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_4n_0**	**Priority of IRQ_4n+0** “0” denotes the highest priority and “3” denotes the lowest priority
[5:0]	**Reserved**	Reserved.

Software Trigger Interrupt Register (STIR)

Register	Offset	R/W	Description	Reset Value
STIR	NVIC_BA+0xE00	R/W	Software Trigger Interrupt Registers	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**INTID**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTID							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**INTID**	**Interrupt ID** Write to the STIR To Generate An Interrupt from Software When the USERSETMPEND bit in the SCR is set to 1, unprivileged software can access the STIR Interrupt ID of the interrupt to trigger, in the range 0-63. For example, a value of 0x03 specifies interrupt IRQ3.

6.2.21.4 NMI Control Registers

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
NMI Base Address: **NMI_BA = 0x4000_0300**				
NMIEN	NMI_BA+0x00	R/W	NMI Source Interrupt Enable Register	0x0000_0000
NMISTS	NMI_BA+0x04	R	NMI Source Interrupt Status Register	0x0000_0000

NMI Source Interrupt Enable Register (NMIEN)

Register	Offset	R/W	Description	Reset Value
NMIEN	NMI_BA+0x00	R/W	NMI Source Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
UART1_INT	**UART0_INT**	**EINT5**	**EINT4**	**EINT3**	**EINT2**	**EINT1**	**EINT0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TAMPER_INT	**RTC_INT**	**Reserved**	**CLKFAIL**	**SRAM_PERR**	**PWRWU_INT**	**IRC_INT**	**BODOUT**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**UART1_INT**	**UART1 NMI Source Enable (Write Protect)** 0 = UART1 NMI source Disabled. 1 = UART1 NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[14]	**UART0_INT**	**UART0 NMI Source Enable (Write Protect)** 0 = UART0 NMI source Disabled. 1 = UART0 NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[13]	**EINT5**	**External Interrupt From INT5 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from INT5 pin NMI source Disabled. 1 = External interrupt from INT5 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[12]	**EINT4**	**External Interrupt From INT4 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from INT4 pin NMI source Disabled. 1 = External interrupt from INT4 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[11]	**EINT3**	**External Interrupt From INT3 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from INT3 pin NMI source Disabled. 1 = External interrupt from INT3 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[10]	**EINT2**	**External Interrupt From INT2 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from INT2 pin NMI source Disabled. 1 = External interrupt from INT2 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[9]	**EINT1**	**External Interrupt From INT1 Pin NMI Source Enable (Write Protect)**

		0 = External interrupt from INT1 pin NMI source Disabled. 1 = External interrupt from INT1 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[8]	**EINT0**	**External Interrupt From INT0 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from INT0 pin NMI source Disabled. 1 = External interrupt from INT0 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7]	**TAMPER_INT**	**TAMPER_INT NMI Source Enable (Write Protect)** 0 = Backup register tamper detected interrupt.NMI source Disabled. 1 = Backup register tamper detected interrupt.NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6]	**RTC_INT**	**RTC NMI Source Enable (Write Protect)** 0 = RTC NMI source Disabled. 1 = RTC NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5]	**Reserved**	Reserved.
[4]	**CLKFAIL**	**Clock Fail Detected NMI Source Enable (Write Protect)** 0 = Clock fail detected interrupt NMI source Disabled. 1 = Clock fail detected interrupt NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**SRAM_PERR**	**SRAM ParityCheck Error NMI Source Enable (Write Protect)** 0 = SRAM parity check error NMI source Disabled. 1 = SRAM parity check error NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**PWRWU_INT**	**Power-down Mode Wake-up NMI Source Enable (Write Protect)** 0 = Power-down mode wake-up NMI source Disabled. 1 = Power-down mode wake-up NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1]	**IRC_INT**	**IRC TRIM NMI Source Enable (Write Protect)** 0 = IRC TRIM NMI source Disabled. 1 = IRC TRIM NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**BODOUT**	**BOD NMI Source Enable (Write Protect)** 0 = BOD NMI source Disabled. 1 = BOD NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

NMI Source Interrupt Status Register (NMISTS)

Register	Offset	R/W	Description	Reset Value
NMISTS	NMI_BA+0x04	R	NMI Source Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
UART1_INT	**UART0_INT**	**EINT5**	**EINT4**	**EINT3**	**EINT2**	**EINT1**	**EINT0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TAMPER_INT	**RTC_INT**	**Reserved**	**CLKFAIL**	**SRAM_PERR**	**PWRWU_INT**	**IRC_INT**	**BODOUT**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**UART1_INT**	**UART1 Interrupt Flag (Read Only)** 0 = UART1 interrupt is deasserted. 1 = UART1 interrupt is asserted.
[14]	**UART0_INT**	**UART0 Interrupt Flag (Read Only)** 0 = UART1 interrupt is deasserted. 1 = UART1 interrupt is asserted.
[13]	**EINT5**	**External Interrupt From INT5 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from INT5 interrupt is deasserted. 1 = External Interrupt from INT5 interrupt is asserted.
[12]	**EINT4**	**External Interrupt From INT4 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from INT4 interrupt is deasserted. 1 = External Interrupt from INT4 interrupt is asserted.
[11]	**EINT3**	**External Interrupt From INT3 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from INT3 interrupt is deasserted. 1 = External Interrupt from INT3 interrupt is asserted.
[10]	**EINT2**	**External Interrupt From INT2 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from INT2 interrupt is deasserted. 1 = External Interrupt from INT2 interrupt is asserted.
[9]	**EINT1**	**External Interrupt From INT1 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from INT1 interrupt is deasserted. 1 = External Interrupt from INT1 interrupt is asserted.
[8]	**EINT0**	**External Interrupt From INT0 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from INT0 interrupt is deasserted. 1 = External Interrupt from INT0 interrupt is asserted.
[7]	**TAMPER_INT**	**TAMPER_INT Interrupt Flag (Read Only)**

		0 = Backup register tamper detected interrupt is deasserted. 1 = Backup register tamper detected interrupt is asserted.
[6]	**RTC_INT**	**RTC Interrupt Flag (Read Only)** 0 = RTC interrupt is deasserted. 1 = RTC interrupt is asserted.
[5]	**Reserved**	Reserved.
[4]	**CLKFAIL**	**Clock Fail Detected Interrupt Flag (Read Only)** 0 = Clock fail detected interrupt is deasserted. 1 = Clock fail detected interrupt is asserted.
[3]	**SRAM_PERR**	**SRAM ParityCheck Error Interrupt Flag (Read Only)** 0 = SRAM parity check error interrupt is deasserted. 1 = SRAM parity check error interrupt is asserted.
[2]	**PWRWU_INT**	**Power-down Mode Wake-up Interrupt Flag (Read Only)** 0 = Power-down mode wake-up interrupt is deasserted. 1 = Power-down mode wake-up interrupt is asserted.
[1]	**IRC_INT**	**IRC TRIM Interrupt Flag (Read Only)** 0 = HIRC TRIM interrupt is deasserted. 1 = HIRC TRIM interrupt is asserted.
[0]	**BODOUT**	**BOD Interrupt Flag (Read Only)** 0 = BOD interrupt is deasserted. 1 = BOD interrupt is asserted.

6.2.21.5 AHB Bus Matrix Priority Control Register

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
AHB Base Address: **AHB_BA = 0x4000_0400**				
AHBMCTL	0x40000400	R/W	AHB Bus Matrix Priority Control Register	0x0000_0001

AHB Bus Matrix Priority Control Register (AHBMCTL)

Register	Offset	R/W	Description	Reset Value
AHBMCTL	0x40000400	R/W	AHB Bus Matrix Priority Control Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**INTACTEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**INTACTEN**	**Highest AHB Bus Priority of Cortex-M23 Core Enable Bit (Write Protect)** Enable Cortex-M23 core With Highest AHB Bus Priority In AHB Bus Matrix 0 = Run robin mode. 1 = Cortex-M23 core with highest bus priority when interrupt occusr. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

6.2.22 System Control Register

The Cortex-M23 status and operation mode control are managed by System Control Registers. Including CPUID, Cortex-M23 interrupt priority and Cortex-M23 power management can be controlled through these system control registers.

For more detailed information, please refer to the "*Arm Cortex-M23 Technical Reference Manual*" and "*Arm v8-M Architecture Reference Manual*".

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
SCR Base Address: **SCS_BA = 0xE000_E000**				
ICSR	SCS_BA+0xD04	R/W	Interrupt Control and State Register	0x0000_0000
VTOR	SCS_BA+0xD08	R/W	Vector Table Offset Register	0x0000_0000
AIRCR	SCS_BA+0xD0C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000
SCR	SCS_BA+0xD10	R/W	System Control Register	0x0000_0000
SHPR2	SCS_BA+0xD1C	R/W	System Handler Priority Register 2	0x0000_0000
SHPR3	SCS_BA+0xD20	R/W	System Handler Priority Register 3	0x0000_0000
SHCSR	SCS_BA+0xD24	R/W	System Handler Control and State Register	0x0000_0000

Interrupt Control State Register (ICSR)

Register	Offset	R/W	Description	Reset Value
ICSR	SCS_BA+0xD04	R/W	Interrupt Control and State Register	0x0000_0000

31	30	29	28	27	26	25	24
NMIPENDSET	Reserved		PENDSVSET	PENDSVRTC_CAL	PENDSTSET	PENDSTRTC_CAL	Reserved
23	22	21	20	19	18	17	16
ISRPREEMPT	ISRPENDING	Reserved				VECTPENDING	
15	14	13	12	11	10	9	8
VECTPENDING				RETTOBASE	Reserved		
7	6	5	4	3	2	1	0
Reserved	VECTACTIVE						

Bits	Description	
[31]	**NMIPENDSET**	**NMI Set-pending Bit** Write Operation: 0 = No effect. 1 = Change NMI exception state to pending. Read Operation: 0 = NMI exception is not pending. 1 = NMI exception is pending. **Note:** Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it detects a write of 1 to this bit. Entering the handler then clears this bit to 0. This means a read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler.
[30:29]	**Reserved**	Reserved.
[28]	**PENDSVSET**	**PendSV Set-pending Bit** Write Operation: 0 = No effect. 1 = Change PendSV exception state to pending. Read Operation: 0 = PendSV exception is not pending. 1 = PendSV exception is pending. **Note:** Writing 1 to this bit is the only way to set the PendSV exception state to pending.
[27]	**PENDSVRTC_CAL**	**PendSV Clear-pending Bit** Write Operation: 0 = No effect. 1 = Remove the pending state from the PendSV exception. **Note:** This is a write only bit. To clear the PENDSV bit, you must “write 0 to PENDSVSET and write 1 to PENDSVRTC_CAL” at the same time.

[26]	**PENDSTSET**	**SysTick Exception Set-pending Bit** Write Operation: 0 = No effect. 1 = Change SysTick exception state to pending. Read Operation: 0 = SysTick exception is not pending. 1 = SysTick exception is pending.
[25]	**PENDSTRTC_CAL**	**SysTick Exception Clear-pending Bit** Write Operation: 0 = No effect. 1 = Remove the pending state from the SysTick exception. **Note:** This is a write only bit. To clear the PENDST bit, you must "write 0 to PENDSTSET and write 1 to PENDSTRTC_CAL" at the same time.
[24]	**Reserved**	Reserved.
[23]	**ISRPREEMPT**	**Interrupt Preempt Bit (Read Only)** If set, a pending exception will be serviced on exit from the debug halt state.
[22]	**ISRPENDING**	**Interrupt Pending Flag, Excluding NMI and Faults (Read Only)** 0 = Interrupt not pending. 1 = Interrupt pending.
[21:18]	**Reserved**	Reserved.
[17:12]	**VECTPENDING**	**Number of the Highest Pended Exception** Indicate the Exception Number of the Highest Priority Pending Enabled Exception 0 = No pending exceptions. Nonzero = the exception number of the highest priority pending enabled exception. The value indicated by this field includes the effect of the BASEPRI and FAULTMASK registers, but not any effect of the PRIMASK register.
[11]	**RETTOBASE**	**Preempted Active Exceptions Indicator** Indicate whether There are Preempted Active Exceptions 0 = there are preempted active exceptions to execute. 1 = there are no active exceptions, or the currently-executing exception is the only active exception.
[10:7]	**Reserved**	Reserved.
[6:0]	**VECTACTIVE**	**Number of the Current Active Exception** 0 = Thread mode. Non-zero = The exception number of the currently active exception.

Vector Table Offset Register (VTOR)

Register	Offset	R/W	Description	Reset Value
VTOR	SCS_BA+0xD08	R/W	Vector Table Offset Register	0x0000_0000

31	30	29	28	27	26	25	24
TBLOFF							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TBLOFF							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TBLOFF							**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:9]	**TBLOFF**	**Table Offset Bits** The vector table address for the selected Security state.
[8:0]	**Reserved**	Reserved.

Application Interrupt and Reset Control Register (AIRCR)

Register	Offset	R/W	Description	Reset Value
AIRCR	SCS_BA+0xD0C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000

31	30	29	28	27	26	25	24
VECTORKEY							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
VECTORKEY							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ENDIANNESS	**Reserved**				**PRIGROUP**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**SYSRESETREQ**	**VECTCLRACTIVE**	**VECTRESET**

Bits	Description	
[31:16]	**VECTORKEY**	**Register Access Key** When writing this register, this field should be 0x05FA, otherwise the write action will be unpredictable. The VECTORKEY filed is used to prevent accidental write to this register from resetting the system or clearing of the exception status.
[15]	**ENDIANNESS**	**Data Endianness** 0 = Little-endian. 1 = Big-endian.
[14:11]	**Reserved**	Reserved.
[10:8]	**PRIGROUP**	**Interrupt Priority Grouping** This field determines the Split Of Group priority from subpriority,
[7:3]	**Reserved**	Reserved.
[2]	**SYSRESETREQ**	**System Reset Request** Writing This Bit to 1 Will Cause A Reset Signal To Be Asserted To The Chip And Indicate A Reset Is Requested This bit is write only and self-cleared as part of the reset sequence.
[1]	**VECTCLRACTIVE**	**Exception Active Status Clear Bit** Setting This Bit To 1 Will Clears All Active State Information For Fixed And Configurable Exceptions This bit is write only and can only be written when the core is halted. **Note:** It is the debugger's responsibility to re-initialize the stack.
[0]	**VECTRESET**	Reserved.

PRIGROUP	Binary Point	Group Priority Bits	Subpriority Bits	Number Of Group Priorities	Subpriorities
0b000	bxxxxxxx.y	[7:1]	[0]	128	2
0b001	bxxxxxx.yy	[7:2]	[1:0]	64	4
0b010	bxxxxx.yyy	[7:3]	[2:0]	32	8
0b011	bxxxx.yyyy	[7:4]	[3;0]	16	16
0b100	bxxx.yyyyy	[7:5]	[4:0]	8	32
0b101	bxx.yyyyyy	[7:6]	[5:0]	4	64
0b110	bx.yyyyyyy	[7]	[6:0]	2	128
0b111	b.yyyyyyyy	None	[7:0]	1	256

Table 6.2-14 Priority Grouping

System Control Register (SCR)

Register	Offset	R/W	Description	Reset Value
SCR	SCS_BA+0xD10	R/W	System Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**SEVONPEND**	**Reserved**	**SLEEPDEEP**	**SLEEPONEXIT**	**Reserved**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**SEVONPEND**	**Send Event on Pending** 0 = Only enabled interrupts or events can wake up the processor, while disabled interrupts are excluded. 1 = Enabled events and all interrupts, including disabled interrupts, can wake up the processor. When an event or interrupt enters pending state, the event signal wakes up the processor from WFE. If the processor is not waiting for an event, the event is registered and affects the next WFE. The processor also wakes up on execution of an SEV instruction or an external event.
[3]	**Reserved**	Reserved.
[2]	**SLEEPDEEP**	**Processor Deep Sleep and Sleep Mode Selection** Control whether the Processor Uses Sleep Or Deep Sleep as its Low Power Mode. 0 = Sleep. 1 = Deep sleep.
[1]	**SLEEPONEXIT**	**Sleep-on-exit Enable Control** This bit indicate Sleep-On-Exit when Returning from Handler Mode to Thread Mode. 0 = Do not sleep when returning to Thread mode. 1 = Enter sleep, or deep sleep, on return from an ISR to Thread mode. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.
[0]	**Reserved**	Reserved.

System Handler Priority Register 2 (SHPR2)

Register	Offset	R/W	Description	Reset Value
SHPR2	SCS_BA+0xD1C	R/W	System Handler Priority Register 2	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11				**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:28]	**PRI_11**	**Priority of System Handler 11 – SVCall** “0” denotes the highest priority and “3” denotes the lowest priority.
[27:0]	**Reserved**	Reserved.

System Handler Priority Register 3 (SHPR3)

Register	Offset	R/W	Description	Reset Value
SHPR3	SCS_BA+0xD20	R/W	System Handler Priority Register 3	0x0000_0000

31	30	29	28	27	26	25	24
PRI_15				**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_14				**Reserved**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:28]	**PRI_15**	**Priority of System Handler 15 – SysTick** “0” denotes the highest priority and “3” denotes the lowest priority.
[27:24]	**Reserved**	Reserved.
[23:20]	**PRI_14**	**Priority of System Handler 14 – PendSV** “0” denotes the highest priority and “3” denotes the lowest priority.
[19:0]	**Reserved**	Reserved.

System Handler Control and State Register (SHCSR)

Register	Offset	R/W	Description	Reset Value
SHCSR	SCS_BA+0xD24	R/W	System Handler Control and State Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**HARDFAULTPENDED**	**Reserved**				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21]	**HARDFAULTPENDED**	**HardFault Exception Pended State** This bit indicates and allows modification of the pending state of the HardFault exception corresponding to the selected Security state. This bit is banked between Security states. The possible values of this bit are: 0 = HardFault exception not pending for the selected Security state. 1 = HardFault exception pending for the selected Security state.
[20:0]	**Reserved**	Reserved.

6.3 Clock Controller

6.3.1 Overview

The clock controller generates clocks for the whole chip, including system clocks and all peripheral clocks. The clock controller also implements the power control function with the individually clock ON/OFF control, clock source selection and a clock divider. The chip will not enter Power-down mode until CPU sets the Power-down enable bit PDEN(CLK_PWRCTL[7]) and Cortex®-M23 core executes the WFI instruction. After that, chip enters Power-down mode and wait for wake-up interrupt source triggered to leave Power-down mode. In Power-down mode, the clock controller turns off the 4~32 MHz external high speed crystal (HXT), 12 MHz internal high speed RC oscillator (HIRC), 48 MHz internal high speed RC oscillator (HIRC48M) and 1/2/4/8 MHz internal multiple speed RC oscillator (MIRC) to reduce the overall system power consumption. Figure 6.3-1 shows the clock generator and the overview of the clock source control.

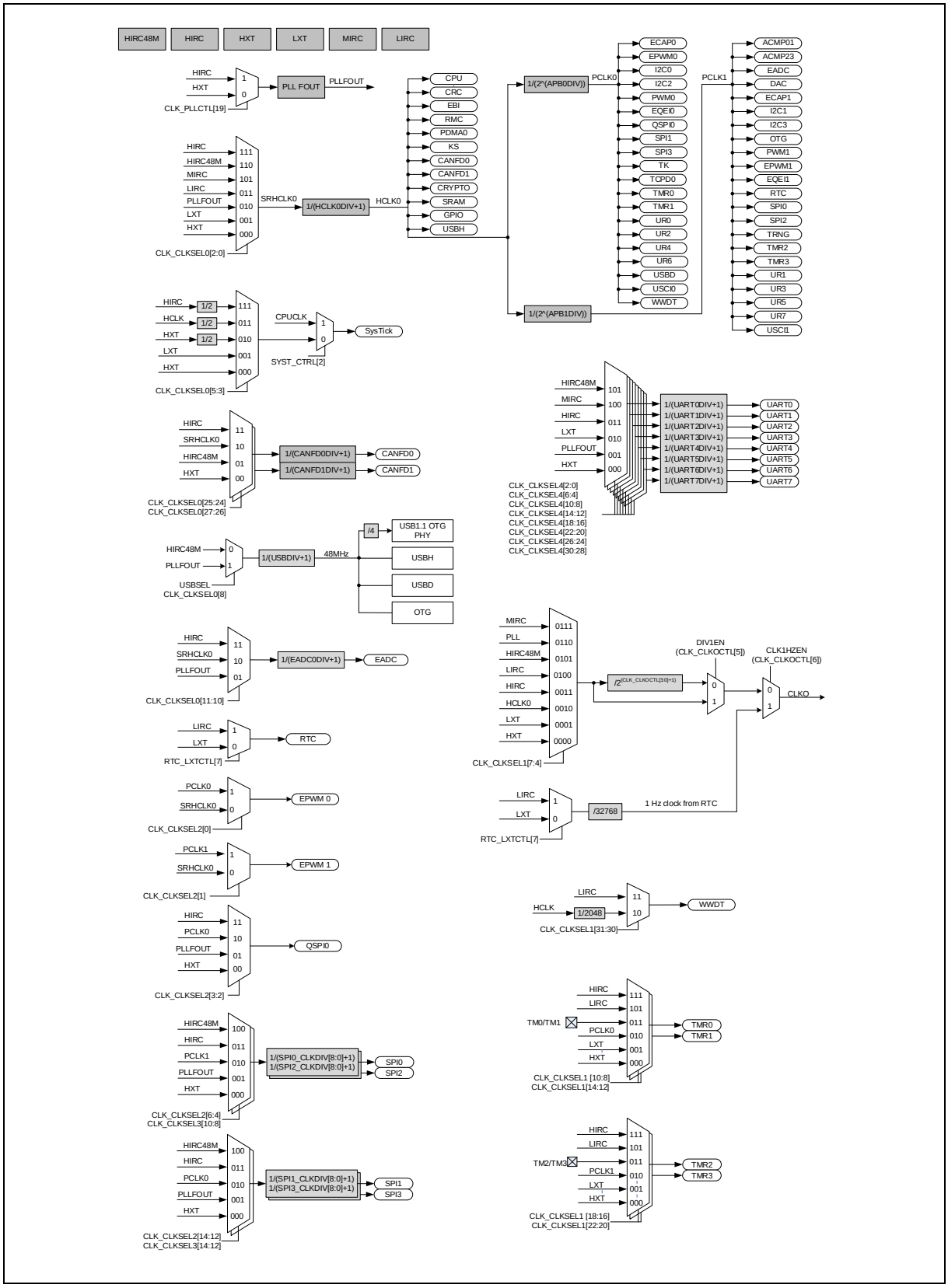


Figure 6.3-1 Clock Generator Global View Diagram

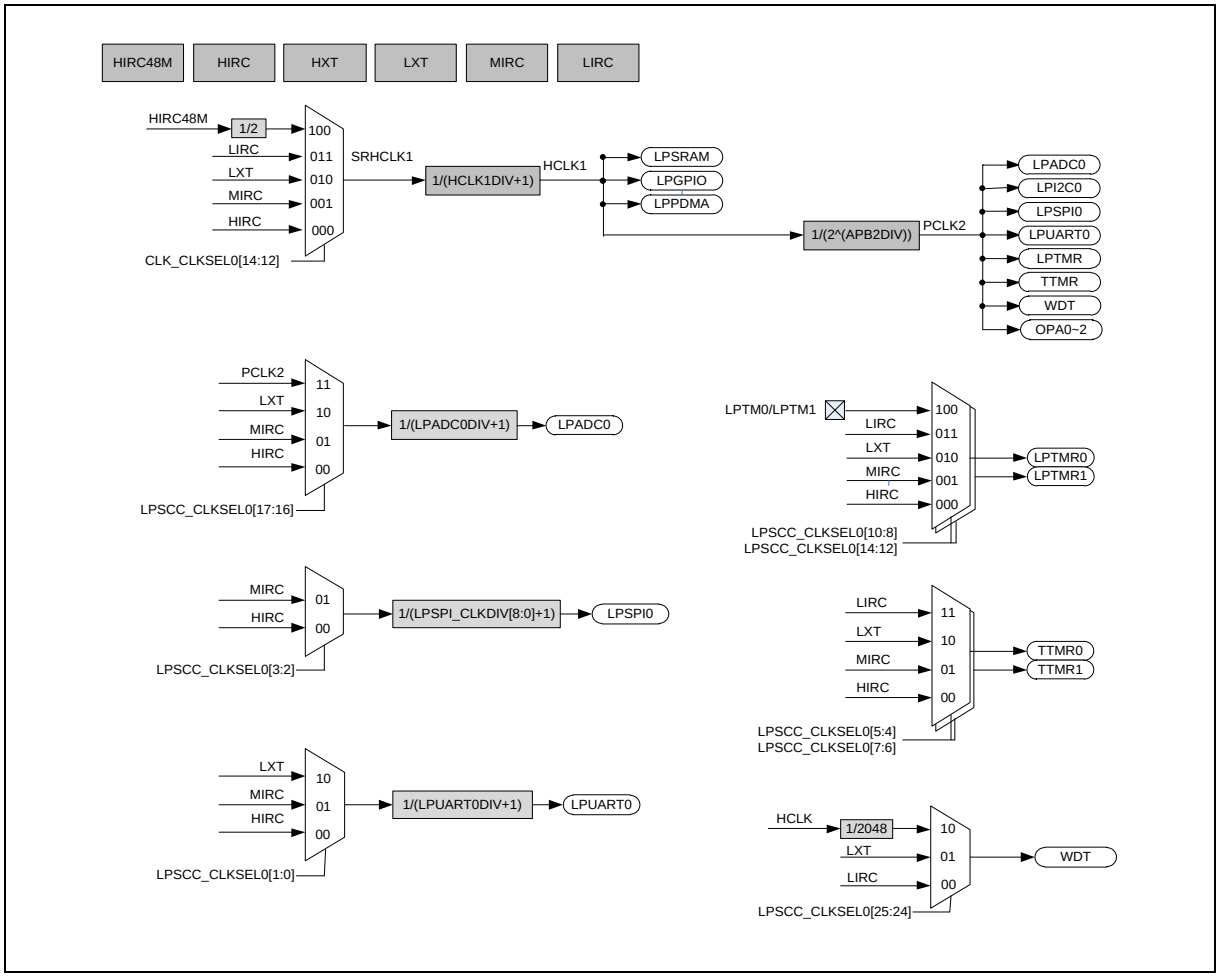


Figure 6.3-2 Low Power Domain Clock Generator Global View Diagram

6.3.2 Clock Generator

The clock generator consists of seven clock sources, which are listed below:

- 32.768 kHz external low speed crystal oscillator (LXT)
- 4~32 MHz external high speed crystal oscillator (HXT)
- Programmable PLL output clock frequency (PLLFOUT), PLL source can be selected from external 4~32 MHz external high speed crystal (HXT) or 12 MHz internal high speed oscillator (HIRC)
- 12 MHz internal high speed RC oscillator (HIRC)
- 32 kHz internal low speed RC oscillator (LIRC)
- 48 MHz internal high speed RC oscillator (HIRC48M)
- 1/2/4/8 MHz internal multiple speed RC oscillator (MIRC)

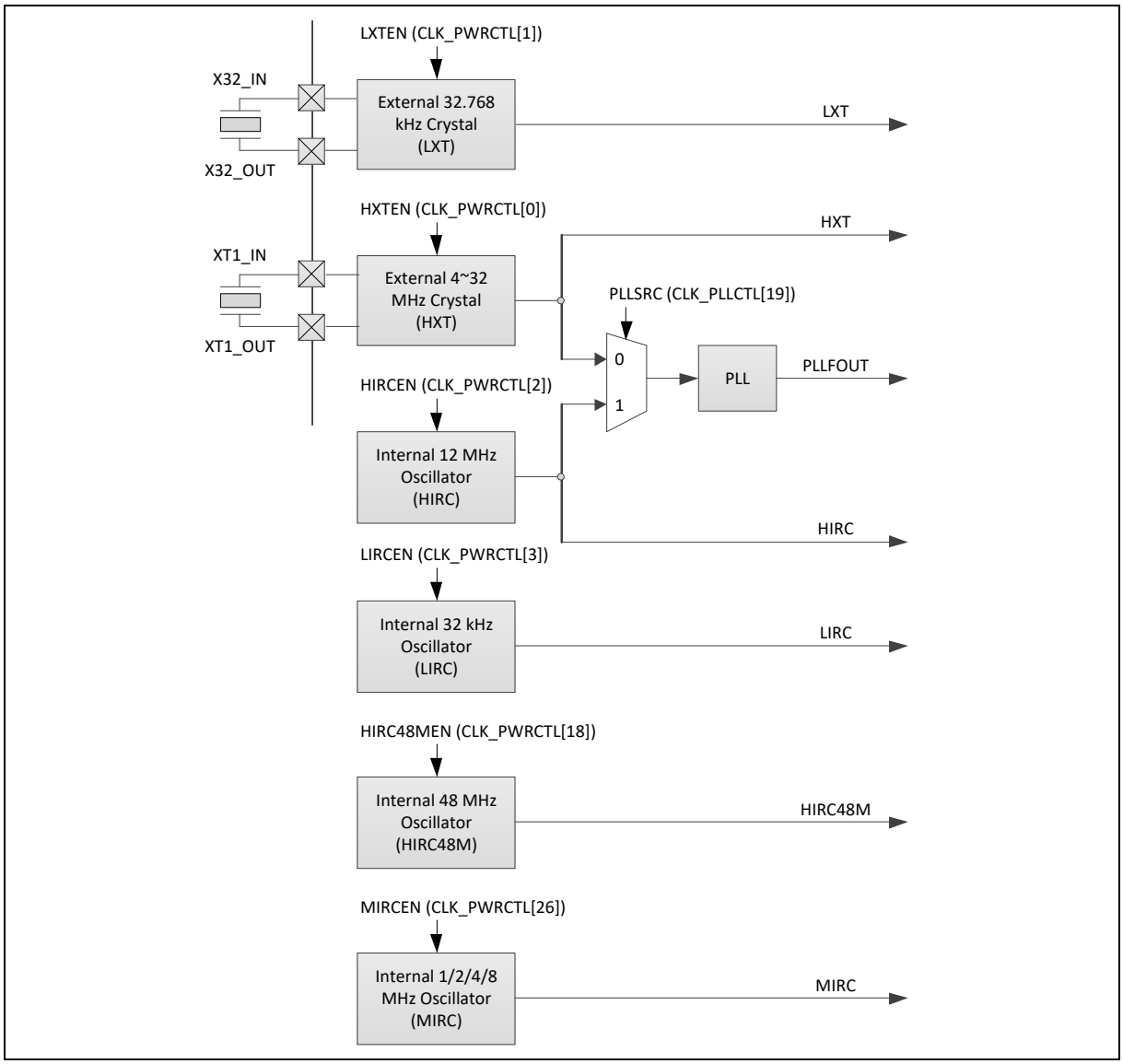


Figure 6.3-3 Clock Generator Block Diagram

6.3.3 Internal Multiple Speed RC Oscillator (MIRC)

The chip has a built-in internal multiple speed RC oscillator (MIRC) for a variable clock frequency with lower power consumption. Setting MIRCFSEL (CLK_PWRCTL[30:28]) can select MIRC output clock frequency to 1/2/4/8 MHz. The frequency setting of MIRC cannot be changed during MIRC is enabled. Before changing the MIRC frequency of MIRC, MIRC should be disabled and check MIRCSTB (CLK_STATUS[5]) flag go low. After changing clock frequency setting, users can enable MIRC and wait MIRCSTB (CLK_STATUS[5]) flag go high, then, MIRC can be used by system or peripherals clock source.

6.3.4 System Clock

The system clock include CPUCLK, HCLK0, HCLK1, PCLK0, PCLK1 and PCLK2. HCLK0 has seven clock sources, which can be selected by HCLK0SEL (CLK_CLKSEL0[2:0]). HCLK1 has five clock sources, which can be selected by HCLK1SEL (CLK_CLKSEL0[7:6]). PCLK0 and PCLK1 are generated from HCLK0, can be HCLK0 divided by 1/2/4/8/16. PCLK2 is generated from HCLK1, can be HCLK1 divided by 1/2/4/8/16. The block diagram is shown in Figure 6.3-4.

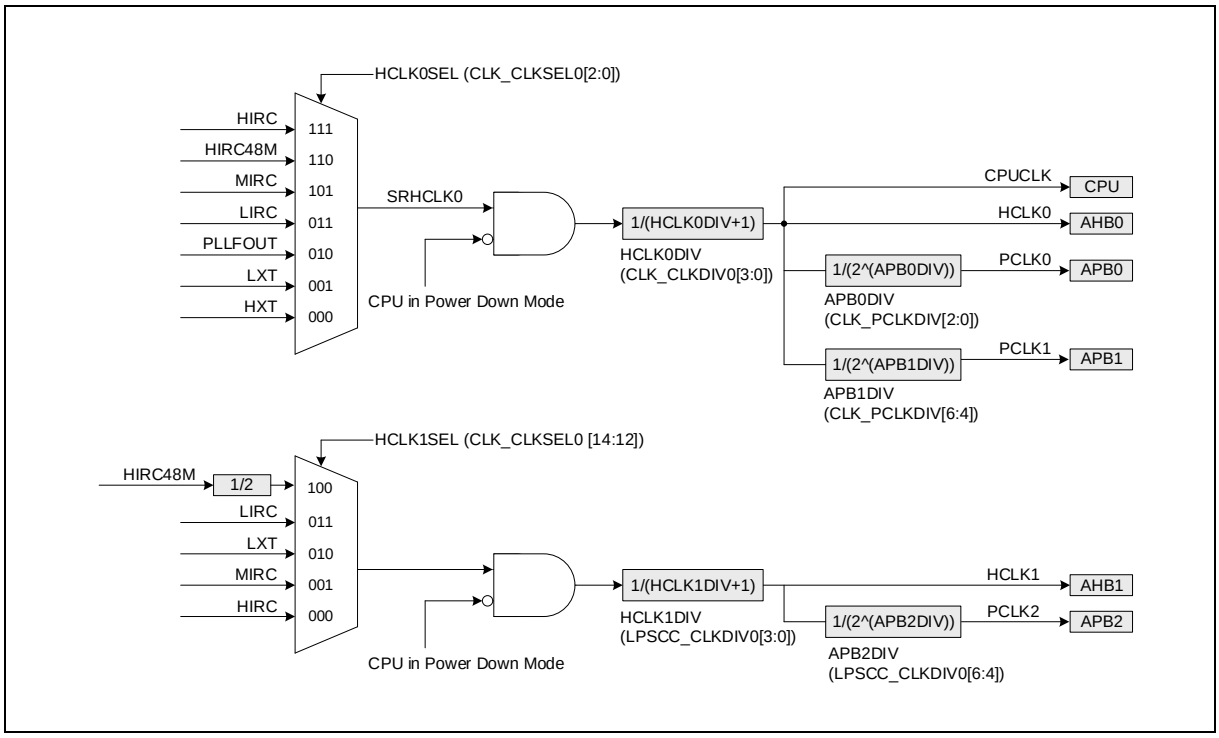


Figure 6.3-4 System Clock Block Diagrams

The clock source of SysTick in Cortex-M23 core can use CPU clock or external clock (SYST_CTL[2]). If using external clock, the SysTick clock (STCLK) has 5 clock sources. The clock source switch depends on the setting of the register STCLKSEL (CLK_CLKSEL0[5:3]). The block diagram is shown in Figure 6.3-5.

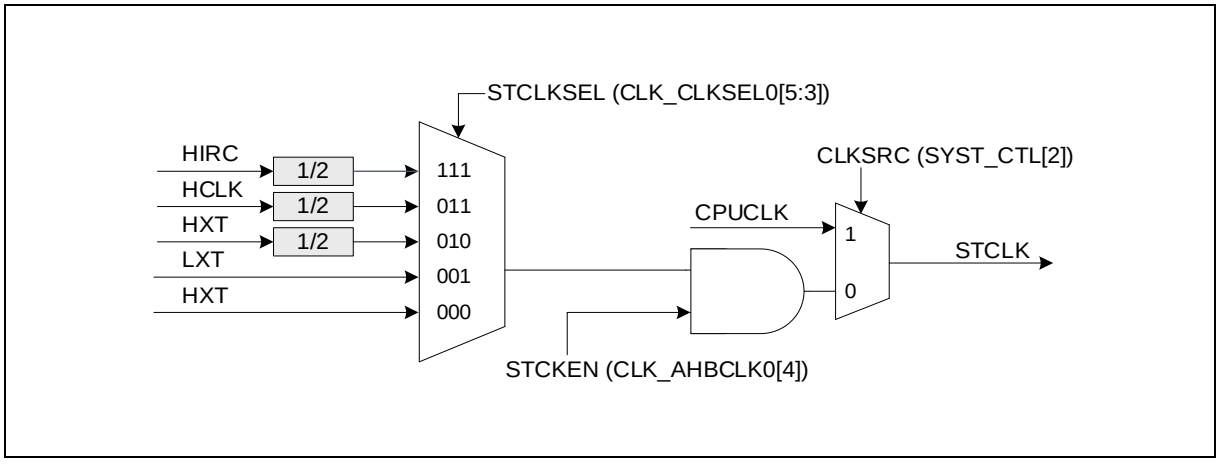


Figure 6.3-5 SysTick Clock Control Block Diagram

6.3.5 Clock Fail Detection

There are two clock fail detectors to observe HXT and LXT clock source and they have individual enable and interrupt control. When HXT detector is enabled, the HIRC clock is enabled automatically. When LXT detector is enabled, the LIRC clock is enabled automatically.

When HXT clock detector is enabled, the system clock will auto switch to HIRC if HXT clock stop being detected on the following condition: system clock source comes from HXT or system clock source comes from PLL with HXT as the input of PLL. If HXT clock stop condition is detected, the HXTFIF (CLK_CLKDSTS[0]) is set to 1 and chip will enter interrupt if HXTFIEN (CLK_CLKDCTL[5]) is set to 1. User can trying to recover HXT by disable HXT and enable HXT again to check if the clock stable bit is set to 1 or not. If HXT clock stable bit is set to 1, it means HXT is recover to oscillate after re-enable action and user can switch system clock to HXT again.

When LXT clock detector is enabled, the system clock will auto switch to LIRC if LXT clock stop is detected and system clock source comes from LXT. If LXT clock stop condition is detected, the LXTFIF (CLK_CLKDSTS[1]) is set to 1 and chip will enter interrupt if LXTFIEN (CLK_CLKDCTL[13]) is set to 1. User can try to recover LXT by disabling LXT and enabling LXT again to check if the clock stable bit is set to 1 or not. If LXT clock stable bit is set to 1, it means LXT is recovered to oscillate after re-enable action and user can switch system clock to LXT again.

Figure 6.3-6 shows The HXT clock stops detection and system clock switches to HIRC procedure

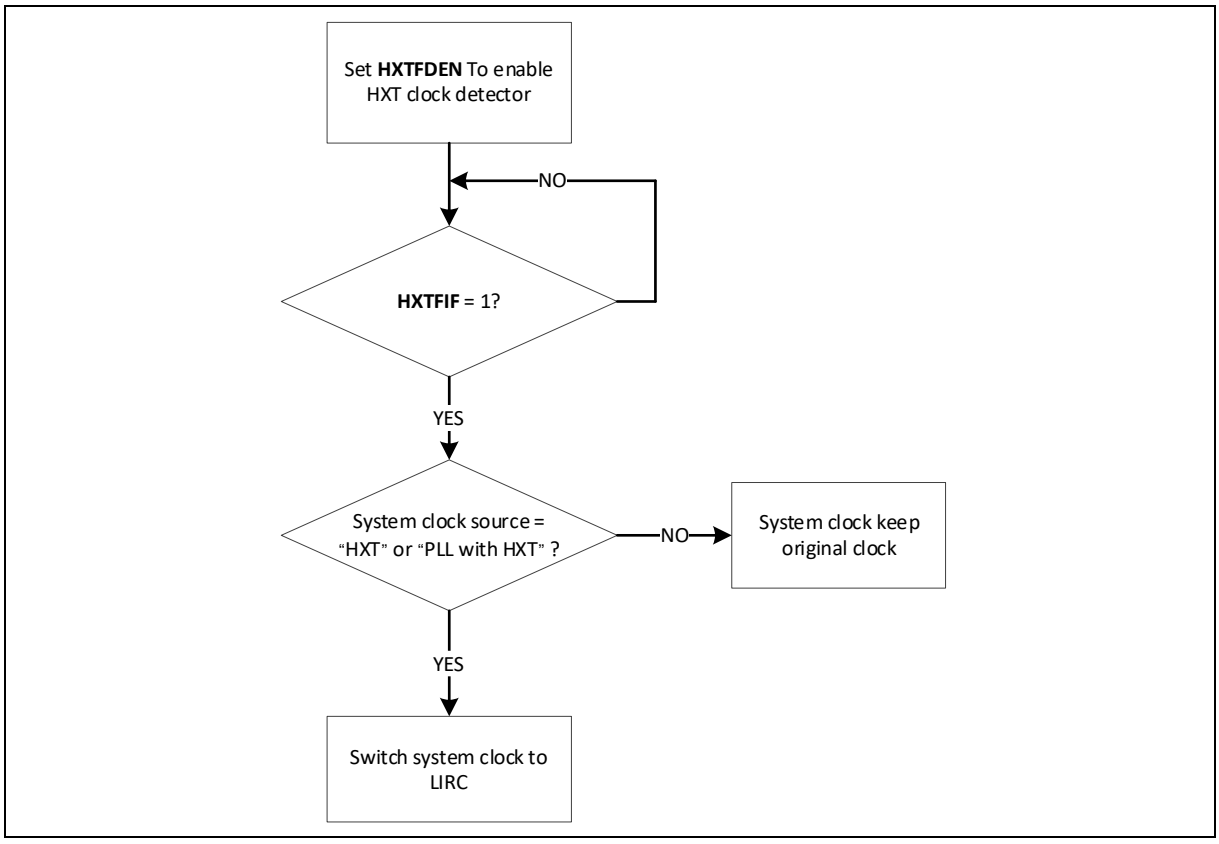


Figure 6.3-6 HXT Stop Protect Procedure

Figure 6.3-7 shows that the LXT clock stops detection and system clock switches to HIRC procedure.

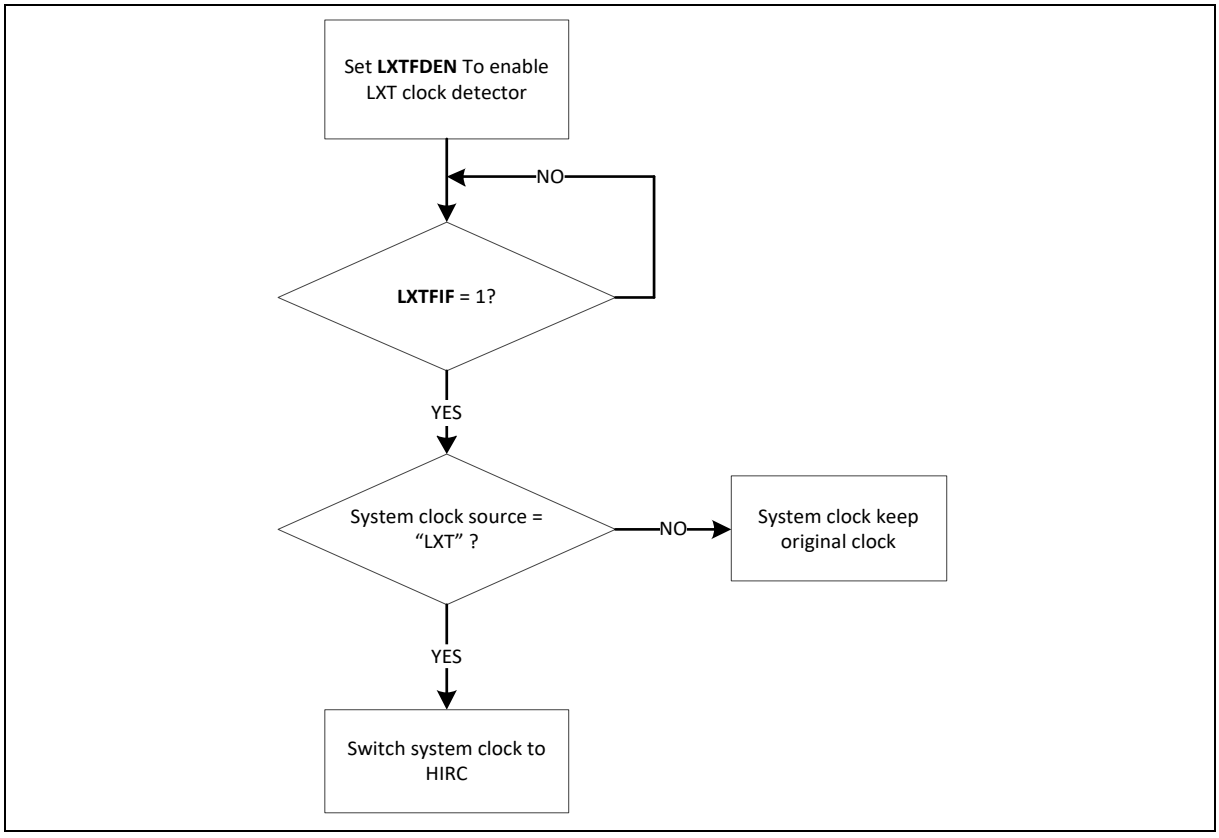


Figure 6.3-7 LXT Stop Protect Procedure

6.3.6 Peripherals Clock

Each peripheral clock has its own clock source selection. Refer to the CLK_CLKSEL0, CLK_CLKSEL1, CLK_CLKSEL2, CLK_CLKSEL3 and CLK_CLKSEL4 register.

6.3.7 Power-down Mode Clock

When entering Power-down mode, the system clocks, some of clock sources and peripheral clocks are disabled. But some of clock sources and peripheral clocks are still active in Power-down mode.

The clocks which are still active in Power-down mode are listed below:

- Clock Generator
  - 32 kHz internal low speed RC oscillator (LIRC) clock
  - 32.768 kHz external low speed crystal oscillator (LXT) clock
  - 12 MHz internal high speed RC oscillator (HIRC) clock if needed
  - 1/2/4/8 MHz internal multiple speed RC oscillator (MIRC) clock if needed
- Peripherals Clock (When the modules adopt LXT or LIRC as clock source)
- Peripherials Clock in which in low power domain

6.3.8 Clock Output

This device is equipped with a power-of-2 frequency divider which is composed of 16 chained divide-by-2 shift registers. One of the 16 shift register outputs selected by a sixteen to one multiplexer is reflected to CLKO function pin. Therefore there are 16 options of power-of-2 divided clocks with the frequency from $F_{in}/2^1$ to $F_{in}/2^{16}$ where F_{in} is input clock frequency to the clock divider.

The output formula is $\mathbf{F_{out} = F_{in}/2^{(N+1)}}$, where F_{in} is the input clock frequency, F_{out} is the clock divider output frequency and N is the 4-bit value in FREQSEL (CLK_CLKOCTL[3:0]).

When writing 1 to CLKOEN (CLK_CLKOCTL[4]), the chained counter starts to count. When writing 0 to CLKOEN (CLK_CLKOCTL[4]), the chained counter continuously runs till divided clock reaches low state and stays in low state. When entering Power-down mode, clock output does not output clock even if the CKO clock source is LXT.

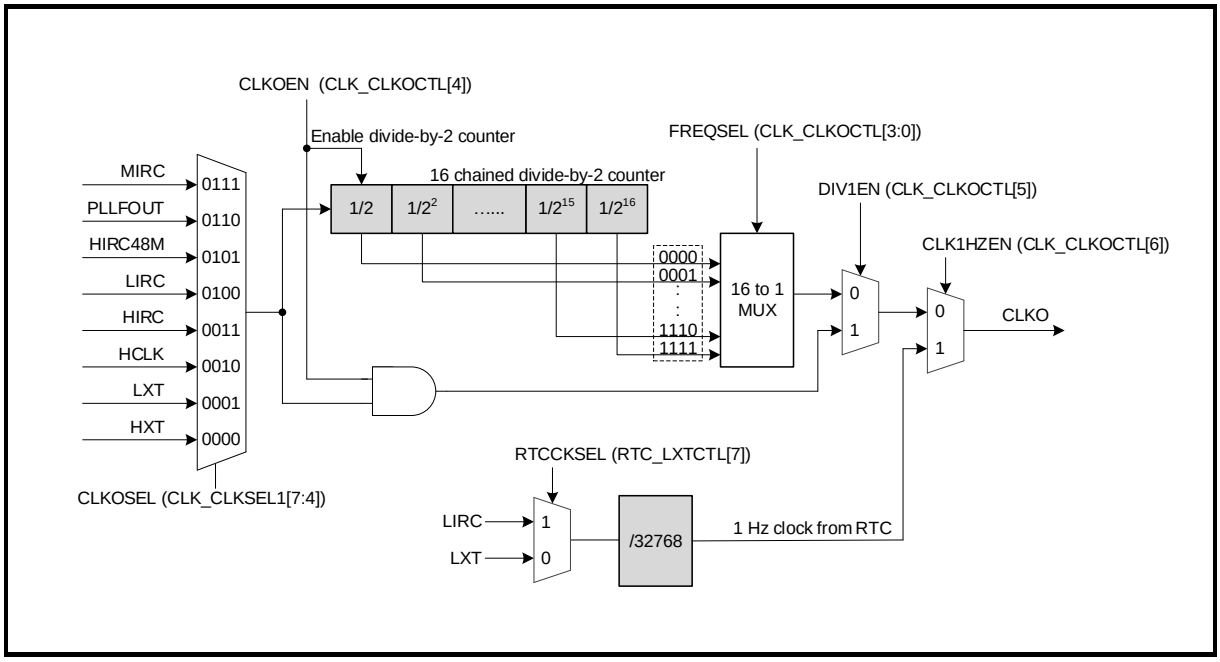


Figure 6.3-8 Clock Output Block Diagram

6.3.9 USB Clock Source

The clock sources of USB 1.1 and 2.0 systems are generated from USB2.0 PHY clock or programmable PLL output. The generated clocks are shown in Figure 6.3-9.

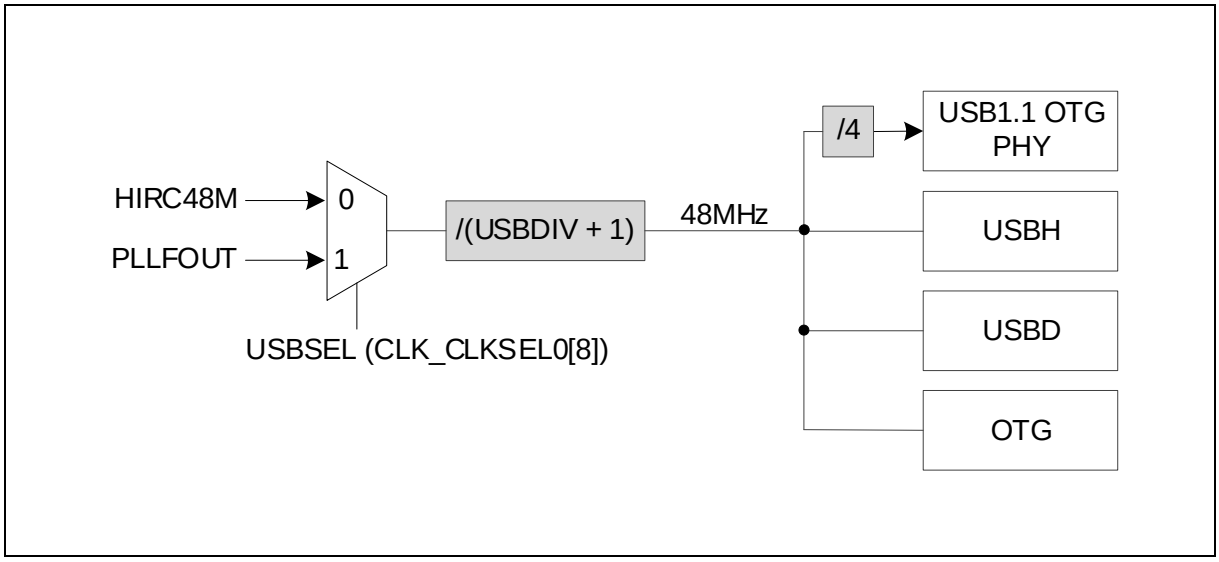


Figure 6.3-9 USB Clock Block Diagram

6.3.10 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
CLK Base Address: **CLK_BA = 0x4000_0200**				
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power-down Control Register	0x2000_X01C
CLK_AHBCLK0	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register 0	0x0000_8004
CLK_APBCLK0	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register 0	0x0000_0000
CLK_APBCLK1	CLK_BA+0x0C	R/W	APB Devices Clock Enable Control Register 1	0x0000_0000
CLK_CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0F00_0D3F
CLK_CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0x8077_7730
CLK_CLKSEL2	CLK_BA+0x18	R/W	Clock Source Select Control Register 2	0x0000_202B
CLK_CLKSEL3	CLK_BA+0x1C	R/W	Clock Source Select Control Register 3	0x0000_2200
CLK_CLKDIV0	CLK_BA+0x20	R/W	Clock Divider Number Register 0	0x0000_0000
CLK_CLKDIV4	CLK_BA+0x30	R/W	Clock Divider Number Register 4	0x0000_0000
CLK_PCLKDIV	CLK_BA+0x34	R/W	APB Clock Divider Register	0x0000_0000
CLK_APBCLK2	CLK_BA+0x38	R/W	APB Devices Clock Enable Control Register 2	0x0000_0000
CLK_CLKDIV5	CLK_BA+0x3C	R/W	Clock Divider Number Register 5	0x0000_0000
CLK_PLLCTL	CLK_BA+0x40	R/W	PLL Control Register	0x0001_0000
CLK_PLLCTL2	CLK_BA+0x44	R/W	PLL Control Register 2	0x0000_8422
CLK_CLKSEL4	CLK_BA+0x48	R/W	Clock Source Select Control Register 4	0x3333_3333
CLK_STATUS	CLK_BA+0x50	R	Clock Status Monitor Register	0x0000_0018
CLK_AHBCLK1	CLK_BA+0x58	R/W	AHB Devices Clock Enable Control Register 1	0x1000_0000
CLK_CLKOCTL	CLK_BA+0x60	R/W	Clock Output Control Register	0x0000_0000
CLK_CLKDCTL	CLK_BA+0x70	R/W	Clock Fail Detector Control Register	0x0000_0000
CLK_CLKDSTS	CLK_BA+0x74	R/W	Clock Fail Detector Status Register	0x0000_0000
CLK_CDUPB	CLK_BA+0x78	R/W	Clock Frequency Range Detector Upper Boundary Register	0x0000_0000
CLK_CDLOWB	CLK_BA+0x7C	R/W	Clock Frequency Range Detector Lower Boundary Register	0x0000_0000
CLK_STOPREQ	CLK_BA+0x80	R/W	Clock Stop Request Register	0x0000_0000
CLK_STOPACK	CLK_BA+0x84	R	Clock Stop Acknowledge Register	0x0000_0000
CLK_PMUCTL	CLK_BA+0x90	R/W	Power Manager Control Register	0x0001_0600
CLK_PMUSTS	CLK_BA+0x94	R/W	Power Manager Status Register	0x0000_0000

CLK_PMUWKCTL	CLK_BA+0x98	R/W	Power Manager Wake-up Control Register	0x0000_0000
CLK_PWDBCTL	CLK_BA+0x9C	R/W	GPIO Pin WKIO De-bounce Control Register	0x0000_0000
CLK_PAPWCTL	CLK_BA+0xA0	R/W	GPA Pin WKIO Control Register	0x0000_0000
CLK_PBPWCTL	CLK_BA+0xA4	R/W	GPB Pin WKIO Control Register	0x0000_0000
CLK_PCPWCTL	CLK_BA+0xA8	R/W	GPC Pin WKIO Control Register	0x0000_0000
CLK_PDPWCTL	CLK_BA+0xAC	R/W	GPD Pin WKIO Control Register	0x0000_0000
CLK_IOPDCTL	CLK_BA+0xB0	R/W	GPIO Power-down Control Register	0x0000_0000
CLK_PMUINTC	CLK_BA+0xC0	R/W	Power Manager Interrupt Control Register	0x0000_0000
CLK_PMUINTS	CLK_BA+0xC4	R/W	Power Manager Interrupt Status Register	0x0000_0000

6.3.11 Register Description

System Power-down Control Register (CLK_PWRCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power-down Control Register	0x2000_X01C

31	30	29	28	27	26	25	24
HXTMD	MIRCFSEL			Reserved	MIRCEN	MIRCSTBS	
23	22	21	20	19	18	17	16
Reserved	HXTGAIN			Reserved	HIRC48MEN	HIRCSTBS	
15	14	13	12	11	10	9	8
HIRC48MSTBS		Reserved	HXTSELTYP	Reserved			
7	6	5	4	3	2	1	0
PDEN	PDWKIF	PDWKIEN	Reserved	LIRCEN	HIRCEN	LXTEN	HXTEN

Bits	Description	
[31]	**HXTMD**	**HXT Bypass Mode (Write Protect)** 0 = HXT work as crystal mode. PF.2 and PF.3 are configured as external high speed crystal (HXT) pins. 1 = HXT works as external clock mode. PF.3 is configured as external clock input pin. **Note 1:** When HXTMD = 1, PF.3 MFP should be set as GPIO mode. The DC characteristic of XT1_IN is the same as GPIO. **Note 2:** This bit is write protected. Refer to the SYS_REGCTL register.
[30:28]	**MIRCFSEL**	**MIRC Frequency Select Bits (Write Protect)** 000 = Internal middle speed RC oscillator frerquency is 1 MHz. 001 = Internal middle speed RC oscillator frerquency is 2 MHz. 010 = Internal middle speed RC oscillator frerquency is 4 MHz. 011 = Internal middle speed RC oscillator frerquency is 8 MHz. Others = Reserved. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[27]	**Reserved**	Reserved.
[26]	**MIRCEN**	**MIRC Enable Bit (Write Protect)** 0 = 1/2/4/8 MHz internal multiple speed RC oscillator (MIRC) Disabled. 1 = 1/2/4/8 MHz internal multiple speed RC oscillator (MIRC) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[25:24]	**MIRCSTBS**	**MIRC Stable Count Select (Write Protect)** 00 = MIRC stable count is 128 clocks. 01 = MIRC stable count is 32 clocks. Others = Reserved. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[23]	**Reserved**	Reserved.

[22:20]	**HXTGAIN**	**HXT Gain Control Bit (Write Protect)** Gain control is used to enlarge the gain of crystal to make sure crystal work normally. If gain control is enabled, crystal will consume more power than gain control off. 000 = HXT frequency is from 1 MHz to 4 MHz. 001 = HXT frequency is from 8 MHz to 12 MHz. 010 = HXT frequency is from 12 MHz to 16 MHz. 011 = HXT frequency is from 16 MHz to 24 MHz. 100 = HXT frequency is from 24 MHz to 32 MHz. 101 = Reserved. 110 = Reserved. 111 = Reserved. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[19]	**Reserved**	Reserved.
[18]	**HIRC48MEN**	**HIRC48M Enable Bit (Write Protect)** 0 = 48 MHz internal high speed RC oscillator (HIRC48M) Disabled. 1 = 48 MHz internal high speed RC oscillator (HIRC48M) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[17:16]	**HIRCSTBS**	**HIRC Stable Count Select (Write Protect)** 00 = HIRC stable count is 64 clocks. 01 = HIRC stable count is 24 clocks. Others = Reserved. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[15:14]	**HIRC48MSTBS**	**HIRC48M Stable Count Select (Write Protect)** 00 = HIRC48M stable count is 1024 clocks. 01 = HIRC48M stable count is 512 clocks. Others = Reserved. **Note:** Thes bits are write protected. Refer to the SYS_REGLCTL register.
[13]	**Reserved**	Reserved.
[12]	**HXTSELTYP**	**HXT Crystal Type Select Bit (Write Protect)** 0 = Select INV type. 1 = Select GM type. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[11:8]	**Reserved**	Reserved.
[7]	**PDEN**	**System Power-down Enable (Write Protect)** When this bit is set to 1, Power-down mode is enabled and chip keeps active till the CPU sleep mode is also active and then the chip enters Power-down mode. When chip wakes up from Power-down mode, this bit is auto cleared. Users need to set this bit again for next Power-down. In Power-down mode, HXT, HIRC and the HIRC48M will be disabled in this mode, but LXT and LIRC are not controlled by Power-down mode. In Power-down mode, the PLL and system clock are disabled, and ignored the clock source selection. The clocks of peripheral are not controlled by Power-down mode, if the peripheral clock source is from LXT or LIRC. 0 = Chip will not enter Power-down mode after CPU sleep command WFI. 1 = Chip enters Power-down mode after CPU sleep command WFI. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6]	**PDWKIF**	**Power-down Mode Wake-up Interrupt Status**

		Set by “Power-down wake-up event”, it indicates that resume from Power-down mode”. The flag is set if any wake-up source is occurred. Refer Power Modes and Wake-up Sources chapter. **Note 1:** Write 1 to clear the bit to 0. **Note 2:** This bit works only if PDWKIEN (CLK_PWRCTL[5]) set to 1.
[5]	**PDWKIEN**	**Power-down Mode Wake-up Interrupt Enable Bit (Write Protect)** 0 = Power-down mode wake-up interrupt Disabled. 1 = Power-down mode wake-up interrupt Enabled. **Note 1:** The interrupt will occur when both PDWKIF and PDWKIEN are high. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[4]	**Reserved**	Reserved.
[3]	**LIRCEN**	**LIRC Enable Bit (Write Protect)** 0 = 32 kHz internal low speed RC oscillator (LIRC) Disabled. 1 = 32 kHz internal low speed RC oscillator (LIRC) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**HIRCEN**	**HIRC Enable Bit (Write Protect)** 0 = 12 MHz internal high speed RC oscillator (HIRC) Disabled. 1 = 12 MHz internal high speed RC oscillator (HIRC) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1]	**LXTEN**	**LXT Enable Bit (Write Protect)** 0 = 32.768 kHz external low speed crystal (LXT) Disabled. 1 = 32.768 kHz external low speed crystal (LXT) Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** When LXT is enabled, PF.4(X32_OUT) and PF.5(X32_IN) must be set as input mode.
[0]	**HXTEN**	**HXT Enable Bit (Write Protect)** 0 = 4~32 MHz external high speed crystal (HXT) Disabled. 1 = 4~32 MHz external high speed crystal (HXT) Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** When HXT is enabled, PF.2(XT1_OUT) and PF.3(XT1_IN) must be set as input mode.

AHB Devices Clock Enable Control Register 0 (CLK_AHBCLK0)

The bits in this register are used to enable/disable clock for system clock, AHB bus devices clock.

Register	Offset	R/W	Description	Reset Value
CLK_AHBCLK0	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register 0	0x0000_8004

31	30	29	28	27	26	25	24
GPHCKEN	GPGCKEN	GPFCKEN	GPECKEN	GPDCKEN	GPCCKEN	GPBCKEN	GPACKEN
23	22	21	20	19	18	17	16
RMCFDIS	Reserved						USBHCKEN
15	14	13	12	11	10	9	8
RMCIDLE	Reserved	KSCKEN	CRPTCKEN	Reserved			
7	6	5	4	3	2	1	0
CRCCKEN	Reserved		STCKEN	EBICKEN	ISPCKEN	PDMA0CKEN	Reserved

Bits	Description	
[31]	GPHCKEN	**GPIOH Clock Enable Bit** 0 = GPIOH port clock Disabled. 1 = GPIOH port clock Enabled.
[30]	GPGCKEN	**GPIOG Clock Enable Bit** 0 = GPIOG port clock Disabled. 1 = GPIOG port clock Enabled.
[29]	GPFCKEN	**GPIOF Clock Enable Bit** 0 = GPIOF port clock Disabled. 1 = GPIOF port clock Enabled.
[28]	GPECKEN	**GPIOE Clock Enable Bit** 0 = GPIOE port clock Disabled. 1 = GPIOE port clock Enabled.
[27]	GPDCKEN	**GPIOD Clock Enable Bit** 0 = GPIOD port clock Disabled. 1 = GPIOD port clock Enabled.
[26]	GPCCKEN	**GPIOC Clock Enable Bit** 0 = GPIOC port clock Disabled. 1 = GPIOC port clock Enabled.
[25]	GPBCKEN	**GPIOB Clock Enable Bit** 0 = GPIOB port clock Disabled. 1 = GPIOB port clock Enabled.
[24]	GPACKEN	**GPIOA Clock Enable Bit** 0 = GPIOA port clock Disabled. 1 = GPIOA port clock Enabled.

[23]	**RMCFDIS**	**RMC Clock Force Disable Bit** 0 = RMC clock Enabled. 1 = RMC clock force Disable to save power. **Note:** User should make sure program no RRAM access during this bit is 1
[22:17]	**Reserved**	Reserved.
[16]	**USBHCKEN**	**USB HOST Controller Clock Enable Bit** 0 = USB HOST peripheral clock Disabled. 1 = USB HOST peripheral clock Enabled.
[15]	**RMCIDLE**	**RRAM Memory Controller Clock Enable Bit in IDLE Mode** 0 = RMC clock Disabled when chip is under IDLE mode. 1 = RMC clock Enabled when chip is under IDLE mode.
[14]	**Reserved**	Reserved.
[13]	**KSCKEN**	**Key Store Clock Enable Bit** 0 = Key Store clock Disabled. 1 = Key Store clock Enabled.
[12]	**CRPTCKEN**	**Cryptographic Accelerator Clock Enable Bit** 0 = Cryptographic Accelerator clock Disabled. 1 = Cryptographic Accelerator clock Enabled.
[11:8]	**Reserved**	Reserved.
[7]	**CRCCKEN**	**CRC Generator Controller Clock Enable Bit** 0 = CRC peripheral clock Disabled. 1 = CRC peripheral clock Enabled.
[6:5]	**Reserved**	Reserved.
[4]	**STCKEN**	**System Tick Clock Enable Bit** 0 = System tick clock Disabled. 1 = System tick clock Enabled.
[3]	**EBICKEN**	**EBI Controller Clock Enable Bit** 0 = EBI peripheral clock Disabled. 1 = EBI peripheral clock Enabled.
[2]	**ISPCKEN**	**RRAM ISP Controller Clock Enable Bit** 0 = RRAM ISP peripheral clock Disabled. 1 = RRAM ISP peripheral clock Enabled.
[1]	**PDMA0CKEN**	**PDMA0 Controller Clock Enable Bit** 0 = PDMA0 peripheral clock Disabled. 1 = PDMA0 peripheral clock Enabled.
[0]	**Reserved**	Reserved.

APB Devices Clock Enable Control Register 0 (CLK_APBCLK0)

The bits in this register are used to enable/disable clock for peripheral controller clocks.

Register	Offset	R/W	Description	Reset Value
CLK_APBCLK0	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
TRNGCKEN	Reserved		EADC0CKEN	USBDCKEN	OTGCKEN	Reserved	
23	22	21	20	19	18	17	16
UART7CKEN	UART6CKEN	UART5CKEN	UART4CKEN	UART3CKEN	UART2CKEN	UART1CKEN	UART0CKEN
15	14	13	12	11	10	9	8
SPI2CKEN	SPI1CKEN	SPI0CKEN	QSPI0CKEN	I2C3CKEN	I2C2CKEN	I2C1CKEN	I2C0CKEN
7	6	5	4	3	2	1	0
ACMP01CKEN	CLKOCKEN	TMR3CKEN	TMR2CKEN	TMR1CKEN	TMR0CKEN	RTCCKEN	Reserved

Bits	Description	
[31]	**TRNGCKEN**	**TRNG Clock Enable Bit** 0 = TRNG clock Disabled. 1 = TRNG clock Enabled.
[30:29]	**Reserved**	Reserved.
[28]	**EADC0CKEN**	**EADC0 Clock Enable Bit** 0 = EADC0 clock Disabled. 1 = EADC0 clock Enabled.
[27]	**USBDCKEN**	**USB Device Clock Enable Bit** 0 = USB device clock Disabled. 1 = USB device clock Enabled.
[26]	**OTGCKEN**	**USB OTG Clock Enable Bit** 0 = USB OTG clock Disabled. 1 = USB OTG clock Enabled.
[25:24]	**Reserved**	Reserved.
[23]	**UART7CKEN**	**UART7 Clock Enable Bit** 0 = UART7 clock Disabled. 1 = UART7 clock Enabled.
[22]	**UART6CKEN**	**UART6 Clock Enable Bit** 0 = UART6 clock Disabled. 1 = UART6 clock Enabled.
[21]	**UART5CKEN**	**UART5 Clock Enable Bit** 0 = UART5 clock Disabled. 1 = UART5 clock Enabled.
[20]	**UART4CKEN**	**UART4 Clock Enable Bit**

		0 = UART4 clock Disabled. 1 = UART4 clock Enabled.
[19]	**UART3CKEN**	**UART3 Clock Enable Bit** 0 = UART3 clock Disabled. 1 = UART3 clock Enabled.
[18]	**UART2CKEN**	**UART2 Clock Enable Bit** 0 = UART2 clock Disabled. 1 = UART2 clock Enabled.
[17]	**UART1CKEN**	**UART1 Clock Enable Bit** 0 = UART1 clock Disabled. 1 = UART1 clock Enabled.
[16]	**UART0CKEN**	**UART0 Clock Enable Bit** 0 = UART0 clock Disabled. 1 = UART0 clock Enabled.
[15]	**SPI2CKEN**	**SPI2 Clock Enable Bit** 0 = SPI2 clock Disabled. 1 = SPI2 clock Enabled.
[14]	**SPI1CKEN**	**SPI1 Clock Enable Bit** 0 = SPI1 clock Disabled. 1 = SPI1 clock Enabled.
[13]	**SPI0CKEN**	**SPI0 Clock Enable Bit** 0 = SPI0 clock Disabled. 1 = SPI0 clock Enabled.
[12]	**QSPI0CKEN**	**QSPI0 Clock Enable Bit** 0 = QSPI0 clock Disabled. 1 = QSPI0 clock Enabled.
[11]	**I2C3CKEN**	**I2C3 Clock Enable Bit** 0 = I2C3 clock Disabled. 1 = I2C3 clock Enabled.
[10]	**I2C2CKEN**	**I2C2 Clock Enable Bit** 0 = I2C2 clock Disabled. 1 = I2C2 clock Enabled.
[9]	**I2C1CKEN**	**I2C1 Clock Enable Bit** 0 = I2C1 clock Disabled. 1 = I2C1 clock Enabled.
[8]	**I2C0CKEN**	**I2C0 Clock Enable Bit** 0 = I2C0 clock Disabled. 1 = I2C0 clock Enabled.
[7]	**ACMP01CKEN**	**ACMP01 Clock Enable Bit** 0 = ACMP01 clock Disabled. 1 = ACMP01 clock Enabled.
[6]	**CLKOCKEN**	**CLKO Clock Enable Bit** 0 = CLKO clock Disabled. 1 = CLKO clock Enabled.

[5]	**TMR3CKEN**	**Timer3 Clock Enable Bit** 0 = Timer3 clock Disabled. 1 = Timer3 clock Enabled.
[4]	**TMR2CKEN**	**Timer2 Clock Enable Bit** 0 = Timer2 clock Disabled. 1 = Timer2 clock Enabled.
[3]	**TMR1CKEN**	**Timer1 Clock Enable Bit** 0 = Timer1 clock Disabled. 1 = Timer1 clock Enabled.
[2]	**TMR0CKEN**	**Timer0 Clock Enable Bit** 0 = Timer0 clock Disabled. 1 = Timer0 clock Enabled.
[1]	**RTCCKEN**	**Real-time-clock APB Interface Clock Enable Bit** This bit is used to control the RTC APB clock only. The RTC peripheral clock source is selected from RTCCKSEL (RTC_LXTCTL[7]). It can be selected to external low speed crystal oscillator (LXT) or internal low speed RC oscillator (LIRC). 0 = RTC clock Disabled. 1 = RTC clock Enabled.
[0]	**Reserved**	Reserved.

APB Devices Clock Enable Control Register 1 (CLK_APBCLK1)

The bits in this register are used to enable/disable clock for peripheral controller clocks.

Register	Offset	R/W	Description	Reset Value
CLK_APBCLK1	CLK_BA+0x0C	R/W	APB Devices Clock Enable Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				ECAP1CKEN	ECAP0CKEN	TKCKEN	Reserved
23	22	21	20	19	18	17	16
EQEI1CKEN	EQEI0CKEN	Reserved				EPWM1CKEN	EPWM0CKEN
15	14	13	12	11	10	9	8
Reserved			DACEN	WWDTCKEN	Reserved	USCI1CKEN	USCI0CKEN
7	6	5	4	3	2	1	0
Reserved	SPI3CKEN	Reserved					

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**ECAP1CKEN**	**ECAP1 Clock Enable Bit** 0 = ECAP1 clock Disabled. 1 = ECAP1 clock Enabled.
[26]	**ECAP0CKEN**	**ECAP0 Clock Enable Bit** 0 = ECAP0 clock Disabled. 1 = ECAP0 clock Enabled.
[25]	**TKCKEN**	**TK Clock Enable Bit** 0 = TK clock Disabled. 1 = TK clock Enabled.
[24]	**Reserved**	Reserved.
[23]	**EQEI1CKEN**	**EQEI1 Clock Enable Bit** 0 = EQEI1 clock Disabled. 1 = EQEI1 clock Enabled.
[22]	**EQEI0CKEN**	**EQEI0 Clock Enable Bit** 0 = EQEI0 clock Disabled. 1 = EQEI0 clock Enabled.
[21:18]	**Reserved**	Reserved.
[17]	**EPWM1CKEN**	**EPWM1 Clock Enable Bit** 0 = EPWM1 clock Disabled. 1 = EPWM1 clock Enabled.
[16]	**EPWM0CKEN**	**EPWM0 Clock Enable Bit** 0 = EPWM0 clock Disabled. 1 = EPWM0 clock Enabled.

[15:13]	**Reserved**	Reserved.
[12]	**DACEN**	**DAC Clock Enable Bit** 0 = DAC clock Disabled. 1 = DAC clock Enabled.
[11]	**WWDTCKEN**	**Window Watchdog Timer Clock Enable Bit (Write Protect)** 0 = Window Watchdog timer clock Disabled. 1 = Window Watchdog timer clock Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[10]	**Reserved**	Reserved.
[9]	**USCI1CKEN**	**USCI1 Clock Enable Bit** 0 = USCI1 clock Disabled. 1 = USCI1 clock Enabled.
[8]	**USCI0CKEN**	**USCI0 Clock Enable Bit** 0 = USCI0 clock Disabled. 1 = USCI0 clock Enabled.
[7]	**Reserved**	Reserved.
[6]	**SPI3CKEN**	**SPI3 Clock Enable Bit** 0 = SPI3 clock Disabled. 1 = SPI3 clock Enabled.
[5:0]	**Reserved**	Reserved.

Clock Source Select Control Register 0 (CLK_CLKSEL0)

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0F00_0D3F

31	30	29	28	27	26	25	24
Reserved				CANFD1SEL		CANFD0SEL	
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	HCLK1SEL			EADC0SEL		Reserved	USBSEL
7	6	5	4	3	2	1	0
Reserved		STCLKSEL			HCLK0SEL		

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:26]	**CANFD1SEL**	**CANFD1 Clock Source Selection (Write Protect)** 00 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 01 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). 10 = Clock source from HCLK0. 11 = Clock source from 12 MHz internal high speed RC oscillator (HIRC).
[25:24]	**CANFD0SEL**	**CANFD0 Clock Source Selection (Write Protect)** 00 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 01 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). 10 = Clock source from SRHCLK0. 11 = Clock source from 12 MHz internal high speed RC oscillator (HIRC).
[23:15]	**Reserved**	Reserved.
[14:12]	**HCLK1SEL**	**HCLK1 Clock Source Selection (Write Protect)** Before clock switching, the related clock sources (both pre-select and new-select) must be turned on. 000 = Clock source from HIRC. 001 = Clock source from MIRC. 010 = Clock source from LXT. 011 = Clock source from LIRC. 100 = Clock source from HIRC48M_divider2. Others = Reserved. **Note:** HIRC48M_divider2 does not work at NPD3/NPD4/NPD5/SPD0~2/DPD0~1 Power-down mode; change to another clk source if needed.
[11:10]	**EADC0SEL**	**EADC0 Clock Source Selection (Write Protect)** 01 = Clock source from PLL. 10 = Clock source from SRHCLK0. 11 = Clock source from HIRC.

		Others = Reserved.
[9]	**Reserved**	Reserved.
[8]	**USBSEL**	**USB Clock Source Selection (Write Protect)** 0 = Clock source from HIRC48M. 1 = Clock source from PLL**.**
[7:6]	**Reserved**	Reserved.
[5:3]	**STCLKSEL**	**Cortex-M4 SysTick Clock Source Selection (Write Protect)** If SYST_CTRL[2]=0, SysTick uses listed clock source below. 000 = Clock source from HXT. 001 = Clock source from LXT. 010 = Clock source from HXT/2. 011 = Clock source from HCLK/2. 111 = Clock source from HIRC/2. Others = Reserved. **Note:** if SysTick clock source is not from CPU clock (i.e. SYST_CTRL[2] = 0), SysTick clock source must be less than or equal to CPU clock divided by 2.
[2:0]	**HCLK0SEL**	**HCLK0 Clock Source Selection (Write Protect)** Before clock switching, the related clock sources (both pre-select and new-select) must be turned on. 000 = Clock source from HXT. 001 = Clock source from LXT. 010 = Clock source from PLL. 011 = Clock source from LIRC. 101 = Clock source from MIRC. 110 = Clock source from HIRC48M. 111 = Clock source from HIRC. Other = Reserved. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Clock Source Select Control Register 1 (CLK_CLKSEL1)

Before clock switching, the related clock sources (pre-selected and newly-selected) must be turned on.

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0x8077_7730

31	30	29	28	27	26	25	24
WWDTSEL		Reserved					
23	22	21	20	19	18	17	16
Reserved	TMR3SEL			Reserved	TMR2SEL		
15	14	13	12	11	10	9	8
Reserved	TMR1SEL			Reserved	TMR0SEL		
7	6	5	4	3	2	1	0
CLKOSEL				Reserved			

Bits	Description	
[31:30]	**WWDTSEL**	**Window Watchdog Timer Clock Source Selection** 10 = Clock source from HCLK/2048. 11 = Clock source from 32 kHz internal low speed RC oscillator (LIRC). Others = Reserved.
[29:23]	**Reserved**	Reserved.
[22:20]	**TMR3SEL**	**TIMER3 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 010 = Clock source from PCLK1. 011 = Clock source from external clock TM3 pin. 101 = Clock source from 32 kHz internal low speed RC oscillator (LIRC). 111 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). Others = Reserved.
[19]	**Reserved**	Reserved.
[18:16]	**TMR2SEL**	**TIMER2 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 010 = Clock source from PCLK1. 011 = Clock source from external clock TM2 pin. 101 = Clock source from 32 kHz internal low speed RC oscillator (LIRC). 111 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). Others = Reserved.
[15]	**Reserved**	Reserved.
[14:12]	**TMR1SEL**	**TIMER1 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT).

		001 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 010 = Clock source from PCLK0. 011 = Clock source from external clock TM1 pin. 101 = Clock source from 32 kHz internal low speed RC oscillator (LIRC). 111 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**TMR0SEL**	**TIMER0 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 010 = Clock source from PCLK0. 011 = Clock source from external clock TM0 pin. 101 = Clock source from 32 kHz internal low speed RC oscillator (LIRC). 111 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). Others = Reserved.
[7:4]	**CLKOSEL**	**Clock Divider Clock Source Selection** 0000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 0001 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 0010 = Clock source from HCLK0. 0011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 0100 = Clock source from 32 kHz internal high speed RC oscillator (LIRC). 0101 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M).. 0110 = Clock source from PLL. 0111 = Clock source from 1/2/4/8 MHz internal miltiple speed RC oscillator (MIRC). others = Reserved. **Note:** CLKOSEL can only be changed under CLKOCKEN = 0.
[3:0]	**Reserved**	Reserved.

Clock Source Select Control Register 2 (CLK_CLKSEL2)

Before clock switching, the related clock sources (pre-select and new-select) must be turned on.

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL2	CLK_BA+0x18	R/W	Clock Source Select Control Register 2	0x0000_202B

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	SPI1SEL			Reserved			
7	6	5	4	3	2	1	0
TKSEL	SPI0SEL			QSPI0SEL		EPWM1SEL	EPWM0SEL

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**SPI1SEL**	**SPI1 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from PCLK0. 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). Others = Reserved.
[11:8]	**Reserved**	Reserved.
[7]	**TKSEL**	**TK Clock Source Selection** 00 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 01 = Clock source from 1/2/4/8 MHz internal middle speed RC oscillator (MIRC). **Note:** The TKSEL cannot be changed when TK is operating. Used should make TK disable before change TKSEL, and reset TK after change TKSEL.
[6:4]	**SPI0SEL**	**SPI0 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from PCLK1. 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). Others = Reserved.
[3:2]	**QSPI0SEL**	**QSPI0 Clock Source Selection** 00 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 01 = Clock source from PLL. 10 = Clock source from PCLK0. 11 = Clock source from 12 MHz internal high speed RC oscillator (HIRC).

[1]	**EPWM1SEL**	**EPWM1 Clock Source Selection** The peripheral clock source of EPWM1 is defined by EPWM1SEL. 0 = Clock source from SRHCLK0. 1 = Clock source from PCLK1.
[0]	**EPWM0SEL**	**EPWM0 Clock Source Selection** The peripheral clock source of EPWM0 is defined by EPWM0SEL. 0 = Clock source from SRHCLK0. 1 = Clock source from PCLK0.

Clock Source Select Control Register 3 (CLK_CLKSEL3)

Before clock switching, the related clock sources (pre-select and new-select) must be turned on.

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL3	CLK_BA+0x1C	R/W	Clock Source Select Control Register 3	0x0000_2200

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	SPI3SEL			Reserved	SPI2SEL		
7	6	5	4	3	2	1	0
PWM1SEL	PWM0SEL	Reserved					

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**SPI3SEL**	**SPI3 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from PCLK0. 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**SPI2SEL**	**SPI2 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from PCLK1. 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). Others = Reserved.
[7]	**PWM1SEL**	**PWM1 Clock Source Selection** The peripheral clock source of PWM1 is defined by PWM1SEL. 0 = Clock source from SRHCLK0. 1 = Clock source from PCLK1.
[6]	**PWM0SEL**	**PWM0 Clock Source Selection** The peripheral clock source of PWM0 is defined by PWM0SEL. 0 = Clock source from SRHCLK0. 1 = Clock source from PCLK0.
[5:0]	**Reserved**	Reserved.

Clock Divider Number Register 0 (CLK_CLKDIV0)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV0	CLK_BA+0x20	R/W	Clock Divider Number Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
EADC0DIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
UART1DIV				**UART0DIV**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBDIV				**HCLK0DIV**			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**EADC0DIV**	**EADC0 Clock Divide Number from EADC0 Clock Source** EADC0 clock frequency = (EADC0 clock source frequency) / (EADC0DIV + 1).
[15:12]	**UART1DIV**	**UART1 Clock Divide Number from UART1 Clock Source** UART1 clock frequency = (UART1 clock source frequency) / (UART1DIV + 1).
[11:8]	**UART0DIV**	**UART0 Clock Divide Number from UART0 Clock Source** UART0 clock frequency = (UART0 clock source frequency) / (UART0DIV + 1).
[7:4]	**USBDIV**	**USB Clock Divide Number from USB Clock** USB clock frequency = (USB clock source frequency) / (USBDIV + 1).
[3:0]	**HCLK0DIV**	**HCLK0 Clock Divide Number from HCLK0 Clock Source** HCLK0 clock frequency = (HCLK0 clock source frequency) / (HCLK0DIV + 1). **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Clock Divider Number Register 4 (CLK_CLKDIV4)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV4	CLK_BA+0x30	R/W	Clock Divider Number Register 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
UART7DIV				UART6DIV			
15	14	13	12	11	10	9	8
UART5DIV				UART4DIV			
7	6	5	4	3	2	1	0
UART3DIV				UART2DIV			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**UART7DIV**	**UART7 Clock Divide Number from UART7 Clock Source** UART7 clock frequency = (UART7 clock source frequency) / (UART7DIV + 1).
[19:16]	**UART6DIV**	**UART6 Clock Divide Number from UART6 Clock Source** UART6 clock frequency = (UART6 clock source frequency) / (UART6DIV + 1).
[15:12]	**UART5DIV**	**UART5 Clock Divide Number from UART5 Clock Source** UART5 clock frequency = (UART5 clock source frequency) / (UART5DIV + 1).
[11:8]	**UART4DIV**	**UART4 Clock Divide Number from UART4 Clock Source** UART4 clock frequency = (UART4 clock source frequency) / (UART4DIV + 1).
[7:4]	**UART3DIV**	**UART3 Clock Divide Number from UART3 Clock Source** UART3 clock frequency = (UART3 clock source frequency) / (UART3DIV + 1).
[3:0]	**UART2DIV**	**UART2 Clock Divide Number from UART2 Clock Source** UART2 clock frequency = (UART2 clock source frequency) / (UART2DIV + 1).

APB Clock Divider Register (CLK_PCLKDIV)

Register	Offset	R/W	Description	Reset Value
CLK_PCLKDIV	CLK_BA+0x34	R/W	APB Clock Divider Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	APB1DIV			Reserved	APB0DIV		

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:4]	**APB1DIV**	**APB1 Clock Divider** APB1 clock can be divided from HCLK0. 000 = PCLK1 frequency is HCLK0. 001 = PCLK1 frequency is 1/2 HCLK0. 010 = PCLK1 frequency is 1/4 HCLK0. 011 = PCLK1 frequency is 1/8 HCLK0. 100 = PCLK1 frequency is 1/16 HCLK0. Others = Reserved.
[3]	**Reserved**	Reserved.
[2:0]	**APB0DIV**	**APB0 Clock Divider** APB0 clock can be divided from HCLK0. 000 = PCLK0 frequency is HCLK0. 001= PCLK0 frequency is 1/2 HCLK0. 010 = PCLK0 frequency is 1/4 HCLK0. 011 = PCLK0 frequency is 1/8 HCLK0. 100 = PCLK0 frequency is 1/16 HCLK0. Others = Reserved.

APB Devices Clock Enable Control Register 2 (CLK_APBCLK2)

The bits in this register are used to enable/disable clock for system clock, AHB bus devices clock.

Register	Offset	R/W	Description	Reset Value
CLK_APBCLK2	CLK_BA+0x38	R/W	APB Devices Clock Enable Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
UTCPD0CKEN	Reserved					PWM1CKEN	PWM0CKEN
7	6	5	4	3	2	1	0
ACMP2CKEN	Reserved						

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**UTCPD0CKEN**	**UTCPD0 Clock Enable Bit** 0 = UTCPD0 clock Disabled. 1 = UTCPD0 clock Enabled.
[14:10]	**Reserved**	Reserved.
[9]	**PWM1CKEN**	**PWM1 Clock Enable Bit** 0 = PWM1 clock Disabled. 1 = PWM1 clock Enabled.
[8]	**PWM0CKEN**	**PWM0 Clock Enable Bit** 0 = PWM0 clock Disabled. 1 = PWM0 clock Enabled.
[7]	**ACMP2CKEN**	**ACMP2 Clock Enable Bit** 0 = ACMP2 clock Disabled. 1 = ACMP2 clock Enabled.
[6:0]	**Reserved**	Reserved.

Clock Divider Number Register 5 (CLK_CLKDIV5)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV5	CLK_BA+0x3C	R/W	Clock Divider Number Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CANFD1DIV				**CANFD0DIV**			

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:4]	**CANFD1DIV**	**CANFD1 Clock Divide Number from CANFD1 Clock Source** CANFD1 clock frequency = (CANFD1 clock source frequency) / (CANFD1DIV + 1).
[3:0]	**CANFD0DIV**	**CANFD0 Clock Divide Number from CANFD0 Clock Source** CANFD0 clock frequency = (CANFD0 clock source frequency) / (CANFD0DIV + 1).

PLL Control Register (CLK_PLLCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PLLCTL	CLK_BA+0x40	R/W	PLL Control Register	0x0001_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
STBSEL	Reserved			PLLSRC	OE	BP	PD
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**STBSEL**	**PLL Stable Counter Selection (Write Protect)** 0 = PLL stable time is 2500 PLL source clock (suitable for source clock equal to or less than 12 MHz). 1 = PLL stable time is 7000 PLL source clock (suitable for source clock is larger than 12 MHz). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[22:20]	**Reserved**	Reserved.
[19]	**PLLSRC**	**PLL Source Clock Selection (Write Protect)** 0 = PLL source clock from 4~32 MHz external high-speed crystal oscillator (HXT). 1 = PLL source clock from 12 MHz internal high-speed oscillator (HIRC). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[18]	**OE**	**PLL FOUT Enable Control (Write Protect)** 0 = PLL FOUT Enabled. 1 = PLL FOUT is fixed low. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[17]	**BP**	**PLL Bypass Control (Write Protect)** 0 = PLL is in normal mode (default). 1 = PLL clock output is same as PLL input clock FIN. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[16]	**PD**	**Power-down Mode (Write Protect)** 0 = PLL is in normal mode. 1 = PLL is in Power-down mode (default). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[15:0]	**Reserved**	Reserved.

PLL2 Control Register (CLK_PLLCTL2)

Register	Offset	R/W	Description	Reset Value
CLK_PLLCTL2	CLK_BA+0x44	R/W	PLL Control Register 2	0x0000_8422

31	30	29	28	27	26	25	24
Reserved				**FRDIV**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
FRDIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
OUTDIV		**INDIV**					**FBDIV**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FBDIV							

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**FRDIV**	**PLL Fractional Divider Control (Write Protect)** Refer to the formulas below the table. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[15:14]	**OUTDIV**	**PLL Output Divider Control (Write Protect)** Refer to the formulas below the table. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[13:9]	**INDIV**	**PLL Input Divider Control (Write Protect)** Refer to the formulas below the table. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[8:0]	**FBDIV**	**PLL Feedback Divider Control (Write Protect)** Refer to the formulas below the table. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Output Clock Frequency formula:

$$FREF = FIN \times \frac{1}{NR}$$

$$FVCO = FIN \times \frac{2*(NF.x)}{NR}$$

$$FOUT = FIN \times \frac{2*(NF.x)}{NR} \times \frac{1}{NO}$$

where FREF is the comparison frequency for the PFD (phase frequency detector).

For proper operation in normal mode, the following constraints must be satisfied:

1 MHz ≤ FREF ≤ 8 MHz

144 MHz ≤ FVCO ≤ 500 MHz

36 MHz ≤ FOUT ≤ 144 MHz

Symbol	Description
FOUT	Output Clock Frequency
FIN	Input (Reference) Clock Frequency
NR	Input Divider (INDIV + 1)
NF	Feedback Divider (FBDIV + 2) **Note:** NF min value is 12 and max value is 255
x	Fractional Value (FRDIV/4096)
NO	OUTDIV = “00” : NO = 1 OUTDIV = “01” : NO = 2 OUTDIV = “10” : NO = 2 OUTDIV = “11” : NO = 4

Table 6.3-1 Symbol definition of PLL Output Frequency formula

Clock Source Select Control Register 4 (CLK_CLKSEL4)

Before clock switching, the related clock sources (pre-select and new-select) must be turned on.

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL4	CLK_BA+0x48	R/W	Clock Source Select Control Register 4	0x3333_3333

31	30	29	28	27	26	25	24
Reserved	UART7SEL			Reserved	UART6SEL		
23	22	21	20	19	18	17	16
Reserved	UART5SEL			Reserved	UART4SEL		
15	14	13	12	11	10	9	8
Reserved	UART3SEL			Reserved	UART2SEL		
7	6	5	4	3	2	1	0
Reserved	UART1SEL			Reserved	UART0SEL		

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**UART7SEL**	**UART7 Clock Source Selection** 000 = Clock source from HXT. 001 = Clock source from PLL. 010 = Clock source from LXT. 011 = Clock source from HIRC. 100 = Clock source from MIRC. 101 = Clock source from HIRC48M. Others = Reserved.
[27]	**Reserved**	Reserved.
[26:24]	**UART6SEL**	**UART6 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 1/2/4/8 MHz internal middle speed RC oscillator (MIRC). 101 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). Others = Reserved.
[23]	**Reserved**	Reserved.
[22:20]	**UART5SEL**	**UART5 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 1/2/4/8 MHz internal middle speed RC oscillator (MIRC). 101 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M).

		Others = Reserved.
[19]	**Reserved**	Reserved.
[18:16]	**UART4SEL**	**UART4 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 1/2/4/8 MHz internal middle speed RC oscillator (MIRC). 101 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). Others = Reserved.
[15]	**Reserved**	Reserved.
[14:12]	**UART3SEL**	**UART3 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 1/2/4/8 MHz internal middle speed RC oscillator (MIRC). 101 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**UART2SEL**	**UART2 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 1/2/4/8 MHz internal middle speed RC oscillator (MIRC). 101 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). Others = Reserved.
[7]	**Reserved**	Reserved.
[6:4]	**UART1SEL**	**UART1 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 1/2/4/8 MHz internal middle speed RC oscillator (MIRC). 101 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M). Others = Reserved.
[3]	**Reserved**	Reserved.
[2:0]	**UART0SEL**	**UART0 Clock Source Selection** 000 = Clock source from 4~32 MHz external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from 32.768 kHz external low speed crystal oscillator (LXT). 011 = Clock source from 12 MHz internal high speed RC oscillator (HIRC). 100 = Clock source from 1/2/4/8 MHz internal middle speed RC oscillator (MIRC). 101 = Clock source from 48 MHz internal high speed RC oscillator (HIRC48M).

		Others = Reserved.

Clock Status Monitor Register (CLK_STATUS)

The bits in this register are used to monitor if the chip clock source is stable or not, and whether the clock switch is failed.

Register	Offset	R/W	Description	Reset Value
CLK_STATUS	CLK_BA+0x50	R	Clock Status Monitor Register	0x0000_0018

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CLKSFAIL	HIRC48MSTB	MIRCSTB	HIRCSTB	LIRCSTB	PLLSTB	LXTSTB	HXTSTB

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**CLKSFAIL**	**Clock Switching Fail Flag (Read Only)** This bit is updated when software switches system clock source. If switch target clock is stable, this bit will be set to 0. If switch target clock is not stable, this bit will be set to 1. 0 = Clock switching success. 1 = Clock switching failure. **Note:** This bit is read only. After selected clock source is stable, hardware will switch system clock to selected clock automatically, and CLKSFAIL will be cleared automatically by hardware.
[6]	**HIRC48MSTB**	**HIRC48M Clock Source Stable Flag (Read Only)** 0 = 48 MHz internal high speed RC oscillator (HIRC48M) clock is not stable or disabled. 1 = 48 MHz internal high speed RC oscillator (HIRC48M) clock is stable and enabled.
[5]	**MIRCSTB**	**MIRC Clock Source Stable Flag (Read Only)** 0 = 1/2/4/8 MHz internal high speed RC oscillator (MIRC) clock is not stable or disabled. 1 = 1/2/4/8 MHz internal high speed RC oscillator (MIRC) clock is stable and enabled.
[4]	**HIRCSTB**	**HIRC Clock Source Stable Flag (Read Only)** 0 = 12 MHz internal high speed RC oscillator (HIRC) clock is not stable or disabled. 1 = 12 MHz internal high speed RC oscillator (HIRC) clock is stable and enabled.
[3]	**LIRCSTB**	**LIRC Clock Source Stable Flag (Read Only)** 0 = 32 kHz internal low speed RC oscillator (LIRC) clock is not stable or disabled. 1 = 32 kHz internal low speed RC oscillator (LIRC) clock is stable and enabled.
[2]	**PLLSTB**	**Internal PLL Clock Source Stable Flag (Read Only)** 0 = Internal PLL clock is not stable or disabled. 1 = Internal PLL clock is stable and enabled.
[1]	**LXTSTB**	**LXT Clock Source Stable Flag (Read Only)**

		0 = 32.768 kHz external low speed crystal oscillator (LXT) clock is not stable or disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) clock is stabled and enabled.
[0]	**HXTSTB**	**HXT Clock Source Stable Flag (Read Only)** 0 = 4~32 MHz external high speed crystal oscillator (HXT) clock is not stable or disabled. 1 = 4~32 MHz external high speed crystal oscillator (HXT) clock is stable and enabled.

AHB Devices Clock Enable Control Register 1 (CLK_AHBCLK1)

The bits in this register are used to enable/disable clock for AHB bus peripheral controller clocks.

Register	Offset	R/W	Description	Reset Value
CLK_AHBCLK1	CLK_BA+0x58	R/W	AHB Devices Clock Enable Control Register 1	0x1000_0000

31	30	29	28	27	26	25	24
Reserved			HCLK1EN	Reserved			
23	22	21	20	19	18	17	16
Reserved		CANFD1CKEN	CANFD0CKEN	Reserved		CANRAM1EN	CANRAM0EN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**HCLK1EN**	**HCLK1 Clock Enable Bit** 0 = HCLK1 clock Disabled. 1 = HCLK1 clock Enabled.
[27:22]	**Reserved**	Reserved.
[21]	**CANFD1CKEN**	**CANFD1 Clock Enable Bit** 0 = CANFD1 clock Disabled. 1 = CANFD1 clock Enabled.
[20]	**CANFD0CKEN**	**CANFD0 Clock Enable Bit** 0 = CANFD0 clock Disabled. 1 = CANFD0 clock Enabled.
[19:18]	**Reserved**	Reserved.
[17]	**CANRAM1EN**	**CANFD1 Message SRAM Clock Enable Bit** 0 = CANFD1 Message SRAM clock Disabled. 1 = CANFD1 Message SRAM clock Enabled.
[16]	**CANRAM0EN**	**CANFD0 Message SRAM Clock Enable Bit** 0 = CANFD0 Message SRAM clock Disabled. 1 = CANFD0 Message SRAM clock Enabled.
[15:0]	**Reserved**	Reserved.

Clock Output Control Register (CLK_CLKOCTL)

Register	Offset	R/W	Description	Reset Value
CLK_CLKOCTL	CLK_BA+0x60	R/W	Clock Output Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	CLK1HZEN	DIV1EN	CLKOEN	FREQSEL			

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**CLK1HZEN**	**Clock Output 1Hz Enable Bit** 0 = 1 Hz clock output for 32.768 kHz frequency compensation Disabled. 1 = 1 Hz clock output for 32.768 kHz frequency compensation Enabled.
[5]	**DIV1EN**	**Clock Output Divide One Enable Bit** 0 = Clock Output will output clock with source frequency divided by FREQSEL. 1 = Clock Output will output clock with source frequency.
[4]	**CLKOEN**	**Clock Output Enable Bit** 0 = Clock Output function Disabled. 1 = Clock Output function Enabled.
[3:0]	**FREQSEL**	**Clock Output Frequency Selection** The formula of output frequency is $\mathbf{F_{out} = F_{in}/2^{(N+1)}}$. F_{in} is the input clock frequency. F_{out} is the frequency of divider output clock. N is the 4-bit value of FREQSEL[3:0].

Clock Fail Detector Control Register (CLK_CLKDCTL)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDCTL	CLK_BA+0x70	R/W	Clock Fail Detector Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						HXTFQIEN	HXTFQDEN
15	14	13	12	11	10	9	8
Reserved		LXTFIEN	LXTFDEN	Reserved			
7	6	5	4	3	2	1	0
Reserved		HXTFIEN	HXTFDEN	Reserved			

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**HXTFQIEN**	**HXT Clock Frequency Range Detector Interrupt Enable Bit** 0 = 4~32 MHz external high speed crystal oscillator (HXT) clock frequency range detector fail interrupt Disabled. 1 = 4~32 MHz external high speed crystal oscillator (HXT) clock frequency range detector fail interrupt Enabled.
[16]	**HXTFQDEN**	**HXT Clock Frequency Range Detector Enable Bit** 0 = 4~32 MHz external high speed crystal oscillator (HXT) clock frequency range detector Disabled. 1 = 4~32 MHz external high speed crystal oscillator (HXT) clock frequency range detector Enabled. **Note:** HIRC must be enabled and stabled before enabling HXT clock frequency monitor.
[15:14]	**Reserved**	Reserved.
[13]	**LXTFIEN**	**LXT Clock Fail Interrupt Enable Bit** 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock fail interrupt Disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) clock fail interrupt Enabled.
[12]	**LXTFDEN**	**LXT Clock Fail Detector Enable Bit** 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock fail detector Disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) clock fail detector Enabled.
[11:6]	**Reserved**	Reserved.
[5]	**HXTFIEN**	**HXT Clock Fail Interrupt Enable Bit** 0 = 4~32 MHz external high speed crystal oscillator (HXT) clock fail interrupt Disabled. 1 = 4~32 MHz external high speed crystal oscillator (HXT) clock fail interrupt Enabled.
[4]	**HXTFDEN**	**HXT Clock Fail Detector Enable Bit** 0 = 4~32 MHz external high speed crystal oscillator (HXT) clock fail detector Disabled. 1 = 4~32 MHz external high speed crystal oscillator (HXT) clock fail detector Enabled.

[3:0]	**Reserved**	Reserved.

Clock Fail Detector Status Register (CLK_CLKDSTS)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDSTS	CLK_BA+0x74	R/W	Clock Fail Detector Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							HXTFQIF
7	6	5	4	3	2	1	0
Reserved						LXTFIF	HXTFIF

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**HXTFQIF**	**HXT Clock Frequency Range Detector Interrupt Flag (Write Protect)** 0 = 4~32 MHz external high speed crystal oscillator (HXT) clock frequency is normal. 1 = 4~32 MHz external high speed crystal oscillator (HXT) clock frequency is abnormal. **Note 1:** Write 1 to clear the bit to 0. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7:2]	**Reserved**	Reserved.
[1]	**LXTFIF**	**LXT Clock Fail Interrupt Flag (Write Protect)** 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock is normal. 1 = 32.768 kHz external low speed crystal oscillator (LXT) stops. **Note 1:** Write 1 to clear the bit to 0. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**HXTFIF**	**HXT Clock Fail Interrupt Flag (Write Protect)** 0 = 4~32 MHz external high speed crystal oscillator (HXT) clock is normal. 1 = 4~32 MHz external high speed crystal oscillator (HXT) clock stops. **Note 1:** Write 1 to clear the bit to 0. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.

Clock Frequency Range Detector Upper Boundary Register (CLK_CDUPB)

Register	Offset	R/W	Description	Reset Value
CLK_CDUPB	CLK_BA+0x78	R/W	Clock Frequency Range Detector Upper Boundary Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					UPERBD		
7	6	5	4	3	2	1	0
UPERBD							

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**UPERBD**	**HXT Clock Frequency Range Detector Upper Boundary Value** The bits define the maximum value of frequency range detector window. When HXT frequency higher than this maximum frequency value, the HXT Clock Frequency Range Detector Interrupt Flag HXTFQIF(CLK_CLKDSTS[8]) will set to 1.

Clock Frequency Range Detector Lower Boundary Register (CLK_CDLOWB)

Register	Offset	R/W	Description	Reset Value
CLK_CDLOWB	CLK_BA+0x7C	R/W	Clock Frequency Range Detector Lower Boundary Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**LOWERBD**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LOWERBD							

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**LOWERBD**	**HXT Clock Frequency Range Detector Lower Boundary Value** The bits define the minimum value of frequency range detector window. When HXT frequency lower than this minimum frequency value, the HXT Clock Frequency Range Detector Interrupt Flag HXTFQIF(CLK_CLKDSTS[8]) will set to 1.

Clock Stop Request Register (CLK_STOPREQ)

Register	Offset	R/W	Description	Reset Value
CLK_STOPREQ	CLK_BA+0x80	R/W	Clock Stop Request Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CANFD1REQ	CANFD0REQ

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CANFD1REQ**	**CANFD1 Clock Stop Request** Set this bit and check the CANFD1ACK(CLK_STOPACK[1]) =1, then CANFD1 engine clock stopped. Set this bit = 0 when CANFD1 clock is not stopped by this bit (default).
[0]	**CANFD0REQ**	**CANFD0 Clock Stop Request** Set this bit and check the CANFD0ACK(CLK_STOPACK[0]) =1, then CANFD0 engine clock stopped. Set this bit = 0 when CANFD0 clock is not stopped by this bit (default).

Clock Stop Acknowledge Register (CLK_STOPACK)

Register	Offset	R/W	Description	Reset Value
CLK_STOPACK	CLK_BA+0x84	R	Clock Stop Acknowledge Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CANFD1ACK	CANFD0ACK

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CANFD1ACK**	**CANFD1 Clock Stop Acknowledge** When this bit =1, CANFD1 engine clock stopped by setting CANFD1REQ(CLK_STOPREQ[1]).
[0]	**CANFD0ACK**	**CANFD0 Clock Stop Acknowledge** When this bit =1, CANFD0 engine clock stopped by setting CANFD0REQ(CLK_STOPREQ[0]).

Power Manager Control Register (CLK_PMUCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PMUCTL	CLK_BA+0x90	R/W	Power Manager Control Register	0x0001_0600

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			CARETDIS	Reserved			LSRETSEL
15	14	13	12	11	10	9	8
Reserved					SRETSEL		
7	6	5	4	3	2	1	0
Reserved				PDMSEL			

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**CARETDIS**	**Cache RAM Retention Disable Bit (Write Protect)** 0 = Cache RAM retention when chip enters NPD3/NPD4/NPD5 mode. 1 = Cache RAM shut down when chip enters NPD3/NPD4/NPD5 mode. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[19:17]	**Reserved**	Reserved.
[16]	**LSRETSEL**	**LPSRAM Retention Range Select Bit (Write Protect)** Select LPSRAM retention range when chip enters NPD3/NPD4/NPD5/SPD0~2/DPD0~1 mode. 0 = LPSRAM shut down. 1 = 8K LPSRAM retention (0x2800_0000 - 0x2800_1FFF) when chip enters NPD3/NPD4/NPD5/SPD0~2 mode. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** This bit should be set 1 if LPSRAM is active in NPD3/NPD4/NPD5 mode.
[15:11]	**Reserved**	Reserved.
[10:8]	**SRETSEL**	**SRAM Retention Range Select Bit (Write Protect)** Select SRAM retention range when chip enters NPD3/NPD4/NPD5/SPD0~2 mode. 000 = No SRAM retention. 001 = 8K SRAM retention when chip enters NPD3/NPD4/NPD5/SPD0~2 mode. 010 = 24K SRAM retention when chip enters NPD3/NPD4/NPD5/SPD0~2 mode. 011 = 40K SRAM retention when chip enters NPD3/NPD4/NPD5/SPD0~2 mode. 100 = 72K SRAM retention when chip enters NPD3/NPD4/NPD5/SPD0~2 mode. 101 = 104K SRAM retention when chip enters NPD3/NPD4/NPD5/SPD0~2 mode. 110 = 168K SRAM retention when chip enters NPD3/NPD4/NPD5/SPD0~2 mode. Others = Reserved. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** Users should make sure program stack is within SRAM retention range before chip enters NPD3/NPD4/NPD5/SPD0~2 mode.

[7:4]	**Reserved**	Reserved.
[3:0]	**PDMSEL**	**Power-down Mode Selection (Write Protect)** These bits control chip power-down mode grade selection when CPU execute WFI/WFE instruction. 0000 = Normal Power-down mode 0 is selected (NPD0). 0001 = Normal Power-down mode 1 is selected (NPD1). 0010 = Normal Power-down mode 2 is selected (NPD2). 0011 = Normal Power-down mode 3 is selected (NPD3). 0100 = Normal Power-down mode 4 is selected (NPD4). 0101 = Normal Power-down mode 5 is selected (NPD5). 1000 = Standby Power-down mode 0 is selected (SPD0). 1001 = Standby Power-down mode 1 is selected (SPD1). 1010 = Standby Power-down mode 2 is selected (SPD2). 1100 = Deep Power-down mode 0 is selected (DPD0). 1101 = Deep Power-down mode 1 is selected (DPD1). Other = Reserved. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Power Manager Status Register (CLK_PMUSTS)

Register	Offset	R/W	Description	Reset Value
CLK_PMUSTS	CLK_BA+0x94	R/W	Power Manager Status Register	0x0000_0000

31	30	29	28	27	26	25	24
CLRWK	Reserved			GPDWK1	GPCWK1	GPBWK1	GPAWK1
23	22	21	20	19	18	17	16
Reserved					ACMPWK2	ACMPWK1	ACMPWK0
15	14	13	12	11	10	9	8
RSTWK	Reserved	BODWK	LVRWK	GPDWK0	GPCWK0	GPBWK0	GPAWK0
7	6	5	4	3	2	1	0
WKPIN5	WKPIN4	WKPIN3	WKPIN2	WKPIN1	RTCWK	TMRWK	WKPIN0

Bits	Description	
[31]	**CLRWK**	**Clear Wake-up Flag** 0 = Not cleared. 1 = Clear all wake-up flags. **Note1:** This bit is auto cleared by hardware. **Note2:** If DISAUTOC (CLK_PMUCLK[31])=0, all wake-up flags are auto cleared when chip enters NPD3/NPD4/NPD5/SPD0~2/DPD0~1 Power-down mode.
[30:28]	**Reserved**	Reserved.
[27]	**GPDWK1**	**GPD Wake-up 1 Flag (Read Only)** This flag indicates that wake-up of chip from NPD0~5 or Standby Power-down mode was requested by a transition of selected one GPD group pins. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD0~5/ SPD0~2 mode.
[26]	**GPCWK1**	**GPC Wake-up 1 Flag (Read Only)** This flag indicates that wake-up of chip from NPD0~5 or Standby Power-down mode was requested by a transition of selected one GPC group pins. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD0~5/SPD0~2 mode.
[25]	**GPBWK1**	**GPB Wake-up 1 Flag (Read Only)** This flag indicates that wake-up of chip from NPD0~5 or Standby Power-down mode was requested by a transition of selected one GPB group pins. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD0~5/SPD0~2 mode.
[24]	**GPAWK1**	**GPA Wake-up 1 Flag (Read Only)** This flag indicates that wake-up of chip from NPD0~5 or Standby Power-down mode was requested by a transition of selected one GPA group pins. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD0~5/SPD0~2 mode.
[23:19]	**Reserved**	Reserved.
[18]	**ACMPWK2**	**ACMP2 Wake-up Flag (Read Only)**

		This flag indicates that wakeup of device from Standby Power-down mode (SPD0~2) was requested with a ACMP2 transition. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering SPD0~2 mode.
[17]	**ACMPWK1**	**ACMP1 Wake-up Flag (Read Only)** This flag indicates that wakeup of device from Standby Power-down mode (SPD0~2) was requested with a ACMP1 transition. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering SPD0~2 mode.
[16]	**ACMPWK0**	**ACMP0 Wake-up Flag (Read Only)** This flag indicates that wakeup of device from Standby Power-down mode (SPD0~2) was requested with a ACMP0 transition. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering SPD0~2 mode.
[15]	**RSTWK**	**RST pin Wake-up Flag (Read Only)** This flag indicates that wakeup of device from NPD3/NPD4/NPD5 or Deep Power-down mode (DPD0~1) or Standby Power-down mode (SPD0~2) was requested with a RST pin trigger happened. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD3/NPD4/NPD5/SPD0~2/DPD0~1 mode.
[14]	**Reserved**	Reserved.
[13]	**BODWK**	**BOD Wake-up Flag (Read Only)** This flag indicates that wakeup of device from NPD3/NPD4/NPD5 or Standby Power-down mode (SPD0~2) was requested with a BOD happened. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD3/NPD4/NPD5/SPD0~2 mode.
[12]	**LVRWK**	**LVR Wake-up Flag (Read Only)** This flag indicates that wakeup of device from NPD3/NPD4/NPD5 or Standby Power-down mode (SPD0~2) was requested with a LVR happened. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD3/NPD4/NPD5/SPD0~2 mode.
[11]	**GPDWK0**	**GPD Wake-up 0 Flag (Read Only)** This flag indicates that wake-up of chip from NPD0~5 or Standby Power-down mode (SPD0~2) was requested by a transition of selected one GPD group pins. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD0~5/SPD0~2 mode.
[10]	**GPCWK0**	**GPC Wake-up 0 Flag (Read Only)** This flag indicates that wake-up of chip from NPD0~5 or Standby Power-down mode (SPD0~2) was requested by a transition of selected one GPC group pins. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD0~5/SPD0~2 mode.
[9]	**GPBWK0**	**GPB Wake-up 0 Flag (Read Only)** This flag indicates that wake-up of chip from NPD0~5 or Standby Power-down mode (SPD0~2) was requested by a transition of selected one GPB group pins. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD0~5/SPD0~2 mode.
[8]	**GPAWK0**	**GPA Wake-up 0 Flag (Read Only)** This flag indicates that wake-up of chip from NPD0~5 or Standby Power-down mode (SPD0~2) was requested by a transition of selected one GPA group pins. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD0~5/SPD0~2 mode.

[7]	**WKPIN5**	**Pin5 Wake-up Flag(Read Only)** This flag indicates that wake-up of chip from Deep Power-down mode was requested by a transition of the WAKEUP pin (PA.12). **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering DPD0/DPD1 mode.
[6]	**WKPIN4**	**Pin4 Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Deep Power-down mode was requested by a transition of the WAKEUP pin (PF.6). **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering DPD0/DPD1 mode.
[5]	**WKPIN3**	**Pin3 Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Deep Power-down mode was requested by a transition of the WAKEUP pin (PB.12). **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering DPD0/DPD1 mode.
[4]	**WKPIN2**	**Pin2 Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Deep Power-down mode was requested by a transition of the WAKEUP pin (PB.2). **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering DPD0/DPD1 mode.
[3]	**WKPIN1**	**Pin1 Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Deep Power-down mode was requested by a transition of the WAKEUP pin (PB.0). **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering DPD0/DPD1 mode.
[2]	**RTCWK**	**RTC Wake-up Flag (Read Only)** This flag indicates that wakeup of device from NPD3/NPD4/NPD5 or Deep Power-down mode (DPD0~1) or Standby Power-down (SPD0~2) mode was requested with a RTC alarm, tick time or tamper happened. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD3/NPD4/NPD5/ SPD0~2/DPD0~1 mode.
[1]	**TMRWK**	**Timer Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from NPD0~5 or Deep Power-down mode (DPD0~1) or Standby Power-down (SPD0~2) mode was requested by wakeup timer time-out. **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering NPD0~5/SPD0~2/DPD0~1 mode.
[0]	**WKPIN0**	**Pin0 Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Deep Power-down mode was requested by a transition of the WAKEUP pin (GPC.0). **Note:** This flag needs to be cleared by setting CLRWK(CLK_PMUSTS[31] when entering DPD0~1 mode.

Power Manager Control Register (CLK_PMUWKCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PMUWKCTL	CLK_BA+0x98	R/W	Power Manager Wake-up Control Register	0x0000_0000

31	30	29	28	27	26	25	24
DISAUTOC	Reserved			WKPINEN5		WKPINEN4	
23	22	21	20	19	18	17	16
WKPINEN3		WKPINEN2		WKPINEN1		WKPINEN0	
15	14	13	12	11	10	9	8
Reserved				WKTMRIS			
7	6	5	4	3	2	1	0
RTCWKEN	ACMPWKEN	Reserved				WKTMRMOD	WKTMREN

Bits	Description	
[31]	**DISAUTOC**	**Disable Auto Clear Wakeup flag (Write Protect)** 0 = When CPU enter NPD3/NPD4/NPD5/SPD0~2/DPD0~1, all of wake-up flags in CLK_PMUSTS are auto cleared. 1 = Disable auto clear function. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** Wakeup from NPD3/NPD4/NPD5, this bits keep original setting. **Note 3:** Wakeup from SPD0~2/DPD0~1, this bits reseted. **Note 4 :** During hardware do auto clear wakeup flag (1 HCLK cycle period), those wakeup function not work.
[30:28]	**Reserved**	Reserved.
[27:26]	**WKPINEN5**	**Wake-up Pin5 Enable Bit (Write Protect)** This is control register for GPA.12 to wake-up pin. 00 = Wake-up pin disable at Deep Power-down mode. 01 = Wake-up pin rising edge enabled at Deep Power-down mode. 10 = Wake-up pin falling edge enabled at Deep Power-down mode. 11 = Wake-up pin both edge enabled at Deep Power-down mode. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[25:24]	**WKPINEN4**	**Wake-up Pin4 Enable Bit (Write Protect)** This is control register for GPF.6 to wake-up pin. 00 = Wake-up pin disable at Deep Power-down mode. 01 = Wake-up pin rising edge enabled at Deep Power-down mode. 10 = Wake-up pin falling edge enabled at Deep Power-down mode. 11 = Wake-up pin both edge enabled at Deep Power-down mode. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note:** Setting IOCTLSEL(RTC_LXTCTL[8]) to avoid GPF6 unexpected falling edge.
[23:22]	**WKPINEN3**	**Wake-up Pin3 Enable Bit (Write Protect)** This is control register for GPB.12 to wake-up pin. 00 = Wake-up pin disable at Deep Power-down mode.

		01 = Wake-up pin rising edge enabled at Deep Power-down mode. 10 = Wake-up pin falling edge enabled at Deep Power-down mode. 11 = Wake-up pin both edge enabled at Deep Power-down mode. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[21:20]	**WKPINEN2**	**Wake-up Pin2 Enable Bit (Write Protect)** This is control register for GPB.2 to wake-up pin. 00 = Wake-up pin disable at Deep Power-down mode. 01 = Wake-up pin rising edge enabled at Deep Power-down mode. 10 = Wake-up pin falling edge enabled at Deep Power-down mode. 11 = Wake-up pin both edge enabled at Deep Power-down mode. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[19:18]	**WKPINEN1**	**Wake-up Pin1 Enable Bit (Write Protect)** This is control register for GPB.0 to wake-up pin. 00 = Wake-up pin disable at Deep Power-down mode. 01 = Wake-up pin rising edge enabled at Deep Power-down mode. 10 = Wake-up pin falling edge enabled at Deep Power-down mode. 11 = Wake-up pin both edge enabled at Deep Power-down mode. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[17:16]	**WKPINEN0**	**Wake-up Pin0 Enable Bit (Write Protect)** This is control register for GPC.0 to wake-up pin. 00 = Wake-up pin disable at Deep Power-down mode. 01 = Wake-up pin rising edge enabled at Deep Power-down mode. 10 = Wake-up pin falling edge enabled at Deep Power-down mode. 11 = Wake-up pin both edge enabled at Deep Power-down mode. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[15:12]	**Reserved**	Reserved.
[11:8]	**WKTMRIS**	**Wake-up Timer Time-out Interval Select (Write Protect)** These bits control wake-up timer time-out interval when chip at DPD/SPD mode. 0000 = Time-out interval is 512 LIRC clocks (16ms). 0001 = Time-out interval is 1024 LIRC clocks (32ms). 0010 = Time-out interval is 2048 LIRC clocks (64ms). 0011 = Time-out interval is 4096 LIRC clocks (128ms). 0100 = Time-out interval is 8192 LIRC clocks (256ms). 0101 = Time-out interval is 16384 LIRC clocks (512ms). 0110 = Time-out interval is 32768 LIRC clocks (1024ms). 0111 = Time-out interval is 65536 LIRC clocks (2048ms). 1000 = Time-out interval is 131072 LIRC clocks (4096ms). 1001 = Time-out interval is 262144 LIRC clocks (8192ms). 1010 = Time-out interval is 524288 LIRC clocks (16384ms). 1011 = Time-out interval is 1048576 LIRC clocks (32768ms). 1100 = Time-out interval is 2097152 LIRC clocks (65536ms). 1101 = Time-out interval is 4194304 LIRC clocks (131072ms). Others = Time-out interval is 512 LIRC clocks (16ms). **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[7]	**RTCWKEN**	**RTC Wake-up Enable Bit (Write Protect)** 0 = RTC wake-up disable at Deep Power-down mode or Standby Power-down mode. 1 = RTC wake-up enabled at Deep Power-down mode or Standby Power-down mode.

		Note: This bit is write protected. Refer to the SYS_REGLCTL register.
[6]	**ACMPWKEN**	**ACMP Standby Power-down Mode Wake-up Enable Bit (Write Protect)** 0 = ACMP wake-up disable at Standby Power-down mode. 1 = ACMP wake-up enabled at Standby Power-down mode. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note:** Set FILTSEL(ACMP_CTLx[15:13]) for comparator output filter count selection, the filter clock is LIRC in ACMP SPD/USPD mode wakeup function.
[5:2]	**Reserved**	Reserved.
[1]	**WKTMRMOD**	**Wake-up Timer Mode (Write Protect)** 0 = Wake-up timer started when entering any of Power-down mode (except CPU idle mode). 1 = Wake-up timer started immedially when WKTMREN (CLK_PMUWKCTL[0]) =1. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**WKTMREN**	**Wake-up Timer Enable Bit (Write Protect)** 0 = Wake-up timer Disabled. 1 = Wake-up timer Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

GPIO Pin WKIO De-bounce Control Register (CLK_PWDBCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PWDBCTL	CLK_BA+0x9C	R/W	GPIO Pin WKIO De-bounce Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				SWKDBCLKSEL			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**SWKDBCLKSEL**	**WKIO De-bounce Sampling Cycle Selection** 0000 = Sample wake-up input once per 1 clocks. 0001 = Sample wake-up input once per 2 clocks. 0010 = Sample wake-up input once per 4 clocks. 0011 = Sample wake-up input once per 8 clocks. 0100 = Sample wake-up input once per 16 clocks. 0101 = Sample wake-up input once per 32 clocks. 0110 = Sample wake-up input once per 64 clocks. 0111 = Sample wake-up input once per 128 clocks. 1000 = Sample wake-up input once per 256 clocks. 1001 = Sample wake-up input once per 2*256 clocks. 1010 = Sample wake-up input once per 4*256 clocks. 1011 = Sample wake-up input once per 8*256 clocks. 1100 = Sample wake-up input once per 16*256 clocks. 1101 = Sample wake-up input once per 32*256 clocks. 1110 = Sample wake-up input once per 64*256 clocks. 1111 = Sample wake-up input once per 128*256 clocks.. **Note:** De-bounce counter clock source is the 32 kHz internal low speed RC oscillator (LIRC).

GPA Pin WKIO Control Register (CLK_PAPWCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PAPWCTL	CLK_BA+0xA0	R/W	GPA Pin WKIO Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				NMR1	TRIGM1	Reserved	DBEN1
23	22	21	20	19	18	17	16
WKPSEL1				Reserved	PFWKEN1	PRWKEN1	WKEN1
15	14	13	12	11	10	9	8
Reserved				NMR0	TRIGM0	Reserved	DBEN0
7	6	5	4	3	2	1	0
WKPSEL0				Reserved	PFWKEN0	PRWKEN0	WKEN0

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**NMR1**	**GPA Pin 1 Function Enable at Normal Run Mode Select** 0 = GPA group pin 1 wake-up function enable when chip enters power down. 1 = GPA group pin 1 wake-up function enable when chip is normal run. **Note**: Enable this bit and the WKIO function is enabled immediately.
[26]	**TRIGM1**	**GPA Pin 1 Wake-up Pin trigger Mode Select** 0 = GPA group pin 1 wake-up chip Enabled, trigger ip Enabled. 1 = GPA group pin 1 wake-up chip Disabled, trigger ip Enabled.
[25]	**Reserved**	Reserved.
[24]	**DBEN1**	**GPA Pin 1 Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. If the input signal pulse width cannot be sampled by continuous two de-bounce sample cycle, the input signal transition is seen as the signal bounce and will not trigger the wake-up. The de-bounce clock source is the 32 kHz internal low speed RC oscillator. 0 = GPA group pin 1 Wakeup-I/O pin De-bounce function Disabled. 1 = GPA group pin 1 Wakeup-I/O pin De-bounce function Enabled. The de-bounce function is valid only for edge triggered.
[23:20]	**WKPSEL1**	**GPA Pin 1 Wakeup-I/O Pin Select** 0000 = GPA.0 as Wakeup-I/O function select. 0001 = GPA.1 as Wakeup-I/O function select. 0010 = GPA.2 as Wakeup-I/O function select. 0011 = GPA.3 as Wakeup-I/O function select. 0100 = GPA.4 as Wakeup-I/O function select. 0101 = GPA.5 as Wakeup-I/O function select. 0110 = GPA.6 as Wakeup-I/O function select. 0111 = GPA.7 as Wakeup-I/O function select. 1000 = GPA.8 as Wakeup-I/O function select. 1001 = GPA.9 as Wakeup-I/O function select.

		1010 = GPA.10 as Wakeup-I/O function select. 1011 = GPA.11 as Wakeup-I/O function select. 1100 = GPA.12 as Wakeup-I/O function select. 1101 = GPA.13 as Wakeup-I/O function select. 1110 = GPA.14 as Wakeup-I/O function select. 1111 = GPA.15 as Wakeup-I/O function select.
[19]	**Reserved**	Reserved.
[18]	**PFWKEN1**	**GPA Pin 1 Falling Edge Detect Enable Bit** 0 = GPA group pin 1 falling edge detect function Disabled. 1 = GPA group pin 1 falling edge detect function Enabled.
[17]	**PRWKEN1**	**GPA Pin 1 Rising Edge Detect Enable Bit** 0 = GPA group pin 1 rising edge detect function Disabled. 1 = GPA group pin 1 rising edge detect function Enabled.
[16]	**WKEN1**	**GPA Pin 1 Wake-up Enable Bit** 0 = GPA group pin 1 wake-up function Disabled. 1 = GPA group pin 1 wake-up function Enabled.
[15:12]	**Reserved**	Reserved.
[11]	**NMR0**	**GPA Pin 0 Function Enable at Normal Run Mode Select** 0 = GPA group pin 0 wake-up function enable when chip enters power down. 1 = GPA group pin 0 wake-up function enable when chip is normal run. **Note**: Enable this bit and the I/O wake-up function is enabled immediately.
[10]	**TRIGM0**	**GPA Pin 0 Wake-up Pin Trigger Mode Select** 0 = GPA group pin 0 wake-up chip Enabled, trigger ip Enabled. 1 = GPA group pin 0 wake-up chip Disabled, trigger ip Enabled.
[9]	**Reserved**	Reserved.
[8]	**DBEN0**	**GPA Pin 0 Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. If the input signal pulse width cannot be sampled by continuous two de-bounce sample cycle, the input signal transition is seen as the signal bounce and will not trigger the wake-up. The de-bounce clock source is the 32 kHz internal low speed RC oscillator. 0 = GPA group pin 0 Wake-up-I/O pin De-bounce function Disabled. 1 = GPA group pin 0 Wakeup-I/O pin De-bounce function Enabled. The de-bounce function is valid only for edge triggered.
[7:4]	**WKPSEL0**	**GPA Pin 0 Wakeup-I/O Pin Select** 0000 = GPA.0 as Wake-up-I/O function select. 0001 = GPA.1 as Wakeup-I/O function select. 0010 = GPA.2 as Wakeup-I/O function select. 0011 = GPA.3 as Wakeup-I/O function select. 0100 = GPA.4 as Wakeup-I/O function select. 0101 = GPA.5 as Wakeup-I/O function select. 0110 = GPA.6 as Wakeup-I/O function select. 0111 = GPA.7 as Wakeup-I/O function select. 1000 = GPA.8 as Wakeup-I/O function select. 1001 = GPA.9 as Wakeup-I/O function select. 1010 = GPA.10 as Wakeup-I/O function select. 1011 = GPA.11 as Wakeup-I/O function select.

		1100 = GPA.12 as Wakeup-I/O function select. 1101 = GPA.13 as Wakeup-I/O function select. 1110 = GPA.14 as Wakeup-I/O function select. 1111 = GPA.15 as Wakeup-I/O function select.
[3]	**Reserved**	Reserved.
[2]	**PFWKEN0**	**GPA Pin 0 Falling Edge Detect Enable Bit** 0 = GPA group pin 0 falling edge detect function Disabled. 1 = GPA group pin 0 falling edge detect function Enabled.
[1]	**PRWKEN0**	**GPA Pin 0 Rising Edge Detect Enable Bit** 0 = GPA group pin 0 rising edge detect function Disabled. 1 = GPA group pin 0 rising edge detect function Enabled.
[0]	**WKEN0**	**GPA Pin 0 Wake-up-I/O Enable Bit** 0 = GPA group pin 0 Wake-up-I/O function Disabled. 1 = GPA group pin 0 Wakeup-I/O function Enabled.

GPB Pin WKIO Control Register (CLK_PBPWCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PBPWCTL	CLK_BA+0xA4	R/W	GPB Pin WKIO Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				NMR1	TRIGM1	Reserved	DBEN1
23	22	21	20	19	18	17	16
WKPSEL1				Reserved	PFWKEN1	PRWKEN1	WKEN1
15	14	13	12	11	10	9	8
Reserved				NMR0	TRIGM0	Reserved	DBEN0
7	6	5	4	3	2	1	0
WKPSEL0				Reserved	PFWKEN0	PRWKEN0	WKEN0

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**NMR1**	**GPB Pin 1 Function Enable at Normal Run Mode Select** 0 = GPB group pin 1 wake-up function enable when chip enters power down. 1 = GPB group pin 1 wake-up function enable when chip is normal run. **Note:** Enable this bit and the WKIO function is enabled immediately.
[26]	**TRIGM1**	**GPB Pin 1 Wake-up Pin Trigger Mode Select** 0 = GPB group pin 1 wake-up chip Enabled, trigger ip Enabled. 1 = GPB group pin 1 wake-up chip Disabled, trigger ip Enabled.
[25]	**Reserved**	Reserved.
[24]	**DBEN1**	**GPB Pin 1 Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. If the input signal pulse width cannot be sampled by continuous two de-bounce sample cycle, the input signal transition is seen as the signal bounce and will not trigger the wake-up. The de-bounce clock source is the 32 kHz internal low speed RC oscillator. 0 = GPB group pin 1 Wakeup-I/O pin De-bounce function Disabled. 1 = GPB group pin 1 Wakeup-I/O pin De-bounce function Enabled. The de-bounce function is valid only for edge triggered.
[23:20]	**WKPSEL1**	**GPB Pin 1 Wakeup-I/O Pin Select** 0000 = GPB.0 as Wakeup-I/O function select. 0001 = GPB.1 as Wakeup-I/O function select. 0010 = GPB.2 as Wakeup-I/O function select. 0011 = GPB.3 as Wakeup-I/O function select. 0100 = GPB.4 as Wakeup-I/O function select. 0101 = GPB.5 as Wakeup-I/O function select. 0110 = GPB.6 as Wakeup-I/O function select. 0111 = GPB.7 as Wakeup-I/O function select. 1000 = GPB.8 as Wakeup-I/O function select. 1001 = GPB.9 as Wakeup-I/O function select.

		1010 = GPB.10 as Wakeup-I/O function select. 1011 = GPB.11 as Wakeup-I/O function select. 1100 = GPB.12 as Wakeup-I/O function select. 1101 = GPB.13 as Wakeup-I/O function select. 1110 = GPB.14 as Wakeup-I/O function select. 1111 = GPB.15 as Wakeup-I/O function select.
[19]	**Reserved**	Reserved.
[18]	**PFWKEN1**	**GPB Pin 1 Falling Edge Detect Enable Bit** 0 = GPB group pin 1 falling edge detect function Disabled. 1 = GPB group pin 1 falling edge detect function Enabled.
[17]	**PRWKEN1**	**GPB Pin 1 Rising Edge Detect Enable Bit** 0 = GPB group pin 1 rising edge detect function Disabled. 1 = GPB group pin 1 rising edge detect function Enabled.
[16]	**WKEN1**	**GPB Pin 1 Wake-up Enable Bit** 0 = GPB group pin 1 wake-up function Disabled. 1 = GPB group pin 1 wake-up function Enabled.
[15:12]	**Reserved**	Reserved.
[11]	**NMR0**	**GPB Pin 0 Function Enable at Normal Run Mode Select** 0 = GPB group pin 0 wake-up function enable when chip enters power down. 1 = GPB group pin 0 wake-up function enable when chip is normal run. **Note:** Enable this bit and the I/O wake-up function is enabled immediately.
[10]	**TRIGM0**	**GPB Pin 0 Wake-up Pin Trigger Mode Select** 0 = GPB group pin 0 wake-up chip Enabled, trigger ip Enabled. 1 = GPB group pin 0 wake-up chip Disabled, trigger ip Enabled.
[9]	**Reserved**	Reserved.
[8]	**DBEN0**	**GPB Pin 0 Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. If the input signal pulse width cannot be sampled by continuous two de-bounce sample cycle, the input signal transition is seen as the signal bounce and will not trigger the wake-up. The de-bounce clock source is the 32 kHz internal low speed RC oscillator. 0 = GPB group pin 0 Wakeup-I/O pin De-bounce function Disabled. 1 = GPB group pin 0 Wakeup-I/O pin De-bounce function Enabled. The de-bounce function is valid only for edge triggered.
[7:4]	**WKPSEL0**	**GPB Pin 0 Wakeup-I/O Pin Select** 0000 = GPB.0 as Wakeup-I/O function select. 0001 = GPB.1 as Wakeup-I/O function select. 0010 = GPB.2 as Wakeup-I/O function select. 0011 = GPB.3 as Wakeup-I/O function select. 0100 = GPB.4 as Wakeup-I/O function select. 0101 = GPB.5 as Wakeup-I/O function select. 0110 = GPB.6 as Wakeup-I/O function select. 0111 = GPB.7 as Wakeup-I/O function select. 1000 = GPB.8 as Wakeup-I/O function select. 1001 = GPB.9 as Wakeup-I/O function select. 1010 = GPB.10 as Wakeup-I/O function select. 1011 = GPB.11 as Wakeup-I/O function select.

		1100 = GPB.12 as Wakeup-I/O function select. 1101 = GPB.13 as Wakeup-I/O function select. 1110 = GPB.14 as Wakeup-I/O function select. 1111 = GPB.15 as Wakeup-I/O function select.
[3]	**Reserved**	Reserved.
[2]	**PFWKEN0**	**GPB Pin 0 Falling Edge Detect Enable Bit** 0 = GPB group pin 0 falling edge detect function Disabled. 1 = GPB group pin 0 falling edge detect function Enabled.
[1]	**PRWKEN0**	**GPB Pin 0 Rising Edge Detect Enable Bit** 0 = GPB group pin 0 rising edge detect function Disabled. 1 = GPB group pin 0 rising edge detect function Enabled.
[0]	**WKEN0**	**GPB Pin 0 Wakeup-I/O Enable Bit** 0 = GPB group pin 0 Wakeup-I/O function Disabled. 1 = GPB group pin 0 Wakeup-I/O function Enabled.

GPC Pin WKIO Control Register (CLK_PCPWCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PCPWCTL	CLK_BA+0xA8	R/W	GPC Pin WKIO Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				NMR1	TRIGM1	Reserved	DBEN1
23	22	21	20	19	18	17	16
WKPSEL1				Reserved	PFWKEN1	PRWKEN1	WKEN1
15	14	13	12	11	10	9	8
Reserved				NMR0	TRIGM0	Reserved	DBEN0
7	6	5	4	3	2	1	0
WKPSEL0				Reserved	PFWKEN0	PRWKEN0	WKEN0

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**NMR1**	**GPC Pin 1 Function Enable at Normal Run Mode Select** 0 = GPC group pin 1 wake-up function enabled when chip enters power down. 1 = GPC group pin 1 wake-up function enabled when chip is normal run. **Note:** Enable this bit and the WKIO function is enabled immediately.
[26]	**TRIGM1**	**GPC Pin 1 Wake-up Pin Trigger Mode Select** 0 = GPC group pin 1 wake-up chip Enabled, trigger ip Enabled. 1 = GPC group pin 1 wake-up chip Disabled, trigger ip Enabled.
[25]	**Reserved**	Reserved.
[24]	**DBEN1**	**GPC Pin 1 Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. If the input signal pulse width cannot be sampled by continuous two de-bounce sample cycle, the input signal transition is seen as the signal bounce and will not trigger the wake-up. The de-bounce clock source is the 32 kHz internal low speed RC oscillator. 0 = GPC group pin 1 Wakeup-I/O pin De-bounce function Disabled. 1 = GPC group pin 1 Wakeup-I/O pin De-bounce function Enabled. The de-bounce function is valid only for edge triggered.
[23:20]	**WKPSEL1**	**GPC Pin 1 Wakeup-I/O Pin Select** 0000 = GPC.0 as Wakeup-I/O function select. 0001 = GPC.1 as Wakeup-I/O function select. 0010 = GPC.2 as Wakeup-I/O function select. 0011 = GPC.3 as Wakeup-I/O function select. 0100 = GPC.4 as Wakeup-I/O function select. 0101 = GPC.5 as Wakeup-I/O function select. 0110 = GPC.6 as Wakeup-I/O function select. 0111 = GPC.7 as Wakeup-I/O function select. 1000 = GPC.8 as Wakeup-I/O function select. 1001 = GPC.9 as Wakeup-I/O function select.

		1010 = GPC.10 as Wakeup-I/O function select. 1011 = GPC.11 as Wakeup-I/O function select. 1100 = GPC.12 as Wakeup-I/O function select. 1101 = GPC.13 as Wakeup-I/O function select. 1110 = GPC.14 as Wakeup-I/O function select. 1111 = GPC.15 as Wakeup-I/O function select.
[19]	**Reserved**	Reserved.
[18]	**PFWKEN1**	**GPC Pin 1 Falling Edge Detect Enable Bit** 0 = GPC group pin 1 falling edge detect function Disabled. 1 = GPC group pin 1 falling edge detect function Enabled.
[17]	**PRWKEN1**	**GPC Pin 1 Rising Edge Detect Enable Bit** 0 = GPC group pin 1 rising edge detect function Disabled. 1 = GPC group pin 1 rising edge detect function Enabled.
[16]	**WKEN1**	**GPC Pin 1 Wake-up Enable Bit** 0 = GPC group pin 1 wake-up function Disabled. 1 = GPC group pin 1 wake-up function Enabled.
[15:12]	**Reserved**	Reserved.
[11]	**NMR0**	**GPC Pin 0 Function Enable at Normal Run Mode Select** 0 = GPC group pin 0 wake-up function enabled when chip enters power down. 1 = GPC group pin 0 wake-up function enabled when chip is normal run. **Note:** Enable this bit and the I/O wake-up function is enabled immediately.
[10]	**TRIGM0**	**GPC Pin 0 Wake-up Pin trigger Mode Select** 0 = GPC group pin 0 wake-up chip Enabled, trigger ip Enabled. 1 = GPC group pin 0 wake-up chip Disabled, trigger ip Enabled.
[9]	**Reserved**	Reserved.
[8]	**DBEN0**	**GPC Pin 0 Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. If the input signal pulse width cannot be sampled by continuous two de-bounce sample cycle, the input signal transition is seen as the signal bounce and will not trigger the wake-up. The de-bounce clock source is the 32 kHz internal low speed RC oscillator. 0 = GPC group pin 0 Wakeup-I/O pin De-bounce function Disabled. 1 = GPC group pin 0 Wakeup-I/O pin De-bounce function Enabled. The de-bounce function is valid only for edge triggered.
[7:4]	**WKPSEL0**	**GPC Pin 0 Wakeup-I/O Pin Select** 0000 = GPC.0 as Wakeup-I/O function select. 0001 = GPC.1 as Wakeup-I/O function select. 0010 = GPC.2 as Wakeup-I/O function select. 0011 = GPC.3 as Wakeup-I/O function select. 0100 = GPC.4 as Wakeup-I/O function select. 0101 = GPC.5 as Wakeup-I/O function select. 0110 = GPC.6 as Wakeup-I/O function select. 0111 = GPC.7 as Wakeup-I/O function select. 1000 = GPC.8 as Wakeup-I/O function select. 1001 = GPC.9 as Wakeup-I/O function select. 1010 = GPC.10 as Wakeup-I/O function select. 1011 = GPC.11 as Wakeup-I/O function select.

		1100 = GPC.12 as Wakeup-I/O function select. 1101 = GPC.13 as Wakeup-I/O function select. 1110 = GPC.14 as Wakeup-I/O function select. 1111 = GPC.15 as Wakeup-I/O function select.
[3]	**Reserved**	Reserved.
[2]	**PFWKEN0**	**GPC Pin 0 Falling Edge Detect Enable Bit** 0 = GPC group pin 0 falling edge detect function Disabled. 1 = GPC group pin 0 falling edge detect function Enabled.
[1]	**PRWKEN0**	**GPC Pin 0 Rising Edge Detect Enable Bit** 0 = GPC group pin 0 rising edge detect function Disabled. 1 = GPC group pin 0 rising edge detect function Enabled.
[0]	**WKEN0**	**GPC Pin 0 Wakeup-I/O Enable Bit** 0 = GPC group pin 0 Wakeup-I/O function Disabled. 1 = GPC group pin 0 Wakeup-I/O function Enabled.

GPD Pin WKIO Control Register (CLK_PDPWCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PDPWCTL	CLK_BA+0xAC	R/W	GPD Pin WKIO Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				NMR1	TRIGM1	Reserved	DBEN1
23	22	21	20	19	18	17	16
WKPSEL1				Reserved	PFWKEN1	PRWKEN1	WKEN1
15	14	13	12	11	10	9	8
Reserved				NMR0	TRIGM0	Reserved	DBEN0
7	6	5	4	3	2	1	0
WKPSEL0				Reserved	PFWKEN0	PRWKEN0	WKEN0

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**NMR1**	**GPD Pin 1 Function Enable at Normal Run Mode Select** 0 = GPD group pin 1 wake-up function enable when chip enters power down. 1 = GPD group pin 1 wake-up function enable when chip is normal run. **Note:** Enable this bit and the WKIO function is enabled immediately.
[26]	**TRIGM1**	**GPD Pin 1 Wake-up Pin Trigger Mode Select** 0 = GPD group pin 1 wake-up chip Enabled, trigger ip Enabled. 1 = GPD group pin 1 wake-up chip Disabled, trigger ip Enabled.
[25]	**Reserved**	Reserved.
[24]	**DBEN1**	**GPD Pin 1 Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. If the input signal pulse width cannot be sampled by continuous two de-bounce sample cycle, the input signal transition is seen as the signal bounce and will not trigger the wake-up. The de-bounce clock source is the 32 kHz internal low speed RC oscillator. 0 = GPD group pin 1 Wakeup-I/O pin De-bounce function Disabled. 1 = GPD group pin 1 Wakeup-I/O pin De-bounce function Enabled. The de-bounce function is valid only for edge triggered.
[23:20]	**WKPSEL1**	**GPD Pin 1 Wakeup-I/O Pin Select** 0000 = GPD.0 as Wakeup-I/O function select. 0001 = GPD.1 as Wakeup-I/O function select. 0010 = GPD.2 as Wakeup-I/O function select. 0011 = GPD.3 as Wakeup-I/O function select. 0100 = GPD.4 as Wakeup-I/O function select. 0101 = GPD.5 as Wakeup-I/O function select. 0110 = GPD.6 as Wakeup-I/O function select. 0111 = GPD.7 as Wakeup-I/O function select. 1000 = GPD.8 as Wakeup-I/O function select. 1001 = GPD.9 as Wakeup-I/O function select.

		1010 = GPD.10 as Wakeup-I/O function select. 1011 = GPD.11 as Wakeup-I/O function select. 1100 = GPD.12 as Wakeup-I/O function select. 1101 = GPD.13 as Wakeup-I/O function select. 1110 = GPD.14 as Wakeup-I/O function select. 1111 = GPD.15 as Wakeup-I/O function select.
[19]	**Reserved**	Reserved.
[18]	**PFWKEN1**	**GPD Pin 1 Falling Edge Detect Enable Bit** 0 = GPD group pin 1 falling edge detect function Disabled. 1 = GPD group pin 1 falling edge detect function Enabled.
[17]	**PRWKEN1**	**GPD Pin 1 Rising Edge Detect Enable Bit** 0 = GPD group pin 1 rising edge detect function Disabled. 1 = GPD group pin 1 rising edge detect function Enabled.
[16]	**WKEN1**	**GPD Pin 1 Wake-up Enable Bit** 0 = GPD group pin 1 wake-up function Disabled. 1 = GPD group pin 1 wake-up function Enabled.
[15:12]	**Reserved**	Reserved.
[11]	**NMR0**	**GPD Pin 0 Function Enable at Normal Run Mode Select** 0 = GPD group pin 0 wake-up function enable when chip enters power down. 1 = GPD group pin 0 wake-up function enable when chip is normal run. **Note:** Enable this bit and the I/O wake-up function is enabled immediately.
[10]	**TRIGM0**	**GPD Pin 0 Wake-up Pin Trigger Mode Select** 0 = GPD group pin 0 wake-up chip Enabled, trigger ip Enabled. 1 = GPD group pin 0 wake-up chip Disabled, trigger ip Enabled.
[9]	**Reserved**	Reserved.
[8]	**DBEN0**	**GPD Pin 0 Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. If the input signal pulse width cannot be sampled by continuous two de-bounce sample cycle, the input signal transition is seen as the signal bounce and will not trigger the wake-up. The de-bounce clock source is the 32 kHz internal low speed RC oscillator. 0 = GPD group pin 0 Wakeup-I/O pin De-bounce function Disabled. 1 = GPD group pin 0 Wakeup-I/O pin De-bounce function Enabled. The de-bounce function is valid only for edge triggered.
[7:4]	**WKPSEL0**	**GPD Pin 0 Wakeup-I/O Pin Select** 0000 = GPD.0 as Wakeup-I/O function select. 0001 = GPD.1 as Wakeup-I/O function select. 0010 = GPD.2 as Wakeup-I/O function select. 0011 = GPD.3 as Wakeup-I/O function select. 0100 = GPD.4 as Wakeup-I/O function select. 0101 = GPD.5 as Wakeup-I/O function select. 0110 = GPD.6 as Wakeup-I/O function select. 0111 = GPD.7 as Wakeup-I/O function select. 1000 = GPD.8 as Wakeup-I/O function select. 1001 = GPD.9 as Wakeup-I/O function select. 1010 = GPD.10 as Wakeup-I/O function select. 1011 = GPD.11 as Wakeup-I/O function select.

		1100 = GPD.12 as Wakeup-I/O function select. 1101 = GPD.13 as Wakeup-I/O function select. 1110 = GPD.14 as Wakeup-I/O function select. 1111 = GPD.15 as Wakeup-I/O function select.
[3]	**Reserved**	Reserved.
[2]	**PFWKEN0**	**GPD Pin 0 Falling Edge Detect Enable Bit** 0 = GPD group pin 0 falling edge detect function Disabled. 1 = GPD group pin 0 falling edge detect function Enabled.
[1]	**PRWKEN0**	**GPD Pin 0 Rising Edge Detect Enable Bit** 0 = GPD group pin 0 rising edge detect function Disabled. 1 = GPD group pin 0 rising edge detect function Enabled.
[0]	**WKEN0**	**GPD Pin 0 Wakeup-I/O Enable Bit** 0 = GPD group pin 0 Wakeup-I/O function Disabled. 1 = GPD group pin 0 Wakeup-I/O function Enabled.

GPIO Power-down Control Register (CLK_IOPDCTL)

Register	Offset	R/W	Description	Reset Value
CLK_IOPDCTL	CLK_BA+0xB0	R/W	GPIO Power-down Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**DPDHOLDEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**IOHR**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**DPDHOLDEN**	**Deep-Power-Down Mode GPIO Hold Enable Bit** 0= When GPIO enters deep power-down mode, all I/O status are tri-state. 1= When GPIO enters deep power-down mode, all I/O status are hold to keep normal operating status. After chip was waked up from deep power-down mode, the I/O are still keep hold status until user set CLK_IOPDCTL[0] to release I/O hold status.
[7:1]	**Reserved**	Reserved.
[0]	**IOHR**	**GPIO Hold Release** When GPIO enters deep power-down mode or standby power-down mode, all I/O status are hold to keep normal operating status. After chip was waked up from deep power-down mode or standby power-down mode, the I/O are still keep hold status until user set this bit to release I/O hold status. **Note:** This bit is auto cleared by hardware.

Power Manager Interrupt Control Register (CLK_PMUINTC)

Register	Offset	R/W	Description	Reset Value
CLK_PMUINTC	CLK_BA+0xC0	R/W	Power Manager Interrupt Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
WKIOD1IE	WKIOC1IE	WKIOB1IE	WKIOA1IE	WKIOD0IE	WKIOC0IE	WKIOB0IE	WKIOA0IE
7	6	5	4	3	2	1	0
Reserved							WKTMRIE

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**WKIOD1IE**	**Wakeup-I/O GPD group Pin 1 Interrupt Enable Bit** 0 = Wakeup-I/O interrupt function Disabled. 1 = Wakeup-I/O interrupt function Enabled.
[14]	**WKIOC1IE**	**Wakeup-I/O GPC group Pin 1 Interrupt Enable Bit** 0 = Wakeup-I/O interrupt function Disabled. 1 = Wakeup-I/O interrupt function Enabled.
[13]	**WKIOB1IE**	**Wakeup-I/O GPB group Pin 1 Interrupt Enable Bit** 0 = Wakeup-I/O interrupt function Disabled. 1 = Wakeup-I/O interrupt function Enabled.
[12]	**WKIOA1IE**	**Wakeup-I/O GPA group Pin 1 Interrupt Enable Bit** 0 = Wakeup-I/O interrupt function Disabled. 1 = Wakeup-I/O interrupt function Enabled.
[11]	**WKIOD0IE**	**Wakeup-I/O GPD group Pin 0 Interrupt Enable Bit** 0 = Wakeup-I/O interrupt function Disabled. 1 = Wakeup-I/O interrupt function Enabled.
[10]	**WKIOC0IE**	**Wakeup-I/O GPC group Pin 0 Interrupt Enable Bit** 0 = Wakeup-I/O interrupt function Disabled. 1 = Wakeup-I/O interrupt function Enabled.
[9]	**WKIOB0IE**	**Wakeup-I/O GPB group Pin 0 Interrupt Enable Bit** 0 = Wakeup-I/O interrupt function Disabled. 1 = Wakeup-I/O interrupt function Enabled.
[8]	**WKIOA0IE**	**Wakeup-I/O GPA group Pin 0 Interrupt Enable Bit** 0 = Wakeup-I/O interrupt function Disabled. 1 = Wakeup-I/O interrupt function Enabled.

[7:1]	**Reserved**	Reserved.
[0]	**WKTMRIE**	**Wakeup-Timer Interrupt Enable Bit** 0 = Wakeup-Timer interrupt function Disabled. 1 = Wakeup-Timer interrupt function Enabled.

Power Manager Interrupt Status Register (CLK_PMUINTS)

Register	Offset	R/W	Description	Reset Value
CLK_PMUINTS	CLK_BA+0xC4	R/W	Power Manager Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
WKIOD1IF	WKIOC1IF	WKIOB1IF	WKIOA1IF	WKIOD0IF	WKIOC0IF	WKIOB0IF	WKIOA0IF
7	6	5	4	3	2	1	0
Reserved							WKTMRIF

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**WKIOD1IF**	**Wakeup-I/O GPD group Pin 1 Interrupt Flag** This flag indicates that Wakeup-I/O interrupt happened. Flag is set by hardware while Wakeup-I/O event happen when WKIOPD1IE(CLK_PMUINTC[15])=1. **Note:** Software can clear this bit by writing 1 to it.
[14]	**WKIOC1IF**	**Wakeup-I/O GPC group Pin 1 Interrupt Flag** This flag indicates that Wakeup-I/O interrupt happened. Flag is set by hardware while Wakeup-I/O event happen when WKIOPC1IE(CLK_PMUINTC[14])=1. **Note:** Software can clear this bit by writing 1 to it.
[13]	**WKIOB1IF**	**Wakeup-I/O GPB group Pin 1 Interrupt Flag** This flag indicates that Wakeup-I/O interrupt happened. Flag is set by hardware while Wakeup-I/O event happen when WKIOPB1IE(CLK_PMUINTC[13])=1. **Note:** Software can clear this bit by writing 1 to it.
[12]	**WKIOA1IF**	**Wakeup-I/O GPA group Pin 1 Interrupt Flag** This flag indicates that Wakeup-I/O interrupt happened. Flag is set by hardware while Wakeup-I/O event happen when WKIOPA1IE(CLK_PMUINTC[12])=1. **Note:** Software can clear this bit by writing 1 to it.
[11]	**WKIOD0IF**	**Wakeup-I/O GPD group Pin 0 Interrupt Flag** This flag indicates that Wakeup-I/O interrupt happened. Flag is set by hardware while Wakeup-I/O event happen when WKIOPD0IE (CLK_PMUINTC[11])=1. **Note:** Software can clear this bit by writing 1 to it.
[10]	**WKIOC0IF**	**Wakeup-I/O GPC group Pin 0 Interrupt Flag** This flag indicates that Wakeup-I/O interrupt happened.

		Flag is set by hardware while Wakeup-I/O event happen when WKIOPC0IE(CLK_PMUINTC[10])=1. **Note:** Software can clear this bit by writing 1 to it.
[9]	**WKIOB0IF**	**Wakeup-I/O GPB group Pin 0 Interrupt Flag** This flag indicates that Wakeup-I/O interrupt happened. Flag is set by hardware while Wakeup-I/O event happen when WKIOPB0IE(CLK_PMUINTC[9])=1. **Note:** Software can clear this bit by writing 1 to it.
[8]	**WKIOA0IF**	**Wakeup-I/O GPA group Pin 0 Interrupt Flag** This flag indicates that Wakeup-I/O interrupt happened. Flag is set by hardware while Wakeup-I/O event happen when WKIOPA0IE(CLK_PMUINTC[8])=1. **Note:** Software can clear this bit by writing 1 to it.
[7:1]	**Reserved**	Reserved.
[0]	**WKTMRIF**	**Wakeup-Timer Interrupt Flag** This flag indicates that Wakeup-Timer interrupt happened. Flag is set by hardware while Wakeup-Timer event happen when WKTMRIE(CLK_PMUINTC[0])=1. **Note:** Software can clear this bit by writing 1 to it.

6.4 True Random Number Generator (TRNG)

6.4.1 Overview

This is a True Random Number Generator (TRNG) consists of an entropy source from analog macro and a CTR_DRBG. It can generate true random number by analog macro. There is a calibration mechanism of internal TRNG for randomness of generating true random number. Besides, there is a CTR_DRBG design in this TRNG for the DRBG (deterministic random bit generator) of SP800-90A and the NRBG (non-deterministic random bit generator) of SP800-90C.

6.4.2 Features

- Up to 10Mbit/s data rate for entropy source
- Pass NIST SP800-22A standard
- Include NIST SP800-90A known answer test for DRBG and NIST SP800-90B health test for entropy source (Start-up test, Repetition count test, Adaptive proportion test).
- Provides the true random number seed for PRNG

6.4.3 Block Diagram

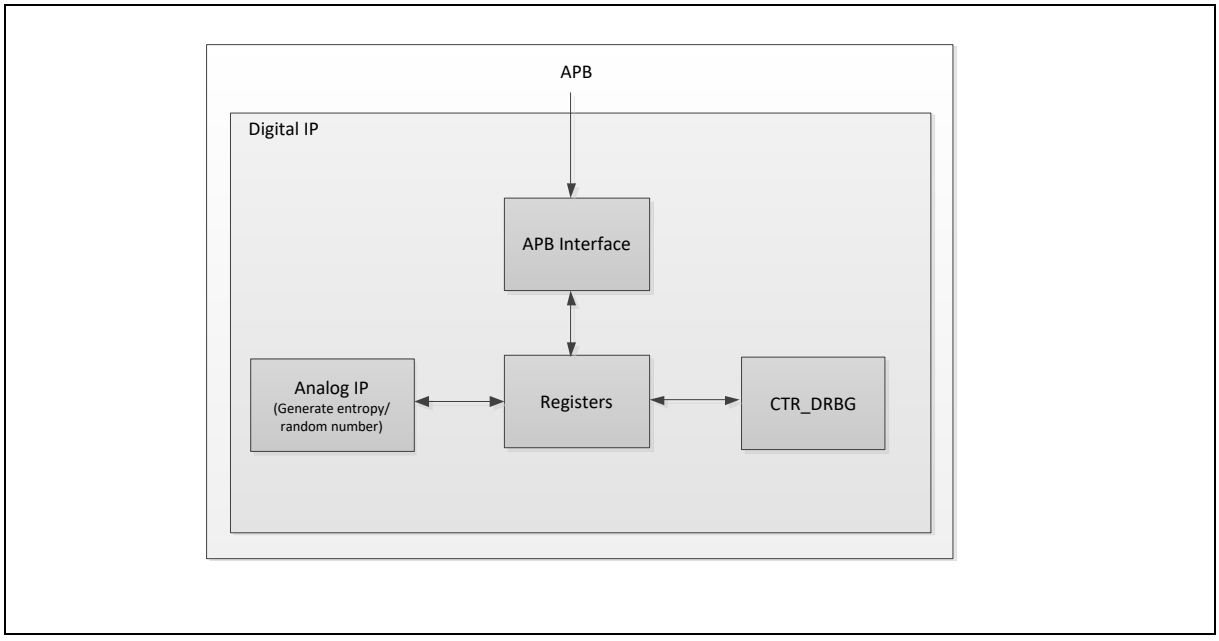


Figure 6.4-1 True Random Number Generator Block Diagram

6.4.4 Basic Configuration

6.4.4.1 Clock Source Configuration

- Enable TRNG peripheral clock in TRNGCKEN (CLK_APBCLK0[31]).

6.4.4.2 Reset Configuration

- Reset TRNG controller in TRNGRST (SYS_IPRST1[31]).

6.4.5 Functional Description

There are two ways generating the random number by an analog macro or the CTR_DRBG.

6.4.5.1 Analog Macro

The TRNG can provide the random number. If DVIEN (TRNG_CTL[31]) is enabled, the TRNG will generate interrupt to CPU when DVIF is set. When DVIF (TRNG_STS[31]) bit is set, the true random can be read at TRNG_DATAx register. If DVIF is 0, the TRNG_DATAx register returns the value 0x0 when read.

The programming steps to get the true random number are depicted below.

1. Enable LDOEN (TRNG_CTL[0]) bit.
2. Wait LDORDY(TRNG_STS[0]) bit to be set to 1.
3. Set NRST (TRNG_CTL[1]) bit to 1 and enable TRNGEN (TRNG_CTL[2]) bit.
4. Configure TRNG control register TRNG_CTL for MODE (TRNG_CTL[5:4]) and START (TRNG_CTL[3]).
5. Wait DVIF (TRNG_STS[31]) bit to be set to 1.
6. Read TRNG_DATA0 register to get random number.

Repeat Step 4 and 6 to get more random number.

TRNG generates the true random number seed for PRNG

The programming steps of PRNG to get the random number with the true random number seed from TRNG are depicted below.

1. Enable LDOEN (TRNG_CTL[0]) bit.
2. Wait LDORDY(TRNG_STS[0]) bit to be set to 1.
3. Set NRST (TRNG_CTL[1]) bit to 1 and enable TRNGEN (TRNG_CTL[2]) bit.
4. Check the BUSY(CRYPTO_PRNG_CTL[8]) until it comes to 0.
5. Configure PRNG control register CRYPTO_PRNG_CTL for seed source SEEDSRC (CRYPTO_PRNG_CTL[16]), key size KETSZ (CRYPTO_PRNG_CTL[4:2]), seed reload SEEDRLD (CRYPTO_PRNG_CTL[1]), and PRNG start START (CRYPTO_PRNG_CTL[0]).
6. Software checks BUSY(CRYPTO_PRNG_CTL[8]) until it comes to 0, or waits for the PRNGIF(CRYPTO_INTSTS[16]) (must enable PRNGIEN(CRYPTO_INTEN[16])). Then software can read the output random numbers from CRYPTO_PRNG_KEY0 ~ CRYPTO_PRNG_KEY7.

Some Notices of TRNG:

1. If initial setting is incorrect, then TRNGRDY (TRNG_STS[1]) won't be '1'.
2. If users want to reset TRNG, the signal NRST (TRNG_CTL[1]) should keep '0' over 100ns.
3. After DVIF (TRNG_STS[31]) bit is set to 1. If users execute that TRNG generates the true random number seed for PRNG, the DVIF (TRNG_STS[31]) bit will be set to 0 because the random number has been taken away for PRNG.

6.4.5.2 CTR_DRBG

The CTR_DRBG hardware that supports AES-128 security strength is designed for the DRBG of SP800-90A and the NRBG of SP800-90C. There are three main functions including instantiation, reseed and update function to adjust the internal state (i.e., key and counter) of CTR_DRBG. The instantiation function will obtain entropy input to initialize the internal state of CTR_DRBG. The reseed function will combime new entropy input with the current internal state to create a new internal state of CTR_DRBG. The update function will just use the current internal state to renew the internal state of CTR_DRBG.

The CTR_DRBG provides hardware automatic or user manual instantiation, reseed and update function for adjusting its security strength. In the hardware automatic mechanism, user can configure the counter field length (abbreviated to ctr_len) and the maximum number of requests between reseeds

(abbreviated to reseed_interval) variables to determine the security of CTR_DRBG. When the block number of generated random data reaches (2^{ctr_len} – 4), the CTR_DRBG will perform update function automatically. When the execution times of update function is equal to the value of reseed_interval, the CTR_DRBG will automatically perform reseed function. In the user manual mechanism, users can force the CTR_DRBG to perform the update, reseed or instantiation functions in the next random number generation. The programming steps of CTR_DRBG are depicted below.

CTR_DRBG generates a deterministic / non-deterministic 128-bit random number

1. Enable LDOEN (TRNG_CTL[0]) bit.
2. Wait LDORDY(TRNG_STS[0]) bit to be set to 1.
3. Set NRST (TRNG_CTL[1]) bit to 1 and enable TRNGEN (TRNG_CTL[2]) bit
4. If users want to change the security strength of CTR_DRBG, configure CTR_DRBG configure register TRNG_CFG for the counter field length CTRLEN (TRNG_CFG[31:28]) and the maximum number of requests between reseeds RESEED_INTERVAL (TRNG_CFG[25:0]).
5. If users want to force CTR_DRBG to perform the update, reseed or instantiation functions, configure CTR_DRBG control register TRNG_CTL for user manual update function UPDATE (TRNG_CTL[10]), user manual reseed function RESEED (TRNG_CTL[9]) and user manual instantiation function INSTANT (TRNG_CTL[8]).
6. Configure CTR_DRBG control register TRNG_CTL for random bit generator selection MODE (TRNG_CTL[5:4]), and CTR_DRBG start START (TRNG_CTL[3]).
7. Wait DVIF (TRNG_STS[31]) bit to be set to 1. Then software can read the output random numbers from TRNG_DATA0 ~ TRNG_DATA3.

Some Notices of CTR_DRBG:

1. When the instantiation, reseed and update meet their execution condition at the same time, our CTR_DRBG will only execute the instantiation function
2. When the reseed and update meet their execution condition at the same time, our CTR_DRBG will only execute the reseed function.
3. Do not enable START (TRNG_CTL[3]) and KATEN (TRNG_CTL[24]) at the same time.
4. The instantiation, reseed and update will be executed before generating a new random number.
5. If user wite CTR_DRBG configure register (TRNG_CFG), CTR_DRBG must execute instantiation function automatically.

6.4.5.3 Known answer test (KAT) and health test

The CTR_DRBG supports three known answer testing (KAT) function for testing if the instantiation, reseed and generate function work normally. The programming steps of KAT are depicted below.

CTR_DRBG executes KAT (Known Answer Test) for self-testing

1. Configure CTR_DRBG control register TRNG_CTL for the KAT selection KATSEL (TRNG_CTL[26:25]) and the KAT enable KATEN (TRNG_CTL[24]).
2. Software checks DVIF (TRNG_STS[31]) until it comes to 1. Then software can read the result of KAT from KATPASS (TRNG_STS[27]).

Some Notices of KAT:

1. To avoid violating the first and second notices of CTR_DRBG, we suggest that the test sequence of KAT is instantiation, reseed and generation testing.
2. After fininshing any testing of KAT, the key and initial vector of CTR_DRBG had been changed to the fixed and known value. Thus, user manual instantiation function must be enable when generating the next random number for security.

The TRNG analog macro supports three health test function of entropy source. Start-up test, Repetition count test and Adaptive proportion test. Start-up health test is designed to be performed after powering up, or rebooting, and before the first use of the entropy source. Repetition count test is to quickly detect catastrophic failures that cause the noise source to become stuck on a single output value for a long period of time. Adaptive proportion test is designed to detect a large loss of entropy that might occur as a result of some physical failure or environmental change affecting the noise source. The above three health test are automatic in this TRNG design. These health test will execute automatically after TRNGEN (TRNG_CTL[2]) enabled and test continuously until TRNGEN (TRNG_CTL[2]) disabled. If any health test is fail, TRNGRDY (TRNG_STS[1]) will be low. If TRNGRDY (TRNG_STS[1]) became low over 1ms , ERRIF (TRNG_STS[30]) will rise. Then, users should reset the TRNG IP to recover it.

6.4.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
TRNG Base Address: **TRNG_BA = 0x400B_9000**				
TRNG_CTL	TRNG_BA+0x00	R/W	TRNG Control Register	0x0000_0000
TRNG_CFG	TRNG_BA+0x04	R/W	TRNG Configure Register	0xC200_0000
TRNG_STS	TRNG_BA+0x08	R	TRNG Status Register	0x0000_0060
TRNG_DATA0	TRNG_BA+0x0c	R	TRNG Data Output Word 0 Register	0x0000_0000
TRNG_DATA1	TRNG_BA+0x10	R	TRNG Data Output Word 1 Register	0x0000_0000
TRNG_DATA2	TRNG_BA+0x14	R	TRNG Data Output Word 2 Register	0x0000_0000
TRNG_DATA3	TRNG_BA+0x18	R	TRNG Data Output Word 3 Register	0x0000_0000

6.4.7 Register Description

TRNG Control Register (TRNG_CTL)

Register	Offset	R/W	Description	Reset Value
TRNG_CTL	TRNG_BA+0x00	R/W	TRNG Control Register	0x0000_0000

31	30	29	28	27	26	25	24
DVIEN	ERRIEN	Reserved			KATSEL		KATEN
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					UPDATE	RESEED	INSTANT
7	6	5	4	3	2	1	0
Reserved		MODE		START	TRNGEN	NRST	LDOEN

Bits	Description	
[31]	**DVIEN**	**Data Valid Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[30]	**ERRIEN**	**TRNG Error Interrupt Enable Bit** 0 = Error interrupt Disabled. 1 = Error interrupt Enabled.
[29:27]	**Reserved**	Reserved.
[26:25]	**KATSEL**	**CTR_DRBG Known Answer Test Selection** 0 = Instantiation testing. 1 = Reseed testing. 2 = Generation testing. Others = reserved.
[24]	**KATEN**	**CTR_DRBG Known Answer Test Enable Bit** 0 = Disabled. 1 = Enabled. **Note:** Do not enable START (TRNG_CTL[3]) and KATEN (TRNG_CTL[24]) at the same time.
[23:11]	**Reserved**	Reserved.
[10]	**UPDATE**	**CTR_DRBG User Manual Update Function** 0 = No effect. 1 = Enable CTR_DRBG update function.
[9]	**RESEED**	**CTR_DRBG User Manual Reseed Function** 0 = No effect. 1 = Enable CTR_DRBG reseed function.
[8]	**INSTANT**	**CTR_DRBG User Manual Instant Function**

		0 = No effect. 1 = Enable CTR_DRBG instant function.
[7:6]	**Reserved**	Reserved.
[5:4]	**MODE**	**Random Bit Generator Output Selection** 00 = Output data is from entropy (32-bits). 01 = Output data is from NRBG (128-bits). 10 = Output data is from DRBG (128-bits). 11 = Reserved.
[3]	**START**	**True Random Number Generator Start** 0 = No effect. 1 = Start TRNG. **Note:** Do not enable START (TRNG_CTL[3]) and KATEN (TRNG_CTL[24]) at the same time. This bit is always 0 when it is read back.
[2]	**TRNGEN**	**True Random Number Generator Macro Enable Bit** 0 = TRNG macro Disabled. 1 = TRNG macro Enabled.
[1]	**NRST**	**Negative-edge Trigger Reset** 0 = Keep reset. 1 = No reset.
[0]	**LDOEN**	**LDO Enable Bit** 0 = LDO Disabled. 1 = LDO Enabled.

TRNG Configure Register (TRNG_CFG)

Register	Offset	R/W	Description	Reset Value
TRNG_CFG	TRNG_BA+0x04	R/W	TRNG Configure Register	0xC200_0000

31	30	29	28	27	26	25	24
CTRLEN				**Reserved**		**RESEED_INTERVAL**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RESEED_INTERVAL							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RESEED_INTERVAL							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RESEED_INTERVAL							

Bits	Description	
[31:28]	**CTRLEN**	**CTR_DRBG Bits Length Per Request** CTR_DRBG bits length per request = (2^{CTRLEN} -4) x block length. CTR_DRBG block length is 128-bits (2^7). CTRLEN is 4~12. **Note:** If users select out of this range (4~12), CTRLEN will be set the maximum value = 12.
[27:26]	**Reserved**	Reserved.
[25:0]	**RESEED_INTERVAL**	**Reseed Interval** Maximum number of requests between reseeds. The value is 1 ~ 2^{25}. **Note:** If users select out of this range (1 ~ 2^{25}), reseed_interval will be set the maximum value = 2^{25}.

TRNG Status Register (TRNG_STS)

Register	Offset	R/W	Description	Reset Value
TRNG_STS	TRNG_BA+0x08	R	TRNG Status Register	0x0000_0060

31	30	29	28	27	26	25	24
DVIF	ERRIF	Reserved		KATPASS	Reserved		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	ESAPT	ESRCT	ESSUT	Reserved		TRNGRDY	LDORDY

Bits	Description	
[31]	**DVIF**	**Data Valid Interrupt Flag** 0 = Data is invalid. 1 = Data is valid. A valid random number can be read form TRNG_DATAx. **Note:** This bit is cleared to '0' by reading TRNG_DATA0 ~ TRNG_DATA3. If MODE (TRNG_CTL[5:4]) = '00', this bit is cleared to '0' by reading TRNG_DATA0.
[30]	**ERRIF**	**TRNG Error Interrupt Flag** 0 = No TRNG error. 1 = TRNGRDY became '0' over 1ms , TRNG error interrupt.
[29:28]	**Reserved**	Reserved.
[27]	**KATPASS**	**CTR_DRBG Known Answer Test Pass** 0 = Test fail. 1 = Test pass. **Note:** When users enable KATEN (TRNG_CTL[24]), they can check this bit after DVIF (TRNG_STS[31]) become '1'.
[26:7]	**Reserved**	Reserved.
[6]	**ESAPT**	**Entropy Source Adaptive Proportion Test** 0 = Entropy source is still under testing or test fail. 1 = Test pass.
[5]	**ESRCT**	**Entropy Source Repetition Count Test** 0 = Entropy source is still under testing or test fail. 1 = Test pass.
[4]	**ESSUT**	**Entropy Source Start-Up Test** 0 = Entropy source is still under testing or test fail. 1 = Test pass.
[3:2]	**Reserved**	Reserved.
[1]	**TRNGRDY**	**TRNG Ready Signal**

		0 = True random number generator is not ready. 1 = True random number generator is ready.
[0]	**LDORDY**	**LDO Ready Signal** 0 = LDO is not ready. 1 = LDO is ready.

TRNG Data Output Word x Register (TRNG_DATAx)

Register	Offset	R/W	Description	Reset Value
TRNG_DATA0	TRNG_BA+0x0c	R	TRNG Data Output Word 0 Register	0x0000_0000
TRNG_DATA1	TRNG_BA+0x10	R	TRNG Data Output Word 1 Register	0x0000_0000
TRNG_DATA2	TRNG_BA+0x14	R	TRNG Data Output Word 2 Register	0x0000_0000
TRNG_DATA3	TRNG_BA+0x18	R	TRNG Data Output Word 3 Register	0x0000_0000

31	30	29	28	27	26	25	24
DATA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DATA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DATA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATA							

Bits	Description	
[31:0]	**DATA**	**True Random Number Generator Data (Read Only)** The DATA stores the output data generated by TRNG, it can be read when DVIF (TRNG_STS[31]) become 1 and can be read only once. TRNG_DATA0 stores Entropy / NRBG word 0 / DRBG word 0. TRNG_DATA1 stores NRBG word 1 / DRBG word 1. TRNG_DATA2 stores NRBG word 2 / DRBG word 2. TRNG_DATA3 stores NRBG word 3 / DRBG word 3.

6.5 RRAM Memory Controller (RMC)

6.5.1 Overview

The RRAM memory controller (RMC) is equipped with 512/256/128/64 Kbytes on-chip embedded RRAM for application and configurable Data RRAM to store some application dependent data. Thus, the total size of application rom (APROM) is 512/256/128/64 Kbytes. This chip supports Dual-bank RRAM macro for safe firmware upgrade. A User Configuration block provides for system initiation. A 8 Kbytes loader ROM (LDROM) is used for In-System-Programming (ISP) function. A 16 Kbytes Bootloader consists of native ISP functions and secure boot function. A 2 Kbytes one-time-program ROM (OTP) is used for recording one-time-program data. 4 Kbytes cache with zero wait cycle is used to improve RRAM access performance. This chip also supports In-Application-Programming (IAP) function. User switches the code executing without chip reset after the embedded RRAM is updated.

6.5.2 Features

- Supports Dual-bank RRAM macro for safe firmware upgrade
- Supports Dual-bank remapping
- Supports 512/256/128/64 Kbytes application ROM (APROM)
- Supports 8 Kbytes loader ROM (LDROM)
- Supports 4 XOM (eXecution Only Memory) regions to conceal user program in APROM
- Supports Data RRAM with configurable memory size
- Supports 256 bytes User Configuration block to control system initiation
- Supports 2 Kbytes one-time-program ROM (OTP)
- Supports Bootloader with native In-System-Programming (ISP) functions
- Supports Secure Boot function for code integrity and authenticity
- Supports 32-bit and word line RRAM programming function
- Supports CRC32 checksum calculation function
- Supports RRAM all one verification function
- Supports In-System-Programming (ISP) / In-Application-Programming (IAP) to update embedded RRAM memory
- Supports cache memory to improve RRAM access performance and reduce power consumption

RMC Features	M2L31xxDAE	M2L31xx4AE
Dual-bank RRAM macro	●	

Table 6.5-1 RMC Features Comparison Table at Different Chip

6.5.3 Block Diagram

RMC consists of AHB slave interface, cache memory controller, RRAM control registers, RRAM initialization controller, RRAM operation control and embedded RRAM memory. The block diagram of RRAM memory controller is shown as follows.

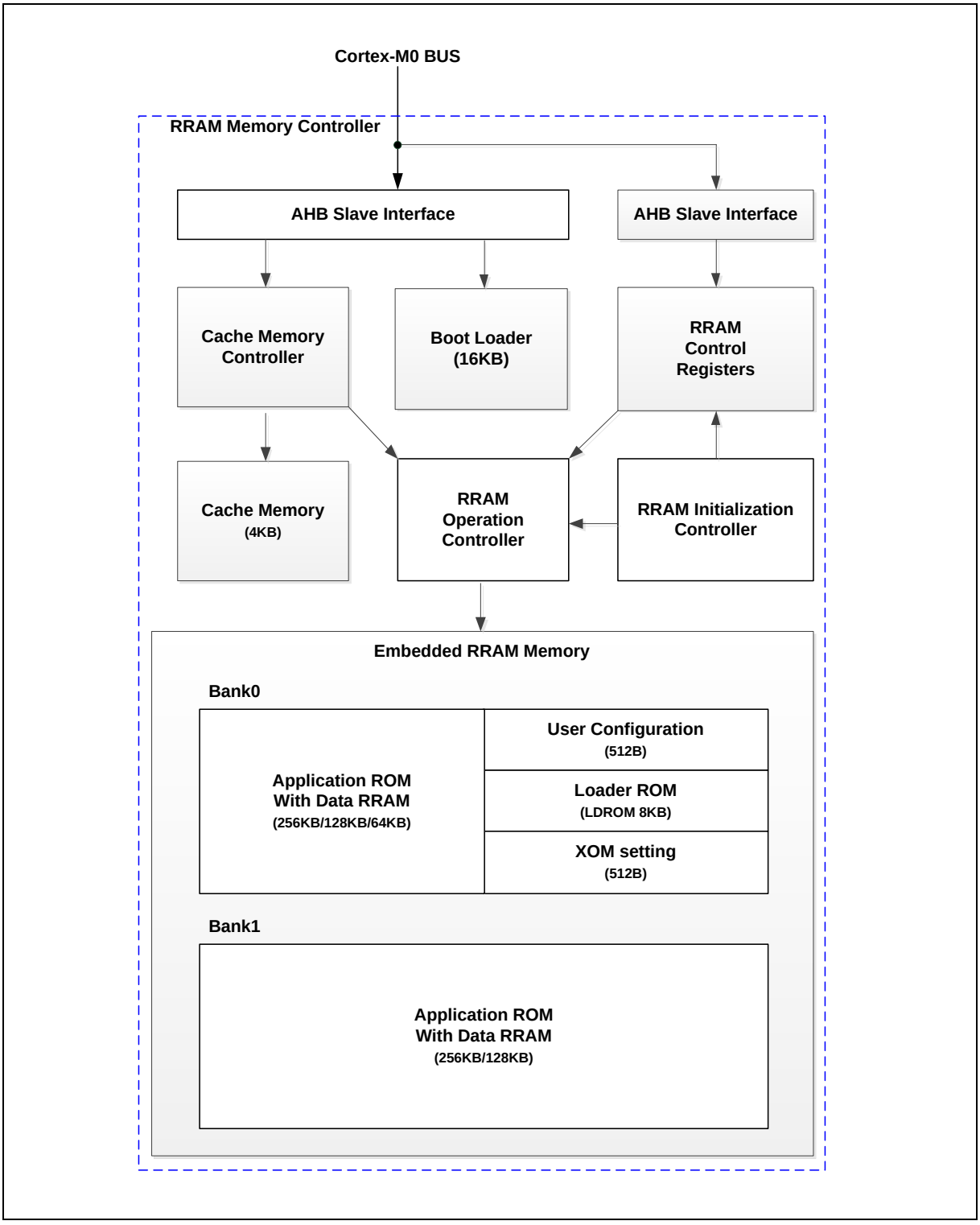


Figure 6.5-1 RRAM Memory Controller Block Diagram

Cache Memory Controller

A 4 Kbytes cache with zero wait cycle is implemented between Cortex-M0 CPU and embedded RRAM memory. This cache memory controller improves the RRAM access performance and reduces power consumption of the embedded RRAM memory.

Bootloader

The Bootloader is 16 Kbytes, and consists of native ISP functions to update embedded RRAM memory. The Bootloader content is read only, but not programmable.

RRAM Control Registers

All of ISP control and status registers are in the RRAM control registers. The detailed registers description is in the Register Description section

RRAM Initialization Controller

When chip is powered on or active from reset, the RRAM initialization controller will start to access RRAM automatically and check the RRAM stability, and also reload User Configuration content to the RRAM control registers for system initiation.

RRAM Operation Controller

The RRAM operations, such as RRAM mass erase, RRAM program, and RRAM read operation, have specific control timing for embedded RRAM memory. The RRAM operation controller generates those control timing by requested from the cache memory controller, the RRAM control registers and the RRAM initialization controller.

Embedded RRAM Memory

The embedded RRAM memory is the main memory for user application code and parameters. It is consists of the user configuration block, 8 Kbytes LDROM, two 512B XOM setting pages, 2 Kbytes OTP and 512 Kbytes/256 Kbytes/128 Kbytes/64 Kbytes APROM with Data RRAM.

6.5.4 Functional Description

RMC functions include the memory organization, boot selection,IAP, ISP, the embedded RRAM programming, and checksum calculation. The RRAM memory map and system memory map are also introduced in the memory organization.

6.5.4.1 Memory Organization

The RMC memory consists of the embedded RRAM memory. The embedded RRAM memory is programmable, and includes APROM, LDROM, XOM setting page, Data RRAM, the User Configuration block and OTP. The address map includes RRAM memory map and three system address maps: LDROM with IAP, APROM with IAP and Bootloader with IAP functions.

Bank	RRAM Memory Block	Address Range
0	APROM with 512 Kbytes	0x000_0000 ~ 0x003_FFFF
	APROM with 256 Kbytes	0x000_0000 ~ 0x001_FFFF
	LDROM	0xF10_0000 ~ 0xF10_1FFF
	XOM setting	0xF20_0000 ~ 0xF20_01FF
	User Configuration	0xF30_0000 ~ 0xF30_01FF
	OTP	0xF31_0000 ~ 0xF31_0BFF
1	APROM with 512 Kbytes	0x004_0000 ~ 0x007_FFFF
	APROM with 256 Kbytes	0x002_0000 ~ 0x003_FFFF

Table 6.5-2 Dual-bank Block Address Range

Bank	RRAM Memory Block	Address Range
0	APROM with 256 Kbytes	0x000_0000 ~ 0x003_FFFF
	APROM with 128 Kbytes	0x000_0000 ~ 0x001_FFFF
	APROM with 64 Kbytes	0x000_0000 ~ 0x000_FFFF
	LDROM	0xF10_0000 ~ 0xF10_1FFF
	XOM setting	0xF20_0000 ~ 0xF20_01FF
	User Configuration	0xF30_0000 ~ 0xF30_01FF
	OTP	0xF31_0000 ~ 0xF31_0BFF

Table 6.5-3 Single-bank Block Address Range

6.5.4.2 Dual-bank Architecture

The M2L31 RRAM also provides Dual-bank architecture where APROM is separated into Bank0 and Bank1. Bank0's RRAM address range is from 0x0 to 0x1FFFF/3FFFF while Bank1's RRAM address range is from 0x20000/40000 to 0x3FFFF/7FFFF. In such Dual-bank architecture, a firmware upgrade mechanism is provided, which uses APROM Bank0 as firmware executing region and APROM Bank1 as storage region for newer version of firmware. The firmware in Bank0 can erase Bank1 and program new firmware to Bank1 without stopping instruction fetching form Bank0. This is so called read-while-write (RWW), which provides the ability for firmware to keep software services while firmware updating. The firmware upgrade mechanism here also provides a rollback process to roll back to the original firmware if a new firmware woks abnormally. Such mechanism ensures the firmware to be safe to update in any circumstance.

6.5.4.3 LDROM APROM and Data RRAM

LDROM is designed for a loader to implement In-System-Programming (ISP) function by user. LDROM is a 8 Kbytes embedded RRAM memory, the RRAM address range is from 0x0F10_0000 to 0x0F10_1FFF. APROM is main memory for user applications. APROM size is 512 Kbytes/256 Kbytes/128 Kbytes/64 Kbytes. Data RRAM is used to store application parameters (not instruction). Data RRAM is shared with APROM and size is configurable. The base address of Data RRAM is determined by DFBA (CONFIG1[19:0]).

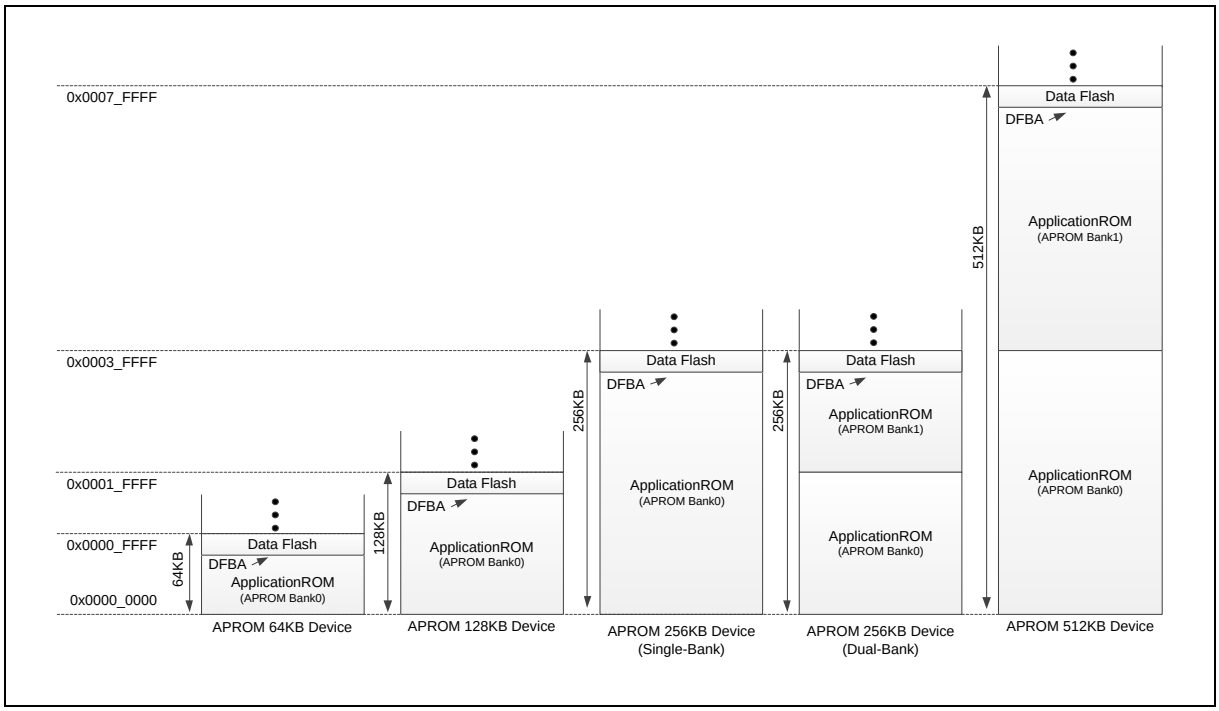


Figure 6.5-2 Data RRAM Shared with APROM Examples

6.5.4.4 Physical and Virtual Address Concept

For OTA application, the memory space concept is a little different form RMC commands and CPU viewpoint.

- From RMC command viewpoint, physical address concept is adopted. All of RMC commands regard APROM memory as physical address except “remapping” command. It means APROM memory will be flat memory model regardless of memory remapping state.
- From Data RRAM viewpoint, virtual address concept is adopted.
- From CPU viewpoint, virtual address concept is adopted. APROM memory space is separated into 2 segments. As runtime stage, the virtual address depends on what address operation model (OP0, OP1) is, as shown in Figure 6.5-3.

6.5.4.5 APROM Reboot Address Operation Model Selection

In M2L31, to activate APROM reboot with different address operation model, use address operation model selection command 0x2C with RMC_ISPDAT “0x5AA5_5AA5” and RMC_ISPADDR ”0x0”(Address OP0) or ”0x1”(Address OP1) to select address operation model.

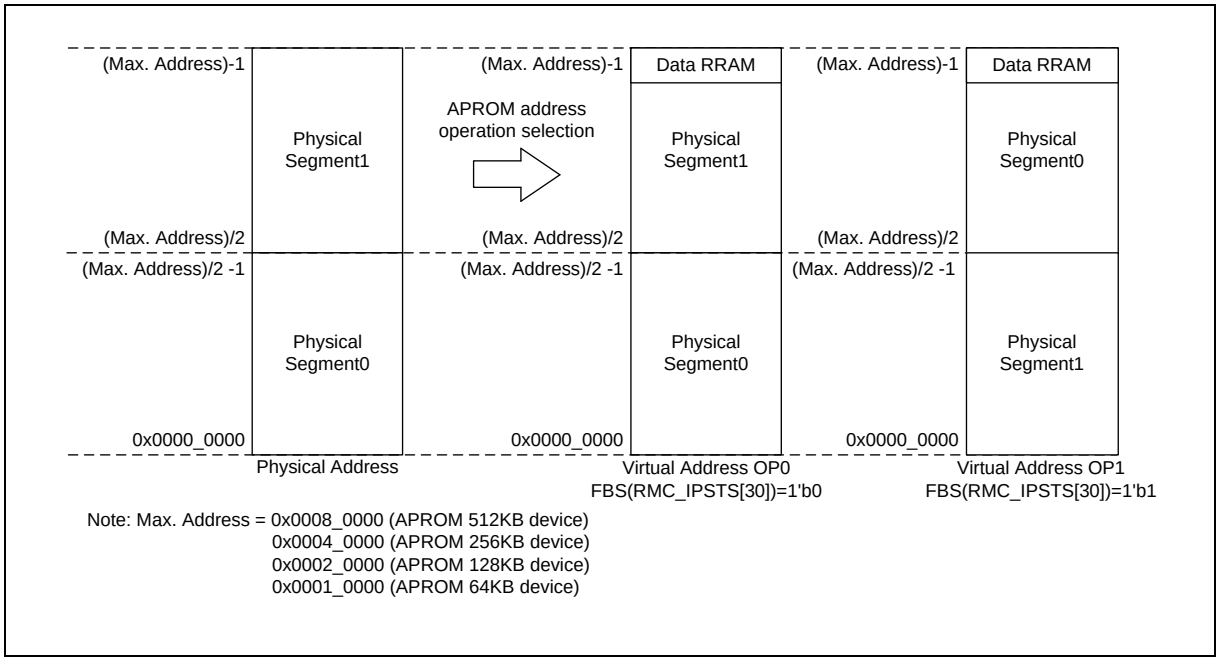


Figure 6.5-3 Address Operation Model

6.5.4.6 User Configuration Block

User Configuration block is internal programmable configuration area for boot options, such as RRAM security lock, boot select, brown-out voltage level, and Data RRAM base address. It works like a fuse for power on setting. It is loaded from RRAM memory to its corresponding control registers during chip power on. User can set these bits according to different application requests. User Configuration block can be updated by ISP function and located at 0x0F30_0000 with four 32 bits words (CONFIG0, CONFIG1, CONFIG2 and CONFIG3). Any change on User Configuration block will take effect after system reboot.

CONFIG0 (Address = 0x0F30_0000)

31	30	29	28	27	26	25	24
CWDTEN[2]	Reserved						
23	22	21	20	19	18	17	16
CBOV			CBORST	CBODEN	Reserved		
15	14	13	12	11	10	9	8
Reserved				ICELOCK	Reserved		
7	6	5	4	3	2	1	0
CBS	Reserved		CWDTEN[1:0]		Reserved	LOCK	DFEN

Bits	Descriptions	
[31]	**CWDTEN[2]**	**Watchdog Timer Hardware Enable Bit** When watchdog timer hardware enable function is enabled, the watchdog enable bit WDTEN (WDT_CTL[7]) and watchdog reset enable bit RSTEN (WDT_CTL[1]) is set to 1 automatically after power on. The clock source of watchdog timer is force at LIRC and LIRC can't be disabled. **CWDTEN[2:0]** is CONFIG0[31][4][3], 111 = WDT hardware enable function is inactive. Others = WDT hardware enable function is active. WDT clock is always on.
[30:27]	**Reserved**	Reserved.
[23:21]	**CBOV**	**Brown-out Voltage Selection** 000 = Brown-out voltage is 1.6V. 001 = Brown-out voltage is 1.8V. 010 = Brown-out voltage is 2.0V. 011 = Brown-out voltage is 2.2V. 100 = Brown-out voltage is 2.4V. 101 = Brown-out voltage is 2.6V. 110 = Brown-out voltage is 2.8V. 111 = Brown-out voltage is 3.0V.
[20]	**CBORST**	**Brown-out Reset Enable Bit** 0 = Brown-out reset Enabled after powered on. 1 = Brown-out reset Disabled after powered on.
[19]	**CBODEN**	**Brown-out Detector Enable Bit** 0= Brown-out detect Enabled after powered on. 1= Brown-out detect Disabled after powered on.
[18:12]	**Reserved**	Reserved.
[11]	**ICELOCK**	**ICE Lock Bit** This bit is only used to disable ICE function. User may use it with LOCK (CONFIG0[1]) bit or ALOCK (CONFIG2[7:0]) bits to increase security level. 0 = ICE function Disabled. 1 = ICE function Enabled.
[10:8]	**Reserved**	Reserved.

[7]	**CBS**	**Chip Booting Selection** IAP mode is supported in this chip, and the code in LDROM and APROM can be called by each otherWhen CBS = 0, the LDROM base address is mapping to 0x100000 and APROM base address is mapping to 0x0. User could access both APROM and LDROM without boot switching.. 0 = Boot from LDROM with IAP mode. 1 = Boot from APROM with IAP mode.
[6:4]	**Reserved**	Reserved.
[4:3]	**CWDTEN**	**Watchdog Timer Hardware Enable Bit** When watchdog timer hardware enable function is enabled, the watchdog enable bit WDTEN (WDT_CTL[7]) and watchdog reset enable bit RSTEN (WDT_CTL[1]) is set to 1 automatically after power on. The clock source of watchdog timer is force at LIRC and LIRC can't be disable. **CWDTEN[2:0]** is CONFIG0[31][4][3], 111 = WDT hardware enable function is inactive. Others = WDT hardware enable function is active. WDT clock is always on.
[2]	**Reserved**	Reserved.
[1]	**LOCK**	**Security Lock Control** 0 = RRAM memory content is locked. 1 = RRAM memory content is not locked if ALOCK(CONFIG2[7:0]) and KSPSBLOCK (CONFIG2[15:8]) are 0x5A. When RRAM data is locked by LOCK,user can look the lock effect of RMC up in lock effect tables.
[0]	**DFEN**	**Data RRAM Enable Bit** The Data RRAM is shared with APROM, and the base address of Data RRAM is decided by DFBA (CONFIG1[19:0]) when DFEN is 0. 0 = Data RRAM Enabled. 1 = Data RRAM Disabled.

Note: The config bits should be 1 if reserved.

CONFIG1 (Address = 0x0F30_0004)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				DFBA			
15	14	13	12	11	10	9	8
DFBA							
7	6	5	4	3	2	1	0
DFBA							

Bits	Descriptions	
[31:20]	**Reserved**	Reserved.
[19:0]	**DFBA**	**Data RRAM Base Address** This register works only when DFEN (CONFIG0[0]) is set to 0. If DFEN (CONFIG0[0]) is set to 0, the Data RRAM base address is defined by user. Data RRAM Base Address must be page-aligned, it is mandatory to keep bit 11-0 as 0. **Note:** The page size is 4 Kbytes.

Note: The config bits should be 1 if reserved.

CONFIG2 (Address = 0x0F30_0008)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
KSPLOCK							
7	6	5	4	3	2	1	0
ALOCK							

Bits	Descriptions	
[31:16]	**Reserved**	Reserved.
[15:8]	**KSPLOCK**	**Key Store Protection Lock** 0x5A = RRAM memory content is unlocked if LOCK (CONFIG0[1]) is 1 and ALOCK (CONFIG2[7:0]) is 0x5A. Others = RRAM memory content is also locked. When KSPLOCK ≠ 0x5A with ICE, only Chip Erase ISP command can be performed via ICE. When RRAM data is locked by KSPLOCK, user must look the lock effect of RMC up in lock effect tables. When KSPLOCK is enabled, using mass erase would cause Key Store to revoke all OTP key.
[7:0]	**ALOCK**	**Advance Security Lock Control** 0x5A = RRAM memory content is unlocked if LOCK (CONFIG0[1]) is 1. Others = RRAM memory content is locked. When RRAM data is locked by ALOCK,user must look the lock effect of RMC up in lock effect tables.

Note: The config bits should be 1 if reserved.

CONFIG3 (Address = 0x0F30_000C)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							BOOTSP
15	14	13	12	11	10	9	8
Reserved			SPIISPEN	I2CISPEN	CANISPEN	USBISPDIS	UARTISPEN
7	6	5	4	3	2	1	0
Reserved						UART0PSL	

Bits	Descriptions	
[31:17]	**Reserved**	Reserved.
[16]	**BOOTSP**	**Boot Speed Select Bit** This field defines secure boot execution CPU speed. User can select higher speed for fast boot up or lower speed for lower power comsumption during chip booting up. 0= Use HIRC12M. 1= Use PLL configure as 72 MHz.
[15:13]	**Reserved**	Reserved.
[12]	**SPIISPEN**	**SPI ISP Mode Enable Bit** The pins used in SPI ISP mode are SPI0 MOSI on PA.0, MISO on PA.1, CLK on PA.2 and SS on PA.3. 0= SPI ISP mode Disabled. 1= SPI ISP mode Enabled.
[11]	**I2CISPEN**	**I^{2}C ISP Mode Enable Bit** The pins used in I^{2}C ISP mode are I2C0 SDA on PB.4 and I2C0 SCL on PB.5. 0= I^{2}C ISP mode Disabled. 1= I^{2}C ISP mode Enabled.
[10]	**CANISPEN**	**CAN ISP Mode Enable Bit** The pins used in CAN ISP mode are CAN0 RXD on PA.4 and CAN0 TXD on PA.5. 0= CAN ISP mode Disabled. 1= CAN ISP mode Enabled.
[9]	**USBISPDIS**	**USB ISP Mode Disable Bit** The USB ISP mode runs in HS-USBD mode. 0= USB ISP mode Enabled. 1= USB ISP mode Disabled.
[8]	**UARTISPEN**	**UART ISP Mode Enable Bit** 0= UART ISP mode Disabled. 1= UART ISP mode Enabled.
[7:2]	**Reserved**	Reserved.

[1:0]	**UART0PSL**	**Bootloader UART0 Multi-function Pin Select** 00 = UART0_TXD (PB.9), UART0_RXD (PB.8). 01 = UART0_TXD (PA.7), UART0_RXD (PA.6). 10 = Reserved. 11 = UART0_TXD (PB.13), UART0_RXD (PB.12). **Note:** When chip is reset, the default value comes from CONFIG3[1:0]

Note: The config bits should be 1 if reserved.

6.5.4.7 Execution Only Memory (XOM)

The execution only memory (XOM) is used to store instructions for security application which are not allowed for data access via AHB-Bus. There are four XOM regions in APROM at most. To define a new XOM region, user must complete XOM settings in XOM setting page (0xF20_0000 ~ 0xF20_0008 for XOM region 0; 0xF20_0010 ~ 0xF20_0018 for XOM region 1 ; 0xF20_0020 ~ 0xF20_0028 for XOM region 2 ; 0xF20_0030 ~ 0xF20_0038 for XOM region 3) via In-System-Programming (ISP) function. When user setting XOM region,need to set Base address first then set XOM page size, as shown in Figure 6.5-4. User should not set XOM at region where chip can boot from. User should not set the XOMs overlap each other, because it would cause XOM unable to be normal active. After setting and restaring RRAM initialization, user can check XOM status from RMC_XOMSTS and check each XOM range (i.e., base and size) from RMC_XOMR0STS~RMC_XOMR3STS. In ICP function or WRITER mode, user can execute command 0x0 to read XOM setting page and check XOM status. User need to do chip reset (RRAM initial) after setting or erasing XOM.

When using XOM setting page and XOM regions, users must follow the limitations below:

- All XOM regions must be located in APROM with page size alignment.
- Any XOM region cannot be programmed after its XOM setting is active.
- Any XOM setting cannot be changed during runtime.
- The base address of all XOM cannot be set as zero (first page).
- Only use RRAM 32-bit program command to program XOM setting page
- Clear XOM setting and XOM region by mass erase command or XOM page erase function (a special erase command for XOM)
- User must restart RRAM initialization to activate any new XOM setting.
- ICE cannot step in XOM only if debug mode.

Once the XOM region is active, user can only adopt mass erase command or XOM page erase function to clear the XOM setting of one XOM region. To execute XOM page erase function, user must perform RRAM Page Erase (0x22) by writing the base address of the target XOM region (address in APROM, not in XOM setting page) in RMC_ISPADDR and the constant value 0x0055aa03 in RMC_ISPDAT. Note that, the valid ISP commands in every active XOM region are chip erase, page erase, chksum and read CID/DID. The XOM setting page is not accessed, except for chip erase, word program and 32-bit read.

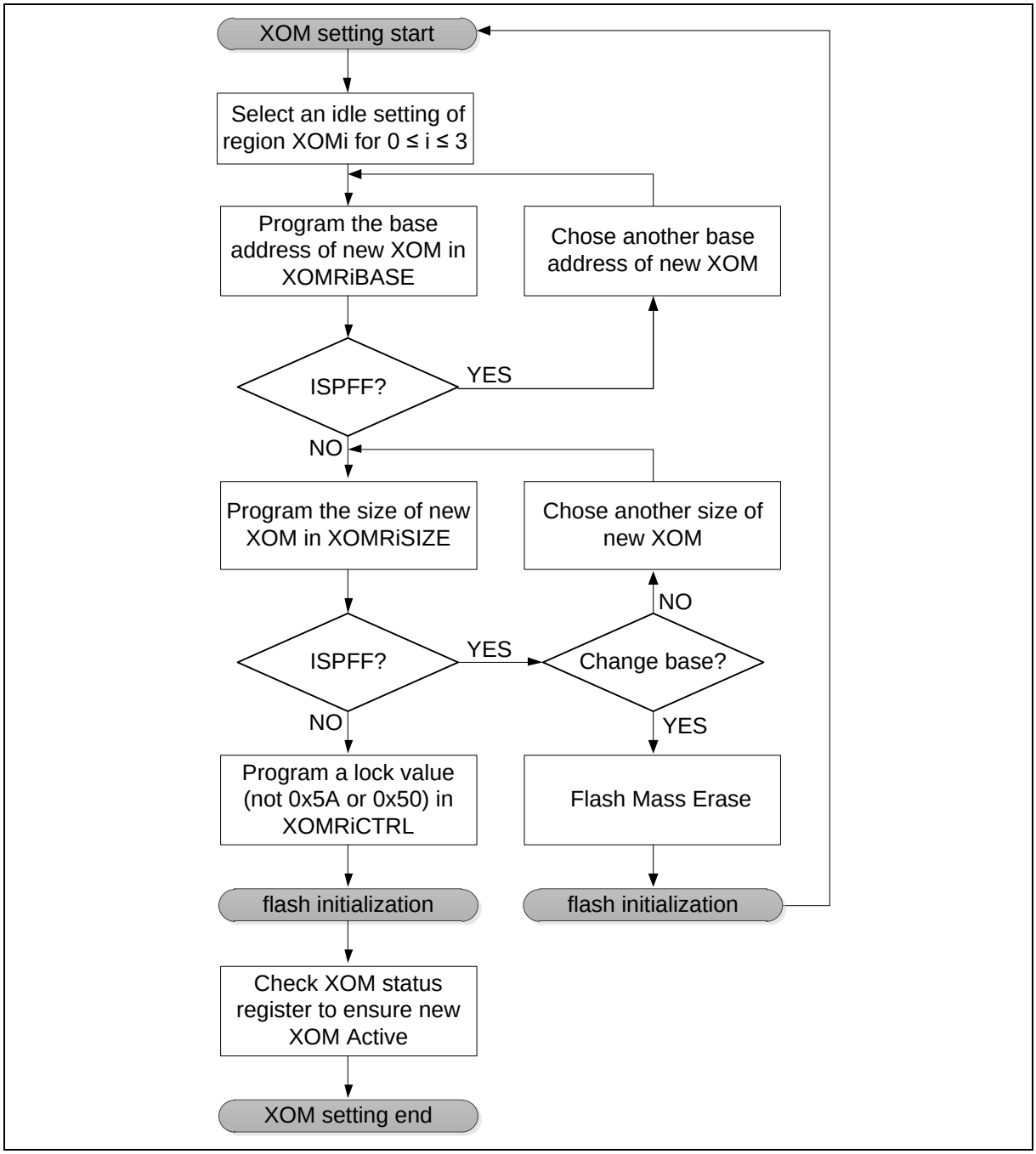


Figure 6.5-4 New XOM Setting Flow

XOMR0 BASE (Address = 0x0F20_0000)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR0BASE							
15	14	13	12	11	10	9	8
XOMR0BASE							
7	6	5	4	3	2	1	0
XOMR0BASE							

Bits	Descriptions	
[31:24]	**Reserved**	Reserved.
[23:0]	**XOMR0BASE**	**Base Address of XOM Region 0** XOMR0BASE must be page-aligned. **Note:** The page size is 4 Kbytes.

XOMR0SIZE (Address = 0x0F20_0004)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR0SIZE							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR0SIZE**	**Page Number of XOM Region 0** XOMR0SIZE must be page-aligned. The XOMR0SIZE minimum value is 1 page. If XOMR0SIZE is written to 0, the register value will be set to 1. **Note:** The page size is 4 Kbytes.

XOMR0CTRL (Address = 0x0F20_0008)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR0CTRL							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR0CTRL**	**Control of XOM Region 0** 0x5a = XOM region 0 is off. 0x50= XOM is in debug mode. Others = XOM region 0 is active. **Note:** The active state has come into effect after power-on or reset cycle.

XOMR1BASE (Address = 0x0F20_0010)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR1BASE							
15	14	13	12	11	10	9	8
XOMR1BASE							
7	6	5	4	3	2	1	0
XOMR1BASE							

Bits	Descriptions	
[31:24]	**Reserved**	Reserved.
[23:0]	**XOMR1BASE**	**Base Address of XOM Region 1** XOMR1BASE must be page-aligned. **Note:** The page size is 4 Kbytes.

XOMR1SIZE (Address = 0x0F20_0014)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR1SIZE							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR1SIZE**	**Page Number of XOM Region 1** XOMR1SIZE must be page-aligned. The XOMR1SIZE minimum value is 1 page. If XOMR1SIZE is written to 0, the register value will be set to 1. **Note:** The page size is 4 Kbytes.

XOMR1CTRL (Address = 0x0F20_0018)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR1CTRL							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR1CTRL**	**Control of XOM Region 1** 0x5a = XOM region 1 is off. 0x50= XOM is in debug mode. Others = XOM region 1 is active. **Note:** The active state has come into effect after power-on or reset cycle.

XOMR2BASE (Address = 0x0F20_0020)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR2BASE							
15	14	13	12	11	10	9	8
XOMR2BASE							
7	6	5	4	3	2	1	0
XOMR2BASE							

Bits	Descriptions	
[31:24]	**Reserved**	Reserved.
[23:0]	**XOMR2BASE**	**Base Address of XOM Region 2** XOMR2BASE must be page-aligned. **Note:** The page size is 4 Kbytes.

XOMR2SIZE (Address = 0x0F20_0024)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR2SIZE							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR2SIZE**	**Page Number of XOM Region 2** XOMR2SIZE must be page-aligned. The XOMR2SIZE minimum value is 1 page. If XOMR2SIZE is written to 0, the register value will be set to 1. **Note:** The page size is 4 Kbytes.

XOMR2CTRL (Address = 0x0F20_0028)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR2CTRL							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR2CTRL**	**Control of XOM Region 2** 0x5a = XOM region 2 is off. 0x50= XOM is in debug mode. Others = XOM region 2 is active. **Note:** The active state has come into effect after power-on or reset cycle.

XOMR3BASE (Address = 0x0F20_0030)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR3BASE							
15	14	13	12	11	10	9	8
XOMR3BASE							
7	6	5	4	3	2	1	0
XOMR3BASE							

Bits	Descriptions	
[31:24]	**Reserved**	Reserved.
[23:0]	**XOMR3BASE**	**Base Address of XOM Region 3** XOMR3BASE must be page-aligned. **Note:** The page size is 4 Kbytes.

XOMR3SIZE (Address = 0x0F20_0034)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR3SIZE							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR3SIZE**	**Page Number of XOM Region 3** XOMR3SIZE must be page-aligned. The XOMR3SIZE minimum value is 1 page. If XOMR3SIZE is written to 0, the register value will be set to 1. **Note:** The page size is 4 Kbytes.

XOMR3CTRL (Address = 0x0F20_0038)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR3CTRL							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR3CTRL**	**Control of XOM Region 3** 0x5a = XOM region 3 is off. 0x50= XOM is in debug mode. Others = XOM region 3 is active. **Note:** The active state has come into effect after power-on or reset cycle.

6.5.4.8 One Time Program Memory (OTP)

The One time program memory (OTP) is used to store the important information that are not allowed user to modify them twice. The maximum size of OTP data are 2 Kbytes from 0xF31_0000 ~ 0xF31_07FF. Every 64 bits of OTP data has one 32 bits “LOCK BIT” from 0xF31_0800 to 0xF31_0BFF, as shown in Figure 6.5-5. Thus, the OTP is 3 Kbytes with location address 0xF31_0000 ~ 0xF31_0BFF.

The “LOCK BIT” is for recording if the programed address had been locked (LOCK BIT≠0xFFFF_FFFF) or not (LOCK BIT= 0xFFFF_FFFF) in OTP. For example, when LOCK BIT0 is not “0xFFFFFFFF”, OTP0 cannot be programmed again, regardless its content is all 1 or not. The “RRAM mass erase command” are never allowed to be executed in OTP. Before updating the content of OTP from 0xF31_0000 to 0xF31_07FF, users must check their “LOCK BIT” firstly. After finishing programing OTP data, users must write a non-0xFFFF_FFFF in the “LOCK BIT” of the above programed data to make sure that no one can modify them. OTP is only readable by ISP command and un-readable by CPU read directly.

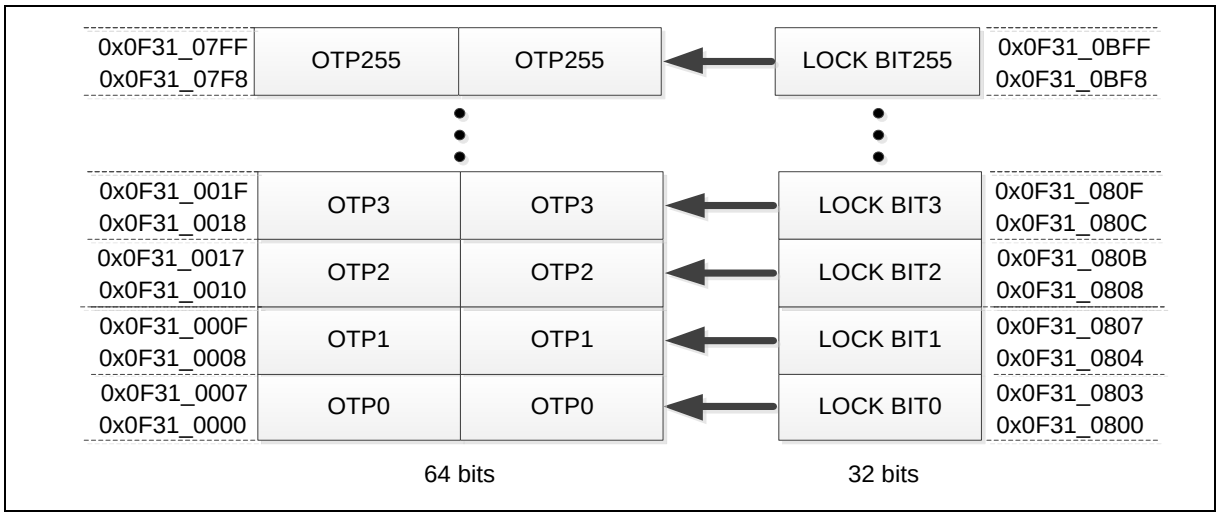


Figure 6.5-5 OTP Memory Map

6.5.4.9 RRAM Memory Map

The RRAM memory map is different from system memory map. The system memory map is used by CPU fetch code or data from RMC memory. The RRAM memory map is used for ISP function to read, program or erase RMC memory. The RRAM memory map is as Figure 6.5-6.

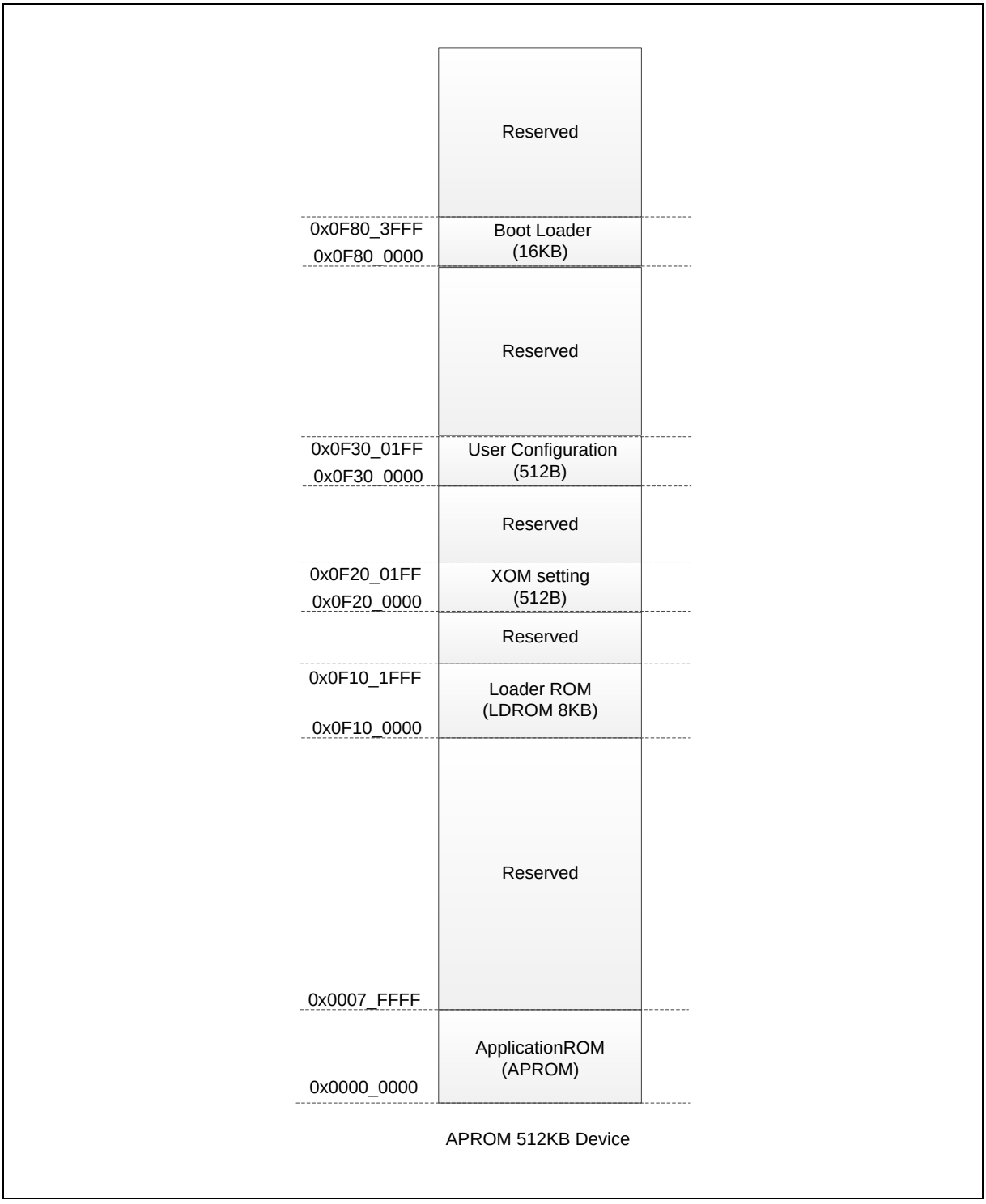


Figure 6.5-6 RRAM Memory Map

6.5.4.10 System Memory Map with IAP Mode

The system memory map is used by CPU to fetch code or data from RMC memory.The Data RRAM is shared with APROM and the Data RRAM base address is defined by CONFIG1. The content of CONFIG1 is loaded into DFBA (Data RRAM Base Address Register) at the RRAM initiation.

The DFBA~(0x0003_FFFF/0x0007_FFFF) is the Data RRAM region for Cortex-M0 data access, and 0x0000_0200~(DFBA-1) is APROM region for Cortex-M0 instruction access when APROM size is 256/512 Kbytes, respectively.

The address from 0x0000_0000 to 0x0000_01FF is called system memory vector.

Figure 6.5-7 shows the system memory map with IAP mode.

APROM and LDROM can map to the system memory vector for CPU start up. There are two kinds of system memory map with IAP mode when chip booting: (1) LDROM with IAP and (2) APROM with IAP

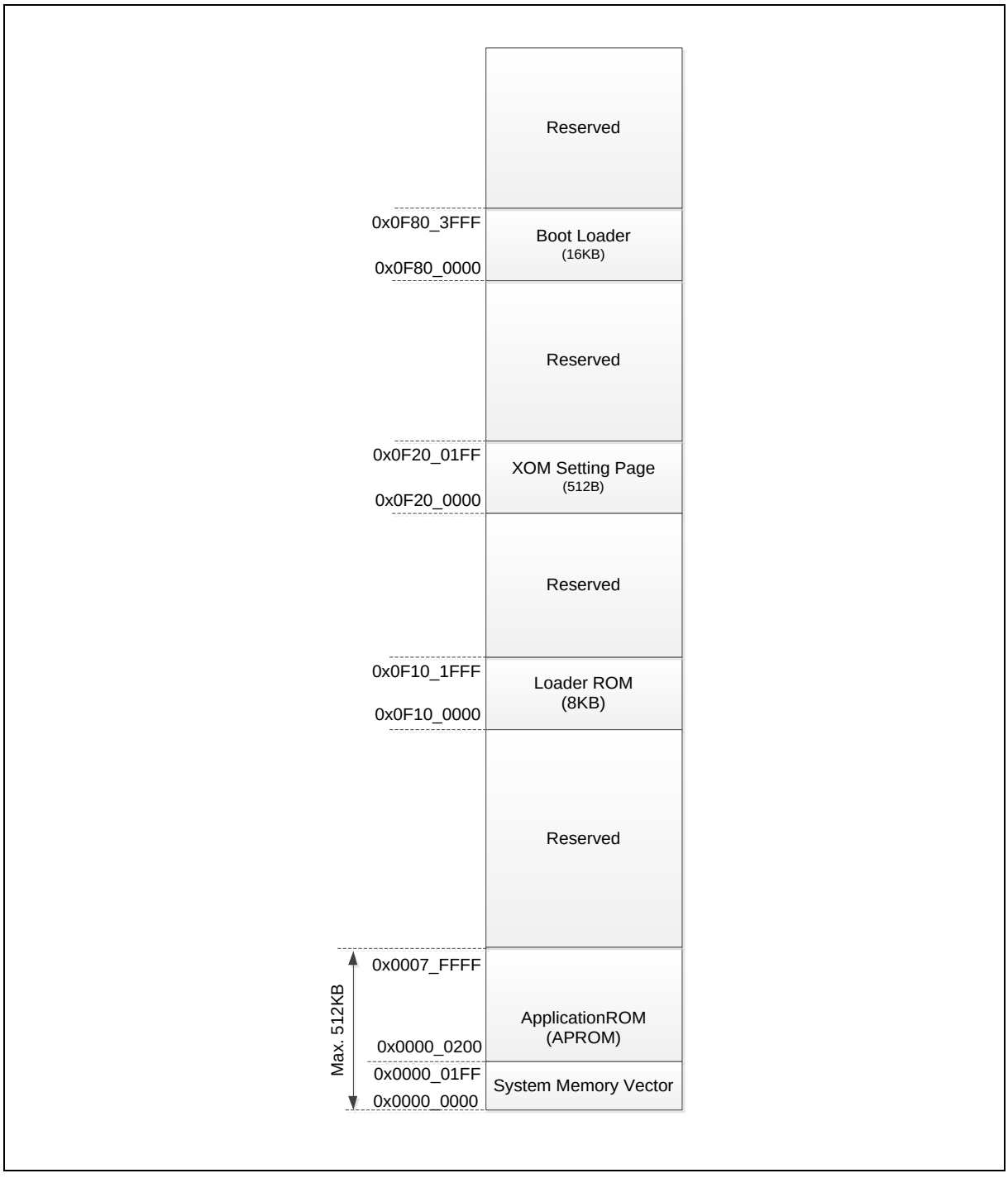


Figure 6.5-7 System Memory Map with IAP Mode

In LDROM with IAP mode, the LDROM (0x0F10_0000~0x0F10_01FF) is mapping to the system memory vector for Cortex-M0 instruction or data access. Figure 6.5-8 shows the memory map of LDROM with IAP mode.

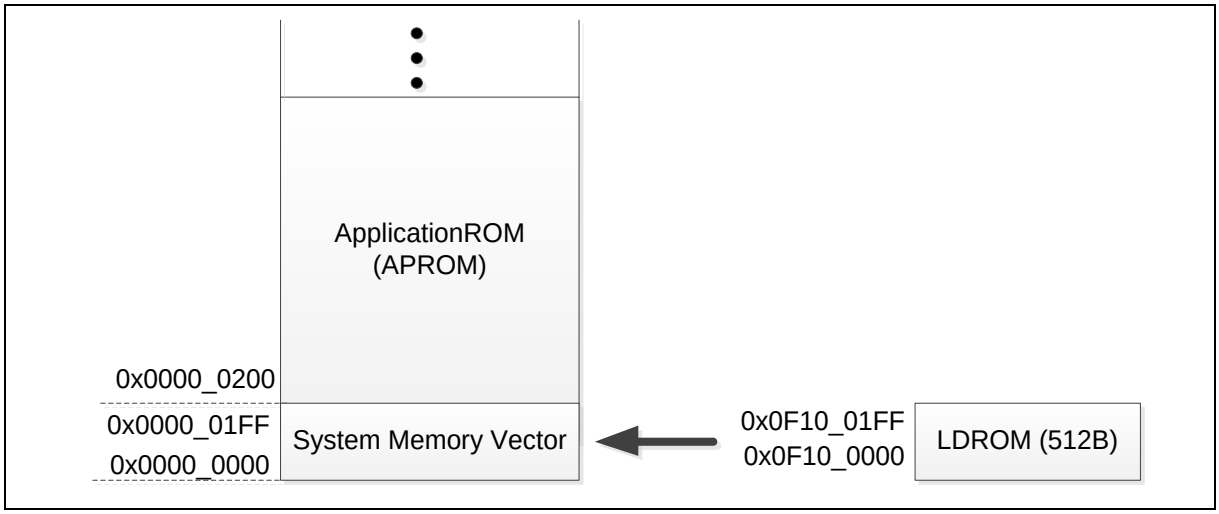


Figure 6.5-8 LDROM with IAP Mode

In APROM with IAP mode, the APROM (0x0000_0000~0x0000_01FF) is mapping to the system memory vector for Cortex-M0 instruction or data access. Figure 6.5-9 shows the memory map of APROM with IAP mode.

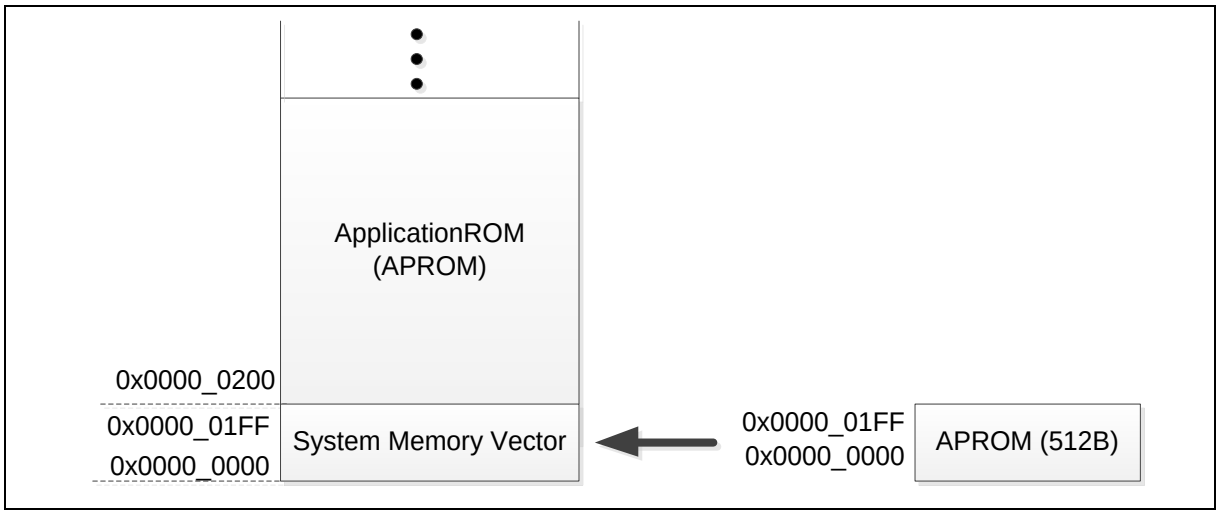


Figure 6.5-9 APROM with IAP Mode

In system memory map with IAP mode, APROM, LDROM, APROM and Bootloader can remap to the system memory vector when CPU running. User can write the target remap address to RMC_ISPADDR register and then trigger ISP procedure with the “Vector Remap” command (0x2E). In VECMAP (RMC_ISPSTS[23:9]), shows the finial system memory vector mapping address.

6.5.4.11 In-Application-Programming (IAP)

The M2L31 series provides In-Application-Programming (IAP) function for user to switch the system memory vector code executing between APROM and LDROM. User can enable the IAP function by booting chip and setting the chip boot selection bits in in CBS (CONFIG0[7]) as 1 or 0.

When chip boots with IAP function is enabled, any executable code (align to 512 bytes) is allowed to map to the system memory vector any time. User can change the remap address to RMC_ISPADDR

and then trigger ISP procedure with the “Vector Remap” command.

6.5.4.12 In-System-Programming (ISP)

The M2L31 series supports In-System-Programming (ISP) function allowing the embedded RRAM memory to be reprogrammed under software control. ISP is performed without removing the microcontroller from the system through the firmware and on-chip connectivity interface, such as UART, USB, I^2C, SPI, and CAN (depended on chip feature). The target RRAM memory space that ISP function operates cannot be across banks. Table 6.5-4 lists all ISP commands.

The ISP provides the following functions for embedded RRAM memory.

- Supports RRAM load data buffer function
- Supports RRAM clear data buffer function
- Supports RRAM data program function
- Supports RRAM data read function
- Supports company ID read function
- Supports device ID read function
- Supports unique ID read function
- Supports memory CRC32 checksum calculation function
- Supports RRAM all one verification function
- Supports system memory vector remap function

ISP Commands

ISP Command	RMC_ISPCMD	RMC_ISPADDR	RMC_ISPDAT
RRAM 32-bits Program (This command will write data of data buffer into RRAM, data buffer can’t be empty when this command is assert)	0x21	Valid address of RRAM memory organization	N/A
RRAM 32-bits Load Data Buffer (This command should be asserted after 0x2F)	0x27	Valid address of RRAM memory organization	RMC_ISPDAT: Data
RRAM 32-bits Clear Data Buffer	0x2F	0x0000_0000 or 0x0F31_1000[1]	N/A
RRAM Mass Erase (This command is only valid while MERASE(CONFIG0[13]) bit = 0.)	0x26	0x0000_0000 or 0x0F31_1000[1]	N/A
RRAM Read	0x00	Valid address of RRAM memory organization	RMC_ISPDAT: Return Data
Read Company ID	0x0B	0x0000_0000 or 0x0F31_1000[1]	RMC_ISPDAT: 0x5300_00DA
Read Device ID	0x0C	0x0000_0000 or 0x0F31_1000[1]	RMC_ISPDAT: Return Device ID
Read CRC32 Checksum	0x0D	0x0000_0000 or 0x0F31_1000[1]	RMC_ISPDAT: Return Checksum

Run CRC32 Checksum Calculation	0x2D	Valid start address of memory organization It must be 4 Kbytes page alignment	RMC_ISPDAT: Size It must be 4 Kbytes alignment
Read RRAM All One Result	0x08	Keep address of “Run RRAM All One Verification”	RMC_ISPDAT: Return Result 0xA110_0000 : RRAM is not all one 0xA11F_FFFF: RRAM is all one.
Bank REMAP	0x2C	0x0 (switch to Bank0), 0x1 (switch to Bank1)	0x5AA5_5AA5
Run RRAM All One Verification	0x28	Valid start address of memory organization It must be 4 Kbytes page alignment	RMC_ISPDAT: Size It must be 4 Kbytes alignment
Read Unique ID	0x04	0x0000_0000 or 0x0F31_1000[1]	RMC_ISPDAT: Unique ID Word 0
		0x0000_0004 or 0x0F31_1004[1]	RMC_ISPDAT: Unique ID Word 1
		0x0000_0008 or 0x0F31_1008[1]	RMC_ISPDAT: Unique ID Word 2
Vector Remap	0x2E	Valid address in APROM,LDROM or Bootloader It must be 512 bytes alignment	N/A

Table 6.5-4 ISP Command List

Note: When secure conceal is activated, user should set RMC_ISPADDR to 0x0F31_100X, X=0, 1, 4 or 8.

ISP Procedure

RMC provides embedded RRAM memory read, erase and program operation. Several control bits of RMC control register are write-protected, thus it is necessary to unlock before setting.

After unlocking the protected register bits, user needs to set the RMC_ISPCTL control register to decide to update LDROM, APROM or user configuration block, and then set ISPEN (RMC_ISPCTL[0]) to enable ISP function.

Once the RMC_ISPCTL register is set properly, user can set RMC_ISPCMD (refer to Table 6.5-4 ISP command list) for specify operation. Set RMC_ISPADDR for target RRAM memory based on RRAM memory organization. RMC_ISPDAT can be used to set the data to program or used to return the read data according to RMC_ISPCMD. The ISP procedure flow is shown in Figure 6.5-10.

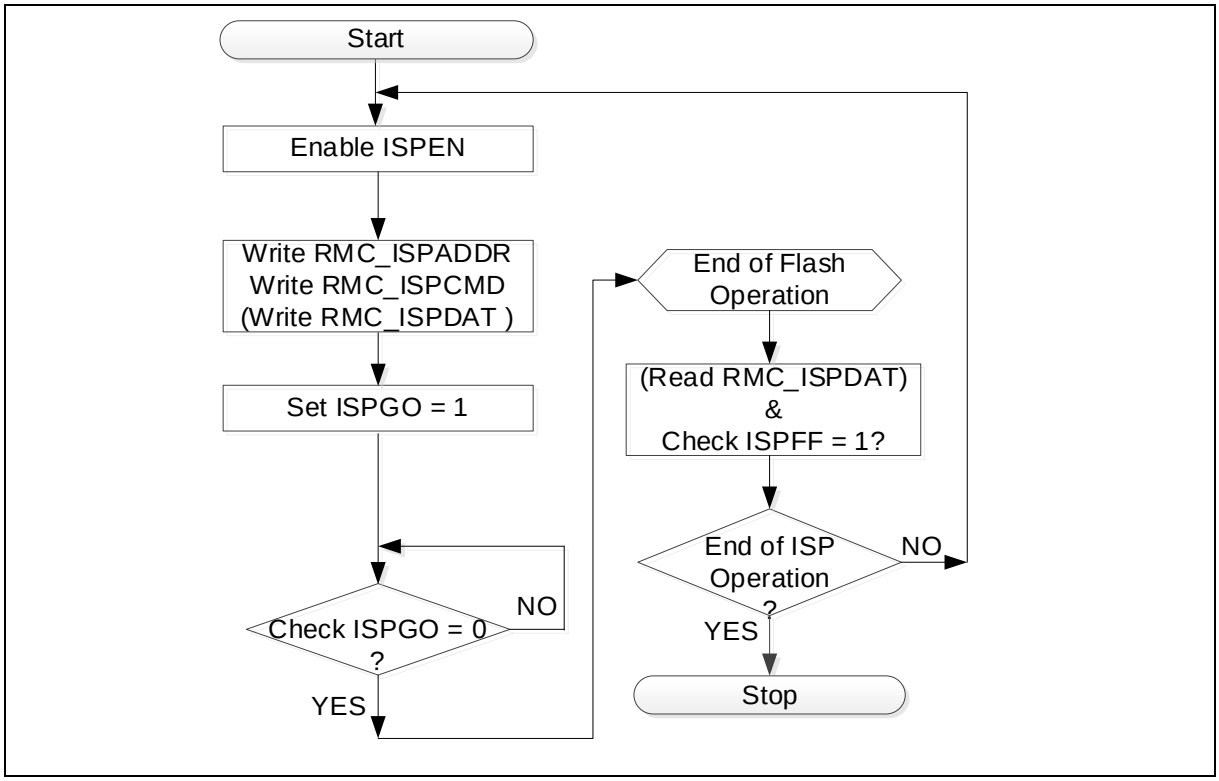


Figure 6.5-10 ISP Procedure Example

Finally, set the ISPGO (RMC_ISPTRG[0]) register to perform the relative ISP function. When ISP function is active, the ISPBUSY(RMC_ISPSTS[0]) and MPBUSY(RMC_MPSTS[0]) is set to 1. The ISPGO(RMC_ISPTRG[0]), ISPBUSY(RMC_ISPSTS[0]) and MPBUSY(RMC_MPSTS[0]) are self-cleared when ISP function has been done.

Several error conditions will be checked after ISP is completed. If an error condition occurs, ISP operation is not started and the ISP fail flag will be set instead. ISPFF(RMC_ISPSTS[6]) flag can only be cleared by software. The next ISP procedure can be started even ISPFF(RMC_ISPSTS[6]) bit is kept as 1. Therefore, it is recommended to check the ISPFF(RMC_ISPSTS[6]) bit and clear it after each ISP operation if it is set to 1.

When ISP is not compleled, user should not set CHIP reset or MCU reset. User must wait ISPGO(RMC_ISPTRG[0]) bit to be cleared before asserting CHIP reset and MUC reset.

When the ISPGO(RMC_ISPTRG[0]) bit is set and then CPU access the same bank, CPU will wait for ISP operation to finish during this period; the peripheral still keeps working as usual. If any interrupt request occurs, CPU will not service it till ISP operation is finished. When ISP operation is finished, the ISPGO bit will be cleared by hardware automatically. User can check whether ISP operation is finished or not by the ISPGO(RMC_ISPTRG[0]) bit.

When CPU access operation and ISP command are executed in different bank, CPU and ISP command can operate in parallel.

6.5.4.13 Embedded RRAM Memory Programming

RMC provides two kinds of programming function. 32-bit single word RRAM memory programming function and 32-bit word line programing function to speed up RRAM updated procedure. The mass erase and program command are only supported at PL1.

Embedded RRAM Memory 32-bits Single Word Programming

User has to set MPEN(RMC_ISPCTL[8]) to 0 to enable single word program function. ISPADDR should

set in target bank to tell which bank's load buffer need to be cleared. Figure 6.5-12 is the ISP 32-bit single word programing procedure flow.

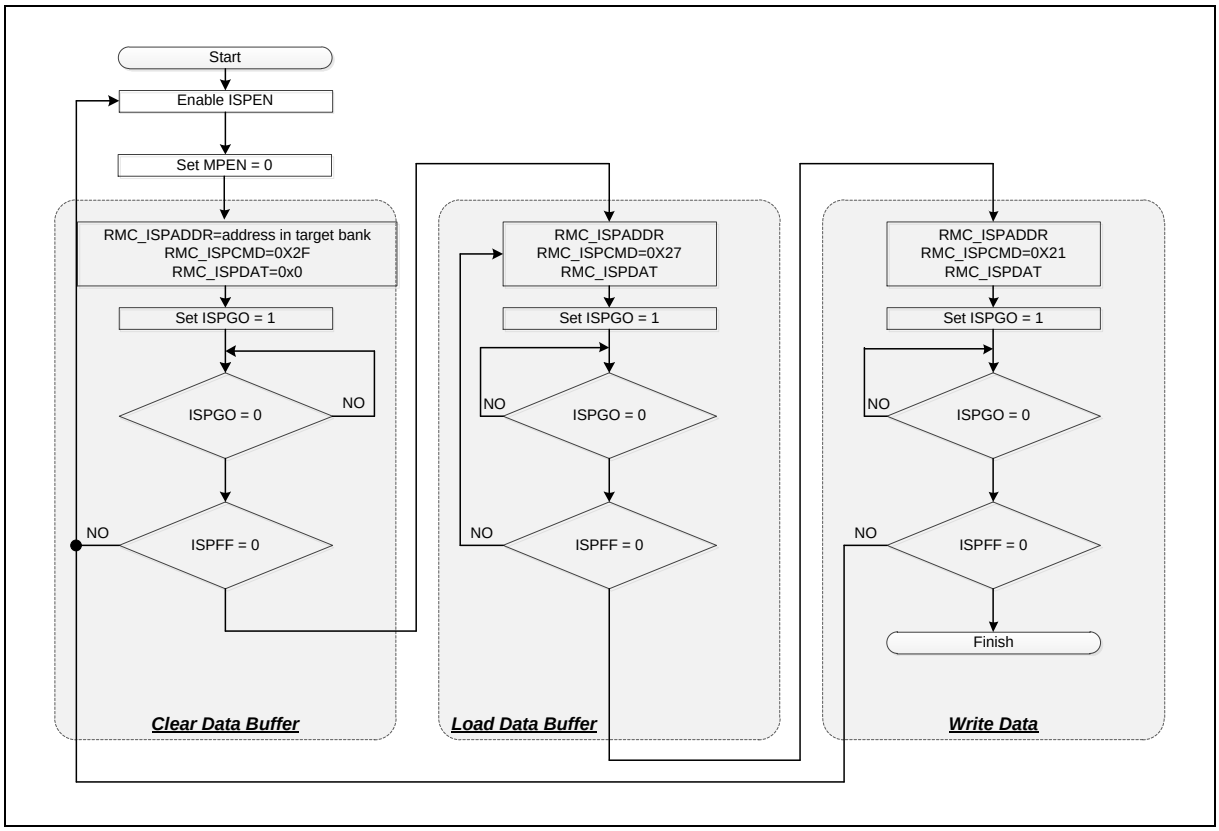


Figure 6.5-11 ISP 32-bit Single Word Programming Procedure

Embedded RRAM Memory 32-bits Word Line Programming

User can load serval data into data buffer and write all data of data buffer to one word line in RRAM.User have to set MPEN(RMC_ISPCTL[8]) to 1 to enable word line programming function. In word line programming, there are two programming data registers, one is RMC_MPDAT0 for LSB word, and the other is RMC_MPDAT1 for MSB word. Maximum number of data buffer is 64. Data width of RRAM is 64-bits. All data will write on one word line so the address have to be 512 bytes alignment. ISPADDR should set in target bank to tell which bank's load buffer need to be cleared. Figure 6.5-12 is the ISP 32-bit word line programing procedure flow.

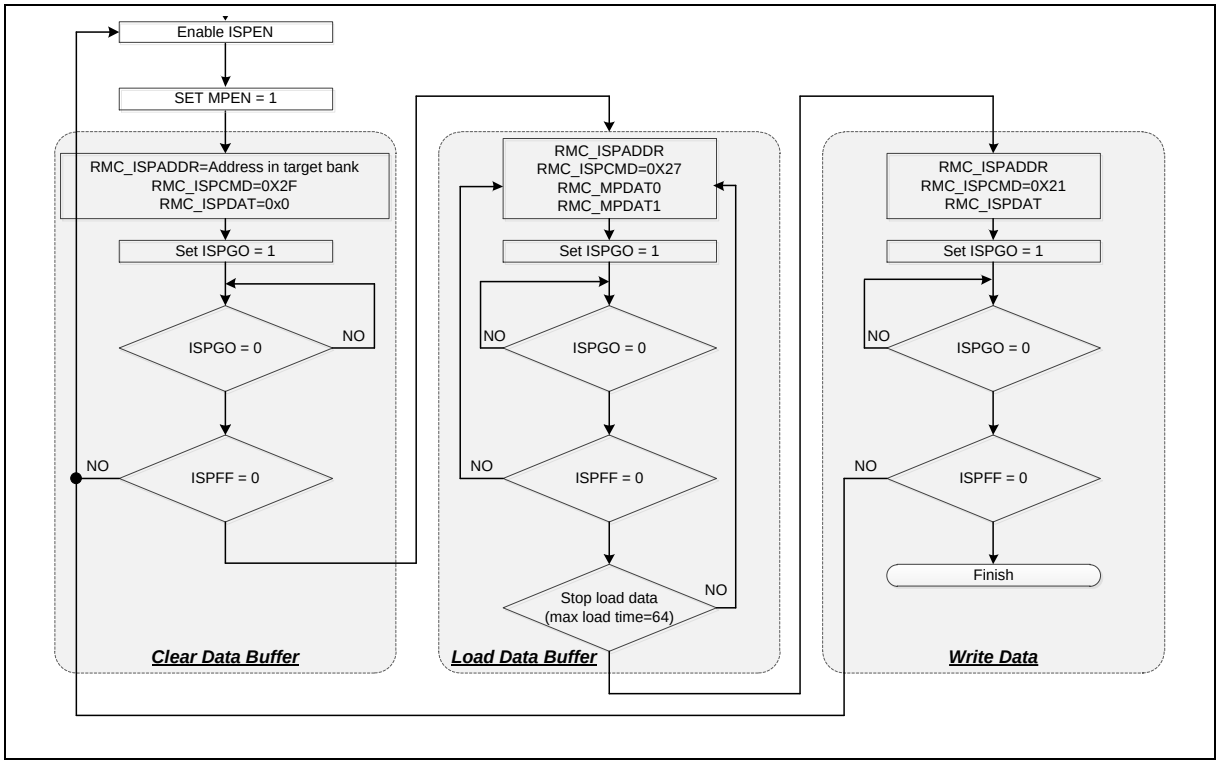


Figure 6.5-12 ISP 32-bit Word Line Programming Procedure

6.5.4.14 CRC32 Checksum Calculation

The M2L31 series supports the CRC32 checksum calculation function to help user quickly check the memory content includes APROM, LDROM and Bootloader. The CRC32 polynomial is

CRC-32: $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$

With seed = 0xFFFF_FFFF

The CRC32 checksum calculation flow is shown in Figure 6.5-13.

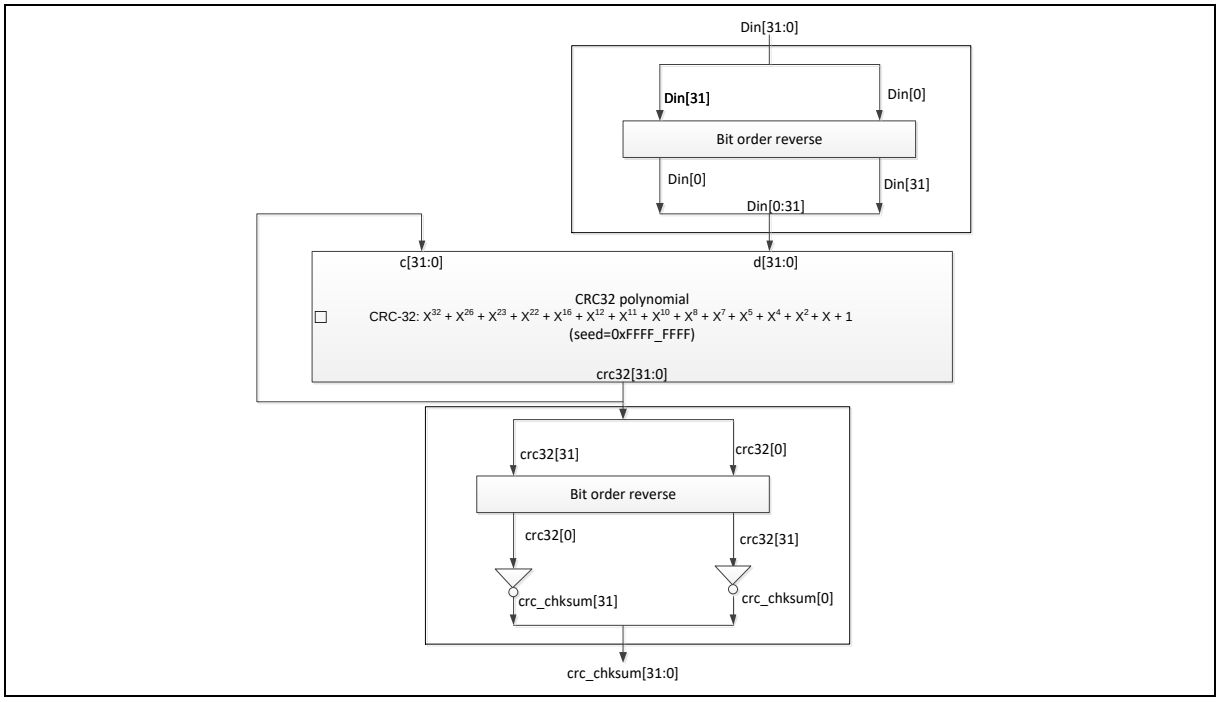


Figure 6.5-13 RRAM CRC32 Checksum Calculation

Use the following three steps to complete the CRC32 checksum calculation.

1. Perform ISP “Run Memory CRC32 Checksum” operation
2. Perform ISP “Read Memory CRC32 Checksum” operation
3. Read RMC_ISPDAT to get checksum.

In Step 1, user has to set the memory starting address (RMC_ISPADDR) and size (RMC_ISPDAT) to calculate. Both address and size have to be 4 Kbytes alignment, the size should be ≥ 4 Kbytes and the starting address includes APROM, LDROM and Bootloader.

In Step 2, the RMC_ISPADDR should be kept the same as Step 1.

In Step 3, the checksum is read from RMC_ISPDAT. If the checksum is 0x0000_0000, there is one of two conditions (1) Checksum calculation is in-progress, (2) Address and size is over device limitation

Note that the range of CRC32 checksum cannot cross bank in one operation.

6.5.4.15 RRAM All One Verification

The M2L31 series supports the RRAM all one verification function to help user quickly check a memory block content blanking for APROM and LDROM after RRAM erase operation.

Use Two or Three steps to complete this RRAM all one verification.

Two-step flow:

1. Perform ISP “Run RRAM All One Verification” operation
2. Read ALLONE(RMC_ISPSTA[7])bit to get the verification result

ALLONE: 1, all of RRAM bits are 1 in verification block memory.

ALLONE: 0, RRAM bits are not all 1 in verification block memory.

Three-step flow:

1. Perform ISP “Run RRAM All One Verification” operation

2. Perform ISP “Read RRAM All One Result” operation
3. Read RMC_ISPDAT to get the verification result.

RMC_ISPDATA: 0xA11F_FFFF, all of RRAM bits are 1 in verification block memory.

RMC_ISPDATA: 0xA110_0000, RRAM bits are not all 1 in verification block memory

In Step 1, user has to set the memory starting address (RMC_ISPADDR) and size (RMC_ISPDAT) to verify. Both address and size have to be 4 Kbytes alignment, the size should be ≥ 4 Kbytes and the starting address includes APROM and LDROM.

In Step 2, the RMC_ISPADDR should be kept the same as Step 1.

Note that the range of “Run All One Verification” cannot cross bank in one operation.

6.5.4.16 APROM Function Safety Protect

RMC supports safety protect function in APROM. User can use the function to avoid user code doing accidental operation. To use the function, user needs to write the enable bit at register RMC_APWROT. Each RMC_APWROT bit protect 8 Kbytes memory.The RMC_APWROT[0] protects memory from 0x0 to 0x1fff, RMC_APWROT[1] protects memory from 0x2000 to 0x3fff and so on. When APROM region is protected, user cannot do any programming at protect region, but still can do mass erase. User can also set RMC_APWPKEEP to lock up the setting of RMC_APWPROT0/1. When RMC_APWPKEEP is active, setting of RMC_APWPROT0/1 can't be modified. When RMC_APWPKEEP[15:0] is 0x55aa, the RMC_APWPROT0 and RMC_APWPKEEP[15:0] content cannot be modified. When RMC_APWPKEEP[31:16] is 0x55aa, the RMC_APWPROT1 and RMC_APWPKEEP[31:16] content cannot be modified. RMC_APWPKEEP can reset by all reset source.

This function can also be set by user configure. CONFIG8 is inverse to RMC_APWPROT0 and CONFIG9 is inverse to RMC_APWPROT1. The setting of CONFIG10 is shown in the table below. All settings in user configure will be activated after chip reset. Setting of CONFIG8 will be loaded to RMC_APWPROT0 and Setting of CONFIG9 will be loaded to RMC_APWPROT1. CONFIG8/9/10 will not be allowed to be modified when CONFIG10 has be set as one time lock. For ISP path, there is a hardware to monitor ISP command match CONFIG8/9/10 be program/write or CONFIG page erase .If this action is found, the ISP will be ignored and this monitor flag is set to 1.

When arranging APROM function safety protect region, user should not make APROM function safety protect region overlap with XOM region and secure conceal region. When APROM function safety protect region is overlapped with XOM region, the XOM erase command is forbidden.

Mode	CONFIG10	CONFIG8/9/10 (Changed by ISP)	Register (Changed by CPU)	Mass Erase [1] (Execute by ISP)
APWPLVL0	0xFFFF	Yes	Yes/No[2]	Yes
APWPLVL1	0x005A/others	No	No	Yes
APWPLVL2	0x335A/335B/335C/335D[4]	No	Yes/No[3]	Yes
APWPLVL3	0x995A/995B/995C/995D[4]	No	Yes/No[3]	Yes/No[3]

Note:

1. The Mass Erase through ISP can be performed in MKROM/APROM/LDROM/SRAM.
2. The write protect lock is decided by RMC_APWPKEEP register through SOFTWARE path (only work in APWPLVL0). Set RMC_APWPKEEP[15:0] to 55AA will keep RMC_APWPROT0 and RMC_APWPKEEP[15:0] value. Setting RMC_APWPKEEP[31:16] to 55AA will keep RMC_APWPROT1 and RMC_APWPKEEP[31:16] value.
3. The write protect lock is decided by pin (WPPIN). When pin is low, RMC_APWPROT0/1 registers cannot be changed. When pin is high, RMC_APWPROT0/1 registers can be changed.
4. The pin (WPPIN) can be selected by APWPLS. Setting APWPLS[3:0] to 0xA will select GPB.2 as WPPIN. Setting APWPLS[3:0] to 0xB will select GPA.6 as WPPIN. Setting APWPLS[3:0] to 0xC will select GPC.14 as WPPIN. Setting APWPLS[3:0] to 0xD will select GPD.13 as WPPIN.

Table 6.5-5 APROM Write Protect Level

CONFIG8 (Address = 0x0F30_0020)

31	30	29	28	27	26	25	24
APPROEN31	APPROEN30	APPROEN29	APPROEN28	APPROEN27	APPROEN26	APPROEN25	APPROEN24
23	22	21	20	19	18	17	16
APPROEN23	APPROEN22	APPROEN21	APPROEN20	APPROEN19	APPROEN18	APPROEN17	APPROEN16
15	14	13	12	11	10	9	8
APPROEN15	APPROEN14	APPROEN13	APPROEN12	APPROEN11	APPROEN10	APPROEN9	APPROEN8
7	6	5	4	3	2	1	0
APPROEN7	APPROEN6	APPROEN5	APPROEN4	APPROEN03	APPROEN02	APPROEN1	APPROEN0

Bits	Descriptions	
[n] n=0,1..31	**APPROENn**	**APROM Protect enable** This bit indicates which APROM region is protected. 0 = APROM region n is protected. 1 = APROM region n is not protected. **Note:** APROM protect region is 0x0 + n*(0x2000) to 0x1fff + n*(0x2000)

CONFIG9 (Address = 0x0F30_0024)

31	30	29	28	27	26	25	24
APPROEN63	APPROEN62	APPROEN61	APPROEN60	APPROEN59	APPROEN58	APPROEN57	APPROEN56
23	22	21	20	19	18	17	16
APPROEN55	APPROEN54	APPROEN53	APPROEN52	APPROEN51	APPROEN50	APPROEN49	APPROEN48
15	14	13	12	11	10	9	8
APPROEN47	APPROEN46	APPROEN45	APPROEN44	APPROEN43	APPROEN42	APPROEN41	APPROEN40
7	6	5	4	3	2	1	0
APPROEN39	APPROEN38	APPROEN37	APPROEN36	APPROEN35	APPROEN34	APPROEN33	APPROEN32

Bits	Descriptions	
[n] n=32,33..63	**APPROENn**	**APROM Protect enable** This bit indicates which APROM region is protected. 0 = APROM region n is protected. 1 = APROM region n is not protected. **Note:** APROM protect region is 0x0 + n*(0x2000) to 0x1fff + n*(0x2000).

CONFIG10 (Address = 0x0F30_0028)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
APWPLS							
7	6	5	4	3	2	1	0
APWPLS							

Bits	Descriptions	
[31:16]	**Reserved**	Reserved.
[15:0]	**APWPLS**	**Setting of APROM Write Protect One Time Lock Level Selection** This bit define APROM protection level. 0xFFFF = APROM Write Protect level is APWPLVL0. 0x005A = APROM Write Protect level is APWPLVL1. 0x335A/0x335B/0x335C/0x335D = APROM Write Protect level is APWPLVL2. 0x995A/0x995B/0x995C/0x995D = APROM Write Protect level is APWPLVL3. Others = APROM Write Protect level is APWPLVL1.

6.5.4.17 Secure Conceal Function

RMC support secure conceal function in APROM/LDROM. When APROM/LDROM region is concealed, this region can't be accessed as data or instruction and can't be programed.

Set CONFIG4 to describe start address of secure conceal region and set CONFIG5 to describe region size. Set CONFIG6 to allow to enable secure conceal function. These setting will activated after chip reset. Setting of CONFIG4/5 must be page-aligned. Page size in this project is 4 Kbytes. User can set SCACT(RMC_SCACT[0]) to active secure conceal function. Hardware will monitor the CPU access range out of the secure conceal region and then automatically activate secure conceal function. The only way to exit active mode after activating secure conceal function is to reset SCACT(RMC_SCACT[0]).

When secure conceal function is enabled but not active, the secure conceal region should be protect also for ICE path. In this situation, hardware will block ICE access for secure conceal region and RMC registers. For ISP path, there is a hardware to monitor ISP command match CONFIG4/5/6 be program/write after secure conceal function is active. If found this action, the ISP will be ignore and this monitor flag is set to 1.

After secure conceal is activated, user should set ISP_ADDR to 0x0F31_100X, where X can be 0, 1, 4, or 8. For detailed ISP command instructions, please refer to Table 6.5-4. It is important to note that the ISP command 'Bank REMAP' is not supported when the secure conceal function is active.

When arrange secure conceal region, user should not make secure conceal region overlap with XOM region and APROM function safety protect region. When secure conceal region is overlap with XOM region, the XOM erase command is forbidden.

CONFIG4 (Address = 0x0F30_0010)

31	30	29	28	27	26	25	24
SCADDR							
23	22	21	20	19	18	17	16
SCADDR							
15	14	13	12	11	10	9	8
SCADDR							
7	6	5	4	3	2	1	0
SCADDR							

Bits	Descriptions	
[31:0]	**SCADDR**	**Base Address of Secure Conceal Function** SCADDR must be page-aligned. **Note:** The page size is 4 Kbytes.

CONFIG5 (Address = 0x0F30_0014)

31	30	29	28	27	26	25	24
SCSIZE							
23	22	21	20	19	18	17	16
SCSIZE							
15	14	13	12	11	10	9	8
SCSIZE							
7	6	5	4	3	2	1	0
SCSIZE							

Bits	Descriptions	
[31:0]	**SCSIZE**	**Page Number of Secure Conceal Function** SCSIZE must be page-aligned. The SCSIZE minimum value is 1 page. If SCSIZE is written to 0, the register value will be set to 1. **Note:** The page size is 4 Kbytes.

CONFIG6 (Address = 0x0F30_0018)

31	30	29	28	27	26	25	24
SCEN							
23	22	21	20	19	18	17	16
SCEN							
15	14	13	12	11	10	9	8
SCEN							
7	6	5	4	3	2	1	0
SCEN							

Bits	Descriptions	
[31:0]	**SCEN**	**Active Enable of Secure Conceal Function** All 1 = secure conceal function Disabled. Others = secure conceal function Enabled. **Note:** When secure conceal function is enabled, the concealable region can still be accessed until the secure conceal is active by setting SCACT (RMC_SCACT[0]) to 1.

6.5.4.18 Lock Effect Tables

The M2L31 series supports two kinds of protections include the Security Lock Control (i.e., LOCK in CONFIG0[1] and ALOCK in CONFIG2[7:0]). In this section, the M2L31 prepares some lock effect tables for user to understand the lock effects on APROM, LDROM, CONFIG and others with above four protections for CPU, ICE and ICP/Writer.

LOCK/ALOCK	**OFF**		**ON**		**OFF**	
KSPLOCK	OFF		OFF		ON	
APROM	R	W	-	-	-	-
Data RRAM	R	W	-	-	-	-
LDROM	R	W	-	-	-	-
CFG	R	W	R	-	R	-
OTP	R	W_2	-	-	-	-
DID/UID/Checksum	R	-	R	-	R	-

Figure 6.5-14 Lock Effect Table with Two Protections for ICE/ ICP/Writer

LOCK/ALOCK	**OFF**		**ON**		**OFF**	
KSPLOCK	OFF		OFF		ON	
APROM	R	W	R	W	R	W
Data RRAM	R	W	R	W	R	W
LDROM	R	W	R	W	R	W
CFG	R	W	R	W	R	W
OTP	R	W_2	R	W_2	R	W_2
Secure Boot Key	-	W	-	W	-	W
Bootloader	R	-	R	-	R	-
DID/UID/Checksum	R	-	R	-	R	-

Table 6.5-6 Lock Effect Table with Two Protections for CPU

Note:

1. The symbol “R” means readable and the symbol “W” means writable.
2. The symbol “X” or “-” means no matter what value is.
3. The symbol “W_2” means OTP is writable only when the corresponding “LOCK BIT” is 0xFFFF_FFFF.

MassErase (i.e. whole chip erase) can be executed any time and any address, except for OTP.

6.5.5 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
RMC Base Address **RMC_BA = 0x4000_C000**				
RMC_ISPCTL	RMC_BA+0x00	R/W	ISP Control Register	0x0000_0000
RMC_ISPADDR	RMC_BA+0x04	R/W	ISP Address Register	0x0000_0000
RMC_ISPDAT	RMC_BA+0x08	R/W	ISP Data Register	0x0000_0000
RMC_ISPCMD	RMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000
RMC_ISPTRG	RMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000
RMC_DFBA	RMC_BA+0x14	R	Data RRAM Base Address	0xXXXX_XXXX
RMC_FTCTL	RMC_BA+0x18	R/W	RRAM Access Time Control Register	0x0000_0000
RMC_ISPSTS	RMC_BA+0x40	R/W	ISP Status Register	0x00XX_XX0X
RMC_CYCCTL	RMC_BA+0x4C	R/W	RRAM Access Cycle Control Register	0x0000_0002
RMC_MPDAT0	RMC_BA+0x80	R/W	ISP Word Line Program Data0 Register	0x0000_0000
RMC_MPDAT1	RMC_BA+0x84	R/W	ISP Word Line Program Data1 Register	0x0000_0000
RMC_XOMR0STS	RMC_BA+0xD0	R	XOM Region 0 Status Register	0x1FF0_00FF
RMC_XOMR1STS	RMC_BA+0xD4	R	XOM Region 1 Status Register	0x1FF0_00FF
RMC_XOMR2STS	RMC_BA+0xD8	R	XOM Region 2 Status Register	0x1FF0_00FF
RMC_XOMR3STS	RMC_BA+0xDC	R	XOM Region 3 Status Register	0x1FF0_00FF
RMC_XOMSTS	RMC_BA+0xE0	R	XOM Status Register	0x0000_0000
RMC_APWPROT0	RMC_BA+0x110	R/W	APROM Write Protect Register0	0x0000_0000
RMC_APWPROT1	RMC_BA+0x114	R/W	APROM Write Protect Register1	0x0000_0000
RMC_APWPKEEP	RMC_BA+0x118	R/W	APROM Write Protect Keep Register	0x0000_0000
RMC_SCACT	RMC_BA+0x11C	R/W	APROM Secure Conceal Active Register	0x0000_0000

6.5.6 Register Description

ISP Control Register (RMC_ISPCTL)

Register	Offset	R/W	Description	Reset Value
RMC_ISPCTL	RMC_BA+0x00	R/W	ISP Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							**INTEN**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**MPEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**ISPFF**	**LDUEN**	**CFGUEN**	**APUEN**	**Reserved**	**BS**	**ISPEN**

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**INTEN**	**Secure ISP INT Enable Bit (Write Protect)** 0= ISP INT Disabled. 1= ISP INT Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. Before using INT, user needs to clear the INTFLAG(RMC_ISPSTS[24]) make sure INT happen at correct time.
[23:9]	**Reserved**	Reserved.
[8]	**MPEN**	**Word Line Program Function Enable Bit (Write Protect)** 0= Word line program function is disabled. 1= Word line program function is enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7]	**Reserved**	Reserved.

[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is set by hardware when a triggered ISP meets any of the following conditions: ● APROM writes to itself if APUEN is set to 0. ● LDROM writes to itself if LDUEN is set to 0. ● CONFIG is programmed if CFGUEN is set to 0. ● Program command at brown-out detected ● Destination address is illegal, such as over an available range. ● Invalid ISP commands ● The base and size of new XOM regions is wrong, overlap. ● The input setting of XOM page erase function is wrong ● The active XOM region is accessed (except for chip erase, page erase, chksum and read CID/DID) ● The XOM setting page is accessed (except for chip erase, word program and read) ● The XOM setting page is erased when XOM off. ● Violate the load code read protection ● OTP is erased. ● Checksum or RRAM All One Verification is not executed in their valid range **Note 1:** This bit needs to be cleared by writing 1 to it. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5]	**LDUEN**	**LDROM Update Enable Bit (Write Protect)** 0 = LDROM cannot be updated. 1 = LDROM can be updated. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[4]	**CFGUEN**	**CONFIG Update Enable Bit (Write Protect)** 0 = CONFIG cannot be updated. 1 = CONFIG can be updated. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**APUEN**	**APROM Update Enable Bit (Write Protect)** 0 = APROM cannot be updated when the chip runs in APROM. 1 = APROM can be updated when the chip runs in APROM. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**Reserved**	Reserved.
[1]	**BS**	**Boot Select (Write Protect)** 0 = Boot from APROM. 1 = Boot from LDROM. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**ISPEN**	**ISP Enable Bit (Write Protect)** ISP function enable bit. Set this bit to enable ISP function. 0 = ISP function Disabled. 1 = ISP function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

ISP Address (RMC_ISPADDR)

Register	Offset	R/W	Description	Reset Value
RMC_ISPADDR	RMC_BA+0x04	R/W	ISP Address Register	0x0000_0000

31	30	29	28	27	26	25	24
ISPADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISPADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ISPADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISPADDR							

Bits	Description	
[31:0]	**ISPADDR**	**ISP Address** The M2L31 series is equipped with embedded RRAM. ISPADDR[1:0] must be kept 00 for ISP 32-bit operation. For CRC32 Checksum Calculation command, this field is the RRAM starting address for checksum calculation, 4 Kbytes alignment is necessary for CRC32 checksum calculation. For RRAM 32-bit Program, ISP address needs word alignment (4-byte).

ISP Data Register (RMC_ISPDAT)

Register	Offset	R/W	Description	Reset Value
RMC_ISPDAT	RMC_BA+0x08	R/W	ISP Data Register	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ISPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISPDAT							

Bits	Description	
[31:0]	**ISPDAT**	**ISP Data** Write data to this register before ISP program operation. Read data from this register after ISP read operation. When ISPFF (RMC_ISPCTL[6]) is 1, ISPDAT = 0xffff_ffff. For Run CRC32 Checksum Calculation command, ISPDAT is the memory size (byte) and 4 Kbytes alignment. For ISP Read CRC32 Checksum command, ISPDAT is the checksum result. If ISPDAT = 0x0000_0000, it means that (1) the checksum calculation is in progress, or (2) the memory range for checksum calculation is incorrect.

ISP Command Register (RMC_ISPCMD)

Register	Offset	R/W	Description	Reset Value
RMC_ISPCMD	RMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CMD**						

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:0]	**CMD**	**ISP Command** ISP command table is shown below: 0x00= RRAM Read. 0x04= Read Unique ID. 0x08= Read RRAM All-One Result. 0x0B= Read Company ID. 0x0C= Read Device ID. 0x0D= Read Checksum. 0x21= RRAM 32-bits Program. 0x26= RRAM Mass Erase. Erase all pages in all banks, except for OTP. 0x27= RRAM 32-bits Load Data Buffer. 0x28= Run RRAM All-One Verification. 0x2C=Bank REMAP. 0x2D= Run Checksum Calculation. 0x2E= Vector Remap. 0x2F= RRAM 32-bits Clear Data Buffer. The other commands are invalid.

ISP Trigger Control Register (RMC_ISPTRG)

Register	Offset	R/W	Description	Reset Value
RMC_ISPTRG	RMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**ISPGO**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**ISPGO**	**ISP Start Trigger (Write Protect)** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. 0 = ISP operation is finished. 1 = ISP is progressed. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Data RRAM Base Address Register (RMC_DFBA)

Register	Offset	R/W	Description	Reset Value
RMC_DFBA	RMC_BA+0x14	R	Data RRAM Base Address	0xXXXX_XXXX

31	30	29	28	27	26	25	24
DFBA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DFBA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DFBA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DFBA							

Bits	Description	
[31:0]	**DFBA**	**Data RRAM Base Address (Read Only)** This register indicates Data RRAM start address. The Data RRAM is shared with APROM. the content of this register is loaded from CONFIG1 This register is valid when DFEN (CONFIG0[0]) =0 .

RRAM Access Time Control Register (RMC_FTCTL)

Register	Offset	R/W	Description	Reset Value
RMC_FTCTL	RMC_BA+0x18	R/W	RRAM Access Time Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**CACHEINV**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**CACHEINV**	**RRAM Cache Invalidation (Write Protect)** 0 = RRAM Cache Invalidation finished (default). 1 = RRAM Cache Invalidation. **Note 1:** Write 1 to start cache invalidation. The value will be changed to 0 once the process finishes. **Note 2:** This bit is write-protected. Refer to the SYS_REGLCTL register.
[8:0]	**Reserved**	Reserved.

ISP Status Register (RMC_ISPSTS)

Register	Offset	R/W	Description	Reset Value
RMC_ISPSTS	RMC_BA+0x40	R/W	ISP Status Register	0x00XX_XX0X

31	30	29	28	27	26	25	24
Reserved	FBS	Reserved				SCFF	INTFLAG
23	22	21	20	19	18	17	16
VECMAP							
15	14	13	12	11	10	9	8
VECMAP							Reserved
7	6	5	4	3	2	1	0
ALLONE	ISPFF	Reserved			CBS	Reserved	ISPBUSY

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**FBS**	**RRAM Bank Selection (Read Only)** This bit indicate which bank is selected to boot. 0 = Booting from Bank0. 1 = Booting from Bank1.
[29:26]	**Reserved**	Reserved.
[25]	**SCFF**	**Secure Conceal Fail Flag (Write Protect)** 0 = ISP did not access secure conceal region. 1 = ISP accessed secure conceal region. **Note 1:** This bit is set by hardware if any ISP command accesses secure region when secure conceal function is active. **Note 2:** This bit needs to be cleared by writing 1 to it.
[24]	**INTFLAG**	**ISP Interrupt Flag** 0 = ISP Not Finished. 1 = ISP done or ISPFF set. **Note:** This function needs to be enabled by RMC_ISPCTRL[24].
[23:9]	**VECMAP**	**Vector Page Mapping Address (Read Only)** All access to 0x0000_0000~0x0000_01FF is remapped to the RRAM memory address {VECMAP[14:0], 9'h000} ~ {VECMAP[14:0], 9'h1FF} **Note:** If VECMAP[14:11]=0x1, LDROM code is mapped to system memory vector. If VECMAP[14:11]=0x0, APROM code is mapped to system memory vector.
[8]	**Reserved**	Reserved.
[7]	**ALLONE**	**RRAM All-one Verification Flag** This bit is set by hardware if all of RRAM bits are 1, and clear if RRAM bits are not all 1 after "Run RRAM All-One Verification" complete; this bit also can be clear by writing 1 0 = All of RRAM bits are 1 after "Run RRAM All-One Verification" completed. 1 = RRAM bits are not all 1 after "Run RRAM All-One Verification" completed.

[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is the mirror of ISPFF (RMC_ISPCTL[6]), it needs to be cleared by writing 1 to RMC_ISPCTL[6] or RMC_ISPSTS[6]. This bit is set by hardware when a triggered ISP meets any of the following conditions: ● APROM writes to itself if APUEN is set to 0. ● LDROM writes to itself if LDUEN is set to 0. ● CONFIG is programmed if CFGUEN is set to 0. ● Program command at brown-out detected ● Destination address is illegal, such as over an available range. ● Invalid ISP commands ● The base and size of new XOM regions is wrong, overlap. ● The input setting of XOM erase function is wrong ● The active XOM region is accessed (except for chip erase, chksum and read CID/DID) ● The XOM setting page is accessed (except for chip erase, word program and read) ● The XOM setting page is erased when XOM off. ● Violate the load code read protection ● OTP is erased. ● Checksum or RRAM All One Verification is not executed in their valid range **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5:3]	**Reserved**	Reserved.
[2]	**CBS**	**Boot Selection of CONFIG (Read Only)** This bit is initiated with the CBS (CONFIG0[7]) after any reset is happened except CPU reset (CPU is 1) or system reset (SYS) is happened. 0 = LDROM with IAP mode. 1 = APROM with IAP mode.
[1]	**Reserved**	Reserved.
[0]	**ISPBUSY**	**ISP Busy Flag (Read Only)** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. This bit is the mirror of ISPGO(RMC_ISPTRG[0]). 0 = ISP operation is finished. 1 = ISP is progressed.

RRAM Access Cycle Control Register (RMC_CYCCTL)

Register	Offset	R/W	Description	Reset Value
RMC_CYCCTL	RMC_BA+0x4C	R/W	RRAM Access Cycle Control Register	0x0000_0002

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**CYCLE**			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**CYCLE**	**RRAM Access Cycle Control (Write Protect)** 0010 = CPU access with two wait cycles if cache miss. The HCLK working frequency range is <25 MHz 0011 = CPU access with three wait cycles if cache miss. The optimized HCLK working frequency range is 25~50 MHz 0100 = CPU access with four wait cycles if cache miss. The optimized HCLK working frequency range is 50~72 MHz Others = Reserved. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

ISP Word Line Program Data 0 Register (RMC_MPDAT0)

Register	Offset	R/W	Description	Reset Value
RMC_MPDAT0	RMC_BA+0x80	R/W	ISP Word Line Program Data0 Register	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISPDAT0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ISPDAT0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISPDAT0							

Bits	Description	
[31:0]	**ISPDAT0**	**ISP Data 0** This register is the first 32-bit data for word line programming, and it is also the mirror of RMC_ISPDAT, both registers keep the same data.

ISP Word Line Program Data 1 Register (RMC_MPDAT1)

Register	Offset	R/W	Description	Reset Value
RMC_MPDAT1	RMC_BA+0x84	R/W	ISP Word Line Program Data1 Register	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT1							
23	22	21	20	19	18	17	16
ISPDAT1							
15	14	13	12	11	10	9	8
ISPDAT1							
7	6	5	4	3	2	1	0
ISPDAT1							

Bits	Description	
[31:0]	**ISPDAT1**	**ISP Data 1** This register is the second 32-bit data for word line programming.

XOM Region0 Status Register (RMC_XOMR0STS)

Register	Offset	R/W	Description	Reset Value
RMC_XOMR0STS	RMC_BA+0xD0	R	XOM Region 0 Status Register	0x1FF0_00FF

31	30	29	28	27	26	25	24
BASE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BASE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BASE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SIZE							

Bits	Description	
[31:8]	**BASE**	**XOM Region 0 Base Address** BASE is the base address of XOM Region 0 must be page-aligned..
[7:0]	**SIZE**	**XOM Region 0 Size** SIZE is the page number of XOM Region 0 must be page-aligned..

XOM Region1 Status Register (RMC_XOMR1STS)

Register	Offset	R/W	Description	Reset Value
RMC_XOMR1STS	RMC_BA+0xD4	R	XOM Region 1 Status Register	0x1FF0_00FF

31	30	29	28	27	26	25	24
BASE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BASE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BASE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SIZE							

Bits	Description	
[31:8]	**BASE**	**XOM Region 1 Base Address** BASE is the base address of XOM Region 1 must be page-aligned..
[7:0]	**SIZE**	**XOM Region 1 Size** SIZE is the page number of XOM Region 1 must be page-aligned..

XOM Region2 Status Register (RMC_XOMR2STS)

Register	Offset	R/W	Description	Reset Value
RMC_XOMR2STS	RMC_BA+0xD8	R	XOM Region 2 Status Register	0x1FF0_00FF

31	30	29	28	27	26	25	24
BASE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BASE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BASE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SIZE							

Bits	Description	
[31:8]	**BASE**	**XOM Region 2 Base Address** BASE is the base address of XOM Region 2 must be page-aligned..
[7:0]	**SIZE**	**XOM Region 2 Size** SIZE is the page number of XOM Region 2 must be page-aligned..

XOM Region3 Status Register (RMC_XOMR3STS)

Register	Offset	R/W	Description	Reset Value
RMC_XOMR3STS	RMC_BA+0xDC	R	XOM Region 3 Status Register	0x1FF0_00FF

31	30	29	28	27	26	25	24
BASE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BASE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BASE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SIZE							

Bits	Description	
[31:8]	**BASE**	**XOM Region 3 Base Address** BASE is the base address of XOM Region 3 must be page-aligned.
[7:0]	**SIZE**	**XOM Region 3 Size** SIZE is the page number of XOM Region 3 must be page-aligned.

XOM Status Register (RMC_XOMSTS)

Register	Offset	R/W	Description	Reset Value
RMC_XOMSTS	RMC_BA+0xE0	R	XOM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			XOMPEF	XOMR3ON	XOMR2ON	XOMR1ON	XOMR0ON

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**XOMPEF**	**XOM Page Erase Function Fail** XOM page erase function status. If XOMPEF is set to 1, user needs to erase XOM region again. 0 = Sucess. 1 = Fail.
[3]	**XOMR3ON**	**XOM Region 3 On** XOM Region 3 active status. 0 = No active. 1 = XOM region 3 is active.
[2]	**XOMR2ON**	**XOM Region 2 On** XOM Region 2 active status. 0 = No active. 1 = XOM region 2 is active.
[1]	**XOMR1ON**	**XOM Region 1 On** XOM Region 1 active status. 0 = No active. 1 = XOM region 1 is active.
[0]	**XOMR0ON**	**XOM Region 0 On** XOM Region 0 active status. 0 = No active. 1 = XOM region 0 is active.

APROM Write Protect Register0 (RMC_APWPROT0)

Register	Offset	R/W	Description	Reset Value
RMC_APWPROT0	RMC_BA+0x110	R/W	APROM Write Protect Register0	0x0000_0000

31	30	29	28	27	26	25	24
APPROEN31	APPROEN30	APPROEN29	APPROEN28	APPROEN27	APPROEN26	APPROEN25	APPROEN24
23	22	21	20	19	18	17	16
APPROEN23	APPROEN22	APPROEN21	APPROEN20	APPROEN19	APPROEN18	APPROEN17	APPROEN16
15	14	13	12	11	10	9	8
APPROEN15	APPROEN14	APPROEN13	APPROEN12	APPROEN11	APPROEN10	APPROEN9	APPROEN8
7	6	5	4	3	2	1	0
APPROEN7	APPROEN6	APPROEN5	APPROEN4	APPROEN03	APPROEN02	APPROEN1	APPROEN0

Bits	Description	
[n] n=0,1..31	**APPROENn**	**APROM Protect enable** This bit indicates which APROM region is protected. 0 = APROM region n is not protected. 1 = APROM region n is protected. **Note:** APROM protect region is 0x0 + n*(0x2000) to 0x1fff + n*(0x2000)

APROM Write Protect Register1 (RMC_APWPROT1)

Register	Offset	R/W	Description	Reset Value
RMC_APWPROT1	RMC_BA+0x114	R/W	APROM Write Protect Register1	0x0000_0000

31	30	29	28	27	26	25	24
APPROEN63	**APPROEN62**	**APPROEN61**	**APPROEN60**	**APPROEN59**	**APPROEN58**	**APPROEN57**	**APPROEN56**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
APPROEN55	**APPROEN54**	**APPROEN53**	**APPROEN52**	**APPROEN51**	**APPROEN50**	**APPROEN49**	**APPROEN48**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
APPROEN47	**APPROEN46**	**APPROEN45**	**APPROEN44**	**APPROEN43**	**APPROEN42**	**APPROEN41**	**APPROEN40**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
APPROEN39	**APPROEN38**	**APPROEN37**	**APPROEN36**	**APPROEN35**	**APPROEN34**	**APPROEN33**	**APPROEN32**

Bits	Description	
[n-32] n=32,33..63	**APPROENn**	**APROM Protect enable** This bit indicates which APROM region is protected. 0 = APROM region n is not protected. 1 = APROM region n is protected. **Note:** APROM protect region is 0x0 + n*(0x2000) to 0x1fff + n*(0x2000)

APROM Write Protect Keep Register (RMC_APWPKEEP)

Register	Offset	R/W	Description	Reset Value
RMC_APWPKEEP	RMC_BA+0x118	R/W	APROM Write Protect Keep Register	0x0000_0000

31	30	29	28	27	26	25	24
APWPKEEP1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
APWPKEEP1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
APWPKEEP0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
APWPKEEP0							

Bits	Description	
[31:16]	**APWPKEEP1**	**APROM Write Protect Keep 1** 0x55AA = APWPROT1 register is locked. Others = APWPROT1 register is free.
[15:0]	**APWPKEEP0**	**APROM Write Protect Keep 0** 0x55AA = APWPROT0 register is locked. Ohers = APWPROT0 register is free.

APROM Secure Conceal Active Register (RMC_SCACT)

Register	Offset	R/W	Description	Reset Value
RMC_SCACT	RMC_BA+0x11C	R/W	APROM Secure Conceal Active Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SCACT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SCACT**	**Secure Conceal Function Active** 0 = Secure conceal function inactive. 1 = Secure conceal function active. **Note:** Secure conceal function active will base on this bit and setting of CONFIG6 is all 0. **Note:** After SCACT is set to 1, the secure conceal function is active and cannot be disabled by software. Thus, reading SCACT will always get 1 after setting this bit.

6.6 General Purpose I/O (GPIO)

6.6.1 Overview

This chip has up to 109 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 109 pins are arranged in 8 ports named as PA, PB, PC, PD, PE, PF, PG and PH. PA, PB, PD and PE has 16 pins on port. PC has 15 pins on port. PF has 12 pins on port. PG has 10 pins on port. PH has 8 pins on port. Each of the 109 pins is independent and has the corresponding register bits to control the pin mode function and data.

The I/O type of each of I/O pins can be configured by software individually as Input, Push-pull output, Open-drain output or Quasi-bidirectional mode.

6.6.2 Features

- Four I/O modes:
  - Quasi-bidirectional mode
  - Push-Pull Output mode
  - Open-Drain Output mode
  - Input only with high impendence mode
- TTL/Schmitt trigger input selectable
- I/O pin can be configured as interrupt source with edge/level setting
- Supports High Drive and High Slew Rate I/O mode
- Supports independent pull-up and pull-down control
- Enabling the pin interrupt function will also enable the wake-up function

6.6.3 Block Diagram

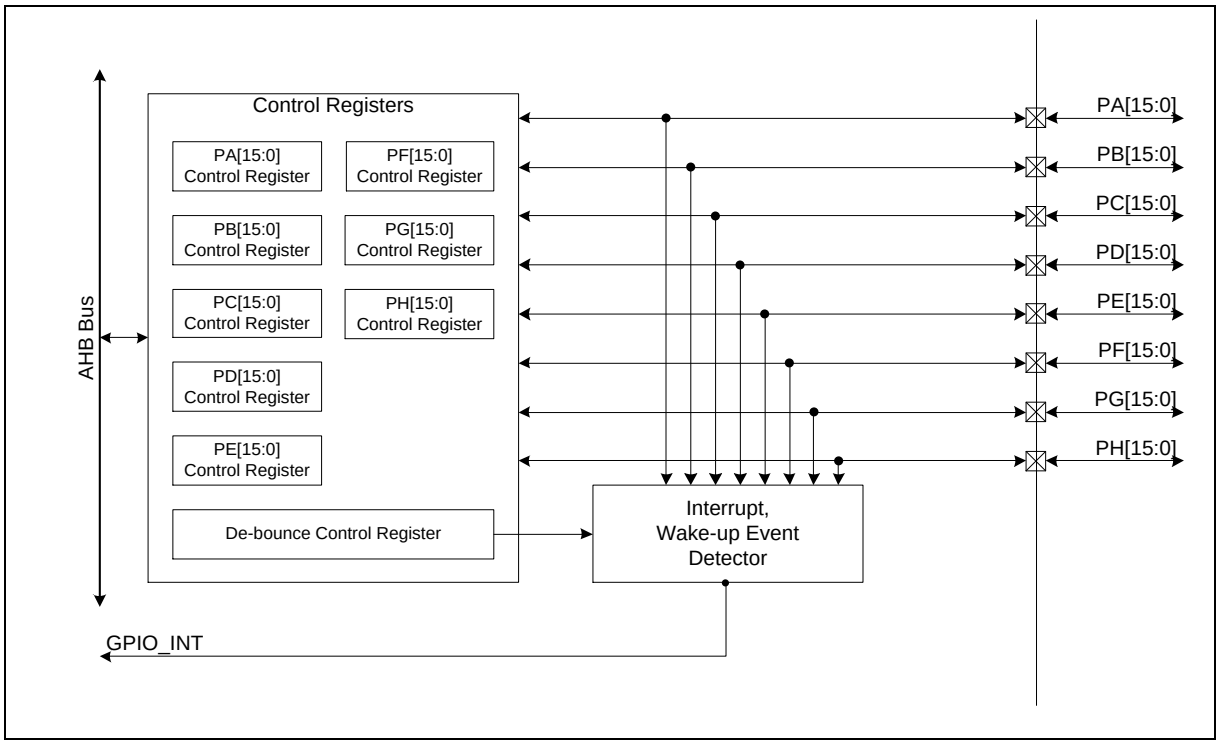


Figure 6.6-1 GPIO Controller Block Diagram

Note: The PC.15/PF.12/PF.13/PF.14/PF.15/PG.0/PG.1/PG.5/PG.6/PG.7/PG.8/PH.0/PH.1/PH.2/PH.3/PH.12/PH.13/PH.14/PH.15 pins are not available.

6.6.4 Basic Configuration

- Reset configuration
  - Reset GPIO in GPIORST SYS_IPRST1[1]

6.6.5 Functional Description

6.6.5.1 Power-on Status

When chip power-on, all I/O keeps in tri-state except PC.0, PC.1, PC.2, PC.3, PF.0 and PF.1.

The PC.0 and PC.1 has a pull-down to V_{SS} effect caused by default state of UTCPD0_CC1 and UTCPD0_CC2 for dead battery application. Setting both IOMODE (SYS_UTCPDCTL[0]) to 1 and PROEN0 (SYS_UTCPDCTL[1]) to 1 can remove this pull-down effect.

The PC.2 and PC.3 has a pull-down to V_{SS} effect caused by default state of UTCPD0_CCDB1 and UTCPD0_CCDB2.

The PF.0 and PF.1 has a pull-up to V_{DD} effect caused by the default multi-function setting is ICE debug pins (ICE_CLK and ICE_DAT). Change PF.0 and PF.1 multi-function setting can remove this pull-up effect.

6.6.5.2 Input Mode

Set MODEn (Px_MODE[2n+1:2n]) to 00 as the Px.n pin is in Input mode and the I/O pin is in tri-state (high impedance) without output drive capability. The PIN (Px_PIN[n]) value reflects the status of the corresponding port pins.

6.6.5.3 Push-pull Output Mode

Figure 6.6-2 shows the diagram of Push-pull Output Mode. Set MODEn (Px_MODE[2n+1:2n]) to 01 as Px.n pin is in Push-pull Output mode and the I/O pin supports digital output function with source/sink current capability. The bit value in the corresponding DOUT (Px_DOUT[n]) is driven on the pin.

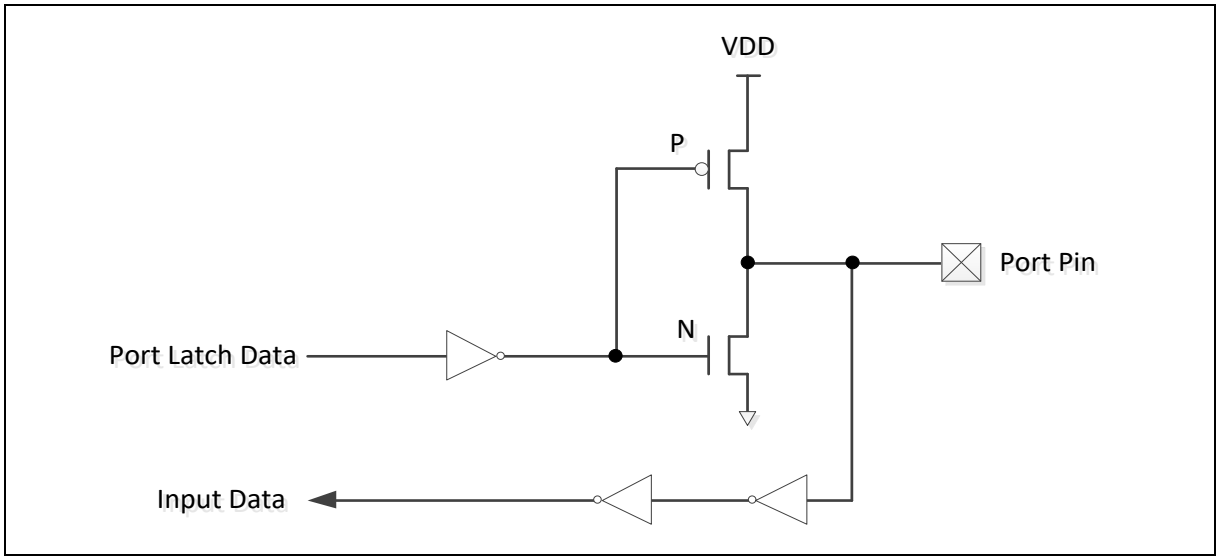


Figure 6.6-2 Push-Pull Output

6.6.5.4 Open-drain Mode

Figure 6.6-3 shows the diagram of Open-drain Mode. Set MODEn (Px_MODE[2n+1:2n]) to 10 the Px.n pin is in Open-drain mode and the digital output function of I/O pin supports only sink current capability, an external pull-up register is needed for driving high state. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 0, the pin drive a low output on the pin. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 1, the pin output drives high that is controlled by external pull high resistor.

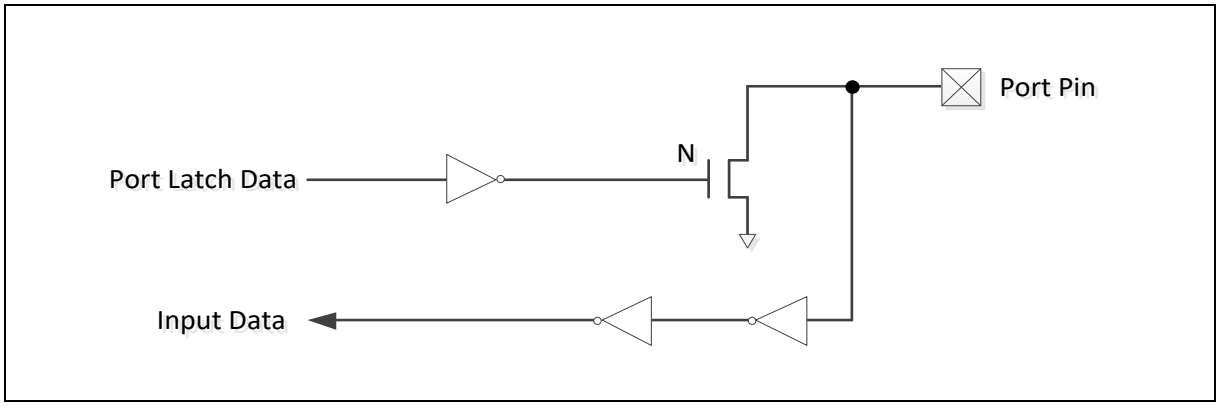


Figure 6.6-3 Open-Drain Output

6.6.5.5 Quasi-bidirectional Mode

Figure 6.6-4 shows the diagram of Quasi-bidirectional Mode. Set MODEn (Px_MODE[2n+1:2n]) to 11 as the Px.n pin is in Quasi-bidirectional mode and the I/O pin supports digital output and input function at the same time but the source current is only up to hundreds uA. Before the digital input function is performed the corresponding DOUT (Px_DOUT[n]) bit must be set to 1. The quasi-bidirectional output is common on the 80C51 and most of its derivatives. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 0, the pin drive a low output on the pin. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 1, the pin will check the pin value. If pin value is high, no action takes. If pin state is

low, the pin will drive strong high with 2 clock cycles on the pin and then disable the strong output drive. Meanwhile, the pin status is controlled by internal pull-up resistor. Note that the source current capability in quasi-bidirectional mode is only about 200 uA to 30 uA for V_{DD} is form 5.0 V to 2.5 V.

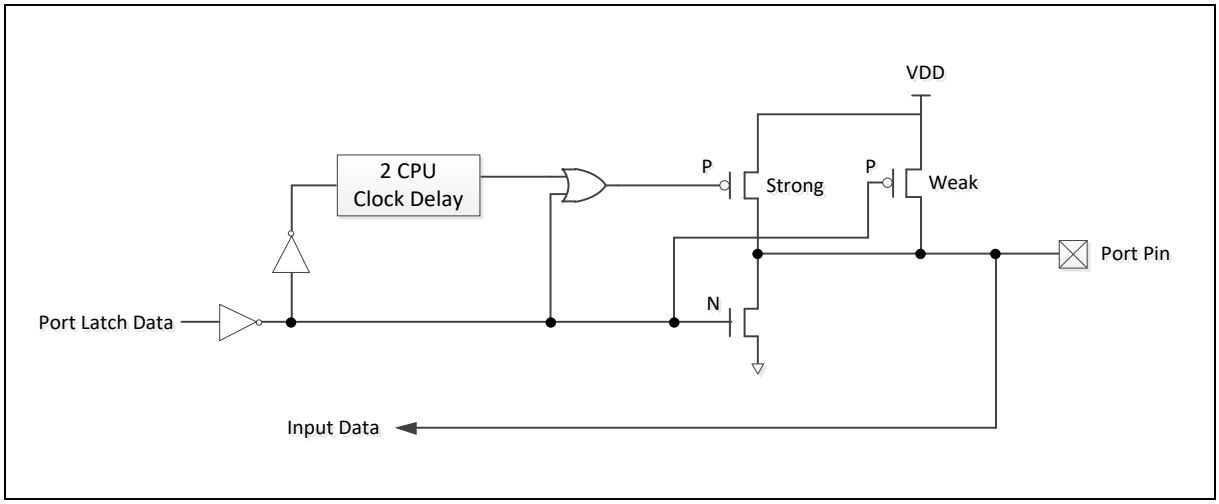


Figure 6.6-4 Quasi-Bidirectional I/O Mode

6.6.5.6 GPIO Interrupt and Wake-up Function

Each GPIO pin can be set as chip interrupt source by setting correlative RHIEN (Px_INTEN[n+16])/ FLIEN (Px_INTEN[n]) bit and TYPE (Px_INTTYPE[n]). There are five types of interrupt condition can be selected: low level trigger, high level trigger, falling edge trigger, rising edge trigger and both rising and falling edge trigger. For edge trigger types, the interval between the two detected edges must be longer than one HCLK period. When chip is in Power-down mode, this HCLK cycle interval is not required. The GPIO can also be the chip wake-up source when chip enters Idle/Power-down mode.

6.6.5.7 GPIO De-bounce Function

GPIO de-bounce function can be used to sample interrupt input for each GPIO pin and prevent unexpected interrupt happened which caused by noise. GPIO de-bounce function only supports edge detection trigger type. For edge trigger condition, there are three types of interrupt condition can be selected for de-bounce function: falling edge trigger, rising edge trigger and both rising and falling edge trigger by setting correlative RHIEN (Px_INTEN[n+16])/ FLIEN (Px_INTEN[n]) bit and TYPE (Px_INTTYPE[n]). If user wants to use de-bounce function, de-bounce enable control register Px_DBEN must be set for corresponding GPIO pin. The de-bounce clock source can be HCLK or LIRC by setting DBCLKSRC (Px_DBCTL[4]) register. And DBCLKSEL (Px_DBCTL[3:0]) register can control sampling cycle period.

Figure 6.6-5 shows GPIO rising edge trigger interrupt. The interval of time between the two valid sample signal is determined by DBCLKSRC (Px_DBCTL[4]) and DBCLKSEL (Px_DBCTL[3:0]). Each valid data from GPIO pin need to be sample twice. For rising edge setting, if pin status is low before setting DBEN (Px_DBEN), interrupt will happen when generating a pin high valid data. But, if pin status is high before setting DBEN (Px_DBEN), interrupt will happen when generating a pin low valid data first, and then generating a pin high valid data. For falling edge trigger, Figure 6.6-6 shows the situation is opposite to rising edge trigger.

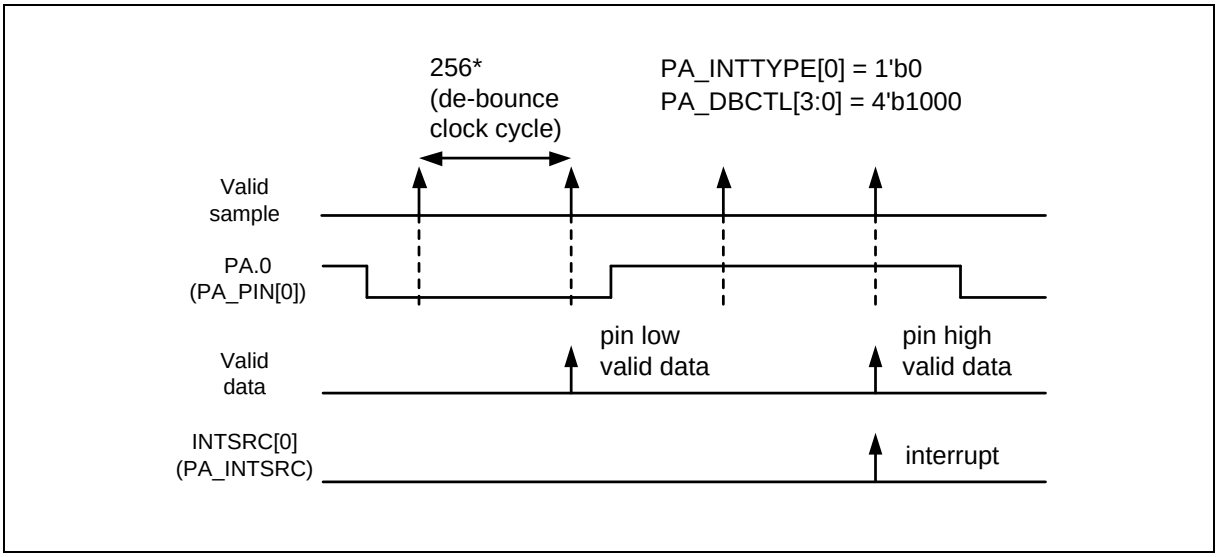


Figure 6.6-5 GPIO Rising Edge Trigger Interrupt

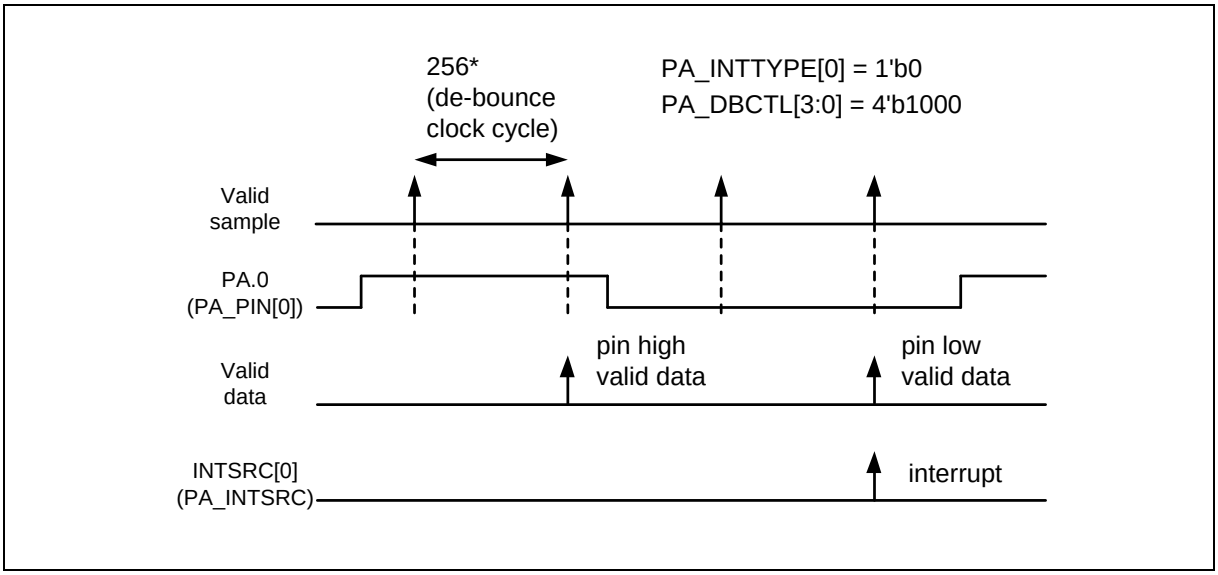


Figure 6.6-6 GPIO Falling Edge Trigger Interrupt

GPIO de-bounce function are also supported in Power-down mode. Table 6.6-1 shows the de-bounce function support situation in different system status. The de-bounce function can support in Power-down mode by setting DBENn(Px_DBEN[n]) and DBCLKSRC(Px_DBCTL[4]) to 1. DBCLKSEL (Px_DBCTL[3:0]) can be set to control the GPIO de-bounce time to wake up system.

System Status	DBEN	DBCLKSRC	Description
Normal Mode / Idle Mode	0	0	No de-bounce function
		1	No de-bounce function
	1	0	De-bounce function using HCLK
		1	De-bounce function using LIRC
Power Down	0	0	No de-bounce function

Mode		1	No de-bounce function
	1	0	No de-bounce function
		1	De-bounce function using LIRC

Table 6.6-1 De-Bounce Function Setting Table

6.6.5.8 INT Edge Detect and Trigger Function

There are many INT0~7 GPIO pins that can be selected by multi-function. Each INT pin will be filtered by a 3-bit noise filter. User can enable the noise filter function by NFEN (INTn_INNF[0], n=0~7) bit, and noise filter sampling clock can be selected by setting NFSEL (INTn_INNF[6:4], n=0~7) bits to fit different noise properties. Moreover, by setting the NFCNT (INTn_INNF[10:8], n=0~7) bits, user can define by how many sampling clock cycles a filter will recognize the effective edge of the INT pin signal.

Each INT pin can be detected by setting EDETCTLn (INT_EDETCTL[2n+1:2n]) and EDIFn (INT_EDSTS[n], n=0~7) will be set if INT pin matches condition. Interrupt will happen if user enables EDIENn (INT_EDINTEN[n], n=0~7) and INT pin match condition.

In addition to detect function, it can also trigger EPWM and TIMER (PWM mode) if INT pin matches condition. More detailed setting is in EPWM or TIMER (PWM mode) “External Pin Event Trigger” section.

6.6.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
GPIO Base Address: **GPIO_BA = 0x4000_4000**				
PA_MODE	GPIO_BA+0x000	R/W	PA I/O Mode Control	0x0000_0000
PA_DINOFF	GPIO_BA+0x004	R/W	PA Digital Input Path Disable Control	0x0000_0000
PA_DOUT	GPIO_BA+0x008	R/W	PA Data Output Value	0x0000_FFFF
PA_DATMSK	GPIO_BA+0x00C	R/W	PA Data Output Write Mask	0x0000_0000
PA_PIN	GPIO_BA+0x010	R	PA Pin Value	0x0000_XXXX
PA_DBEN	GPIO_BA+0x014	R/W	PA De-bounce Enable Control Register	0x0000_0000
PA_INTTYPE	GPIO_BA+0x018	R/W	PA Interrupt Trigger Type Control	0x0000_0000
PA_INTEN	GPIO_BA+0x01C	R/W	PA Interrupt Enable Control Register	0x0000_0000
PA_INTSRC	GPIO_BA+0x020	R/W	PA Interrupt Source Flag	0x0000_XXXX
PA_SMTEN	GPIO_BA+0x024	R/W	PA Input Schmitt Trigger Enable Register	0x0000_0000
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA High Slew Rate Control Register	0x0000_0000
PA_PUSEL	GPIO_BA+0x030	R/W	PA Pull-up and Pull-down Selection Register	0x0000_0000
PA_DBCTL	GPIO_BA+0x034	R/W	PA Interrupt De-bounce Control Register	0x0000_0020
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0x0000_0000
PB_DINOFF	GPIO_BA+0x044	R/W	PB Digital Input Path Disable Control	0x0000_0000
PB_DOUT	GPIO_BA+0x048	R/W	PB Data Output Value	0x0000_FFFF
PB_DATMSK	GPIO_BA+0x04C	R/W	PB Data Output Write Mask	0x0000_0000
PB_PIN	GPIO_BA+0x050	R	PB Pin Value	0x0000_XXXX
PB_DBEN	GPIO_BA+0x054	R/W	PB De-bounce Enable Control Register	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB Interrupt Trigger Type Control	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB Interrupt Enable Control Register	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	PB Interrupt Source Flag	0x0000_XXXX
PB_SMTEN	GPIO_BA+0x064	R/W	PB Input Schmitt Trigger Enable Register	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB High Slew Rate Control Register	0x0000_0000
PB_PUSEL	GPIO_BA+0x070	R/W	PB Pull-up and Pull-down Selection Register	0x0000_0000
PB_DBCTL	GPIO_BA+0x074	R/W	PB Interrupt De-bounce Control Register	0x0000_0020
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0x0000_0000

PC_DINOFF	GPIO_BA+0x084	R/W	PC Digital Input Path Disable Control	0x0000_0000
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_7FFF
PC_DATMSK	GPIO_BA+0x08C	R/W	PC Data Output Write Mask	0x0000_0000
PC_PIN	GPIO_BA+0x090	R	PC Pin Value	0x0000_XXXX
PC_DBEN	GPIO_BA+0x094	R/W	PC De-bounce Enable Control Register	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC Interrupt Trigger Type Control	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC Interrupt Enable Control Register	0x0000_0000
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC Interrupt Source Flag	0x0000_XXXX
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC Input Schmitt Trigger Enable Register	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC High Slew Rate Control Register	0x0000_0000
PC_PUSEL	GPIO_BA+0x0B0	R/W	PC Pull-up and Pull-down Selection Register	0x0000_0000
PC_DBCTL	GPIO_BA+0x0B4	R/W	PC Interrupt De-bounce Control Register	0x0000_0020
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0x0000_0000
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD Digital Input Path Disable Control	0x0000_0000
PD_DOUT	GPIO_BA+0x0C8	R/W	PD Data Output Value	0x0000_FFFF
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD Data Output Write Mask	0x0000_0000
PD_PIN	GPIO_BA+0x0D0	R	PD Pin Value	0x0000_XXXX
PD_DBEN	GPIO_BA+0x0D4	R/W	PD De-bounce Enable Control Register	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD Interrupt Trigger Type Control	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD Interrupt Enable Control Register	0x0000_0000
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD Interrupt Source Flag	0x0000_XXXX
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD Input Schmitt Trigger Enable Register	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD High Slew Rate Control Register	0x0000_0000
PD_PUSEL	GPIO_BA+0x0F0	R/W	PD Pull-up and Pull-down Selection Register	0x0000_0000
PD_DBCTL	GPIO_BA+0x0F4	R/W	PD Interrupt De-bounce Control Register	0x0000_0020
PE_MODE	GPIO_BA+0x100	R/W	PE I/O Mode Control	0x0000_0000
PE_DINOFF	GPIO_BA+0x104	R/W	PE Digital Input Path Disable Control	0x0000_0000
PE_DOUT	GPIO_BA+0x108	R/W	PE Data Output Value	0x0000_FFFF
PE_DATMSK	GPIO_BA+0x10C	R/W	PE Data Output Write Mask	0x0000_0000
PE_PIN	GPIO_BA+0x110	R	PE Pin Value	0x0000_XXXX
PE_DBEN	GPIO_BA+0x114	R/W	PE De-bounce Enable Control Register	0x0000_0000

PE_INTTYPE	GPIO_BA+0x118	R/W	PE Interrupt Trigger Type Control	0x0000_0000
PE_INTEN	GPIO_BA+0x11C	R/W	PE Interrupt Enable Control Register	0x0000_0000
PE_INTSRC	GPIO_BA+0x120	R/W	PE Interrupt Source Flag	0x0000_XXXX
PE_SMTEN	GPIO_BA+0x124	R/W	PE Input Schmitt Trigger Enable Register	0x0000_0000
PE_SLEWCTL	GPIO_BA+0x128	R/W	PE High Slew Rate Control Register	0x0000_0000
PE_PUSEL	GPIO_BA+0x130	R/W	PE Pull-up and Pull-down Selection Register	0x0000_0000
PE_DBCTL	GPIO_BA+0x134	R/W	PE Interrupt De-bounce Control Register	0x0000_0020
PF_MODE	GPIO_BA+0x140	R/W	PF I/O Mode Control	0x0000_0000
PF_DINOFF	GPIO_BA+0x144	R/W	PF Digital Input Path Disable Control	0x0000_0000
PF_DOUT	GPIO_BA+0x148	R/W	PF Data Output Value	0x0000_0FFF
PF_DATMSK	GPIO_BA+0x14C	R/W	PF Data Output Write Mask	0x0000_0000
PF_PIN	GPIO_BA+0x150	R	PF Pin Value	0x0000_XXXX
PF_DBEN	GPIO_BA+0x154	R/W	PF De-bounce Enable Control Register	0x0000_0000
PF_INTTYPE	GPIO_BA+0x158	R/W	PF Interrupt Trigger Type Control	0x0000_0000
PF_INTEN	GPIO_BA+0x15C	R/W	PF Interrupt Enable Control Register	0x0000_0000
PF_INTSRC	GPIO_BA+0x160	R/W	PF Interrupt Source Flag	0x0000_XXXX
PF_SMTEN	GPIO_BA+0x164	R/W	PF Input Schmitt Trigger Enable Register	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF High Slew Rate Control Register	0x0000_0000
PF_PUSEL	GPIO_BA+0x170	R/W	PF Pull-up and Pull-down Selection Register	0x0000_0000
PF_DBCTL	GPIO_BA+0x174	R/W	PF Interrupt De-bounce Control Register	0x0000_0020
PG_MODE	GPIO_BA+0x180	R/W	PG I/O Mode Control	0x0000_0000
PG_DINOFF	GPIO_BA+0x184	R/W	PG Digital Input Path Disable Control	0x0000_0000
PG_DOUT	GPIO_BA+0x188	R/W	PG Data Output Value	0x0000_FE1C
PG_DATMSK	GPIO_BA+0x18C	R/W	PG Data Output Write Mask	0x0000_0000
PG_PIN	GPIO_BA+0x190	R	PG Pin Value	0x0000_XXXX
PG_DBEN	GPIO_BA+0x194	R/W	PG De-bounce Enable Control Register	0x0000_0000
PG_INTTYPE	GPIO_BA+0x198	R/W	PG Interrupt Trigger Type Control	0x0000_0000
PG_INTEN	GPIO_BA+0x19C	R/W	PG Interrupt Enable Control Register	0x0000_0000
PG_INTSRC	GPIO_BA+0x1A0	R/W	PG Interrupt Source Flag	0x0000_XXXX
PG_SMTEN	GPIO_BA+0x1A4	R/W	PG Input Schmitt Trigger Enable Register	0x0000_0000
PG_SLEWCTL	GPIO_BA+0x1A8	R/W	PG High Slew Rate Control Register	0x0000_0000

PG_PUSEL	GPIO_BA+0x1B0	R/W	PG Pull-up and Pull-down Selection Register	0x0000_0000
PG_DBCTL	GPIO_BA+0x1B4	R/W	PG Interrupt De-bounce Control Register	0x0000_0020
PH_MODE	GPIO_BA+0x1C0	R/W	PH I/O Mode Control	0x0000_0000
PH_DINOFF	GPIO_BA+0x1C4	R/W	PH Digital Input Path Disable Control	0x0000_0000
PH_DOUT	GPIO_BA+0x1C8	R/W	PH Data Output Value	0x0000_0FF0
PH_DATMSK	GPIO_BA+0x1CC	R/W	PH Data Output Write Mask	0x0000_0000
PH_PIN	GPIO_BA+0x1D0	R	PH Pin Value	0x0000_XXXX
PH_DBEN	GPIO_BA+0x1D4	R/W	PH De-bounce Enable Control Register	0x0000_0000
PH_INTTYPE	GPIO_BA+0x1D8	R/W	PH Interrupt Trigger Type Control	0x0000_0000
PH_INTEN	GPIO_BA+0x1DC	R/W	PH Interrupt Enable Control Register	0x0000_0000
PH_INTSRC	GPIO_BA+0x1E0	R/W	PH Interrupt Source Flag	0x0000_XXXX
PH_SMTEN	GPIO_BA+0x1E4	R/W	PH Input Schmitt Trigger Enable Register	0x0000_0000
PH_SLEWCTL	GPIO_BA+0x1E8	R/W	PH High Slew Rate Control Register	0x0000_0000
PH_PUSEL	GPIO_BA+0x1F0	R/W	PH Pull-up and Pull-down Selection Register	0x0000_0000
PH_DBCTL	GPIO_BA+0x1F4	R/W	PH Interrupt De-bounce Control Register	0x0000_0020
INT0_INNF	GPIO_BA+0x450	R/W	INT0 Input Noise Filter Register	0x0000_0000
INT1_INNF	GPIO_BA+0x454	R/W	INT1 Input Noise Filter Register	0x0000_0000
INT2_INNF	GPIO_BA+0x458	R/W	INT2 Input Noise Filter Register	0x0000_0000
INT3_INNF	GPIO_BA+0x45C	R/W	INT3 Input Noise Filter Register	0x0000_0000
INT4_INNF	GPIO_BA+0x460	R/W	INT4 Input Noise Filter Register	0x0000_0000
INT5_INNF	GPIO_BA+0x464	R/W	INT5 Input Noise Filter Register	0x0000_0000
INT6_INNF	GPIO_BA+0x468	R/W	INT6 Input Noise Filter Register	0x0000_0000
INT7_INNF	GPIO_BA+0x46C	R/W	INT7 Input Noise Filter Register	0x0000_0000
INT_EDETCTL	GPIO_BA+0x490	R/W	INT Edge Detect Control Register	0x0000_0000
INT_EDINTEN	GPIO_BA+0x498	R/W	INT Edge Detect Interrupt Enable Control Register	0x0000_0000
INT_EDSTS	GPIO_BA+0x49C	R/W	INT Edge Detect Interrupt Flag Register	0x0000_0000
PAn_PDIO **n=0,1..15**	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO **n=0,1..15**	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO **n=0,1..14**	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X

PDn_PDIO **n=0,1..15**	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.n Pin Data Input/Output Register	0x0000_000X
PEn_PDIO **n=0,1..15**	GPIO_BA+0x900+(0x04 * n)	R/W	GPIO PE.n Pin Data Input/Output Register	0x0000_000X
PFn_PDIO **n=0,1..11**	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n Pin Data Input/Output Register	0x0000_000X
PGn_PDIO **n=2,3,4,9,10,11,12,13,14,15**	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n Pin Data Input/Output Register	0x0000_000X
PHn_PDIO **n=4,5..11**	GPIO_BA+0x9C0+(0x04 * n)	R/W	GPIO PH.n Pin Data Input/Output Register	0x0000_000X

6.6.7 Register Description

Port A-H I/O Mode Control (Px_MODE)

Register	Offset	R/W	Description	Reset Value
PA_MODE	GPIO_BA+0x000	R/W	PA I/O Mode Control	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0x0000_0000
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0x0000_0000
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0x0000_0000
PE_MODE	GPIO_BA+0x100	R/W	PE I/O Mode Control	0x0000_0000
PF_MODE	GPIO_BA+0x140	R/W	PF I/O Mode Control	0x0000_0000
PG_MODE	GPIO_BA+0x180	R/W	PG I/O Mode Control	0x0000_0000
PH_MODE	GPIO_BA+0x1C0	R/W	PH I/O Mode Control	0x0000_0000

31	30	29	28	27	26	25	24
MODE15		MODE14		MODE13		MODE12	
23	22	21	20	19	18	17	16
MODE11		MODE10		MODE9		MODE8	
15	14	13	12	11	10	9	8
MODE7		MODE6		MODE5		MODE4	
7	6	5	4	3	2	1	0
MODE3		MODE2		MODE1		MODE0	

Bits	Description	
[2n+1:2n] n=0,1..15	**MODEn**	**Port A-H I/O Pin[n] Mode Control** Determine each I/O mode of Px.n pins. 00 = Px.n is in Input mode. 01 = Px.n is in Push-pull Output mode. 10 = Px.n is in Open-drain Output mode. 11 = Px.n is in Quasi-bidirectional mode. **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-H Digital Input Path Disable Control (Px_DINOFF)

Register	Offset	R/W	Description	Reset Value
PA_DINOFF	GPIO_BA+0x004	R/W	PA Digital Input Path Disable Control	0x0000_0000
PB_DINOFF	GPIO_BA+0x044	R/W	PB Digital Input Path Disable Control	0x0000_0000
PC_DINOFF	GPIO_BA+0x084	R/W	PC Digital Input Path Disable Control	0x0000_0000
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD Digital Input Path Disable Control	0x0000_0000
PE_DINOFF	GPIO_BA+0x104	R/W	PE Digital Input Path Disable Control	0x0000_0000
PF_DINOFF	GPIO_BA+0x144	R/W	PF Digital Input Path Disable Control	0x0000_0000
PG_DINOFF	GPIO_BA+0x184	R/W	PG Digital Input Path Disable Control	0x0000_0000
PH_DINOFF	GPIO_BA+0x1C4	R/W	PH Digital Input Path Disable Control	0x0000_0000

31	30	29	28	27	26	25	24
DINOFF							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DINOFF							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[n+16] n=0,1..15	**DINOFF[n]**	**Port A-H Pin[n] Digital Input Path Disable Bit** Each of these bits is used to control if the digital input path of corresponding Px.n pin is disabled. If input is analog signal, users can disable Px.n digital input path to avoid input current leakage. 0 = Px.n digital input path Enabled. 1 = Px.n digital input path Disabled (digital input tied to low). **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.
[15:0]	**Reserved**	Reserved.

Port A-H Data Output Value (Px_DOUT)

Register	Offset	R/W	Description	Reset Value
PA_DOUT	GPIO_BA+0x008	R/W	PA Data Output Value	0x0000_FFFF
PB_DOUT	GPIO_BA+0x048	R/W	PB Data Output Value	0x0000_FFFF
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_7FFF
PD_DOUT	GPIO_BA+0x0C8	R/W	PD Data Output Value	0x0000_FFFF
PE_DOUT	GPIO_BA+0x108	R/W	PE Data Output Value	0x0000_FFFF
PF_DOUT	GPIO_BA+0x148	R/W	PF Data Output Value	0x0000_0FFF
PG_DOUT	GPIO_BA+0x188	R/W	PG Data Output Value	0x0000_FE1C
PH_DOUT	GPIO_BA+0x1C8	R/W	PH Data Output Value	0x0000_0FF0

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DOUT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DOUT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DOUT[n]**	**Port A-H Pin[n] Output Value** Each of these bits controls the status of a Px.n pin when the Px.n is configured as Push-pull output, Open-drain output or Quasi-bidirectional mode. 0 = Px.n will drive Low if the Px.n pin is configured as Push-pull output, Open-drain output or Quasi-bidirectional mode. 1 = Px.n will drive High if the Px.n pin is configured as Push-pull output or Quasi-bidirectional mode. **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-H Data Output Write Mask (Px_DATMSK)

Register	Offset	R/W	Description	Reset Value
PA_DATMSK	GPIO_BA+0x00C	R/W	PA Data Output Write Mask	0x0000_0000
PB_DATMSK	GPIO_BA+0x04C	R/W	PB Data Output Write Mask	0x0000_0000
PC_DATMSK	GPIO_BA+0x08C	R/W	PC Data Output Write Mask	0x0000_0000
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD Data Output Write Mask	0x0000_0000
PE_DATMSK	GPIO_BA+0x10C	R/W	PE Data Output Write Mask	0x0000_0000
PF_DATMSK	GPIO_BA+0x14C	R/W	PF Data Output Write Mask	0x0000_0000
PG_DATMSK	GPIO_BA+0x18C	R/W	PG Data Output Write Mask	0x0000_0000
PH_DATMSK	GPIO_BA+0x1CC	R/W	PH Data Output Write Mask	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DATMSK							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATMSK							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DATMSK[n]**	**Port A-H Pin[n] Data Output Write Mask** These bits are used to protect the corresponding DOUT (Px_DOUT[n]) bit. When the DATMSK (Px_DATMSK[n]) bit is set to 1, the corresponding DOUT (Px_DOUT[n]) bit is protected. If the write signal is masked, writing data to the protect bit is ignored. 0 = Corresponding DOUT (Px_DOUT[n]) bit can be updated. 1 = Corresponding DOUT (Px_DOUT[n]) bit protected. **Note 1:** This function only protects the corresponding DOUT (Px_DOUT[n]) bit, and will not protect the corresponding PDIO (Pxn_PDIO[0]) bit. **Note 2:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-H Pin Value (Px_PIN)

Register	Offset	R/W	Description	Reset Value
PA_PIN	GPIO_BA+0x010	R	PA Pin Value	0x0000_XXXX
PB_PIN	GPIO_BA+0x050	R	PB Pin Value	0x0000_XXXX
PC_PIN	GPIO_BA+0x090	R	PC Pin Value	0x0000_XXXX
PD_PIN	GPIO_BA+0x0D0	R	PD Pin Value	0x0000_XXXX
PE_PIN	GPIO_BA+0x110	R	PE Pin Value	0x0000_XXXX
PF_PIN	GPIO_BA+0x150	R	PF Pin Value	0x0000_XXXX
PG_PIN	GPIO_BA+0x190	R	PG Pin Value	0x0000_XXXX
PH_PIN	GPIO_BA+0x1D0	R	PH Pin Value	0x0000_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PIN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PIN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**PIN[n]**	**Port A-H Pin[n] Pin Value** Each bit of the register reflects the actual status of the respective Px.n pin. If the bit is 1, it indicates the corresponding pin status is high; else the pin status is low. **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-H De-bounce Enable Control Register (Px_DBEN)

Register	Offset	R/W	Description	Reset Value
PA_DBEN	GPIO_BA+0x014	R/W	PA De-bounce Enable Control Register	0x0000_0000
PB_DBEN	GPIO_BA+0x054	R/W	PB De-bounce Enable Control Register	0x0000_0000
PC_DBEN	GPIO_BA+0x094	R/W	PC De-bounce Enable Control Register	0x0000_0000
PD_DBEN	GPIO_BA+0x0D4	R/W	PD De-bounce Enable Control Register	0x0000_0000
PE_DBEN	GPIO_BA+0x114	R/W	PE De-bounce Enable Control Register	0x0000_0000
PF_DBEN	GPIO_BA+0x154	R/W	PF De-bounce Enable Control Register	0x0000_0000
PG_DBEN	GPIO_BA+0x194	R/W	PG De-bounce Enable Control Register	0x0000_0000
PH_DBEN	GPIO_BA+0x1D4	R/W	PH De-bounce Enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DBEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBEN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DBEN[n]**	**Port A-H Pin[n] Input Signal De-bounce Enable Bit** The DBEN[n] bit is used to enable the de-bounce function for each corresponding bit. If the input signal pulse width cannot be sampled by continuous two de-bounce sample cycle, the input signal transition is seen as the signal bounce and will not trigger the interrupt. The de-bounce clock source is controlled by DBCLKSRC (GPIO_DBCTL [4]), one de-bounce sample cycle period is controlled by DBCLKSEL (GPIO_DBCTL [3:0]). 0 = Px.n de-bounce function Disabled. 1 = Px.n de-bounce function Enabled. The de-bounce function is valid only for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored. **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-H Interrupt Type Control (Px_INTTYPE)

Register	Offset	R/W	Description	Reset Value
PA_INTTYPE	GPIO_BA+0x018	R/W	PA Interrupt Trigger Type Control	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB Interrupt Trigger Type Control	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC Interrupt Trigger Type Control	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD Interrupt Trigger Type Control	0x0000_0000
PE_INTTYPE	GPIO_BA+0x118	R/W	PE Interrupt Trigger Type Control	0x0000_0000
PF_INTTYPE	GPIO_BA+0x158	R/W	PF Interrupt Trigger Type Control	0x0000_0000
PG_INTTYPE	GPIO_BA+0x198	R/W	PG Interrupt Trigger Type Control	0x0000_0000
PH_INTTYPE	GPIO_BA+0x1D8	R/W	PH Interrupt Trigger Type Control	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TYPE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TYPE							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**TYPE[n]**	**Port A-H Pin[n] Edge or Level Detection Interrupt Trigger Type Control** TYPE (Px_INTTYPE[n]) bit is used to control the triggered interrupt is by level trigger or by edge trigger. If the interrupt is by edge trigger, the trigger source can be controlled by de-bounce. If the interrupt is by level trigger, the input source is sampled by one HCLK clock and generates the interrupt. 0 = Edge trigger interrupt. 1 = Level trigger interrupt. If the pin is set as the level trigger interrupt, only one level can be set on the registers RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n]). If both levels to trigger interrupt are set, the setting is ignored and and there is a chance that unexpected interrupts will occur. The de-bounce function is valid only for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored. **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-H Interrupt Enable Control Register (Px_INTEN)

Register	Offset	R/W	Description	Reset Value
PA_INTEN	GPIO_BA+0x01C	R/W	PA Interrupt Enable Control Register	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB Interrupt Enable Control Register	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC Interrupt Enable Control Register	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD Interrupt Enable Control Register	0x0000_0000
PE_INTEN	GPIO_BA+0x11C	R/W	PE Interrupt Enable Control Register	0x0000_0000
PF_INTEN	GPIO_BA+0x15C	R/W	PF Interrupt Enable Control Register	0x0000_0000
PG_INTEN	GPIO_BA+0x19C	R/W	PG Interrupt Enable Control Register	0x0000_0000
PH_INTEN	GPIO_BA+0x1DC	R/W	PH Interrupt Enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
RHIEN							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RHIEN							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FLIEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FLIEN							

Bits	Description	
[n+16] n=0,1..15	**RHIEN[n]**	**Port A-H Pin[n] Rising Edge or High Level Interrupt Trigger Type Enable Bit** The RHIEN (Px_INTEN[n+16]) bit is used to enable the interrupt for each of the corresponding input Px.n pin. Set bit to 1 also enable the pin wake-up function. When setting the RHIEN (Px_INTEN[n+16]) bit to 1 : If the interrupt is level trigger (TYPE (Px_INTTYPE[n]) bit is set to 1), the input Px.n pin will generate the interrupt while this pin state is at high level. If the interrupt is edge trigger (TYPE (Px_INTTYPE[n]) bit is set to 0), the input Px.n pin will generate the interrupt while this pin state changed from low to high. 0 = Px.n level high or low to high interrupt Disabled. 1 = Px.n level high or low to high interrupt Enabled. **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.
[n] n=0,1..15	**FLIEN[n]**	**Port A-H Pin[n] Falling Edge or Low Level Interrupt Trigger Type Enable Bit** The FLIEN (Px_INTEN[n]) bit is used to enable the interrupt for each of the corresponding input Px.n pin. Set bit to 1 also enable the pin wake-up function. When setting the FLIEN (Px_INTEN[n]) bit to 1 : If the interrupt is level trigger (TYPE (Px_INTTYPE[n]) bit is set to 1), the input Px.n pin will generate the interrupt while this pin state is at low level. If the interrupt is edge trigger(TYPE (Px_INTTYPE[n]) bit is set to 0), the input Px.n pin

		will generate the interrupt while this pin state changed from high to low. 0 = Px.n level low or high to low interrupt Disabled. 1 = Px.n level low or high to low interrupt Enabled. **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-H Interrupt Source Flag (Px_INTSRC)

Register	Offset	R/W	Description	Reset Value
PA_INTSRC	GPIO_BA+0x020	R/W	PA Interrupt Source Flag	0x0000_XXXX
PB_INTSRC	GPIO_BA+0x060	R/W	PB Interrupt Source Flag	0x0000_XXXX
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC Interrupt Source Flag	0x0000_XXXX
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD Interrupt Source Flag	0x0000_XXXX
PE_INTSRC	GPIO_BA+0x120	R/W	PE Interrupt Source Flag	0x0000_XXXX
PF_INTSRC	GPIO_BA+0x160	R/W	PF Interrupt Source Flag	0x0000_XXXX
PG_INTSRC	GPIO_BA+0x1A0	R/W	PG Interrupt Source Flag	0x0000_XXXX
PH_INTSRC	GPIO_BA+0x1E0	R/W	PH Interrupt Source Flag	0x0000_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
INTSRC							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTSRC							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**INTSRC[n]**	**Port A-H Pin[n] Interrupt Source Flag** Write Operation : 0 = No action. 1 = Clear the corresponding pending interrupt. Read Operation : 0 = No interrupt at Px.n. 1 = Px.n generates an interrupt. **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-H Input Schmitt Trigger Enable Register (Px_SMTEN)

Register	Offset	R/W	Description	Reset Value
PA_SMTEN	GPIO_BA+0x024	R/W	PA Input Schmitt Trigger Enable Register	0x0000_0000
PB_SMTEN	GPIO_BA+0x064	R/W	PB Input Schmitt Trigger Enable Register	0x0000_0000
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC Input Schmitt Trigger Enable Register	0x0000_0000
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD Input Schmitt Trigger Enable Register	0x0000_0000
PE_SMTEN	GPIO_BA+0x124	R/W	PE Input Schmitt Trigger Enable Register	0x0000_0000
PF_SMTEN	GPIO_BA+0x164	R/W	PF Input Schmitt Trigger Enable Register	0x0000_0000
PG_SMTEN	GPIO_BA+0x1A4	R/W	PG Input Schmitt Trigger Enable Register	0x0000_0000
PH_SMTEN	GPIO_BA+0x1E4	R/W	PH Input Schmitt Trigger Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SMTEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMTEN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**SMTEN[n]**	**Port A-H Pin[n] Input Schmitt Trigger Enable Bit** 0 = Px.n input schmitt trigger function Disabled. 1 = Px.n input schmitt trigger function Enabled. **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-H High Slew Rate Control Register (Px_SLEWCTL)

Register	Offset	R/W	Description	Reset Value
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA High Slew Rate Control Register	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB High Slew Rate Control Register	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC High Slew Rate Control Register	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD High Slew Rate Control Register	0x0000_0000
PE_SLEWCTL	GPIO_BA+0x128	R/W	PE High Slew Rate Control Register	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF High Slew Rate Control Register	0x0000_0000
PG_SLEWCTL	GPIO_BA+0x1A8	R/W	PG High Slew Rate Control Register	0x0000_0000
PH_SLEWCTL	GPIO_BA+0x1E8	R/W	PH High Slew Rate Control Register	0x0000_0000

31	30	29	28	27	26	25	24
HSREN15		HSREN14		HSREN13		HSREN12	
23	22	21	20	19	18	17	16
HSREN11		HSREN10		HSREN9		HSREN8	
15	14	13	12	11	10	9	8
HSREN07		HSREN6		HSREN5		HSREN4	
7	6	5	4	3	2	1	0
HSREN03		HSREN2		HSREN1		HSREN0	

Bits	Description	
[2n+1:2n] n=0,1..15	**HSRENn**	**Port A-H Pin[n] High Slew Rate Control** 00 = Px.n output with normal slew rate mode (maximum 40 MHz at 2.7V). 01 = Px.n output with high slew rate mode (maximum 72 MHz at 2.7V). Others = Reserved. **Note:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-H Pull-up and Pull-down Selection Register (Px_PUSEL)

Register	Offset	R/W	Description	Reset Value
PA_PUSEL	GPIO_BA+0x030	R/W	PA Pull-up and Pull-down Selection Register	0x0000_0000
PB_PUSEL	GPIO_BA+0x070	R/W	PB Pull-up and Pull-down Selection Register	0x0000_0000
PC_PUSEL	GPIO_BA+0x0B0	R/W	PC Pull-up and Pull-down Selection Register	0x0000_0000
PD_PUSEL	GPIO_BA+0x0F0	R/W	PD Pull-up and Pull-down Selection Register	0x0000_0000
PE_PUSEL	GPIO_BA+0x130	R/W	PE Pull-up and Pull-down Selection Register	0x0000_0000
PF_PUSEL	GPIO_BA+0x170	R/W	PF Pull-up and Pull-down Selection Register	0x0000_0000
PG_PUSEL	GPIO_BA+0x1B0	R/W	PG Pull-up and Pull-down Selection Register	0x0000_0000
PH_PUSEL	GPIO_BA+0x1F0	R/W	PH Pull-up and Pull-down Selection Register	0x0000_0000

31	30	29	28	27	26	25	24
PUSEL15		PUSEL14		PUSEL13		PUSEL12	
23	22	21	20	19	18	17	16
PUSEL11		PUSEL10		PUSEL9		PUSEL8	
15	14	13	12	11	10	9	8
PUSEL7		PUSEL6		PUSEL5		PUSEL4	
7	6	5	4	3	2	1	0
PUSEL3		PUSEL2		PUSEL1		PUSEL0	

Bits	Description	
[2n+1:2n] n=0,1..15	**PUSELn**	**Port A-H Pin[n] Pull-up and Pull-down Enable Register** Determine each I/O Pull-up/pull-down of Px.n pins. 00 = Px.n pull-up and pull-down disable. 01 = Px.n pull-up enable. 10 = Px.n pull-down enable. 11 = Reserved. **Note 1:** Basically, the pull-up control and pull-down control has following behavior limitation The independent pull-up control register only valid when MODEn set as tri-state and open-drain mode The independent pull-down control register only valid when MODEn set as tri-state mode When both pull-up pull-down is set as 1 at “tri-state” mode, keep I/O in tri-state mode **Note 2:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Port A-J Interrupt De-bounce Control Register (Px_DBCTL)

Register	Offset	R/W	Description	Reset Value
PA_DBCTL	GPIO_BA+0x034	R/W	PA Interrupt De-bounce Control Register	0x0000_0020
PB_DBCTL	GPIO_BA+0x074	R/W	PB Interrupt De-bounce Control Register	0x0000_0020
PC_DBCTL	GPIO_BA+0x0B4	R/W	PC Interrupt De-bounce Control Register	0x0000_0020
PD_DBCTL	GPIO_BA+0x0F4	R/W	PD Interrupt De-bounce Control Register	0x0000_0020
PE_DBCTL	GPIO_BA+0x134	R/W	PE Interrupt De-bounce Control Register	0x0000_0020
PF_DBCTL	GPIO_BA+0x174	R/W	PF Interrupt De-bounce Control Register	0x0000_0020
PG_DBCTL	GPIO_BA+0x1B4	R/W	PG Interrupt De-bounce Control Register	0x0000_0020
PH_DBCTL	GPIO_BA+0x1F4	R/W	PH Interrupt De-bounce Control Register	0x0000_0020

31	30	29	28	27	26	25	24
DBCLKBUSY	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**ICLKON**	**DBCLKSRC**	**DBCLKSEL**			

Bits	Description	
[31]	**DBCLKBUSY**	**De-bounce Clock Switching Busy Flag (Read Only)** This bit is set when de-bounce clock source is changed by setting DBCLKSRC(Px_DBCTL[4]). It is cleared after de-bounce clock source switching is finished. De-bounce function can work normally after de-bounce clock switch done. 0 = De-bounce clock switch done. 1 = De-bounce clock is switching.
[30:6]	**Reserved**	Reserved.
[5]	**ICLKON**	**Interrupt Clock on Mode** 0 = Edge detection circuit is active only if I/O pin corresponding RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n]) bit is set to 1. If corresponding RHIEN or FLIEN is 0, the clock of I/O detect circuit is stopped and interrupt source flag(Px_INTSRC) cannot be clear also. 1 = All I/O pins edge detection circuit is always active after reset. **Note 1:** It is recommended to disable this bit to save system power if no special application concern. **Note 2:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

Bits	Description	
[4]	**DBCLKSRC**	**De-bounce Counter Clock Source Selection** 0 = De-bounce counter clock source is the HCLK. 1 = De-bounce counter clock source is the 32 kHz internal low speed RC oscillator (LIRC).
[3:0]	**DBCLKSEL**	**De-bounce Sampling Cycle Selection** 0000 = Sample interrupt input once per 1 clocks. 0001 = Sample interrupt input once per 2 clocks. 0010 = Sample interrupt input once per 4 clocks. 0011 = Sample interrupt input once per 8 clocks. 0100 = Sample interrupt input once per 16 clocks. 0101 = Sample interrupt input once per 32 clocks. 0110 = Sample interrupt input once per 64 clocks. 0111 = Sample interrupt input once per 128 clocks. 1000 = Sample interrupt input once per 256 clocks. 1001 = Sample interrupt input once per 2*256 clocks. 1010 = Sample interrupt input once per 4*256 clocks. 1011 = Sample interrupt input once per 8*256 clocks. 1100 = Sample interrupt input once per 16*256 clocks. 1101 = Sample interrupt input once per 32*256 clocks. 1110 = Sample interrupt input once per 64*256 clocks. 1111 = Sample interrupt input once per 128*256 clocks.

IINT0~7 Input Noise Filter Register (INTx_INNF)

Register	Offset	R/W	Description	Reset Value
INT0_INNF	GPIO_BA+0x450	R/W	INT0 Input Noise Filter Register	0x0000_0000
INT1_INNF	GPIO_BA+0x454	R/W	INT1 Input Noise Filter Register	0x0000_0000
INT2_INNF	GPIO_BA+0x458	R/W	INT2 Input Noise Filter Register	0x0000_0000
INT3_INNF	GPIO_BA+0x45C	R/W	INT3 Input Noise Filter Register	0x0000_0000
INT4_INNF	GPIO_BA+0x460	R/W	INT4 Input Noise Filter Register	0x0000_0000
INT5_INNF	GPIO_BA+0x464	R/W	INT5 Input Noise Filter Register	0x0000_0000
INT6_INNF	GPIO_BA+0x468	R/W	INT6 Input Noise Filter Register	0x0000_0000
INT7_INNF	GPIO_BA+0x46C	R/W	INT7 Input Noise Filter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**NFCNT**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**NFSEL**			**Reserved**			**NFEN**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**NFCNT**	**Noise Filter Count** The register bits control the filter counter to count from 0 to NFCNT.
[7]	**Reserved**	Reserved.
[6:4]	**NFSEL**	**Noise Filter Clock Selection** 000 = Filter clock is HCLK. 001 = Filter clock is HCLK/2. 010 = Filter clock is HCLK/4. 011 = Filter clock is HCLK/8. 100 = Filter clock is HCLK/16. 101 = Filter clock is HCLK/32. 110 = Filter clock is HCLK/64. 111 = Filter clock is HCLK/128.
[3:1]	**Reserved**	Reserved.
[0]	**NFEN**	**Noise Filter Enable** 0 = Noise Filter function Disabled.

		1 = Noise Filter function Enabled.

INT Edge Detect Control Register (INT_EDETCTL)

Register	Offset	R/W	Description	Reset Value
INT_EDETCTL	GPIO_BA+0x490	R/W	INT Edge Detect Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
EDETCTL7		**EDETCTL6**		**EDETCTL5**		**EDETCTL4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EDETCTL3		**EDETCTL2**		**EDETCTL1**		**EDETCTL0**	

Bits	Description	
[31:16]	**Reserved**	Reserved.
[2n+1:2n] n=0,1..7	**EDETCTLn**	**INTn Edge Detect Control Bits** 00 = Not detect. 01 = INTn low to high detection Enabled. 10 = INTn high to low detection Enabled. 11 = INTn both low to high and high to low detection Enabled.

INT Egde Detect Interrupt Enable Control Register (INT_EDINTEN)

Register	Offset	R/W	Description	Reset Value
INT_EDINTEN	GPIO_BA+0x498	R/W	INT Edge Detect Interrupt Enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EDIEN7	**EDIEN6**	**EDIEN5**	**EDIEN4**	**EDIEN3**	**EDIEN2**	**EDIEN1**	**EDIEN0**

Bits	Description	
[31:8]	**Reserved**	Reseved.
[n] n=0,1..7	**EDIENn**	**INTn Edge Detect Interrupt Enable Bit** 0 = INTn Edge Detect Interrupt Disabled. 1 = INTn Edge Detect Interrupt Enabled.

INT Edge Detect Interrupt Flag Register (INT_EDSTS)

Register	Offset	R/W	Description	Reset Value
INT_EDSTS	GPIO_BA+0x49C	R/W	INT Edge Detect Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EDIF7	**EDIF6**	**EDIF5**	**EDIF4**	**EDIF3**	**EDIF2**	**EDIF1**	**EDIF0**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**EDIFn**	**INTn Edge Detect Interrupt Flag** 0 = No Edge Detection happened. 1 = Rising Edge or Falling edge has been detected. **Note:** This bit is cleared by writing 1 to it.

GPIO Px.n Pin Data Input/Outut Register (Pxn_PDIO)

Register	Offset	R/W	Description	Reset Value
PAn_PDIO **n=0,1..15**	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO **n=0,1..15**	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO **n=0,1..14**	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO **n=0,1..15**	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.n Pin Data Input/Output Register	0x0000_000X
PEn_PDIO **n=0,1..15**	GPIO_BA+0x900+(0x04 * n)	R/W	GPIO PE.n Pin Data Input/Output Register	0x0000_000X
PFn_PDIO **n=0,1..11**	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n Pin Data Input/Output Register	0x0000_000X
PGn_PDIO **n=2,3,4,9,10,11,12,13,14,15**	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n Pin Data Input/Output Register	0x0000_000X
PHn_PDIO **n=4,5..11**	GPIO_BA+0x9C0+(0x04 * n)	R/W	GPIO PH.n Pin Data Input/Output Register	0x0000_000X

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**PDIO**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PDIO**	**GPIO Px.n Pin Data Input/Output** Writing this bit can control one GPIO pin output value. 0 = Corresponding GPIO pin set to low. 1 = Corresponding GPIO pin set to high. Read this register to get GPIO pin status. For example, writing PA0_PDIO will reflect the written value to bit DOUT (Px_DOUT[0]), reading PA0_PDIO will return the value of PIN (PA_PIN[0]). **Note 1:** The writing operation will not be affected by register DATMSK (Px_DATMSK[n]). **Note 2:** For more information about Px.n, please refer to the "PIN CONFIGURATION" chapter.

6.7 Low Power General Purpose I/O (LPGPIO)

6.7.1 Overview

This chip has up to 8 Low Power General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 8 pins are named as LPIO0, LPIO1, LPIO2, LPIO3, LPIO4, LPIO5, LPIO6 and LPIO7. Each of the 8 pins is independent and has the corresponding register bits to control the pin function and data. These 8 pins also support output data set/reset registers to allow individual set/reset bitwise control.

6.7.2 Features

- Supports function operating in NPD0/1/2/3/4
- Two I/O modes:
  - Push-Pull Output mode
  - Input only with high impendence mode
- Supports pin output data individual bit set/reset

6.7.3 Block Diagram

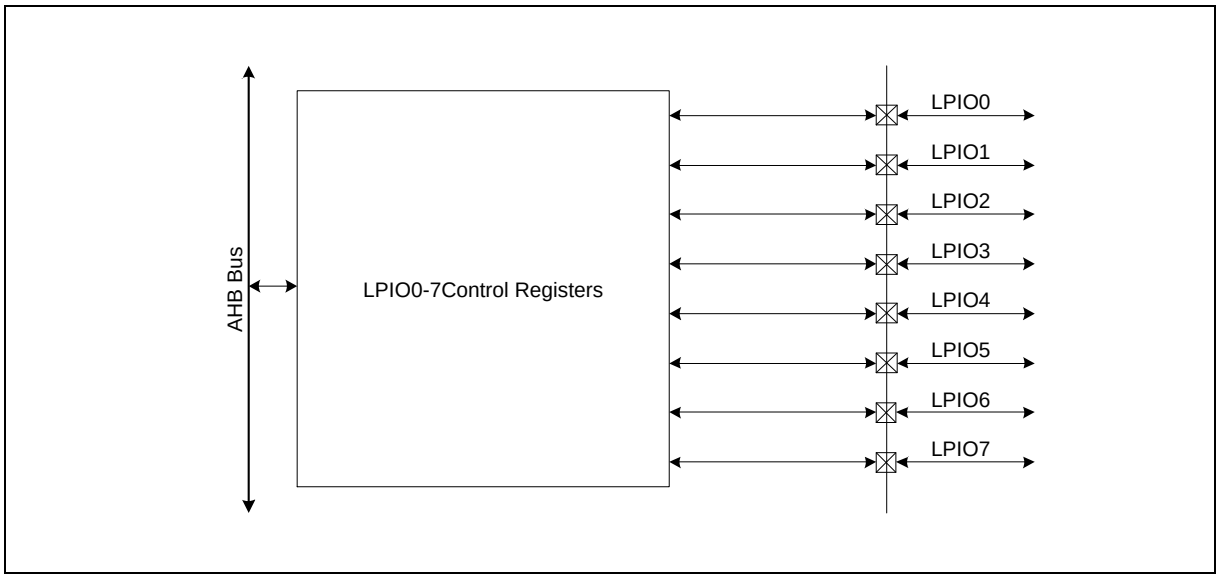


Figure 6.7-1 LPGPIO Controller Block Diagram

6.7.4 Basic Configuration

- Clock source Configuration
  - Enable LPGPIO clock in LPGPIOCKEN (LPSCC_CLKEN0 [1]).
- Reset configuration
  - Reset LPGPIO in LPGPIORST (LPSCC_IPRST0 [1])

6.7.5 Functional Description

6.7.5.1 Input Mode

Set MODE (LPGPIO_MODE[n]) to 0 as the LPIOn is in Input mode and the I/O pin is in tri-state (high impedance) without output drive capability. The PIN (LPGPIO_PIN[n]) value reflects the status of the corresponding port pins.

6.7.5.2 Push-pull Output Mode

LPGPIO only supports Push-pull Output mode, and the I/O pin supports digital output function with source/sink current capability. Figure 6.6-2 shows the diagram of Push-pull Output Mode. The bit value in the corresponding DOUT (LPGPIO_DOUT[n]) is driven on the pin.

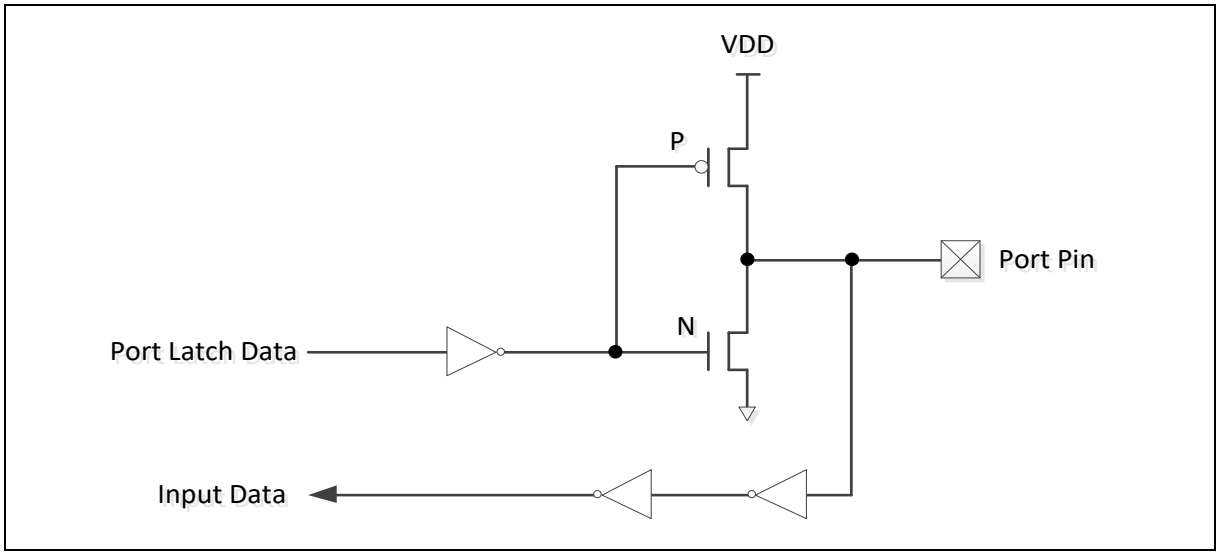


Figure 6.7-2 Push-Pull Output

6.7.5.3 Auto-operation Mode

LPGPIO allows function operating when chip enters NPD0/1/2/3/4 mode, and pin output data can be controlled by DSRST(LPGPIO_DSRST[n]) and DRST(LPGPIO_DRST[n]) and pin input data can be detected by PIN(LPGPIO_PIN[n]). User should set LPPDMA_EN (LPGPIO_MODE[31]) to 1 so that LPPDMA can access LPGPIO while chip is in NPD0/1/2/3/4 mode.

When chip enters NPD3/4 mode, slew rate setting is forced to 2'b01 and schmitt trigger control can be enabled or disabled by SCHDIS (LPSCC_IOHCTL[0]).

6.7.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
LPGPIO Base Address: **LPGPIO_BA = 0x4003_A000**				
LPGPIO_MODE	LPGPIO_BA+0x000	R/W	LPIOn Mode Control	0x0000_0000
LPGPIO_DOUT	LPGPIO_BA+0x004	R/W	LPIOn Data Output Value	0x0000_00FF
LPGPIO_PIN	LPGPIO_BA+0x008	R	LPIOn Pin Value	0x0000_00XX
LPGPIO_DSRST	LPGPIO_BA+0x00C	W	LPIOn Data Output Set and Reset Control	0x0000_0000
LPGPIO_DRST	LPGPIO_BA+0x010	W	LPIOn Data Output Reset Control	0x0000_0000

6.7.7 Register Description

LPIOn Mode Control (LPGPIO_MODE)

Register	Offset	R/W	Description	Reset Value
LPGPIO_MODE	LPGPIO_BA+0x000	R/W	LPIOn Mode Control	0x0000_0000

31	30	29	28	27	26	25	24
LPPDMA_EN	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
MODE							

Bits	Description	
[31]	**LPPDMA_EN**	**LPPDMA Enable Bit** This bit is used to enable LPPDMA to access LPGPIO when chip is in NPD0/1/2/3/4. 0 = LPPDMA cannot access LPGPIO when chip is in NPD0/1/2/3/4. 1 = LPPDMA can access LPGPIO when chip is in NPD0/1/2/3/4.
[30:8]	**Reserved**	Reserved.
[n] n=0,1..7	**MODE[n]**	**LPIOn I/O Pin Mode Control** Determine each I/O mode of LPIOn pins. 0 = LPIOn is in Input mode. 1 = LPIOn is in Push-pull Output mode.

LPIOn Data Output Value (LPGPIO_DOUT)

Register	Offset	R/W	Description	Reset Value
LPGPIO_DOUT	LPGPIO_BA+0x004	R/W	LPIOn Data Output Value	0x0000_00FF

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DOUT							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**DOUT[n]**	**LPIOn Output Value** Each of these bits controls the status of a LPIOn pin when the LPIOn is configured as Push-pull output mode. 0 = LPIOn will drive Low if the LPIOn pin is configured as Push-pull output mode. 1 = LPIOn will drive High if the LPIOn pin is configured as Push-pull output mode.

LPIOn Pin Value (LPGPIO_PIN)

Register	Offset	R/W	Description	Reset Value
LPGPIO_PIN	LPGPIO_BA+0x008	R	LPIOn Pin Value	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PIN							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**PIN[n]**	**LPIOn Pin Value** Each bit of the register reflects the actual status of the respective Pn pin. If the bit is 1, it indicates the corresponding pin status is high; else the pin status is low.

LPIOn Data Ouput Set and Reset Register (LPGPIO_DSRST)

Register	Offset	R/W	Description	Reset Value
LPGPIO_DSRST	LPGPIO_BA+0x00C	W	LPIOn Data Output Set and Reset Control	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DRESET							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DSET							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[n+16] n=0,1..7	**DRESET[n]**	**LPIOn Data Ouput Reset Control (Write Only)** Writing 1 to each bit can reset LPIOn pin output data to 0. 0 = No action. 1 = Reset LPIOn to 0. **Note 1 :** This bit field is write only, and reading this field will respond with 0. **Note 2 :** If DRESET[n] and DSET[n] are both set, DSET[n] has higher pirority. **Note 3 :** Writing 1 to DRESET[n] will make DOUT[n] reflect the reset value.
[15:8]	**Reserved**	Reserved.
[n] n=0,1..7	**DSET[n]**	**LPIOn Data Ouput Set Control (Write Only)** Writing 1 to each bit can set LPIOn pin output data to 1. 0 = No action. 1 = Reset LPIOn to 1. **Note 1 :** This bit field is write only, and reading this field will respond with 0. **Note 2 :** If DRESET[n] and DSET[n] are both set, DSET[n] has higher pirority. **Note 3 :** Writing 1 to DSET[n] will make DOUT[n] reflect the set value.

LPIOn Data Ouput Reset Register (LPGPIO_DRST)

Register	Offset	R/W	Description	Reset Value
LPGPIO_DRST	LPGPIO_BA+0x010	W	LPIOn Data Output Reset Control	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DRESET							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**DRESET[n]**	**LPIOn Data Ouput Reset Control (Write Only)** Writing 1 to each bit can reset LPIOn pin output data to 0. 0 = No action. 1 = Reset LPIOn to 0. **Note 1 :** This bit field is write only, and reading this field will respond with 0. **Note 2 :** Writing 1 to DRESET[n] will make DOUT[n] reflect the reset value.

6.8 PDMA Controller (PDMA)

6.8.1 Overview

The peripheral direct memory access (PDMA) controller is used to provide high-speed data transfer. PDMA can transfer data from one address to another without CPU intervention. This has the benefit of reducing the workload of CPU and keeps CPU resources free for other applications. PDMA has a total of 16 channels and each channel can perform transfer between memory and peripherals or between memory and memory.

6.8.2 Features

- Supports 16 independently configurable channels
- Supports selectable 2 level of priority (fixed priority or round-robin priority)
- Supports transfer data width of 8, 16, and 32 bits
- Supports source and destination address increment size can be byte, half-word, word or no increment
- Supports software and USB, UART, USCI, SPI, QSPI, ACMP, PWM, EINT, EPWM, I^2C, Timer, EADC, and DAC request
- Supports Scatter-gather mode to perform sophisticated transfer through the use of the descriptor link list table
- Supports single and burst transfer type
- Supports 2 time-out function channels

6.8.3 Block Diagram

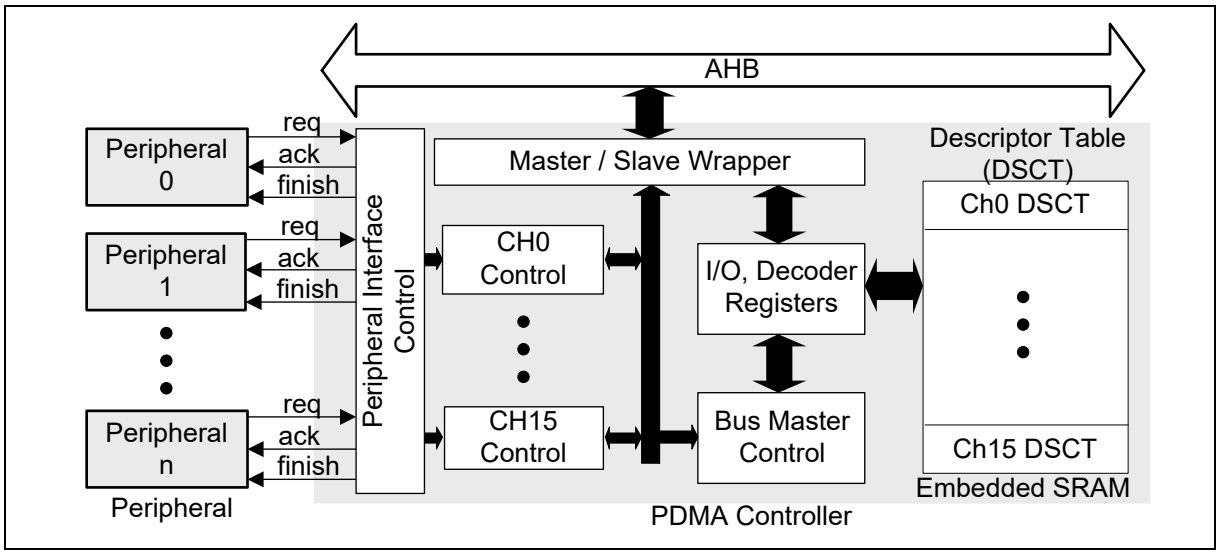


Figure 6.8-1 PDMA Controller Block Diagram

6.8.4 Basic Configuration

6.8.4.1 Basic Configuration of PDMA0

- Clock Source Configuration
  - Enable PDMA0 controller clock in PDMA0CKEN (CLK_AHBCLK [1]).
- Reset Configuration

- Reset PDMA0 controller in PDMA0RST (SYS_IPRST0[2]).

6.8.5 Functional Description

The PDMA controller transfers data from one address to another without CPU intervention. PDMA supports 16 independent channels, and only one channel transfers data at a time. Therefore, it supports two levels of channel priority: fixed and round-robin, and transfers data in order from the highest to the lowest priority channel.

PDMA supports two operation modes, Basic and Scatter-gather. Basic mode is used to perform one descriptor table transfer. Scatter-gather mode has more entries for each PDMA channel, and thus PDMA supports sophisticated transfer through the entries. The descriptor table entry data structure contains many transfer information including the transfer source address, transfer destination address, transfer count, burst size, transfer type and operation mode. Figure 6.9-2 shows the diagram of descriptor table (DSCT) data structure.

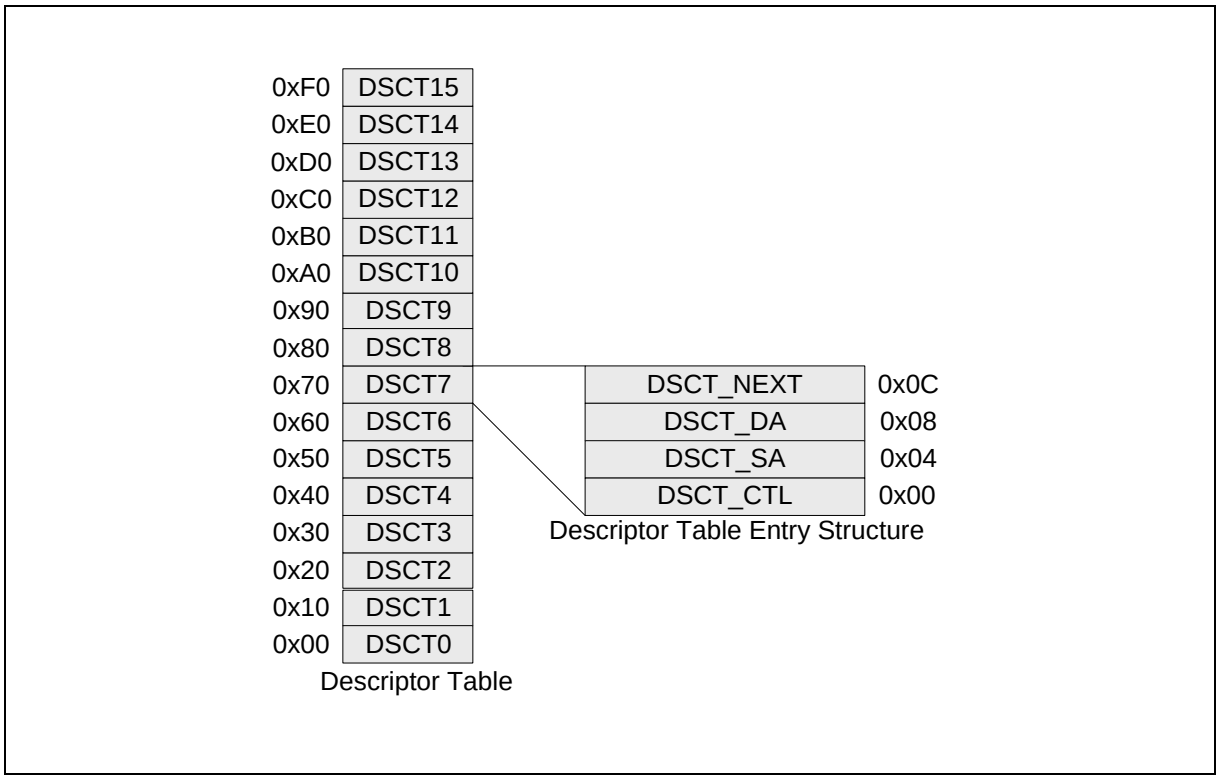


Figure 6.8-2 Descriptor Table Entry Structure

PDMA supports single and burst data transfer type and the request source can be from software or peripheral request. It also supports data transfer between memory to memory by software request. A single transfer means that software or peripheral is ready to transfer one data (every data needs one request), and the burst transfer means that software or peripherals will transfer multiple data (multiple data only need one request).

6.8.5.1 Channel Priority

PDMA supports two level channel priorities including fixed and round-robin priority. The fixed priority channel has higher priority than round-robin priority channel. If multiple channels are set as fixed or round-robin priority, the higher channel will have higher priority. The priority order is listed in Table 6.9-1.

PDMA_PRISET	Channel Number	Priority Setting	Arbitration Priority In Descending Order
1	15	Channel15, Fixed Priority	Highest

1	14	Channel14, Fixed Priority	---
---	---	---	---
1	0	Channel0, Fixed Priority	---
0	15	Channel15, Round-Robin Priority	---
0	14	Channel14, Round-Robin Priority	---
---	---	---	---
0	0	Channel0, Round-Robin Priority	Lowest

Table 6.8-1 Channel Priority Table

6.8.5.2 PDMA Operation Mode

PDMA supports two operation modes: Basic and Scatter-gather modes.

Basic Mode

Basic mode is used to perform one descriptor table transfer. This mode can be used to transfer data between memory and memory, peripherals and memory or peripherals and peripherals. If PDMA transfer data between peripherals and peripherals, user should make sure the transferred data is ready before peripherals send a request. PDMA operation mode can be set from OPMODE (PDMA_DSCTn_CTL[1:0], n denotes PDMA channel), the default setting is in idle state and recommend user configure the descriptor table in idle state. If operation mode is not in idle state, user re-configure channel setting may make some operation error.

User must fill the transfer count TXCNT (PDMA_DSCTn_CTL[31:16]) register and select transfer width TXWIDTH (PDMA_DSCTn_CTL[13:12]), destination address increment size DAINC (PDMA_DSCTn_CTL[11:10]), source address increment size SAINC (PDMA_DSCTn_CTL[9:8]), burst size BURSIZE (PDMA_DSCTn_CTL[6:4]) and transfer type TXTYPE (PDMA_DSCTn_CTL[2]), then PDMA will perform transfer operation in transfer state after receiving request signal. Finishing this task will generate an interrupt to CPU if corresponding PDMA interrupt bit INTENn (PDMA_INTEN[15:0]) is enabled and the operation mode will be updated to idle state as shown in Figure 6.9-3. If software configures the operation mode to idle state, PDMA will not perform any transfer and then clear this operation request. Finishing this task will also generate an interrupt to CPU if corresponding PDMA interrupt bit is enabled.

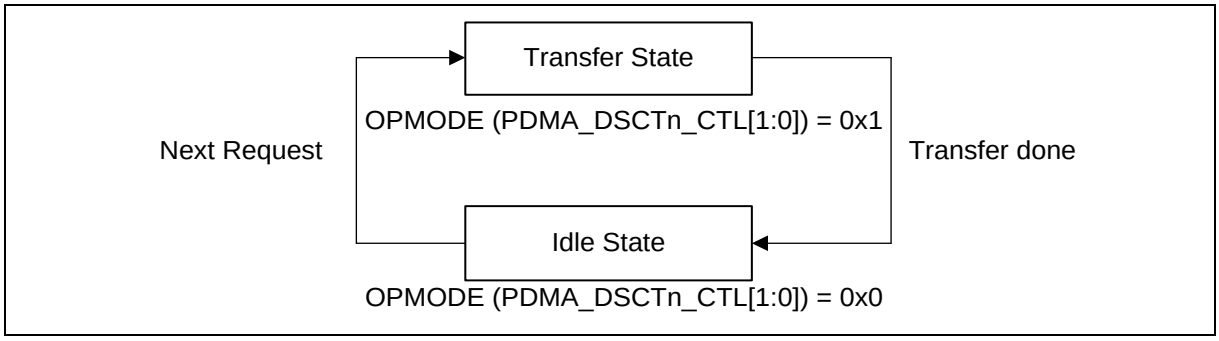


Figure 6.8-3 Basic Mode Finite State Machine

Scatter-gather Mode

Scatter-gather mode is a complex mode and can perform sophisticated transfer through the description link list table which is shown inFigure 6.9-4. It performs peripheral wrapper-around data transfer, and also performs multiple task data transfer between varied locations in system memory instead of a set of contiguous locations. Scatter-gather mode only needs a request to finish all table tasks, then OPMODE (PDMA_DSCTn_CTL[1:0]) will be cleared to idle state automatically.

In Scatter-gather mode, the table is just used for jumping to the next table entry. The first task will not perform any operation transfer. Finishing each task will generate an interrupt to CPU if corresponding PDMA interrupt bit is enabled and TBINTDIS (PDMA_DSCTn_CTL[7]) bit is “0” (when finishing task and TBINTDIS bit is “0”, corresponding TDIFn (PDMA_TDSTS[15:0]) flag will be asserted and if this bit is “1” TDIFn will not be active).

If channel 15 has been triggered, and the operation mode is in Scatter-gather mode (OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x2), the hardware will load the real PDMA information task from the address generated by adding PDMA_DSCTn_NEXT (link address) and PDMA_SCATBA (base address) registers. For example, base address is 0x2000_0000 (only MSB 16 bits valid in PDMA_SCATBA), the current link address is 0x0000_0100 (only LSB 16-bits without last two bits [1:0] valid in PDMA_DSCTn_NEXT), and then the next DSCT entry start address is 0x2000_0100.

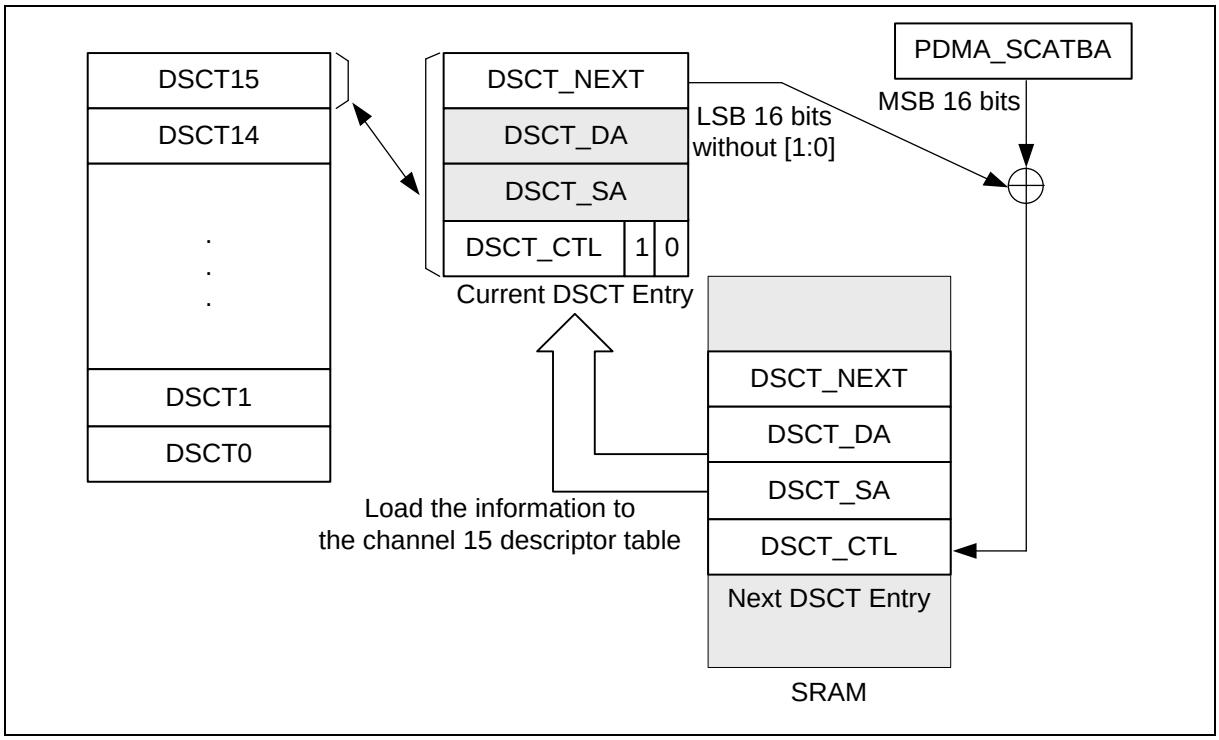


Figure 6.8-4 Descriptor Table Link List Structure

The above link list table operation is DSCT state in Scatter-gather Mode as shown in Figure 6.9-5. When loading the information is finished, it will go to transfer state and start transfer by this information automatically. However, if the next PDMA information is also set to Scatter-gather mode, the hardware will catch the next PDMA information block when the current task is finished. The Scatter-gather mode switches to basic mode when doing the next task. Then, the basic mode switches to Idle state when the last task is finished.

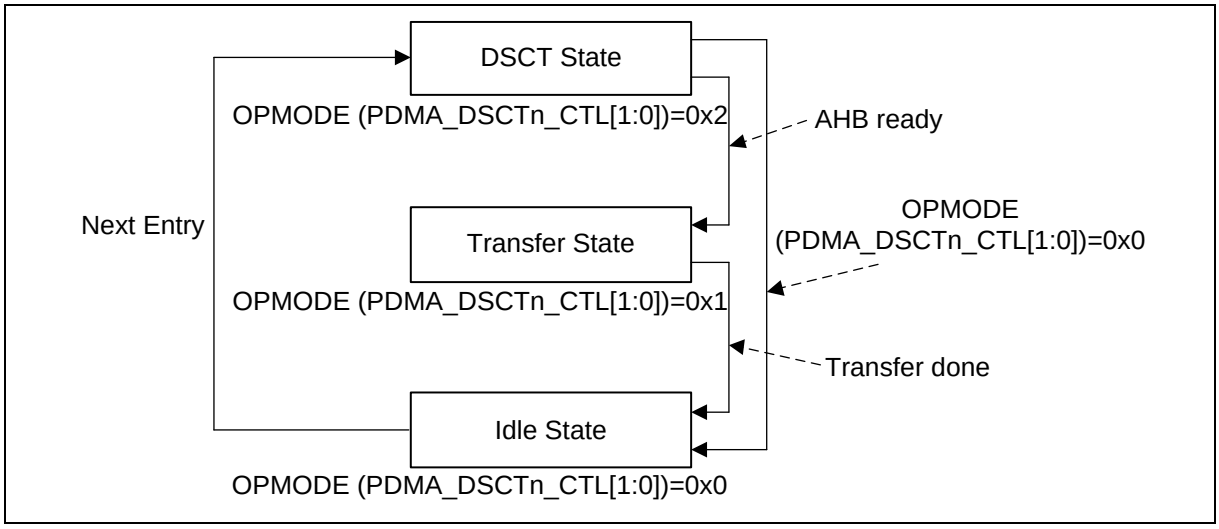


Figure 6.8-5 Scatter-gather Mode Finite State Machine

6.8.5.3 Transfer Type

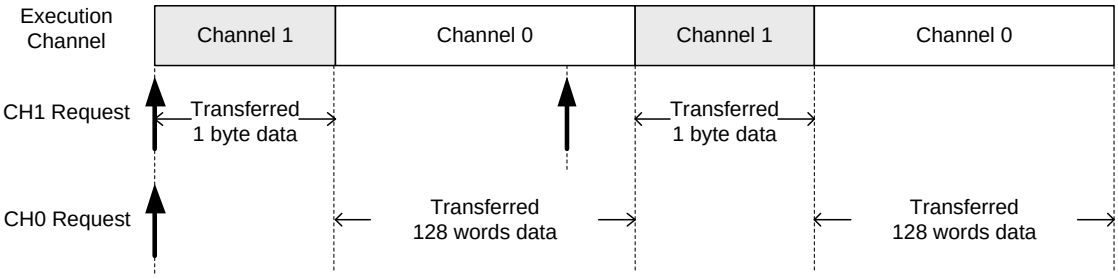
PDMA supports two transfer types: single transfer type and burst transfer type, configured by setting TXTYPE (PDMA_DSCTn_CTL[2]).

When PDMA is operated in single transfer type, each transfer needs one request signal. After data is transferred completely, transfer count TXCNT (PDMA_DSCTn_CTL[31:16]) will be decreased 1. Transfer will be finished after the TXCNT decreases to 0. In this mode, the BURSIZE (PDMA_DSCTn_CTL[6:4]) is not useful to control the transfer size. The BURSIZE will be fixed as one.

When PDMA is operated in burst transfer type, only one request signal is needed to transfer TXCNT amount of data. After PDMA transfer BURSIZE amount of data, TXCNT will be automatically reduced the number of BURSIZE. Transfer will be done after TXCNT decreases to 0. The burst transfer type can only be used for memory-to-memory transfers. User must use single transfer type for memory-to-peripheral and peripheral-to-memory transfers. Please note that, PDMA transfer data should finish before chip enter idle mode or power done mode to prevent access wrong data.

Figure 6.9-6 shows an example about single and burst transfer type in basic mode. In this example, channel 1 is single transfer type and TXCNT = 127. Channel 0 is burst transfer type, BURSIZE = 128 and TXCNT = 255. The operation sequence is described below:

1. Channel 0 and channel 1 get the trigger signal at the same time.
2. Channel 1 has higher priority than channel 0 by default; PDMA will load the channel 1 descriptor table first and executing. But channel 1 is single transfer type, and thus PDMA will only transfer one transfer data.
3. Then, PDMA turns to the channel 0 and loads channel 0's descriptor table. The channel 0 is burst transfer type and the burst size selected to 128. Therefore, PDMA will transfer 128 transfer data.
4. When channel 0 transfers 128 data, channel 1 gets another request signal, then after channel 0 finishes 128 transfer data, PDMA will turn to channel 1 and transfer next one data.
5. After channel 1 transfers data, PDMA switches to low priority channel 0 to continuous next 128 data transfer. If no channel 1 request receives, PDMA will start next channel 0, 128 data transfer.
6. PDMA will complete transfer when channel 0 finishes data transfer 256 times, and channel 1 finishes transferring 128 times.



	CH1	CH0
TXCNT (PDMA_DSCTn_CTL[31:16])	127	255
TXWIDTH (PDMA_DSCTn_CTL[13:12])	0 (1 byte)	2 (1 word)
BURSIZE (PDMA_DSCTn_CTL[6:4])	Non-useful	0 (128 transfers)
TXTYPE (PDMA_DSCTn_CTL[2])	1 (Single Transfer)	0 (Burst Transfer)

Figure 6.8-6 Example of Single Transfer Type and Burst Transfer Type in Basic Mode

6.8.5.4 Channel Time-out

PDMA support 2 channels time-out function. When the transfer channel is enabled and selected to the peripheral, corresponding channel time-out TOUTENn (PDMA_TOUTEN [n]) is enabled, then channel's corresponding time-out counter will start count up from 0 while the channel has received trigger signal from the peripheral.

The time-out counter is based on output of HCLK prescaler, which is setting by corresponding channel's TOUTPSC. If time-out counter counts up from 0 to corresponding channel's TOCn, PDMA will generate interrupt signal when corresponding TOUTIENn (PDMA_TOUTIEN [n]) is enabled. When time-out occurred, corresponding channel's REQTOFn (PDMA_INTSTS [n+8], n=0~1) will be set to indicate channel time-out is happened.

Time-out counter will restart from 0 while counter count to TOCn, received trigger signal, time-out function is disabled or chip enters Power-down mode. The time-out counter will keep counting until time-out function is disabled.

Figure 6.8-7 shows an example about time-out counter operation. The operation sequence is described below:

1. The channel 0 time-out counter is not counting when time-out function is enabled by setting TOUTEN0(PDMA_TOUTEN[0]) bit to 1.
2. Time-out counter starts counting from 0 to the value of TOC0(PDMA_TOC0_1[15:0]) bits when receiving the first peripheral request.
3. Time-out counter is reset to 0 by received second peripheral request.
4. Channel 0 request time-out flag(REQTOF0(PDMA_INTSTS[8])) is set to high when time-out counter counts to 5. The counter will keep counting.
5. Time-out counter is reset to 0 when time-out function is disabled.

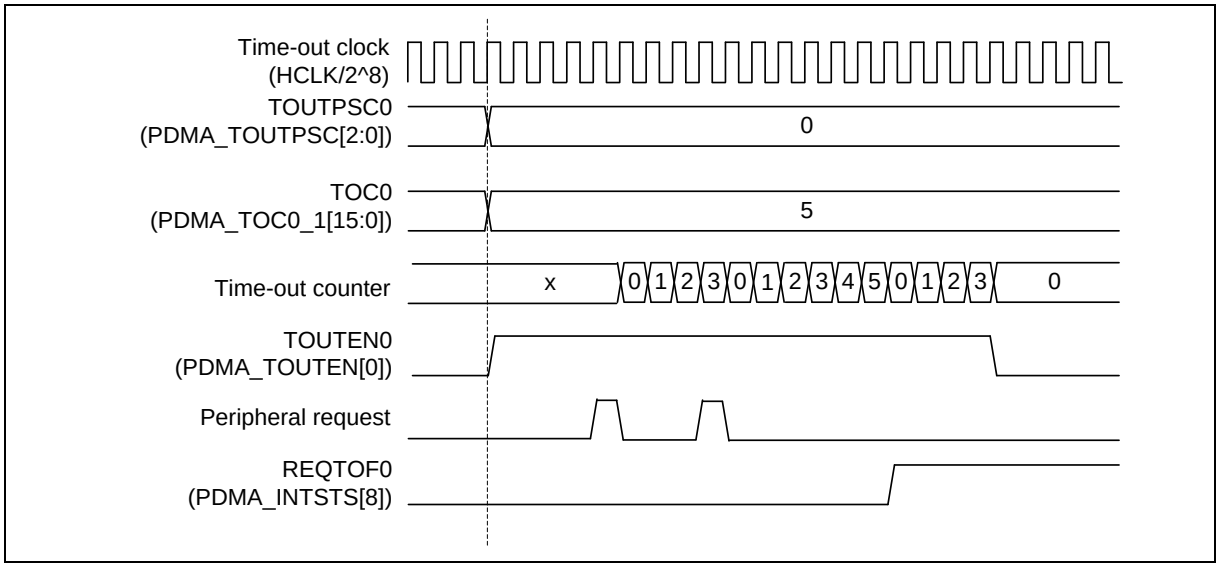


Figure 6.8-7 Example of PDMA Channel 0 Time-out Counter Operation

6.8.6 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMAx_BA = 0x4000_8000+(0x1_0000*x)** **x=0**				
PDMAx_DSCTn_CTL **n = 0,1..15**	PDMAx_BA+0x10*n	R/W	Descriptor Table Control Register of PDMA Channel n	0xXXXX_XXXX
PDMAx_DSCTn_SA **n = 0,1..15**	PDMAx_BA+0x0004+0x10*n	R/W	Source Address Register of PDMA Channel n	0xXXXX_XXXX
PDMAx_DSCTn_DA **n = 0,1..15**	PDMAx_BA+0x0008+0x10*n	R/W	Destination Address Register of PDMA Channel n	0xXXXX_XXXX
PDMAx_DSCTn_NEXT **n = 0,1..15**	PDMAx_BA+0x000C+0x10*n	R/W	Next Scatter-gather Descriptor Table Offset Address of PDMA Channel n	0xXXXX_XXXX
PDMAx_CURSCATn **n = 0,1..15**	PDMAx_BA+0x0100+0x004*n	R	Current Scatter-gather Descriptor Table Address of PDMA Channel n	0xXXXX_XXXX
PDMAx_CHCTL	PDMAx_BA+0x400	R/W	PDMA Channel Control Register	0x0000_0000
PDMAx_PAUSE	PDMAx_BA+0x404	W	PDMA Transfer Pause Control Register	0x0000_0000
PDMAx_SWREQ	PDMAx_BA+0x408	W	PDMA Software Request Register	0x0000_0000
PDMAx_TRGSTS	PDMAx_BA+0x40C	R	PDMA Channel Request Status Register	0x0000_0000
PDMAx_PRISET	PDMAx_BA+0x410	R/W	PDMA Fixed Priority Setting Register	0x0000_0000
PDMAx_PRICLR	PDMAx_BA+0x414	W	PDMA Fixed Priority Clear Register	0x0000_0000
PDMAx_INTEN	PDMAx_BA+0x418	R/W	PDMA Interrupt Enable Register	0x0000_0000
PDMAx_INTSTS	PDMAx_BA+0x41C	R/W	PDMA Interrupt Status Register	0x0000_0000
PDMAx_ABTSTS	PDMAx_BA+0x420	R/W	PDMA Channel Read/Write Target Abort Flag Register	0x0000_0000
PDMAx_TDSTS	PDMAx_BA+0x424	R/W	PDMA Channel Transfer Done Flag Register	0x0000_0000
PDMAx_ALIGN	PDMAx_BA+0x428	R/W	PDMA Transfer Alignment Status Register	0x0000_0000
PDMAx_TACTSTS	PDMAx_BA+0x42C	R	PDMA Transfer Active Flag Register	0x0000_0000
PDMAx_TOUTPSC	PDMAx_BA+0x430	R/W	PDMA Time-out Prescaler Register(CH0 to CH1)	0x0000_0000
PDMAx_TOUTEN	PDMAx_BA+0x434	R/W	PDMA Time-out Enable Register	0x0000_0000
PDMAx_TOUTIEN	PDMAx_BA+0x438	R/W	PDMA Time-out Interrupt Enable Register	0x0000_0000
PDMAx_SCATBA	PDMAx_BA+0x43C	R/W	PDMA Scatter-gather Descriptor Table Base Address Register	0x2000_0000
PDMAx_TOC	PDMAx_BA+0x440	R/W	PDMA Time-out Counter Ch0 and Ch1 Register	0x0000_0000
PDMAx_CHRST	PDMAx_BA+0x460	R/W	PDMA Channel Reset Register	0x0000_0000
PDMAx_REQSEL0_3	PDMAx_BA+0x480	R/W	PDMA Request Source Select Register 0	0x0000_0000
PDMAx_REQSEL4_7	PDMAx_BA+0x484	R/W	PDMA Request Source Select Register 1	0x0000_0000

PDMAx_REQSEL8_11	PDMAx_BA+0x488	R/W	PDMA Request Source Select Register 2	0x0000_0000
PDMAx_REQSEL12_15	PDMAx_BA+0x48C	R/W	PDMA Request Source Select Register 3	0x0000_0000

6.8.7 Register Description

Descriptor Table Control Register (PDMAx_DSCTn_CTL)

Register	Offset	R/W	Description	Reset Value
PDMAx_DSCTn_CTL	PDMAx_BA+0x10*n	R/W	Descriptor Table Control Register of PDMA Channel n	0xXXXX_XXXX

31	30	29	28	27	26	25	24
TXCNT							
23	22	21	20	19	18	17	16
TXCNT							
15	14	13	12	11	10	9	8
Reserved		TXWIDTH		DAINC		SAINC	
7	6	5	4	3	2	1	0
TBINTDIS	BURSIZE			Reserved	TXTYPE	OPMODE	

Bits	Description	
[31:16]	**TXCNT**	**Transfer Count** The TXCNT represents the required number of PDMA transfer, the real transfer count is (TXCNT + 1); The maximum transfer count is 65536, every transfer may be byte, half-word or word that is dependent on TXWIDTH field. **Note:** When PDMA finishes each transfer data, this field will be decreased immediately.
[15:14]	**Reserved**	Reserved.
[13:12]	**TXWIDTH**	**Transfer Width Selection** This field is used for transfer width. 00 = One byte (8 bit) is transferred for every operation. 01= One half-word (16 bit) is transferred for every operation. 10 = One word (32-bit) is transferred for every operation. 11 = Reserved. **Note:** PDMA transfer source address (PDMA_DSCTn_SA) and PDMA transfer destination address (PDMA_DSCTn_DA) should be alignment under the TXWIDTH selection
[11:10]	**DAINC**	**Destination Address Increment** This field is used to set the destination address increment size. 11 = No increment (fixed address). Others = Increment and size is depended on TXWIDTH selection. **Note:** The fixed address function does not support in memory to memory transfer type.
[9:8]	**SAINC**	**Source Address Increment** This field is used to set the source address increment size. 11 = No increment (fixed address). Others = Increment and size is depended on TXWIDTH selection. **Note:** The fixed address function does not support in memory to memory transfer type.
[7]	**TBINTDIS**	**Table Interrupt Disable Bit** This field can be used to decide whether to enable table interrupt or not. If the TBINTDIS bit is 1, it will not

Bits	Description	
		generates TDIFn (PDMA_TDSTS[15:0]) when PDMA controller finishes transfer task. 0 = Table interrupt Enabled. 1 = Table interrupt Disabled. **Note 1:** This function only for Scatter-gather mode. **Note 2:** This bit must be set to 0 in basic mode.
[6:4]	**BURSIZE**	**Burst Size** This field is used for peripheral to determine the burst size or used for determine the re-arbitration size. 000 = 128 Transfers. 001 = 64 Transfers. 010 = 32 Transfers. 011 = 16 Transfers. 100 = 8 Transfers. 101 = 4 Transfers. 110 = 2 Transfers. 111 = 1 Transfers. **Note:** This field is only useful in burst transfer type.
[3]	**Reserved**	Reserved.
[2]	**TXTYPE**	**Transfer Type** 0 = Burst transfer type. 1 = Single transfer type.
[1:0]	**OPMODE**	**PDMA Operation Mode Selection** 00 = Idle state: Channel is stopped or this table is complete, when PDMA finish channel table task, OPMODE will be cleared to idle state automatically. 01 = Basic mode: The descriptor table only has one task. When this task is finished, PDMA_INTSTS[1] will be asserted. 10 = Scatter-gather mode: When operating in this mode, user must give the next descriptor table address in PDMA_DSCTn_NEXT register; PDMA controller will ignore this task, then load the next task to execute. 11 = Reserved. **Note:** Before filling new transfer task in the Descriptor Table, user must check PDMA_INTSTS[1] to make sure the current task is complete.

Start Source Address Register (PDMAx_DSCTn_SA)

Register	Offset	R/W	Description	Reset Value
PDMAx_DSCTn_SA	PDMAx_BA+0x0004+0x10*n	R/W	Source Address Register of PDMA Channel n	0xXXXX_XXXX

31	30	29	28	27	26	25	24
SA							
23	22	21	20	19	18	17	16
SA							
15	14	13	12	11	10	9	8
SA							
7	6	5	4	3	2	1	0
SA							

Bits	Description	
[31:0]	SA	**PDMA Transfer Source Address** This field indicates a 32-bit source address of PDMA controller.

Destination Address Register (PDMAx_DSCTn_DA)

Register	Offset	R/W	Description	Reset Value
PDMAx_DSCTn_DA	PDMAx_BA+0x0008+0x10*n	R/W	Destination Address Register of PDMA Channel n	0xXXXX_XXXX

31	30	29	28	27	26	25	24
DA							
23	22	21	20	19	18	17	16
DA							
15	14	13	12	11	10	9	8
DA							
7	6	5	4	3	2	1	0
DA							

Bits	Description	
[31:0]	DA	**PDMA Transfer Destination Address** This field indicates a 32-bit destination address of PDMA controller.

Next Scatter-gather Descriptor Table Offset Address (PDMAx_DSCTn_NEXT)

Register	Offset	R/W	Description	Reset Value
PDMAx_DSCTn_NEXT	PDMAx_BA+0x000C+0x10*n	R/W	Next Scatter-gather Descriptor Table Offset Address of PDMA Channel n	0xXXXX_XXXX

31	30	29	28	27	26	25	24
EXENEXT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
EXENEXT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NEXT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NEXT							

Bits	Description	
[31:16]	**EXENEXT**	**PDMA Execution Next Descriptor Table Offset** This field indicates the offset of next descriptor table address of current execution descriptor table in system memory. **Note:** Write operation is useless in this field.
[15:0]	**NEXT**	**PDMA Next Descriptor Table Offset** This field indicates the offset of the next descriptor table address in system memory. **Write Operation:** If the system memory based address is 0x2000_0000 (PDMA_SCATBA), and the next descriptor table is start from 0x2000_0100, then this field must fill in 0x0100. **Read Operation:** When operating in Scatter-gather mode, the last two bits NEXT[1:0] will become reserved, and indicate the first next address of system memory. **Note 1:** The descriptor table address must be word boundary. **Note 2:** Before filling transfer task in the descriptor table, user must check if the descriptor table is complete.

Current Scatter-gather Descriptor Table Address (PDMAx_CURSCATn)

Register	Offset	R/W	Description	Reset Value
PDMAx_CURSCATn	PDMAx_BA+0x0100+0x004*n	R	Current Scatter-gather Descriptor Table Address of PDMA Channel n	0xXXXX_XXXX

31	30	29	28	27	26	25	24
CURADDR							
23	22	21	20	19	18	17	16
CURADDR							
15	14	13	12	11	10	9	8
CURADDR							
7	6	5	4	3	2	1	0
CURADDR							

Bits	Description	
[31:0]	CURADDR	**PDMA Current Description Address (Read Only)** This field indicates a 32-bit current external description address of PDMA controller. **Note:** This field is read only and used for Scatter-gather mode only to indicate the current external description address.

Channel Control Register (PDMAx_CHCTL)

Register	Offset	R/W	Description	Reset Value
PDMAx_CHCTL	PDMAx_BA+0x400	R/W	PDMA Channel Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CHEN15	**CHEN14**	**CHEN13**	**CHEN12**	**CHEN11**	**CHEN10**	**CHEN9**	**CHEN8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHEN7	**CHEN6**	**CHEN5**	**CHEN4**	**CHEN3**	**CHEN2**	**CHEN1**	**CHEN0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**CHENn**	**PDMA Channel Enable Bits** Set this bit to 1 to enable PDMAn operation. Channel cannot be active if it is not set as enabled. 0 = PDMA channel [n] Disabled. 1 = PDMA channel [n] Enabled. **Note:** Setting the corresponding bit of PDMA_PAUSE or PDMA_CHRST register will also clear this bit.

PDMA Transfer Pause Control Register (PDMAx_PAUSE)

Register	Offset	R/W	Description	Reset Value
PDMAx_PAUSE	PDMAx_BA+0x404	W	PDMA Transfer Pause Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PAUSE15	**PAUSE14**	**PAUSE13**	**PAUSE12**	**PAUSE11**	**PAUSE10**	**PAUSE9**	**PAUSE8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAUSE7	**PAUSE6**	**PAUSE5**	**PAUSE4**	**PAUSE3**	**PAUSE2**	**PAUSE1**	**PAUSE0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**PAUSEn**	**PDMA Channel n Transfer Pause Control (Write Only)** User can set PAUSEn bit field to pause PDMA transfer. When user sets PAUSEn bit, PDMA will pause the on-going transfer, then clear channel enable bit CHENn (PDMAx_CHCTL [n], n=0,1.. 15) and clear request active flag (PDMAx_TRGSTS[n:0], n=0,1.. 15). If the paused channel is re-enabled again, the remaining transfers will be processed. 0 = No effect. 1 = Pause PDMA channel n transfer. Note:Only for memory to memory transfer.

PDMA Software Request Register (PDMAx_SWREQ)

Register	Offset	R/W	Description	Reset Value
PDMAx_SWREQ	PDMAx_BA+0x408	W	PDMA Software Request Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SWREQ15	**SWREQ14**	**SWREQ13**	**SWREQ12**	**SWREQ11**	**SWREQ10**	**SWREQ9**	**SWREQ8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SWREQ7	**SWREQ6**	**SWREQ5**	**SWREQ4**	**SWREQ3**	**SWREQ2**	**SWREQ1**	**SWREQ0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**SWREQn**	**PDMA Software Request (Write Only)** Set this bit to 1 to generate a software request to PDMA [n]. 0 = No effect. 1 = Generate a software request. **Note 1:** User can read PDMAx_TRGSTS register to know which channel is on active. Active flag may be triggered by software request or peripheral request. **Note 2:** If user does not enable corresponding PDMA channel, the software request will be ignored.

PDMA Channel Request Status Register (PDMAx_TRGSTS)

Register	Offset	R/W	Description	Reset Value
PDMAx_TRGSTS	PDMAx_BA+0x40C	R	PDMA Channel Request Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
REQSTS15	**REQSTS14**	**REQSTS13**	**REQSTS12**	**REQSTS11**	**REQSTS10**	**REQSTS9**	**REQSTS8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REQSTS7	**REQSTS6**	**REQSTS5**	**REQSTS4**	**REQSTS3**	**REQSTS2**	**REQSTS1**	**REQSTS0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**REQSTSn**	**PDMA Channel Request Status (Read Only)** This flag indicates whether channel[n] have a request or not, no matter request from software or peripheral. When PDMA controller finishes channel transfer, this bit will be cleared automatically. 0 = PDMA Channel n has no request. 1 = PDMA Channel n has a request. **Note:** If user pauses or resets each PDMA transfer by setting PDMA_PAUSE or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing the current transfer.

PDMA Fixed Priority Setting Register (PDMAx_PRISET)

Register	Offset	R/W	Description	Reset Value
PDMAx_PRISET	PDMAx_BA+0x410	R/W	PDMA Fixed Priority Setting Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FPRISET15	**FPRISET14**	**FPRISET13**	**FPRISET12**	**FPRISET11**	**FPRISET10**	**FPRISET9**	**FPRISET8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FPRISET7	**FPRISET6**	**FPRISET5**	**FPRISET4**	**FPRISET3**	**FPRISET2**	**FPRISET1**	**FPRISET0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**FPRISETn**	**PDMA Fixed Priority Setting** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel [n] to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel is round-robin priority. 1 = Corresponding PDMA channel is fixed priority. **Note:** This field is only set to fixed priority. To clear fixed priority, use PDMA_PRICLR register.

PDMA Fix Priority Clear Register (PDMAx_PRICLR)

Register	Offset	R/W	Description	Reset Value
PDMAx_PRICLR	PDMAx_BA+0x414	W	PDMA Fixed Priority Clear Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FPRICLR15	**FPRICLR14**	**FPRICLR13**	**FPRICLR12**	**FPRICLR11**	**FPRICLR10**	**FPRICLR9**	**FPRICLR8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FPRICLR7	**FPRICLR6**	**FPRICLR5**	**FPRICLR4**	**FPRICLR3**	**FPRICLR2**	**FPRICLR1**	**FPRICLR0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**FPRICLRn**	**PDMA Fixed Priority Clear Bits (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel [n] fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.

PDMA Interrupt Enable Register (PDMAx_INTEN)

Register	Offset	R/W	Description	Reset Value
PDMAx_INTEN	PDMAx_BA+0x418	R/W	PDMA Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
INTEN15	**INTEN14**	**INTEN13**	**INTEN12**	**INTEN11**	**INTEN10**	**INTEN9**	**INTEN8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTEN7	**INTEN6**	**INTEN5**	**INTEN4**	**INTEN3**	**INTEN2**	**INTEN1**	**INTEN0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**INTENn**	**PDMA Interrupt Enable Bits** This field is used to enable PDMA channel[n] interrupt. 0 = PDMA channel n interrupt Disabled. 1 = PDMA channel n interrupt Enabled. **Note:** The interrupt flag is time-out, abort, transfer done and align.

PDMA Interrupt Status Register (PDMAx_INTSTS)

Register	Offset	R/W	Description	Reset Value
PDMAx_INTSTS	PDMAx_BA+0x41C	R/W	PDMA Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**REQTOF1**	**REQTOF0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ALIGNF**	**TDIF**	**ABTIF**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**REQTOF1**	**Request Time-out Flag for Channel 1** This flag indicates that PDMA controller has waited peripheral request for a period defined by TOC1(PDMA_TOC0_1[31:16]). 0 = No request time-out. 1 = Peripheral request time-out. **Note 1:** Please disable time-out function before clearing this bit. **Note 2:** User can write 1 to clear this bit.
[8]	**REQTOF0**	**Request Time-out Flag for Channel 0** This flag indicates that PDMA controller has waited peripheral request for a period defined by TOC0(PDMA_TOC0_1[15:0]. 0 = No request time-out. 1 = Peripheral request time-out. **Note 1:** Please disable time-out function before clearing this bit. **Note 2:** User can write 1 to clear this bit.
[7:3]	**Reserved**	Reserved.
[2]	**ALIGNF**	**Transfer Alignment Interrupt Flag (Read Only)** 0 = PDMA channel source address and destination address both follow transfer width setting. 1 = PDMA channel source address or destination address is not follow transfer width setting.
[1]	**TDIF**	**Transfer Done Interrupt Flag (Read Only)** This bit indicates that PDMA controller has finished transmission; User can read PDMA_TDSTS register to indicate which channel finished transfer. 0 = Not finished yet. 1 = PDMA channel has finished transmission.
[0]	**ABTIF**	**PDMA Read/Write Target Abort Interrupt Flag (Read Only)** This bit indicates that PDMA has target abort error; Software can read PDMA_ABTSTS register to find which channel has target abort error. 0 = No AHB bus ERROR response received.

Bits	Description	
		1 = AHB bus ERROR response received.

PDMA Channel Read/Write Target Abort Flag Register (PDMAx_ABTSTS)

Register	Offset	R/W	Description	Reset Value
PDMAx_ABTSTS	PDMAx_BA+0x420	R/W	PDMA Channel Read/Write Target Abort Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ABTIF15	**ABTIF14**	**ABTIF13**	**ABTIF12**	**ABTIF11**	**ABTIF10**	**ABTIF9**	**ABTIF8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ABTIF7	**ABTIF6**	**ABTIF5**	**ABTIF4**	**ABTIF3**	**ABTIF2**	**ABTIF1**	**ABTIF0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**ABTIFn**	**PDMA Read/Write Target Abort Interrupt Status Flag** This bit indicates which PDMA controller has target abort error. 0 = No AHB bus ERROR response received when channel n transfer. 1 = AHB bus ERROR response received when channel n transfer. **Note 1:** If channel n target abort, REQSRCn should set 0 to disable peripheral request. **Note 2:** User can write 1 to clear this bit.

PDMA Channel Transfer Done Flag Register (PDMAx_TDSTS)

Register	Offset	R/W	Description	Reset Value
PDMAx_TDSTS	PDMAx_BA+0x424	R/W	PDMA Channel Transfer Done Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TDIF15	**TDIF14**	**TDIF13**	**TDIF12**	**TDIF11**	**TDIF10**	**TDIF9**	**TDIF8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TDIF7	**TDIF6**	**TDIF5**	**TDIF4**	**TDIF3**	**TDIF2**	**TDIF1**	**TDIF0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**TDIFn**	**Transfer Done Flag** This bit indicates whether PDMA controller channel transfer has been finished or not. 0 = PDMA channel transfer has not finished. 1 = PDMA channel has finished transmission. **Note:** User can write 1 to clear these bits.

PDMA Transfer Alignment Status Register (PDMAx_ALIGN)

Register	Offset	R/W	Description	Reset Value
PDMAx_ALIGN	PDMAx_BA+0x428	R/W	PDMA Transfer Alignment Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ALIGN15	**ALIGN14**	**ALIGN13**	**ALIGN12**	**ALIGN11**	**ALIGN10**	**ALIGN9**	**ALIGN8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ALIGN7	**ALIGN6**	**ALIGN5**	**ALIGN4**	**ALIGN3**	**ALIGN2**	**ALIGN1**	**ALIGN0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**ALIGNn**	**Transfer Alignment Flag** This bit indicates whether source and destination address both follow transfer width setting. 0 = PDMA channel source address and destination address both follow transfer width setting. 1 = PDMA channel source address or destination address is not follow transfer width setting. **Note:** User can write 1 to clear these bits.

PDMA Transfer Active Flag Register (PDMAx_TACTSTS)

Register	Offset	R/W	Description	Reset Value
PDMAx_TACTSTS	PDMAx_BA+0x42C	R	PDMA Transfer Active Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXACTF15	**TXACTF14**	**TXACTF13**	**TXACTF12**	**TXACTF11**	**TXACTF10**	**TXACTF9**	**TXACTF8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXACTF7	**TXACTF6**	**TXACTF5**	**TXACTF4**	**TXACTF3**	**TXACTF2**	**TXACTF1**	**TXACTF0**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**TXACTFn**	**Transfer on Active Flag (Read Only)** This bit indicates which PDMA channel is in active. 0 = PDMA channel is finished. 1 = PDMA channel is active.

1PDMA Time-out Prescaler Register (PDMAx_TOUTPSC)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOUTPSC	PDMAx_BA+0x430	R/W	PDMA Time-out Prescaler Register(CH0 to CH1)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**TOUTPSC1**			**Reserved**	**TOUTPSC0**		

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:4]	**TOUTPSC1**	**PDMA Channel 1 Time-out Clock Source Prescaler Bits** 000 = PDMA channel 1 time-out clock source is HCLK/2^8. 001 = PDMA channel 1 time-out clock source is HCLK/2^9. 010 = PDMA channel 1 time-out clock source is HCLK/2^{10}. 011 = PDMA channel 1 time-out clock source is HCLK/2^{11}. 100 = PDMA channel 1 time-out clock source is HCLK/2^{12}. 101 = PDMA channel 1 time-out clock source is HCLK/2^{13}. 110 = PDMA channel 1 time-out clock source is HCLK/2^{14}. 111 = PDMA channel 1 time-out clock source is HCLK/2^{15}.
[3]	**Reserved**	Reserved.
[2:0]	**TOUTPSC0**	**PDMA Channel 0 Time-out Clock Source Prescaler Bits** 000 = PDMA channel 0 time-out clock source is HCLK/2^8. 001 = PDMA channel 0 time-out clock source is HCLK/2^9. 010 = PDMA channel 0 time-out clock source is HCLK/2^{10}. 011 = PDMA channel 0 time-out clock source is HCLK/2^{11}. 100 = PDMA channel 0 time-out clock source is HCLK/2^{12}. 101 = PDMA channel 0 time-out clock source is HCLK/2^{13}. 110 = PDMA channel 0 time-out clock source is HCLK/2^{14}. 111 = PDMA channel 0 time-out clock source is HCLK/2^{15}.

PDMA Time-out Enable Register (PDMAx_TOUTEN)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOUTEN	PDMAx_BA+0x434	R/W	PDMA Time-out Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**TOUTEN1**	**TOUTEN0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[n] n=0,1	**TOUTENn**	**PDMA Time-out Enable Bits** 0 = PDMA Channel n time-out function Disabled. 1 = PDMA Channel n time-out function Enabled.

PDMA Time-out Interrupt Enable Register (PDMAx_TOUTIEN)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOUTIEN	PDMAx_BA+0x438	R/W	PDMA Time-out Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**TOUTIEN1**	**TOUTIEN0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[n] n=0,1	**TOUTIENn**	**PDMA Time-out Interrupt Enable Bits** 0 = PDMA Channel n time-out interrupt Disabled. 1 = PDMA Channel n time-out interrupt Enabled.

PDMA Scatter-gather Descriptor Table Base Address Register (PDMAx_SCATBA)

Register	Offset	R/W	Description	Reset Value
PDMAx_SCATBA	PDMAx_BA+0x43C	R/W	PDMA Scatter-gather Descriptor Table Base Address Register	0x2000_0000

31	30	29	28	27	26	25	24
SCATBA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SCATBA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:16]	**SCATBA**	**PDMA Scatter-gather Descriptor Table Address** In Scatter-gather mode, this is the base address for calculating the next link - list address. The next link address equation is Next Link Address = PDMA_SCATBA + PDMA_DSCTn_NEXT. **Note:** Only useful in Scatter-gather mode.
[15:0]	**Reserved**	Reserved.

PDMA Time-out Period Counter Register 0 (PDMAx_TOC)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOC	PDMAx_BA+0x440	R/W	PDMA Time-out Counter Ch0 and Ch1 Register	0x0000_0000

31	30	29	28	27	26	25	24
TOC1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TOC1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TOC0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TOC0							

Bits	Description	
[31:16]	**TOC1**	**Time-out Counter for Channel 1** This controls the period of time-out function for channel 1. The calculation unit is based on TOUTPSC1 (PDMA_TOUTPSC[6:4]) clock. The example of time-out period can refer TOC0 bit description.
[15:0]	**TOC0**	**Time-out Counter for Channel 0** This controls the period of time-out function for channel 0. The calculation unit is based on TOUTPSC0 (PDMA_TOUTPSC[2:0]) clock. Time-out period = (Period of time-out clock) * (16-bit TOCn),n=0,1.

PDMA Channel Reset Register (PDMAx_CHRST)

Register	Offset	R/W	Description	Reset Value
PDMAx_CHRST	PDMAx_BA+0x460	R/W	PDMA Channel Reset Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CH15RST	**CH14RST**	**CH13RST**	**CH12RST**	**CH11RST**	**CH10RST**	**CH9RST**	**CH8RST**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CH7RST	**CH6RST**	**CH5RST**	**CH4RST**	**CH3RST**	**CH2RST**	**CH1RST**	**CH0RST**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**CHnRST**	**PDMA Channel n Reset Control** User can set CHnRST bit field to reset PDMA transfer. When user sets CHnRST bit, PDMA will stop the on-going transfer, then clear channel enable bit CHENn (PDMAx_CHCTL[n], n=0,1.. 15) and clear request active flag (PDMAx_TRGSTS[n:0], n=0,1.. 15). User should re-initialize corresponding channel setting. 0 = Corresponding channel n is not reset. 1 = Corresponding channel n is reset. **Note 1:** After PDMA current request transfer is complete, this bit will be cleared automatically. **Note 2**: If channel trigger source come from peripheral, set the REQSRCn to 0 before set the channel reset.

PDMA Request Source Select Register 0 (PDMAx_REQSEL0_3)

Register	Offset	R/W	Description	Reset Value
PDMAx_REQSEL0_3	PDMAx_BA+0x480	R/W	PDMA Request Source Select Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC3						
23	22	21	20	19	18	17	16
Reserved	REQSRC2						
15	14	13	12	11	10	9	8
Reserved	REQSRC1						
7	6	5	4	3	2	1	0
Reserved	REQSRC0						

Bits	Description	
[31]	**Reserved**	Reserved.
[30:24]	**REQSRC3**	**Channel 3 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 3. User can configure the peripheral setting by REQSRC3. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23]	**Reserved**	Reserved.
[22:16]	**REQSRC2**	**Channel 2 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 2. User can configure the peripheral setting by REQSRC2. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15]	**Reserved**	Reserved.
[14:8]	**REQSRC1**	**Channel 1 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 1. User can configure the peripheral setting by REQSRC1. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7]	**Reserved**	Reserved.
[6:0]	**REQSRC0**	**Channel 0 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 0. User can configure the peripheral by setting REQSRC0. 0 = Disable PDMA peripheral request. 1 = Reserved. 2 = Channel connects to USB_TX. 3 = Channel connects to USB_RX. 4 = Channel connects to UART0_TX. 5 = Channel connects to UART0_RX.

Bits	Description	
		6 = Channel connects to UART1_TX. 7 = Channel connects to UART1_RX. 8 = Channel connects to UART2_TX. 9 = Channel connects to UART2_RX. 10=Channel connects to UART3_TX. 11 = Channel connects to UART3_RX. 12 = Channel connects to UART4_TX. 13 = Channel connects to UART4_RX. 14 = Channel connects to UART5_TX. 15 = Channel connects to UART5_RX. 16 = Channel connects to USCI0_TX. 17 = Channel connects to USCI0_RX. 18 = Channel connects to USCI1_TX. 19 = Channel connects to USCI1_RX . 20 = Channel connects to QSPI0_TX. 21 = Channel connects to QSPI0_RX. 22 = Channel connects to SPI0_TX. 23 = Channel connects to SPI0_RX. 24 = Channel connects to SPI1_TX. 25 = Channel connects to SPI1_RX. 26 = Channel connects to SPI2_TX. 27 = Channel connects to SPI2_RX. 28 = Channel connects to SPI3_TX. 29 = Channel connects to SPI3_RX. 30 = Channel connects to ACMP0. 31 = Channel connects to ACMP1. 32 = Channel connects to EPWM0_P1_RX. 33 = Channel connects to EPWM0_P2_RX. 34 = Channel connects to EPWM0_P3_RX. 35 = Channel connects to EPWM1_P1_RX. 36 = Channel connects to EPWM1_P2_RX. 37 = Channel connects to EPWM1_P3_RX. 38 = Channel connects to I2C0_TX. 39 = Channel connects to I2C0_RX. 40 = Channel connects to I2C1_TX. 41 = Channel connects to I2C1_RX. 42 = Channel connects to I2C2_TX. 43 = Channel connects to I2C2_RX. 44 = Channel connects to I2C3_TX. 45 = Channel connects to I2C3_RX. 46 = Channel connects to TMR0. 47 = Channel connects to TMR1. 48 = Channel connects to TMR2. 49 = Channel connects to TMR3. 50 = Channel connects to DAC0_TX. 51 = Channel connects to DAC1_TX. 52 = Channel connects to EPWM0_CH0_TX. 53 = Channel connects to EPWM0_CH1_TX.

Bits	Description	
		54 = Channel connects to EPWM0_CH2_TX. 55 = Channel connects to EPWM0_CH3_TX. 56 = Channel connects to EPWM0_CH4_TX. 57 = Channel connects to EPWM0_CH5_TX. 58 = Channel connects to EPWM1_CH0_TX. 59 = Channel connects to EPWM1_CH1_TX. 60 = Channel connects to EPWM1_CH2_TX. 61 = Channel connects to EPWM1_CH3_TX. 62 = Channel connects to EPWM1_CH4_TX. 63 = Channel connects to EPWM1_CH5_TX. 64 = Channel connects to EADC0_RX. 65 = Channel connects to EADC1_RX. 66 = Channel connects to UART6_TX. 67 = Channel connects to UART6_RX. 68 = Channel connects to UART7_TX. 69 = Channel connects to UART7_RX. 70 = Channel connects to PWM0_P1_RX. 71 = Channel connects to PWM0_P2_RX. 72 = Channel connects to PWM0_P3_RX. 73 = Channel connects to PWM1_P1_RX. 74 = Channel connects to PWM1_P2_RX. 75 = Channel connects to PWM1_P3_RX. 76 = Channel connects to PWM0_CH0_TX. 77 = Channel connects to PWM0_CH2_TX. 78 = Channel connects to PWM0_CH4_TX. 79 = Channel connects to PWM1_CH0_TX. 80 = Channel connects to PWM1_CH2_TX. 81 = Channel connects to PWM1_CH4_TX. 82 = Channel connects to EINT0. 83 = Channel connects to EINT1. 84 = Channel connects to EINT2. 85 = Channel connects to EINT3. 86 = Channel connects to EINT4. 87 = Channel connects to EINT5. 88 = Channel connects to EINT6. 89 = Channel connects to EINT7. 90 = Channel connects to ACMP2. Others = Reserved. **Note 1:** A peripheral cannot be assigned to two channels at the same time. **Note 2:** This field is useless when transfer between memory and memory.

PDMA Request Source Select Register 1 (PDMAx_REQSEL4_7)

Register	Offset	R/W	Description	Reset Value
PDMAx_REQSEL4_7	PDMAx_BA+0x484	R/W	PDMA Request Source Select Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC7						
23	22	21	20	19	18	17	16
Reserved	REQSRC6						
15	14	13	12	11	10	9	8
Reserved	REQSRC5						
7	6	5	4	3	2	1	0
Reserved	REQSRC4						

Bits	Description	
[31]	**Reserved**	Reserved.
[30:24]	**REQSRC7**	**Channel 7 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 7. User can configure the peripheral setting by REQSRC7. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23]	**Reserved**	Reserved.
[22:16]	**REQSRC6**	**Channel 6 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 6. User can configure the peripheral setting by REQSRC6. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15]	**Reserved**	Reserved.
[14:8]	**REQSRC5**	**Channel 5 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 5. User can configure the peripheral setting by REQSRC5. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7]	**Reserved**	Reserved.
[6:0]	**REQSRC4**	**Channel 4 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 4. User can configure the peripheral setting by REQSRC4. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.

PDMA Request Source Select Register 2 (PDMAx_REQSEL8_11)

Register	Offset	R/W	Description	Reset Value
PDMAx_REQSEL8_11	PDMAx_BA+0x488	R/W	PDMA Request Source Select Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC11						
23	22	21	20	19	18	17	16
Reserved	REQSRC10						
15	14	13	12	11	10	9	8
Reserved	REQSRC9						
7	6	5	4	3	2	1	0
Reserved	REQSRC8						

Bits	Description	
[31]	**Reserved**	Reserved.
[30:24]	**REQSRC11**	**Channel 11 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 11. User can configure the peripheral setting by REQSRC11. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23]	**Reserved**	Reserved.
[22:16]	**REQSRC10**	**Channel 10 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 10. User can configure the peripheral setting by REQSRC10. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15]	**Reserved**	Reserved.
[14:8]	**REQSRC9**	**Channel 9 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 9. User can configure the peripheral setting by REQSRC9. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7]	**Reserved**	Reserved.
[6:0]	**REQSRC8**	**Channel 8 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 8. User can configure the peripheral setting by REQSRC8. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.

PDMA Request Source Select Register 3 (PDMAx_REQSEL12_15)

Register	Offset	R/W	Description	Reset Value
PDMAx_REQSEL12_15	PDMAx_BA+0x48C	R/W	PDMA Request Source Select Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**REQSRC15**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**REQSRC14**						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**REQSRC13**						
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**REQSRC12**						

Bits	Description	
[31]	**Reserved**	Reserved.
[30:24]	**REQSRC15**	**Channel 15 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 15. User can configure the peripheral setting by REQSRC15. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23]	**Reserved**	Reserved.
[22:16]	**REQSRC14**	**Channel 14 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 14. User can configure the peripheral setting by REQSRC14. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15]	**Reserved**	Reserved.
[14:8]	**REQSRC13**	**Channel 13 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 13. User can configure the peripheral setting by REQSRC13. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7]	**Reserved**	Reserved.
[6:0]	**REQSRC12**	**Channel 12 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 12. User can configure the peripheral setting by REQSRC12. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.

6.9 Low Power PDMA Controller (LPPDMA)

6.9.1 Overview

The low power peripheral direct memory access (LPPDMA) controller is used to provide high-speed data transfer. LPPDMA can transfer data from one address to another without CPU intervention. This has the benefit of reducing the workload of CPU and keeps CPU resources free for other applications. LPPDMA has a total of 4 channels and each channel can perform transfer between memory and peripherals or between memory and memory.

6.9.2 Features

- Supports 4 independently configurable channels
- Supports selectable 2 level of priority (fixed priority or round-robin priority)
- Supports transfer data width of 8, 16, and 32 bits
- Supports source and destination address increment size can be byte, half-word, word or no increment
- Supports software and LPUART, LPSPI, LPI^{2}C, LPTIMER, TTIMER, LPADC and ACMP request
- Supports Scatter-gather mode to perform sophisticated transfer through the use of the descriptor link list table
- Supports single and burst transfer type

6.9.3 Block Diagram

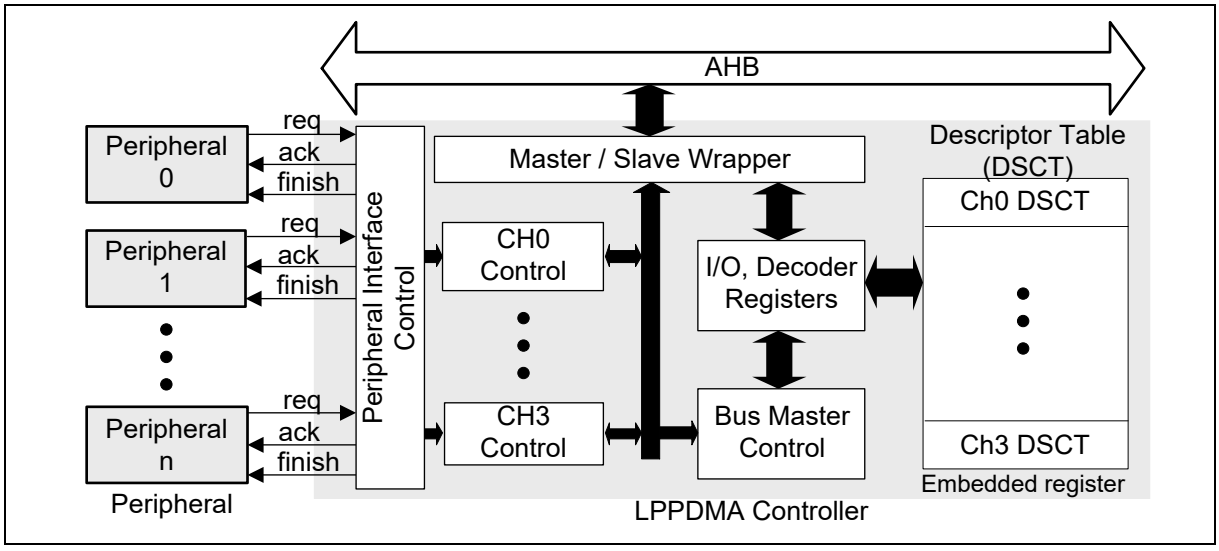


Figure 6.9-1 LPPDMA Controller Block Diagram

6.9.4 Basic Configuration

6.9.4.1 Basic Configuration of LPPDMA0

- Clock Source Configuration
  - Enable LPPDMA0 controller clock in LPPDMA0CKEN (LPSCC_CLKEN0[0]).
- Reset Configuration
  - Reset LPPDMA0 controller in LPPDMA0RST (LPSCC_IPRST0[2]).

6.9.5 Functional Description

The LPPDMA controller transfers data from one address to another without CPU intervention. LPPDMA supports 4 independent channels, and only one channel transfers data at a time. Therefore, it supports two levels of channel priority: fixed and round-robin, and transfers data in order from the highest to the lowest priority channel.

LPPDMA supports two operation modes, Basic and Scatter-gather. Basic mode is used to perform one descriptor table transfer. Scatter-gather mode has more entries for each LPPDMA channel, and thus LPPDMA supports sophisticated transfer through the entries. The descriptor table entry data structure contains many transfer information including the transfer source address, transfer destination address, transfer count, burst size, transfer type and operation mode. Figure 6.9-2 shows the diagram of descriptor table (DSCT) data structure.

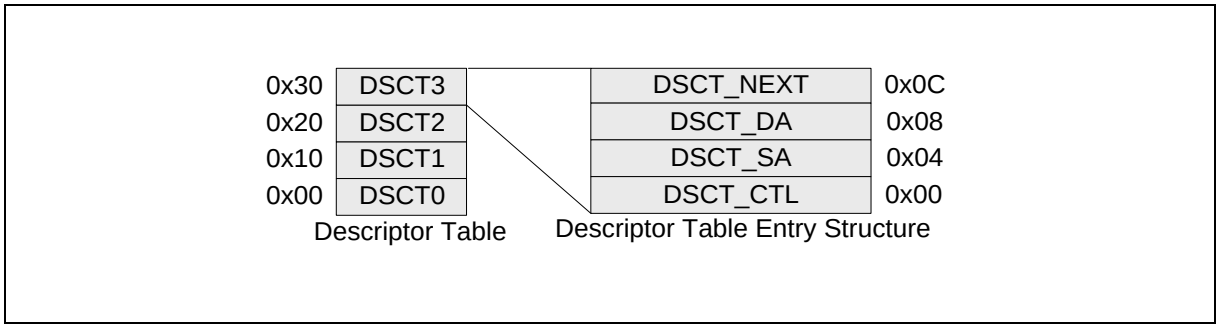


Figure 6.9-2 Descriptor Table Entry Structure

LPPDMA supports single and burst transfer type and the request source can be from software or peripheral request. It also supports data transfer between memory to memory by software request. A single transfer means that software or peripheral is ready to transfer one data (every data needs one request), and the burst transfer means that software or peripherals will transfer multiple data (multiple data only need one request).

6.9.5.1 Channel Priority

LPPDMA supports two level channel priorities including fixed and round-robin priority. The fixed priority channel has higher priority than round-robin priority channel. If multiple channels are set as fixed or round-robin priority, the higher channel will have higher priority. The priority order is listed in Table 6.9-1.

LPPDMA_PRISET	Channel Number	Priority Setting	Arbitration Priority In Descending Order
1	3	Channel3, Fixed Priority	Highest
1	2	Channel2, Fixed Priority	---
---	---	---	---
1	0	Channel0, Fixed Priority	---
0	3	Channel3, Round-Robin Priority	---
0	2	Channel2, Round-Robin Priority	---
---	---	---	---
0	0	Channel0, Round-Robin Priority	Lowest

Table 6.9-1 Channel Priority Table

6.9.5.2 LPPDMA Operation Mode

LPPDMA supports two operation modes: Basic mode and Scatter-gather modes.

Basic Mode

Basic mode is used to perform one descriptor table transfer. This mode can be used to transfer data between memory and memory, peripherals and memory or peripherals and peripherals. If LPPDMA transfers data between peripherals and peripherals, user should make sure the transferred data is ready before peripherals send a request. LPPDMA operation mode can be set from OPMODE (LPPDMA_DSCTn_CTL[1:0], n denotes LPPDMA channel), the default setting is in idle state and recommend user configure the descriptor table in idle state. If operation mode is not in idle state, user re-configure channel setting may make some operation error.

User must fill the transfer count TXCNT (LPPDMA_DSCTn_CTL[31:16]) register and select transfer width TXWIDTH (LPPDMA_DSCTn_CTL[13:12]), destination address increment size DAINC (LPPDMA_DSCTn_CTL[11:10]), source address increment size SAINC (LPPDMA_DSCTn_CTL[9:8]), burst size BURSIZE (LPPDMA_DSCTn_CTL[6:4]) and transfer type TXTYPE (LPPDMA_DSCTn_CTL[2]), then LPPDMA will perform transfer operation in transfer state after receiving request signal. Finishing this task will generate an interrupt to CPU if corresponding LPPDMA interrupt bit INTENn (LPPDMA_INTEN[3:0]) is enabled and the operation mode will be updated to idle state as shown in Figure 6.9-3. If software configures the operation mode to idle state, LPPDMA will not perform any transfer and then clear this operation request. Finishing this task will also generate an interrupt to CPU if corresponding LPPDMA interrupt bit is enabled.

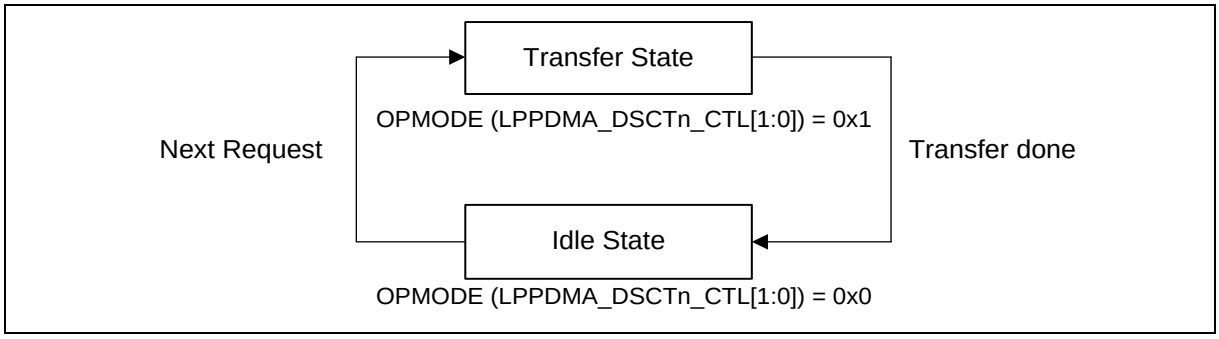


Figure 6.9-3 Basic Mode Finite State Machine

Scatter-gather Mode

Scatter-gather mode is a complex mode and can perform sophisticated transfer through the description link list table which is shown in Figure 6.9-4. It performs peripheral wrapper-around data transfer, and also performs multiple task data transfer between varied locations in system memory instead of a set of contiguous locations. Scatter-gather mode only needs a request to finish all table tasks, then OPMODE (LPPDMA_DSCTn_CTL[1:0]) will be cleared to idle state automatically.

In Scatter-gather mode, the table is just used for jumping to the next table entry. The first task will not perform any operation transfer. Finishing each task will generate an interrupt to CPU if corresponding LPPDMA interrupt bit is enabled and TBINTDIS (LPPDMA_DSCTn_CTL[7]) bit is “0” (when finishing task and TBINTDIS bit is “0”, corresponding TDIFn (LPPDMA_TDSTS[15:0]) flag will be asserted and if this bit is “1” TDIFn will not be active).

If channel 3 has been triggered, and the operation mode is in Scatter-gather mode (OPMODE (LPPDMA_DSCTn_CTL[1:0]) = 0x2), the hardware will load the real LPPDMA information task from the address generated by adding LPPDMA_DSCTn_NEXT (link address) and LPPDMA_SCATBA (base address) registers. For example, base address is 0x2800_0000 (only MSB 16 bits valid in LPPDMA_SCATBA), the current link address is 0x0000_0100 (only LSB 16-bits without last two bits [1:0] valid in LPPDMA_DSCTn_NEXT), and then the next DSCT entry start address is 0x2800_0100.

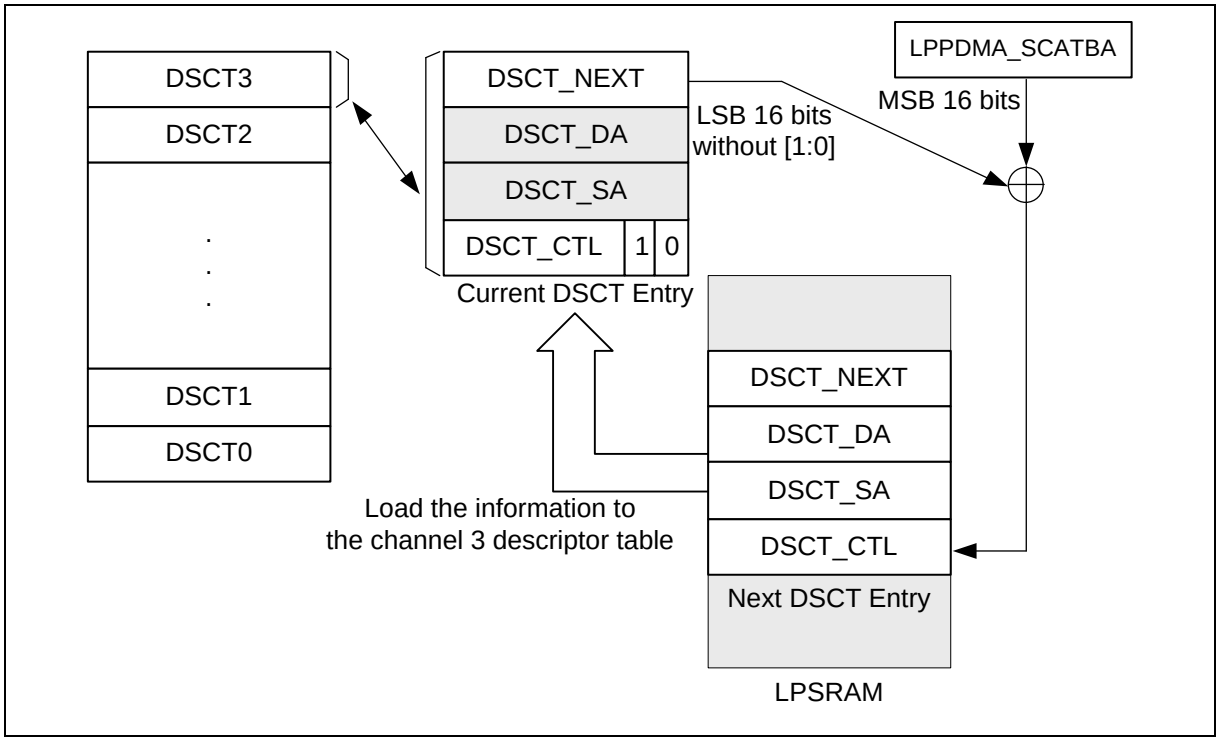


Figure 6.9-4 Descriptor Table Link List Structure

The above link list table operation is DSCT state in Scatter-gather Mode as shown in Figure 6.9-5. When loading the information is finished, it will go to transfer state and start transfer by this information automatically. However, if the next LPPDMA information is also set to Scatter-gather mode, the hardware will catch the next LPPDMA information block when the current task is finished. The Scatter-gather mode switches to basic mode when doing the next task. Then, the basic mode switches to Idle state when the last task is finished.

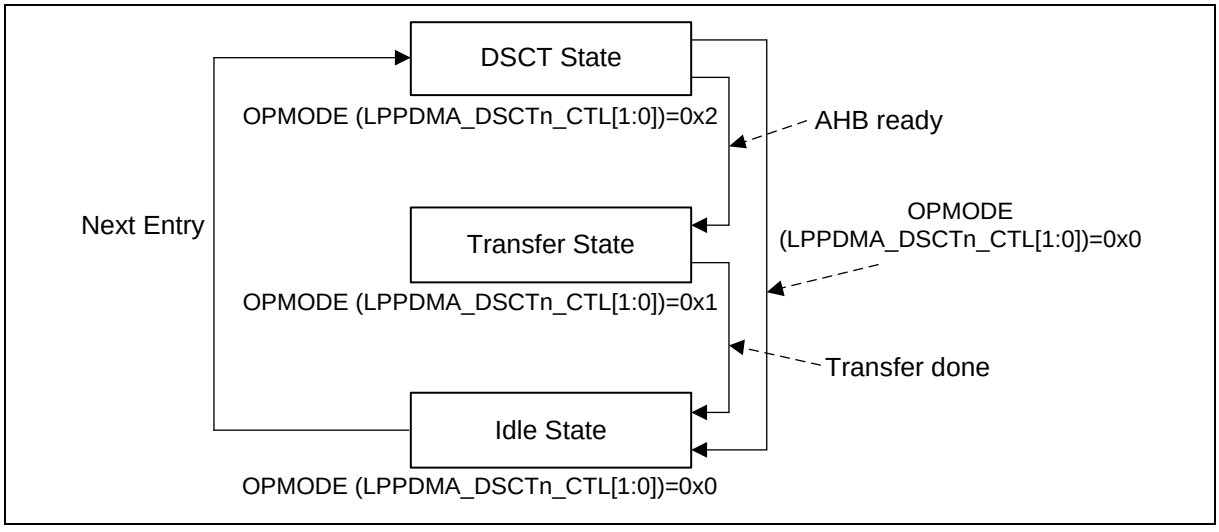


Figure 6.9-5 Scatter-gather Mode Finite State Machine

6.9.5.3 Transfer Type

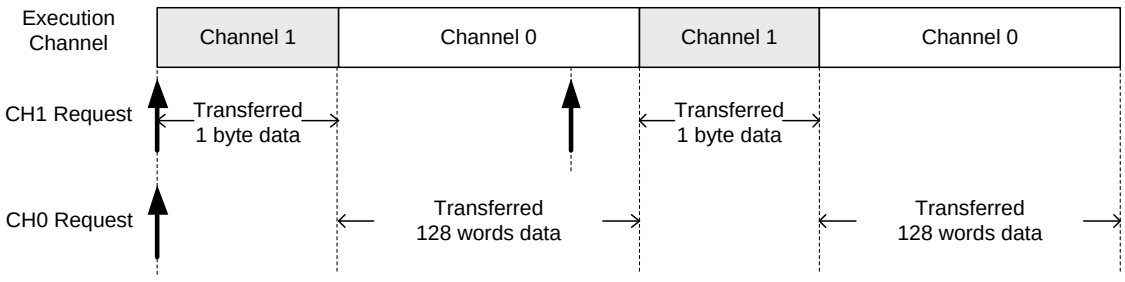
LPPDMA supports two transfer types: single transfer type and burst transfer type, configure by setting TXTYPE (LPPDMA_DSCTn_CTL[2]).

When LPPDMA is operated in single transfer type, each transfer needs one request signal. After data is transferred completely, transfer count TXCNT (LPPDMA_DSCTn_CTL[31:16]) will be decreased 1. Transfer will be finished after the TXCNT decreases to 0. In this mode, the BURSIZE (LPPDMA_DSCTn_CTL[6:4]) is not useful to control the transfer size. The BURSIZE will be fixed as one.

When LPPDMA is operated in burst transfer type, only one request signal is needed to transfer TXCNT amount of data. After LPPDMA transfer BURSIZE amount of data, TXCNT will be automatically reduced the number of BURSIZE. Transfer will be done after TXCNT decreases to 0. The burst transfer type can only be used for memory-to-memory transfers. User must use single transfer type for memory-to-peripheral and peripheral-to-memory transfers. Please note that, LPPDMA transfer data should be finished before chip enters idle mode or Power-done mode to prevent wrong data access.

Figure 6.9-6 shows an example about single and burst transfer type in basic mode. In this example, channel 1 uses single transfer type and TXCNT = 127. Channel 0 uses burst transfer type, BURSIZE = 128 and TXCNT = 255. The operation sequence is described below:

1. Channel 0 and channel 1 get the trigger signal at the same time.
2. Channel 1 has higher priority than channel 0 by default; LPPDMA will load the channel 1 descriptor table first and executing. But channel 1 is single transfer type, and thus LPPDMA will only transfer one transfer data.
3. Then, LPPDMA turns to the channel 0 and loads channel 0's descriptor table. The channel 0 is burst transfer type and the burst size selected to 128. Therefore, LPPDMA will transfer 128 transfer data.
4. When channel 0 transfers 128 data, channel 1 gets another request signal, then after channel 0 finishes 128 transfer data, LPPDMA will turn to channel 1 and transfer next one data.
5. After channel 1 transfers data, LPPDMA switches to low priority channel 0 to continuous next 128 data transfer. If no channel 1 request receives, LPPDMA will start next channel 0, 128 data transfer.
6. LPPDMA will complete transfer when channel 0 finishes data transfer 256 times, and channel 1 finishes transferring 128 times.



	CH1	CH0
TXCNT (LPPDMA_DSCTn_CTL[31:16])	127	255
TXWIDTH (LPDMA_DSCTn_CTL[13:12])	0 (1 byte)	2 (1 word)
BURSIZE (LPPDMA_DSCTn_CTL[6:4])	Non-useful	0 (128 transfers)
TXTYPE (LPPDMA_DSCTn_CTL[2])	1 (Single Transfer)	0 (Burst Transfer)

Figure 6.9-6 Example of Single Transfer Type and Burst Transfer Type in Basic Mode

6.9.5.4 Auto-operation Mode

LPPDMA is allowed to keep operating when chip enters NPD0/1/2/3/4 mode. User should refer the peripheral chapter and follow auto-operation setting flow to send the trigger signal to LPPDMA.

6.9.6 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
LPPDMA Base Address: **LPPDMAx_BA = 0x4003_9000** **x=0**				
LPPDMAx_DSCTn_CTL **n = 0,1..3**	LPPDMAx_BA+0x10*n	R/W	Descriptor Table Control Register of LPPDMA Channel n	0xXXXX_XXXX
LPPDMAx_DSCTn_SA **n = 0,1..3**	LPPDMAx_BA+0x0004+0x10*n	R/W	Source Address Register of LPPDMA Channel n	0xXXXX_XXXX
LPPDMAx_DSCTn_DA **n = 0,1..3**	LPPDMAx_BA+0x0008+0x10*n	R/W	Destination Address Register of LPPDMA Channel n	0xXXXX_XXXX
LPPDMAx_DSCTn_NEXT **n = 0,1..3**	LPPDMAx_BA+0x000C+0x10*n	R/W	Next Scatter-gather Descriptor Table Offset Address of LPPDMA Channel n	0xXXXX_XXXX
LPPDMAx_CURSCATn **n = 0,1..3**	LPPDMAx_BA+0x0040+0x004*n	R	Current Scatter-gather Descriptor Table Address of LPPDMA Channel n	0xXXXX_XXXX
LPPDMAx_CHCTL	LPPDMAx_BA+0x400	R/W	LPPDMA Channel Control Register	0x0000_0000
LPPDMAx_PAUSE	LPPDMAx_BA+0x404	W	LPPDMA Transfer Pause Control Register	0x0000_0000
LPPDMAx_SWREQ	LPPDMAx_BA+0x408	W	LPPDMA Software Request Register	0x0000_0000
LPPDMAx_TRGSTS	LPPDMAx_BA+0x40C	R	LPPDMA Channel Request Status Register	0x0000_0000
LPPDMAx_PRISET	LPPDMAx_BA+0x410	R/W	LPPDMA Fixed Priority Setting Register	0x0000_0000
LPPDMAx_PRICLR	LPPDMAx_BA+0x414	W	LPPDMA Fixed Priority Clear Register	0x0000_0000
LPPDMAx_INTEN	LPPDMAx_BA+0x418	R/W	LPPDMA Interrupt Enable Register	0x0000_0000
LPPDMAx_INTSTS	LPPDMAx_BA+0x41C	R/W	LPPDMA Interrupt Status Register	0x0000_0000
LPPDMAx_ABTSTS	LPPDMAx_BA+0x420	R/W	LPPDMA Channel Read/Write Target Abort Flag Register	0x0000_0000
LPPDMAx_TDSTS	LPPDMAx_BA+0x424	R/W	LPPDMA Channel Transfer Done Flag Register	0x0000_0000
LPPDMAx_ALIGN	LPPDMAx_BA+0x428	R/W	LPPDMA Transfer Alignment Status Register	0x0000_0000
LPPDMAx_TACTSTS	LPPDMAx_BA+0x42C	R	LPPDMA Transfer Active Flag Register	0x0000_0000
LPPDMAx_SCATBA	LPPDMAx_BA+0x43C	R/W	LPPDMA Scatter-gather Descriptor Table Base Address Register	0x2800_0000
LPPDMAx_CHRST	LPPDMAx_BA+0x460	R/W	LPPDMA Channel Reset Register	0x0000_0000
LPPDMAx_REQSEL0_3	LPPDMAx_BA+0x480	R/W	LPPDMA Request Source Select Register 0	0x0000_0000

6.9.7 Register Description

Descriptor Table Control Register (LPPDMAx_DSCTn_CTL)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_DSCTn_CTL	LPPDMAx_BA+0x10*n	R/W	Descriptor Table Control Register of LPPDMA Channel n	0xXXXX_XXXX

31	30	29	28	27	26	25	24
TXCNT							
23	22	21	20	19	18	17	16
TXCNT							
15	14	13	12	11	10	9	8
Reserved		TXWIDTH		DAINC		SAINC	
7	6	5	4	3	2	1	0
TBINTDIS	BURSIZE			Reserved	TXTYPE	OPMODE	

Bits	Description	
[31:16]	**TXCNT**	**Transfer Count** The TXCNT represents the required number of LPPDMA transfer, the real transfer count is (TXCNT + 1); The maximum transfer count is 65536, every transfer may be byte, half-word or word that is dependent on TXWIDTH field. **Note:** When LPPDMA finishes each transfer data, this field will be decreased immediately.
[15:14]	**Reserved**	Reserved.
[13:12]	**TXWIDTH**	**Transfer Width Selection** This field is used for transfer width. 00 = One byte (8 bit) is transferred for every operation. 01= One half-word (16 bit) is transferred for every operation. 10 = One word (32-bit) is transferred for every operation. 11 = Reserved. **Note:** LPPDMA transfer source address (LPPDMA_DSCTn_SA) and LPPDMA transfer destination address (LPPDMA_DSCTn_DA) should be alignment under the TXWIDTH selection
[11:10]	**DAINC**	**Destination Address Increment** This field is used to set the destination address increment size. 11 = No increment (fixed address). Others = Increment and size is depended on TXWIDTH selection. **Note:** The fixed address function does not support in memory to memory transfer type.
[9:8]	**SAINC**	**Source Address Increment** This field is used to set the source address increment size. 11 = No increment (fixed address). Others = Increment and size is depended on TXWIDTH selection. **Note:** The fixed address function does not support in memory to memory transfer type.
[7]	**TBINTDIS**	**Table Interrupt Disable Bit** This field can be used to decide whether to enable table interrupt or not. If the TBINTDIS bit is 1, it will not

Bits	Description	
		generates TDIFn (LPPDMA_TDSTS[3:0]) when LPPDMA finishes transfer task. 0 = Table interrupt Enabled. 1 = Table interrupt Disabled. **Note1:** This function only for Scatter-gather mode. **Note2:** This bit must be set to 0 in basic mode.
[6:4]	**BURSIZE**	**Burst Size** This field is used for peripheral to determine the burst size or used for determine the re-arbitration size. 000 = 128 Transfers. 001 = 64 Transfers. 010 = 32 Transfers. 011 = 16 Transfers. 100 = 8 Transfers. 101 = 4 Transfers. 110 = 2 Transfers. 111 = 1 Transfers. **Note:** This field is only useful in burst transfer type.
[3]	**Reserved**	Reserved.
[2]	**TXTYPE**	**Transfer Type** 0 = Burst transfer type. 1 = Single transfer type.
[1:0]	**OPMODE**	**LPPDMA Operation Mode Selection** 00 = Idle state: Channel is stopped or this table is complete, when LPPDMA finish channel table task, OPMODE will be cleared to idle state automatically. 01 = Basic mode: The descriptor table only has one task. When this task is finished, LPPDMA_INTSTS[1] will be asserted. 10 = Scatter-gather mode: When operating in this mode, user must give the next descriptor table address in LPPDMA_DSCTn_NEXT register; LPPDMA will ignore this task, then load the next task to execute. 11 = Reserved. **Note:** Before filling new transfer task in the Descriptor Table, user must check LPPDMA_INTSTS[1] to make sure the current task is complete.

Start Source Address Register (LPPDMAx_DSCTn_SA)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_DSCTn_SA	LPPDMAx_BA+0x0004+0x10*n	R/W	Source Address Register of LPPDMA Channel n	0xXXXX_XXXX

31	30	29	28	27	26	25	24
SA							
23	22	21	20	19	18	17	16
SA							
15	14	13	12	11	10	9	8
SA							
7	6	5	4	3	2	1	0
SA							

Bits	Description	
[31:0]	SA	**LPPDMA Transfer Source Address** This field indicates a 32-bit source address of LPPDMA.

Destination Address Register (LPPDMAx_DSCTn_DA)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_DSCTn_DA	LPPDMAx_BA+0x0008+0x10*n	R/W	Destination Address Register of LPPDMA Channel n	0xXXXX_XXXX

31	30	29	28	27	26	25	24
DA							
23	22	21	20	19	18	17	16
DA							
15	14	13	12	11	10	9	8
DA							
7	6	5	4	3	2	1	0
DA							

Bits	Description	
[31:0]	DA	**LPPDMA Transfer Destination Address** This field indicates a 32-bit destination address of LPPDMA.

Next Scatter-gather Descriptor Table Offset Address (LPPDMAx_DSCTn_NEXT)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_DSCTn_NEXT	LPPDMAx_BA+0x000C+0x10*n	R/W	Next Scatter-gather Descriptor Table Offset Address of LPPDMA Channel n	0xXXXX_XXXX

31	30	29	28	27	26	25	24
EXENEXT							
23	22	21	20	19	18	17	16
EXENEXT							
15	14	13	12	11	10	9	8
NEXT							
7	6	5	4	3	2	1	0
NEXT							

Bits	Description	
[31:16]	EXENEXT	**LPPDMA Execution Next Descriptor Table Offset** This field indicates the offset of next descriptor table address of current execution descriptor table in system memory. **Note:** Write operation is useless in this field.
[15:0]	NEXT	**LPPDMA Next Descriptor Table Offset** This field indicates the offset of the next descriptor table address in system memory. **Write Operation:** If the system memory based address is 0x2800_0000 (LPPDMA_SCATBA), and the next descriptor table is start from 0x2800_0100, then this field must fill in 0x0100. **Read Operation:** When operating in Scatter-gather mode, the last two bits NEXT[1:0] will become reserved, and indicate the first next address of system memory. **Note 1:** The descriptor table address must be word boundary. **Note 2:** Before filling transfer task in the descriptor table, user must check if the descriptor table is complete.

Current Scatter-gather Descriptor Table Address (LPPDMAx_CURSCATn)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_CURSCATn	LPPDMAx_BA+0x0040+0x004*n	R	Current Scatter-gather Descriptor Table Address of LPPDMA Channel n	0xXXXX_XXXX

31	30	29	28	27	26	25	24
CURADDR							
23	22	21	20	19	18	17	16
CURADDR							
15	14	13	12	11	10	9	8
CURADDR							
7	6	5	4	3	2	1	0
CURADDR							

Bits	Description	
[31:0]	CURADDR	**LPPDMA Current Description Address (Read Only)** This field indicates a 32-bit current external description address of LPPDMA. **Note:** This field is read only and used for Scatter-gather mode only to indicate the current external description address.

Channel Control Register (LPPDMAx_CHCTL)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_CHCTL	LPPDMAx_BA+0x400	R/W	LPPDMA Channel Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**CHEN3**	**CHEN2**	**CHEN1**	**CHEN0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**CHENn**	**LPPDMA Channel Enable Bits** Set this bit to 1 to enable LPPDMAn operation. Channel cannot be active if it is not set as enabled. 0 = LPPDMA channel [n] Disabled. 1 = LPPDMA channel [n] Enabled. **Note:** Setting the corresponding bit of LPPDMA_PAUSE or LPPDMA_CHRST register will also clear this bit.

LPPDMA Transfer Pause Control Register (LPPDMAx_PAUSE)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_PAUSE	LPPDMAx_BA+0x404	W	LPPDMA Transfer Pause Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PAUSE3**	**PAUSE2**	**PAUSE1**	**PAUSE0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**PAUSEn**	**LPPDMA Channel n Transfer Pause Control (Write Only)** User can set PAUSEn bit field to pause LPPDMA transfer. When user sets PAUSEn bit, LPPDMA will pause the on-going transfer, then clear the channel enable bit CHENn (LPPDMAx_CHCTL[n], n=0,1.. 3) and clear request active flag (LPPDMAx_TRGSTS[n:0], n=0,1.. 3). If the paused channel is re-enabled again, the remaining transfers will be processed. 0 = No effect. 1 = Pause LPPDMA channel n transfer. Note:Only for memmory to memory transfer.

LPPDMA Software Request Register (LPPDMAx_SWREQ)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_SWREQ	LPPDMAx_BA+0x408	W	LPPDMA Software Request Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**SWREQ3**	**SWREQ2**	**SWREQ1**	**SWREQ0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**SWREQn**	**LPPDMA Software Request (Write Only)** Set this bit to 1 to generate a software request to LPPDMA [n]. 0 = No effect. 1 = Generate a software request. **Note 1:** User can read LPPDMAx_TRGSTS register to know which channel is on active. Active flag may be triggered by software request or peripheral request. **Note 2:** If user does not enable corresponding LPPDMA channel, the software request will be ignored.

LPPDMA Channel Request Status Register (LPPDMAx_TRGSTS)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_TRGSTS	LPPDMAx_BA+0x40C	R	LPPDMA Channel Request Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**REQSTS3**	**REQSTS2**	**REQSTS1**	**REQSTS0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**REQSTSn**	**LPPDMA Channel Request Status (Read Only)** This flag indicates whether channel[n] have a request or not, no matter request from software or peripheral. When LPPDMA finishes channel transfer, this bit will be cleared automatically. 0 = LPPDMA Channel n has no request. 1 = LPPDMA Channel n has a request. **Note:** If user pauses or resets each LPPDMA transfer by setting LPPDMA_PAUSE or LPPDMA_CHRST register respectively, this bit will be cleared automatically after finishing the current transfer.

LPPDMA Fixed Priority Setting Register (LPPDMAx_PRISET)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_PRISET	LPPDMAx_BA+0x410	R/W	LPPDMA Fixed Priority Setting Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**FPRISET3**	**FPRISET2**	**FPRISET1**	**FPRISET0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**FPRISETn**	**LPPDMA Fixed Priority Setting** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set LPPDMA channel [n] to fixed priority channel. **Read Operation:** 0 = Corresponding LPPDMA channel is round-robin priority. 1 = Corresponding LPPDMA channel is fixed priority. **Note:** This field is only set to fixed priority. To clear fixed priority, use LPPDMA_PRICLR register.

LPPDMA Fix Priority Clear Register (LPPDMAx_PRICLR)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_PRICLR	LPPDMAx_BA+0x414	W	LPPDMA Fixed Priority Clear Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**FPRICLR3**	**FPRICLR2**	**FPRICLR1**	**FPRICLR0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**FPRICLRn**	**LPPDMA Fixed Priority Clear Bits (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear LPPDMA channel [n] fixed priority setting. **Note:** User can read LPPDMA_PRISET register to know the channel priority.

LPPDMA Interrupt Enable Register (LPPDMAx_INTEN)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_INTEN	LPPDMAx_BA+0x418	R/W	LPPDMA Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**INTEN3**	**INTEN2**	**INTEN1**	**INTEN0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**INTENn**	**LPPDMA Interrupt Enable Bits** This field is used to enable LPPDMA channel[n] interrupt. 0 = LPPDMA channel n interrupt Disabled. 1 = LPPDMA channel n interrupt Enabled. **Note:** The interrupt flag is abort, transfer done and align.

NLPPDMA Interrupt Status Register (LPPDMAx_INTSTS)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_INTSTS	LPPDMAx_BA+0x41C	R/W	LPPDMA Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**WKF**	**ALIGNF**	**TDIF**	**ABTIF**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**WKF**	**Wake Up Flag** 0 = LPPDMA no wake up event. 1 = LPPDMA wake up event happened. **Note:** User can write 1 to clear these bit
[2]	**ALIGNF**	**Transfer Alignment Interrupt Flag (Read Only)** 0 = LPPDMA channel source address and destination address both follow transfer width setting. 1 = LPPDMA channel source address or destination address is not follow transfer width setting.
[1]	**TDIF**	**Transfer Done Interrupt Flag (Read Only)** This bit indicates that LPPDMA controller has finished transmission; User can read LPPDMA_TDSTS register to indicate which channel finished transfer. 0 = Not finished yet. 1 = LPPDMA channel has finished transmission.
[0]	**ABTIF**	**LPPDMA Read/Write Target Abort Interrupt Flag (Read Only)** This bit indicates that LPPDMA has target abort error; Software can read LPPDMA_ABTSTS register to find which channel has target abort error. 0 = No AHB bus ERROR response received. 1 = AHB bus ERROR response received.

LPPDMA Channel Read/Write Target Abort Flag Register (LPPDMAx_ABTSTS)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_ABTSTS	LPPDMAx_BA+0x420	R/W	LPPDMA Channel Read/Write Target Abort Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**ABTIF3**	**ABTIF2**	**ABTIF1**	**ABTIF0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**ABTIFn**	**LPPDMA Read/Write Target Abort Interrupt Status Flag** This bit indicates which LPPDMA controller has target abort error. 0 = No AHB bus ERROR response received when channel n transfer. 1 = AHB bus ERROR response received when channel n transfer. **Note 1:** If channel n target abort, REQSRCn should set 0 to disable peripheral request. **Note 2:** User can write 1 to clear this bit.

LPPDMA Channel Transfer Done Flag Register (LPPDMAx_TDSTS)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_TDSTS	LPPDMAx_BA+0x424	R/W	LPPDMA Channel Transfer Done Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**TDIF3**	**TDIF2**	**TDIF1**	**TDIF0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**TDIFn**	**Transfer Done Flag** This bit indicates whether LPPDMA controller channel transfer has been finished or not. 0 = LPPDMA channel transfer has not finished. 1 = LPPDMA channel has finished transmission. **Note:** User can write 1 to clear these bits.

LPPDMA Transfer Alignment Status Register (LPPDMAx_ALIGN)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_ALIGN	LPPDMAx_BA+0x428	R/W	LPPDMA Transfer Alignment Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**ALIGN3**	**ALIGN2**	**ALIGN1**	**ALIGN0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**ALIGNn**	**Transfer Alignment Flag** This bit indicates whether source and destination address both follow transfer width setting. 0 = LPPDMA channel source address and destination address both follow transfer width setting. 1 = LPPDMA channel source address or destination address is not follow transfer width setting. **Note:** User can write 1 to clear these bits.

LPPDMA Transfer Active Flag Register (LPPDMAx_TACTSTS)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_TACTSTS	LPPDMAx_BA+0x42C	R	LPPDMA Transfer Active Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**TXACTF3**	**TXACTF2**	**TXACTF1**	**TXACTF0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**TXACTFn**	**Transfer on Active Flag (Read Only)** This bit indicates which LPPDMA channel is in active. 0 = LPPDMA channel is finished. 1 = LPPDMA channel is active.

LPPDMA Scatter-gather Descriptor Table Base Address Register (LPPDMAx_SCATBA)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_SCATBA	LPPDMAx_BA+0x43C	R/W	LPPDMA Scatter-gather Descriptor Table Base Address Register	0x2800_0000

31	30	29	28	27	26	25	24
SCATBA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SCATBA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:16]	**SCATBA**	**LPPDMA Scatter-gather Descriptor Table Address** In Scatter-gather mode, this is the base address for calculating the next link - list address. The next link address equation is Next Link Address = LPPDMA_SCATBA + LPPDMA_DSCTn_NEXT. **Note:** Only useful in Scatter-gather mode.
[15:0]	**Reserved**	Reserved.

LPPDMA Channel Reset Register (LPPDMAx_CHRST)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_CHRST	LPPDMAx_BA+0x460	R/W	LPPDMA Channel Reset Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**CH3RST**	**CH2RST**	**CH1RST**	**CH0RST**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[n] n=0,1..3	**CHnRST**	**LPPDMA Channel n Reset Control** User can set CHnRST bit field to reset LPPDMA transfer. When user sets CHnRST bit, LPPDMA will stop the on-going transfer, then clear channel enable bit CHENn (LPPDMAx_CHCTL[n], n=0,1.. 3) and clear request active flag (LPPDMAx_TRGSTS[n:0], n=0,1.. 3). User should re-initialize corresponding channel setting. 0 = Corresponding channel n is not reset. 1 = Corresponding channel n is reset. **Note1:** After LPPDMA current request transfer is complete, this bit will be cleared automatically. Note2: If channel trigger source come from peripheral, set the REQSRCn to 0 before set the channel reset.

LPPDMA Request Source Select Register 0 (LPPDMAx_REQSEL0_3)

Register	Offset	R/W	Description	Reset Value
LPPDMAx_REQSEL0_3	LPPDMAx_BA+0x480	R/W	LPPDMA Request Source Select Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC3						
23	22	21	20	19	18	17	16
Reserved	REQSRC2						
15	14	13	12	11	10	9	8
Reserved	REQSRC1						
7	6	5	4	3	2	1	0
Reserved	REQSRC0						

Bits	Description	
[31]	**Reserved**	Reserved.
[30:24]	**REQSRC3**	**Channel 3 Request Source Selection** This filed defines which peripheral is connected to LPPDMA channel 3. User can configure the peripheral setting by REQSRC3. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23]	**Reserved**	Reserved.
[22:16]	**REQSRC2**	**Channel 2 Request Source Selection** This filed defines which peripheral is connected to LPPDMA channel 2. User can configure the peripheral setting by REQSRC2. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15]	**Reserved**	Reserved.
[14:8]	**REQSRC1**	**Channel 1 Request Source Selection** This filed defines which peripheral is connected to LPPDMA channel 1. User can configure the peripheral setting by REQSRC1. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7]	**Reserved**	Reserved.
[6:0]	**REQSRC0**	**Channel 0 Request Source Selection** This filed defines which peripheral is connected to LPPDMA channel 0. User can configure the peripheral by setting REQSRC0. 0 = Disable LPPDMA peripheral request. 1 = Channel connects to LPUART0_TX. 2 = Channel connects to LPUART0_RX . 3 = Channel connects to LPSPI0_TX. 4 = Channel connects to LPSPI0_RX. 5 = Channel connects to LPI2C0_TX.

Bits	Description	
		6 = Channel connects to LPI2C0_RX. 7 = Channel connects to LPTIMER0. 8 = Channel connects to LPTIMER1. 9 = Channel connects to TTIMER0. 10=Channel connects to TTIMER1. 11 = Channel connects to LPADC0_RX. 12 = Reserved. 13 = Reserved . 14 = Channel connects to ACMP0. 15 = Channel connects to ACMP1. 16 = Channel connects to ACMP2. Others = Reserved. **Note 1:** A peripheral cannot be assigned to two channels at the same time. **Note 2:** This field is useless when transfer between memory and memory.

6.10 External Bus Interface (EBI)

6.10.1 Overview

This chip is equipped with an external bus interface (EBI) for external device use. To save the connections between an external device and a chip, EBI is operating at address bus and data bus multiplex mode. The EBI supports three chip selects that can connect three external devices with different timing setting requirements.

6.10.2 Features

- Supports up to three memory banks
- Supports dedicated external chip select pin with polarity control for each bank
- Supports accessible space up to 1 Mbytes for each bank, actually external addressable space is dependent on package pin out
- Supports 8-/16-bit data width
- Supports byte write in 16-bit data width mode
- Supports address bus and data bus multiplex mode
- Supports address bus and data bus separate mode
- Supports Timing parameters individual adjustment for each memory block
- Supports LCD interface i80 mode
- Supports PDMA mode
- Supports variable external bus base clock (MCLK) which based on HCLK
- Supports configurable idle cycle for different access condition: Idle of Write command finish (W2X) and Idle of Read-to-Read (R2R)

6.10.3 Block Diagram

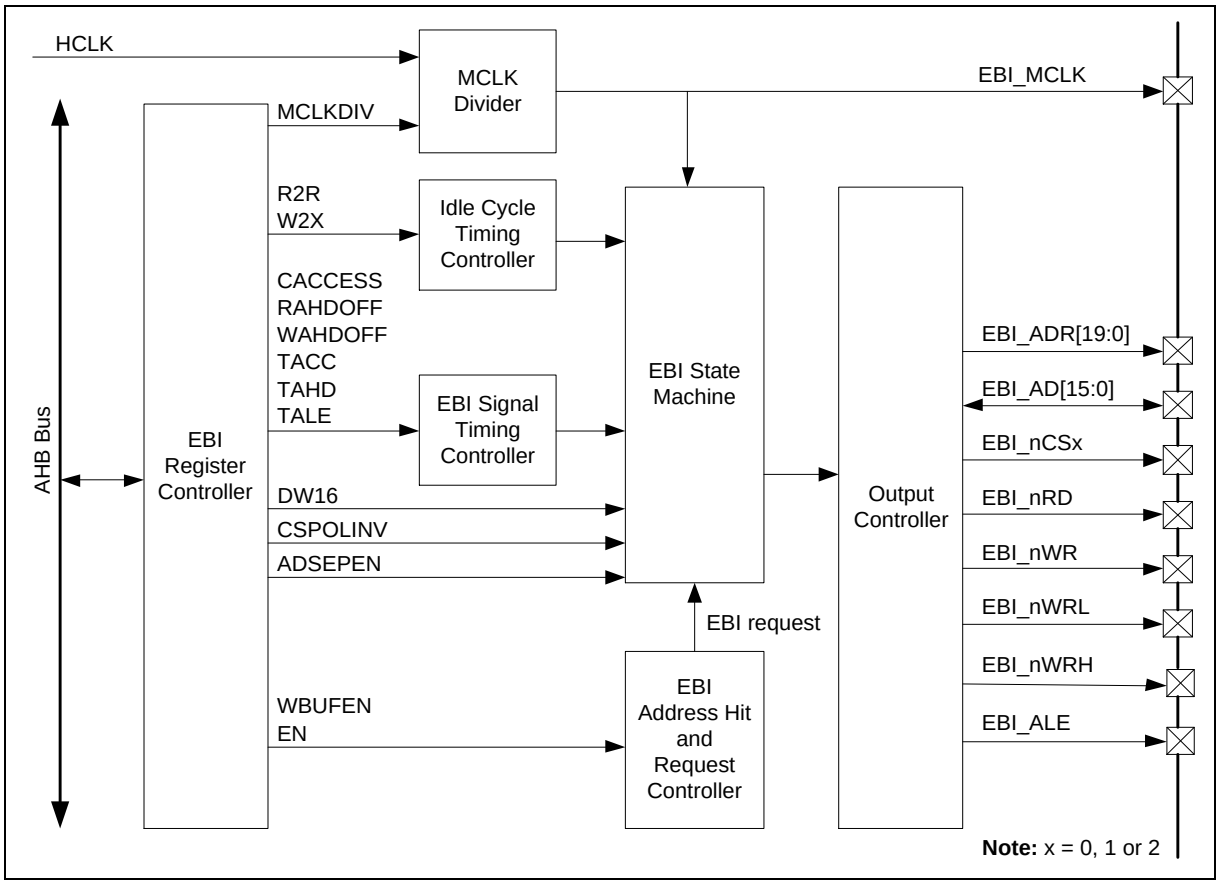


Figure 6.10-1 EBI Block Diagram

6.10.4 Basic Configuration

- Clock Source Configuration
  - Enable EBI controller clock in EBICKEN (CLK_AHBCLK0[3]).
- Reset Configuration
  - Reset EBI controller in EBIRST (SYS_IPRST0[3]).

6.10.5 Functional Description

6.10.5.1 EBI Area and Address Hit

The EBI mapping address is located at 0x6000_0000 ~ 0x602F_FFFF and the total memory space is 3 Mbytes. When system request address hits EBI's memory space, the corresponding EBI chip select signal is assert and EBI state machine operates.

Chip Select	Address Mapping
EBI_nCS0	0x6000_0000 ~ 0x600F_FFFF
EBI_nCS1	0x6010_0000 ~ 0x601F_FFFF
EBI_nCS2	0x6020_0000 ~ 0x602F_FFFF

Table 6.10-1 EBI Address Mapping

To map the whole EBI memory space, it requires 20-bit address for 8-bit data width device and 19-bit address for 16-bit data width device. For package that output less than 20-bit address, EBI will map device to mirror space. For example, the package with 18-bit EBI address, EBI will mapped external device (for Bank0/EBI_nCS0) to 0x6000_0000 ~ 0x6003_FFFF, 0x6004_0000 ~ 0x6007_FFFF, 0x6008_0000 ~ 0x600B_FFFF and 0x600C_0000 ~ 0x600F_FFFF simultaneously.

6.10.5.2 EBI Data Width Connection - Address Bus and Data Bus Multiplex Mode

The EBI supports the device whose address bus and data bus are multiplexed. For the external device with separated address and data bus, the connection to device needs additional latch device to latch the address. In this case, the pin EBI_ALE is connected to the latch device to latch the address value. Pin EBI_AD is the input of the latch device, and the output of the latch device is connected to the address of external device.

For 16-bit device, the EBI_AD[15:0] is shared by address and 16-bit data, and EBI_ADR[18:16] is dedicated for address and could be connected to 16-bit device directly. The EBI_ADR[19] will be ignored when EBI data width is set as 16-bit width. For 8-bit device, only EBI_AD[7:0] is shared by address and 8-bit data, EBI_AD[15:8] and EBI_ADR[19:16] are dedicated for address and could be connected to 8-bit device directly. Figure 6.10-2 shows the connection of 16-bit data width device and Figure 6.10-3 shows the connection of 8-bit data width device.

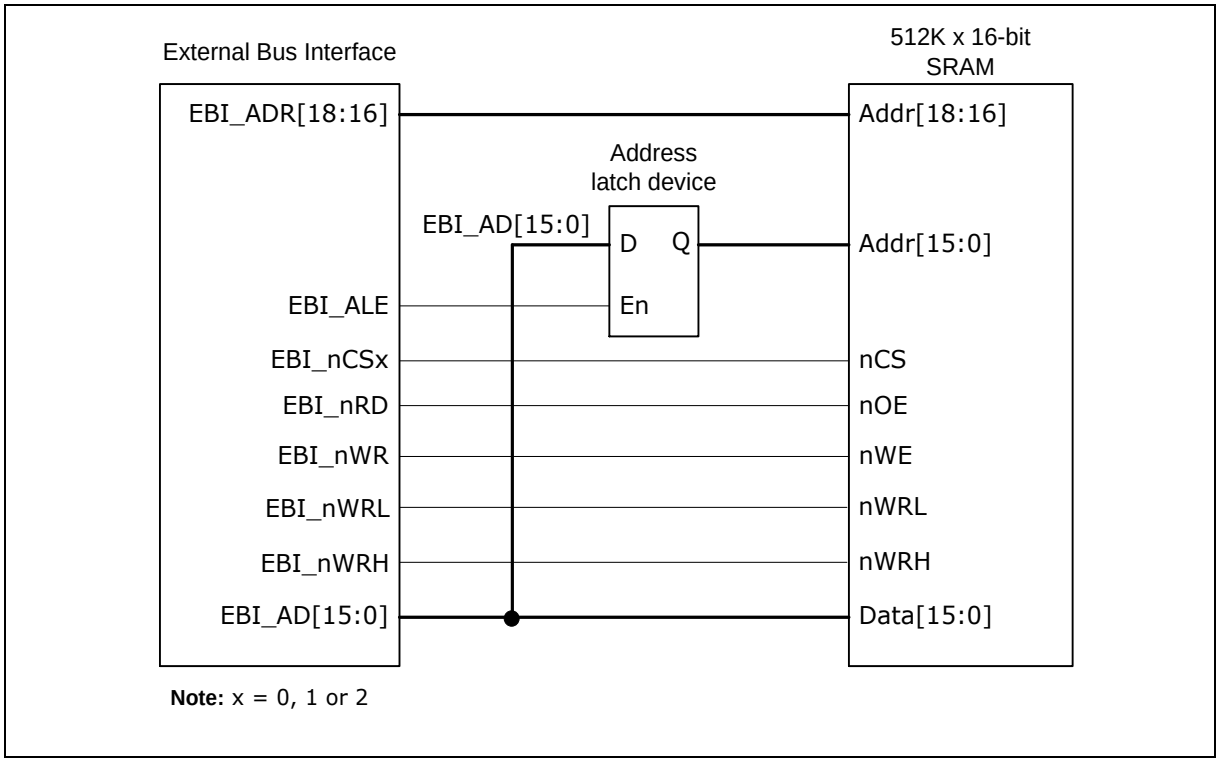


Figure 6.10-2 Connection of 16-bit EBI Data Width with 16-bit Device

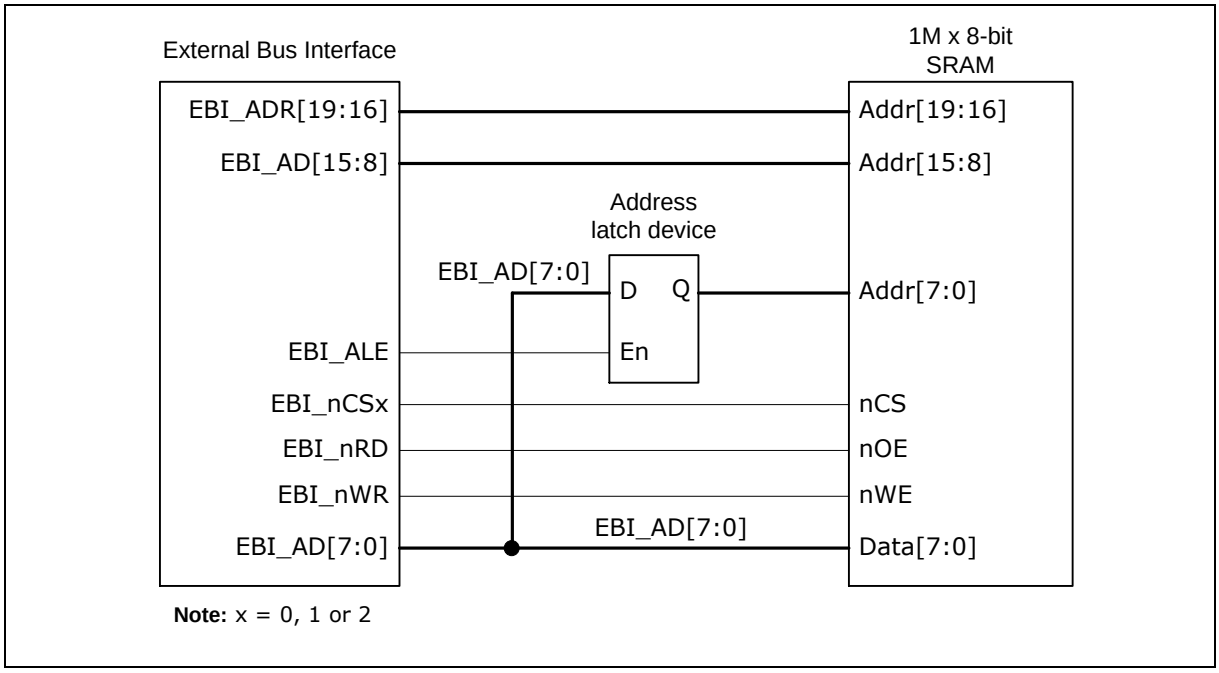


Figure 6.10-3 Connection of 8-bit EBI Data Width with 8-bit Device

When system access data width is larger than EBI data width, the EBI controller will finish a system access command by operating EBI access more than once. For example, if system requests a 32-bit data through EBI device, the EBI controller will operate accessing four times when setting EBI data width with 8-bit.

6.10.5.3 EBI Data Width Connection - Address Bus and Data Bus Separate Mode

The EBI supports address and data bus separate mode. User can enable this mode by setting ADSEPEN (EBI_CTLx[3]). When separate mode is enabled, EBI_AD is dedicated for data bus and connected directly to device data bus, EBI_ADR is dedicated for address bus.

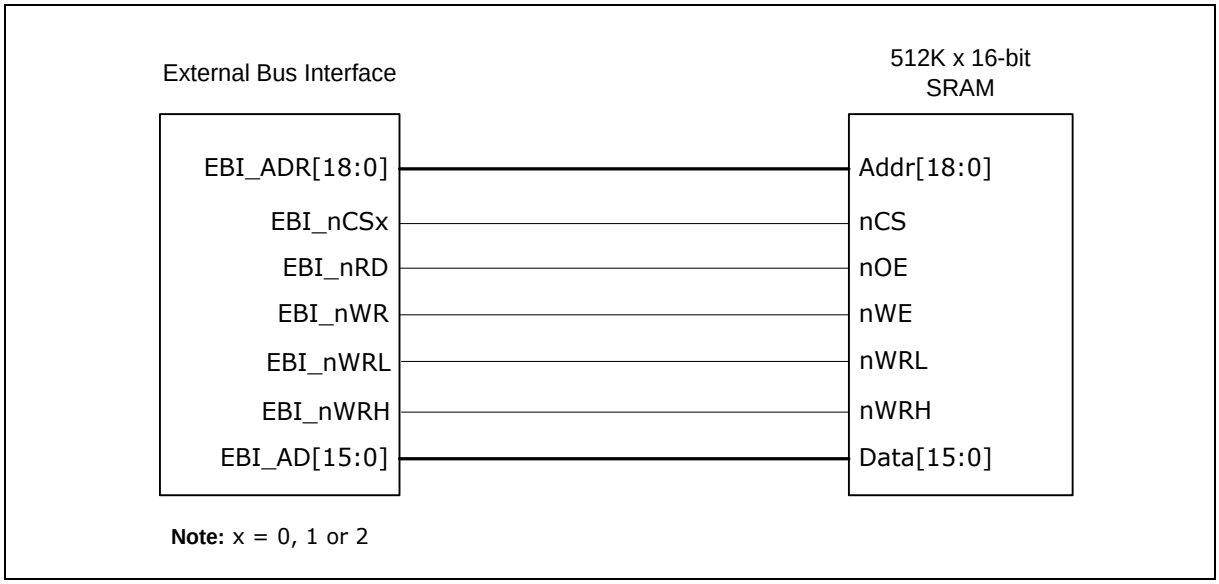


Figure 6.10-4 Connection of 16-bit EBI Data Width with 16-bit Device in Separate mode

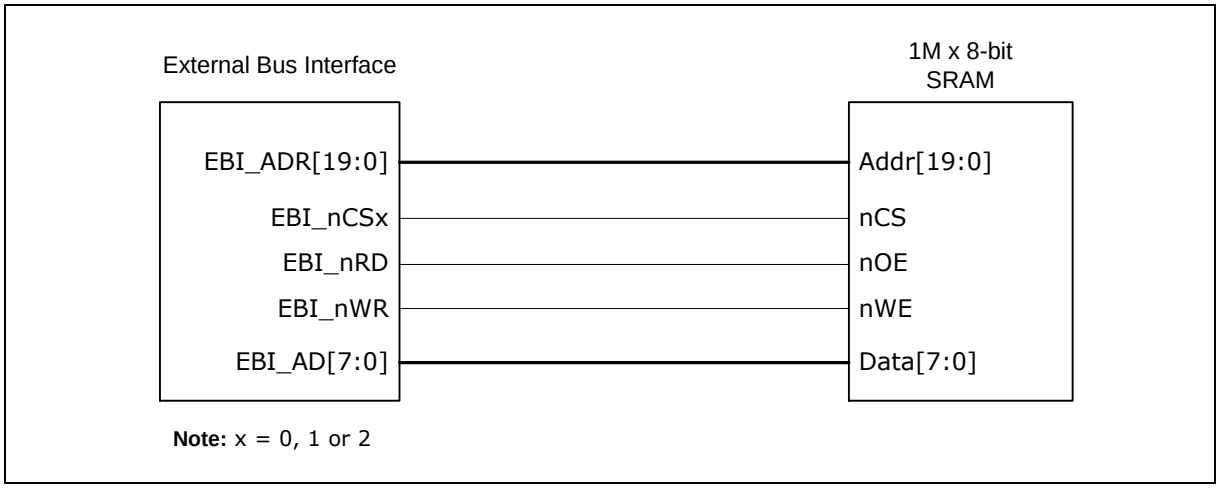


Figure 6.10-5 Connection of 8-bit EBI Data Width with 8-bit Device in Separate mode

6.10.5.4 EBI Operating Control

MCLK Control

In the chip, all EBI signals will be synchronized by EBI_MCLK when EBI is operating. When chip connects to the external device with slower operating frequency, the EBI_MCLK can divide most to HCLK/128 by setting MCLKDIV (EBI_CTLx[10:8]). Therefore, chip can be suitable for a wide frequency range of EBI device. EBI signals are all synchronized by negative edge of EBI_MCLK.

Operation and Access Timing Control

At the start of EBI access, chip select (EBI_nCS0, EBI_nCS1 and EBI_nCS2) asserts to low and wait one EBI_MCLK for address setup time (tASU) for address stable. Then EBI_ALE asserts to high after address is stable and keeps for a period of time (tALE) for address latch. After latch address, EBI_ALE asserts to low and wait one EBI_MCLK for latch hold time (tLHD) and another one EBI_MCLK cycle (tA2D) that is inserted behind address hold time to be the bus turn-around time for address change to data. Then EBI_nRD asserts to low when read access or EBI_nWR asserts to low when write access. Then EBI_nRD or EBI_nWR asserts to high after keeps access time (tACC) for reading output stable or writing finish. After that, EBI signals keep for data access hold time (tAHD) and chip select asserts to high, address is released by current access control.

The EBI controller provides a flexible timing control for different external device. In EBI timing control, tASU, tLHD and tA2D are fixed to 1 EBI_MCLK cycle, tAHD can modulate to 1~8 EBI_MCLK cycles by setting TAHD (EBI_TCTLx[10:8]), tACC can modulate to 1~32 EBI_MCLK cycles by setting TACC (EBI_TCTLx[7:3]), and tALE can modulate to 1~8 EBI_MCLK cycles by setting TALE (EBI_CTL0[18:16]). Some external device can support zero data access hold time accessing, the EBI controller can skipped tAHD to increase access speed by setting WAHDOFF (EBI_TCTLx[23]) and RAHDOFF (EBI_TCTLx[22]).

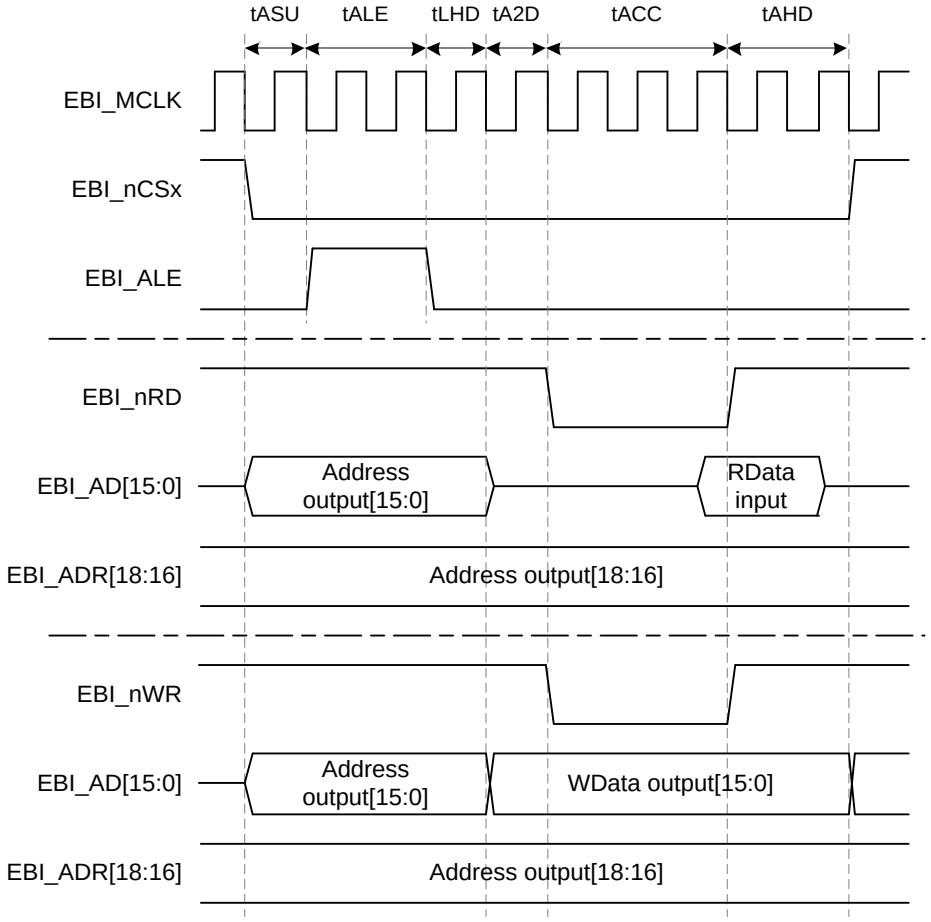
For each chip select, the EBI provides individual register with timing control except that tALE can only be controlled by EBI_CTL0.

Parameter	Value	Unit	Description
tASU	1	MCLK	Address Latch Setup Time.
tALE	1 ~ 8	MCLK	ALE High Period. Controlled by TALE (EBI_CTL0[18:16]).
tLHD	1	MCLK	Address Latch Hold Time.

tA2D	1	MCLK	Address To Data Delay (Bus Turn-Around Time).
tACC	1 ~ 32	MCLK	Data Access Time. Controlled by TACC (EBI_TCTLx[7:3]).
tAHD	1 ~ 8	MCLK	Data Access Hold Time. Controlled by TAHD (EBI_TCTLx[10:8]).
IDLE	0 ~ 15	MCLK	Idle Cycle. Controlled by R2R (EBI_TCTLx[27:24]) and W2X (EBI_TCTLx[15:12]).

Table 6.10-2 Timing Control Parameter

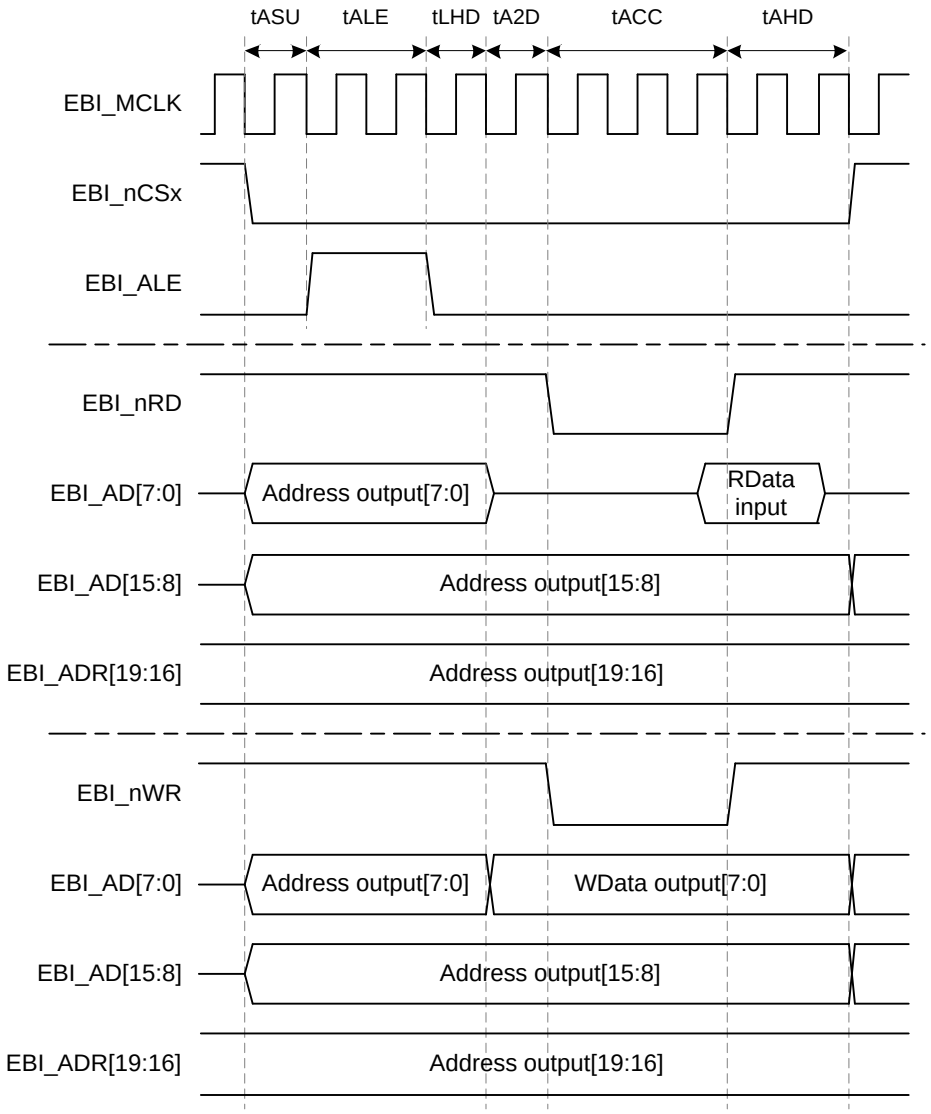
Figure 6.10-6 shows an example of setting 16-bit data width. In this example, EBI_AD bus is used for being address [15:0] and data [15:0]. When EBI_ALE assert to high, EBI_AD is address output. After address is latched, EBI_ALE asserts to low and the EBI_AD bus change to high impedance to wait device output data in read access operation, or it is used for being write data output.



Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1 or 2

Figure 6.10-6 Timing Control Waveform for 16-bit Data Width

Figure 6.10-7 shows an example of setting 8-bit data width. The difference between 8-bit and 16-bit data width is EBI_AD[15:8]. In 8-bit data width setting, EBI_AD[15:8] is always Address [15:8] output so that external latch needs only 8-bit width.

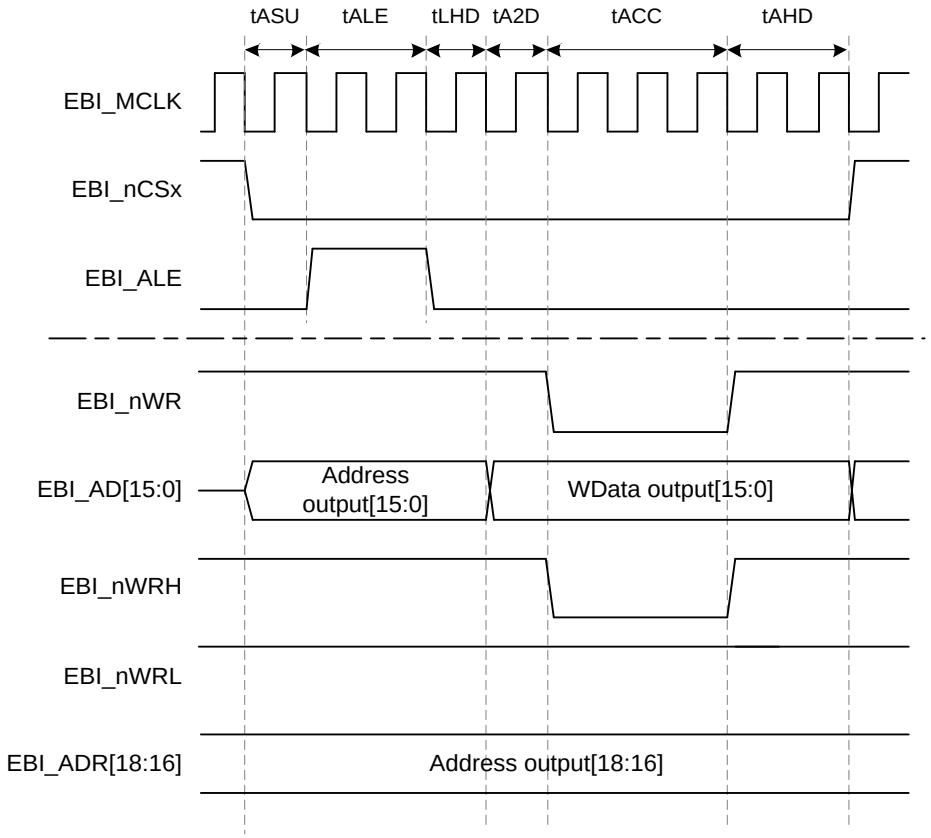


Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1 or 2

Figure 6.10-7 Timing Control Waveform for 8-bit Data Width

Byte Access

The EBI supports byte access when connected to 16-bit device. The pin EBI_nWRH and EBI_nWRL assertion indicate high byte enable and low byte enable in 16-bit data bus. Figure 6.10-8 shows the write operation of 8-bit width data in EBI_AD[15:8] with EBI_nWRH assertion.

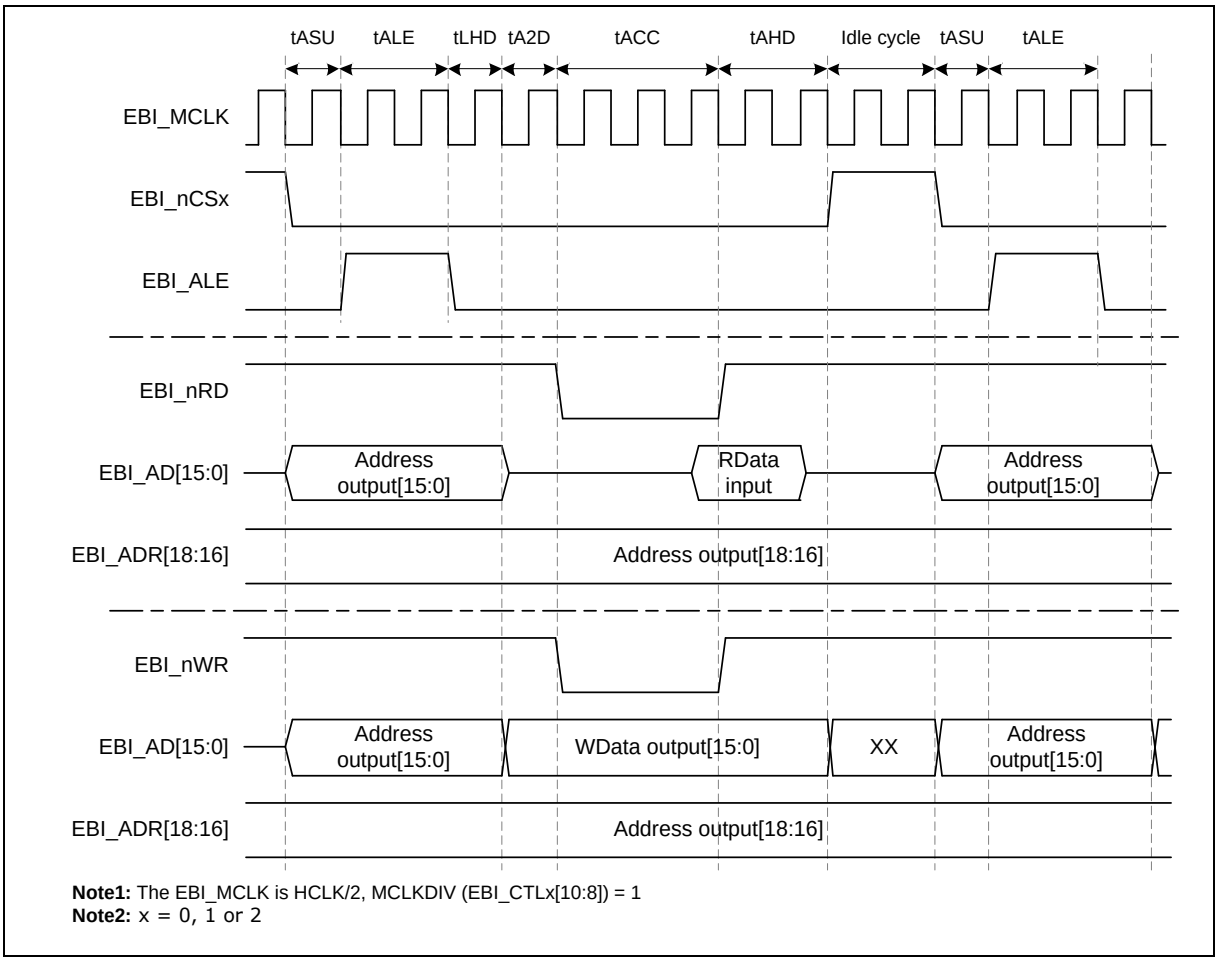


Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1 or 2
Note3: WData[15:8] is valid with EBI_nWRH assertion

Figure 6.10-8 Timing Control Waveform for Byte Write in 16-bit Data Mode

Insert Idle Cycle

When EBI accesses continuously, there may occur bus conflict if the device access time is much slow with system operating. The EBI controller supplies additional idle cycle to solve this problem. During idle cycle, all control signals of EBI are inactive. Figure 6.10-9 shows idle cycles.



Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1 or 2

Figure 6.10-9 Timing Control Waveform for Insert Idle Cycle

There are two conditions that EBI can insert idle cycle by timing control:

1. After write access (W2X idle cycle)
2. After read access and before next read access (R2R idle cycle)

By setting W2X (EBI_TCTLx[15:12]), and R2R (EBI_TCTLx[27:24]), the time of idle cycle can be specified from 0~15 EBI_MCLK.

Chip Select Polarity Control

The EBI supports chip select polarity control for connecting to variable external device. When CSPOLINV (EBI_CTLx[2]) is set to 0, the chip select pins (EBI_nCSx) works as low active behavior. It means the external device can be access under EBI_nCSx at low state. When CSPOLINV (EBI_CTLx[2]) is set to 1, the chip select pin (EBI_nCS) works as high active behavior. It means the external device can be access under EBI_nCSx at high state.

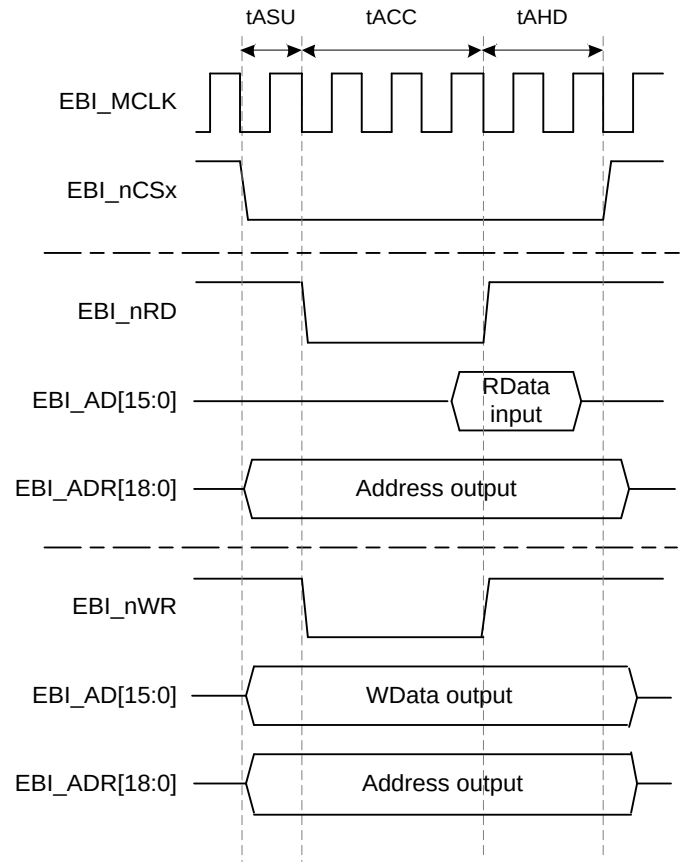
Write Buffer

When user writes data to an external device through EBI bus, the EBI controller will start processing the write action immediately and the CPU is held until current EBI write action is finished. User can enable write buffer function to improve CPU and EBI access performance. When EBI write buffer function is enabled, the CPU can continuously execute other instruction during EBI controller process the write action to external device. There is one exception condition for this case. If CPU executes another data access through EBI bus when EBI process write action, the CPU will be held.

User can enable write buffer by setting WBUFEN (EBI_CTL0[24]).

Address Data Separate Mode

When EBI is set as separate mode, the tALE, tLHD, tA2D cycles are ignored. EBI_AD and EBI_ADR are dedicated for data and address bus separately.



Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1 or 2

Figure 6.10-10 Timing Control Waveform for 16-bit Data Width for Separate Mode

Continuous Data Access Mode

The EBI supports continuous data access mode for the device which needs faster data access and do not need address control interface. User can enable this mode by setting CACCESS (EBI_CTLx[4]) for each bank. When EBI is set as continuous data access mode, the tASU, tALE, tLHD cycles are ignored and EBI can access data continuously within one read or write command. There will be dummy cycle between each access command. The timing waveform is shown as Figure 6.10-11.

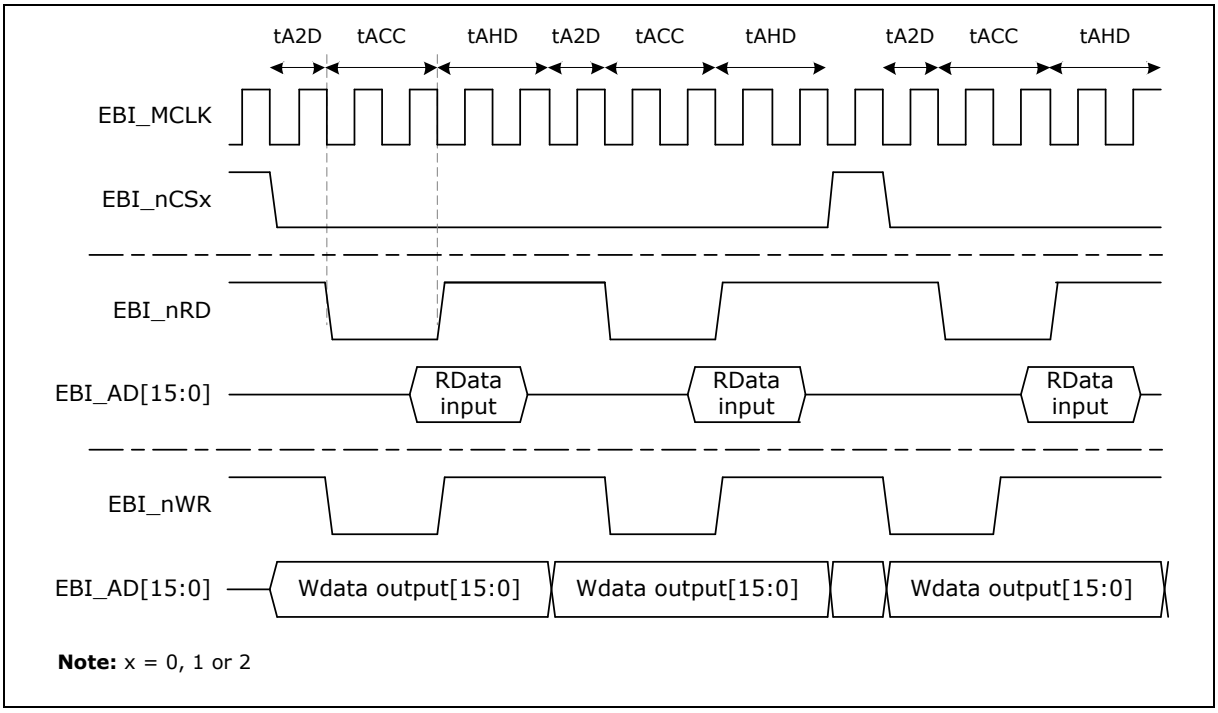


Figure 6.10-11 Timing Control Waveform for Continuous Data Access Mode

6.10.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
EBI Base Address: **EBI_BA = 0x4001_0000**				
EBI_CTL0	EBI_BA+0x00	R/W	External Bus Interface Bank0 Control Register	0x0000_0000
EBI_TCTL0	EBI_BA+0x04	R/W	External Bus Interface Bank0 Timing Control Register	0x0000_0000
EBI_CTL1	EBI_BA+0x10	R/W	External Bus Interface Bank1 Control Register	0x0000_0000
EBI_TCTL1	EBI_BA+0x14	R/W	External Bus Interface Bank1 Timing Control Register	0x0000_0000
EBI_CTL2	EBI_BA+0x20	R/W	External Bus Interface Bank2 Control Register	0x0000_0000
EBI_TCTL2	EBI_BA+0x24	R/W	External Bus Interface Bank2 Timing Control Register	0x0000_0000

6.10.7 Register Description

External Bus Interface Control Register (EBI_CTLx)

Register	Offset	R/W	Description	Reset Value
EBI_CTL0	EBI_BA+0x00	R/W	External Bus Interface Bank0 Control Register	0x0000_0000
EBI_CTL1	EBI_BA+0x10	R/W	External Bus Interface Bank1 Control Register	0x0000_0000
EBI_CTL2	EBI_BA+0x20	R/W	External Bus Interface Bank2 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reversed							WBUFEN
23	22	21	20	19	18	17	16
Reversed					TALE		
15	14	13	12	11	10	9	8
Reversed					MCLKDIV		
7	6	5	4	3	2	1	0
Reversed			CACCESS	ADSEPEN	CSPOLINV	DW16	EN

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**WBUFEN**	**EBI Write Buffer Enable Bit** 0 = EBI write buffer Disabled. 1 = EBI write buffer Enabled. **Note:** This bit is only available in EBI_CTL0 register.
[23:19]	**Reserved**	Reserved.
[18:16]	**TALE**	**Extend Time of ALE** The EBI_ALE high pulse period (tALE) to latch the address can be controlled by TALE. tALE = (TALE + 1) * EBI_MCLK. **Note:** This field is only available in EBI_CTL0 register.
[15:11]	**Reserved**	Reserved.
[10:8]	**MCLKDIV**	**External Output Clock Divider** The frequency of EBI output clock (MCLK) is controlled by MCLKDIV as follow: 000 = HCLK/1. 001 = HCLK/2. 010 = HCLK/4. 011 = HCLK/8. 100 = HCLK/16. 101 = HCLK/32. 110 = HCLK/64. 111 = HCLK/128.
[7:5]	**Reserved**	Reserved.

[4]	**CACCESS**	**Continuous Data Access Mode** When continuous access mode enabled, the tASU, tALE and tLHD cycles are bypass for continuous data transfer request. 0 = Continuous data access mode Disabled. 1 = Continuous data access mode Enabled.
[3]	**ADSEPEN**	**EBI Address/Data Bus Separate Mode Enable Bit** 0 = Address/Data Bus Separate Mode Disabled. 1 = Address/Data Bus Separate Mode Enabled.
[2]	**CSPOLINV**	**Chip Select Pin Polar Inverse** This bit defines the active level of EBI chip select pin (EBI_nCS). 0 = Chip select pin (EBI_nCS) is active low. 1 = Chip select pin (EBI_nCS) is active high.
[1]	**DW16**	**EBI Data Width 16-bit Select** This bit defines if the EBI data width is 8-bit or 16-bit. 0 = EBI data width is 8-bit. 1 = EBI data width is 16-bit.
[0]	**EN**	**EBI Enable Bit** This bit is the functional enable bit for EBI. 0 = EBI function Disabled. 1 = EBI function Enabled.

External Bus Interface Timing Control Register (EBI_TCTLx)

Register	Offset	R/W	Description	Reset Value
EBI_TCTL0	EBI_BA+0x04	R/W	External Bus Interface Bank0 Timing Control Register	0x0000_0000
EBI_TCTL1	EBI_BA+0x14	R/W	External Bus Interface Bank1 Timing Control Register	0x0000_0000
EBI_TCTL2	EBI_BA+0x24	R/W	External Bus Interface Bank2 Timing Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				R2R			
23	22	21	20	19	18	17	16
WAHDOFF	RAHDOFF	Reserved					
15	14	13	12	11	10	9	8
W2X				Reversed	TAHD		
7	6	5	4	3	2	1	0
TACC					Reserved		

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**R2R**	**Idle Cycle Between Read-to-read** This field defines the number of R2R idle cycle. R2R idle cycle = (R2R * EBI_MCLK). When read action is finished and the next action is going to read, R2R idle cycle is inserted and EBI_nCS return to idle state.
[23]	**WAHDOFF**	**Access Hold Time Disable Control When Write** 0 = Data Access Hold Time (tAHD) during EBI writing Enabled. 1 = Data Access Hold Time (tAHD) during EBI writing Disabled.
[22]	**RAHDOFF**	**Access Hold Time Disable Control When Read** 0 = Data Access Hold Time (tAHD) during EBI reading Enabled. 1 = Data Access Hold Time (tAHD) during EBI reading Disabled.
[21:16]	**Reserved**	Reserved.
[15:12]	**W2X**	**Idle Cycle After Write** This field defines the number of W2X idle cycle. W2X idle cycle = (W2X * EBI_MCLK). When write action is finished, W2X idle cycle is inserted and EBI_nCS return to idle state.
[11]	**Reserved**	Reserved.
[10:8]	**TAHD**	**EBI Data Access Hold Time** TAHD defines data access hold time (tAHD). tAHD = (TAHD + 1) * EBI_MCLK.
[7:3]	**TACC**	**EBI Data Access Time** TACC defines data access time (tACC).

		tACC = (TACC + 1) * EBI_MCLK.
[2:0]	**Reserved**	Reserved.

6.11 Timer Controller (TMR)

6.11.1 Overview

The timer controller includes four 32-bit timers, Timer0 ~ Timer3, allowing user to easily implement a timer control for applications. The timer can perform functions, such as frequency measurement, delay timing, clock generation, and event counting by external input pins, and interval measurement by external capture pins.

The timer controller also provides four PWM generators. Each PWM generator supports one PWM output and two selectable PWM output channels (TMx or TMx_EXT). The output state of PWM output pin can be control by polarity control, output enable control and output channel select.

6.11.2 Features

6.11.2.1 Timer Function Features

- Four sets of 32-bit timers, each timer having one 24-bit up counter and one 8-bit prescale counter
- Independent clock source for each timer
- Provides one-shot, periodic, toggle-output and continuous counting operation modes
- 24-bit up counter value is readable through CNT (TIMERx_CNT[23:0])
- Support 3-bit capture input noise filter
- Supports event counting function
- 24-bit capture value is readable through CAPDAT (TIMERx_CAP[23:0])
- Supports external capture pin (TMx_EXT) event for interval measurement
- Supports external capture pin (TMx_EXT) event to reset 24-bit up counter
- Supports internal clock (HIRC, LIRC, MIRC) and external clock (HXT, LXT) for capture event
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated
- Supports Timer0 ~ Timer3 time-out interrupt signal or capture interrupt signal to trigger EPWM, PWM, EADC, LPADC, DAC, EQEI, and PDMA function
- Supports internal capture triggered while internal ACMP output signal transition
- Supports Inter-Timer trigger mode
- Supports event counting source from ACMP0 output, ACMP1 output, ACMP2 output or internal USB SOF signal
- Supports Timer0~3 time-out interrupts signal (TIF) to trigger Touch-Key scan.
- Supports internal capture triggered source from ACMP0 output, ACMP1 output, ACMP2 output, HXT, HIRC, LIRC, MIRC and LXT.

6.11.2.2 PWM Function Features

- Supports PWM generator with two selectable output channels
- Supports 16-bit PWM counter
  - Up count operation type
  - One-shot or auto-reload counter operation mode
- Supports 8-bit prescale from 1 to 256

- Supports 16-bit compare register and period register and double buffer for period register and compare register
- Supports tri-state enable and polarity control for each PWM selectable output channels
- Supports interrupt on the following events:
  - PWM period point, up-count compared point events
- Supports wake-up when interrupt occurs when clock source is LXT or LIRC
- PWM can generate output in Power-down mode
- Supports trigger EADC, LPADC, PDMA, and DAC on the following events:
  - PWM period point and up-count compared point events
- Supports PWM output accumulator event to trigger PDMA transfer and EADC
- Supports PWM output accumulator event to stop PWM counting

6.11.3 Block Diagram

The timer controller block diagram and clock control are shown as follows.

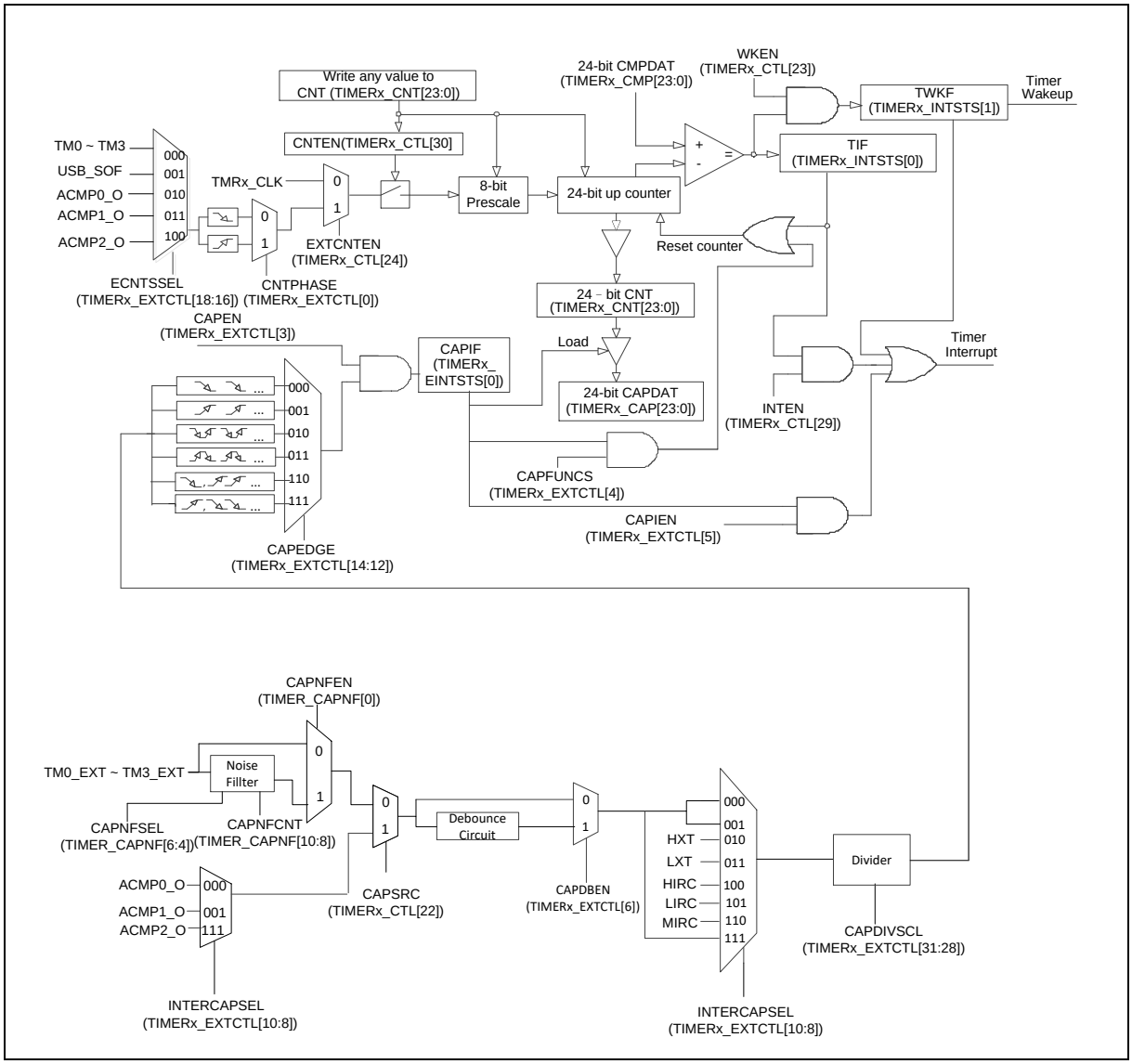


Figure 6.11-1 Timer Controller Block Diagram

Set FUNCSEL (TIMERx_CTL[15]) 0 to enable timer mode. The clock source of Timer0 ~ Timer3 in timer mode can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]) and selected as different frequency in TMR0SEL (CLK_CLKSEL1[10:8]) for Timer0, TMR1SEL (CLK_CLKSEL1[14:12]) for Timer1, TMR2SEL (CLK_CLKSEL1[18:16]) for Timer2 and TMR3SEL (CLK_CLKSEL1[22:20]) for Timer3.

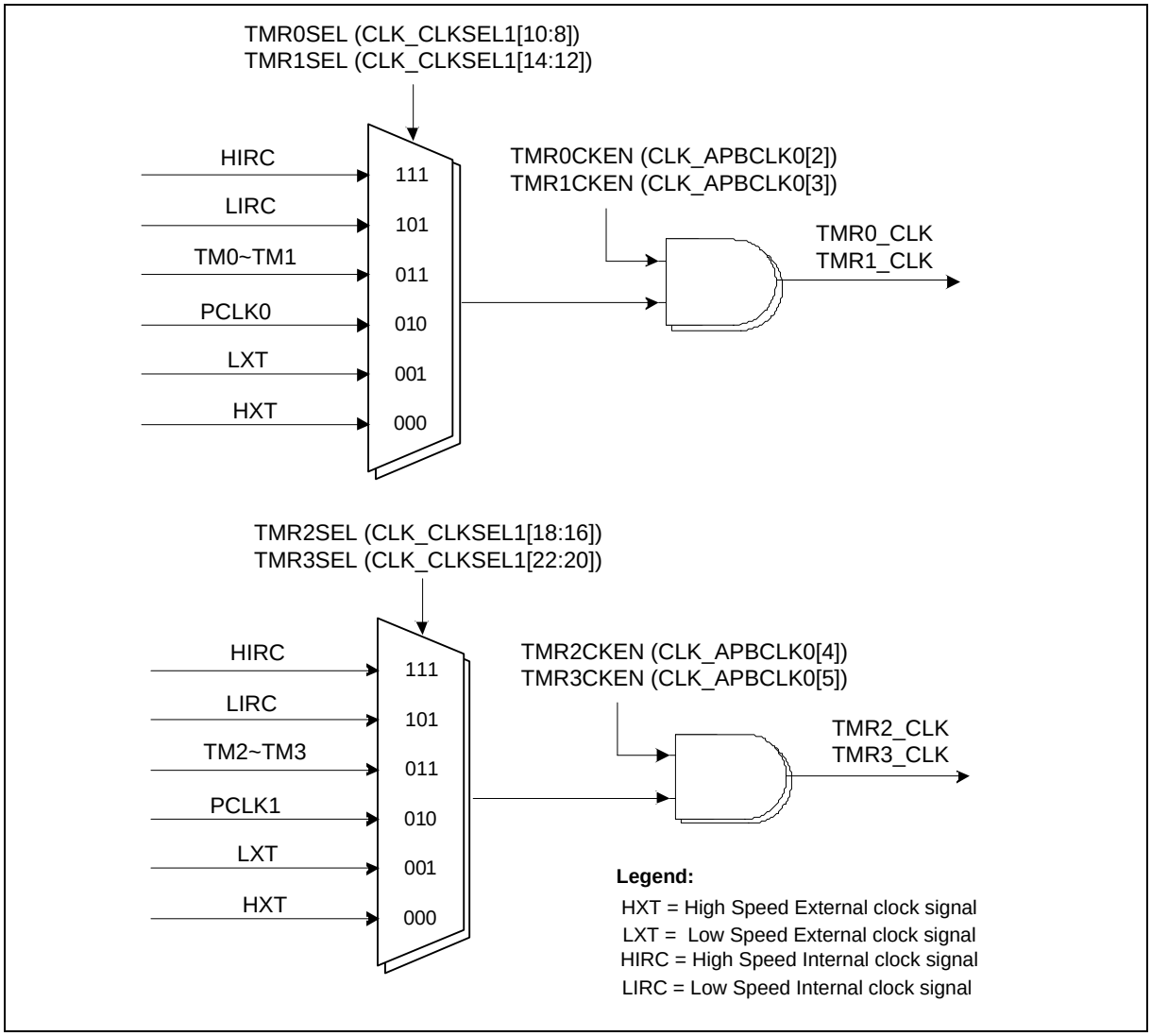


Figure 6.11-2 Clock Source of Timer Controller

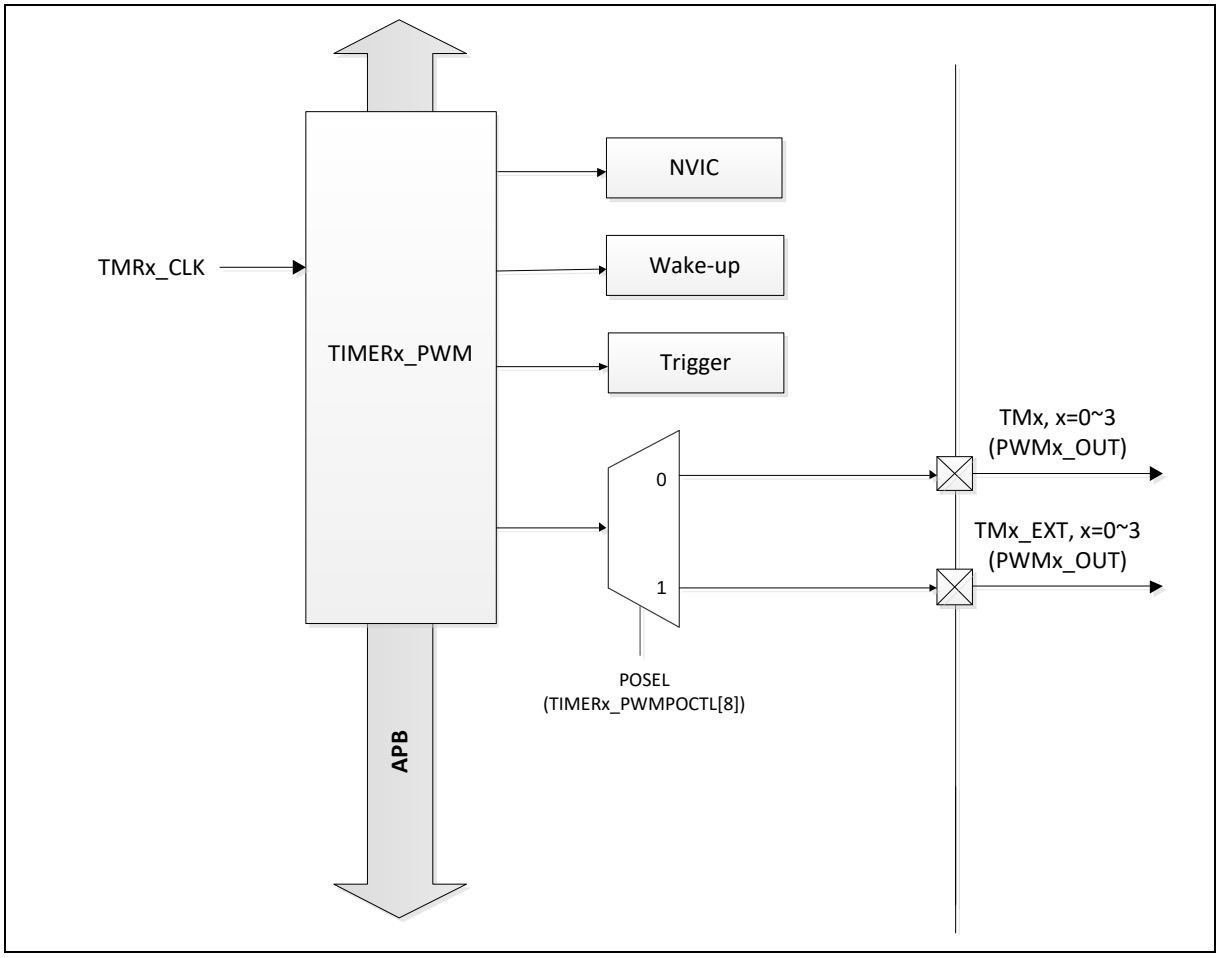


Figure 6.11-3 PWM Generator Overview Block Diagram

Set FUNCSEL (TIMERx_CTL[15]) 1 to enable PWM mode. The clock source of Timer0 ~ Timer3 in PWM mode can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]). PWM system clock and counter clock source are from TMRx_CLK.

Figure 6.12-4 illustrates the PWM architecture that supports one PWM output and two selectable TMx or TMx_EXT output channels in each PWM generator.

When PERIOD (TIMERx_PWMPERIOD[15:0]) is equal to CMP (TIMERx_PWMCMPDAT[15:0]), relative events will be generated. These events are passed to corresponding generators to generate PWM pulse (Pulse Generator), interrupt signal (Interrupt Generator), wake-up (Wake-up Generator) and trigger signal (Trigger Generator) for EADC, LPADC, PDMA and DAC to start conversion. Output Control block is used to decide PWM pulse output.

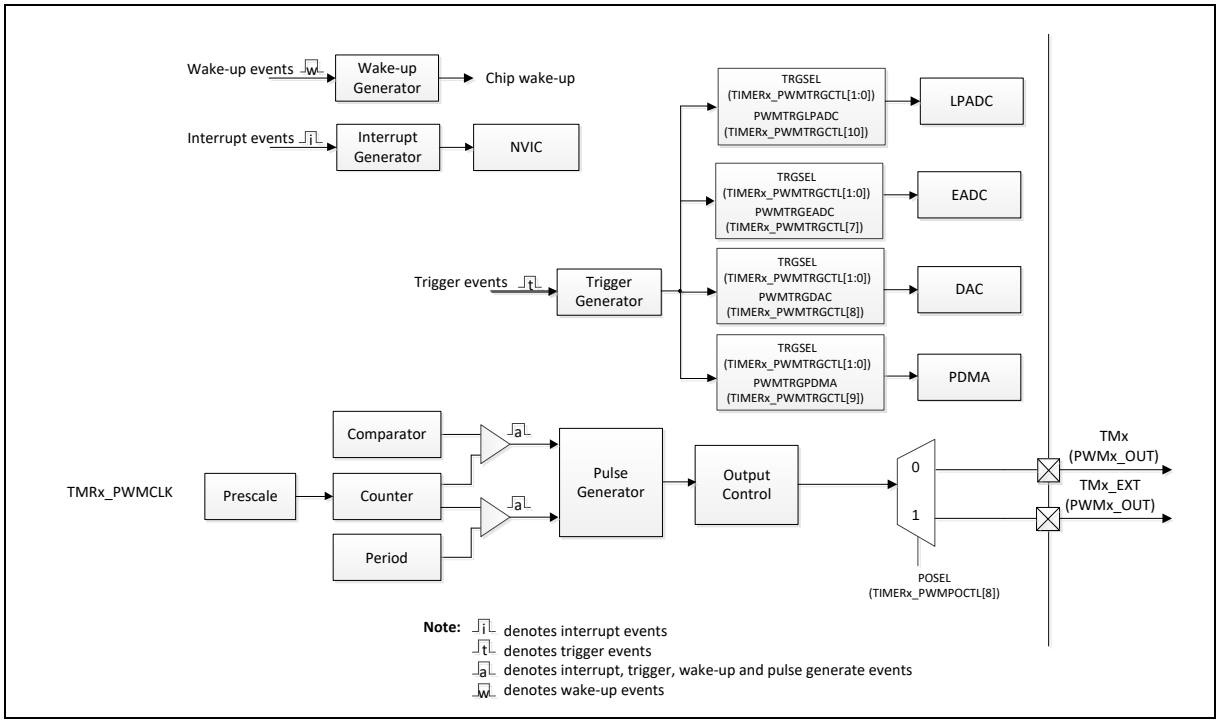


Figure 6.11-4 PWM Architecture Diagram

6.11.4 Basic Configuration

Set FUNCSEL (TIMERx_CTL[15]) 0 to enable timer mode. The clock source of Timer0 ~ Timer3 in timer mode can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]) and selected as different frequency in TMR0SEL (CLK_CLKSEL1[10:8]) for Timer0, TMR1SEL (CLK_CLKSEL1[14:12]) for Timer1, TMR2SEL (CLK_CLKSEL1[18:16]) for Timer2 and TMR3SEL (CLK_CLKSEL1[22:20]) for Timer3.

Set FUNCSEL (TIMERx_CTL[15]) 1 to enable PWM mode. The clock source of Timer0 ~ Timer3 in PWM mode can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]). PWM system clock and counter clock source are from TMRx_CLK.

6.11.4.1 TIMER01 Basic Configurations

- Clock Source Configuration
  - Enable TIMER0 peripheral clock in TMR0CKEN (CLK_APBCLK0[2]).
  - Enable TIMER1 peripheral clock in TMR1CKEN (CLK_APBCLK0[3]).
- Reset Configuration
  - Reset TIMER0 controller in TMR0RST (SYS_IPRST1[2]).
  - Reset TIMER1 controller in TMR1RST (SYS_IPRST1[3]).

6.11.4.2 TIMER23 Basic Configurations

- Clock Source Configuration
  - Enable TIMER2 peripheral clock in TMR2CKEN (CLK_APBCLK0[4]).
  - Enable TIMER3 peripheral clock in TMR3CKEN (CLK_APBCLK0[5]).
- Reset Configuration
  - Reset TIMER2 controller in TMR2RST (SYS_IPRST1[4]).
  - Reset TIMER3 controller in TMR3RST (SYS_IPRST1[5]).

6.11.5 Timer Functional Description

6.11.5.1 Timer Interrupt Flag

The timer controller supports the following interrupt flags: TIF (TIMERx_INTSTS[0]) which is set while timer counter value CNT (TIMERx_CNT[23:0]) matches the timer compared value CMPDAT (TIMERx_CMP[23:0]), and CAPIF (TIMERx_EINTSTS[0]), which is set when the transition on the external TMx_EXT pin, or the internal ACMP0_O, ACMP1_O, ACMP2_O signal and clock source (HXT, LXT, HIRC, LIRC) is associated with CAPEDGE (TIMERx_EXTCTL[14:12]) setting. The TWKF (TIMERx_INTSTS[1]) bit indicates the interrupt wake-up flag status of timer. User can set CAPSRC (TIMERx_CTL[22]) and INTERCAPSEL (TIMERx_EXTCTL[10:8]) to select capture source. Set WKEN (TIMERx_CTL[23] to 1 can use wake-up function.

6.11.5.2 Timer Counting Mode

The timer controller provides four timer counting modes: one-shot, periodic, toggle-output and continuous counting operation modes:

One–shot Mode

If the timer controller is configured at one-shot mode (TIMERx_CTL[28:27] is 00) and CNTEN (TIMERx_CTL[30]) is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF (TIMERx_INTSTS[0]) will be set to 1, CNT value and CNTEN bit is cleared automatically by timer controller then timer counting operation stops. In the meantime, if the INTEN (TIMERx_CTL[29]) is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also.

User can monitor the RSTACT (TIMERx_CNT[31]) to ensure counter reset operation active and disable ICE debug mode acknowledgement effects TIMER counting by setting ICEDEBUG (TIMERx_CTL[31]) to 1.

Periodic Mode

If the timer controller is configured at periodic mode (TIMERx_CTL[28:27] is 01) and CNTEN (TIMERx_CTL[30]) is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF (TIMERx_INTSTS[0]) will be set to 1, CNT value will be cleared automatically by timer controller and timer counter operates counting again. In the meantime, if the INTEN (TIMERx_CTL[29]) bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. In this mode, the timer controller operates counting and compares with CMPDAT value periodically until the CNTEN bit is cleared by user.

User can set PERIOSEL (TIMERx_CTL[20]) to select Timer behavior.

Toggle-Output Mode

If the timer controller is configured at toggle-output mode (TIMERx_CTL[28:27] is 10) and CNTEN (TIMERx_CTL[30]) is set, the timer counter starts up counting. The counting operation of toggle-output mode is almost the same as periodic mode, except toggle-output mode has associated TM0 ~ TM3 or TM0_EXT ~ TM3_EXT pin to output signal while specify TIF (TIMERx_INTSTS[0]) is set. Use can set TGLPINSEL (TIMERx_CTL[21]) to choose TMx or TMx_EXT as toggle-output pin. Thus, the toggle-output signal on TM0 ~ TM3 pin is high and changing back and forth with 50% duty cycle.

Continuous Counting Mode

If the timer controller is configured at continuous counting mode (TIMERx_CTL[28:27] is 11) and CNTEN (TIMERx_CTL[30]) is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF (TIMERx_INTSTS[0]) will be set to 1 and CNT value keeps up counting. In the meantime, if the INTEN (TIMERx_CTL[29]) is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. User can change different CMPDAT value immediately without disabling timer counting and restarting timer counting in this mode.

For example, CMPDAT value is set as 80, first. The TIF will set to 1 when CNT value is equal to 80, timer counter is kept counting and CNT value will not goes back to 0, it continues to count 81, 82, 83, … to (2^{24} -1), 0, 1, 2, 3, … to (2^{24} -1) again and again. Next, if user programs CMPDAT value as 200 and clears TIF, the TIF will set to 1 again when CNT value reaches to 200. At last, user programs CMPDAT as 500 and clears TIF, the TIF will set to 1 again when CNT value reaches to 500.

In this mode, the timer counting is continuous. So this operation mode is called as continuous counting mode.

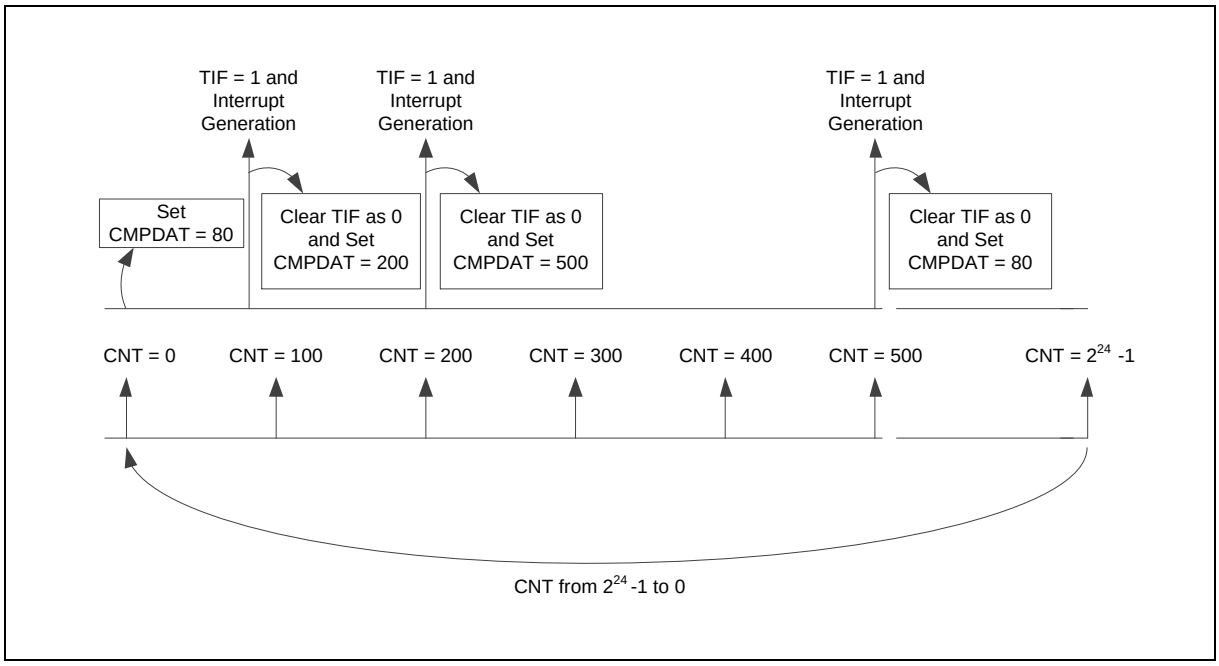


Figure 6.11-5 Continuous Counting Mode

6.11.5.3 Event Counting Mode

The timer controller also provides an application which can count the input event from TMx (x= 0~3) pin and the number of event will reflect to CNT (TIMERx_CNT[23:0]) value. It is also called as event counting function. In this function, EXTCNTEN (TIMERx_CTL[24]) should be set and the timer peripheral clock source should be set as ECLK.

If ECNTSSEL (TIMERx_EXTCTL[18:16]) is 0, the event counter source is from external TMx pin. User can enable or disable TMx pin de-bounce circuit by setting CNTDBEN (TIMERx_EXTCTL[7]). The input event frequency should be less than 1/3 ECLK if TMx pin de-bounce disabled or less than 1/8 ECLK if TMx pin de-bounce enabled to assure the returned CNT value is correct, and user can also select edge detection phase of TMx pin by setting CNTPHASE (TIMERx_EXTCTL[0]) bit.

In event counting mode, the timer counting operation mode can be selected as one-shot, periodic and continuous counting mode to counts the counter value CNT (TIMERx_CNT[23:0]) for TMx. .

If ECNTSSEL (TIMERx_EXTCTL[18:16]) is 1, the event counter source will generate by USB device detect the start-of-frame (SOF) packet. Please refer USB device specifications.

6.11.5.4 Capture Mode

The event capture function is used to load CNT (TIMERx_CNT[23:0]) value to CAPDAT (TIMERx_CAP[23:0]) value while edge transition detected on TMx_EXT (x= 0~3) pin, ACMP, internal clock and external clock. In this mode, CAPFUNCS (TIMERx_EXTCTL[4]) should be as 0 to trigger event capture function and the timer peripheral clock source should be set as ECLK.

If CAPSRC (TIMERx_CTL[22]) is 0, the capture event is triggered by TMx_EXT pin transition. The TMx_EXT signals will be filtered by a 3-bit noise filter. User can enable the noise filter function by CAPNFEN (TIMERx_CAPNF[0]) bit, and noise filter sampling clock can be selected by setting CAPNFSEL (TIMERx_CAPNF[6:4]) bits to fit different noise properties. Moreover, by setting the CAPNFCNT (TIMERx_CAPNF[10:8]) bits, user can define by how many sampling clock cycles a filter will recognize the effective edge of the TMx_EXT signal.

User can also enable or disable TMx_EXT pin de-bounce circuit by setting CAPDBEN (TIMERx_EXTCTL[6]). The transition frequency of TMx_EXT pin should be less than 1/3 ECLK if TMx_EXT pin de-bounce disabled or less than 1/8 ECLK if TMx_EXT pin de-bounce enabled to assure the capture function can be work normally, and user can also select edge transition detection of TMx_EXT pin by setting CAPEDGE (TIMERx_EXTCTL[14:12]).

In event capture mode, user does not consider what timer counting operation mode is selected, the capture event occurred only if edge transition on TMx_EXT pin is detected.

User can enable CAPIEN (TIMERx_EXTCTL[5]) to use capture interrupt function. When the TMx_EXT edge transition meets setting, CAPIF is high.

User must consider the Timer will keep register TIMERx_CAP unchanged and drop the new capture value, if the CPU does not clear the CAPIF status.

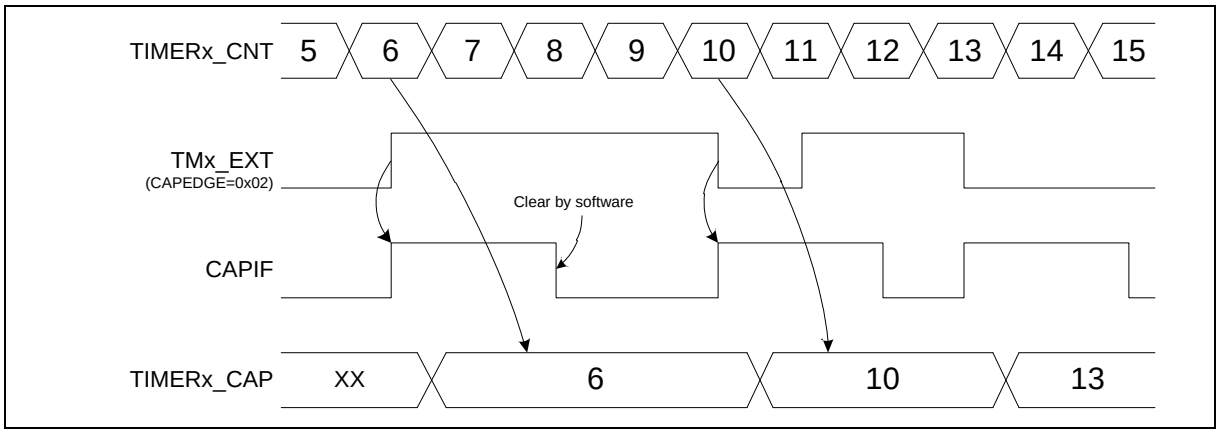


Figure 6.11-6 External Capture Mode

If CAPSRC (TIMERx_CTL[22]) is 1, set INTERCAPSEL (TIMERx_EXTCTL[10:8]) to choose different capture source. The capture event can be triggered by internal output signal transition on ACMP0 if INTERCAPSEL (TIMERx_EXTCTL[10:8]) is 000, or ACMP1 if INTERCAPSEL is 001 ; Other capture sources are HXT, LXT, MIRC, LIRC, HIRC if INTERCAPSEL is 010, 011, 110, 101, 100 respectively. User can switch capture source only when original source and target source is all off. For example, if user wants to switch LXT or LIRC, both of capture source should be set off first.

The capture source can be divided by capture divider. Use can set CAPDIVSCL (TIMERx_EXTCTL[31:28]) to select different divider number. Be aware that timer clock frequency must be four times than capture source at least.

6.11.5.5 Reset Counter Mode

The timer controller also provides reset counter function to reset CNT (TIMERx_CNT[23:0]) value while capture event is generated. In this mode, CAPFUNCS (TIMERx_EXTCTL[4]) should be 1.User must set CAPSRC and INTERCAPSEL to select TMx_EXT transition, internal ACMPx output signal and internal clock or external clock to trigger reset counter value.

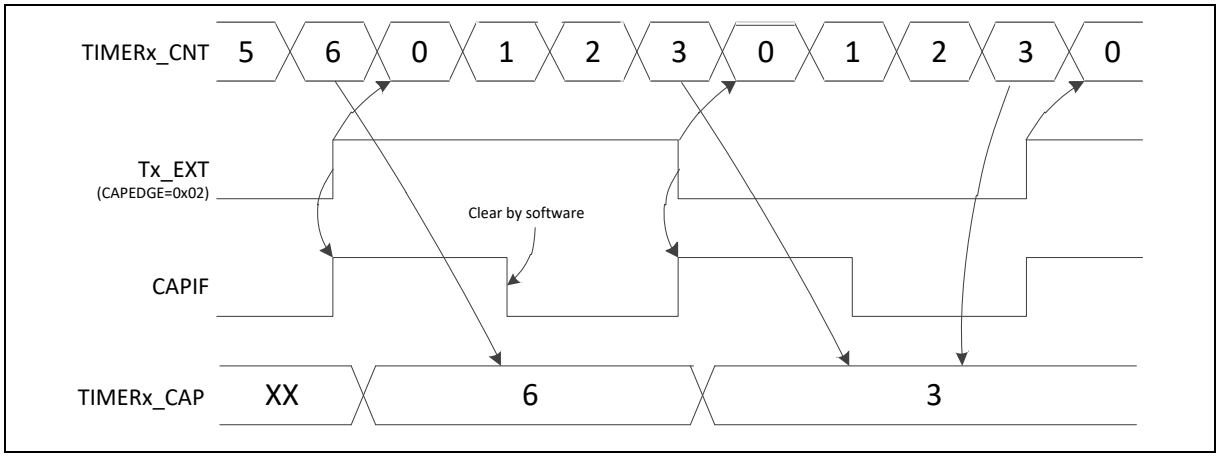


Figure 6.11-7 Reset Counter Mode

6.11.5.6 Timer Trigger Function

The timer controller provides timer time-out interrupt or capture interrupt to trigger PWM, EPWM, EADC, LPADC, DAC, TK and PDMA. If TRGSSEL (TIMERx_TRGCTL[0]) is 0, time-out interrupt signal is used to trigger PWM, EPWM, EADC, LPADC, DAC, TK and PDMA. If TRGSSEL (TIMERx_TRGCTL[0]) is 1, capture interrupt signal is used to trigger PWM, EPWM, EADC, LPADC, DAC and PDMA.

When the TRGPWM (TIMERx_TRGCTL[1]) is set, if the timer interrupt signal is generated, the timer controller will generate a trigger pulse as PWM/EPWM external clock source.

When the TRGEADC (TIMERx_TRGCTL[2]) is set, if the timer interrupt signal is generated, the timer controller will trigger EADC to start converter.

When the TRGDAC (TIMERx_TRGCTL[3]) is set, if the timer interrupt signal is generated, the timer controller will trigger DAC to start converter.

When the TRGPDMA (TIMERx_TRGCTL[4]) is set, if the timer interrupt signal is generated, the timer controller will trigger PDMA.

When the TRGLPADC (TIMERx_TRGCTL[5]) is set, if the timer interrupt signal is generated, the timer controller will trigger LPADC.

When the TRGTK (TIMERx_TRGCTL[8]) is set, if the timer time-out interrupt signal is generated, the timer controller will trigger TK.

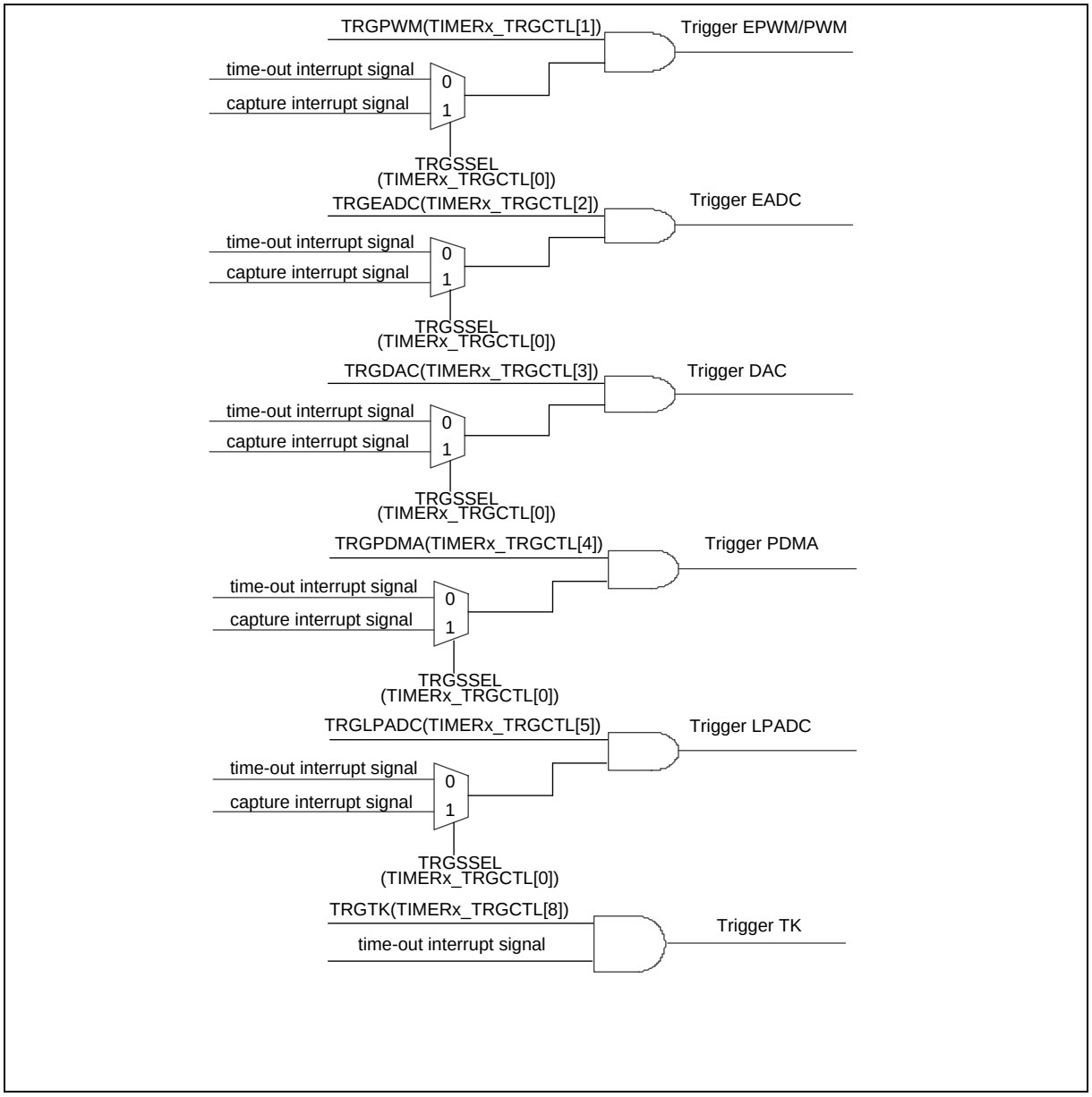


Figure 6.11-8 Internal Timer Trigger

When both the TRGPDMA (TIMERx_TRGCTL[4]) and TRGSSEL (TIMERx_TRGCTL[0]) are 1, TIMERx_CAP will change when encountering timer capture edge no matter whether CPU clear CAPIF status or not. In Trigger EADC, LPADC, DAC or PWM function, user must consider that the Timer will keep register TIMERx_CAP unchanged if the CPU does not clear the CAPIF status.

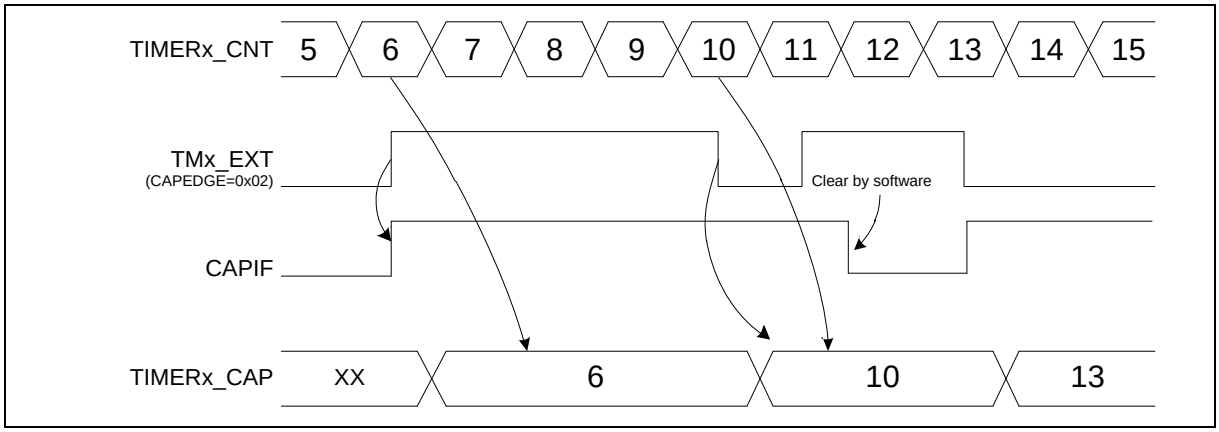


Figure 6.11-9 Capture interrupt trigger PDMA

6.11.5.7 Inter-Timer Trigger Capture Mode

In this mode, the Timer0/2 will be forced in event counting mode, counting with external event, and will generate an internal signal (INTR_TMR_TRG) to trigger Timer1/3 start or stop counting. Also, the Timer1/3 will be forced in capture mode and start/stop trigger-counting by Timer0/2 counter status.

Setting Timer0 Inter-timer Trigger Capture enabled, trigger-counting capture function is forced on Timer1. Setting Timer2 Inter-Timer Trigger Capture enabled, trigger-counting capture function is forced on Timer3.

Start Trigger

While INTRGEN (TIMERx_CTL[19]) in Timer0/2 is set, the Timer0/2 will make a rising-edge transition of INTR_TMR_TRG while Timer0/2 24-bit counter value (CNT) is counting from 0x0 to 0x1 and Timer1/3 counter will start counting immediately and automatically.

Stop Trigger

When Timer0/2 CNT reaches the Timer0/2 CMPDAT(TIMERx_CMP[23:0]) value, the Timer0/2 will make a falling-edge transition of INTR_TMR_TRG. Then Timer0/2 counter mode function will be disabled and INTRGEN (TIMERx_CTL[19]) will be cleared by hardware then Timer1/3 will stop counting also. At the same time, the Timer1/3 CNT value will be saved into Timer1/3 CAPDAT (TIMERx_CAP[23:0]).

User can use inter-timer trigger mode to measure the period of external event (TMx) more precisely. Figure 6.11-10. shows the sample flow of Inter-Timer Trigger Capture Mode for Timer0 as event counting mode and Timer1 as trigger-counting capture mode.

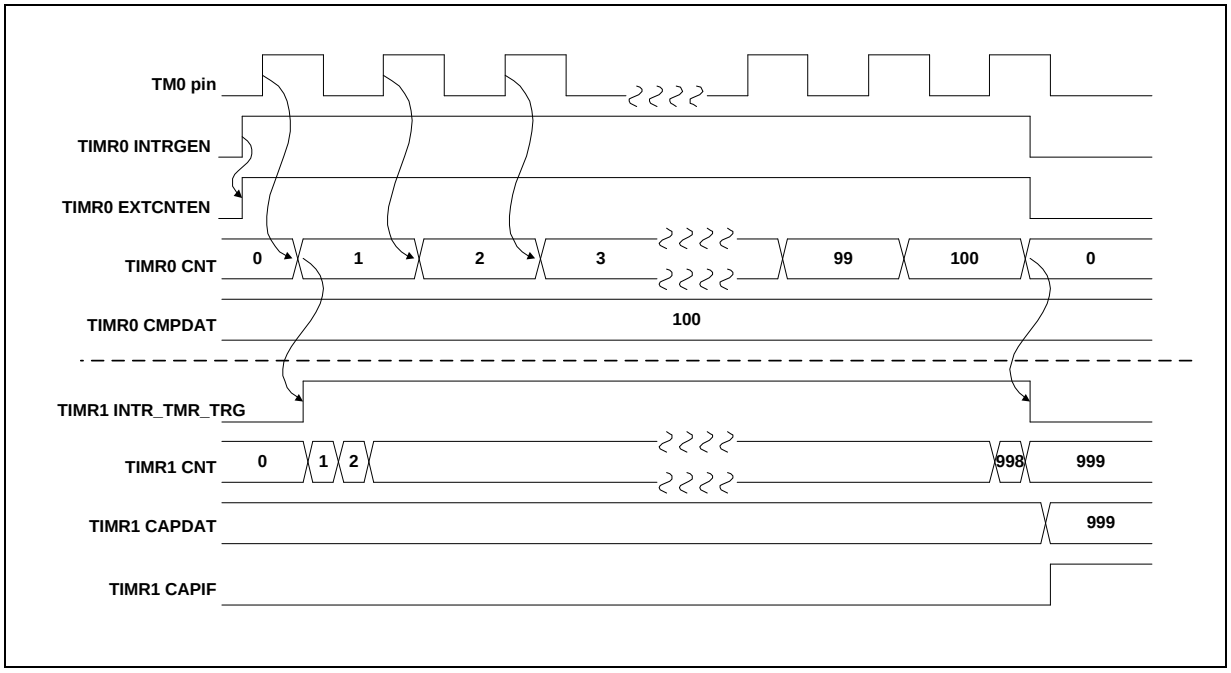


Figure 6.11-10 Inter-Timer Trigger Capture Timing

User must clear Timer1/3 CAPIF if user wants to use second inter-timer trigger function again.

6.11.6 PWM Functional Description

6.11.6.1 PWM Block Diagram

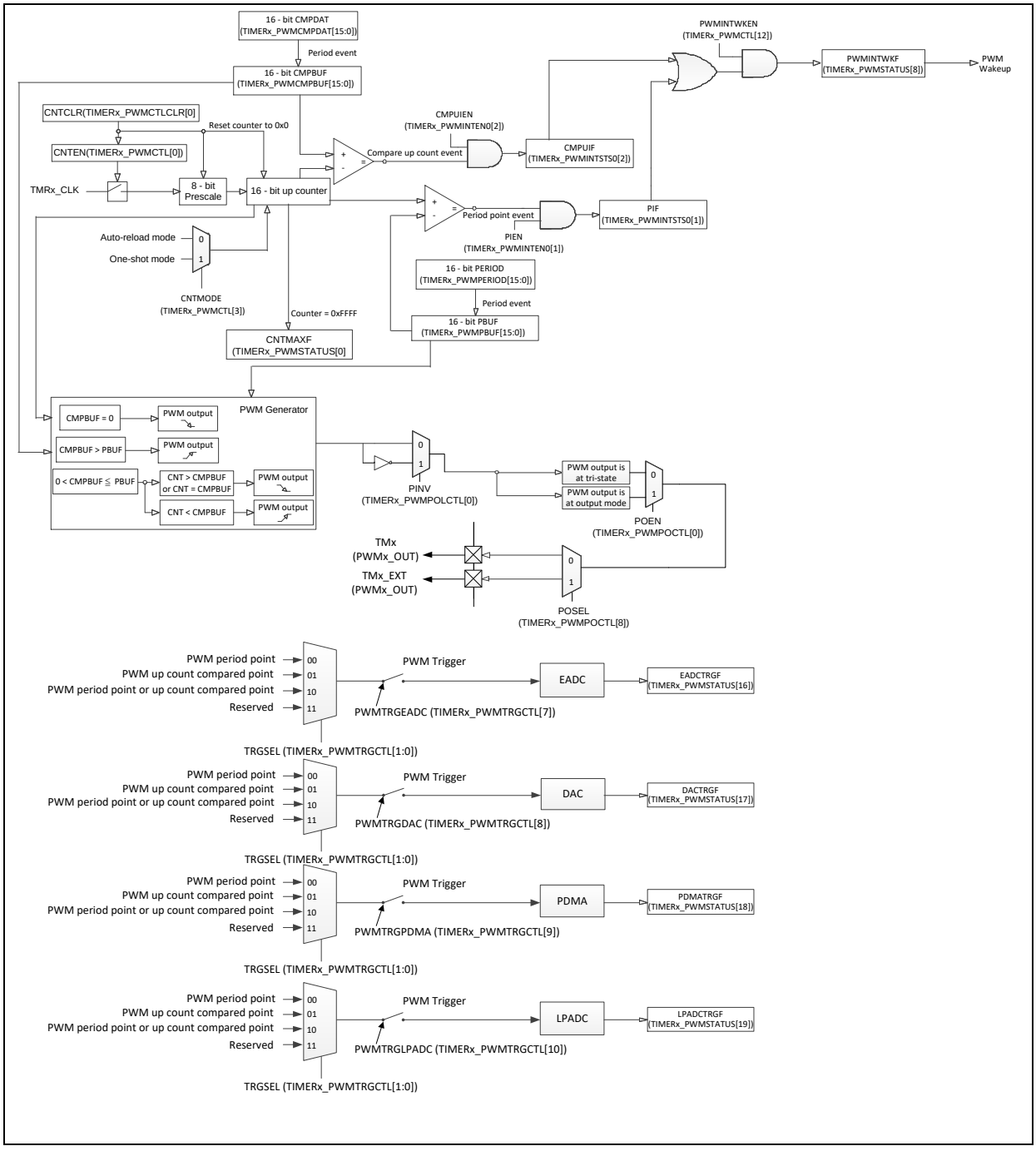


Figure 6.11-11 Timer PWM Block Diagram

6.11.6.2 PWM Prescale

The PWM prescale is used to divide clock source, and the clock of PWM counter is divided by (CLKPSC+ 1). The prescale is set by CLKPSC (TIMERx_PWMCLKPSC[7:0]).Figure 6.11-12. shows an example of PWM prescale waveform in up count type.

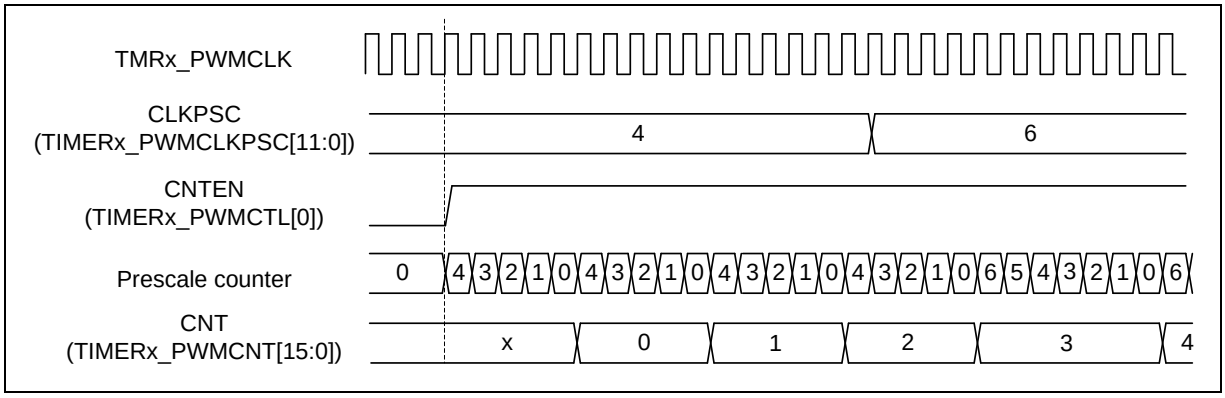


Figure 6.11-12 PWM Prescale Waveform in Up Count Type

6.11.6.3 PWM Counter

The PWM supports up count type

It starts up-counting from 0 to PERIOD (TIMERx_PWMPERIOD[15:0]). The current counter value can be read from the CNT (TIMERx_PWMCNT[15:0]). PWM generates a period point event when the counter counts to PERIOD and prescale counts to 0. Figure 6.11-13. shows an example of PWM up count type, where PWM period time is (PERIOD+1) * (CLKPSC+1) * TMRx_PWMCLK.

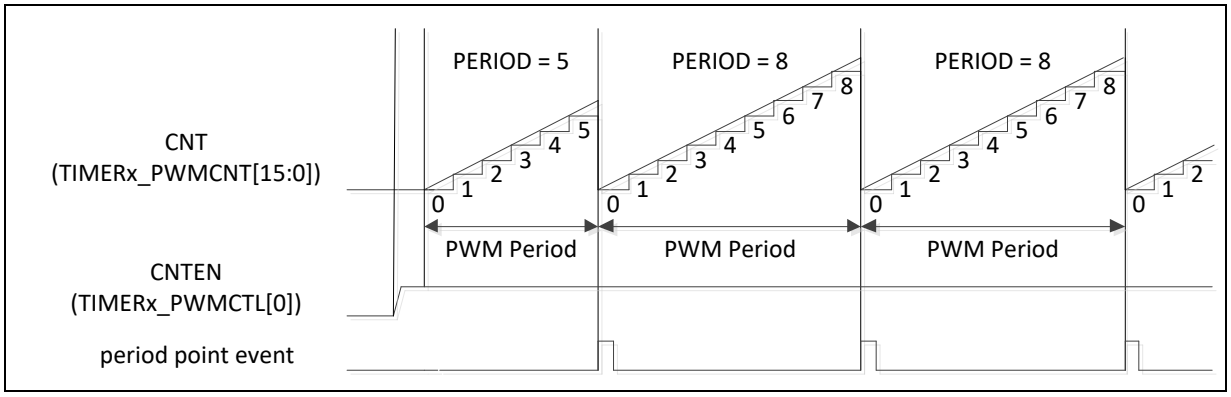


Figure 6.11-13 PWM Up Count Type

6.11.6.4 PWM Counter Operation mode

The PWM counter supports two operation modes: one-shot mode and auto-reload mode. PWM counter will operate in one-shot mode if CNTMODE (TIMERx_PWMCTL[3]) bit is set to 1, and operate in auto-reload mode if CNTMODE bit is set to 0.

In both modes, CMP (TIMERx_PWMCMPDAT[15:0]) and PERIOD (TIMERx_PWMPERIOD[15:0]) should be written first and then set CNTEN (TIMERx_PWMCTL[0]) bit to 1 to start counter running.

In one-shot mode, PWM counter value will reload to 0 after one PWM period is completed. User can write CMP to continuous one-shot operation to generate next one-shot pulse once no matter current one-shot counter is running or completed.

For example, user writes 0x10 to CMP, writes 0x3F to PERIOD and set CMPUIEN (TIMERx_PWMINTEN0[2]) and PIEN (TIMERx_PWMINTEN0[1]) to 1. When counter (CNT) counts to 0x10, CMPUIF (TIMERx_PWMINTSTS0[2]) is set to 1. Counter keeps counting until 0x3F, PIF (TIMERx_PWMINTSTS0[1]) is set to 1 then counter reload to 0, and counter stops counting.

Another example is user writes 0x10 to CMP, writes 0x3F to PERIOD and set CMPUIEN and PIEN to 1. If user writes 0x25 to CMP when counter counting, this value will load to CMPBUF (TIMERx_PWMCMPBUF[15:0]) when counter counts to 0x3F and PIF is set to 1, then counter counts

from 0. If user clears CMPUIF and PIF before counter counts to 0x25, CMPUIF and PIF will be set to 1 when counter counts to 0x25 and 0x3F, respectively.

In auto-reload mode, PWM counter is continuous running with current active PERIOD and CMP. If user sets PERIOD to zero in auto-reload mode, PWM counter value will reload to 0 after one PWM period is completed.

For example, user writes 0x10 to CMP, writes 0x3F to PERIOD. CMPUIEN and PIEN are set to 1. When counter counts to 0x10, CMPUIF is set to 1. Counter keeps counting until 0x3F, PIF is set to 1, and counter will restart counting from 0. If user clear CMPUIF and PIF before counter counts to 0x10 (2nd counts), CMPUIF will equal to 1 and PIF will equal to 1 when counter counts to 0x10 and 0x3F, respectively.

When user writes 0 to PERIOD and this value will be loaded to PBUF (TIMERx_PWMPBUF[15:0]). This action will make counter be set to 0 and stops counting until user rewrites another value (except 0) to PERIOD and this value loads to PBUF, counter counts from 0.

The one-shot and auto-reload mode comparison is shown as Table 6.12-4.

Counter Mode Functions	One-Shot	Auto-Reload
CNTMODE (TIMERx_PWMCTL[3])	1	0
CNT = PERIOD (PERIOD is not 0)	Stops counting	Keeps counting from 0 to PERIOD (Periodic)
PERIOD = 0 and loads to PBUF, then write another value (Except 0) to PERIOD	CNT stops counting even though user writes new PERIOD	CNT = 0 (reset value), CNT stops counting until PERIOD and PBUF is not equal to 0

Table 6.11-4 One-shot and Auto-reload Functions

Note: PERIOD (TIMERx_PWMPERIOD[15:0]) and CMP (TIMERx_PWMCMPDAT[15:0]) will be loaded to PBUF (TIMERx_PWMPBUF[15:0]) and CMPBUF (TIMERx_PWMCMPBUF[15:0]), respectively when one PWM period completes. The first CMP and PERIOD will be loaded to CMPBUF and PBUF, respectively, after CNTEN (TIMERx_PWMCTL[0]) is 1.

6.11.6.5 PWM Comparator

The CMP (TIMERx_PWMCMPDAT[15:0]) is comparator register of PWM. The CMP value is continuously compared to the corresponding counter value. When the counter is equal to CMP, PWM generates a compared point event. This event will generate PWM output pulse, interrupt signal wake-up or trigger EADC, LPADC, PDMA, DAC to start conversion.

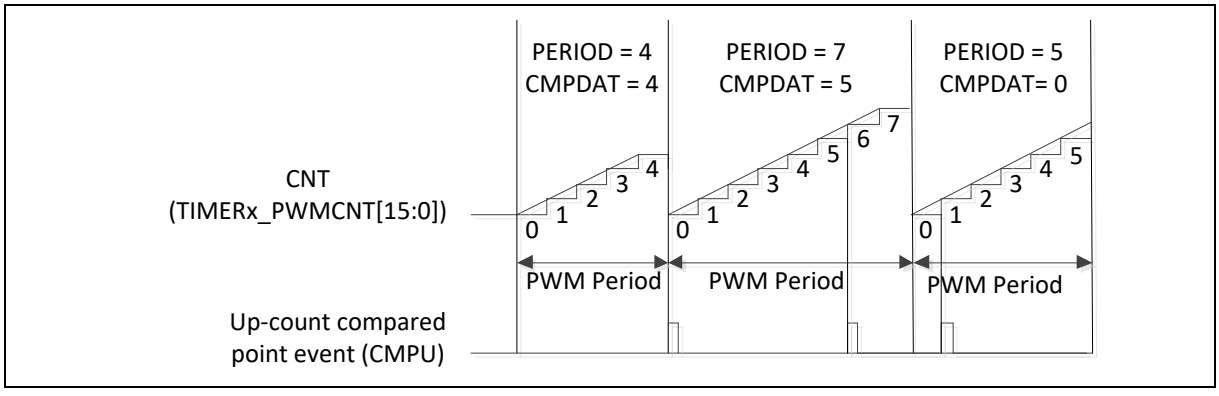


Figure 6.11-14 PWM Comparator Events in Up Count Type

6.11.6.6 Period Loading Mode

The PWM provides PBUF (TIMERx_PWMPBUF[15:0]) is the active PERIOD buffer register and CMPBUF (TIMERx_PWMCMPBUF[15:0]) is the active CMP buffer register. In period loading mode, both PERIOD (TIMERx_PWMPERIOD[15:0]) and CMP (TIMERx_PWMCMPDAT[15:0]) will load to their active PBUF and CMPBU register while each PWM period is completed.Figure 6.11-15. shows period loading timing of up count type, where PERIOD DATA0 denotes the initial data of PERIOD, PERIOD DATA1 denotes the first updated PERIOD data by user and so on, CMP also follows this rule. The following steps are the sequence of Figure 6.11-15.

1. User writes CMP DATA1 to CMP at point 1.
2. Period loading CMP DATA1 to CMPBUF at the end of PWM period at point 2.
3. User writes PERIOD DATA1 to PERIOD at point 3.
4. Period loading PERIOD DATA1 to PBUF at the end of PWM period at point 4.
5. User writes PERIOD DATA2 to PERIOD at point 5.
6. Period loading PERIOD DATA2 to PBUF at the end of PWM period at point 6.

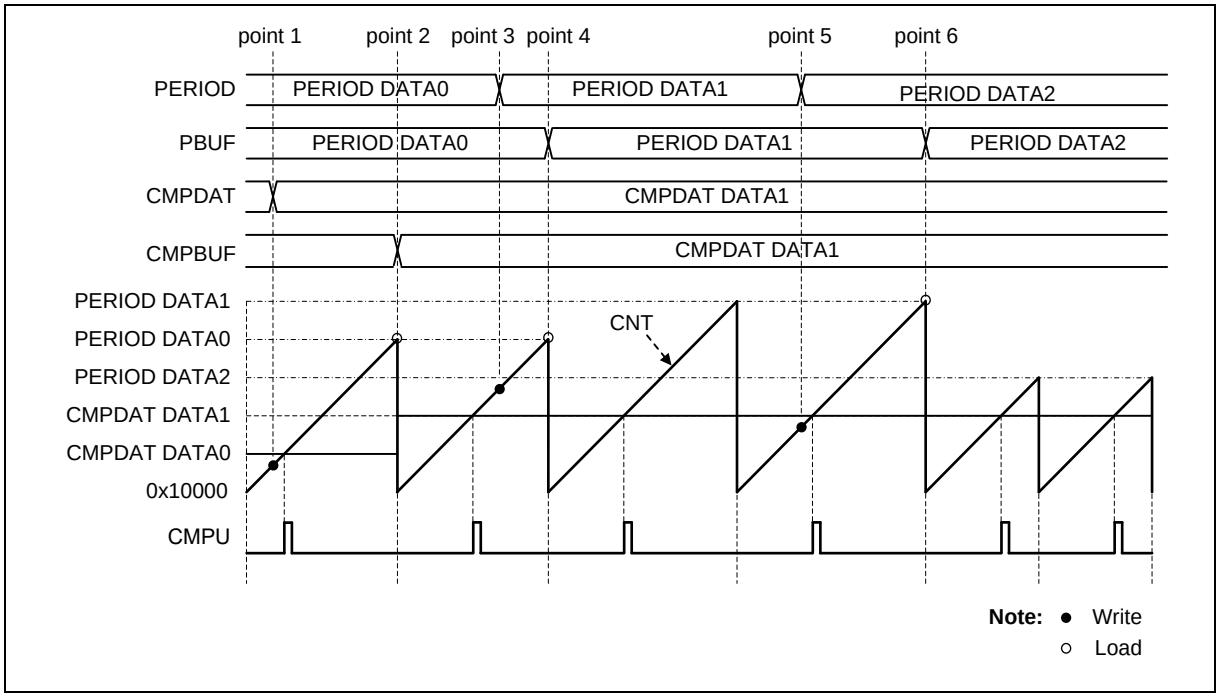


Figure 6.11-15 Period Loading Mode with Up Count Type

6.11.6.7 PWM Pulse Generator

PWM pulse generator uses counter and comparator events to generate PWM output pulse. The events are zero point in up count type and counter equal to comparator point in up count type.

The event point can generate PWM output waveform in up count type as shown in Figure 6.11-16.

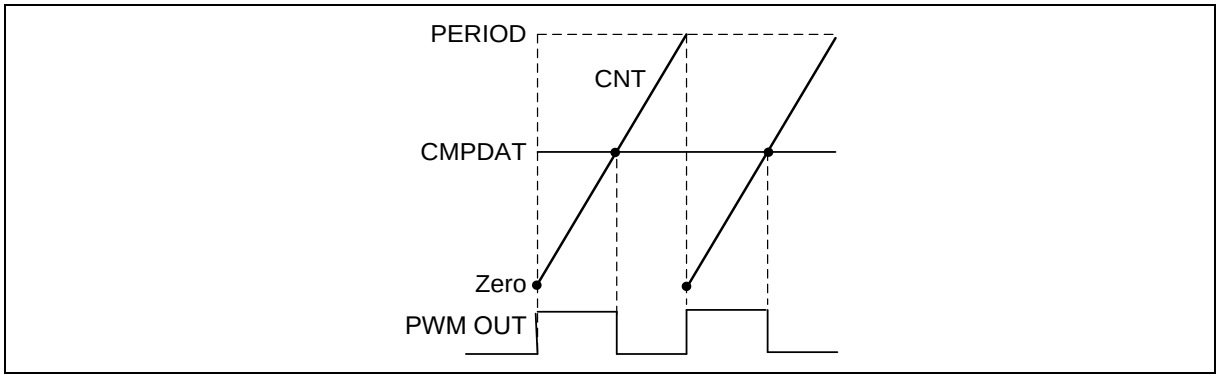


Figure 6.11-16 PWM Pulse Generation in Up Count Type

The PWM output pulse is associated with CMPBUF (TIMERx_PWMCMPBUF[15:0]) and PBUF (TIMERx_PWMPBUF[15:0]). The rules are listed as below.

1. If CMPBUF is zero, PWM output is 100% low.
2. If CMPBUF > PBUF, PWM output is 100% high.
3. If 0 < CMPBUF ≦ PBUF, PWM output high/low duty is according to counter value (TIMERx_PWMCNT[15:0]).
    1) If PWM counter is higher than CMPBUF or equal to CMPBUF, PWM output is low.
    2) If PWM counter is lower than CMPBUF, PWM output is high.

The PWM output level table is shown as Table 6.11-1

Conditions	CMPBUF = 0	CMPBUF > PBUF	0 < CMPBUF ≦ PBUF	
			CNT ≧ CMPBUF	CNT < CMPBUF
PWM Output	Low	High	Low	High

Table 6.11-1 PWM Output Level

Figure 6.11-17. is an example about PWM duty cycle from 0% to 100% in up count type where PERIOD is 4 with different CMP value.

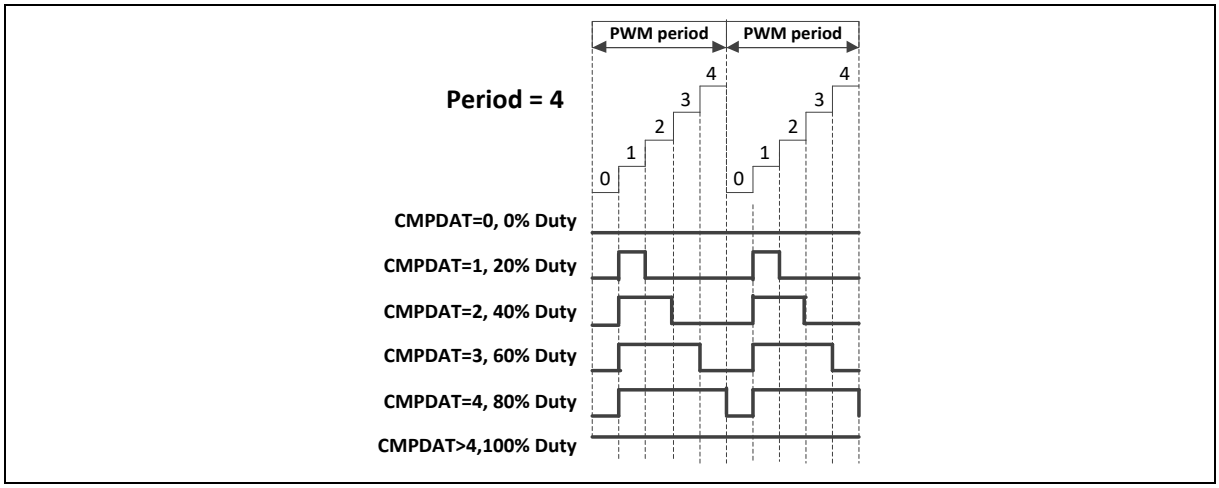


Figure 6.11-17 PWM 0% to 100% Duty Cycle in Up Count Type

6.11.6.8 PWM Output Control

After PWM pulse generator, there are three steps to control output waveform. User can set POEN (TIMERx_PWMPOCTL[0]) 1 to enable PWMx_OUT output waveform and set POSEL (TIMERx_PWMPOCTL[8]) to select TMx or TMx_EXT as PWM output channel.

There are polarity control, output enable control and output channel select to control output waveform as shown inFigure 6.11-18.

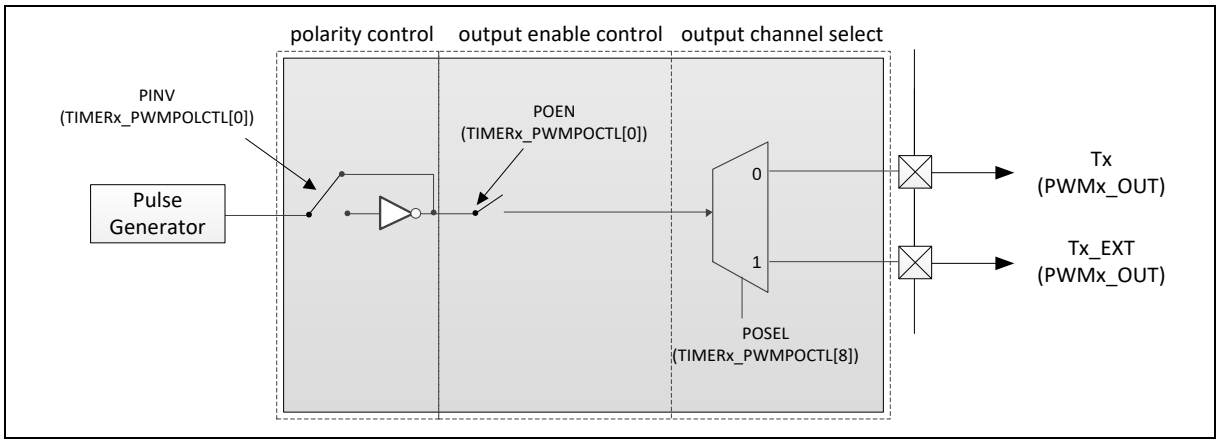


Figure 6.11-18 PWM Output PWMx_OUT Control

6.11.6.9 Polarity Control

Each PWMx_OUT (TMx or TMx_EXT) has an independent polarity control to configure the polarity of the active state of PWM output. User can control polarity state on PINV (TIMERx_PWMPOLCTL[0]). Figure 6.11-19. shows the PWMx_OUT with polarity control.

When PWM output is selected as TMx pin, PINV is 0, TMx_EXT pin is kept as 0. Similarly, TMx_EXT pin is kept as 1 when PINV is 1.

When PWM output is selected as TMx_EXT pin, PINV is 0, TMx pin is kept as 0. Similarly, TMx pin is kept as 1 when PINV is 1.

When PWM output is selected as TMx pin, PINV is 0, but POEN (TIMERx_PWMPOCTL[0]) is set as 0, TMx pin is kept as 0. Similarly, TMx pin is kept as 1 when PINV is 1.

When PWM output is selected as TMx_EXT pin, PINV is 0, but POEN is set as 0, TMx_EXT pin is kept as 0. Similarly, TMx_EXT pin is kept as 1 when PINV is 1.

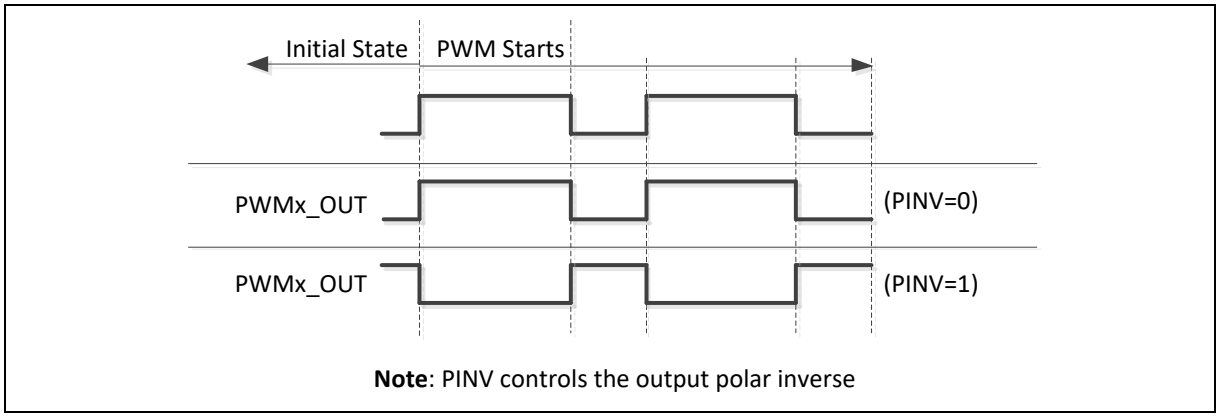


Figure 6.11-19 PWMx_OUT Polarity Control

6.11.6.10 PWM Interrupt Generator

There are independent interrupts for each PWM as shown in Figure 6.12-19.

The PWM interrupt (PWMx_INT) comes from PWM complementary pair events. The counter can generate the period point interrupt flag PIF (TIMERx_PWMINTSTS0[1]). When counter equals to the comparator value stored in CMP (TIMERx_PWMCMPDAT[15:0]) at up-count direction, the comparator up interrupt flag CMPUIF (TIMERx_PWMINTSTS0[2]) is set. If the corresponding interrupt enable bits are set, the interrupt trigger events will also generates interrupt signals.

PWMx_INT can use the TIMERx_PWMIFA register to accumulate the number of times that the interrupt flags have been triggered for each channel. By setting IFAEN (TIMERx_PWMIFA[31]) bit to 1 to enable accumulator, PWMx_INT will switch interrupt source from every event trigger interrupt to trigger interrupt once every accumulate times.

By setting the IFASEL (TIMERx_PWMIFA[29:28]) bits, user can select one of the 4 interrupt sources to accumulate interrupt flag times for each channel, and the number of times interrupt flags will compare with IFACNT (TIMERx_PWMIFA[15:0]) bits. When interrupt accumulator equals IFACNT+1, the IFAIF (TIMERx_PWMAINTSTS[0]) bit will be set as PWMx_INT signal if user enables IFAIEN (TIMERx_PWMAINTEN[0]) bit. Accumulator interrupt of each channel can also be as request source of PDMA. Figure 6.12-19 is an example using TIMERx_PWMIFA register to output PWMx_INT once every IFCNT0+1 times interrupt events occurred.

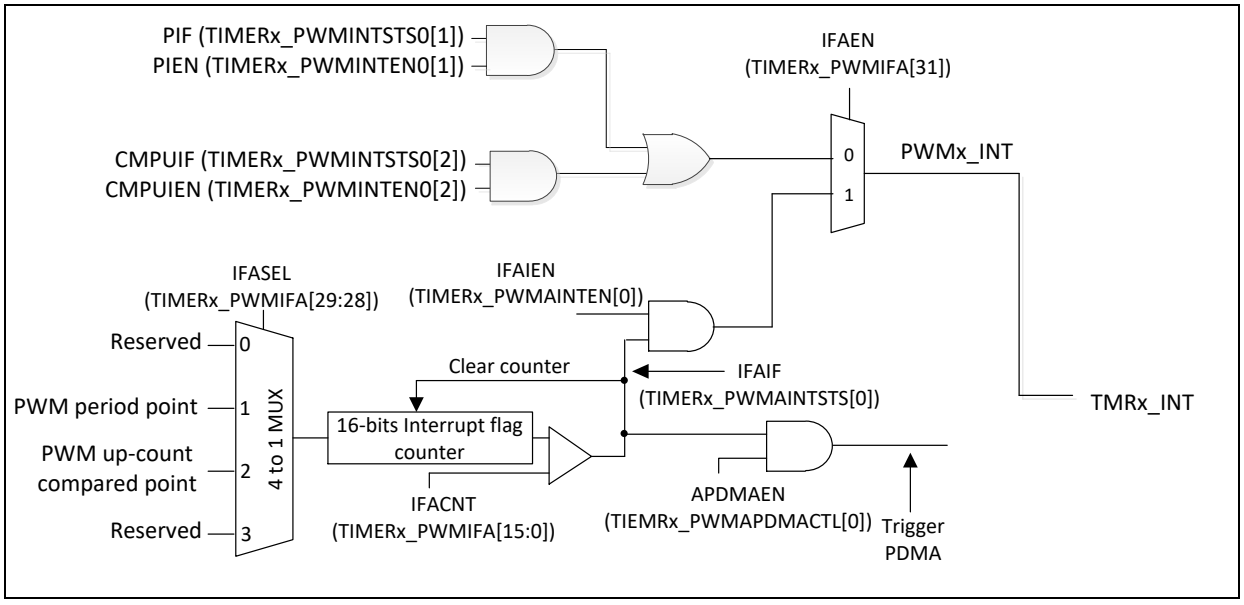


Figure 6.11-20 PWM Interrupt Architecture Diagram

6.11.6.11 PWM Wake-up Generator

User can set PWMINTWKEN (TIMERx_PWMCTL[12]) high to wake up when PWM interrupt occurs. Before enter power-down mode, user must select LXT or LIRC as PWM clock source.

When PWM wake-up occurs, PWMINTWKF (TIMERx_PWMSTATUS[8]) is set to 1. User can write 1 to this bit to clear this flag.

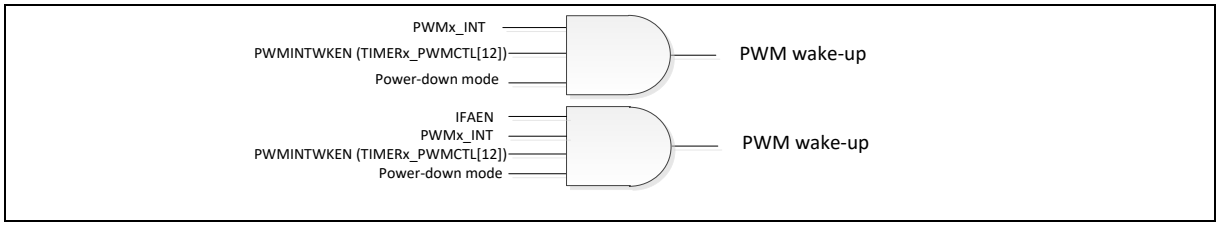


Figure 6.11-21 PWM Wake-up Architecture Diagram

6.11.6.12 PWM Trigger EADC, LPADC, PDMA, DAC Generator

PWM counter event can be one of the EADC, LPADC, PDMA, DAC conversion trigger source. User sets TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select which PWM counter event can trigger conversion.

When the PWMTRGEADC (TIMERx_PWMTRGCTL[7]) is set to 1, the PWM can trigger EADC. When the PWMTRGDAC (TIMERx_PWMTRGCTL[8]) is set to 1, the PWM can trigger DAC. When the PWMTRGPDMA (TIMERx_PWMTRGCTL[9]) is set to 1, the PWM can trigger PDMA. When the PWMTRGLPADC (TIMERx_PWMTRGCTL[10]) is set to 1, the PWM can trigger LPADC.

There are three PWM counter events can be selected as the trigger source to start conversion as shown in Figure 6.11-22.

The PWM interrupt, wake-up and trigger comparison table is shown as Table 6.12-6.

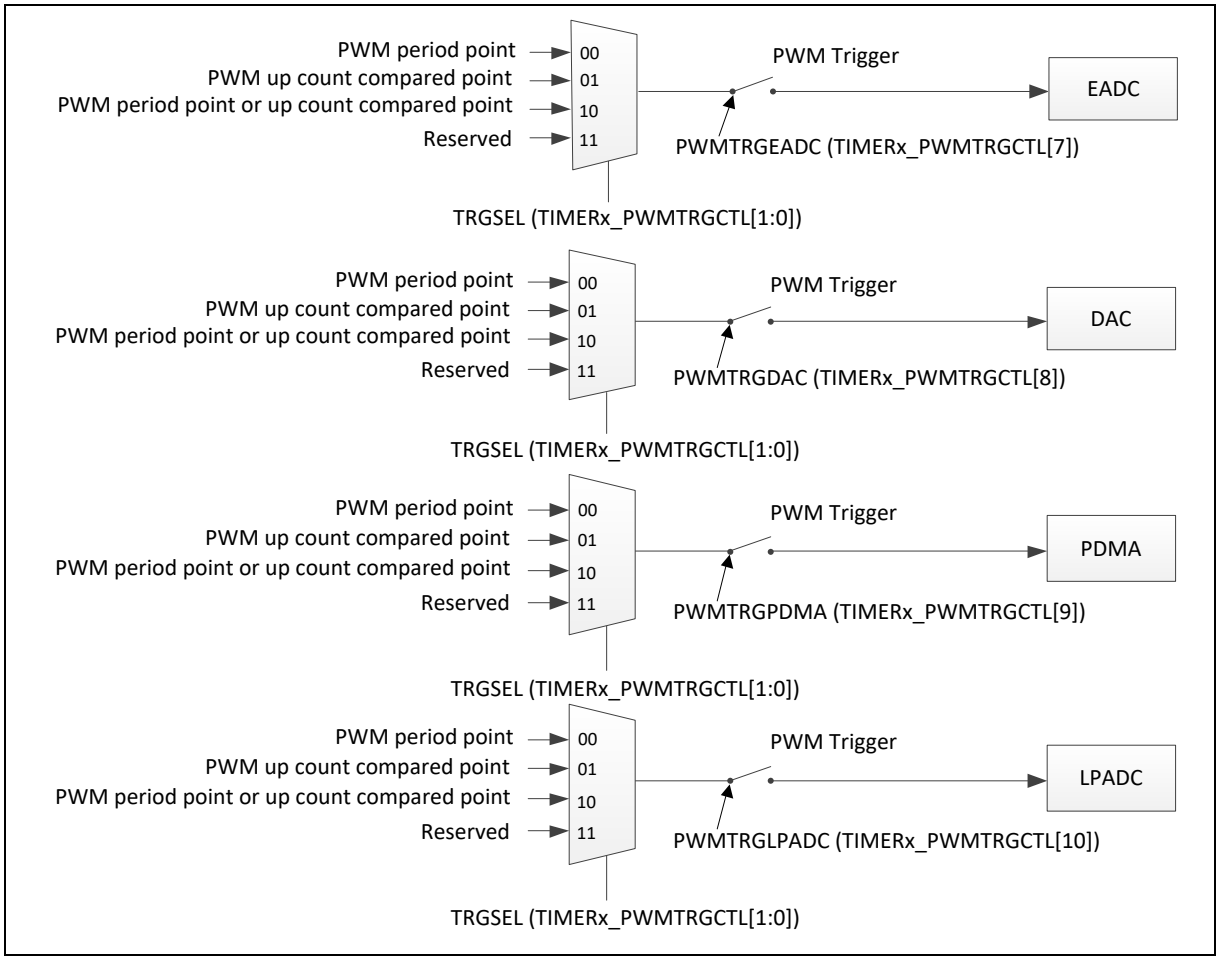


Figure 6.11-22 PWM Trigger Block Diagram

Functions / Events		Interrupt	Wake-Up PWMINTWKEN (TIMERx_PWMCTL[12])		Trigger
			= 0	= 1	
CNT = CMP	CMPUIEN = 1	Yes	No	Yes	EADC, DAC, LPADC PDMA (Note1)
	CMPUIEN = 0	No	No	No	No
CNT = PERIOD	PIEN = 1	Yes	No	Yes	EADC, DAC, LPADC PDMA (Note1)
	PIEN = 0	No	No	No	No

Table 6.11-6 PWM Interrupt, Wake-up and Trigger Events Comparison

Note: Set TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select trigger event and set PWMTRGEADC (TIMERx_PWMTRGCTL[7]), PWMTRGPWM (TIMERx_PWMTRGCTL[8]), PWMTRGPDMA (TIMERx_PWMTRGCTL[9]) , PWMTRGLPADC(TIMERx_PWMTRGCTL[10]) to select which device will be triggered.

6.11.6.13 Mode Transition

If user needs to change timer mode to PWM mode or PWM mode to timer mode, user must do reset before change mode or check the previous mode is end.

The Timer mode to PWM mode transition steps are shown below.

1. Set Timer CNTEN (TIMERx_CTL[30]) as 0
2. Polling ACTSTS (TIMERx_CTL[25]). If ACTSTS is 0, user can set FUNCSEL (TIMERx_CTL[15]) as 1 to enable PWM mode.
3. PWM function settings.

The PWM mode to Timer mode transition steps are shown below.

1. Set PWM CNTEN (TIMERx_PWMCTL[0]) as 0.
2. Set FUNCSEL(TIMERx_CTL[15]) as 0 to disable PWM mode.
3. Polling FUNCSEL(TIMERx_CTL[15]). If FUNCSEL(TIMERx_CTL[15]) is 0, user can set Timer settings.

6.11.6.14 Accumulator PDMA Function

The PWM module supports to trigger the PDMA transfer function when accumulator interrupt happened. When the corresponding PDMA enable bit APDMAEN (TIMERx_PWMAPDMACTL[0]) is set, accumulator module will send a request to PDMA controller when accumulator interrupt has happened, meaning that IFAIF (TIMERx_PWMAINTSTS[0]) is set 1. The PDMA controller will issue an acknowledge to accumulator and auto clear IFAIF (TIMERx_PWMAINTSTS[0]) after it has read memory data and send the data to the particular register (PERIOD, etc.). So, user can use this function to change accumulator interrupt frequency.

6.11.6.15 Accumulator Stop Mode

The PWM module supports the Accumulator Stop Mode to stop counting when accumulator interrupt happened. Figure 6.12-22 shows Accumulator Stop Mode waveform. When the corresponding STPMOD (TIMERx_PWMIFA[24]) is set, accumulator module will disable CNTEN (TIMERx_PWMCTL[0]) after counter finishes whole period.

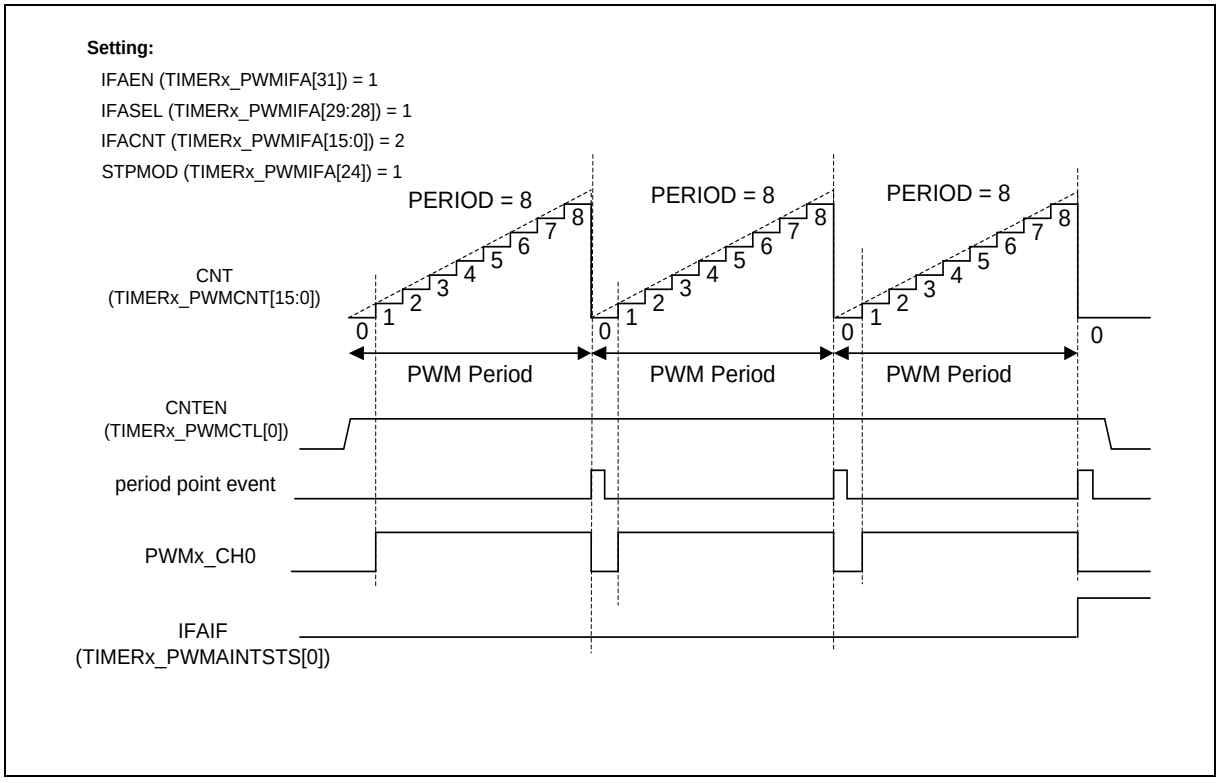


Figure 6.11-23 PWM Accumulator Stop Mode Waveform

6.11.7 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
TIMER Base Address: **TMR01_BA = 0x4005_0000** **TMR23_BA = 0x4005_1000**				
TIMER0_CTL	TMR01_BA+0x00	R/W	Timer0 Control Register	0x0000_0005
TIMER0_CMP	TMR01_BA+0x04	R/W	Timer0 Comparator Register	0x0000_0000
TIMER0_INTSTS	TMR01_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TIMER0_CNT	TMR01_BA+0x0C	R/W	Timer0 Data Register	0x0000_0000
TIMER0_CAP	TMR01_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TIMER0_EXTCTL	TMR01_BA+0x14	R/W	Timer0 External Control Register	0x0000_0000
TIMER0_EINTSTS	TMR01_BA+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TIMER0_TRGCTL	TMR01_BA+0x1C	R/W	Timer0 Trigger Control Register	0x0000_0000
TIMER0_CAPNF	TMR01_BA+0x24	R/W	Timer0 Capture Input Noise Filter Register	0x0000_0000
TIMER0_PWMCTL	TMR01_BA+0x40	R/W	Timer0 PWM Control Register	0x0000_0000
TIMER0_PWMCLKPSC	TMR01_BA+0x44	R/W	Timer0 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER0_PWMCNTCLR	TMR01_BA+0x48	R/W	Timer0 PWM Clear Counter Register	0x0000_0000
TIMER0_PWMPERIOD	TMR01_BA+0x4C	R/W	Timer0 PWM Period Register	0x0000_0000
TIMER0_PWMCMPDAT	TMR01_BA+0x50	R/W	Timer0 PWM Comparator Register	0x0000_0000
TIMER0_PWMCNT	TMR01_BA+0x54	R	Timer0 PWM Counter Register	0x0000_0000
TIMER0_PWMPOLCTL	TMR01_BA+0x58	R/W	Timer0 PWM Pin Output Polar Control Register	0x0000_0000
TIMER0_PWMPOCTL	TMR01_BA+0x5C	R/W	Timer0 PWM Pin Output Control Register	0x0000_0000
TIMER0_PWMINTEN0	TMR01_BA+0x60	R/W	Timer0 PWM Interrupt Enable Register 0	0x0000_0000
TIMER0_PWMINTSTS0	TMR01_BA+0x64	R/W	Timer0 PWM Interrupt Status Register 0	0x0000_0000
TIMER0_PWMTRGCTL	TMR01_BA+0x68	R/W	Timer0 PWM Trigger Control Register	0x0000_0000
TIMER0_PWMSTATUS	TMR01_BA+0x6C	R/W	Timer0 PWM Status Register	0x0000_0000
TIMER0_PWMPBUF	TMR01_BA+0x70	R	Timer0 PWM Period Buffer Register	0x0000_0000

TIMER0_PWMCMPBUF	TMR01_BA+0x74	R	Timer0 PWM Comparator Buffer Register	0x0000_0000
TIMER0_PWMIFA	TMR01_BA+0xA8	R/W	Timer0 PWM Interrupt Flag Accumulator Register	0x0000_0000
TIMER0_PWMAINTSTS	TMR01_BA+0xAC	R/W	Timer0 PWM Accumulator Interrupt Flag Register	0x0000_0000
TIMER0_PWMAINTEN	TMR01_BA+0xB0	R/W	Timer0 PWM Accumulator Interrupt Enable Register	0x0000_0000
TIMER0_PWMAPDMACTL	TMR01_BA+0xB4	R/W	Timer0 PWM Accumulator PDMA Control Register	0x0000_0000
TIMER1_CTL	TMR01_BA+0x100	R/W	Timer1 Control Register	0x0000_0005
TIMER1_CMP	TMR01_BA+0x104	R/W	Timer1 Comparator Register	0x0000_0000
TIMER1_INTSTS	TMR01_BA+0x108	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER1_CNT	TMR01_BA+0x10C	R/W	Timer1 Data Register	0x0000_0000
TIMER1_CAP	TMR01_BA+0x110	R	Timer1 Capture Data Register	0x0000_0000
TIMER1_EXTCTL	TMR01_BA+0x114	R/W	Timer1 External Control Register	0x0000_0000
TIMER1_EINTSTS	TMR01_BA+0x118	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TIMER1_TRGCTL	TMR01_BA+0x11C	R/W	Timer1 Trigger Control Register	0x0000_0000
TIMER1_CAPNF	TMR01_BA+0x124	R/W	Timer1 Capture Input Noise Filter Register	0x0000_0000
TIMER1_PWMCTL	TMR01_BA+0x140	R/W	Timer1 PWM Control Register	0x0000_0000
TIMER1_PWMCLKPSC	TMR01_BA+0x144	R/W	Timer1 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER1_PWMCNTCLR	TMR01_BA+0x148	R/W	Timer1 PWM Clear Counter Register	0x0000_0000
TIMER1_PWMPERIOD	TMR01_BA+0x14C	R/W	Timer1 PWM Period Register	0x0000_0000
TIMER1_PWMCMPDAT	TMR01_BA+0x150	R/W	Timer1 PWM Comparator Register	0x0000_0000
TIMER1_PWMCNT	TMR01_BA+0x154	R	Timer1 PWM Counter Register	0x0000_0000
TIMER1_PWMPOLCTL	TMR01_BA+0x158	R/W	Timer1 PWM Pin Output Polar Control Register	0x0000_0000
TIMER1_PWMPOCTL	TMR01_BA+0x15C	R/W	Timer1 PWM Pin Output Control Register	0x0000_0000
TIMER1_PWMINTEN0	TMR01_BA+0x160	R/W	Timer1 PWM Interrupt Enable Register 0	0x0000_0000
TIMER1_PWMINTSTS0	TMR01_BA+0x164	R/W	Timer1 PWM Interrupt Status Register 0	0x0000_0000
TIMER1_PWMTRGCTL	TMR01_BA+0x168	R/W	Timer1 PWM Trigger Control Register	0x0000_0000

TIMER1_PWMSTATUS	TMR01_BA+0x16C	R/W	Timer1 PWM Status Register	0x0000_0000
TIMER1_PWMPBUF	TMR01_BA+0x170	R	Timer1 PWM Period Buffer Register	0x0000_0000
TIMER1_PWMCMPBUF	TMR01_BA+0x174	R	Timer1 PWM Comparator Buffer Register	0x0000_0000
TIMER1_PWMIFA	TMR01_BA+0x1A8	R/W	Timer1 PWM Interrupt Flag Accumulator Register	0x0000_0000
TIMER1_PWMAINTSTS	TMR01_BA+0x1AC	R/W	Timer1 PWM Accumulator Interrupt Flag Register	0x0000_0000
TIMER1_PWMAINTEN	TMR01_BA+0x1B0	R/W	Timer1 PWM Accumulator Interrupt Enable Register	0x0000_0000
TIMER1_PWMAPDMACTL	TMR01_BA+0x1B4	R/W	Timer1 PWM Accumulator PDMA Control Register	0x0000_0000
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2 Control Register	0x0000_0005
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2 Comparator Register	0x0000_0000
TIMER2_INTSTS	TMR23_BA+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TIMER2_CNT	TMR23_BA+0x0C	R/W	Timer2 Data Register	0x0000_0000
TIMER2_CAP	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TIMER2_EXTCTL	TMR23_BA+0x14	R/W	Timer2 External Control Register	0x0000_0000
TIMER2_EINTSTS	TMR23_BA+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TIMER2_TRGCTL	TMR23_BA+0x1C	R/W	Timer2 Trigger Control Register	0x0000_0000
TIMER2_CAPNF	TMR23_BA+0x24	R/W	Timer2 Capture Input Noise Filter Register	0x0000_0000
TIMER2_PWMCTL	TMR23_BA+0x40	R/W	Timer2 PWM Control Register	0x0000_0000
TIMER2_PWMCLKPSC	TMR23_BA+0x44	R/W	Timer2 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER2_PWMCNTCLR	TMR23_BA+0x48	R/W	Timer2 PWM Clear Counter Register	0x0000_0000
TIMER2_PWMPERIOD	TMR23_BA+0x4C	R/W	Timer2 PWM Period Register	0x0000_0000
TIMER2_PWMCMPDAT	TMR23_BA+0x50	R/W	Timer2 PWM Comparator Register	0x0000_0000
TIMER2_PWMCNT	TMR23_BA+0x54	R	Timer2 PWM Counter Register	0x0000_0000
TIMER2_PWMPOLCTL	TMR23_BA+0x58	R/W	Timer2 PWM Pin Output Polar Control Register	0x0000_0000
TIMER2_PWMPOCTL	TMR23_BA+0x5C	R/W	Timer2 PWM Pin Output Control Register	0x0000_0000
TIMER2_PWMINTEN0	TMR23_BA+0x60	R/W	Timer2 PWM Interrupt Enable Register 0	0x0000_0000
TIMER2_PWMINTSTS0	TMR23_BA+0x64	R/W	Timer2 PWM Interrupt Status Register	0x0000_0000

			0	
TIMER2_PWMTRGCTL	TMR23_BA+0x68	R/W	Timer2 PWM Trigger Control Register	0x0000_0000
TIMER2_PWMSTATUS	TMR23_BA+0x6C	R/W	Timer2 PWM Status Register	0x0000_0000
TIMER2_PWMPBUF	TMR23_BA+0x70	R	Timer2 PWM Period Buffer Register	0x0000_0000
TIMER2_PWMCMPBUF	TMR23_BA+0x74	R	Timer2 PWM Comparator Buffer Register	0x0000_0000
TIMER2_PWMIFA	TMR23_BA+0xA8	R/W	Timer2 PWM Interrupt Flag Accumulator Register	0x0000_0000
TIMER2_PWMAINTSTS	TMR23_BA+0xAC	R/W	Timer2 PWM Accumulator Interrupt Flag Register	0x0000_0000
TIMER2_PWMAINTEN	TMR23_BA+0xB0	R/W	Timer2 PWM Accumulator Interrupt Enable Register	0x0000_0000
TIMER2_PWMAPDMACTL	TMR23_BA+0xB4	R/W	Timer2 PWM Accumulator PDMA Control Register	0x0000_0000
TIMER3_CTL	TMR23_BA+0x100	R/W	Timer3 Control Register	0x0000_0005
TIMER3_CMP	TMR23_BA+0x104	R/W	Timer3 Comparator Register	0x0000_0000
TIMER3_INTSTS	TMR23_BA+0x108	R/W	Timer3 Interrupt Status Register	0x0000_0000
TIMER3_CNT	TMR23_BA+0x10C	R/W	Timer3 Data Register	0x0000_0000
TIMER3_CAP	TMR23_BA+0x110	R	Timer3 Capture Data Register	0x0000_0000
TIMER3_EXTCTL	TMR23_BA+0x114	R/W	Timer3 External Control Register	0x0000_0000
TIMER3_EINTSTS	TMR23_BA+0x118	R/W	Timer3 External Interrupt Status Register	0x0000_0000
TIMER3_TRGCTL	TMR23_BA+0x11C	R/W	Timer3 Trigger Control Register	0x0000_0000
TIMER3_CAPNF	TMR23_BA+0x124	R/W	Timer3 Capture Input Noise Filter Register	0x0000_0000
TIMER3_PWMCTL	TMR23_BA+0x140	R/W	Timer3 PWM Control Register	0x0000_0000
TIMER3_PWMCLKPSC	TMR23_BA+0x144	R/W	Timer3 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER3_PWMCNTCLR	TMR23_BA+0x148	R/W	Timer3 PWM Clear Counter Register	0x0000_0000
TIMER3_PWMPERIOD	TMR23_BA+0x14C	R/W	Timer3 PWM Period Register	0x0000_0000
TIMER3_PWMCMPDAT	TMR23_BA+0x150	R/W	Timer3 PWM Comparator Register	0x0000_0000
TIMER3_PWMCNT	TMR23_BA+0x154	R	Timer3 PWM Counter Register	0x0000_0000
TIMER3_PWMPOLCTL	TMR23_BA+0x158	R/W	Timer3 PWM Pin Output Polar Control Register	0x0000_0000
TIMER3_PWMPOCTL	TMR23_BA+0x15C	R/W	Timer3 PWM Pin Output Control Register	0x0000_0000

TIMER3_PWMINTEN0	TMR23_BA+0x160	R/W	Timer3 PWM Interrupt Enable Register 0	0x0000_0000
TIMER3_PWMINTSTS0	TMR23_BA+0x164	R/W	Timer3 PWM Interrupt Status Register 0	0x0000_0000
TIMER3_PWMTRGCTL	TMR23_BA+0x168	R/W	Timer3 PWM Trigger Control Register	0x0000_0000
TIMER3_PWMSTATUS	TMR23_BA+0x16C	R/W	Timer3 PWM Status Register	0x0000_0000
TIMER3_PWMPBUF	TMR23_BA+0x170	R	Timer3 PWM Period Buffer Register	0x0000_0000
TIMER3_PWMCMPBUF	TMR23_BA+0x174	R	Timer3 PWM Comparator Buffer Register	0x0000_0000
TIMER3_PWMIFA	TMR23_BA+0x1A8	R/W	Timer3 PWM Interrupt Flag Accumulator Register	0x0000_0000
TIMER3_PWMAINTSTS	TMR23_BA+0x1AC	R/W	Timer3 PWM Accumulator Interrupt Flag Register	0x0000_0000
TIMER3_PWMAINTEN	TMR23_BA+0x1B0	R/W	Timer3 PWM Accumulator Interrupt Enable Register	0x0000_0000
TIMER3_PWMAPDMACTL	TMR23_BA+0x1B4	R/W	Timer3 PWM Accumulator PDMA Control Register	0x0000_0000

6.11.8 Register Description

Timer Control Register (TIMERx_CTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_CTL	TMR01_BA+0x00	R/W	Timer0 Control Register	0x0000_0005
TIMER1_CTL	TMR01_BA+0x100	R/W	Timer1 Control Register	0x0000_0005
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2 Control Register	0x0000_0005
TIMER3_CTL	TMR23_BA+0x100	R/W	Timer3 Control Register	0x0000_0005

31	30	29	28	27	26	25	24
ICEDEBUG	CNTEN	INTEN	OPMODE		Reserved	ACTSTS	EXTCNTEN
23	22	21	20	19	18	17	16
WKEN	CAPSRC	TGLPINSEL	PERIOSEL	INTRGEN	Reserved		
15	14	13	12	11	10	9	8
FUNCSEL	Reserved						
7	6	5	4	3	2	1	0
PSC							

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects TIMER counting. TIMER counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. TIMER counter will keep going no matter CPU is held by ICE or not. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[30]	**CNTEN**	**Timer Counting Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. **Note 1:** In stop status, and then set CNTEN to 1 will enable the 24-bit up counter to keep counting from the last stop counting value. **Note 2:** This bit is auto-cleared by hardware in one-shot mode (TIMER_CTL[28:27] = 00) when the timer time-out interrupt flag TIF (TIMERx_INTSTS[0]) is generated. **Note 3:** Set enable/disable this bit needs 2 * TMR_CLK period to become active, user can read ACTSTS (TIMERx_CTL[25]) to check enable/disable command is completed or not.
[29]	**INTEN**	**Timer Interrupt Enable Bit** 0 = Timer time-out interrupt Disabled. 1 = Timer time-out interrupt Enabled. **Note:** If this bit is enabled, when the timer time-out interrupt flag TIF is set to 1, the timer interrupt signal is generated and inform to CPU.
[28:27]	**OPMODE**	**Timer Counting Mode Select** 00 = The timer controller is operated in One-shot mode.

		01 = The timer controller is operated in Periodic mode. 10 = The timer controller is operated in Toggle-output mode. 11 = The timer controller is operated in Continuous Counting mode.
[26]	**Reserved**	Reserved.
[25]	**ACTSTS**	**Timer Active Status Bit (Read Only)** This bit indicates the 24-bit up counter status. 0 = 24-bit up counter is not active. 1 = 24-bit up counter is active. **Note:** This bit may active when CNT 0 transition to CNT 1.
[24]	**EXTCNTEN**	**Event Counter Mode Enable Bit** This bit is for external counting pin function enabled. 0 = Event counter mode Disabled. 1 = Event counter mode Enabled. **Note:** When timer is used as an event counter, this bit should be set to 1.
[23]	**WKEN**	**Wake-up Function Enable Bit** If this bit is set to 1, while timer interrupt flag TIF (TIMERx_INTSTS[0]) is 1 and INTEN (TIMERx_CTL[29]) is enabled, the timer interrupt signal will generate a wake-up trigger event to CPU. 0 = Wake-up function Disabled if timer interrupt signal generated. 1 = Wake-up function Enabled if timer interrupt signal generated.
[22]	**CAPSRC**	**Capture Pin Source Selection** 0 = Capture Function source is from TMx_EXT (x= 0~3) pin. 1 = Capture Function source is from internal ACMP output signal, internal clock (MIRC, LIRC, HIRC), or external clock (HXT, LXT). **Note:** When CAPSRC = 1,User can set INTERCAPSEL (TIMERx_EXTCTL[10:8]) to decide which internal ACMP output signal or which clock is as timer capture source.
[21]	**TGLPINSEL**	**Toggle-output Pin Select** 0 = Toggle mode output to TMx (Timer Event Counter Pin). 1 = Toggle mode output to TMx_EXT (Timer External Capture Pin).
[20]	**PERIOSEL**	**Periodic Mode Behavior Selection Enable Bit** 0 = The behavior selection in periodic mode is Disabled. When user updates CMPDAT while timer is running in periodic mode, CNT will be reset to default value. 1 = The behavior selection in periodic mode is Enabled. When user updates CMPDAT while timer is running in periodic mode, the limitations as bellows list, If updated CMPDAT value > CNT, CMPDAT will be updated and CNT keep running continually. If updated CMPDAT value = CNT, timer time-out interrupt will be asserted immediately. If updated CMPDAT value < CNT, CNT will be reset to default value.
[19]	**INTRGEN**	**Inter-timer Trigger Mode Enable Bit** Setting this bit will enable the inter-timer trigger capture function. The Timer0/2 will be in event counter mode and counting with external clock source or event. Also, Timer1/3 will be in trigger-counting mode of capture function. 0 = Inter-Timer Trigger Capture mode Disabled. 1 = Inter-Timer Trigger Capture mode Enabled. **Note:** For Timer1/3, this bit is ineffective and the read back value is always 0.
[18:16]	**Reserved**	Reserved.
[15]	**FUNCSEL**	**Function Selection**

		0 = Timer controller is used as timer function. 1 = Timer controller is used as PWM function.
[14:8]	**Reserved**	Reserved.
[7:0]	**PSC**	**Prescale Counter** Timer input clock or event source is divided by (PSC+1) before it is fed to the timer up counter. If this field is 0 (PSC = 0), then there is no scaling. **Note:** Update prescale counter value will reset internal 8-bit prescale counter and 24-bit up counter value.

Timer Comparator Register (TIMERx_CMP)

Register	Offset	R/W	Description	Reset Value
TIMER0_CMP	TMR01_BA+0x04	R/W	Timer0 Comparator Register	0x0000_0000
TIMER1_CMP	TMR01_BA+0x104	R/W	Timer1 Comparator Register	0x0000_0000
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2 Comparator Register	0x0000_0000
TIMER3_CMP	TMR23_BA+0x104	R/W	Timer3 Comparator Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPDAT							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CMPDAT**	**Timer Comparator Value** CMPDAT is a 24-bit compared value register. When the internal 24-bit up counter value is equal to CMPDAT value, the TIF (TIMERx_INTSTS[0] Timer Interrupt Flag) will set to 1. Time-out period = (Period of timer clock input) * (8-bit PSC + 1) * (24-bit CMPDAT). **Note 1:** Never write 0x0 or 0x1 in CMPDAT field, or the core will run into unknown state. **Note 2:** When timer is operating at continuous counting mode, the 24-bit up counter will keep counting continuously even if user writes a new value into CMPDAT field. But if timer is operating at other modes, the 24-bit up counter will restart counting from 0 and using newest CMPDAT value to be the timer compared value while user writes a new value into CMPDAT field.

Timer Interrupt Status Register (TIMERx_INTSTS)

Register	Offset	R/W	Description	Reset Value
TIMER0_INTSTS	TMR01_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TIMER1_INTSTS	TMR01_BA+0x108	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER2_INTSTS	TMR23_BA+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TIMER3_INTSTS	TMR23_BA+0x108	R/W	Timer3 Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TWKF	TIF

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TWKF**	**Timer Wake-up Flag** This bit indicates the interrupt wake-up flag status of timer. 0 = Timer does not cause CPU wake-up. 1 = CPU wake-up from Idle or Power-down mode if timer time-out interrupt signal generated. **Note:** This bit is cleared by writing 1 to it.
[0]	**TIF**	**Timer Interrupt Flag** This bit indicates the interrupt flag status of Timer while 24-bit timer up counter CNT (TIMERx_CNT[23:0]) value reaches to CMPDAT (TIMERx_CMP[23:0]) value. 0 = No effect. 1 = CNT value matches the CMPDAT value. **Note:** This bit is cleared by writing 1 to it.

Timer Data Register (TIMERx_CNT)

Register	Offset	R/W	Description	Reset Value
TIMER0_CNT	TMR01_BA+0x0C	R/W	Timer0 Data Register	0x0000_0000
TIMER1_CNT	TMR01_BA+0x10C	R/W	Timer1 Data Register	0x0000_0000
TIMER2_CNT	TMR23_BA+0x0C	R/W	Timer2 Data Register	0x0000_0000
TIMER3_CNT	TMR23_BA+0x10C	R/W	Timer3 Data Register	0x0000_0000

31	30	29	28	27	26	25	24
RSTACT	Reserved						
23	22	21	20	19	18	17	16
CNT							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

Bits	Description	
[31]	**RSTACT**	**Timer Data Register Reset Active (Read Only)** This bit indicates if the counter reset operation active. When user writes this CNT register, timer starts to reset its internal 24-bit timer up-counter to 0 and reload 8-bit pre-scale counter. At the same time, timer set this flag to 1 to indicate the counter reset operation is in progress. Once the counter reset operation done, timer clear this bit to 0 automatically. 0 = Reset operation is done. 1 = Reset operation triggered by writing TIMERx_CNT is in progress.
[30:24]	**Reserved**	Reserved.
[23:0]	**CNT**	**Timer Data Register** Read operation. Read this register to get CNT value. For example: If EXTCNTEN (TIMERx_CTL[24]) is 0, user can read CNT value for getting current 24-bit counter value. If EXTCNTEN (TIMERx_CTL[24]) is 1, user can read CNT value for getting current 24-bit event input counter value. Write operation. Writing any value to this register will reset current CNT value to 0 and reload internal 8-bit prescale counter.

Timer Capture Data Register (TIMERx_CAP)

Register	Offset	R/W	Description	Reset Value
TIMER0_CAP	TMR01_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TIMER1_CAP	TMR01_BA+0x110	R	Timer1 Capture Data Register	0x0000_0000
TIMER2_CAP	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TIMER3_CAP	TMR23_BA+0x110	R	Timer3 Capture Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CAPDAT							
15	14	13	12	11	10	9	8
CAPDAT							
7	6	5	4	3	2	1	0
CAPDAT							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CAPDAT**	**Timer Capture Data Register** When CAPEN (TIMERx_EXTCTL[3]) bit is set, CAPFUNCS (TIMERx_EXTCTL[4]) bit is 0, and a transition on TMx_EXT pin matched the CAPEDGE (TIMERx_EXTCTL[14:12]) setting, CAPIF (TIMERx_EINTSTS[0]) will set to 1 and the current timer counter value CNT (TIMERx_CNT[23:0]) will be auto-loaded into this CAPDAT field.

Timer External Control Register (TIMERx_EXTCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_EXTCTL	TMR01_BA+0x14	R/W	Timer0 External Control Register	0x0000_0000
TIMER1_EXTCTL	TMR01_BA+0x114	R/W	Timer1 External Control Register	0x0000_0000
TIMER2_EXTCTL	TMR23_BA+0x14	R/W	Timer2 External Control Register	0x0000_0000
TIMER3_EXTCTL	TMR23_BA+0x114	R/W	Timer3 External Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CAPDIVSCL				Reserved			
23	22	21	20	19	18	17	16
Reserved					ECNTSSEL		
15	14	13	12	11	10	9	8
Reserved	CAPEDGE			Reserved	INTERCAPSEL		
7	6	5	4	3	2	1	0
CNTDBEN	CAPDBEN	CAPIEN	CAPFUNCS	CAPEN	Reserved		CNTPHASE

Bits	Description	
[31:28]	**CAPDIVSCL**	**Timer Capture Source Divider Scale** **This bits indicate the divide scale for capture source divider** 0000 = Capture source/1. 0001 = Capture source/2. 0010 = Capture source/4. 0011 = Capture source/8. 0100 = Capture source/16. 0101 = Capture source/32. 0110 = Capture source/64. 0111 = Capture source/128. 1000 = Capture source/256. 1001~1111 = Reserved. **Note:** Set INTERCAPSEL (TIMERx_EXTCTL[10:8]) and CAPSRC (TIMERx_CTL[22]) to select capture source.
[27:19]	**Reserved**	Reserved.
[18:16]	**ECNTSSEL**	**Event Counter Source Selection to Trigger Event Counter Function** 000 = Event Counter input source is from external TMx (x= 0~3) pin. 001 = Event Counter input source is from internal USB internal SOF output signal. 010 = Event Counter input source is from internal ACMP0 output signal. 011 = Event Counter input source is from internal ACMP1 output signal. 100 = Event Counter input source is from internal ACMP2 output signal. 101~111 = Reserved.

[15]	**Reserved**	Reserved.
[14:12]	**CAPEDGE**	**Timer External Capture Pin Edge Detect** When first capture event is generated, the CNT (TIMERx_CNT[23:0]) will be reset to 0 and first CAPDAT (TIMERx_CAP[23:0]) should be to 0. 000 = Capture event occurred when detect falling edge transfer on TMx_EXT (x= 0~3) pin. 001 = Capture event occurred when detect rising edge transfer on TMx_EXT (x= 0~3) pin. 010 = Capture event occurred when detect both falling and rising edge transfer on TMx_EXT (x= 0~3) pin, and first capture event occurred at falling edge transfer. 011 = Capture event occurred when detect both rising and falling edge transfer on TMx_EXT (x= 0~3) pin, and first capture event occurred at rising edge transfer. 110 = First capture event occurred at falling edge, follows capture events are at rising edge transfer on TMx_EXT (x= 0~3) pin. 111 = First capture event occurred at rising edge, follows capture events are at falling edge transfer on TMx_EXT (x= 0~3) pin. 100, 101 = Reserved. **Note:** Set CAPSRC (TIMERx_CTL[22]) and INTERCAPSEL (TIMERx_EXTCTL[10:8]) to select capture source.
[11]	**Reserved**	Reserved.
[10:8]	**INTERCAPSEL**	**Internal Capture Source Select** 000 = Capture Function source is from internal ACMP0 output signal. 001 = Capture Function source is from internal ACMP1 output signal. 010 = Capture Function source is from HXT. 011 = Capture Function source is from LXT. 100 = Capture Function source is from HIRC. 101 = Capture Function source is from LIRC. 110 = Capture Function source is from MIRC. 111 = Capture Function source is from internal ACMP2 output signal.. **Note:** these bits only available when CAPSRC (TIMERx_CTL[22]) is 1.
[7]	**CNTDBEN**	**Timer Counter Pin De-bounce Enable Bit** 0 = TMx (x= 0~3) pin de-bounce Disabled. 1 = TMx (x= 0~3) pin de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of TMx pin is detected with de-bounce circuit.
[6]	**CAPDBEN**	**Timer External Capture Pin De-bounce Enable Bit** 0 = TMx_EXT (x= 0~3) pin de-bounce or ACMP output de-bounce Disabled. 1 = TMx_EXT (x= 0~3) pin de-bounce or ACMP output de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of TMx_EXT pin or ACMP output is detected with de-bounce circuit.
[5]	**CAPIEN**	**Timer External Capture Interrupt Enable Bit** 0 = TMx_EXT (x= 0~3) pin, ACMP, internal clock, or external clock detection Interrupt Disabled. 1 = TMx_EXT (x= 0~3) pin, ACMP, internal clock, or external clock detection Interrupt Enabled. **Note:** CAPIEN is used to enable timer external interrupt. If CAPIEN enabled, timer will rise an interrupt when CAPIF (TIMERx_EINTSTS[0]) is 1. For example, while CAPIEN = 1, CAPEN = 1, and CAPEDGE = 00, a 1 to 0 transition on the TMx_EXT (x= 0~3) pin, ACMP, internal clock, or external clock will cause the CAPIF to be set then the interrupt signal is generated and sent to NVIC to inform CPU.
[4]	**CAPFUNCS**	**Capture Function Selection** 0 = External Capture Mode Enabled. 1 = External Reset Mode Enabled. **Note 1:** When CAPFUNCS is 0, transition on TMx_EXT (x= 0~3) pin is using to save current 24-bit

		timer counter value (CNT value) to CAPDAT field. **Note 2:** When CAPFUNCS is 1, transition on TMx_EXT (x= 0~3) pin is using to save current 24-bit timer counter value (CNT value) to CAPDAT field then CNT value will be reset immediately.
[3]	**CAPEN**	**Timer Capture Enable Bit** This bit enables the capture input function. 0 =Capture source Disabled. 1 =Capture source Enabled. **Note:** When CAPEN is 1, user can set INTERCAPSEL (TIMERx_EXTCTL [10:8]) to select capture source.
[2:1]	**Reserved**	Reserved.
[0]	**CNTPHASE**	**Timer External Count Phase** This bit indicates the detection phase of external counting pin TMx (x= 0~3). 0 = A falling edge of external counting pin will be counted. 1 = A rising edge of external counting pin will be counted.

Timer External Interrupt Status Register (TIMERx_EINTSTS)

Register	Offset	R/W	Description	Reset Value
TIMER0_EINTSTS	TMR01_BA+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TIMER1_EINTSTS	TMR01_BA+0x118	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TIMER2_EINTSTS	TMR23_BA+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TIMER3_EINTSTS	TMR23_BA+0x118	R/W	Timer3 External Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CAPIFOV	CAPIF

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CAPIFOV**	**Capture Latch Interrupt Flag Overrun Status (Read Only)** 0 = Capture latch happened when the corresponding CAPIF is 0. 1 = Capture latch happened when the corresponding CAPIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CAPIF.
[0]	**CAPIF**	**Timer External Capture Interrupt Flag** This bit indicates the timer external capture interrupt flag status. 0 = TMx_EXT (x= 0~3) pin, ACMP, internal clock, or external clock interrupt did not occur. 1 = TMx_EXT (x= 0~3) pin, ACMP, internal clock, or external clock interrupt occurred. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** When CAPEN (TIMERx_EXTCTL[3]) bit is set, CAPFUNCS (TIMERx_EXTCTL[4]) bit is 0, and a transition on TMx_EXT (x= 0~3) pin, ACMP, internal clock, or external clock matched the CAPEDGE (TIMERx_EXTCTL[2:1]) setting, this bit will set to 1 by hardware. **Note 3:** There is a new incoming capture event detected before CPU clearing the CAPIF status. If the above condition occurred, the Timer will keep register TIMERx_CAP unchanged and drop the new capture value.

Timer Trigger Control Register (TIMERx_TRGCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_TRGCTL	TMR01_BA+0x1C	R/W	Timer0 Trigger Control Register	0x0000_0000
TIMER1_TRGCTL	TMR01_BA+0x11C	R/W	Timer1 Trigger Control Register	0x0000_0000
TIMER2_TRGCTL	TMR23_BA+0x1C	R/W	Timer2 Trigger Control Register	0x0000_0000
TIMER3_TRGCTL	TMR23_BA+0x11C	R/W	Timer3 Trigger Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	8	13	12	11	10	9	8
Reserved							TRGTK
7	6	5	4	3	2	1	0
Reserved		TRGLPADC	TRGPDMA	TRGDAC	TRGEADC	TRGPWM	TRGSSEL

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**TRGTK**	**Trigger Touch-key Scan Enable Bit** If this bit is set to 1, timer time-out interrupt can trigger Touck-Key start scan. 0 = Timer time-out interrupt signal trigger Touch-Key scan Disabled. 1 = Timer time-out interrupt signal trigger Touch-Key scan Enabled.
[7:6]	**Reserved**	Reserved.
[5]	**TRGLPADC**	**Trigger LPADC Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be triggered LPADC transfer. 0 = Timer interrupt trigger LPADC Disabled. 1 = Timer interrupt trigger LPADC Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal will trigger LPADC. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal will trigger LPADC.
[4]	**TRGPDMA**	**Trigger PDMA Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be triggered PDMA transfer. 0 = Timer interrupt trigger PDMA Disabled. 1 = Timer interrupt trigger PDMA Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal will trigger PDMA transfer. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal will trigger PDMA transfer.
[3]	**TRGDAC**	**Trigger DAC Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can be triggered DAC. 0 = Timer interrupt trigger DAC Disabled.

		1 = Timer interrupt trigger DAC Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal will trigger DAC. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal will trigger DAC.
[2]	**TRGEADC**	**Trigger EADC Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be triggered EADC conversion. 0 = Timer interrupt trigger EADC Disabled. 1 = Timer interrupt trigger EADC Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal will trigger EADC conversion. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal will trigger EADC conversion.
[1]	**TRGPWM**	**Trigger PWM/EPWM Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be as PWM/EPWM counter clock source. 0 = Timer interrupt trigger PWM/EPWM Disabled. 1 = Timer interrupt trigger PWM/EPWM Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal as PWM/EPWM counter clock source. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal as PWM/EPWM counter clock source.
[0]	**TRGSSEL**	**Trigger Source Select Bit** This bit is used to select internal trigger source is form timer time-out interrupt signal or capture interrupt signal. 0 = Time-out interrupt signal is used to internal trigger PWM, EPWM, PDMA, DAC, LPADC and EADC. 1 = Capture interrupt signal is used to internal trigger PWM, EPWM, PDMA, DAC, LPADC and EADC.

Timer Capture Input Noise Filter Register (TIMERx_CAPNF)

Register	Offset	R/W	Description	Reset Value
TIMER0_CAPNF	TMR01_BA+0x24	R/W	Timer0 Capture Input Noise Filter Register	0x0000_0000
TIMER1_CAPNF	TMR01_BA+0x124	R/W	Timer1 Capture Input Noise Filter Register	0x0000_0000
TIMER2_CAPNF	TMR23_BA+0x24	R/W	Timer2 Capture Input Noise Filter Register	0x0000_0000
TIMER3_CAPNF	TMR23_BA+0x124	R/W	Timer3 Capture Input Noise Filter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**CAPNFCNT**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CAPNFSEL**			**Reserved**			**CAPNFEN**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**CAPNFCNT**	**Capture Edge Detector Noise Filter Count** These bits control the capture filter counter to count from 0 to CAPNFCNT.
[7]	**Reserved**	Reserved.
[6:4]	**CAPNFSEL**	**Capture Edge Detector Noise Filter Clock Selection** 000 = Noise filter clock is ECLKx. 001 = Noise filter clock is ECLKx/2. 010 = Noise filter clock is ECLKx/4. 011 = Noise filter clock is ECLKx/8. 100 = Noise filter clock is ECLKx/16. 101 = Noise filter clock is ECLKx/32. 110 = Noise filter clock is ECLKx/64. 111 = Noise filter clock is ECLKx/128.
[3:1]	**Reserved**	Reserved.
[0]	**CAPNFEN**	**Capture Noise Filter Enable** 0 = Capture Noise Filter function Disabled. 1 = Capture Noise Filter function Enabled.

Timer PWM Control Register (TIMERx_PWMCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCTL	TMR01_BA+0x40	R/W	Timer0 PWM Control Register	0x0000_0000
TIMER1_PWMCTL	TMR01_BA+0x140	R/W	Timer1 PWM Control Register	0x0000_0000
TIMER2_PWMCTL	TMR23_BA+0x40	R/W	Timer2 PWM Control Register	0x0000_0000
TIMER3_PWMCTL	TMR23_BA+0x140	R/W	Timer3 PWM Control Register	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	**DBGHALT**	**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**PWMINTWKEN**	**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**CNTMODE**	**Reserved**		**CNTEN**

Bits	Description	
[31]	**DBGTRIOFF**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects PWM output. PWM output pin will be forced as tri-state while ICE debug mode acknowledged. 1 = ICE debug mode acknowledgement disabled. PWM output pin will keep output no matter ICE debug mode acknowledged or not. **Note:** This bit is write protected. Refer to SYS_REGLCTL control register.
[30]	**DBGHALT**	**ICE Debug Mode Counter Halt (Write Protect)** If debug mode counter halt is enabled, PWM counter will keep current value until exit ICE debug mode. 0 = ICE debug mode counter halt Disabled. 1 = ICE debug mode counter halt Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL control register.
[29:13]	**Reserved**	Reserved.
[12]	**PWMINTWKEN**	**PWM Interrupt Wake-up Enable Bit** If PWM interrupt occurs when chip is in Power-down mode, PWMINTWKEN can determine whether chip wake-up occurs or not. 0 = PWM interrupt wake-up Disabled. 1 = PWM interrupt wake-up Enabled.
[11:4]	**Reserved**	Reserved.
[3]	**CNTMODE**	**PWM Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.
[2:1]	**Reserved**	Reserved.

[0]	**CNTEN**	**PWM Counter Enable Bit** 0 = PWM counter and clock prescale Stop Running. 1 = PWM counter and clock prescale Start Running.

Timer PWM Counter Clock Pre-scale Register (TIMERx_PWMCLKPSC)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCLKPSC	TMR01_BA+0x44	R/W	Timer0 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER1_PWMCLKPSC	TMR01_BA+0x144	R/W	Timer1 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER2_PWMCLKPSC	TMR23_BA+0x44	R/W	Timer2 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER3_PWMCLKPSC	TMR23_BA+0x144	R/W	Timer3 PWM Counter Clock Pre-scale Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**CLKPSC**	**PWM Counter Clock Pre-scale** The active clock of PWM counter is decided by counter clock prescale and divided by (CLKPSC + 1). If CLKPSC is 0, then there is no scaling in PWM counter clock source.

Timer PWM Clear Counter Register (TIMERx_PWMCNTCLR)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCNTCLR	TMR01_BA+0x48	W	Timer0 PWM Clear Counter Register	0x0000_0000
TIMER1_PWMCNTCLR	TMR01_BA+0x148	W	Timer1 PWM Clear Counter Register	0x0000_0000
TIMER2_PWMCNTCLR	TMR23_BA+0x48	W	Timer2 PWM Clear Counter Register	0x0000_0000
TIMER3_PWMCNTCLR	TMR23_BA+0x148	W	Timer3 PWM Clear Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTCLR

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTCLR**	**Clear PWM Counter Control Bit** 0 = No effect. 1 = Write 1 to clear 16-bit PWM counter to 0x0000 in up count type. **Note:** When user writes this bit to 1, PWM counter will be cleared after (CLKPSC+16) TMRx_CLK cycles.

Timer PWM Period Register (TIMERx_PWMPERIOD)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPERIOD	TMR01_BA+0x4C	R/W	Timer0 PWM Period Register	0x0000_0000
TIMER1_PWMPERIOD	TMR01_BA+0x14C	R/W	Timer1 PWM Period Register	0x0000_0000
TIMER2_PWMPERIOD	TMR23_BA+0x4C	R/W	Timer2 PWM Period Register	0x0000_0000
TIMER3_PWMPERIOD	TMR23_BA+0x14C	R/W	Timer3 PWM Period Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**PWM Period Register** In up count type: PWM counter counts from 0 to PERIOD, and restarts from 0. In up count type: PWM period time = (PERIOD + 1) * (CLKPSC + 1) * TMRx_PWMCLK.

Timer PWM Comparator Register (TIMERx_PWMCMPDAT)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCMPDAT	TMR01_BA+0x50	R/W	Timer0 PWM Comparator Register	0x0000_0000
TIMER1_PWMCMPDAT	TMR01_BA+0x150	R/W	Timer1 PWM Comparator Register	0x0000_0000
TIMER2_PWMCMPDAT	TMR23_BA+0x50	R/W	Timer2 PWM Comparator Register	0x0000_0000
TIMER3_PWMCMPDAT	TMR23_BA+0x150	R/W	Timer3 PWM Comparator Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMP**	**PWM Comparator Register** PWM CMP is used to compare with PWM CNT to generate PWM output waveform, interrupt events and trigger EADC , LPADC, PDMA, and DAC start conversion.

Timer PWM Counter Register (TIMERx_PWMCNT)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCNT	TMR01_BA+0x54	R	Timer0 PWM Counter Register	0x0000_0000
TIMER1_PWMCNT	TMR01_BA+0x154	R	Timer1 PWM Counter Register	0x0000_0000
TIMER2_PWMCNT	TMR23_BA+0x54	R	Timer2 PWM Counter Register	0x0000_0000
TIMER3_PWMCNT	TMR23_BA+0x154	R	Timer3 PWM Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNT**	**PWM Counter Value Register (Read Only)** User can monitor CNT to know the current counter value in 16-bit period counter.

Timer PWM Pin Output Polar Control Register (TIMERx_PWMPOLCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPOLCTL	TMR01_BA+0x58	R/W	Timer0 PWM Pin Output Polar Control Register	0x0000_0000
TIMER1_PWMPOLCTL	TMR01_BA+0x158	R/W	Timer1 PWM Pin Output Polar Control Register	0x0000_0000
TIMER2_PWMPOLCTL	TMR23_BA+0x58	R/W	Timer2 PWM Pin Output Polar Control Register	0x0000_0000
TIMER3_PWMPOLCTL	TMR23_BA+0x158	R/W	Timer3 PWM Pin Output Polar Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**PINV**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PINV**	**PWMx Output Pin Polar Control Bit** The bit is used to control polarity state of PWMx_OUT pin. 0 = PWMx_OUT pin polar inverse Disabled. 1 = PWMx_OUT polar inverse Enabled. **Note:** Set POSEL (TIMERx_PWMPOCTL[8]) to select TMx or TMx_EXT as PWMx output pin.

Timer PWM Pin Output Control Register (TIMERx PWMPOCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPOCTL	TMR01_BA+0x5C	R/W	Timer0 PWM Pin Output Control Register	0x0000_0000
TIMER1_PWMPOCTL	TMR01_BA+0x15C	R/W	Timer1 PWM Pin Output Control Register	0x0000_0000
TIMER2_PWMPOCTL	TMR23_BA+0x5C	R/W	Timer2 PWM Pin Output Control Register	0x0000_0000
TIMER3_PWMPOCTL	TMR23_BA+0x15C	R/W	Timer3 PWM Pin Output Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							POSEL
7	6	5	4	3	2	1	0
Reserved							POEN

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**POSEL**	**PWM Output Pin Select** 0 = PWMx_OUT pin is TMx. 1 = PWMx_OUT pin is TMx_EXT.
[7:1]	**Reserved**	Reserved.
[0]	**POEN**	**PWMx Output Pin Enable Bit** 0 = PWMx_OUT pin at tri-state mode. 1 = PWMx_OUT pin in output mode. **Note:** Set POSEL (TIMERx_PWMPOCTL[8]) to select TMx or TMx_EXT as PWMx output pin.

Timer PWM Interrupt Enable Register 0 (TIMERx_PWMINTEN0)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMINTEN0	TMR01_BA+0x60	R/W	Timer0 PWM Interrupt Enable Register 0	0x0000_0000
TIMER1_PWMINTEN0	TMR01_BA+0x160	R/W	Timer1 PWM Interrupt Enable Register 0	0x0000_0000
TIMER2_PWMINTEN0	TMR23_BA+0x60	R/W	Timer2 PWM Interrupt Enable Register 0	0x0000_0000
TIMER3_PWMINTEN0	TMR23_BA+0x160	R/W	Timer3 PWM Interrupt Enable Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CMPUIEN	PIEN	Reserved

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CMPUIEN**	**PWM Compare Up Count Interrupt Enable Bit** 0 = Compare up count interrupt Disabled. 1 = Compare up count interrupt Enabled.
[1]	**PIEN**	**PWM Period Point Interrupt Enable Bit** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled.
[0]	**Reserved**	Reserved.

Timer PWM Interrupt Status Register 0 (TIMERx_PWMINTSTS0)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMINTSTS0	TMR01_BA+0x64	R/W	Timer0 PWM Interrupt Status Register 0	0x0000_0000
TIMER1_PWMINTSTS0	TMR01_BA+0x164	R/W	Timer1 PWM Interrupt Status Register 0	0x0000_0000
TIMER2_PWMINTSTS0	TMR23_BA+0x64	R/W	Timer2 PWM Interrupt Status Register 0	0x0000_0000
TIMER3_PWMINTSTS0	TMR23_BA+0x164	R/W	Timer3 PWM Interrupt Status Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CMPUIF	PIF	Reserved

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CMPUIF**	**PWM Compare Up Count Interrupt Flag** This bit is set by hardware when TIMERx_PWM counter in up count direction and reaches CMP. **Note:** If CMP equal to PERIOD, there is no CMPUIF flag in up count type. **Note 2:** This bit is cleared by writing 1 to it.
[1]	**PIF**	**PWM Period Point Interrupt Flag** This bit is set by hardware when TIMERx_PWM counter reaches PERIOD. **Note:** This bit is cleared by writing 1 to it.
[0]	**Reserved**	Reserved.

Timer PWM Trigger Control Register (TIMERx PWMTRGCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMTRGCTL	TMR01_BA+0x68	R/W	Timer0 PWM Trigger Control Register	0x0000_0000
TIMER1_PWMTRGCTL	TMR01_BA+0x168	R/W	Timer1 PWM Trigger Control Register	0x0000_0000
TIMER2_PWMTRGCTL	TMR23_BA+0x68	R/W	Timer2 PWM Trigger Control Register	0x0000_0000
TIMER3_PWMTRGCTL	TMR23_BA+0x168	R/W	Timer3 PWM Trigger Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					PWMTRGLPADC	PWMTRGPDMA	PWMTRGDAC
7	6	5	4	3	2	1	0
PWMTRGEADC	Reserved					TRGSEL	

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**PWMTRGLPADC**	**PWM Counter Event Trigger LPADC Conversion Enable Bit** If this bit is set to 1, PWM can trigger LPADC conversion. 0 = PWM trigger LPADC Disabled. 1 = PWM trigger LPADC Enabled. **Note:** Set TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select PWM trigger conversion source.
[9]	**PWMTRGPDMA**	**PWM Counter Event Trigger PDMA Conversion Enable Bit** If this bit is set to 1, PWM can trigger PDMA conversion. 0 = PWM trigger PDMA Disabled. 1 = PWM trigger PDMA Enabled. **Note:** Set TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select PWM trigger conversion source.
[8]	**PWMTRGDAC**	**PWM Counter Event Trigger DAC Conversion Enable Bit** If this bit is set to 1, PWM can trigger DAC conversion. 0 = PWM trigger DAC Disabled. 1 = PWM trigger DAC Enabled. **Note:** Set TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select PWM trigger conversion source.
[7]	**PWMTRGEADC**	**PWM Counter Event Trigger EADC Conversion Enable Bit** 0 = PWM counter event trigger EADC conversion Disabled. 1 = PWM counter event trigger EADC conversion Enabled. **Note:** Set TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select PWM trigger conversion source.
[6:2]	**Reserved**	Reserved.
[1:0]	**TRGSEL**	**PWM Counter Event Source Select to Trigger Conversion**

		00 = Trigger conversion at period point (PIF). 01 = Trigger conversion at compare up count point (CMPUIF). 10 = Trigger conversion at period or compare up count point (PIF or CMPUIF). 11 = Reserved.

Timer PWM Status Register (TIMERx_PWMSTATUS)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMSTATUS	TMR01_BA+0x6C	R/W	Timer0 PWM Status Register	0x0000_0000
TIMER1_PWMSTATUS	TMR01_BA+0x16C	R/W	Timer1 PWM Status Register	0x0000_0000
TIMER2_PWMSTATUS	TMR23_BA+0x6C	R/W	Timer2 PWM Status Register	0x0000_0000
TIMER3_PWMSTATUS	TMR23_BA+0x16C	R/W	Timer3 PWM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				LPADCTRGF	PDMATRGF	DACTRGF	EADCTRGF
15	14	13	12	11	10	9	8
Reserved							PWMINTWKF
7	6	5	4	3	2	1	0
Reserved							CNTMAXF

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19]	**LPADCTRGF**	**Trigger LPADC Start Conversion Flag** 0 = PWM counter event trigger LPADC start conversion has not occurred. 1 = PWM counter event trigger LPADC start conversion has occurred. **Note:** This bit is cleared by writing 1 to it.
[18]	**PDMATRGF**	**Trigger PDMA Start Conversion Flag** 0 = PWM counter event trigger PDMA start conversion has not occurred. 1 = PWM counter event trigger PDMA start conversion has occurred. **Note:** This bit is cleared by writing 1 to it.
[17]	**DACTRGF**	**Trigger DAC Start Conversion Flag** 0 = PWM counter event trigger DAC start conversion has not occurred. 1 = PWM counter event trigger DAC start conversion has occurred. **Note:** This bit is cleared by writing 1 to it.
[16]	**EADCTRGF**	**Trigger EADC Start Conversion Flag** 0 = PWM counter event trigger EADC start conversion is not occurred. 1 = PWM counter event trigger EADC start conversion has occurred. **Note:** This bit is cleared by writing 1 to it.
[15:9]	**Reserved**	Reserved.
[8]	**PWMINTWKF**	**PWM Interrupt Wake-up Flag** 0 = PWM interrupt wake-up has not occurred. 1 = PWM interrupt wake-up has occurred.

		Note: This bit is cleared by writing 1 to it.
[7:1]	**Reserved**	Reserved.
[0]	**CNTMAXF**	**PWM Counter Equal to 0xFFFF Flag** 0 = The PWM counter value never reached its maximum value 0xFFFF. 1 = The PWM counter value has reached its maximum value. **Note:** This bit is cleared by writing 1 to it.

Timer PWM Period Buffer Register (TIMERx PWMPBUF)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPBUF	TMR01_BA+0x70	R	Timer0 PWM Period Buffer Register	0x0000_0000
TIMER1_PWMPBUF	TMR01_BA+0x170	R	Timer1 PWM Period Buffer Register	0x0000_0000
TIMER2_PWMPBUF	TMR23_BA+0x70	R	Timer2 PWM Period Buffer Register	0x0000_0000
TIMER3_PWMPBUF	TMR23_BA+0x170	R	Timer3 PWM Period Buffer Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PBUF**	**PWM Period Buffer Register (Read Only)** Used as PERIOD active register.

Timer PWM Comparator Buffer Register (TIMERx PWMCMPBUF)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCMPBUF	TMR01_BA+0x74	R	Timer0 PWM Comparator Buffer Register	0x0000_0000
TIMER1_PWMCMPBUF	TMR01_BA+0x174	R	Timer1 PWM Comparator Buffer Register	0x0000_0000
TIMER2_PWMCMPBUF	TMR23_BA+0x74	R	Timer2 PWM Comparator Buffer Register	0x0000_0000
TIMER3_PWMCMPBUF	TMR23_BA+0x174	R	Timer3 PWM Comparator Buffer Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPBUF**	**PWM Comparator Buffer Register (Read Only)** Used as CMP active register.

Timer PWM Interrupt Flag Accumulator Register (TIMERx_PWMIFA)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMIFA	TMR01_BA+0xA8	R/W	Timer0 PWM Interrupt Flag Accumulator Register	0x0000_0000
TIMER1_PWMIFA	TMR01_BA+0x1A8	R/W	Timer1 PWM Interrupt Flag Accumulator Register	0x0000_0000
TIMER2_PWMIFA	TMR23_BA+0xA8	R/W	Timer2 PWM Interrupt Flag Accumulator Register	0x0000_0000
TIMER3_PWMIFA	TMR23_BA+0x1A8	R/W	Timer3 PWM Interrupt Flag Accumulator Register	0x0000_0000

31	30	29	28	27	26	25	24
IFAEN	Reserved	IFASEL		Reserved			STPMOD
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
IFACNT							
7	6	5	4	3	2	1	0
IFACNT							

Bits	Description	
[31]	**IFAEN**	**PWM Interrupt Flag Accumulator Enable Bit** 0 = PWM interrupt flag accumulator function Disabled. 1 = PWM interrupt flag accumulator function Enabled.
[30]	**Reserved**	Reserved.
[29:28]	**IFASEL**	**PWM Interrupt Flag Accumulator Source Select** 01 = Accumulate at each PWM period point. 10 = Accumulate at each PWM up-count compared point.
[27:25]	**Reserved**	Reserved.
[24]	**STPMOD**	**PWM Accumulator Stop Mode Enable Bit** 0 = PWM interrupt accumulator event to stop counting Disabled. 1 = PWM interrupt accumulator event to stop counting Enabled.
[23:16]	**Reserved**	Reserved.
[15:0]	**IFACNT**	**PWM Interrupt Flag Accumulator Counter** This field sets the count number which defines (IFACNT+1) times of specify PWM interrupt occurs to set IFAIF bit to request the PWM accumulator interrupt. PWM accumulator flag (IFAIF) will be set in every (IFACNT+1) times during the the PWM counter operation.

Timer PWM Accumulator Interrupt Status Register (TIMERx_PWMAINTSTS)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMAINTSTS	TMR01_BA+0xAC	R/W	Timer0 PWM Accumulator Interrupt Flag Register	0x0000_0000
TIMER1_PWMAINTSTS	TMR01_BA+0x1AC	R/W	Timer1 PWM Accumulator Interrupt Flag Register	0x0000_0000
TIMER2_PWMAINTSTS	TMR23_BA+0xAC	R/W	Timer2 PWM Accumulator Interrupt Flag Register	0x0000_0000
TIMER3_PWMAINTSTS	TMR23_BA+0x1AC	R/W	Timer3 PWM Accumulator Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**IFAIF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**IFAIF**	**PWM Interrupt Flag Accumulator Interrupt Flag** This bit is set by hardware when the accumulator value reaches (IFACNT+1) **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** If APDMAEN (TIMERx_PWMAPDMACTL[0]) is set, this bit will be auto clear after PDMA transfer done.

Timer PWM Accumulator Interrupt Enable Register (TIMERx_PWMAINTEN)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMAINTEN	TMR01_BA+0xB0	R/W	Timer0 PWM Accumulator Interrupt Enable Register	0x0000_0000
TIMER1_PWMAINTEN	TMR01_BA+0x1B0	R/W	Timer1 PWM Accumulator Interrupt Enable Register	0x0000_0000
TIMER2_PWMAINTEN	TMR23_BA+0xB0	R/W	Timer2 PWM Accumulator Interrupt Enable Register	0x0000_0000
TIMER3_PWMAINTEN	TMR23_BA+0x1B0	R/W	Timer3 PWM Accumulator Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**IFAIEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**IFAIEN**	**PWM Interrupt Flag Accumulator Interrupt Enable Bit** 0 = Interrupt Flag Accumulator interrupt Disabled. 1 = Interrupt Flag Accumulator interrupt Enabled.

Timer PWM Accumulator PDMA Control Register (TIMERx_PWMAPDMACTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMAPDMACTL	TMR01_BA+0xB4	R/W	Timer0 PWM Accumulator PDMA Control Register	0x0000_0000
TIMER1_PWMAPDMACTL	TMR01_BA+0x1B4	R/W	Timer1 PWM Accumulator PDMA Control Register	0x0000_0000
TIMER2_PWMAPDMACTL	TMR23_BA+0xB4	R/W	Timer2 PWM Accumulator PDMA Control Register	0x0000_0000
TIMER3_PWMAPDMACTL	TMR23_BA+0x1B4	R/W	Timer3 PWM Accumulator PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**APDMAEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**APDMAEN**	**PWM Accumulator PDMA Enable Bit** 0 = PWM interrupt accumulator event to trigger PDMA transfer Disabled. 1 = PWM interrupt accumulator event to trigger PDMA transfer Enabled.

6.12 Low Power Timer Controller (LPTMR)

6.12.1 Overview

The low power timer controller includes two 32-bit timers, LPTMR0 ~ LPTMR1, allowing user to easily implement a timer control for applications. The low power can perform functions, such as frequency measurement, delay timing, clock generation, and event counting by external input pins, and interval measurement by external capture pins.

The low power timer controller also provides two PWM generators. Each PWM generator supports one PWM output and two selectable PWM output channels (LPTMRx or LPTMRx_EXT). The output state of PWM output pin can be control by polarity control, output enable control and output channel select.

6.12.2 Features

6.12.2.1 Low Power Timer Function Features

- Two sets of 32-bit timers, each timer having one 24-bit up counter and one 8-bit prescale counter
- Independent clock source for each timer
- Provides one-shot, periodic, toggle-output and continuous counting operation modes
- 24-bit up counter value is readable through CNT (LPTMRx_CNT[23:0])
- Support 3-bit capture input noise filter
- Supports event counting function
- 24-bit capture value is readable through CAPDAT (LPTMRx_CAP[23:0])
- Supports external capture pin (LPTMRx_EXT) event for interval measurement
- Supports external capture pin (LPTMRx_EXT) event to reset 24-bit up counter
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated
- Supports time-out interrupt signal or capture interrupt signal to trigger LPADC0, , LPI2C0, LPSPI0, LPUART, , and LPPDMA function
- Supports internal capture triggered while internal ACMP output signal transition
- Supports event counting source from ACMP0 output, ACMP1 output or ACMP2 output
- Supports internal capture triggered source from ACMP0 output, ACMP1 output and ACMP2 output

6.12.2.2 PWM Function Features

- Supports PWM generator with two selectable output channels
- Supports 16-bit PWM counter
  - Up count operation type
  - One-shot or auto-reload counter operation mode
- Supports 8-bit prescale from 1 to 256
- Supports 16-bit compare register and period register and double buffer for period register and compare register
- Supports tri-state enable and polarity control for each PWM selectable output channels
- Supports interrupt on the following events:
  - PWM period point, up-count compared point events

- Supports wake-up when interrupt occurs when clock source is LXT or LIRC
- PWM can generate output in Power-down mode
  - Supports trigger LPADC, LPPDMA, and on the following events:
  - PWM period point and up-count compared point events
- Supports PWM output accumulator event to trigger LPPDMA transfer and LPADC
- Supports PWM output accumulator event to stop PWM counting
- Support automatic operation mode

6.12.3 Block Diagram

The low power timer controller block diagram and clock control are shown as follows.

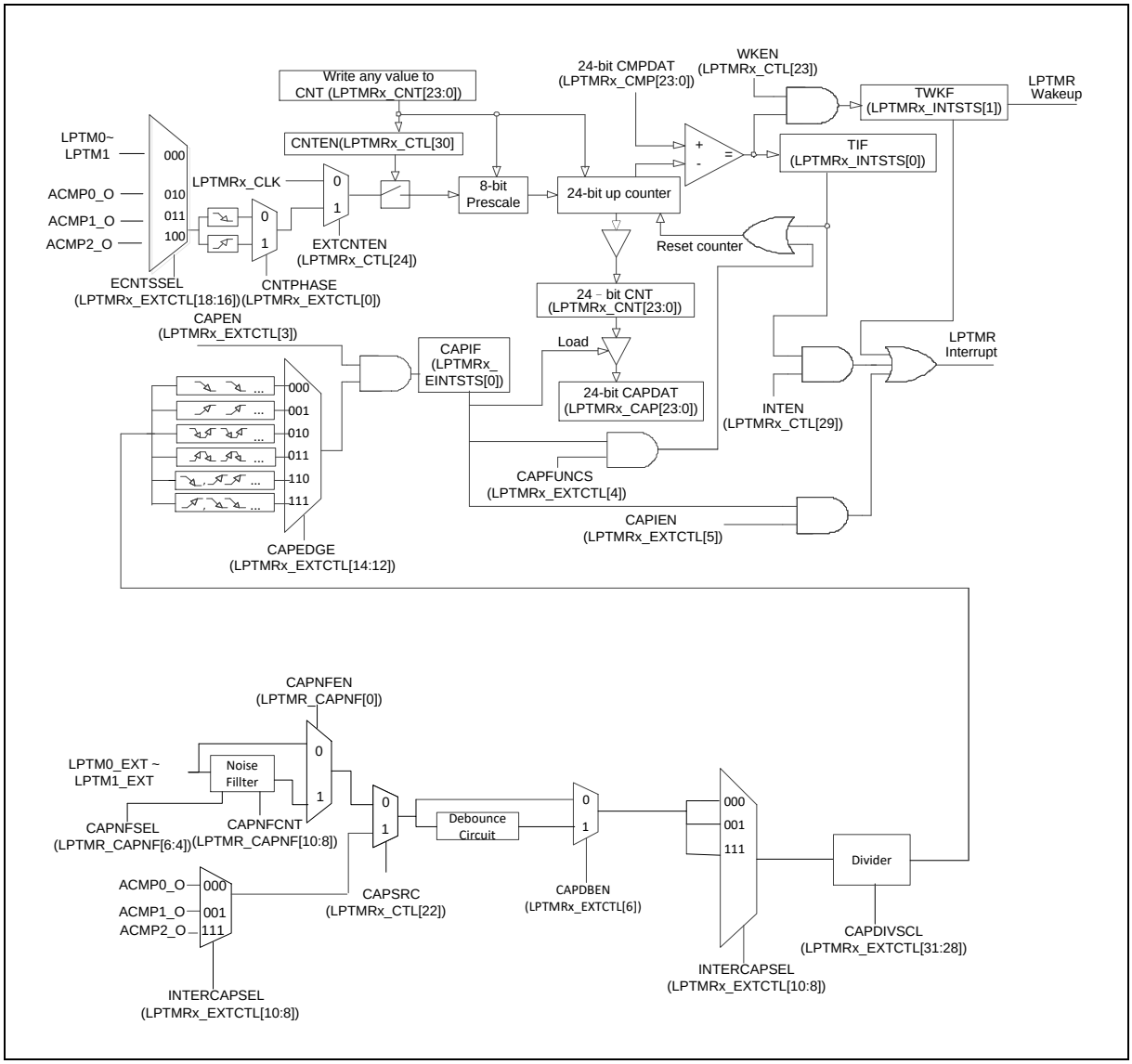


Figure 6.12-1 Low Power Timer Controller Block Diagram

Set FUNCSEL (LPTMRx_CTL[15]) 0 to enable timer mode. The clock source of LPTMR0 ~ LPTMR1 in timer mode can be enabled in LPTMRxCKEN (LPSCC_CLKEN0[21:20]) and selected as different frequency in LPTMR0SEL (LPSCC_CLKSEL0 [10:8]) for LPTMR0, LPTMR1SEL (LPSCC_CLKSEL0 [14:12]) for LPTMR1.

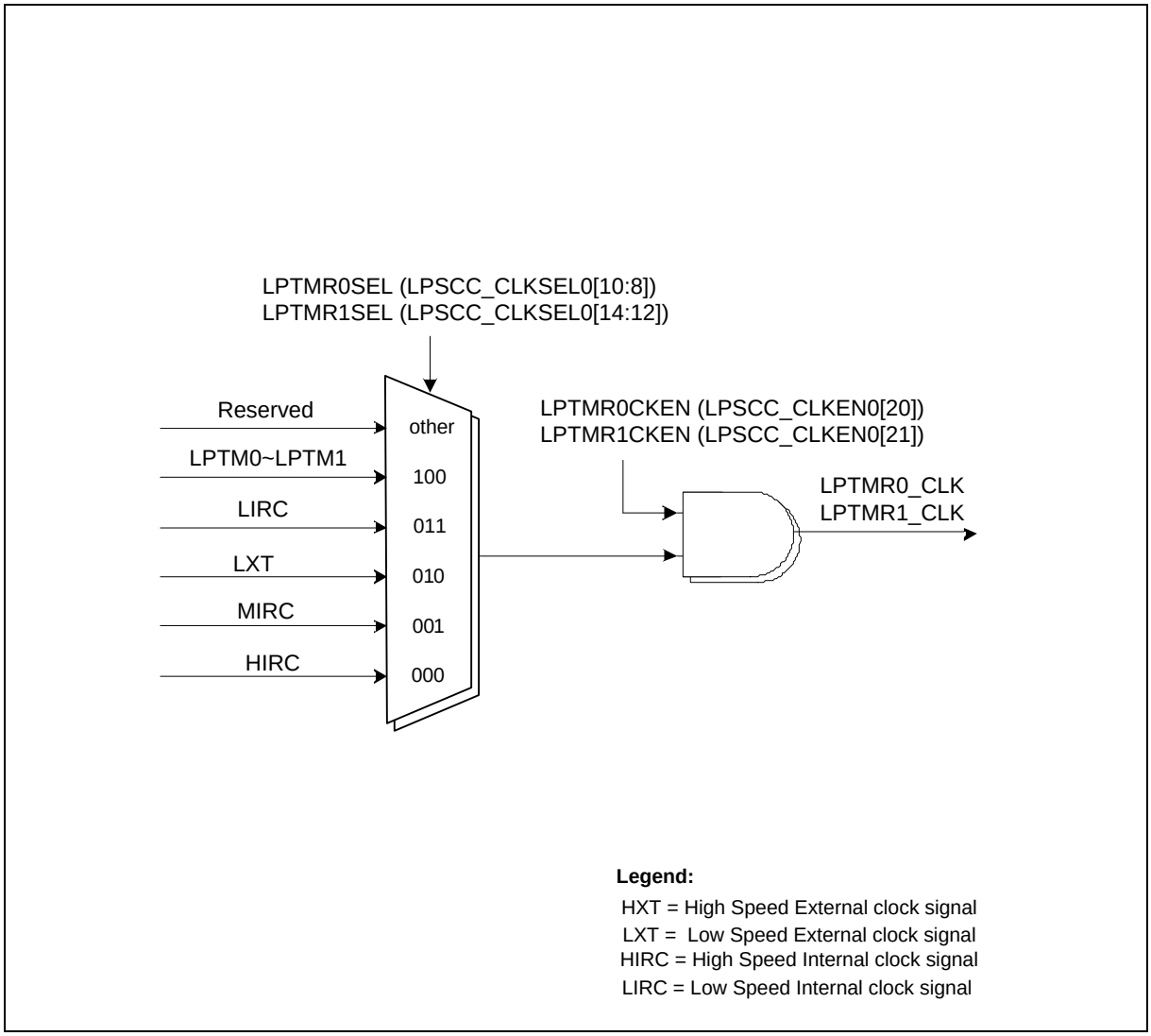


Figure 6.12-2 Clock Source of Low Power Timer Controller

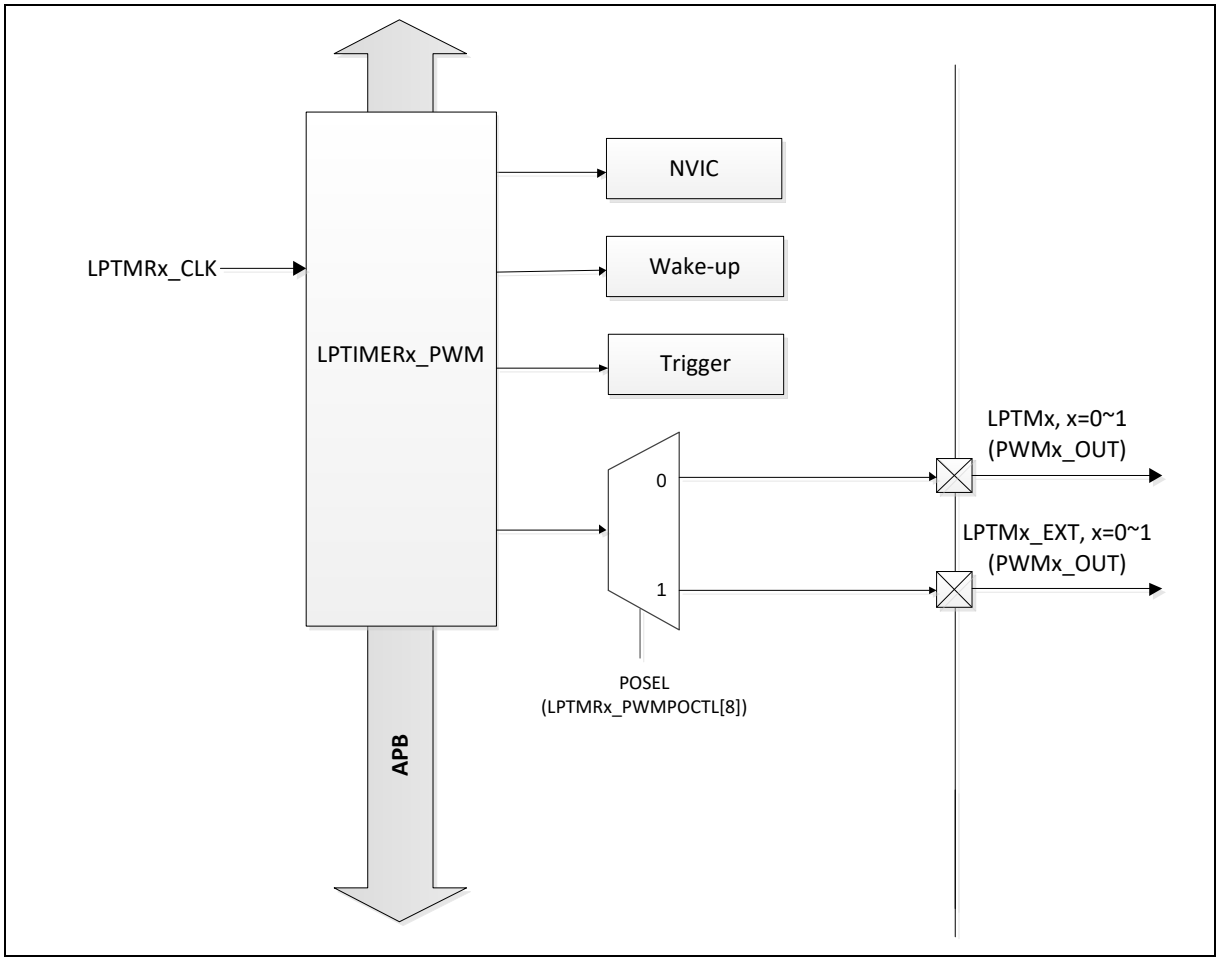


Figure 6.12-3 PWM Generator Overview Block Diagram

Set FUNCSEL (LPTMRx_CTL[15]) 1 to enable PWM mode. The clock source of LPTMR0 ~ LPTMR1 in PWM mode can be enabled in LPTMRxCKEN (LPSCC_CLKEN0 [21:20]). PWM system clock and counter clock source are from LPTMRx_CLK.

Figure 6.12-4 illustrates the PWM architecture that supports one PWM output and two selectable LPTMRx or LPTMRx_EXT output channels in each PWM generator.

When PERIOD (LPTMRx_PWMPERIOD[15:0]) is equal to CMP (LPTMRx_PWMCMPDAT[15:0]), relative events will be generated. These events are passed to corresponding generators to generate PWM pulse (Pulse Generator), interrupt signal (Interrupt Generator), wake-up (Wake-up Generator) and trigger signal (Trigger Generator) for LPADC,LPI2C,LPSPI,LPUART, LPPDMA to start conversion. Output Control block is used to decide PWM pulse output.

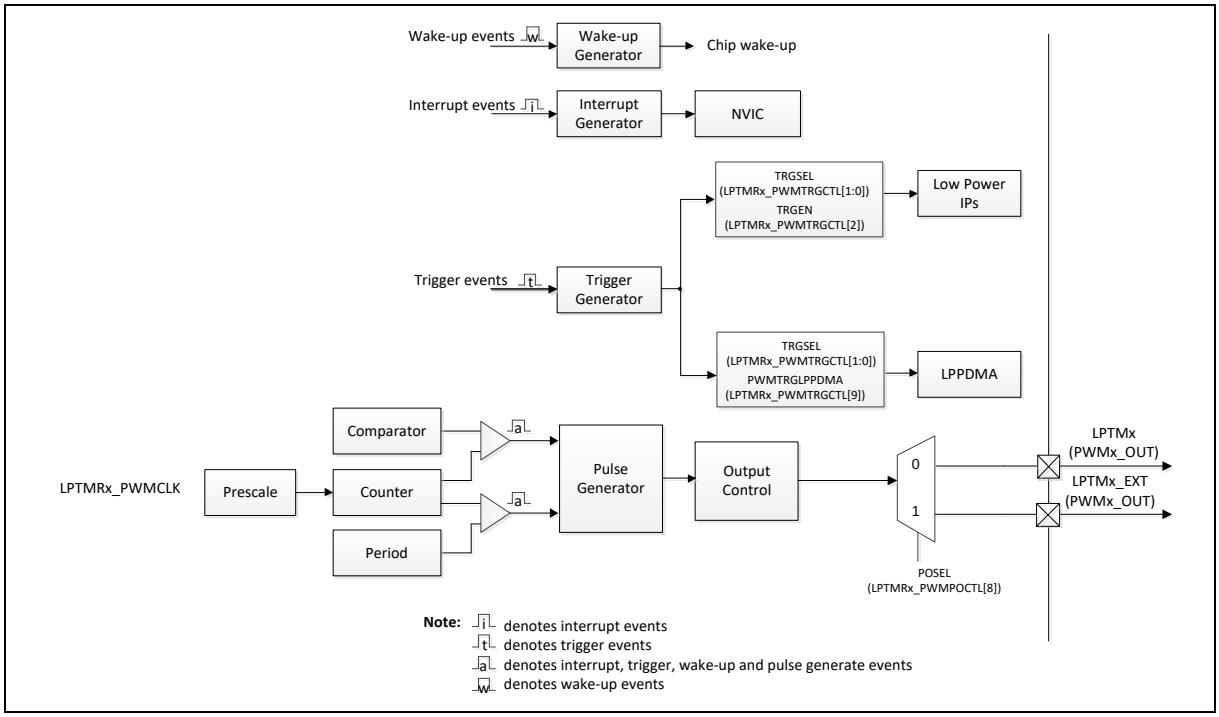


Figure 6.12-4 PWM Architecture Diagram

6.12.4 Basic Configuration

Set FUNCSEL (LPTMRx_CTL[15]) 0 to enable timer mode. The clock source of LPTMR0 ~ LPTMR1 in timer mode can be enabled in LPTMRxCKEN (LPSCC_CLKEN0 [21:20]) and selected as different frequency in LPTMR0SEL (LPSCC_CLKSEL0[10:8]) for LPTMR0, LPTMR1SEL (LPSCC_CLKSEL0[14:12]) for LPTMR1.

Set FUNCSEL (LPTMRx_CTL[15]) 1 to enable PWM mode. The clock source of LPTMR0 ~ LPTMR1 in PWM mode can be enabled in LPTMRxCKEN (LPSCC_CLKEN0 [21:20]). PWM system clock and counter clock source are from LPTMRx_CLK.

6.12.4.1 LPTMR01 Basic Configurations

- Clock Source Configuration
  - Enable LPTMR0 peripheral clock in LPTMR0CKEN (LPSCC_CLKEN0[20]).
  - Enable LPTMR1 peripheral clock in LPTMR1CKEN (LPSCC_CLKEN0[21]).
- Reset Configuration
  - Reset LPTMR0 controller in LPTMR0RST (LPSCC_IPRST0[20]).
  - Reset LPTMR1 controller in LPTMR1RST (LPSCC_IPRST0[21]).

6.12.5 Low Power Timer Functional Description

6.12.5.1 Low Power Timer Interrupt Flag

The low power timer controller supports the following interrupt flags: TIF (LPTMRx_INTSTS[0]) which is set while low power timer counter value CNT (LPTMRx_CNT[23:0]) matches the low power timer compared value CMPDAT (LPTMRx_CMP[23:0]), and CAPIF (LPTMRx_EINTSTS[0]), which is set when the transition on the external LPTMRx_EXT pin, or the internal ACMP0_O, ACMP1_O, ACMP2_O signal is associated with CAPEDGE (LPTMRx_EXTCTL[14:12]) setting. The TWKF (LPTMRx_INTSTS[1]) bit indicates the interrupt wake-up flag status of timer. User can set CAPSRC (LPTMRx_CTL[22]) and INTERCAPSEL (LPTMRx_EXTCTL[10:8]) to select capture source. Set WKEN

(LPTMRx_CTL[23] to 1 can use wake-up function.

6.12.5.2 Low Power Timer Counting Mode

The low power timer controller provides four timer counting modes: one-shot, periodic, toggle-output and continuous counting operation modes:

One–shot Mode

If the low power timer controller is configured at one-shot mode (LPTMRx_CTL[28:27] is 00) and CNTEN (LPTMRx_CTL[30]) is set, the low power timer counter starts up counting. Once the CNT (LPTMRx_CNT[23:0]) value reaches CMPDAT (LPTMRx_CMP[23:0]) value, the TIF (LPTMRx_INTSTS[0]) will be set to 1, CNT value and CNTEN bit is cleared automatically by low power timer controller then timer counting operation stops. In the meantime, if the INTEN (LPTMRx_CTL[29]) is enabled, the low power timer interrupt signal is generated and sent to NVIC to inform CPU also.

User can monitor the RSTACT (LPTMRx_CNT[31]) to ensure counter reset operation active and disable ICE debug mode acknowledgement effects LPTMR counting by setting ICEDEBUG (LPTMRx_CTL[31]) to 1.

Periodic Mode

If the low power timer controller is configured at periodic mode (LPTMRx_CTL[28:27] is 01) and CNTEN (LPTMRx_CTL[30]) is set, the low power timer counter starts up counting. Once the CNT (LPTMRx_CNT[23:0]) value reaches CMPDAT (LPTMRx_CMP[23:0]) value, the TIF (LPTMRx_INTSTS[0]) will be set to 1, CNT value will be cleared automatically by low power timer controller and low power timer counter operates counting again. In the meantime, if the INTEN (LPTMRx_CTL[29]) bit is enabled, the low power timer interrupt signal is generated and sent to NVIC to inform CPU also. In this mode, the low power timer controller operates counting and compares with CMPDAT value periodically until the CNTEN bit is cleared by user.

User can set PERIOSEL (LPTMRx_CTL[20]) to select Low Power Timer behavior.

Toggle-Output Mode

If the low power timer controller is configured at toggle-output mode (LPTMRx_CTL[28:27] is 10) and CNTEN (LPTMRx_CTL[30]) is set, the low power timer counter starts up counting. The counting operation of toggle-output mode is almost the same as periodic mode, except toggle-output mode has associated LPTMR0 ~ LPTMR1 or LPTMR0_EXT ~ LPTMR1_EXT pin to output signal while specify TIF (LPTMRx_INTSTS[0]) is set. Use can set TGLPINSEL (LPTMRx_CTL[21]) to choose LPTMRx or LPTMRx_EXT as toggle-output pin. Thus, the toggle-output signal on LPTMR0 ~ LPTMR1 pin is high and changing back and forth with 50% duty cycle.

Continuous Counting Mode

If the low power timer controller is configured at continuous counting mode (LPTMRx_CTL[28:27] is 11) and CNTEN (LPTMRx_CTL[30]) is set, the low power timer counter starts up counting. Once the CNT (LPTMRx_CNT[23:0]) value reaches CMPDAT (LPTMRx_CMP[23:0]) value, the TIF (LPTMRx_INTSTS[0]) will be set to 1 and CNT value keeps up counting. In the meantime, if the INTEN (LPTMRx_CTL[29]) is enabled, the low power timer interrupt signal is generated and sent to NVIC to inform CPU also. User can change different CMPDAT value immediately without disabling timer counting and restarting timer counting in this mode.

For example, CMPDAT value is set as 80, first. The TIF will set to 1 when CNT value is equal to 80, timer counter is kept counting and CNT value will not goes back to 0, it continues to count 81, 82, 83, … to (2^{24} -1), 0, 1, 2, 3, … to (2^{24} -1) again and again. Next, if user programs CMPDAT value as 200 and clears TIF, the TIF will set to 1 again when CNT value reaches to 200. At last, user programs CMPDAT as 500 and clears TIF, the TIF will set to 1 again when CNT value reaches to 500.

In this mode, the low power timer counting is continuous. So this operation mode is called as continuous counting mode.

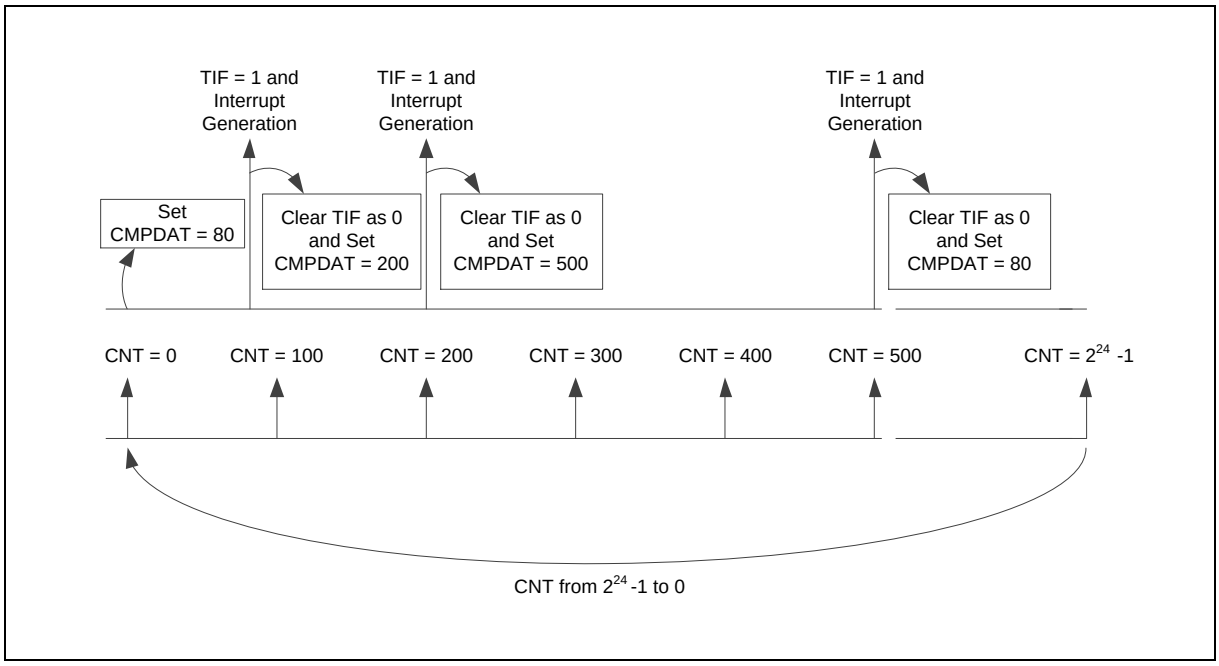


Figure 6.12-5 Continuous Counting Mode

6.12.5.3 Event Counting Mode

The low power timer controller also provides an application which can count the input event from LPTMRx (x= 0~1) pin and the number of event will reflect to CNT (LPTMRx_CNT[23:0]) value. It is also called as event counting function. In this function, EXTCNTEN (LPTMRx_CTL[24]) should be set and the low power timer peripheral clock source should be set as ECLK.

If ECNTSSEL (LPTMRx_EXTCTL[18:16]) is 0, the event counter source is from external LPTMRx pin. User can enable or disable LPTMRx pin de-bounce circuit by setting CNTDBEN (LPTMRx_EXTCTL[7]). The input event frequency should be less than 1/3 ECLK if LPTMRx pin de-bounce disabled or less than 1/8 ECLK if LPTMRx pin de-bounce enabled to assure the returned CNT value is correct, and user can also select edge detection phase of LPTMRx pin by setting CNTPHASE (LPTMRx_EXTCTL[0]) bit.

In event counting mode, the low power timer counting operation mode can be selected as one-shot, periodic and continuous counting mode to counts the counter value CNT (LPTMRx_CNT[23:0]) for LPTMRx. .

6.12.5.4 Capture Mode

The event capture function is used to load CNT (LPTMRx_CNT[23:0]) value to CAPDAT (LPTMRx_CAP[23:0]) value while edge transition detected on LPTMRx_EXT (x= 0~1) pin, ACMP, internal clock and external clock. In this mode, CAPFUNCS (LPTMRx_EXTCTL[4]) should be as 0 to trigger event capture function and the low power timer peripheral clock source should be set as ECLK.

If CAPSRC (LPTMRx_CTL[22]) is 0, the capture event is triggered by LPTMRx_EXT pin transition. The LPTMRx_EXT signals will be filtered by a 3-bit noise filter. User can enable the noise filter function by CAPNFEN (LPTMRx_CAPNF[0]) bit, and noise filter sampling clock can be selected by setting CAPNFSEL (LPTMRx_CAPNF[6:4]) bits to fit different noise properties. Moreover, by setting the CAPNFCNT (LPTMRx_CAPNF[10:8]) bits, user can define by how many sampling clock cycles a filter will recognize the effective edge of the LPTMRx_EXT signal.

User can also enable or disable LPTMRx_EXT pin de-bounce circuit by setting CAPDBEN (LPTMRx_EXTCTL[6]). The transition frequency of LPTMRx_EXT pin should be less than 1/3 ECLK if LPTMRx_EXT pin de-bounce disabled or less than 1/8 ECLK if LPTMRx_EXT pin de-bounce enabled

to assure the capture function can be work normally, and user can also select edge transition detection of LPTMRx_EXT pin by setting CAPEDGE (LPTMRx_EXTCTL[14:12]).

In event capture mode, user does not consider what timer counting operation mode is selected, the capture event occurred only if edge transition on LPTMRx_EXT pin is detected.

User can enable CAPIEN (LPTMRx_EXTCTL[5]) to use capture interrupt function. When the LPTMRx_EXT edge transition meets setting, CAPIF is high.

User must consider the Low Power Timer will keep register LPTMRx_CAP unchanged and drop the new capture value, if the CPU does not clear the CAPIF status.

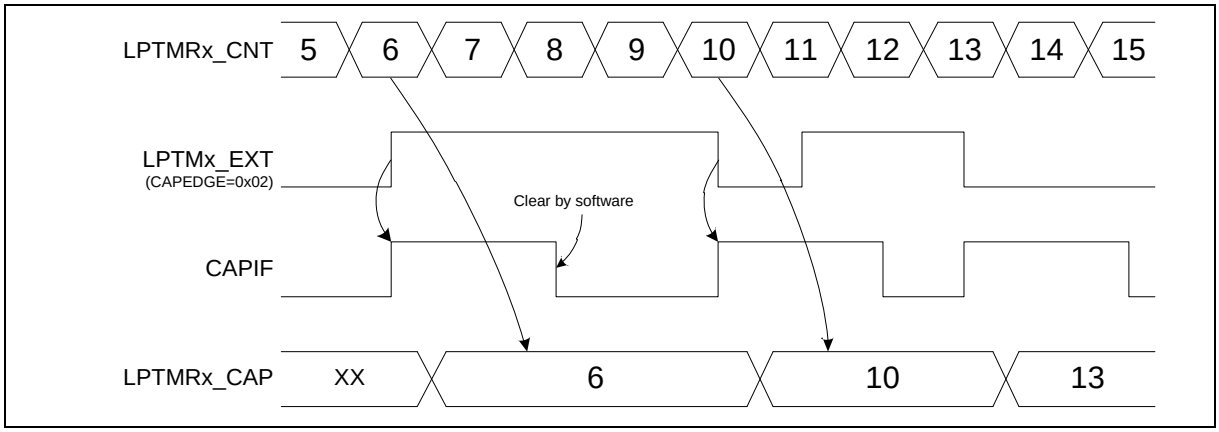


Figure 6.12-6 External Capture Mode

If CAPSRC (LPTMRx_CTL[22]) is 1, set INTERCAPSEL (LPTMRx_EXTCTL[10:8]) to choose different capture source. The capture event can be triggered by internal output signal transition on ACMP0 if INTERCAPSEL (LPTMRx_EXTCTL[10:8]) is 000, or ACMP1 if INTERCAPSEL is 001 or ACMP2 if INTERCAPSEL is 111 ; User can switch capture source only when original source and target source is all off. For example, if user wants to switch ACMP0 or ACMP1, both of capture source should be set off first.

The capture source can be divided by capture divider. Use can set CAPDIVSCL (LPTMRx_EXTCTL[31:28]) to select different divider number. Be aware that timer clock frequency must be four times than capture source at least.

6.12.5.5 Reset Counter Mode

The low power timer controller also provides reset counter function to reset CNT (LPTMRx_CNT[23:0]) value while capture event is generated. In this mode, CAPFUNCS (LPTMRx_EXTCTL[4]) should be 1.User must set CAPSRC and INTERCAPSEL to select LPTMRx_EXT transition, internal ACMPx output signal to trigger reset counter value.

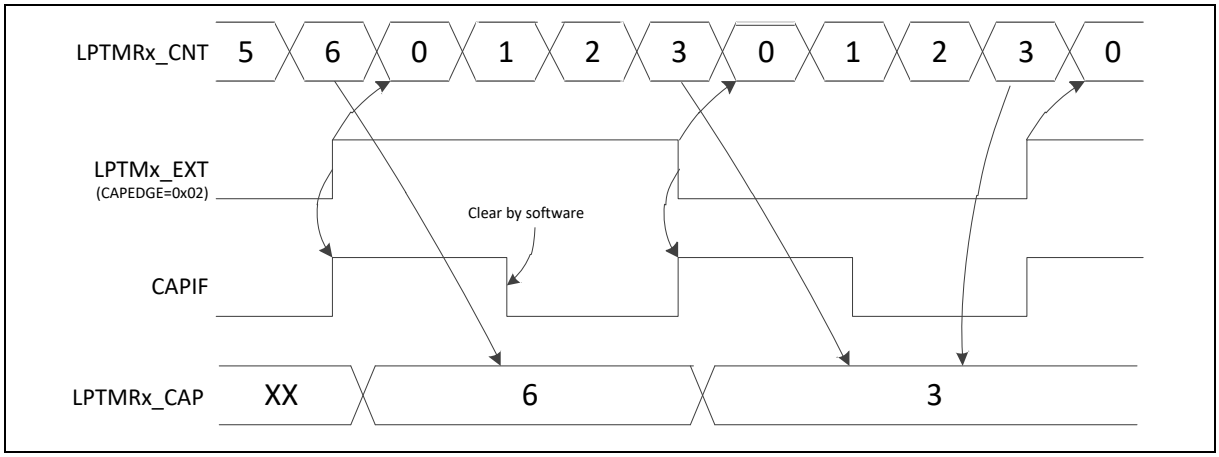


Figure 6.12-7 Reset Counter Mode

6.12.5.6 Low Power Timer Trigger Function

The low power timer controller provides timer time-out interrupt or capture interrupt to trigger LPADC, LPI2C, LPSPI, LPUART and LPPDMA. If TRGSSEL (LPTMRx_TRGCTL[0]) is 0, time-out interrupt signal is used to trigger LPADC, LPI2C, LPSPI, LPUART and LPPDMA. If TRGSSEL (LPTMRx_TRGCTL[0]) is 1, capture interrupt signal is used to trigger LPADC, LPI2C, LPSPI, LPUART and LPPDMA.

When the TRGEN (LPTMRx_TRGCTL[1]) is set, if the low power timer interrupt signal is generated, the low power timer controller will generate a trigger other low power IPs.

When the TRGLPPDMA (LPTMRx_TRGCTL[4]) is set, if the low power timer interrupt signal is generated, the low power timer controller will trigger LPPDMA.

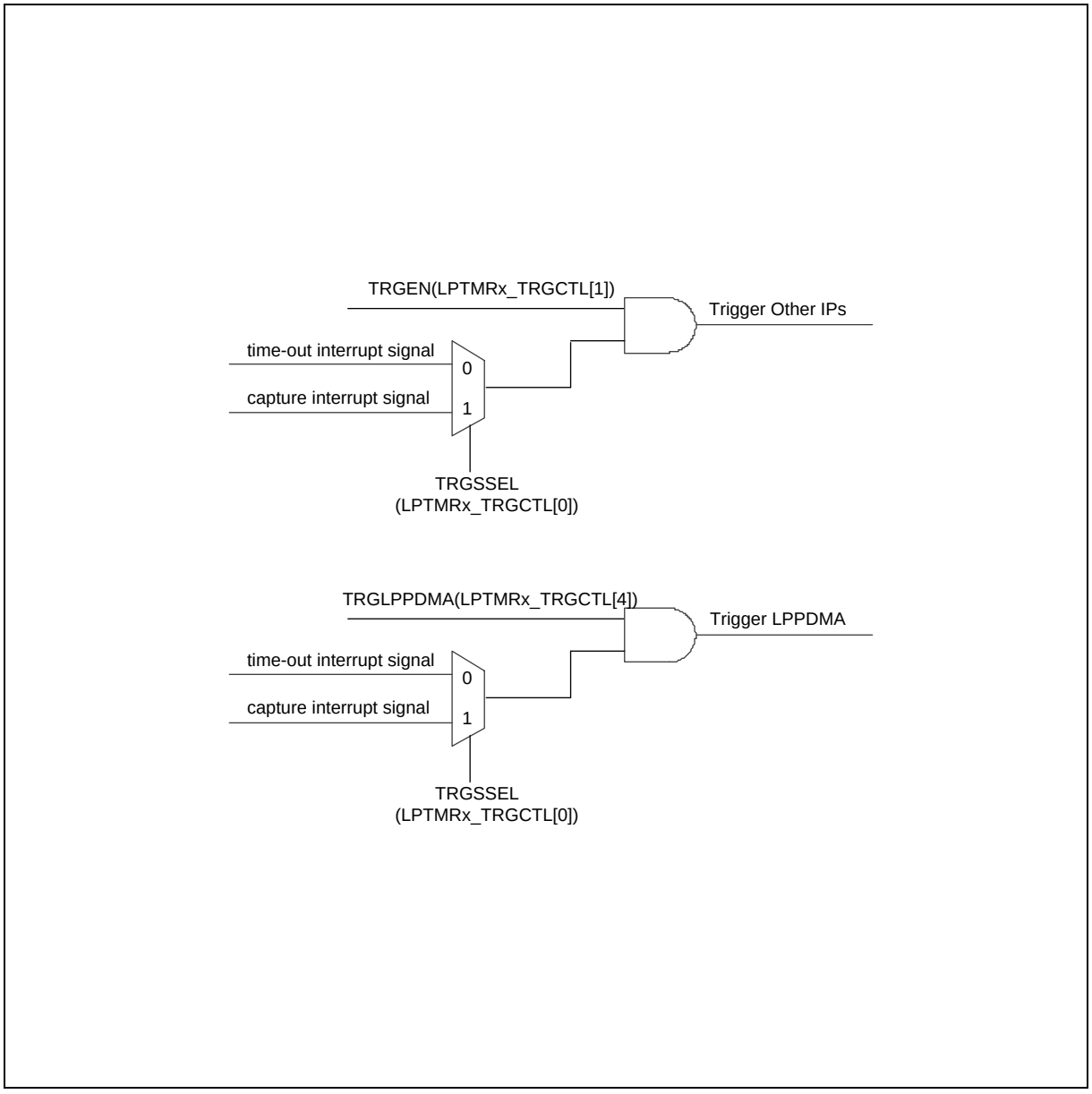


Figure 6.12-8 Internal Low Power Timer Trigger

When both the TRGLPPDMA (LPTMRx_TRGCTL[4]) and TRGSSEL (LPTMRx_TRGCTL[0]) are 1, LPTMRx_CAP will change when encountering timer capture edge no matter whether CPU clear CAPIF status or not. In Trigger other Low Power IPs function, user must consider that the Low Power Timer will keep register LPTMRx_CAP unchanged if the CPU does not clear the CAPIF status.

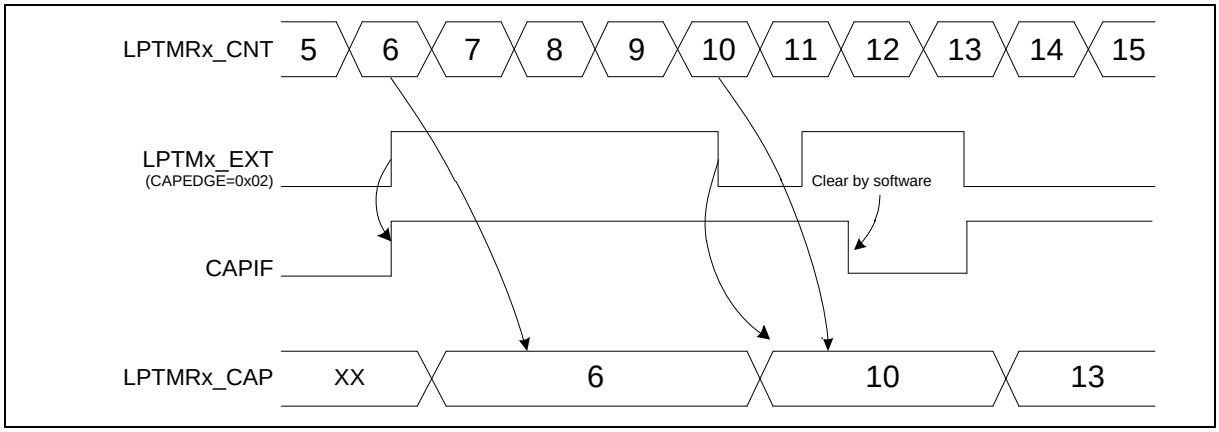


Figure 6.12-9 Capture interrupt trigger LPPDMA

6.12.5.7 Auto-operation Mode

The LPTMR controller supports auto-operation mode to make LPTMR can trigger other low power IPs when chip enters NPD0/1/2/3/4 mode. Configure TRGEN(LPTMRx_TRGCTL[1]) and TRGSSEL(LPTMR_TRGCTL[0]) to enable auto-operation mode first. There are two functions to trigger LPPDMA. One is Counting Mode, and the other is Capture Mode. User can change mode by setting 1 or 0 to TRGSSEL(LPTMR_TRGCTL[0]). LPTMR can broadcast a 4T pulse to other low power IPs when TRGSSEL(LPTMR_TRGCTL[0]) condition meets, then other low power IPs will receive the pulse. Especially for LPPDMA, LPTMR needs to be configured TRGLPPDMA(LPTMRx_TRGCTL[4]) and TRGSSEL(LPTMR_TRGCTL[0]). LPTMR and LPPDMA have a handshake mechanism. LPTMR send a request to LPPDMA, and then LPPDMA will send an ack. This mechanism is different from other low power IPs. Other low power IPs don't need to send an ack to LPTMR.

Trigger Low Power IPs:

1. Enable LPTMR peripheral clock.

Set PDCLKEN(LPTMRx_CTL[16]) to open the engine clock in NPD0/1/2/3/4 mode. Engine clock should be always on to operate the Timer Function.

2. Configure Trigger Function

Set TRGEN(LPTMRx_TRGCTL[1]) and TRGSSEL(LPTMR_TRGCT[0]) first. LPTMR can use counting mode or capture mode to trigger other low power IPs.

Trigger LPPDMA:

1. Enable LPTMR peripheral clock.

Set PDCLKEN(LPTMRx_CTL[16]) to open the engine clock in NPD0/1/2/3/4 mode. Engine clock should be always on to operate the Timer Function. When LPTMR in counting mode trigger LPPDMA in NPD0/1/2/3/4 mode, AHB1, AHB1 bus matrix, LPPDMA and LPSRAM also need clock to work. But the AHB and APB clock are not always on in NPD0/1/2/3/4. AHB and APB clock is on when time-out flags occurs. Then AHB and APB clock will turn off after LPPDMA transfer done.

2. Configure Trigger Function

Set TRGLPPDMA(LPTMRx_TRGCTL[4]) and TRGSSEL(LPTMR_TRGCTL[0]) first. LPTMR can trigger LPPDMA in Counting mode when time-out flag occurs.In capture mode, LPTMRx_CAP will change when encountering timer capture edge. At the same time, LPPDMA will move data from LPTMR Capture Data Register(LPTMRx_CAP) to LPSRAM.

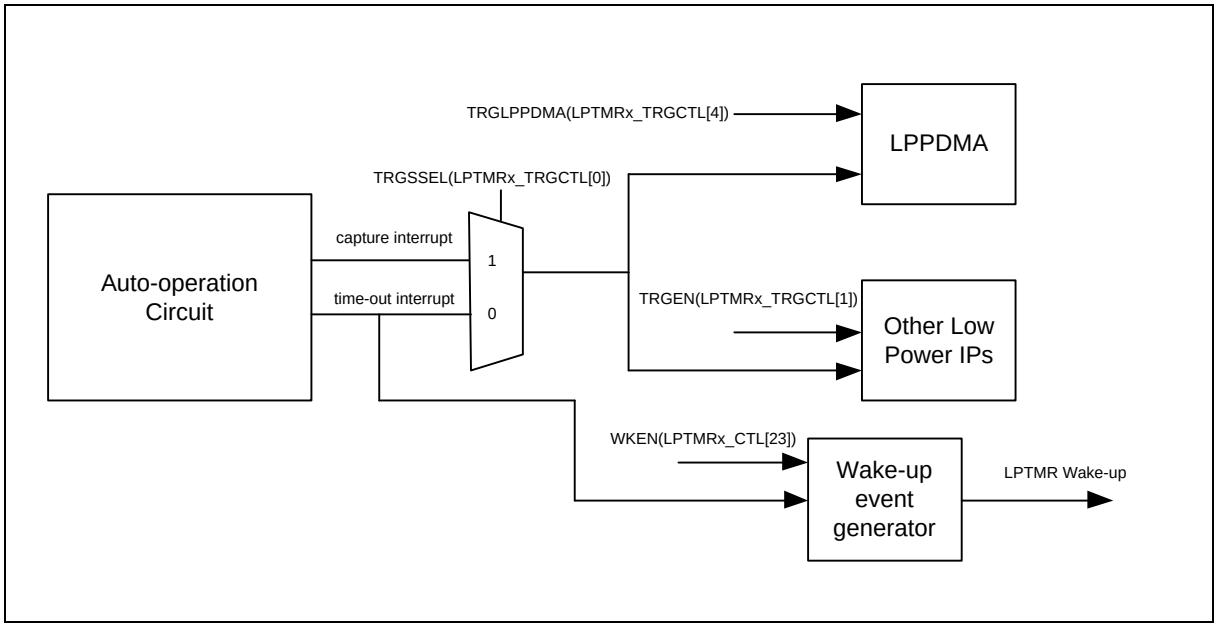


Figure 6.12-11 LPTMR Auto-operation mode Block Diagram

6.12.6 PWM Functional Description

6.12.6.1 PWM Block Diagram

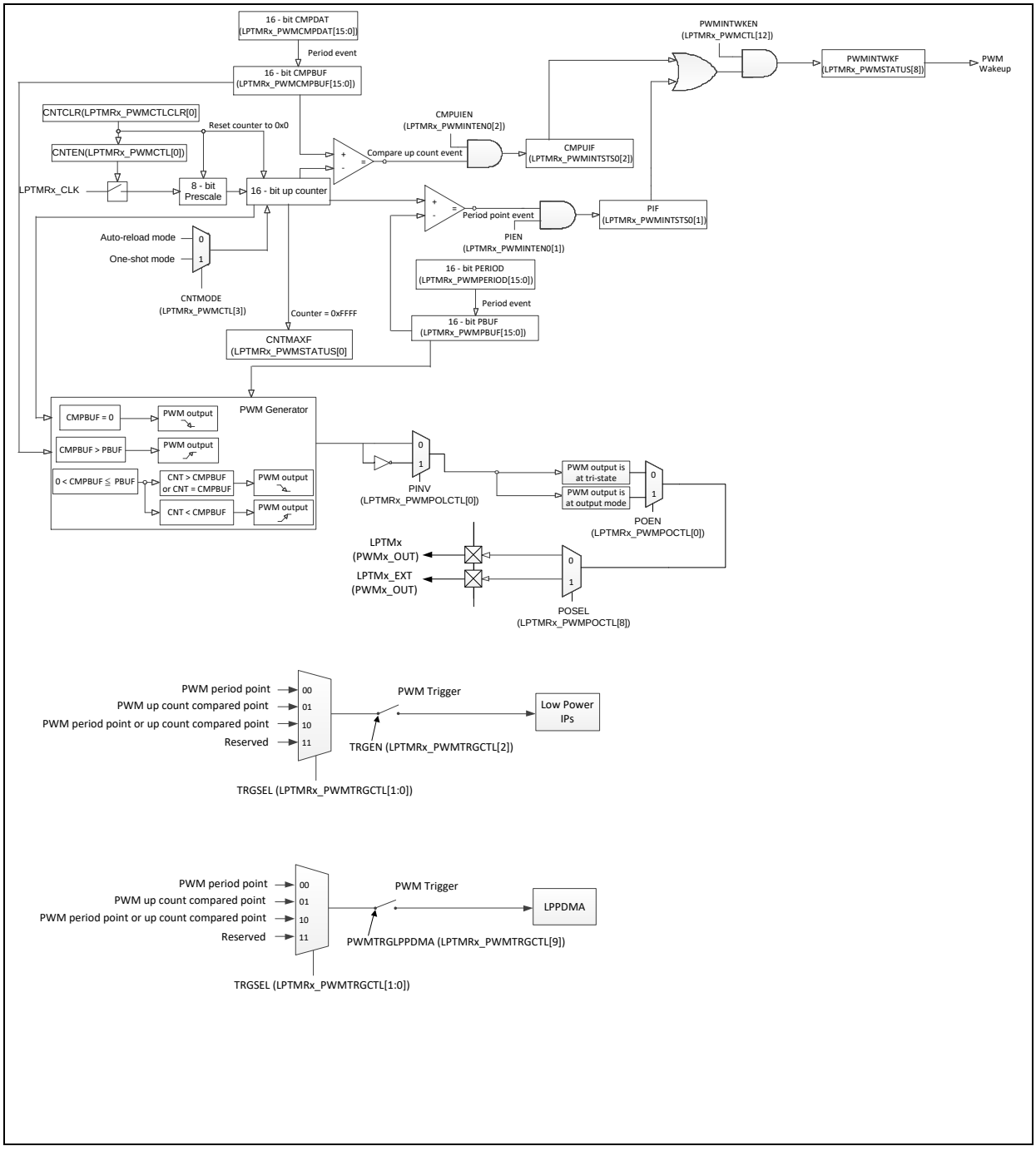


Figure 6.12-10 Low Power Timer PWM Block Diagram

6.12.6.2 PWM Prescale

The PWM prescale is used to divide clock source, and the clock of PWM counter is divided by (CLKPSC+ 1). The prescale is set by CLKPSC (LPTMRx_PWMCLKPSC[7:0]).Figure 6.12-11. shows an example of PWM prescale waveform in up count type.

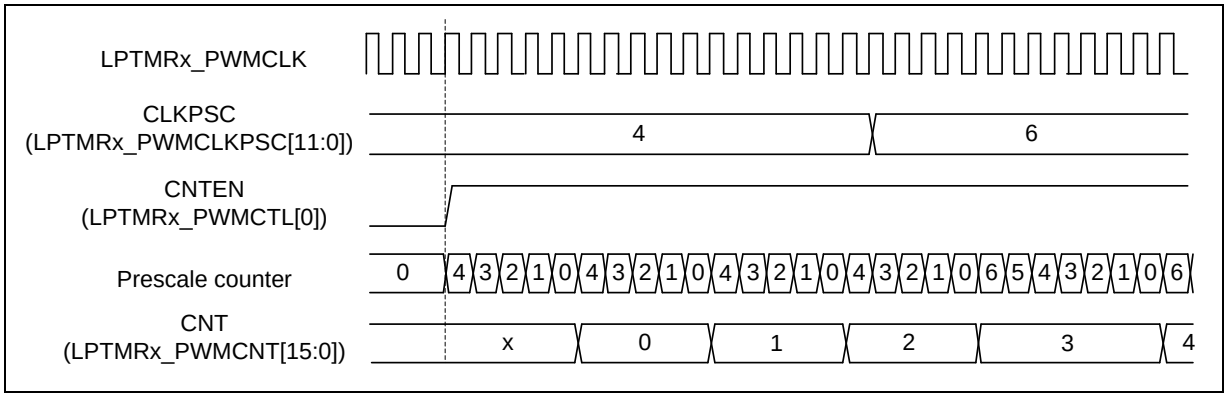


Figure 6.12-11 PWM Prescale Waveform in Up Count Type

6.12.6.3 PWM Counter

The PWM supports up count type

It starts up-counting from 0 to PERIOD (LPTMRx_PWMPERIOD[15:0]). The current counter value can be read from the CNT (LPTMRx_PWMCNT[15:0]). PWM generates a period point event when the counter counts to PERIOD and prescale counts to 0.Figure 6.12-12. shows an example of PWM up count type, where PWM period time is (PERIOD+1) * (CLKPSC+1) * LPTMRx_PWMCLK.

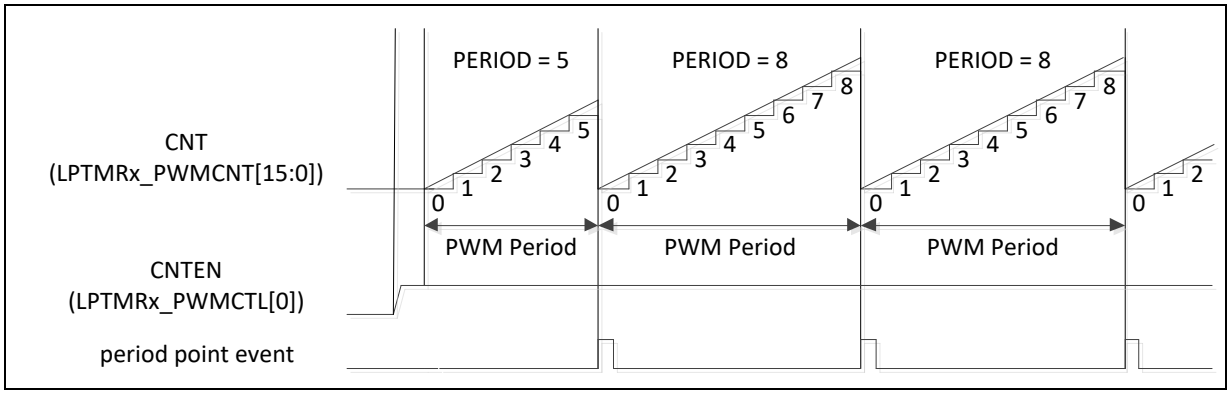


Figure 6.12-12 PWM Up Count Type

6.12.6.4 PWM Counter Operation mode

The PWM counter supports two operation modes: one-shot mode and auto-reload mode. PWM counter will operate in one-shot mode if CNTMODE (LPTMRx_PWMCTL[3]) bit is set to 1, and operate in auto-reload mode if CNTMODE bit is set to 0.

In both modes, CMP (LPTMRx_PWMCMPDAT[15:0]) and PERIOD (LPTMRx_PWMPERIOD[15:0]) should be written first and then set CNTEN (LPTMRx_PWMCTL[0]) bit to 1 to start counter running.

In one-shot mode, PWM counter value will reload to 0 after one PWM period is completed. User can write CMP to continuous one-shot operation to generate next one-shot pulse once no matter current one-shot counter is running or completed.

For example, user writes 0x10 to CMP, writes 0x3F to PERIOD and set CMPUIEN (LPTMRx_PWMINTEN0[2]) and PIEN (LPTMRx_PWMINTEN0[1]) to 1. When counter (CNT) counts to 0x10, CMPUIF (LPTMRx_PWMINTSTS0[2]) is set to 1. Counter keeps counting until 0x3F, PIF (LPTMRx_PWMINTSTS0[1]) is set to 1 then counter reload to 0, and counter stops counting.

Another example is user writes 0x10 to CMP, writes 0x3F to PERIOD and set CMPUIEN and PIEN to 1. If user writes 0x25 to CMP when counter counting, this value will load to CMPBUF (LPTMRx_PWMCMPBUF[15:0]) when counter counts to 0x3F and PIF is set to 1, then counter counts

from 0. If user clears CMPUIF and PIF before counter counts to 0x25, CMPUIF and PIF will be set to 1 when counter counts to 0x25 and 0x3F, respectively.

In auto-reload mode, PWM counter is continuous running with current active PERIOD and CMP. If user sets PERIOD to zero in auto-reload mode, PWM counter value will reload to 0 after one PWM period is completed.

For example, user writes 0x10 to CMP, writes 0x3F to PERIOD. CMPUIEN and PIEN are set to 1. When counter counts to 0x10, CMPUIF is set to 1. Counter keeps counting until 0x3F, PIF is set to 1, and counter will restart counting from 0. If user clear CMPUIF and PIF before counter counts to 0x10 (2nd counts), CMPUIF will equal to 1 and PIF will equal to 1 when counter counts to 0x10 and 0x3F, respectively.

When user writes 0 to PERIOD and this value will be loaded to PBUF (LPTMRx_PWMPBUF[15:0]). This action will make counter be set to 0 and stops counting until user rewrites another value (except 0) to PERIOD and this value loads to PBUF, counter counts from 0.

The one-shot and auto-reload mode comparison is shown as Table 6.12-4.

Counter Mode Functions	One-Shot	Auto-Reload
CNTMODE (LPTMRx_PWMCTL[3])	1	0
CNT = PERIOD (PERIOD is not 0)	Stops counting	Keeps counting from 0 to PERIOD (Periodic)
PERIOD = 0 and loads to PBUF, then write another value (Except 0) to PERIOD	CNT stops counting even though user writes new PERIOD	CNT = 0 (reset value), CNT stops counting until PERIOD and PBUF is not equal to 0

Table 6.12-4 One-shot and Auto-reload Functions

Note: PERIOD (LPTMRx_PWMPERIOD[15:0]) and CMP (LPTMRx_PWMCMPDAT[15:0]) will be loaded to PBUF (LPTMRx_PWMPBUF[15:0]) and CMPBUF (LPTMRx_PWMCMPBUF[15:0]), respectively when one PWM period completes. The first CMP and PERIOD will be loaded to CMPBUF and PBUF, respectively, after CNTEN (LPTMRx_PWMCTL[0]) is 1.

6.12.6.5 PWM Comparator

The CMP (LPTMRx_PWMCMPDAT[15:0]) is comparator register of PWM. The CMP value is continuously compared to the corresponding counter value. When the counter is equal to CMP, PWM generates a compared point event. This event will generate PWM output pulse, interrupt signal wake-up or trigger EADC, PDMA, DAC to start conversion.

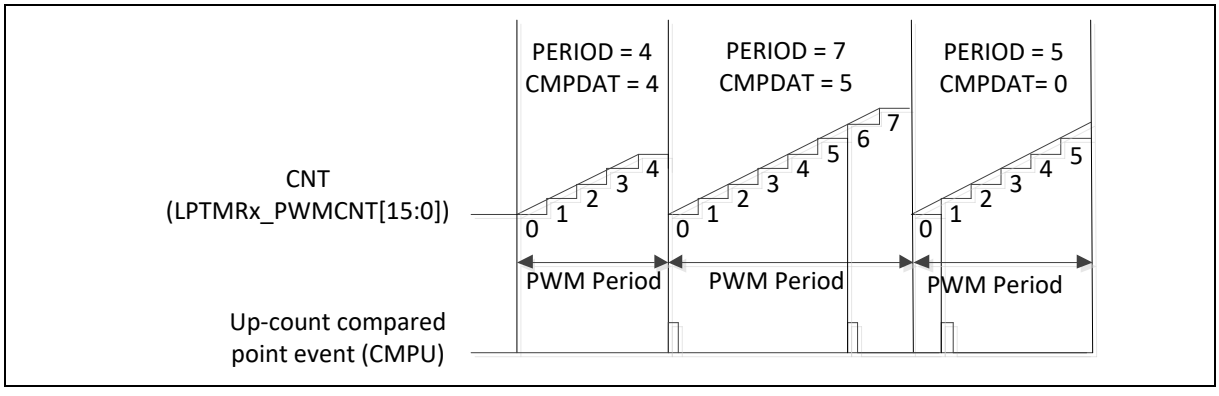


Figure 6.12-13 PWM Comparator Events in Up Count Type

6.12.6.6 Period Loading Mode

The PWM provides PBUF (LPTMRx_PWMPBUF[15:0]) is the active PERIOD buffer register and CMPBUF (LPTMRx_PWMCMPBUF[15:0]) is the active CMP buffer register. In period loading mode, both PERIOD (LPTMRx_PWMPERIOD[15:0]) and CMP (LPTMRx_PWMCMPDAT[15:0]) will load to their active PBUF and CMPBU register while each PWM period is completed.Figure 6.12-14. shows period loading timing of up count type, where PERIOD DATA0 denotes the initial data of PERIOD, PERIOD DATA1 denotes the first updated PERIOD data by user and so on, CMP also follows this rule. The following steps are the sequence of Figure 6.12-14.

1. User writes CMP DATA1 to CMP at point 1.
2. Period loading CMP DATA1 to CMPBUF at the end of PWM period at point 2.
3. User writes PERIOD DATA1 to PERIOD at point 3.
4. Period loading PERIOD DATA1 to PBUF at the end of PWM period at point 4.
5. User writes PERIOD DATA2 to PERIOD at point 5.
6. Period loading PERIOD DATA2 to PBUF at the end of PWM period at point 6.

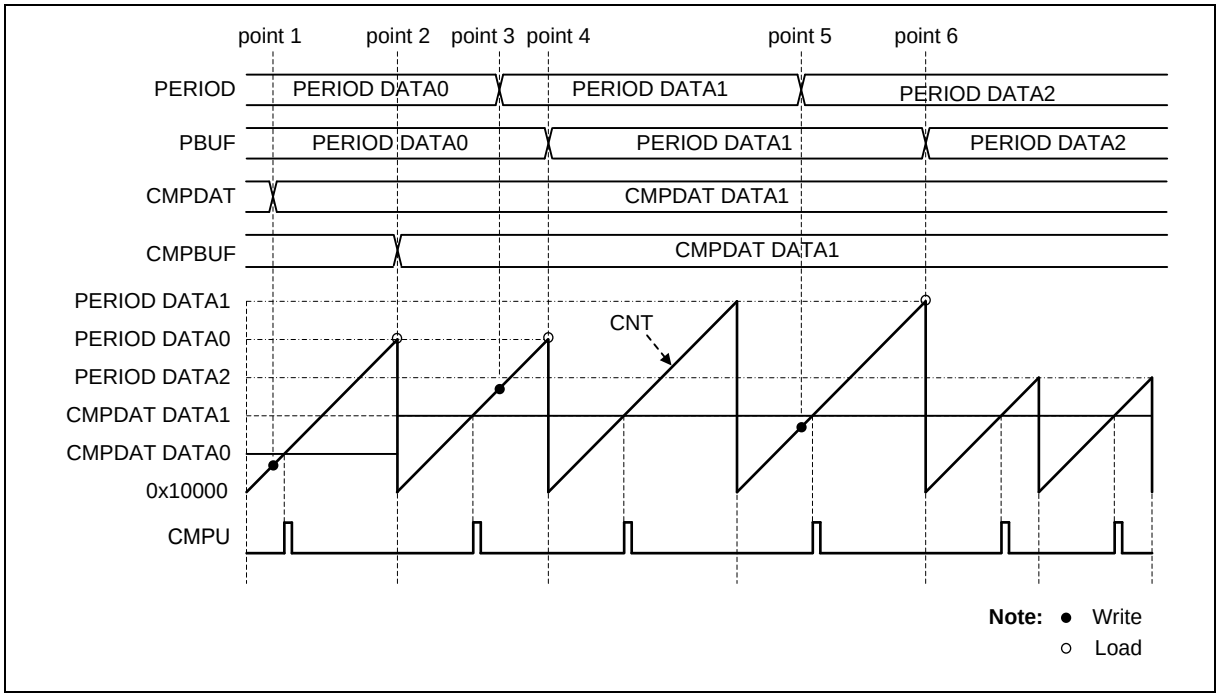


Figure 6.12-14 Period Loading Mode with Up Count Type

6.12.6.7 PWM Pulse Generator

PWM pulse generator uses counter and comparator events to generate PWM output pulse. The events are zero point in up count type and counter equal to comparator point in up count type.

The event point can generate PWM output waveform in up count type as shown in Figure 6.12-15.

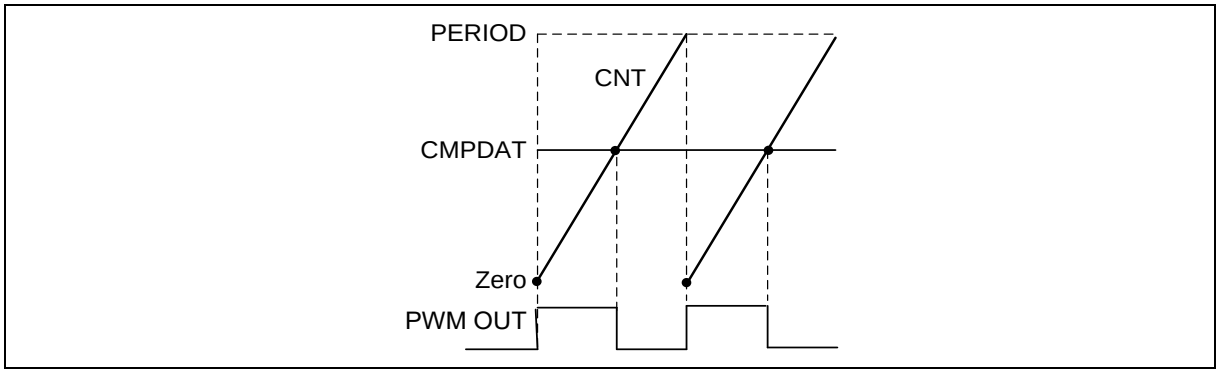


Figure 6.12-15 PWM Pulse Generation in Up Count Type

The PWM output pulse is associated with CMPBUF (LPTMRx_PWMCMPBUF[15:0]) and PBUF (LPTMRx_PWMPBUF[15:0]). The rules are listed as below.

1. If CMPBUF is zero, PWM output is 100% low.
2. If CMPBUF > PBUF, PWM output is 100% high.
3. If 0 < CMPBUF ≦ PBUF, PWM output high/low duty is according to counter value (LPTMRx_PWMCNT[15:0]).
    1) If PWM counter is higher than CMPBUF or equal to CMPBUF, PWM output is low.
    2) If PWM counter is lower than CMPBUF, PWM output is high.

The PWM output level table is shown as Table 6.12-1

Conditions	**CMPBUF = 0**	**CMPBUF > PBUF**	**0 < CMPBUF ≦ PBUF**	
			CNT ≧ CMPBUF	CNT < CMPBUF
PWM Output	Low	High	Low	High

Table 6.12-1 PWM Output Level

Figure 6.12-16. is an example about PWM duty cycle from 0% to 100% in up count type where PERIOD is 4 with different CMP value.

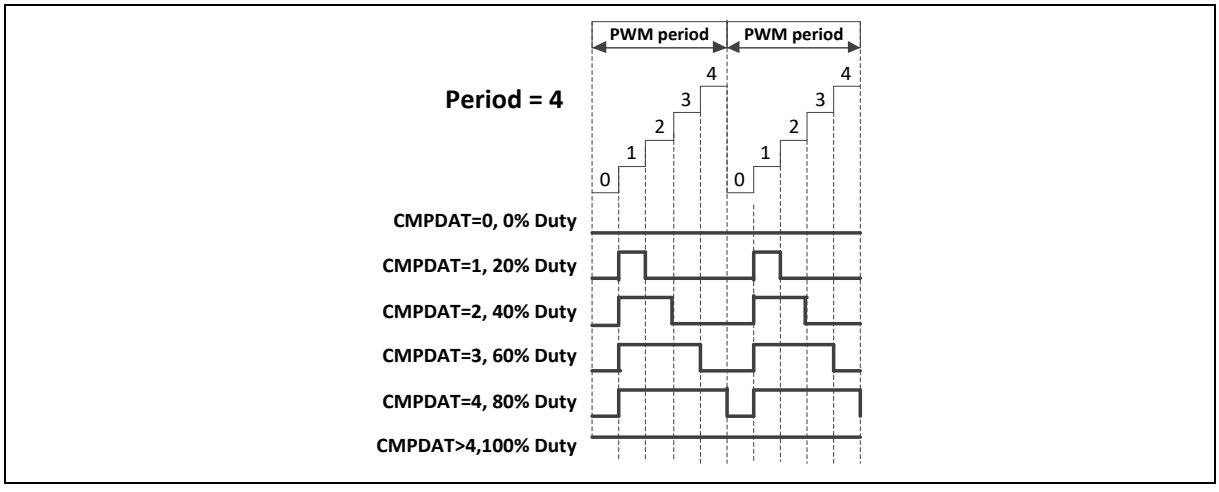


Figure 6.12-16 PWM 0% to 100% Duty Cycle in Up Count Type

6.12.6.8 PWM Output Control

After PWM pulse generator, there are three steps to control output waveform. User can set POEN (LPTMRx_PWMPOCTL[0]) 1 to enable PWMx_OUT output waveform and set POSEL (LPTMRx_PWMPOCTL[8]) to select LPTMRx or LPTMRx_EXT as PWM output channel.

There are polarity control, output enable control and output channel select to control output waveform as shown in Figure 6.12-17.

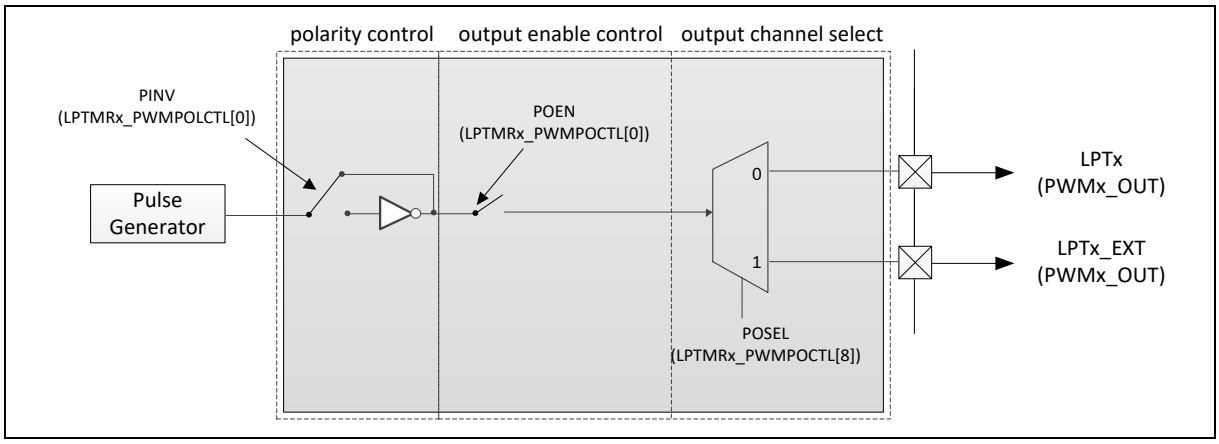


Figure 6.12-17 PWM Output PWMx_OUT Control

6.12.6.9 Polarity Control

Each PWMx_OUT (LPTMRx or LPTMRx_EXT) has an independent polarity control to configure the polarity of the active state of PWM output. User can control polarity state on PINV (LPTMRx_PWMPOLCTL[0]). Figure 6.12-18. shows the PWMx_OUT with polarity control.

When PWM output is selected as LPTMRx pin, PINV is 0, LPTMRx_EXT pin is kept as 0. Similarly, LPTMRx_EXT pin is kept as 1 when PINV is 1.

When PWM output is selected as LPTMRx_EXT pin, PINV is 0, LPTMRx pin is kept as 0. Similarly, LPTMRx pin is kept as 1 when PINV is 1.

When PWM output is selected as LPTMRx pin, PINV is 0, but POEN (LPTMRx_PWMPOCTL[0]) is set as 0, LPTMRx pin is kept as 0. Similarly, LPTMRx pin is kept as 1 when PINV is 1.

When PWM output is selected as LPTMRx_EXT pin, PINV is 0, but POEN is set as 0, LPTMRx_EXT pin is kept as 0. Similarly, LPTMRx_EXT pin is kept as 1 when PINV is 1.

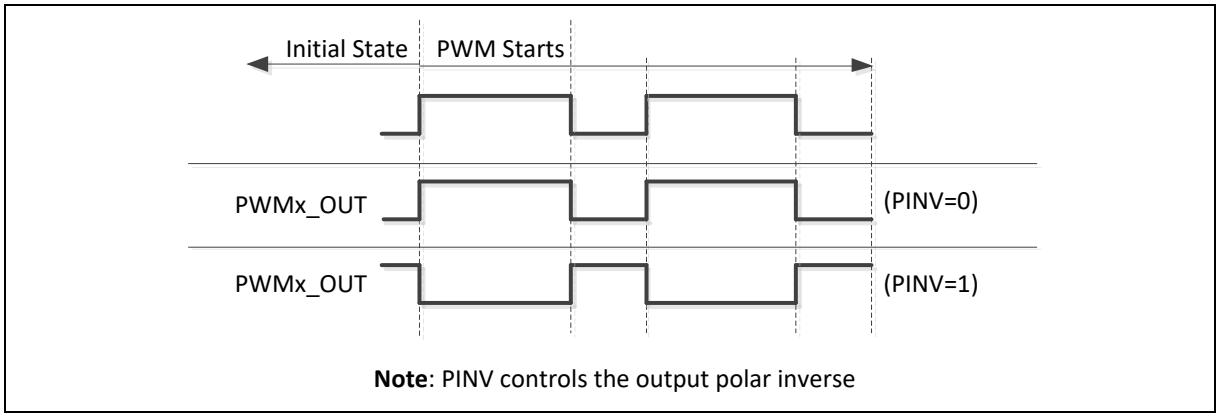


Figure 6.12-18 PWMx_OUT Polarity Control

6.12.6.10 PWM Interrupt Generator

There are independent interrupts for each PWM as shown in Figure 6.12-19.

The PWM interrupt (PWMx_INT) comes from PWM complementary pair events. The counter can generate the period point interrupt flag PIF (LPTMRx_PWMINTSTS0[1]). When counter equals to the comparator value stored in CMP (LPTMRx_PWMCMPDAT[15:0]) at up-count direction, the comparator up interrupt flag CMPUIF (LPTMRx_PWMINTSTS0[2]) is set. If the corresponding interrupt enable bits are set, the interrupt trigger events will also generates interrupt signals.

PWMx_INT can use the LPTMRx_PWMIFA register to accumulate the number of times that the interrupt flags have been triggered for each channel. By setting IFAEN (LPTMRx_PWMIFA[31]) bit to 1 to enable accumulator, PWMx_INT will switch interrupt source from every event trigger interrupt to trigger interrupt once every accumulate times.

By setting the IFASEL (LPTMRx_PWMIFA[29:28]) bits, user can select one of the 4 interrupt sources to accumulate interrupt flag times for each channel, and the number of times interrupt flags will compare with IFACNT (LPTMRx_PWMIFA[15:0]) bits. When interrupt accumulator equals IFACNT+1, the IFAIF (LPTMRx_PWMAINTSTS[0]) bit will be set as PWMx_INT signal if user enables IFAIEN (LPTMRx_PWMAINTEN[0]) bit. Accumulator interrupt of each channel can also be as request source of LPPDMA. Figure 6.12-19 is an example using LPTMRx_PWMIFA register to output PWMx_INT once every IFCNT0+1 times interrupt events occurred.

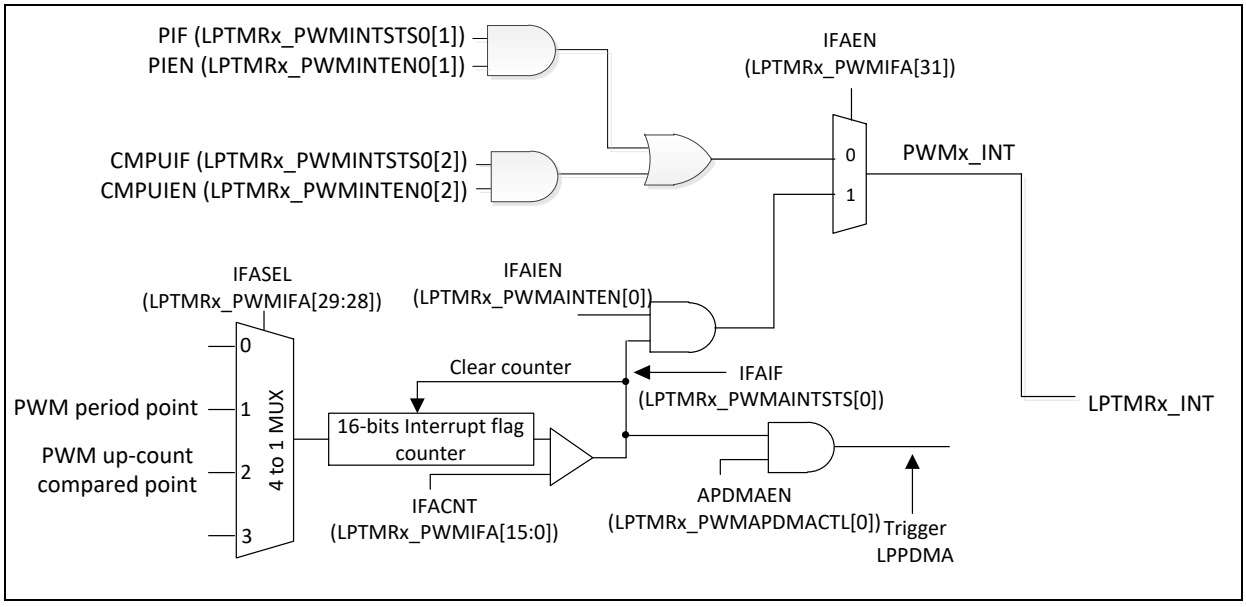


Figure 6.12-19 PWM Interrupt Architecture Diagram

6.12.6.11 PWM Wake-up Generator

User can set PWMINTWKEN (LPTMRx_PWMCTL[12]) high to wake up when PWM interrupt occurs. Before enter power-down mode, user must select LXT or LIRC as PWM clock source.

When PWM wake-up occurs, PWMINTWKF (LPTMRx_PWMSTATUS[8]) is set to 1. User can write 1 to this bit to clear this flag.

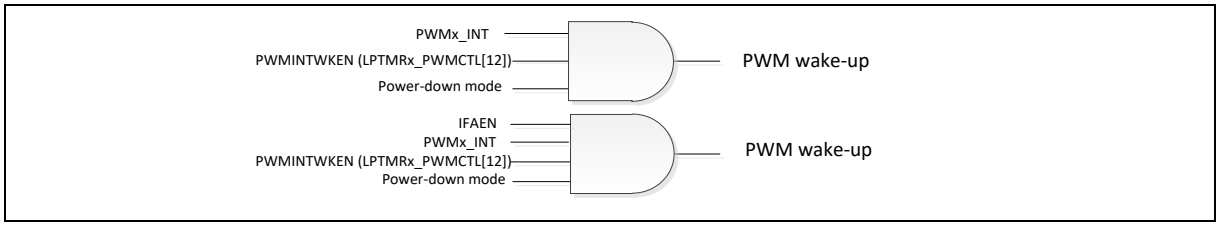


Figure 6.12-20 PWM Wake-up Architecture Diagram

6.12.6.12 PWM Trigger Low power IPs, LPPDMA Generator

PWM counter event can be one of the other Low Power IPs, LPPDMA conversion trigger source. User sets TRGSEL (LPTMRx_PWMTRGCTL[1:0]) to select which PWM counter event can trigger conversion.

When the TRGEN (LPTMRx_PWMTRGCTL[2]) is set to 1, the PWM can trigger Low power IPs. When the TRGEN(LPTMRxPWMTRGCTL[2]) and PWMTRGLPPDMA(LPTMRx_PWMTRGCTL[9]) are set to 1, the PWM can trigger LPPDMA.

There are two PWM counter events can be selected as the trigger source to start conversion as shown in Figure 6.12-21.

The PWM interrupt, wake-up and trigger comparison table is shown as Table 6.12-6.

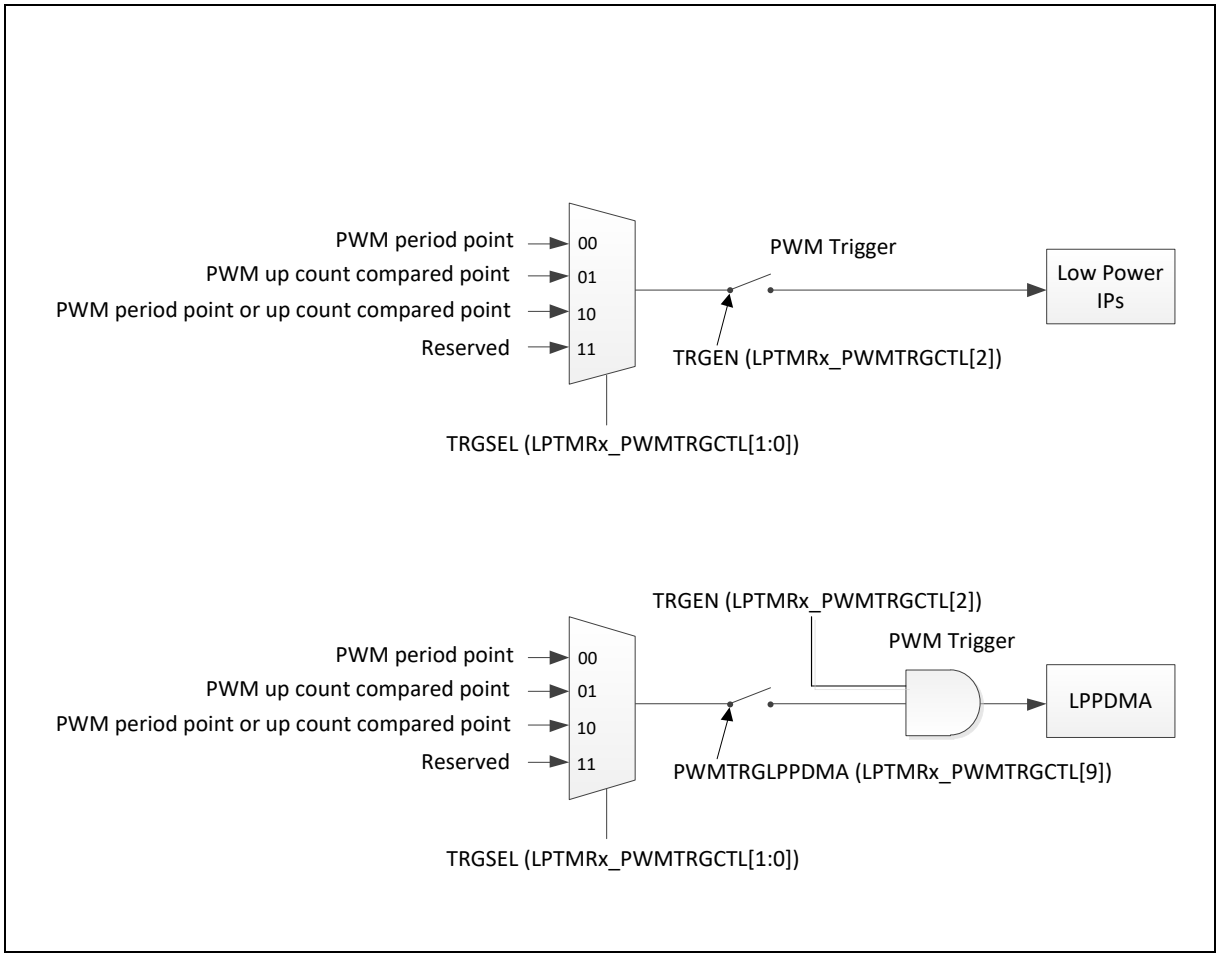


Figure 6.12-21 PWM Trigger Block Diagram

Functions / Events		Interrupt	Wake-Up PWMINTWKEN (LPTMRx_PWMCTL[12])		Trigger
			= 0	= 1	
CNT = CMP	CMPUIEN = 1	Yes	No	Yes	Low Power IPs, LPPDMA (Note1)
	CMPUIEN = 0	No	No	No	No
CNT = PERIOD	PIEN = 1	Yes	No	Yes	Low Power IPs, LPPDMA (Note1)
	PIEN = 0	No	No	No	No

Table 6.12-6 PWM Interrupt, Wake-up and Trigger Events Comparison

Note: Set TRGSEL (LPTMRx_PWMTRGCTL[1:0]) to select trigger event and set TRGEN (LPTMRx_PWMTRGCTL[2]), PWMTRGLPPDMA (LPTMRx_PWMTRGCTL[9]) to select which device will be triggered.

6.12.6.13 Mode Transition

If user needs to change timer mode to PWM mode or PWM mode to timer mode, user must do reset before change mode or check the previous mode is end.

The Low Power Timer mode to PWM mode transition steps are shown below.

1. Set Low Power Timer CNTEN (LPTMRx_CTL[30]) as 0
2. Polling ACTSTS (LPTMRx_CTL[25]). If ACTSTS is 0, user can set FUNCSEL (LPTMRx_CTL[15]) as 1 to enable PWM mode.
3. PWM function settings.

The PWM mode to Low Power Timer mode transition steps are shown below.

1. Set PWM CNTEN (LPTMRx_PWMCTL[0]) as 0.
2. Set FUNCSEL(LPTMRx_CTL[15]) as 0 to disable PWM mode.
3. Polling FUNCSEL(LPTMRx_CTL[15]). If FUNCSEL(LPTMRx_CTL[15]) is 0, user can set Low Power Timer settings.

6.12.6.14 Accumulator PDMA Function

The PWM module supports to trigger the LPPDMA transfer function when accumulator interrupt happened. When the corresponding LPPDMA enable bit APDMAEN (LPTMRx_PWMAPDMACTL[0]) is set, accumulator module will send a request to LPPDMA controller when accumulator interrupt has happened, meaning that IFAIF (LPTMRx_PWMAINTSTS[0]) is set 1. The LPPDMA controller will issue an acknowledge to accumulator and auto clear IFAIF (LPTMRx_PWMAINTSTS[0]) after it has read memory data and send the data to the particular register (PERIOD, etc.). So, user can use this function to change accumulator interrupt frequency.

6.12.6.15 Accumulator Stop Mode

The PWM module supports the Accumulator Stop Mode to stop counting when accumulator interrupt happened. Figure 6.12-22 shows Accumulator Stop Mode waveform. When the corresponding STPMOD (LPTMRx_PWMIFA[24]) is set, accumulator module will disable CNTEN (LPTMRx_PWMCTL[0]) after counter finishes whole period.

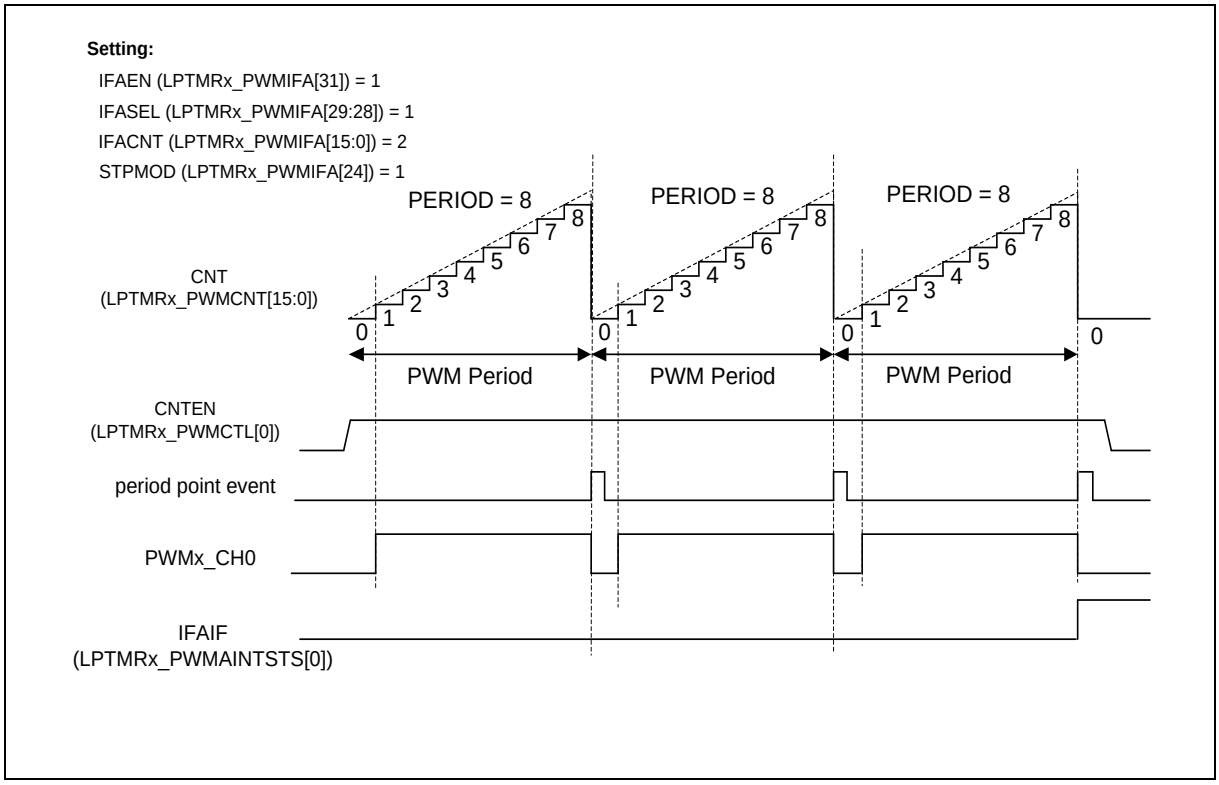


Figure 6.12-22 PWM Accumulator Stop Mode Waveform

6.12.6.16 Auto-operation Mode

The LPTMR controller supports auto-operation mode to make LPTMR can trigger other low power IPs when chip enters NPD0/1/2/3/4 mode. Configure TRGEN(LPTMRx_PWMTRGCTL[2]) and TRGSEL(LPTMR_PWMTRGCTL[1:0]) to enable auto-operation mode first. User can select TRGSEL(LPTMR_PWMTRGCTL[1:0]) to start trigger conversion at period point or compare up count point for auto-operation mode. LPTMR can broadcast a 4T pulse to other low power IPs when TRGSEL(LPTMR_PWMTRGCTL[1:0]) condition meets, then other low power IPs will receive the pulse. Especially for LPPDMA, LPTMR needs to be configured PWMTRGLPPDMA(LPTMRx_PWMTRGCTL[9]), TGREN(LPTMRx_PWMTRGCTL[2]), and TRGSEL(LPTMR_PWMTRGCTL[1:0]). LPTMR and LPPDMA have a handshake mechanism. LPTMR send a request to LPPDMA, and then LPPDMA will send an ack. This mechanism is different from other low power IPs. Other low power IPs don't need to send an ack to LPTMR. For LPPDMA, user can also use Accumulator PDMA Function in auto-operation mode.

Trigger Low Power IPs:

1. Enable LPTMR peripheral clock.

Set PDCLKEN(LPTMRx_CTL[16]) to open the engine clock in NPD0/1/2/3/4 mode. Engine clock should be always on to operate the PWM Function.

2. Configure Trigger Function

Set TRGEN(LPTMRx_PWMTRGCTL[2]) and TRGSEL(LPTMR_PWMTRGCT[1:0]) first. LPTMR has three conditions to trigger other low power IPs. They are period point, compared up point or both period point and compared up point.

Trigger LPPDMA:

1. Enable LPTMR peripheral clock.

Set PDCLKEN(LPTMRx_CTL[16]) to open the engine clock in NPD0/1/2/3/4 mode. Engine clock should

be always on to operation the PWM Function. When LPTMR trigger LPPDMA in NPD0/1/2/3/4 mode, AHB1, AHB1 bus matrix, LPPDMA and LPSRAM also need clock to work. But the AHB and APB clock are not always on in NPD0/1/2/3/4. AHB and APB clock is on when period point, compared up point or interrupt flag accumulator occurs. Then AHB and APB clock will turn off after LPPDMA transfer done.

2. Configure Trigger Function

Set PWMTRGLPPDMA(LPTMRx_PWMTRGCTL[9]), TRGEN(LPTMRx_PWMTRGCTL[2]) and TRGSEL(LPTMR_PWMTRGCT [1:0] first. LPTMR has three conditions to trigger LPPDMA. They are period point, compared up point or both period point and compared up point. User can also use Accumulator PDMA Function to trigger LPPDMA. User need to set APDMAEN(LPTMRx_PWMAPDMACTL) and LPTMRx_PWMIFA register before enter auto-operation mode.

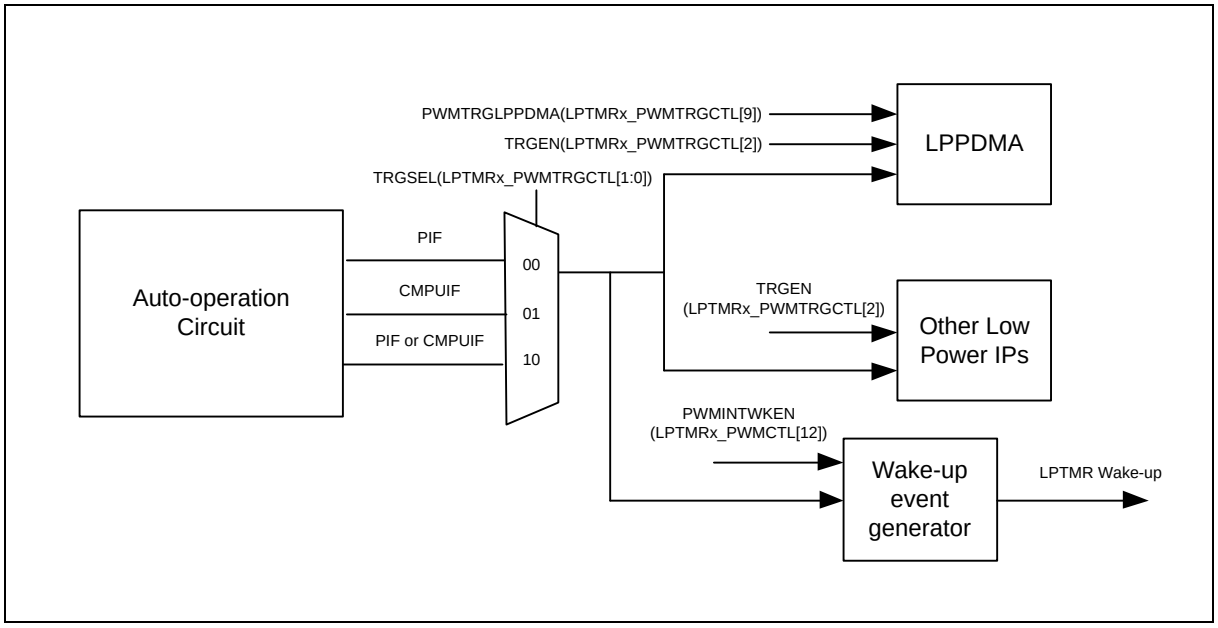


Figure 6.12-24 LPTMR Auto-operation mode Block Diagram

6.12.7 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
LPTMR Base Address: **LPTMR01_BA = 0x400E_4000**				
LPTMR0_CTL	LPTMR01_BA+0x00	R/W	LPTMR0 Control Register	0x0000_0005
LPTMR0_CMP	LPTMR01_BA+0x04	R/W	LPTMR0 Comparator Register	0x0000_0000
LPTMR0_INTSTS	LPTMR01_BA+0x08	R/W	LPTMR0 Interrupt Status Register	0x0000_0000
LPTMR0_CNT	LPTMR01_BA+0x0C	R/W	LPTMR0 Data Register	0x0000_0000
LPTMR0_CAP	LPTMR01_BA+0x10	R	LPTMR0 Capture Data Register	0x0000_0000
LPTMR0_EXTCTL	LPTMR01_BA+0x14	R/W	LPTMR0 External Control Register	0x0000_0000
LPTMR0_EINTSTS	LPTMR01_BA+0x18	R/W	LPTMR0 External Interrupt Status Register	0x0000_0000
LPTMR0_TRGCTL	LPTMR01_BA+0x1C	R/W	LPTMR0 Trigger Control Register	0x0000_0000
LPTMR0_CAPNF	LPTMR01_BA+0x24	R/W	LPTMR0 Capture Input Noise Filter Register	0x0000_0000
LPTMR0_PWMCTL	LPTMR01_BA+0x40	R/W	LPTMR0 PWM Control Register	0x0000_0000
LPTMR0_PWMCLKPSC	LPTMR01_BA+0x44	R/W	LPTMR0 PWM Counter Clock Pre-scale Register	0x0000_0000
LPTMR0_PWMCNTCLR	LPTMR01_BA+0x48	R/W	LPTMR0 PWM Clear Counter Register	0x0000_0000
LPTMR0_PWMPERIOD	LPTMR01_BA+0x4C	R/W	LPTMR0 PWM Period Register	0x0000_0000
LPTMR0_PWMCMPDAT	LPTMR01_BA+0x50	R/W	LPTMR0 PWM Comparator Register	0x0000_0000
LPTMR0_PWMCNT	LPTMR01_BA+0x54	R	LPTMR0 PWM Counter Register	0x0000_0000
LPTMR0_PWMPOLCTL	LPTMR01_BA+0x58	R/W	LPTMR0 PWM Pin Output Polar Control Register	0x0000_0000
LPTMR0_PWMPOCTL	LPTMR01_BA+0x5C	R/W	LPTMR0 PWM Pin Output Control Register	0x0000_0000
LPTMR0_PWMINTEN0	LPTMR01_BA+0x60	R/W	LPTMR0 PWM Interrupt Enable Register 0	0x0000_0000
LPTMR0_PWMINTSTS0	LPTMR01_BA+0x64	R/W	LPTMR0 PWM Interrupt Status Register 0	0x0000_0000
LPTMR0_PWMTRGCTL	LPTMR01_BA+0x68	R/W	LPTMR0 PWM Trigger Control Register	0x0000_0000
LPTMR0_PWMSTATUS	LPTMR01_BA+0x6C	R/W	LPTMR0 PWM Status Register	0x0000_0000

LPTMR0_PWMPBUF	LPTMR01_BA+0x70	R	LPTMR0 PWM Period Buffer Register	0x0000_0000
LPTMR0_PWMCMPBUF	LPTMR01_BA+0x74	R	LPTMR0 PWM Comparator Buffer Register	0x0000_0000
LPTMR0_PWMIFA	LPTMR01_BA+0xA8	R/W	LPTMR0 PWM Interrupt Flag Accumulator Register	0x0000_0000
LPTMR0_PWMAINTSTS	LPTMR01_BA+0xAC	R/W	LPTMR0 PWM Accumulator Interrupt Flag Register	0x0000_0000
LPTMR0_PWMAINTEN	LPTMR01_BA+0xB0	R/W	LPTMR0 PWM Accumulator Interrupt Enable Register	0x0000_0000
LPTMR0_PWMAPDMACTL	LPTMR01_BA+0xB4	R/W	LPTMR0 PWM Accumulator LPPDMA Control Register	0x0000_0000
LPTMR1_CTL	LPTMR01_BA+0x100	R/W	LPTMR1 Control Register	0x0000_0005
LPTMR1_CMP	LPTMR01_BA+0x104	R/W	LPTMR1 Comparator Register	0x0000_0000
LPTMR1_INTSTS	LPTMR01_BA+0x108	R/W	LPTMR1 Interrupt Status Register	0x0000_0000
LPTMR1_CNT	LPTMR01_BA+0x10C	R/W	LPTMR1 Data Register	0x0000_0000
LPTMR1_CAP	LPTMR01_BA+0x110	R	LPTMR1 Capture Data Register	0x0000_0000
LPTMR1_EXTCTL	LPTMR01_BA+0x114	R/W	LPTMR1 External Control Register	0x0000_0000
LPTMR1_EINTSTS	LPTMR01_BA+0x118	R/W	LPTMR1 External Interrupt Status Register	0x0000_0000
LPTMR1_TRGCTL	LPTMR01_BA+0x11C	R/W	LPTMR1 Trigger Control Register	0x0000_0000
LPTMR1_CAPNF	LPTMR01_BA+0x124	R/W	LPTMR1 Capture Input Noise Filter Register	0x0000_0000
LPTMR1_PWMCTL	LPTMR01_BA+0x140	R/W	LPTMR1 PWM Control Register	0x0000_0000
LPTMR1_PWMCLKPSC	LPTMR01_BA+0x144	R/W	LPTMR1 PWM Counter Clock Pre-scale Register	0x0000_0000
LPTMR1_PWMCNTCLR	LPTMR01_BA+0x148	R/W	LPTMR1 PWM Clear Counter Register	0x0000_0000
LPTMR1_PWMPERIOD	LPTMR01_BA+0x14C	R/W	LPTMR1 PWM Period Register	0x0000_0000
LPTMR1_PWMCMPDAT	LPTMR01_BA+0x150	R/W	LPTMR1 PWM Comparator Register	0x0000_0000
LPTMR1_PWMCNT	LPTMR01_BA+0x154	R	LPTMR1 PWM Counter Register	0x0000_0000
LPTMR1_PWMPOLCTL	LPTMR01_BA+0x158	R/W	LPTMR1 PWM Pin Output Polar Control Register	0x0000_0000
LPTMR1_PWMPOCTL	LPTMR01_BA+0x15C	R/W	LPTMR1 PWM Pin Output Control Register	0x0000_0000
LPTMR1_PWMINTEN0	LPTMR01_BA+0x160	R/W	LPTMR1 PWM Interrupt Enable Register 0	0x0000_0000

LPTMR1_PWMINTSTS0	LPTMR01_BA+0x164	R/W	LPTMR1 PWM Interrupt Status Register 0	0x0000_0000
LPTMR1_PWMTRGCTL	LPTMR01_BA+0x168	R/W	LPTMR1 PWM Trigger Control Register	0x0000_0000
LPTMR1_PWMSTATUS	LPTMR01_BA+0x16C	R/W	LPTMR1 PWM Status Register	0x0000_0000
LPTMR1_PWMPBUF	LPTMR01_BA+0x170	R	LPTMR1 PWM Period Buffer Register	0x0000_0000
LPTMR1_PWMCMPBUF	LPTMR01_BA+0x174	R	LPTMR1 PWM Comparator Buffer Register	0x0000_0000
LPTMR1_PWMIFA	LPTMR01_BA+0x1A8	R/W	LPTMR1 PWM Interrupt Flag Accumulator Register	0x0000_0000
LPTMR1_PWMAINTSTS	LPTMR01_BA+0x1AC	R/W	LPTMR1 PWM Accumulator Interrupt Flag Register	0x0000_0000
LPTMR1_PWMAINTEN	LPTMR01_BA+0x1B0	R/W	LPTMR1 PWM Accumulator Interrupt Enable Register	0x0000_0000
LPTMR1_PWMAPDMACTL	LPTMR01_BA+0x1B4	R/W	LPTMR1 PWM Accumulator LPPDMA Control Register	0x0000_0000

6.12.8 Register Description

Low Power Timer Control Register (LPTMRx_CTL)

Register	Offset	R/W	Description	Reset Value
LPTMR0_CTL	LPTMR01_BA+0x00	R/W	LPTMR0 Control Register	0x0000_0005
LPTMR1_CTL	LPTMR01_BA+0x100	R/W	LPTMR1 Control Register	0x0000_0005

31	30	29	28	27	26	25	24
ICEDEBUG	CNTEN	INTEN	OPMODE		Reserved	ACTSTS	EXTCNTEN
23	22	21	20	19	18	17	16
WKEN	CAPSRC	TGLPINSEL	PERIOSEL	Reserved			PDCLKEN
15	14	13	12	11	10	9	8
FUNCSEL	Reserved						
7	6	5	4	3	2	1	0
PSC							

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects TIMER counting. TIMER counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. TIMER counter will keep going no matter CPU is held by ICE or not. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[30]	**CNTEN**	**Low Power Timer Counting Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. **Note 1:** In stop status, and then set CNTEN to 1 will enable the 24-bit up counter to keep counting from the last stop counting value. **Note 2:** This bit is auto-cleared by hardware in one-shot mode (LPTMR_CTL[28:27] = 00) when the low power timer time-out interrupt flag TIF (LPTMRx_INTSTS[0]) is generated. **Note 3:** Set enable/disable this bit needs 2 * LPTMR_CLK period to become active, user can read ACTSTS (LPTMRx_CTL[25]) to check enable/disable command is completed or not.
[29]	**INTEN**	**Low Power Timer Interrupt Enable Bit** 0 = Low Power Timer time-out interrupt Disabled. 1 = Low Power Timer time-out interrupt Enabled. **Note:** If this bit is enabled, when the low power timer time-out interrupt flag TIF is set to 1, the low power timer interrupt signal is generated and inform to CPU.
[28:27]	**OPMODE**	**Low Power Timer Counting Mode Select** 00 = The low power timer controller is operated in One-shot mode. 01 = The low power timer controller is operated in Periodic mode. 10 = The low power timer controller is operated in Toggle-output mode. 11 = The low power timer controller is operated in Continuous Counting mode.

[26]	**Reserved**	Reserved.
[25]	**ACTSTS**	**Low Power Timer Active Status Bit (Read Only)** This bit indicates the 24-bit up counter status. 0 = 24-bit up counter is not active. 1 = 24-bit up counter is active. **Note:** This bit may active when CNT 0 transition to CNT 1.
[24]	**EXTCNTEN**	**Event Counter Mode Enable Bit** This bit is for external counting pin function enabled. 0 = Event counter mode Disabled. 1 = Event counter mode Enabled. **Note:** When timer is used as an event counter, this bit should be set to 1.
[23]	**WKEN**	**Wake-up Function Enable Bit** If this bit is set to 1, while timer interrupt flag TIF (LPTMRx_INTSTS[0]) is 1 and INTEN (LPTMRx_CTL[29]) is enabled, the low power timer interrupt signal will generate a wake-up trigger event to CPU. 0 = Wake-up function Disabled if timer interrupt signal generated. 1 = Wake-up function Enabled if timer interrupt signal generated.
[22]	**CAPSRC**	**Capture Pin Source Selection** 0 = Capture Function source is from LPTMRx_EXT (x= 0~1) pin. 1 = Capture Function source is from internal ACMP output signal **Note:** When CAPSRC = 1,User can set INTERCAPSEL (LPTMRx_EXTCTL[10:8]) to decide which internal ACMP output signal or which clock is as timer capture source.
[21]	**TGLPINSEL**	**Toggle-output Pin Select** 0 = Toggle mode output to LPTMRx (Low Power Timer Event Counter Pin). 1 = Toggle mode output to LPTMRx_EXT (Low Power Timer External Capture Pin).
[20]	**PERIOSEL**	**Periodic Mode Behavior Selection Enable Bit** 0 = The behavior selection in periodic mode is Disabled. When user updates CMPDAT while timer is running in periodic mode, CNT will be reset to default value. 1 = The behavior selection in periodic mode is Enabled. When user updates CMPDAT while timer is running in periodic mode, the limitations as bellows list, If updated CMPDAT value > CNT, CMPDAT will be updated and CNT keep running continually. If updated CMPDAT value = CNT, timer time-out interrupt will be asserted immediately. If updated CMPDAT value < CNT, CNT will be reset to default value.
[19:17]	**Reserved**	Reserved.
[16]	**PDCLKEN**	**Power-down Engine Clock Enable** 0 = Disable engine clock in power-down mode. 1 = Enable engine clock in power-down mode.
[15]	**FUNCSEL**	**Function Selection** 0 = Low Power Timer controller is used as timer function. 1 = Low Power Timer controller is used as PWM function.
[14:8]	**Reserved**	Reserved.
[7:0]	**PSC**	**Prescale Counter** Low Power Timer input clock or event source is divided by (PSC+1) before it is fed to the low power timer up counter. If this field is 0 (PSC = 0), then there is no scaling. **Note:** Update prescale counter value will reset internal 8-bit prescale counter and 24-bit up counter value.

Low Power Timer Comparator Register (LPTMRx_CMP)

Register	Offset	R/W	Description	Reset Value
LPTMR0_CMP	LPTMR01_BA+0x04	R/W	LPTMR0 Comparator Register	0x0000_0000
LPTMR1_CMP	LPTMR01_BA+0x104	R/W	LPTMR1 Comparator Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPDAT							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CMPDAT**	**Low Power Timer Comparator Value** CMPDAT is a 24-bit compared value register. When the internal 24-bit up counter value is equal to CMPDAT value, the TIF (LPTMRx_INTSTS[0] Low Power Timer Interrupt Flag) will set to 1. Time-out period = (Period of timer clock input) * (8-bit PSC + 1) * (24-bit CMPDAT). **Note 1:** Never write 0x0 or 0x1 in CMPDAT field, or the core will run into unknown state. **Note 2:** When timer is operating at continuous counting mode, the 24-bit up counter will keep counting continuously even if user writes a new value into CMPDAT field. But if timer is operating at other modes, the 24-bit up counter will restart counting from 0 and using newest CMPDAT value to be the low power timer compared value while user writes a new value into CMPDAT field.

Low Power Timer Interrupt Status Register (LPTMRx_INTSTS)

Register	Offset	R/W	Description	Reset Value
LPTMR0_INTSTS	LPTMR01_BA+0x08	R/W	LPTMR0 Interrupt Status Register	0x0000_0000
LPTMR1_INTSTS	LPTMR01_BA+0x108	R/W	LPTMR1 Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TWKF	TIF

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TWKF**	**Low Power Timer Wake-up Flag** This bit indicates the interrupt wake-up flag status of timer. 0 = Low Power Timer does not cause CPU wake-up. 1 = CPU wake-up from Idle or Power-down mode if timer time-out interrupt signal generated. **Note:** This bit is cleared by writing 1 to it.
[0]	**TIF**	**Low Power Timer Interrupt Flag** This bit indicates the interrupt flag status of Low Power Timer while 24-bit timer up counter CNT (LPTMRx_CNT[23:0]) value reaches to CMPDAT (LPTMRx_CMP[23:0]) value. 0 = No effect. 1 = CNT value matches the CMPDAT value. **Note:** This bit is cleared by writing 1 to it.

Low Power Timer Data Register (LPTMRx_CNT)

Register	Offset	R/W	Description	Reset Value
LPTMR0_CNT	LPTMR01_BA+0x0C	R/W	LPTMR0 Data Register	0x0000_0000
LPTMR1_CNT	LPTMR01_BA+0x10C	R/W	LPTMR1 Data Register	0x0000_0000

31	30	29	28	27	26	25	24
RSTACT	Reserved						
23	22	21	20	19	18	17	16
CNT							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

Bits	Description	
[31]	**RSTACT**	**Low Power Timer Data Register Reset Active (Read Only)** This bit indicates if the counter reset operation active. When user writes this CNT register, low power timer starts to reset its internal 24-bit timer up-counter to 0 and reload 8-bit pre-scale counter. At the same time, timer set this flag to 1 to indicate the counter reset operation is in progress. Once the counter reset operation done, low power timer clear this bit to 0 automatically. 0 = Reset operation is done. 1 = Reset operation triggered by writing LPTMRx_CNT is in progress.
[30:24]	**Reserved**	Reserved.
[23:0]	**CNT**	**Low Power Timer Data Register** Read operation. Read this register to get CNT value. For example: If EXTCNTEN (LPTMRx_CTL[24]) is 0, user can read CNT value for getting current 24-bit counter value. If EXTCNTEN (LPTMRx_CTL[24]) is 1, user can read CNT value for getting current 24-bit event input counter value. Write operation. Writing any value to this register will reset current CNT value to 0 and reload internal 8-bit prescale counter.

Low Power Timer Capture Data Register (LPTMRx_CAP)

Register	Offset	R/W	Description	Reset Value
LPTMR0_CAP	LPTMR01_BA+0x10	R	LPTMR0 Capture Data Register	0x0000_0000
LPTMR1_CAP	LPTMR01_BA+0x110	R	LPTMR1 Capture Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CAPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CAPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CAPDAT							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CAPDAT**	**Low Power Timer Capture Data Register** When CAPEN (LPTMRx_EXTCTL[3]) bit is set, CAPFUNCS (LPTMRx_EXTCTL[4]) bit is 0, and a transition on LPTMRx_EXT pin matched the CAPEDGE (LPTMRx_EXTCTL[14:12]) setting, CAPIF (LPTMRx_EINTSTS[0]) will set to 1 and the current timer counter value CNT (LPTMRx_CNT[23:0]) will be auto-loaded into this CAPDAT field.

Low Power Timer External Control Register (LPTMRx_EXTCTL)

Register	Offset	R/W	Description	Reset Value
LPTMR0_EXTCTL	LPTMR01_BA+0x14	R/W	LPTMR0 External Control Register	0x0000_0000
LPTMR1_EXTCTL	LPTMR01_BA+0x114	R/W	LPTMR1 External Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CAPDIVSCL				Reserved			
23	22	21	20	19	18	17	16
Reserved					ECNTSSEL		
15	14	13	12	11	10	9	8
Reserved	CAPEDGE			Reserved	INTERCAPSEL		
7	6	5	4	3	2	1	0
CNTDBEN	CAPDBEN	CAPIEN	CAPFUNCS	CAPEN	Reserved		CNTPHASE

Bits	Description	
[31:28]	**CAPDIVSCL**	**Low Power Timer Capture Source Divider Scale** **This bits indicate the divide scale for capture source divider** 0000 = Capture source/1. 0001 = Capture source/2. 0010 = Capture source/4. 0011 = Capture source/8. 0100 = Capture source/16. 0101 = Capture source/32. 0110 = Capture source/64. 0111 = Capture source/128. 1000 = Capture source/256. 1001~1111 = Reserved. **Note:** Set INTERCAPSEL (LPTMRx_EXTCTL[10:8]) and CAPSRC (LPTMRx_CTL[22]) to select capture source.
[27:19]	**Reserved**	Reserved.
[18:16]	**ECNTSSEL**	**Event Counter Source Selection to Trigger Event Counter Function** 000 = Event Counter input source is from external LPTMRx (x= 0~1) pin. 010 = Event Counter input source is from internal ACMP0 output signal. 011 = Event Counter input source is from internal ACMP1 output signal. 100 = Event Counter input source is from internal ACMP2 output signal. 101~111 = Reserved.
[15]	**Reserved**	Reserved.
[14:12]	**CAPEDGE**	**Low Power Timer External Capture Pin Edge Detect** When first capture event is generated, the CNT (LPTMRx_CNT[23:0]) will be reset to 0 and first CAPDAT (LPTMRx_CAP[23:0]) should be to 0.

		000 = Capture event occurred when detect falling edge transfer on LPTMRx_EXT (x= 0~1) pin. 001 = Capture event occurred when detect rising edge transfer on LPTMRx_EXT (x= 0~1) pin. 010 = Capture event occurred when detect both falling and rising edge transfer on LPTMRx_EXT (x= 0~1) pin, and first capture event occurred at falling edge transfer. 011 = Capture event occurred when detect both rising and falling edge transfer on LPTMRx_EXT (x= 0~1) pin, and first capture event occurred at rising edge transfer. 110 = First capture event occurred at falling edge, follows capture events are at rising edge transfer on LPTMRx_EXT (x= 0~1) pin. 111 = First capture event occurred at rising edge, follows capture events are at falling edge transfer on LPTMRx_EXT (x= 0~1) pin. 100, 101 = Reserved. **Note:** Set CAPSRC (LPTMRx_CTL[22]) and INTERCAPSEL (LPTMRx_EXTCTL[10:8]) to select capture source.
[11]	**Reserved**	Reserved.
[10:8]	**INTERCAPSEL**	**Internal Capture Source Select** 000 = Capture Function source is from internal ACMP0 output signal. 001 = Capture Function source is from internal ACMP1 output signal. 111 = Capture Function source is from internal ACMP2 output signal.. **Note:** these bits only available when CAPSRC (LPTMRx_CTL[22]) is 1.
[7]	**CNTDBEN**	**Low Power Timer Counter Pin De-bounce Enable Bit** 0 = LPTMRx (x= 0~1) pin de-bounce Disabled. 1 = LPTMRx (x= 0~1) pin de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of LPTMRx pin is detected with de-bounce circuit.
[6]	**CAPDBEN**	**Low Power Timer External Capture Pin De-bounce Enable Bit** 0 = LPTMRx_EXT (x= 0~1) pin de-bounce or ACMP output de-bounce Disabled. 1 = LPTMRx_EXT (x= 0~1) pin de-bounce or ACMP output de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of LPTMRx_EXT pin or ACMP output is detected with de-bounce circuit.
[5]	**CAPIEN**	**Low Power Timer External Capture Interrupt Enable Bit** 0 = LPTMRx_EXT (x= 0~1) pin, ACMP, internal clock, or external clock detection Interrupt Disabled. 1 = LPTMRx_EXT (x= 0~1) pin, ACMP, internal clock, or external clock detection Interrupt Enabled. **Note:** CAPIEN is used to enable timer external interrupt. If CAPIEN enabled, timer will rise an interrupt when CAPIF (LPTMRx_EINTSTS[0]) is 1. For example, while CAPIEN = 1, CAPEN = 1, and CAPEDGE = 00, a 1 to 0 transition on the LPTMRx_EXT (x= 0~1) pin, ACMP, internal clock, or external clock will cause the CAPIF to be set then the interrupt signal is generated and sent to NVIC to inform CPU.
[4]	**CAPFUNCS**	**Capture Function Selection** 0 = External Capture Mode Enabled. 1 = External Reset Mode Enabled. **Note 1:** When CAPFUNCS is 0, transition on LPTMRx_EXT (x= 0~1) pin is using to save current 24-bit timer counter value (CNT value) to CAPDAT field. **Note 2:** When CAPFUNCS is 1, transition on LPTMRx_EXT (x= 0~1) pin is using to save current 24-bit timer counter value (CNT value) to CAPDAT field then CNT value will be reset immediately.
[3]	**CAPEN**	**Low Power Timer Capture Enable Bit** This bit enables the capture input function. 0 =Capture source Disabled. 1 =Capture source Enabled. **Note:** When CAPEN is 1, user can set INTERCAPSEL (LPTMRx_EXTCTL [10:8]) to select capture source.
[2:1]	**Reserved**	Reserved.

[0]	**CNTPHASE**	**Low Power Timer External Count Phase** This bit indicates the detection phase of external counting pin LPTMRx (x= 0~1). 0 = A falling edge of external counting pin will be counted. 1 = A rising edge of external counting pin will be counted.

Low Power Timer External Interrupt Status Register (LPTMRx_EINTSTS)

Register	Offset	R/W	Description	Reset Value
LPTMR0_EINTSTS	LPTMR01_BA+0x18	R/W	LPTMR0 External Interrupt Status Register	0x0000_0000
LPTMR1_EINTSTS	LPTMR01_BA+0x118	R/W	LPTMR1 External Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CAPIFOV	CAPIF

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CAPIFOV**	**Capture Latch Interrupt Flag Overrun Status (Read Only)** 0 = Capture latch happened when the corresponding CAPIF is 0. 1 = Capture latch happened when the corresponding CAPIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CAPIF.
[0]	**CAPIF**	**Low Power Timer External Capture Interrupt Flag** This bit indicates the low power timer external capture interrupt flag status. 0 = LPTMRx_EXT (x= 0~1) pin, ACMP, internal clock, or external clock interrupt did not occur. 1 = LPTMRx_EXT (x= 0~1) pin, ACMP, internal clock, or external clock interrupt occurred. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** When CAPEN (LPTMRx_EXTCTL[3]) bit is set, CAPFUNCS (LPTMRx_EXTCTL[4]) bit is 0, and a transition on LPTMRx_EXT (x= 0~1) pin, ACMP, internal clock, or external clock matched the CAPEDGE (LPTMRx_EXTCTL[2:1]) setting, this bit will set to 1 by hardware. **Note 3:** There is a new incoming capture event detected before CPU clearing the CAPIF status. If the above condition occurred, the Low Power Timer will keep register LPTMRx_CAP unchanged and drop the new capture value.

Low Power Timer Trigger Control Register (LPTMRx_TRGCTL)

Register	Offset	R/W	Description	Reset Value
LPTMR0_TRGCTL	LPTMR01_BA+0x1C	R/W	LPTMR0 Trigger Control Register	0x0000_0000
LPTMR1_TRGCTL	LPTMR01_BA+0x11C	R/W	LPTMR1 Trigger Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			TRGLPPDMA	Reserved		TRGEN	TRGSSEL

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**TRGLPPDMA**	**Trigger LPPDMA Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be triggered LPPDMA transfer. 0 = Low Power Timer interrupt trigger LPPDMA Disabled. 1 = Low Power Timer interrupt trigger LPPDMA Enabled. **Note:** If TRGSSEL (LPTMRx_TRGCTL[0]) = 0, time-out interrupt signal will trigger LPPDMA transfer. **Note:** If TRGSSEL (LPTMRx_TRGCTL[0]) = 1, capture interrupt signal will trigger LPPDMA transfer.
[3:2]	**Reserved**	Reserved.
[1]	**TRGEN**	**Trigger Low power IPs Enable Bit** If this bit is set to 1, each timer time-out event or capture event can trigger Low Power IPs. 0 = Low Power Timer interrupt trigger Low Power IPs Disabled. 1 = Low Power Timer interrupt trigger Low Power IPs Enabled. **Note:** If TRGSSEL (LPTMRx_TRGCTL[0]) = 0, time-out interrupt signal will trigger Low Power IPs conversion. **Note:** If TRGSSEL (LPTMRx_TRGCTL[0]) = 1, capture interrupt signal will trigger Low Power IPs conversion.
[0]	**TRGSSEL**	**Trigger Source Select Bit** This bit is used to select internal trigger source is from timer time-out interrupt signal or capture interrupt signal. 0 = Time-out interrupt signal is used to internal trigger LPADC0, , LPI2C0, LPSPI0, LPUART, , and LPPDMA. 1 = Capture interrupt signal is used to internal trigger LPADC0, LPI2C0, LPSPI0, LPUART, , and LPPDMA.

Low Power Timer Capture Input Noise Filter Register (LPTMRx_CAPNF)

Register	Offset	R/W	Description	Reset Value
LPTMR0_CAPNF	LPTMR01_BA+0x24	R/W	LPTMR0 Capture Input Noise Filter Register	0x0000_0000
LPTMR1_CAPNF	LPTMR01_BA+0x124	R/W	LPTMR1 Capture Input Noise Filter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					CAPNFCNT		
7	6	5	4	3	2	1	0
Reserved	CAPNFSEL			Reserved			CAPNFEN

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**CAPNFCNT**	**Capture Edge Detector Noise Filter Count** These bits control the capture filter counter to count from 0 to CAPNFCNT.
[7]	**Reserved**	Reserved.
[6:4]	**CAPNFSEL**	**Capture Edge Detector Noise Filter Clock Selection** 000 = Noise filter clock is ECLKx. 001 = Noise filter clock is ECLKx/2. 010 = Noise filter clock is ECLKx/4. 011 = Noise filter clock is ECLKx/8. 100 = Noise filter clock is ECLKx/16. 101 = Noise filter clock is ECLKx/32. 110 = Noise filter clock is ECLKx/64. 111 = Noise filter clock is ECLKx/128.
[3:1]	**Reserved**	Reserved.
[0]	**CAPNFEN**	**Capture Noise Filter Enable** 0 = Capture Noise Filter function Disabled. 1 = Capture Noise Filter function Enabled.

Low Power Timer PWM Control Register (LPTMRx_PWMCTL)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMCTL	LPTMR01_BA+0x40	R/W	LPTMR0 PWM Control Register	0x0000_0000
LPTMR1_PWMCTL	LPTMR01_BA+0x140	R/W	LPTMR1 PWM Control Register	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	DBGHALT	Reserved					
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			PWMINTWKEN	Reserved			
7	6	5	4	3	2	1	0
Reserved				CNTMODE	Reserved		CNTEN

Bits	Description	
[31]	**DBGTRIOFF**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects PWM output. PWM output pin will be forced as tri-state while ICE debug mode acknowledged. 1 = ICE debug mode acknowledgement disabled. PWM output pin will keep output no matter ICE debug mode acknowledged or not. **Note:** This bit is write protected. Refer to SYS_REGLCTL control register.
[30]	**DBGHALT**	**ICE Debug Mode Counter Halt (Write Protect)** If debug mode counter halt is enabled, PWM counter will keep current value until exit ICE debug mode. 0 = ICE debug mode counter halt Disabled. 1 = ICE debug mode counter halt Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL control register.
[29:13]	**Reserved**	Reserved.
[12]	**PWMINTWKEN**	**PWM Interrupt Wake-up Enable Bit** If PWM interrupt occurs when chip is in Power-down mode, PWMINTWKEN can determine whether chip wake-up occurs or not. 0 = PWM interrupt wake-up Disabled. 1 = PWM interrupt wake-up Enabled.
[11:4]	**Reserved**	Reserved.
[3]	**CNTMODE**	**PWM Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.
[2:1]	**Reserved**	Reserved.
[0]	**CNTEN**	**PWM Counter Enable Bit** 0 = PWM counter and clock prescale Stop Running.

		1 = PWM counter and clock prescale Start Running.

Low Power Timer PWM Counter Clock Pre-scale Register (LPTMRx_PWMCLKPSC)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMCLKPSC	LPTMR01_BA+0x44	R/W	LPTMR0 PWM Counter Clock Pre-scale Register	0x0000_0000
LPTMR1_PWMCLKPSC	LPTMR01_BA+0x144	R/W	LPTMR1 PWM Counter Clock Pre-scale Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**CLKPSC**	**PWM Counter Clock Pre-scale** The active clock of PWM counter is decided by counter clock prescale and divided by (CLKPSC + 1). If CLKPSC is 0, then there is no scaling in PWM counter clock source.

Low Power Timer PWM Clear Counter Register (LPTMRx_PWMCNTCLR)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMCNTCLR	LPTMR01_BA+0x48	R/W	LPTMR0 PWM Clear Counter Register	0x0000_0000
LPTMR1_PWMCNTCLR	LPTMR01_BA+0x148	R/W	LPTMR1 PWM Clear Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**CNTCLR**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTCLR**	**Clear PWM Counter Control Bit** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0x0000 in up count type.

Low Power Timer PWM Period Register (LPTMRx_PWMPERIOD)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMPERIOD	LPTMR01_BA+0x4C	R/W	LPTMR0 PWM Period Register	0x0000_0000
LPTMR1_PWMPERIOD	LPTMR01_BA+0x14C	R/W	LPTMR1 PWM Period Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**PWM Period Register** In up count type: PWM counter counts from 0 to PERIOD, and restarts from 0. In up count type: PWM period time = (PERIOD + 1) * (CLKPSC + 1) * LPTMRx_PWMCLK.

Low Power Timer PWM Comparator Register (LPTMRx_PWMCMPDAT)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMCMPDAT	LPTMR01_BA+0x50	R/W	LPTMR0 PWM Comparator Register	0x0000_0000
LPTMR1_PWMCMPDAT	LPTMR01_BA+0x150	R/W	LPTMR1 PWM Comparator Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMP**	**PWM Comparator Register** PWM CMP is used to compare with PWM CNT to generate PWM output waveform, interrupt events and trigger EADC, PDMA, and DAC start conversion.

Low Power Timer PWM Counter Register (LPTMRx_PWMCNT)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMCNT	LPTMR01_BA+0x54	R	LPTMR0 PWM Counter Register	0x0000_0000
LPTMR1_PWMCNT	LPTMR01_BA+0x154	R	LPTMR1 PWM Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNT**	**PWM Counter Value Register (Read Only)** User can monitor CNT to know the current counter value in 16-bit period counter.

Low Power Timer PWM Pin Output Polar Control Register (LPTMRx_PWMPOLCTL)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMPOLCTL	LPTMR01_BA+0x58	R/W	LPTMR0 PWM Pin Output Polar Control Register	0x0000_0000
LPTMR1_PWMPOLCTL	LPTMR01_BA+0x158	R/W	LPTMR1 PWM Pin Output Polar Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**PINV**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PINV**	**PWMx Output Pin Polar Control Bit** The bit is used to control polarity state of PWMx_OUT pin. 0 = PWMx_OUT pin polar inverse Disabled. 1 = PWMx_OUT polar inverse Enabled. **Note:** Set POSEL (LPTMRx_PWMPOCTL[8]) to select LPTMRx or LPTMRx_EXT as PWMx output pin.

Low Power Timer PWM Pin Output Control Register (LPTMRx PWMPOCTL)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMPOCTL	LPTMR01_BA+0x5C	R/W	LPTMR0 PWM Pin Output Control Register	0x0000_0000
LPTMR1_PWMPOCTL	LPTMR01_BA+0x15C	R/W	LPTMR1 PWM Pin Output Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**POSEL**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**POEN**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**POSEL**	**PWM Output Pin Select** 0 = PWMx_OUT pin is LPTMRx. 1 = PWMx_OUT pin is LPTMRx_EXT.
[7:1]	**Reserved**	Reserved.
[0]	**POEN**	**PWMx Output Pin Enable Bit** 0 = PWMx_OUT pin at tri-state mode. 1 = PWMx_OUT pin in output mode. **Note:** Set POSEL (LPTMRx_PWMPOCTL[8]) to select LPTMRx or LPTMRx_EXT as PWMx output pin.

Low Power Timer PWM Interrupt Enable Register 0 (LPTMRx_PWMINTEN0)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMINTEN0	LPTMR01_BA+0x60	R/W	LPTMR0 PWM Interrupt Enable Register 0	0x0000_0000
LPTMR1_PWMINTEN0	LPTMR01_BA+0x160	R/W	LPTMR1 PWM Interrupt Enable Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**CMPUIEN**	**PIEN**	**Reserved**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CMPUIEN**	**PWM Compare Up Count Interrupt Enable Bit** 0 = Compare up count interrupt Disabled. 1 = Compare up count interrupt Enabled.
[1]	**PIEN**	**PWM Period Point Interrupt Enable Bit** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled.
[0]	**Reserved**	Reserved.

Low Power Timer PWM Interrupt Status Register 0 (LPTMRx_PWMINTSTS0)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMINTSTS0	LPTMR01_BA+0x64	R/W	LPTMR0 PWM Interrupt Status Register 0	0x0000_0000
LPTMR1_PWMINTSTS0	LPTMR01_BA+0x164	R/W	LPTMR1 PWM Interrupt Status Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CMPUIF	PIF	Reserved

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CMPUIF**	**PWM Compare Up Count Interrupt Flag** This bit is set by hardware when LPTMRx_PWM counter in up count direction and reaches CMP. **Note:** If CMP equal to PERIOD, there is no CMPUIF flag in up count type. **Note 2:** This bit is cleared by writing 1 to it.
[1]	**PIF**	**PWM Period Point Interrupt Flag** This bit is set by hardware when LPTMRx_PWM counter reaches PERIOD. **Note:** This bit is cleared by writing 1 to it.
[0]	**Reserved**	Reserved.

Low Power Timer PWM Trigger Control Register (LPTMRx PWMTRGCTL)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMTRGCTL	LPTMR01_BA+0x68	R/W	LPTMR0 PWM Trigger Control Register	0x0000_0000
LPTMR1_PWMTRGCTL	LPTMR01_BA+0x168	R/W	LPTMR1 PWM Trigger Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						PWMTRGLPPDMA	Reserved
7	6	5	4	3	2	1	0
Reserved					TRGEN	TRGSEL	

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**PWMTRGLPPDMA**	**PWM Counter Event Trigger LPPDMA Conversion Enable Bit** If this bit is set to 1, PWM can trigger LPPDMA conversion. 0 = PWM trigger LPPDMA Disabled. 1 = PWM trigger LPPDMA Enabled. **Note:** Set TRGSEL (LPTMRx_PWMTRGCTL[1:0]) to select PWM trigger conversion source. **Note:** TRGEN (LPTMRx_PWMTRGCTL[2]) is needed to be set to 1.
[8:3]	**Reserved**	Reserved.
[2]	**TRGEN**	**Trigger Low power IPs Enable Bit** f this bit is set to 1, each timer time-out event or capture event can trigger Low Power IP. 0 = Low Power Timer PWM counter event trigger Low Power IPs Disabled. 1 = Low Power Timer PWM counter event trigger Low Power IPs Enabled.
[1:0]	**TRGSEL**	**PWM Counter Event Source Select to Trigger Conversion** 00 = Trigger conversion at period point (PIF). 01 = Trigger conversion at compare up count point (CMPUIF). 10 = Trigger conversion at period or compare up count point (PIF or CMPUIF). 11 = Reserved.

Low Power Timer PWM Status Register (LPTMRx_PWMSTATUS)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMSTATUS	LPTMR01_BA+0x6C	R/W	LPTMR0 PWM Status Register	0x0000_0000
LPTMR1_PWMSTATUS	LPTMR01_BA+0x16C	R/W	LPTMR1 PWM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							PWMINTWKF
7	6	5	4	3	2	1	0
Reserved							CNTMAXF

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**PWMINTWKF**	**PWM Interrupt Wake-up Flag** 0 = PWM interrupt wake-up has not occurred. 1 = PWM interrupt wake-up has occurred. **Note:** This bit is cleared by writing 1 to it.
[7:1]	**Reserved**	Reserved.
[0]	**CNTMAXF**	**PWM Counter Equal to 0xFFFF Flag** 0 = The PWM counter value never reached its maximum value 0xFFFF. 1 = The PWM counter value has reached its maximum value. **Note:** This bit is cleared by writing 1 to it.

Low Power Timer PWM Period Buffer Register (LPTMRx PWMPBUF)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMPBUF	LPTMR01_BA+0x70	R	LPTMR0 PWM Period Buffer Register	0x0000_0000
LPTMR1_PWMPBUF	LPTMR01_BA+0x170	R	LPTMR1 PWM Period Buffer Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PBUF**	**PWM Period Buffer Register (Read Only)** Used as PERIOD active register.

Low Power Timer PWM Comparator Buffer Register (LPTMRx PWMCMPBUF)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMCMPBUF	LPTMR01_BA+0x74	R	LPTMR0 PWM Comparator Buffer Register	0x0000_0000
LPTMR1_PWMCMPBUF	LPTMR01_BA+0x174	R	LPTMR1 PWM Comparator Buffer Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPBUF**	**PWM Comparator Buffer Register (Read Only)** Used as CMP active register.

Low Power Timer PWM Interrupt Flag Accumulator Register (LPTMRx_PWMIFA)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMIFA	LPTMR01_BA+0xA8	R/W	LPTMR0 PWM Interrupt Flag Accumulator Register	0x0000_0000
LPTMR1_PWMIFA	LPTMR01_BA+0x1A8	R/W	LPTMR1 PWM Interrupt Flag Accumulator Register	0x0000_0000

31	30	29	28	27	26	25	24
IFAEN	Reserved	IFASEL		Reserved			STPMOD
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
IFACNT							
7	6	5	4	3	2	1	0
IFACNT							

Bits	Description	
[31]	**IFAEN**	**PWM Interrupt Flag Accumulator Enable Bit** 0 = PWM interrupt flag accumulator function Disabled. 1 = PWM interrupt flag accumulator function Enabled.
[30]	**Reserved**	Reserved.
[29:28]	**IFASEL**	**PWM Interrupt Flag Accumulator Source Select** 01 = Accumulate at each PWM period point. 10 = Accumulate at each PWM up-count compared point.
[27:25]	**Reserved**	Reserved.
[24]	**STPMOD**	**PWM Accumulator Stop Mode Enable Bit** 0 = PWM interrupt accumulator event to stop counting Disabled. 1 = PWM interrupt accumulator event to stop counting Enabled.
[23:16]	**Reserved**	Reserved.
[15:0]	**IFACNT**	**PWM Interrupt Flag Accumulator Counter** This field sets the count number which defines (IFACNT+1) times of specify PWM interrupt occurs to set IFAIF bit to request the PWM accumulator interrupt. PWM accumulator flag (IFAIF) will be set in every (IFACNT+1) times during the the PWM counter operation.

Low Power Timer PWM Accumulator Interrupt Status Register (LPTMRx_PWMAINTSTS)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMAINTSTS	LPTMR01_BA+0xAC	R/W	LPTMR0 PWM Accumulator Interrupt Flag Register	0x0000_0000
LPTMR1_PWMAINTSTS	LPTMR01_BA+0x1AC	R/W	LPTMR1 PWM Accumulator Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**IFAIF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**IFAIF**	**PWM Interrupt Flag Accumulator Interrupt Flag** This bit is set by hardware when the accumulator value reaches (IFACNT+1) **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** If APDMAEN (LPTMRx_PWMAPDMACTL[0]) is set, this bit will be auto clear after PDMA transfer done.

Low Power Timer PWM Accumulator Interrupt Enable Register (LPTMRx_PWMAINTEN)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMAINTEN	LPTMR01_BA+0xB0	R/W	LPTMR0 PWM Accumulator Interrupt Enable Register	0x0000_0000
LPTMR1_PWMAINTEN	LPTMR01_BA+0x1B0	R/W	LPTMR1 PWM Accumulator Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**IFAIEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**IFAIEN**	**PWM Interrupt Flag Accumulator Interrupt Enable Bit** 0 = Interrupt Flag Accumulator interrupt Disabled. 1 = Interrupt Flag Accumulator interrupt Enabled.

Low Power Timer PWM Accumulator LPPDMA Control Register (LPTMRx_PWMAPDMACTL)

Register	Offset	R/W	Description	Reset Value
LPTMR0_PWMAPDMACTL	LPTMR01_BA+0xB4	R/W	LPTMR0 PWM Accumulator LPPDMA Control Register	0x0000_0000
LPTMR1_PWMAPDMACTL	LPTMR01_BA+0x1B4	R/W	LPTMR1 PWM Accumulator LPPDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**APDMAEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**APDMAEN**	**PWM Accumulator LPPDMA Enable Bit** 0 = PWM interrupt accumulator event to trigger LPPDMA transfer Disabled. 1 = PWM interrupt accumulator event to trigger LPPDMA transfer Enabled.

6.13 Tick Timer Controller (TTMR)

6.13.1 Overview

The tick timer controller includes two 32-bit timers, TTMR0 ~ TTMR1, allowing user to easily implement a timer control for applications. The tick timer can perform functions, such as, delay timing,.

6.13.2 Features

6.13.2.1 Tick Timer Function Features

- Two sets of 32-bit timers, each timer having one 24-bit up counter and one 8-bit prescale counter
- Independent clock source for each timer
- Provides one-shot, periodic and continuous counting operation modes
- 24-bit up counter value is readable through CNT (TTMRx_CNT[23:0])
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated
- Supports time-out interrupt signal to trigger LPADC0, LPI2C0, LPSPI0, LPUART, , and LPPDMA function

6.13.3 Block Diagram

The tick timer controller block diagram and clock control are shown as follows.

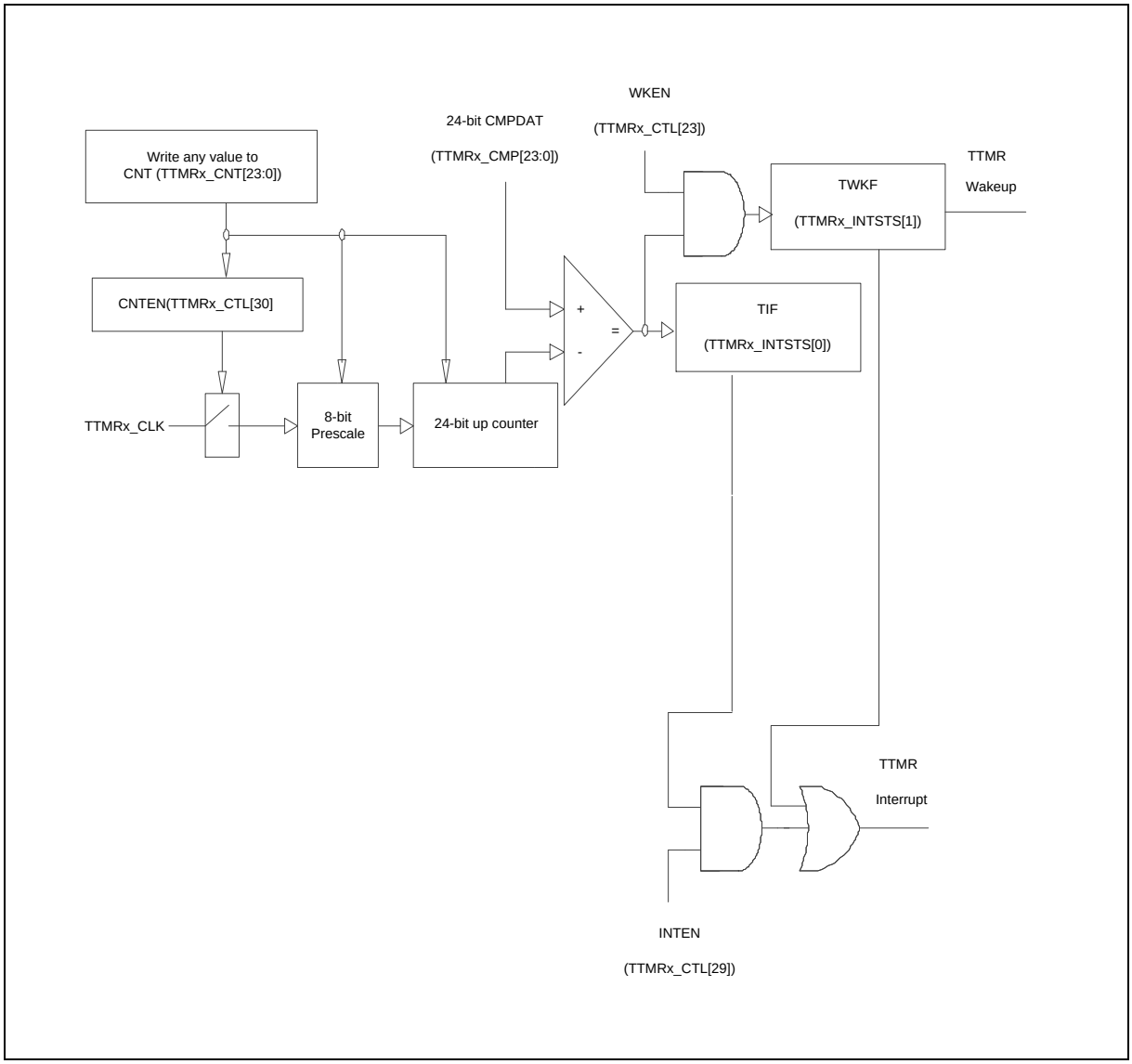


Figure 6.13-1 Tick Timer Controller Block Diagram

The clock source of TTMR0 ~ TTMR1 in timer mode can be enabled in TTMRxCKEN (LPSCC_CLKEN0[23:22]) and selected as different frequency in TTMR0SEL (LPSCC_CLKSEL0[5:4]) for TTMR0, TTMR1SEL (LPSCC_CLKSEL0[7:6]) for TTMR1.

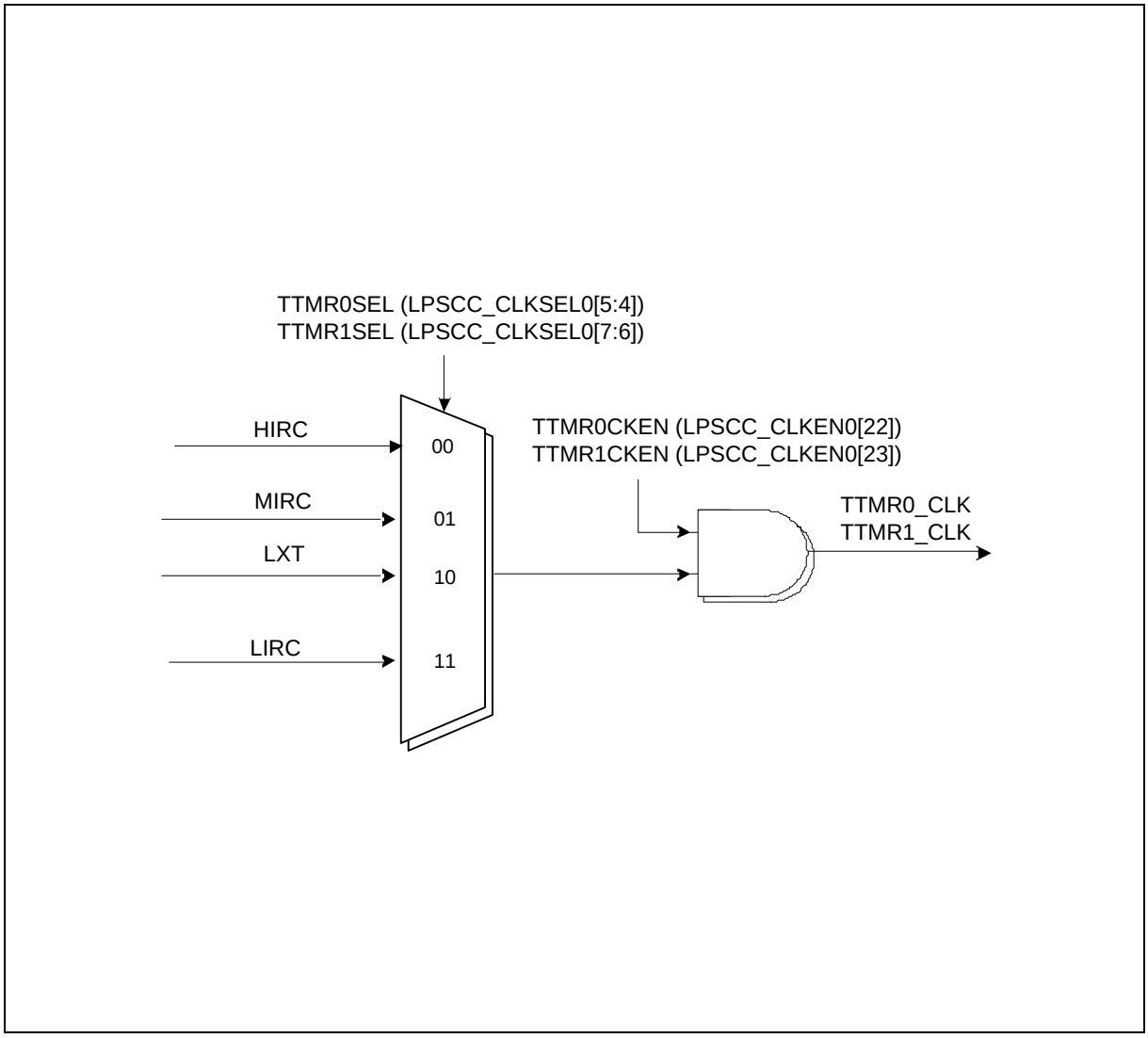


Figure 6.13-2 Clock Source of Tick Timer Controller

6.13.4 Basic Configuration

6.13.4.1 TTMR01 Basic Configurations

- Clock Source Configuration
  - Enable TTMR0 peripheral clock in TTMR0CKEN (LPSCC_CLKEN0[22]).
  - Enable TTMR1 peripheral clock in TTMR1CKEN (LPSCC_CLKEN0[23]).
- Reset Configuration
  - Reset TTMR0 controller in TTMR0RST (LPSCC_IPRST0[22]).
  - Reset TTMR1 controller in TTMR1RST (LPSCC_IPRST0[23]).

6.13.5 Functional Description

6.13.5.1 Tick Timer Interrupt Flag

The tick timer controller supports the following interrupt flags: TIF (TTMRx_INTSTS[0]) which is set while tick timer counter value CNT (TTMRx_CNT[23:0]) matches the tick timer compared value CMPDAT (TTMRx_CMP[23:0]),. The TWKF (TTMRx_INTSTS[1]) bit indicates the interrupt wake-up flag status of timer. Set WKEN (TTMRx_CTL[23] to 1 can use wake-up function.

6.13.5.2 Tick Timer Counting Mode

The tick timer controller provides four timer counting modes: one-shot, periodic, and continuous counting operation modes:

One–shot Mode

If the tick timer controller is configured at one-shot mode (TTMRx_CTL[28:27] is 00) and CNTEN (TTMRx_CTL[30]) is set, the tick timer counter starts up counting. Once the CNT (TTMRx_CNT[23:0]) value reaches CMPDAT (TTMRx_CMP[23:0]) value, the TIF (TTMRx_INTSTS[0]) will be set to 1, CNT value and CNTEN bit is cleared automatically by tick timer controller then timer counting operation stops. In the meantime, if the INTEN (TTMRx_CTL[29]) is enabled, the tick timer interrupt signal is generated and sent to NVIC to inform CPU also.

User can monitor the RSTACT (TTMRx_CNT[31]) to ensure counter reset operation active and disable ICE debug mode acknowledgement effects TTMR counting by setting ICEDEBUG (TTMRx_CTL[31]) to 1.

Periodic Mode

If the tick timer controller is configured at periodic mode (TTMRx_CTL[28:27] is 01) and CNTEN (TTMRx_CTL[30]) is set, the tick timer counter starts up counting. Once the CNT (TTMRx_CNT[23:0]) value reaches CMPDAT (TTMRx_CMP[23:0]) value, the TIF (TTMRx_INTSTS[0]) will be set to 1, CNT value will be cleared automatically by tick timer controller and tick timer counter operates counting again. In the meantime, if the INTEN (TTMRx_CTL[29]) bit is enabled, the tick timer interrupt signal is generated and sent to NVIC to inform CPU also. In this mode, the tick timer controller operates counting and compares with CMPDAT value periodically until the CNTEN bit is cleared by user.

User can set PERIOSEL (TTMRx_CTL[20]) to select Tick Timer behavior.

Continuous Counting Mode

If the tick timer controller is configured at continuous counting mode (TTMRx_CTL[28:27] is 11) and CNTEN (TTMRx_CTL[30]) is set, the tick timer counter starts up counting. Once the CNT (TTMRx_CNT[23:0]) value reaches CMPDAT (TTMRx_CMP[23:0]) value, the TIF (TTMRx_INTSTS[0]) will be set to 1 and CNT value keeps up counting. In the meantime, if the INTEN (TTMRx_CTL[29]) is enabled, the tick timer interrupt signal is generated and sent to NVIC to inform CPU also. User can change different CMPDAT value immediately without disabling timer counting and restarting timer counting in this mode.

For example, CMPDAT value is set as 80, first. The TIF will set to 1 when CNT value is equal to 80, timer counter is kept counting and CNT value will not goes back to 0, it continues to count 81, 82, 83, … to (2^{24} -1), 0, 1, 2, 3, … to (2^{24} -1) again and again. Next, if user programs CMPDAT value as 200 and clears TIF, the TIF will set to 1 again when CNT value reaches to 200. At last, user programs CMPDAT as 500 and clears TIF, the TIF will set to 1 again when CNT value reaches to 500.

In this mode, the tick timer counting is continuous. So this operation mode is called as continuous counting mode.

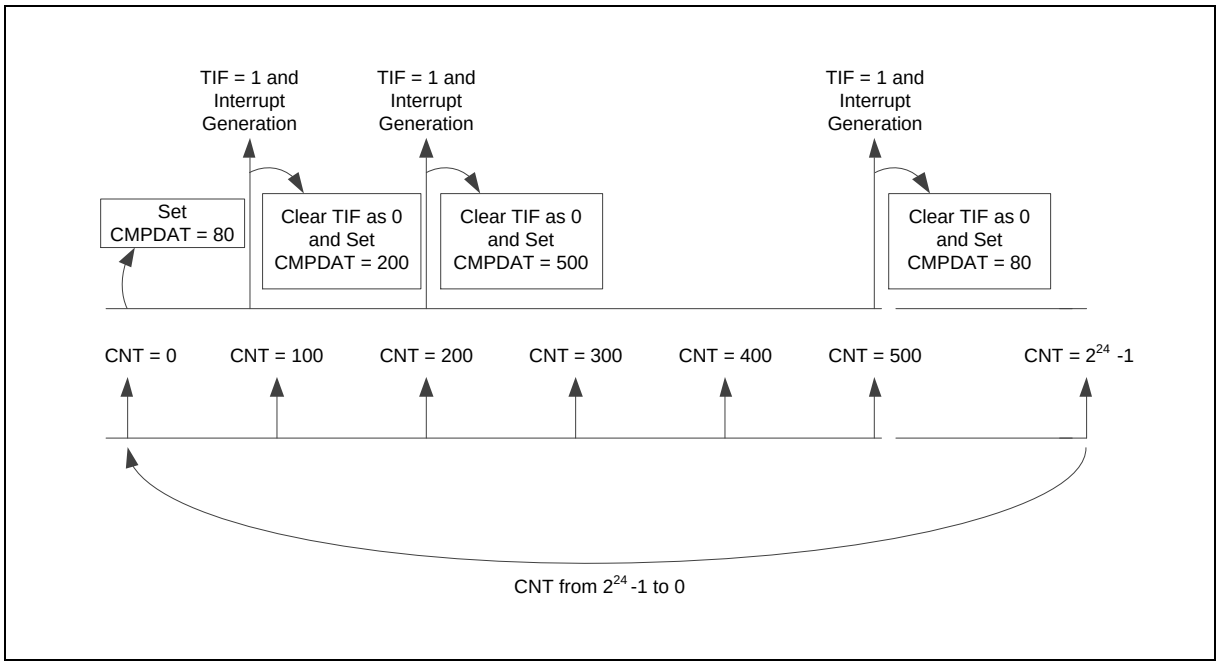


Figure 6.13-3 Continuous Counting Mode

6.13.5.3 Tick Timer Trigger Function

The tick timer controller provides timer time-out interrupt LPADC, LPI2C, LPSPI, LPUART and LPPDMA.

When the TRGEN (TTMRx_TRGCTL[1]) is set, if the tick timer interrupt signal is generated, the tick timer controller will generate a trigger pulse to other Low Power IPs.

When the TRGLPPDMA (TTMRx_TRGCTL[4]) is set, if the tick timer interrupt signal is generated, the tick timer controller will trigger LPPDMA.

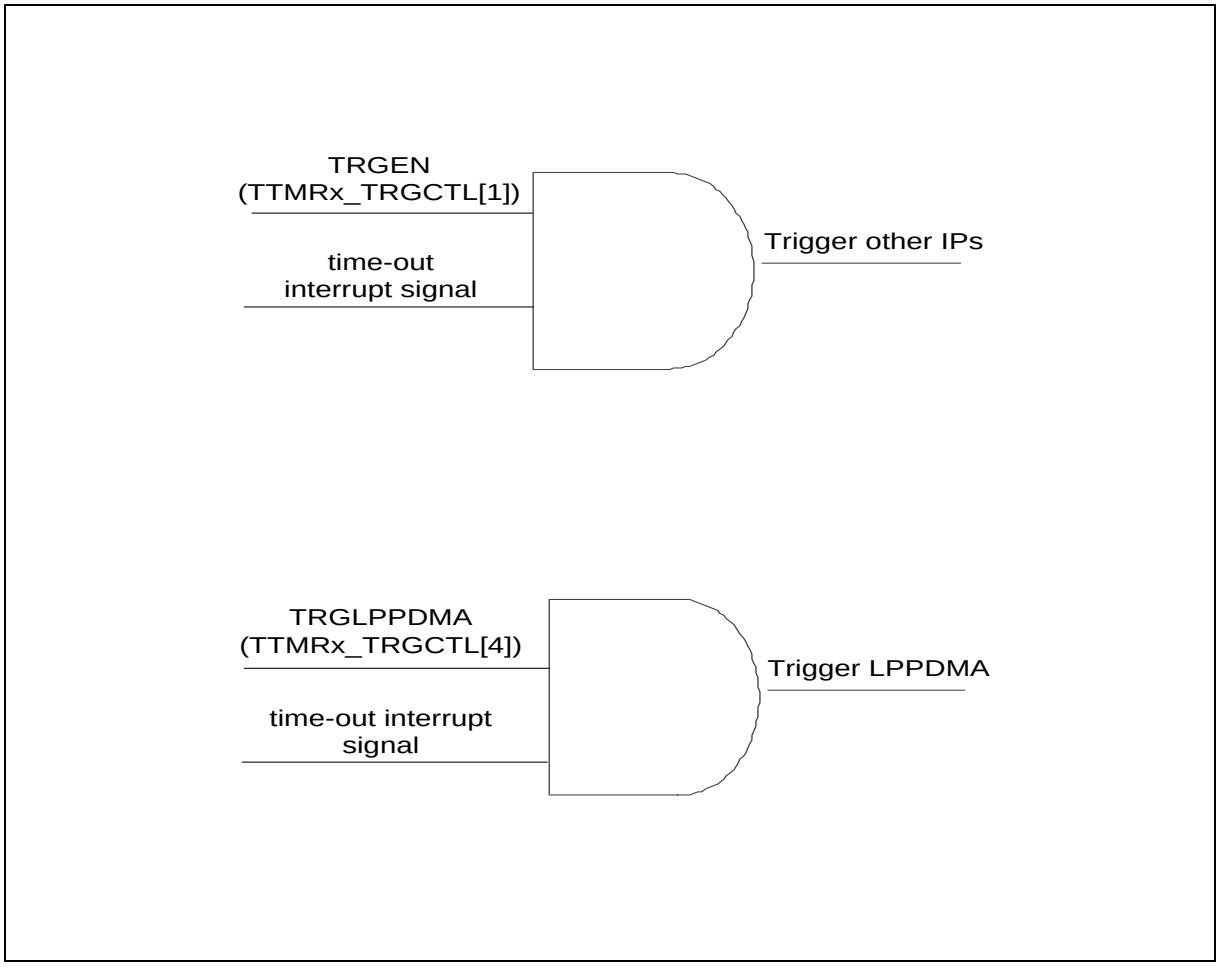


Figure 6.13-4 Internal Tick Timer Trigger

6.13.5.4 Auto-operation Mode

The tick timer controller (TTMR) supports auto-operation mode to trigger other low power IPs when chip enters NPD0/1/2/3/4 mode. Configure TRGEN(TTMRx_TRGCTL[1]) to enable auto-operation mode first. The tick timer controller can broadcast a 4T pulse when TRGEN(TTMR_TRGCTL[1]) condition is met, then other low power IPs will receive the pulse. Especially for LPPDMA, the tick timer controller needs to configure TRGLPPDMA(TTMRx_TRGCTL[4]). The tick timer controller and LPPDMA have a handshake mechanism. The tick timer controller sends a request to LPPDMA ,and then LPPDMA will send an ack. This mechanism is different from other low power IPs since other low power IPs do not need to send an ack to the tick timer controller.

Trigger Low Power IPs:

1. Enable TTMR peripheral clock.

   Set PDCLKEN(TTMRx_CTL[16]) to open the engine clock in NPD0/1/2/3/4 mode. Engine clock should be always on to operate the Timer Function.

2. Configure Trigger Function

   Set TRGEN(TTMRx_TRGCTL[1]) first. The tick timer controller can use counting mode to trigger other low power IPs.

Trigger LPPDMA:

1. Enable TTMR peripheral clock.

Set PDCLKEN(TTMRx_CTL[16]) to open the engine clock in NPD0/1/2/3/4 mode. Engine clock should be always on to operate the Timer Function. When the tick timer controller in counting mode trigger LPPDMA in NPD0/1/2/3/4 mode, AHB1, AHB1 bus matrix, LPPDMA and LPSRAM also need clock to work. But the AHB and APB clock are not always on in NPD0/1/2/3/4. AHB and APB clock is on when time-out flags occurs. Then AHB and APB clock will turn off after LPPDMA transfer done.

2. Configure Trigger Function

   Set TRGLPPDMA(TTMRx_TRGCTL[4]) first. The tick timer controller can trigger LPPDMA in Counting mode when time-out flag occurs.

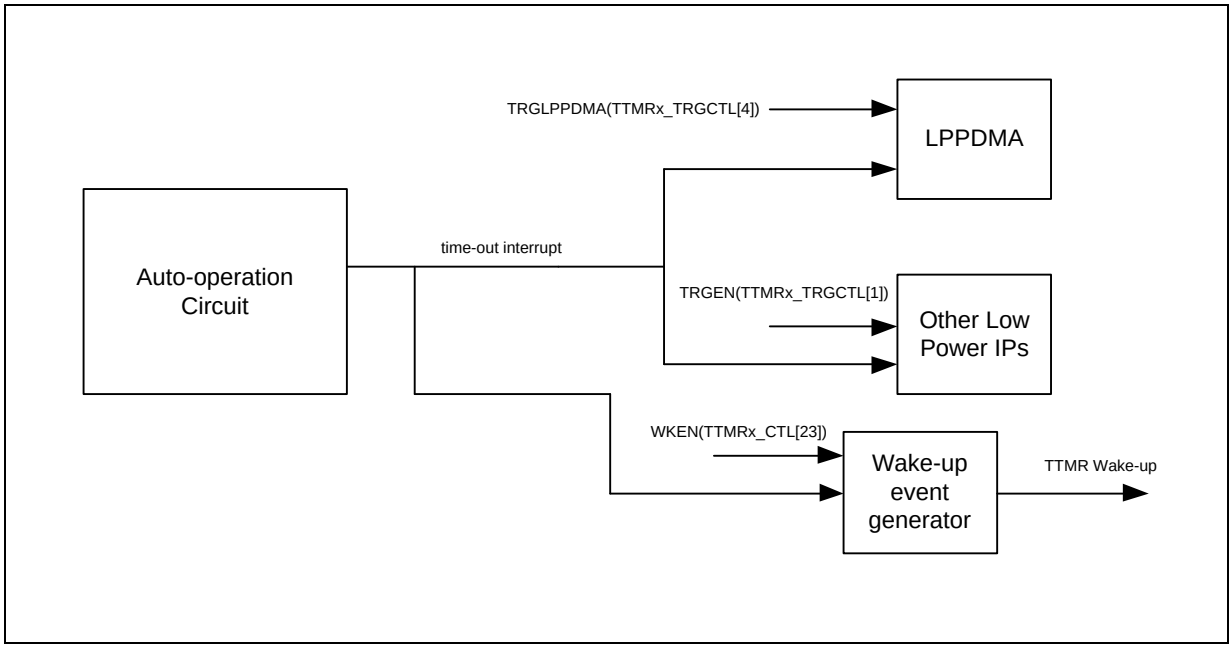


Figure 6.13-11 TTMR Auto-operation Mode Block Diagram

6.13.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
TTMR Base Address: **TTMR01_BA = 0x400E_5000**				
TTMR0_CTL	TTMR01_BA+0x00	R/W	TTMR0 Control Register	0x0000_0005
TTMR0_CMP	TTMR01_BA+0x04	R/W	TTMR0 Comparator Register	0x0000_0000
TTMR0_INTSTS	TTMR01_BA+0x08	R/W	TTMR0 Interrupt Status Register	0x0000_0000
TTMR0_CNT	TTMR01_BA+0x0C	R/W	TTMR0 Data Register	0x0000_0000
TTMR0_TRGCTL	TTMR01_BA+0x1C	R/W	TTMR0 Trigger Control Register	0x0000_0000
TTMR1_CTL	TTMR01_BA+0x100	R/W	TTMR1 Control Register	0x0000_0005
TTMR1_CMP	TTMR01_BA+0x104	R/W	TTMR1 Comparator Register	0x0000_0000
TTMR1_INTSTS	TTMR01_BA+0x108	R/W	TTMR1 Interrupt Status Register	0x0000_0000
TTMR1_CNT	TTMR01_BA+0x10C	R/W	TTMR1 Data Register	0x0000_0000
TTMR1_TRGCTL	TTMR01_BA+0x11C	R/W	TTMR1 Trigger Control Register	0x0000_0000

6.13.7 Register Description

Tick Timer Control Register (TTMRx_CTL)

Register	Offset	R/W	Description	Reset Value
TTMR0_CTL	TTMR01_BA+0x00	R/W	TTMR0 Control Register	0x0000_0005
TTMR1_CTL	TTMR01_BA+0x100	R/W	TTMR1 Control Register	0x0000_0005

31	30	29	28	27	26	25	24
ICEDEBUG	CNTEN	INTEN	OPMODE		Reserved	ACTSTS	Reserved
23	22	21	20	19	18	17	16
WKEN	Reserved		PERIOSEL	Reserved			PDCLKEN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PSC							

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects TIMER counting. TIMER counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. TIMER counter will keep going no matter CPU is held by ICE or not. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[30]	**CNTEN**	**Tick Timer Counting Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. **Note 1:** In stop status, and then set CNTEN to 1 will enable the 24-bit up counter to keep counting from the last stop counting value. **Note 2:** This bit is auto-cleared by hardware in one-shot mode (TTMR_CTL[28:27] = 00) when the tick timer time-out interrupt flag TIF (TTMRx_INTSTS[0]) is generated. **Note 3:** Set enable/disable this bit needs 2 * TTMR_CLK period to become active, user can read ACTSTS (TTMRx_CTL[25]) to check enable/disable command is completed or not.
[29]	**INTEN**	**Tick Timer Interrupt Enable Bit** 0 = Tick Timer time-out interrupt Disabled. 1 = Tick Timer time-out interrupt Enabled. **Note:** If this bit is enabled, when the tick timer time-out interrupt flag TIF is set to 1, the tick timer interrupt signal is generated and inform to CPU.
[28:27]	**OPMODE**	**Tick Timer Counting Mode Select** 00 = Tick timer controller is operated in One-shot mode. 01 = Tick timer controller is operated in Periodic mode. 11 = Tick timer controller is operated in Continuous Counting mode.

[26]	**Reserved**	Reserved.
[25]	**ACTSTS**	**Tick Timer Active Status Bit (Read Only)** This bit indicates the 24-bit up counter status. 0 = 24-bit up counter is not active. 1 = 24-bit up counter is active. **Note:** This bit may be active when CNT 0 is transitioned to CNT 1.
[24]	**Reserved**	Reserved.
[23]	**WKEN**	**Wake-up Function Enable Bit** If this bit is set to 1, while timer interrupt flag TIF (TTMRx_INTSTS[0]) is 1 and INTEN (TTMRx_CTL[29]) is enabled, the tick timer interrupt signal will generate a wake-up trigger event to CPU. 0 = Wake-up function Disabled if timer interrupt signal generated. 1 = Wake-up function Enabled if timer interrupt signal generated.
[22:21]	**Reserved**	Reserved.
[20]	**PERIOSEL**	**Periodic Mode Behavior Selection Enable Bit** 0 = The behavior selection in periodic mode is Disabled. When user updates CMPDAT while timer is running in periodic mode, CNT will be reset to default value. 1 = The behavior selection in periodic mode is Enabled. When user updates CMPDAT while timer is running in periodic mode, the limitations as bellows list, If updated CMPDAT value > CNT, CMPDAT will be updated and CNT keep running continually. If updated CMPDAT value = CNT, timer time-out interrupt will be asserted immediately. If updated CMPDAT value < CNT, CNT will be reset to default value.
[19:17]	**Reserved**	Reserved.
[16]	**PDCLKEN**	**Power-down Engine Clock Enable** 0 = Disable engine clock in Power-down mode. 1 = Enable engine clock in Power-down mode.
[15:8]	**Reserved**	Reserved.
[7:0]	**PSC**	**Prescale Counter** Tick Timer input clock or event source is divided by (PSC+1) before it is fed to the tick timer up counter. If this field is 0 (PSC = 0), then there is no scaling. **Note:** Update prescale counter value will reset internal 8-bit prescale counter and 24-bit up counter value.

Tick Timer Comparator Register (TTMRx_CMP)

Register	Offset	R/W	Description	Reset Value
TTMR0_CMP	TTMR01_BA+0x04	R/W	TTMR0 Comparator Register	0x0000_0000
TTMR1_CMP	TTMR01_BA+0x104	R/W	TTMR1 Comparator Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPDAT							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CMPDAT**	**Tick Timer Comparator Value** CMPDAT is a 24-bit compared value register. When the internal 24-bit up counter value is equal to CMPDAT value, the TIF (TTMRx_INTSTS[0] Tick Timer Interrupt Flag) will set to 1. Time-out period = (Period of timer clock input) * (8-bit PSC + 1) * (24-bit CMPDAT). **Note 1:** Never write 0x0 or 0x1 in CMPDAT field, or the core will run into unknown state. **Note 2:** When timer is operating at continuous counting mode, the 24-bit up counter will keep counting continuously even if user writes a new value into CMPDAT field. But if timer is operating at other modes, the 24-bit up counter will restart counting from 0 and using newest CMPDAT value to be the tick timer compared value while user writes a new value into CMPDAT field.

Tick Timer Interrupt Status Register (TTMRx_INTSTS)

Register	Offset	R/W	Description	Reset Value
TTMR0_INTSTS	TTMR01_BA+0x08	R/W	TTMR0 Interrupt Status Register	0x0000_0000
TTMR1_INTSTS	TTMR01_BA+0x108	R/W	TTMR1 Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TWKF	TIF

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TWKF**	**Tick Timer Wake-up Flag** This bit indicates the interrupt wake-up flag status of timer. 0 = Tick Timer does not cause CPU wake-up. 1 = CPU wake-up from Idle or Power-down mode if timer time-out interrupt signal generated. **Note:** This bit is cleared by writing 1 to it.
[0]	**TIF**	**Tick Timer Interrupt Flag** This bit indicates the interrupt flag status of Tick Timer while 24-bit timer up counter CNT (TTMRx_CNT[23:0]) value reaches to CMPDAT (TTMRx_CMP[23:0]) value. 0 = No effect. 1 = CNT value matches the CMPDAT value. **Note:** This bit is cleared by writing 1 to it.

Tick Timer Data Register (TTMRx_CNT)

Register	Offset	R/W	Description	Reset Value
TTMR0_CNT	TTMR01_BA+0x0C	R/W	TTMR0 Data Register	0x0000_0000
TTMR1_CNT	TTMR01_BA+0x10C	R/W	TTMR1 Data Register	0x0000_0000

31	30	29	28	27	26	25	24
RSTACT	Reserved						
23	22	21	20	19	18	17	16
CNT							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

Bits	Description	
[31]	**RSTACT**	**Tick Timer Data Register Reset Active (Read Only)** This bit indicates if the counter reset operation active. When user writes this CNT register, tick timer starts to reset its internal 24-bit timer up-counter to 0 and reload 8-bit pre-scale counter. At the same time, timer set this flag to 1 to indicate the counter reset operation is in progress. Once the counter reset operation done, tick timer clear this bit to 0 automatically. 0 = Reset operation is done. 1 = Reset operation triggered by writing TTMRx_CNT is in progress.
[30:24]	**Reserved**	Reserved.
[23:0]	**CNT**	**Tick Timer Data Register** Read operation: Read this register to get CNT value. For example: Write operation: Writing any value to this register will reset current CNT value to 0 and reload internal 8-bit prescale counter.

Tick Timer Trigger Control Register (TTMRx_TRGCTL)

Register	Offset	R/W	Description	Reset Value
TTMR0_TRGCTL	TTMR01_BA+0x1C	R/W	TTMR0 Trigger Control Register	0x0000_0000
TTMR1_TRGCTL	TTMR01_BA+0x11C	R/W	TTMR1 Trigger Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			TRGLPPDMA	Reserved		TRGEN	Reserved

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**TRGLPPDMA**	**Trigger LPPDMA Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be triggered LPPDMA transfer. 0 = Tick Timer interrupt trigger LPPDMA Disabled. 1 = Tick Timer interrupt trigger LPPDMA Enabled.
[3:2]	**Reserved**	Reserved.
[1]	**TRGEN**	**Trigger Low power IPs Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be triggered Low Power IPs conversion. 0 = Tick Timer interrupt trigger Low Power IPs Disabled. 1 = Tick Timer interrupt trigger Low Power IPs Enabled.
[0]	**Reserved**	Reserved.

6.14 PWM Generator and Capture Timer (PWM)

6.14.1 Overview

The chip provides two PWM generators — PWM0 and PWM1. Each PWM supports 6 channels of PWM output or input capture. There is a 12-bit prescaler to support flexible clock to the 16-bit PWM counter with 16-bit comparator. The PWM counter supports up, down and up-down counter types. PWM uses comparator compared with counter to generate events. These events use to generate PWM pulse, interrupt and trigger signal for EADC, LPADC and DAC to start conversion.

The PWM generator supports two standard PWM output modes: Independent mode and Complementary mode, they have difference architecture. In Complementary mode, there are two comparators to generate various PWM pulse with 12-bit dead-time generator. For PWM output control unit, it supports polarity output, independent pin mask and brake functions.

The PWM generator also supports input capture function to latch PWM counter value to the corresponding register when input channel has a rising transition, falling transition or both transition is happened. Capture function also support PDMA to transfer captured data to memory.

6.14.2 Features

6.14.2.1 PWM Function Features

- Supports maximum clock frequency up to maximum 72MHz frequency
- Supports up to two PWM modules, each module provides 6 output channels
- Supports independent mode for PWM output/Capture input channel
- Supports complementary mode for 3 complementary paired PWM output channel
  - Dead-time insertion with 12-bit resolution
  - Two compared values during one period
- Supports 12-bit prescaler from 1 to 4096
- Supports 16-bit resolution PWM counter
  - Up, down and up/down counter operation type
- Supports mask function and tri-state enable for each PWM pin
- Supports brake function
  - Brake source from pin, analog comparator, UTCPD, EADC result monitor and system safety events (clock failed, Brown-out detection, SRAM parity error and CPU lockup)
  - Noise filter for brake source from pin
  - Edge detect brake source to control brake state until brake interrupt cleared
  - Level detect brake source to auto recover function after brake condition removed
- Supports interrupt on the following events:
  - PWM counter matches 0, period value or compared value
  - Brake condition happened
- Supports trigger EADC/LPADC/DAC on the following events:
  - PWM counter matches 0, period value or compared value
- Supports PDMA transfer for Interrupt Flag Accumulator Function
- Supports PWM output accumulator stop counter mode
- Support External Pin Trigger Function

6.14.2.2 Capture Function Features

- Supports 3-bit capture input noise filter
- Supports up to 12 capture input channels with 16-bit resolution
- Supports rising or falling capture condition
- Supports input rising/falling capture interrupt
- Supports rising/falling capture with counter reload option
- Supports PDMA transfer function for PWM all channels

6.14.3 Block Diagram

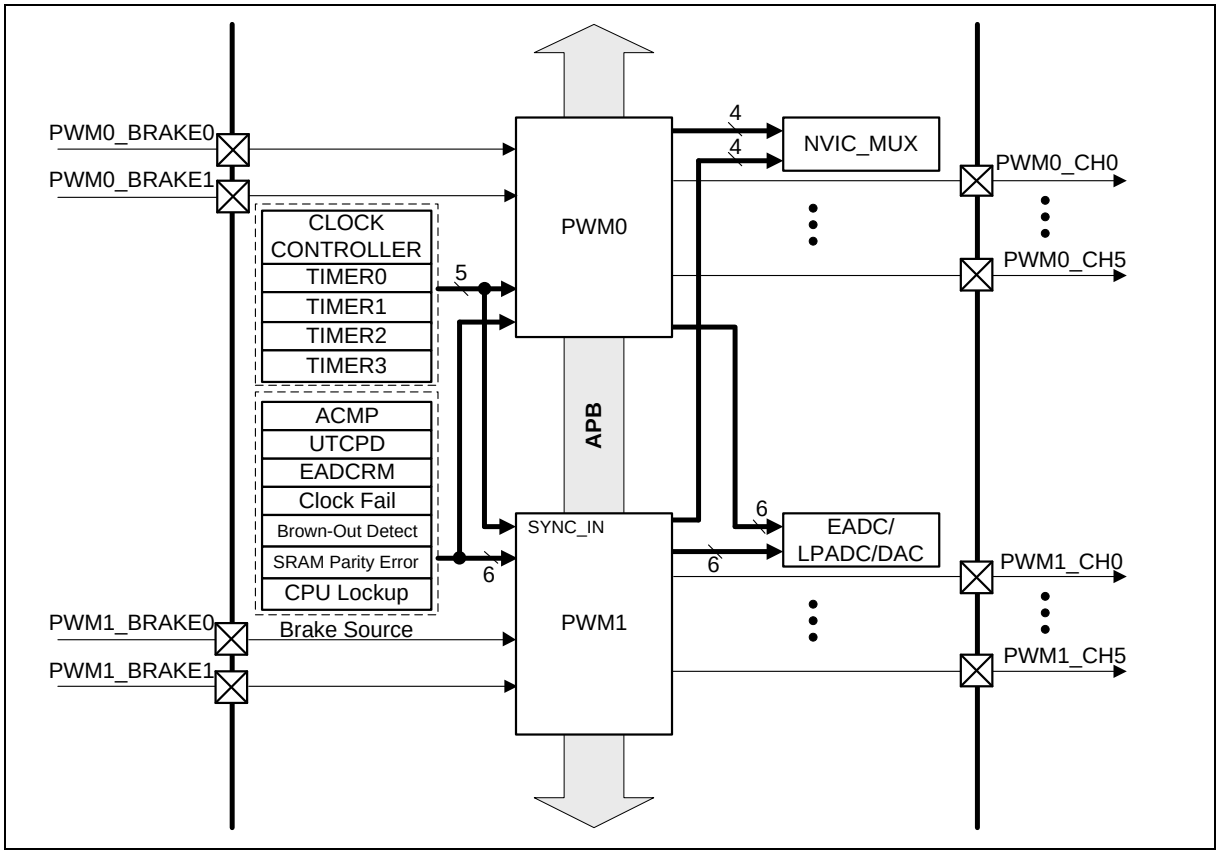


Figure 6.14-1 PWM Generator Overview Block Diagram

PWM Clock frequency can be set equal or double to PCLK frequency as Figure 6.14-2. For the detailed register setting, please refer to Table 6.14-1. Each PWM generator has three clock source inputs; each clock source can be selected from PWM Clock or four TIMER trigger PWM outputs as Figure 6.14-3 by ECLKSRC0 (PWM_CLKSRC[2:0]) for PWM_CLK0, ECLKSRC2 (PWM_CLKSRC[10:8]) for PWM_CLK2 and ECLKSRC4 (PWM_CLKSRC[18:16]) for PWM_CLK4.

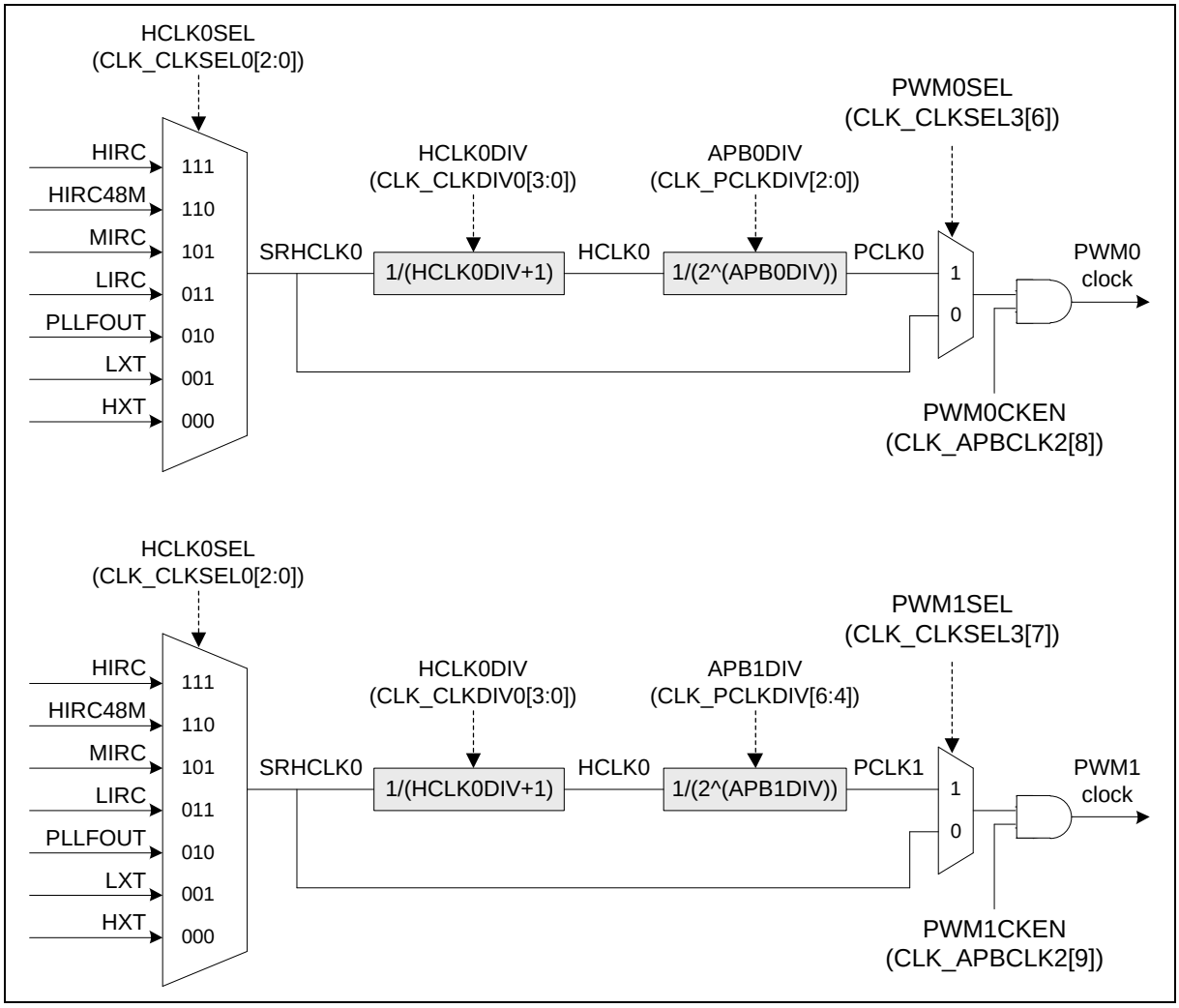


Figure 6.14-2 PWM System Clock Source Control

Frequency Ratio PCLK:PWM Clock	HCLK	PCLK	PWM Clock	HCLK0SEL CLK_CLKSEL0[2:0]	HCLKDIV CLK_CLKDIV0[3:0]	APBnDIV (CLK_PCLKDIV[2+4n:4n]), n denotes 0 or 1	PWMnSEL (CLK_CLKSEL3[6+n]), n denotes 0 or 1
1:1	HCLK	PCLK	PCLK	Don't care	Don't care	Don't care	1

Table 6.14-1 PWM Clock Source Control Registers Setting Table

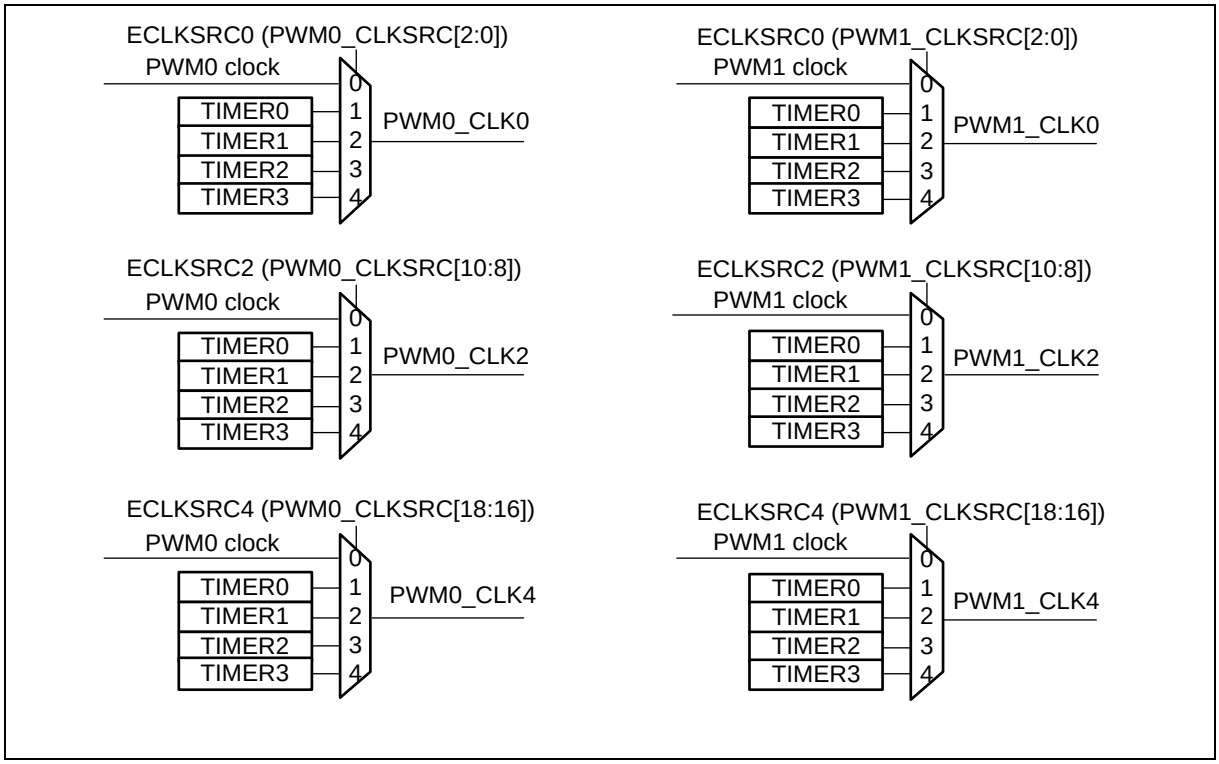


Figure 6.14-3 PWM Clock Source Control

Figure 6.14-4 and Figure 6.14-5 illustrate the architecture of PWM independent mode and complementary mode. No matter independent mode or complementary mode, paired channels' (PWM_CH0 and PWM_CH1, PWM_CH2 and PWM_CH3, PWM_CH4 and PWM_CH5) counters both come from the same clock source and prescaler. When counter count to 0, PERIOD (PWM_PERIODn[15:0]) or equal to comparator, events will be generated. These events are passed to corresponding generators to generate PWM pulse, interrupt signal and trigger signal for EADC/LPADC/DAC to start conversion. Output control is used to changing PWM pulse output state; brake function in output control also generates interrupt events. In complementary mode, even channel use odd channel comparator to generate events.

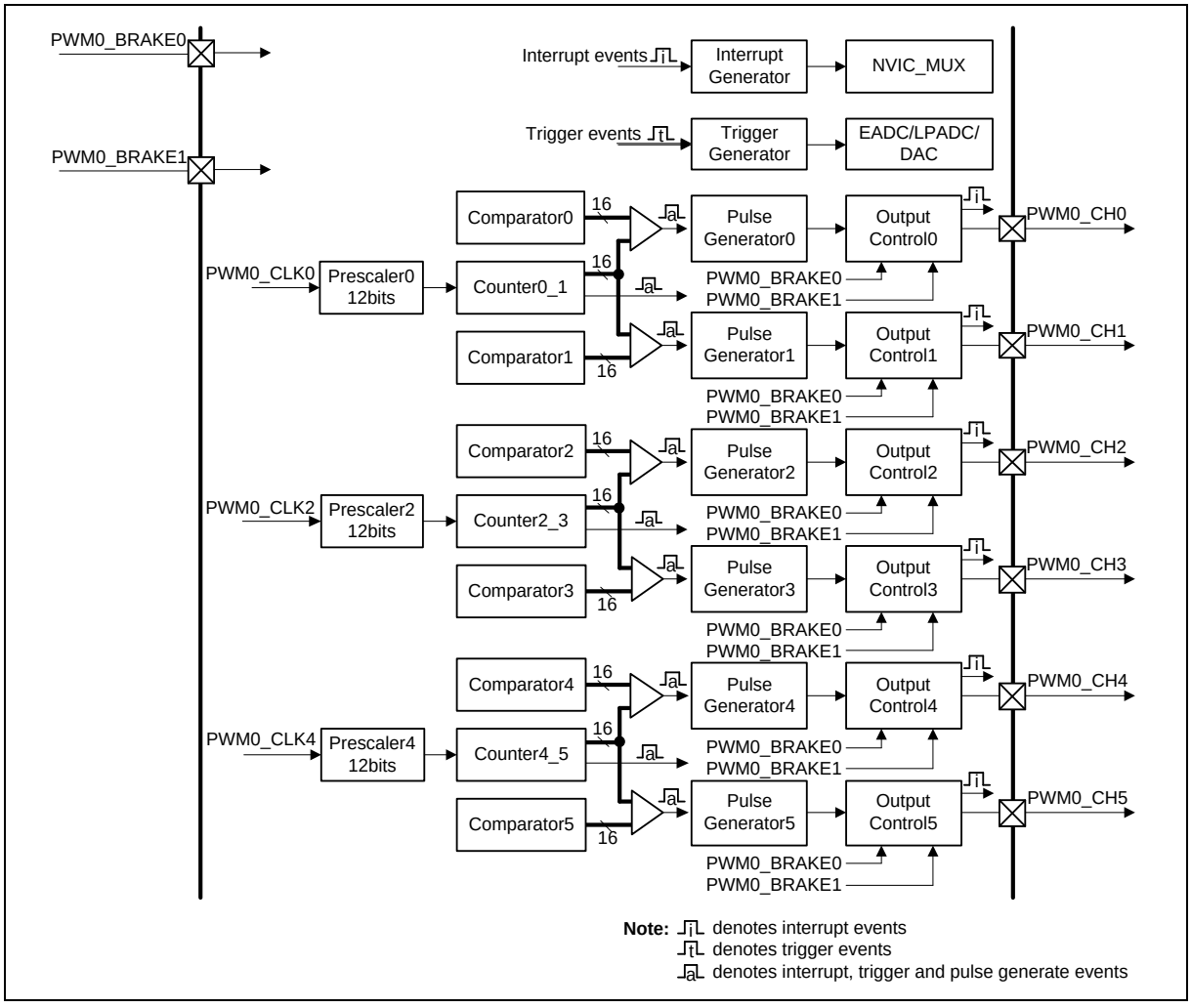


Figure 6.14-4 PWM Independent Mode Architecture Diagram

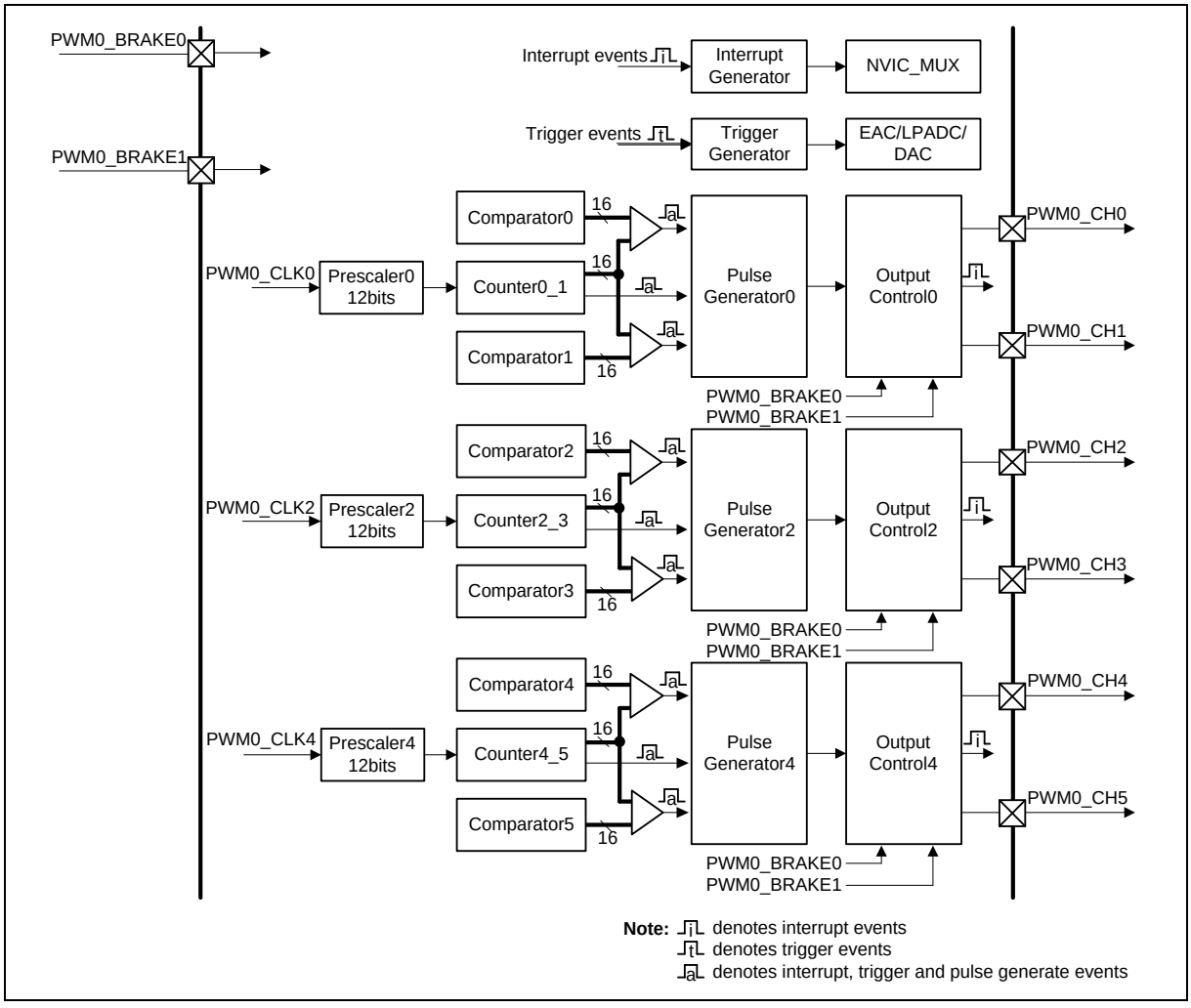


Figure 6.14-5 PWM Complementary Mode Architecture Diagram

6.14.4 Basic Configuration

6.14.4.1 PWM0 Basic Configuration

- Clock Source Configuration
  - Select the source of PWM0 peripheral clock on PWM0SEL (CLK_CLKSEL3[6])
  - Enable PWM0 peripheral clock in PWM0CKEN (CLK_APBCLK2[8]).
- Reset Configuration
  - Reset PWM0 in PWM0RST SYS_IPRST3[8]

6.14.4.2 PWM1 Basic Configuration

- Clock Source Configuration
  - Select the source of PWM1 peripheral clock on PWM1SEL (CLK_CLKSEL3[7])
  - Enable PWM1 peripheral clock in PWM1CKEN (CLK_APBCLK2[9]).
- Reset Configuration
  - Reset PWM1 in PWM1RST SYS_IPRST3[9]

6.14.5 Functional Description

6.14.5.1 PWM Prescaler

The PWM prescaler is used to divide clock source, prescaler counting CLKPSC +1 times, PWM counter only count once. The prescale double buffer is setting by CLKPSC (PWM_CLKPSCn[11:0], n = 0, 2, 4) bits. Figure 6.14-6 is an example of PWM channel 0 prescale waveform. The prescale counter will reload CLKPSC at the begin of the next prescale counter down-count.

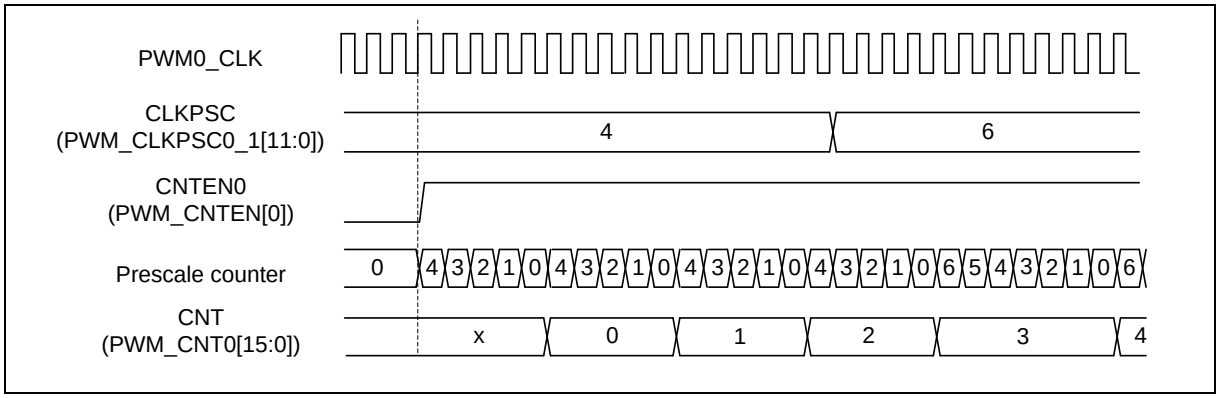


Figure 6.14-6 PWM0_CH0 Prescaler Waveform in Up Counter Type

6.14.5.2 PWM Counter

PWM supports 3 counter types operation: Up Counter, Down Counter and Up-Down Counter types.

For PWM channel0, CNT(PWM_CNT0[15:0]) can clear to 0x00 by CNTCLR0 (PWM_CNTCLR[0]). CNT will be cleared when prescale counter count to 0, and CNTCLR will be set 0 by hardware automatically.

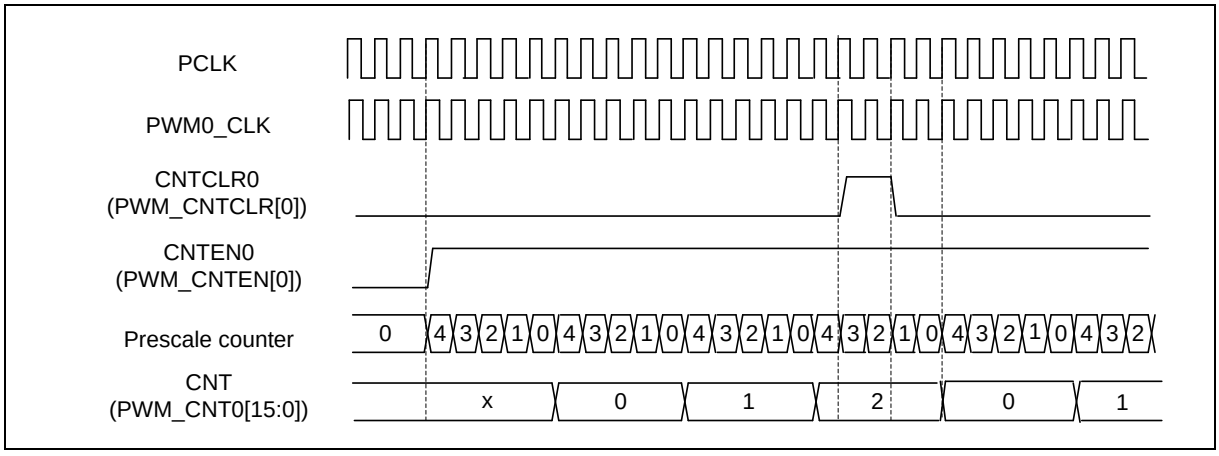


Figure 6.14-7 PWMx Counter Waveform when Setting clear counter

6.14.5.3 Up Counter Type

When PWM counter is set to up counter type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0, 2, 4) is 0x0, it starts up-counting from 0 to PERIOD (PWM_PERIODn[15:0], n = 0, 2, 4) to complete a PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0], n = 0, 2, 4) bits. PWM generates zero point event when the counter counts to 0 and prescale counts to 0. PWM generates period point event when the counter counts to PERIOD and prescale counts to 0. Figure 6.14-8 shows an example of up counter, wherein

PWM period time = (PERIOD+1) *(CLKPSC+1)* PWMx_CLK.

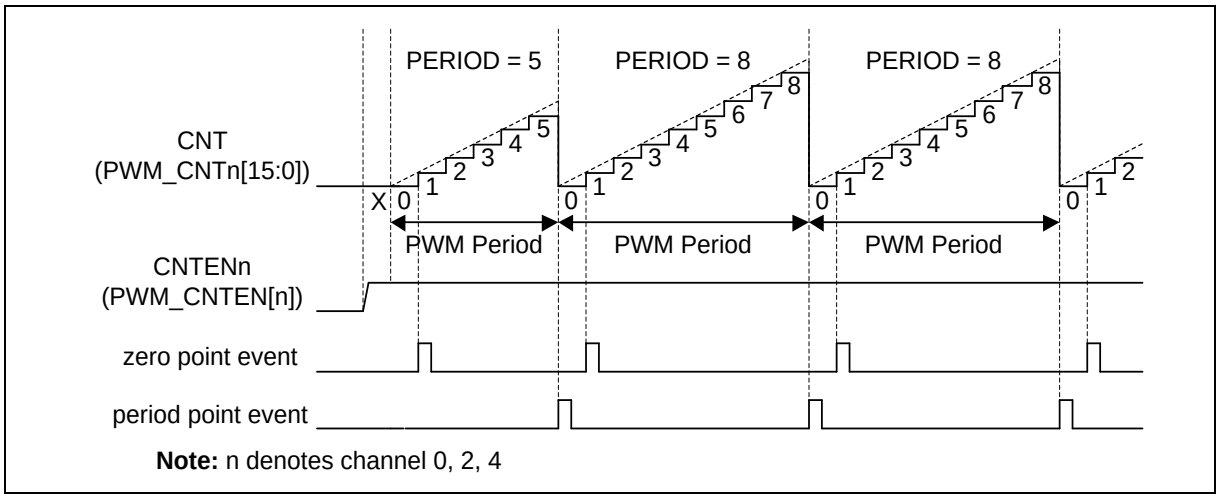


Figure 6.14-8 PWM Up Counter Type

6.14.5.4 Down Counter Type

When PWM counter is set to down counter type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0, 2, 4) is 0x1, it starts down-counting from PERIOD to 0 to complete a PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0], n = 0, 2, 4) bits. PWM generates zero point event when the counter counts to 0 and prescale counts to 0. PWM generates period point event when the counter counts to PERIOD and prescale counts to 0. Figure 6.14-9 shows an example of down counter, wherein

PWM period time = (PERIOD+1) * (CLKPSC+1) * PWMx_CLK.

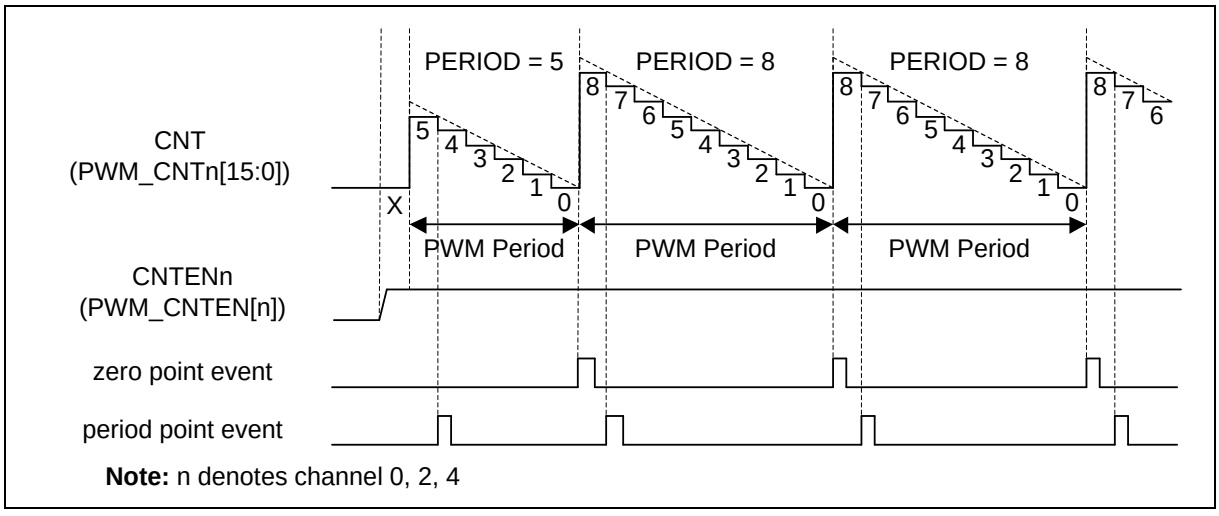


Figure 6.14-9 PWM Down Counter Type

6.14.5.5 Up-Down Counter Type

When PWM counter is set to up-down count type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0, 2, 4) is 0x2, it starts counting-up from 0 to PERIOD and then starts counting down to 0 to complete a PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0], n = 0, 2, 4) bits. PWM generates zero point event when the counter counts to 0 and prescale counts to 0. PWM generates center point event which is equal to period point event when the counter counts to PERIOD. Figure 6.14-10 shows an example of up-down counter, wherein

PWM period time =(2*PERIOD) * (CLKPSC+1) * PWMx_CLK.

The DIRF (PWM_CNTn[16], n = 0, 2, 4) bit is counter direction indicator flag, where high is up counting, and low is down counting.

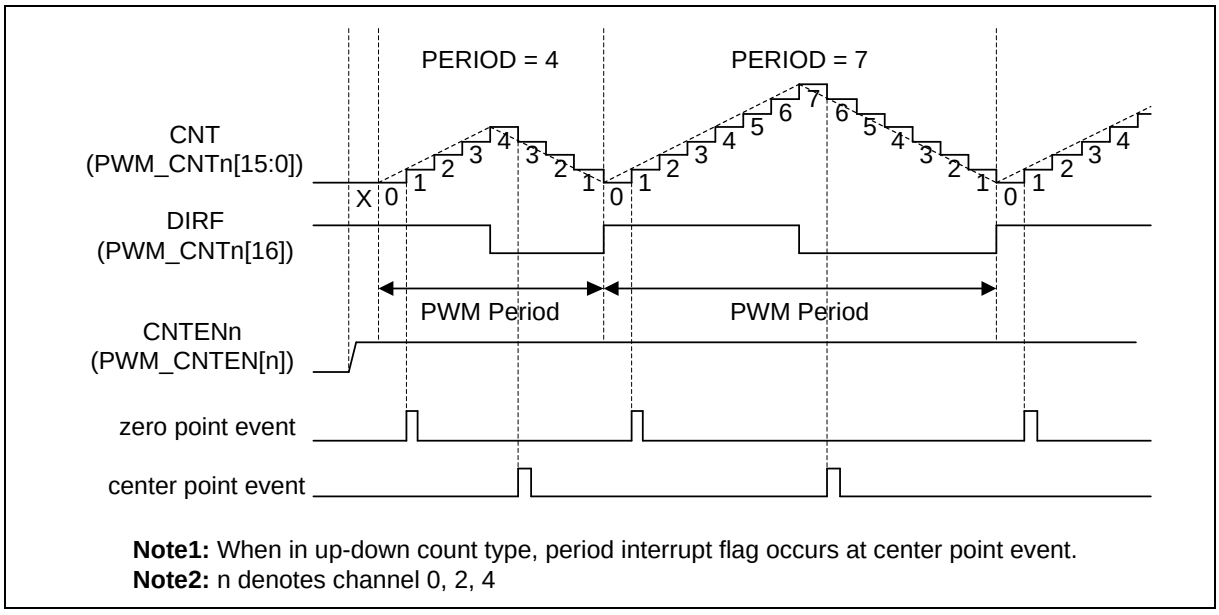
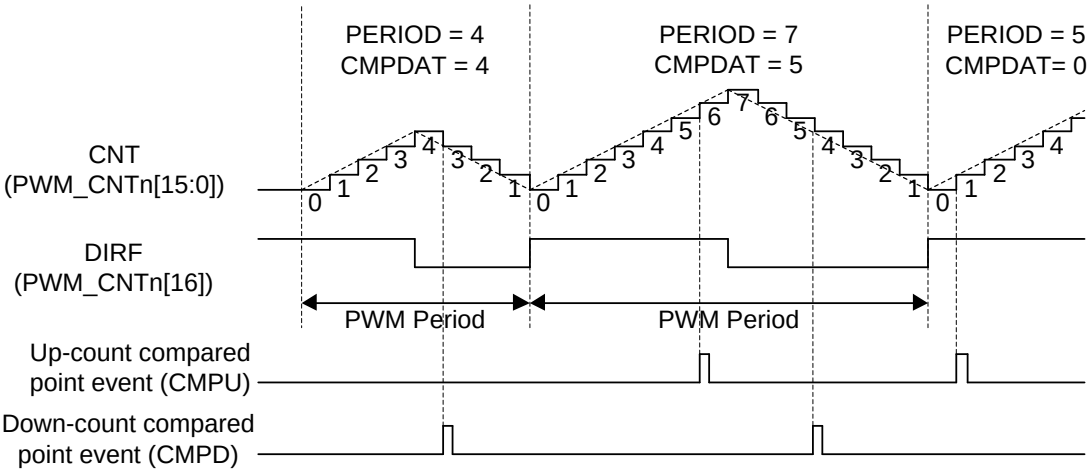


Figure 6.14-10 PWM Up-Down Counter Type

6.14.5.6 PWM Comparator

CMPDATn is a basic comparator register of PWM channel n; In Independent mode each channel only has one comparator, the value of CMPDATn register is continuously compared to the corresponding channel's counter value. In Complementary mode each paired channels has two comparators, and the value of CMPDATn and CMPDATm (n = 0,2,4, m = 1,3,5) registers are continuously compared to the complementary even channel's counter value, because of odd channel's counter is useless. For example, channel 0 and channel 1 are complementary channels, in Complementary mode, channel 1's comparator is continuously compared to channel 0's counter, but not channel 1's. When the counter is equal to value of CMPDAT0 register, PWM generates a compared point event and uses the event to generate PWM pulse, interrupt or use to trigger EADC/LPADC. In up-down counter type, two events will be generated in a PWM period as shown in Figure 6.14-11. The CMPU is up count compared point event and CMPD is down count compared point event.



Note1: No CMPU event occurred when CMPDAT equals to PERIOD.
Note2: n denotes channel 0, 2, 4

Figure 6.14-11 PWM Compared point Events in Up-Down Counter Type

6.14.5.7 PWM Double Buffering

The double buffering uses double buffers to separate software writing and hardware action operation timing. There are three loading modes for loading values to buffer: period loading mode, immediately loading mode, and center loading mode. After registers are modified through software, hardware will load register value to the buffer register according to the loading mode timing. The hardware action is based on the buffer value. This can prevent asynchronously operation problem due to software and hardware asynchronism.

The PWM provides PBUF (PWM_PBUFn[15:0], n = 0, 2, 4) as the active PERIOD buffer register, CMPBUF (PWM_CMPBUFn[15:0], n = 0,1..5) as the active CMPDAT buffer register. The concept of double buffering is used in loading modes, which are described in the following sections. For example, as shown in Figure 6.14-12, in period loading mode, writing PERIOD and CMPDAT through software, PWM will load new values to their buffer PBUF (PWM_PBUFn[15:0], n = 0, 2, 4) and CMPBUF (PWM_CMPBUFn[15:0], n = 0,1..5) at start of the next period without affecting the current period counter operation.

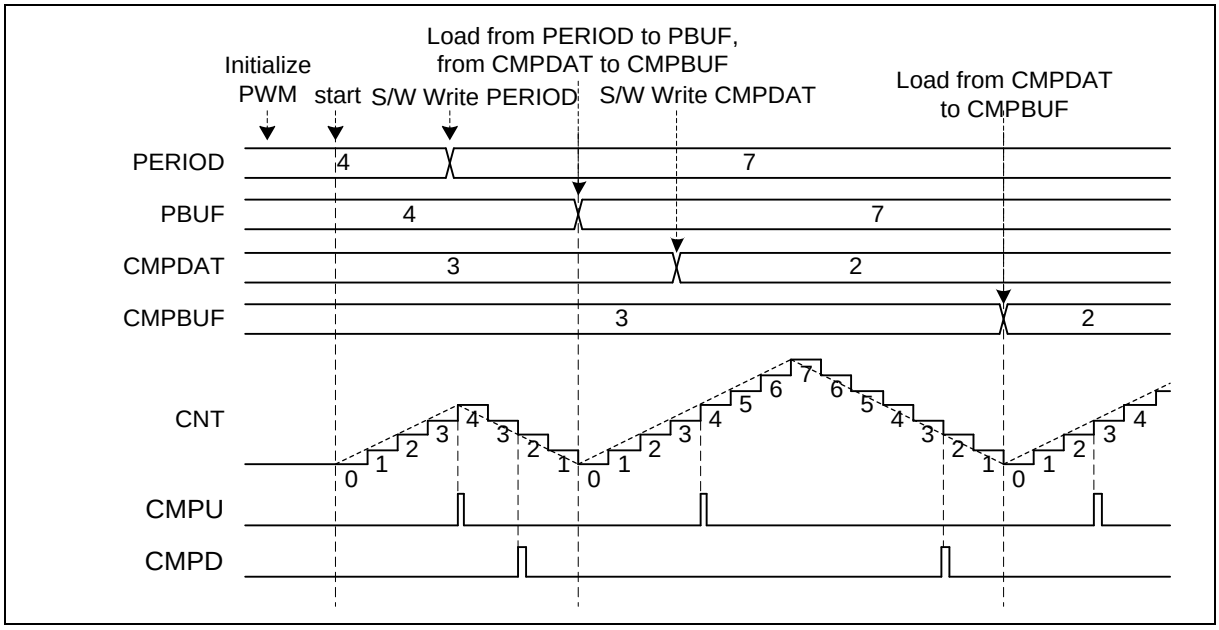


Figure 6.14-12 PWM Double Buffering Illustration

6.14.5.8 Period Loading Mode

When immediately loading mode and center loading mode are disabled that IMMLDENn (PWM_CTL0[16+n], n = 0,1..5) and CTRLDn (PWM_CTL0[n], n = 0,1..5) are set to 0, PWM operates at period Loading mode. In period Loading mode, PERIOD(PWM_PERIODn[15:0], n = 0, 2, 4) and CMP(PWM_CMPDATn[15:0], n = 0,1..5) will all load to their active PBUF and CMPBUF registers while each period is completed. For example, after PWM counter up counts from zero to PERIOD in the up-counter operation or down counts from PERIOD to zero in the down-counter operation or counts up from 0 to PERIOD and then counts down to 0 in the up-down counter operation.

Figure 6.14-13 shows period loading timing of up-count operation, where PERIOD DATA0 denotes the initial data of PERIOD, PERIOD DATA1 denotes the first updated PERIOD data by software and so on. CMPDAT also follows this rule. The following describes steps sequence of Figure 6.14-13. User can know the PERIOD and CMPDAT update condition, by watching PWM period and CMPU event.

1. Software writes CMPDAT DATA1 to CMPDAT at point 1.
2. Hardware loads CMPDAT DATA1 to CMPBUF at the end of PWM period at point 2.
3. Software writes PERIOD DATA1 to PERIOD at point 3.
4. Hardware loads PERIOD DATA1 to PBUF at the end of PWM period at point 4.
5. Software writes PERIOD DATA2 to PERIOD at point 5.
6. Hardware loads PERIOD DATA2 to PBUF at the end of PWM period at point 6.

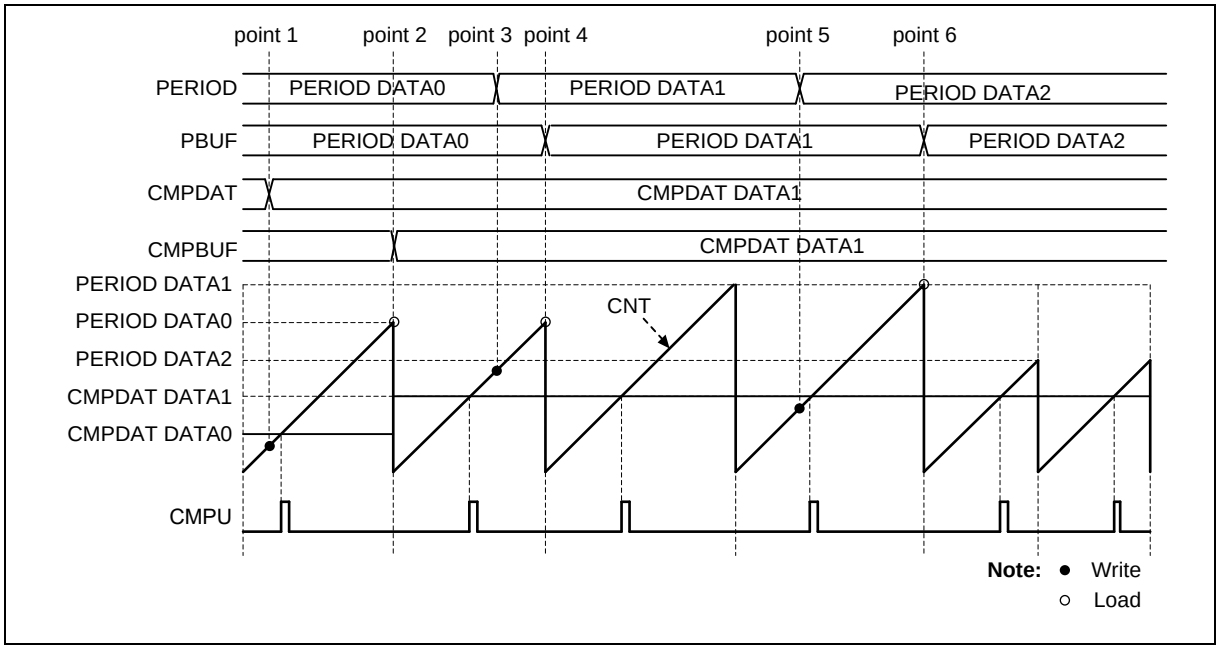


Figure 6.14-13 Period Loading in Up-Count Mode

6.14.5.9 Immediately Loading Mode

If the IMMLDENn (PWM_CTL0[16+n], n = 0,1..5) bit is set to 1, PWM operates at immediately loading mode. In immediately loading mode, when user update CMP (PWM_CMPDATn[15:0], n = 0,1..5), CMPDAT will be load to active CMPBUF (PWM_CMPBUFn[15:0], n = 0,1..5) after current counter count is completed. If the IMMLDENn (PWM_CTL0[16+n], n = 0, 2, 4) bit is set to 1, when user update PERIOD (PWM_PERIODn[15:0], n = 0, 2, 4), PERIOD will be load to active PBUF (PWM_PBUFn[15:0], n = 0, 2, 4) after current counter count is completed. If the updated PERIOD value is less than current counter value, counter will count to 0Xffff, when counter count to 0Xffff and prescale count to 0, the flag CNTMAXF(PWMx_STATUS[n], n = 0, 2, 4) will raise, and then counter will count wraparound. Immediately loading mode has the highest priority. If IMMLDENn has been set, other loading mode for channel n will become invalid. Figure 6.14-14 shows an example and its steps sequence is described below.

1. Software writes CMPDAT DATA1 and hardware immediately loading CMPDAT DATA1 to CMPBUF at point 1.
2. Software writes PERIOD DATA1 which is greater than current counter value at point 2; counter will continue counting until equal to PERIOD DATA1 to finish a period loading.
3. Software writes PERIOD DATA2 which is less than the current counter value at point 3; counter will continue counting to its maximum value 0xFFFF and count wraparound from 0 to PERIOD DATA2 to finish this period loading.

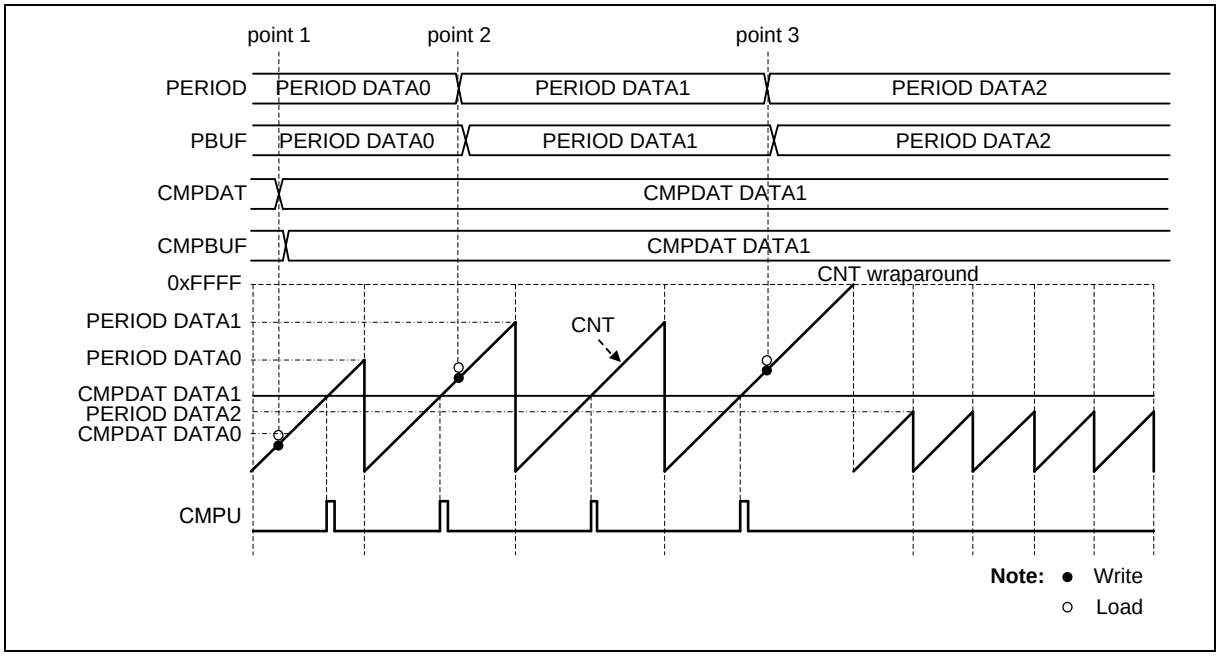


Figure 6.14-14 Immediately Loading in Up-Count Mode

6.14.5.10 Center Loading Mode

When the CTRLDn (PWM_CTL0[n], n = 0,1..5) bit is set to 1 and PWM counter is set to up-down count type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0, 2, 4) is 0x2, PWM operates at center loading mode. In center loading mode, CMP (PWM_CMPDATn[15:0], n = 0,1..5) will load to active CMPBUF register in center of each period, that is, counter counts to PERIOD (PWM_PERIODn[15:0], n = 0, 2, 4). PERIOD will all load to their active PBUF registers while each period is completed. Figure 6.14-15 shows an example and its steps sequence is described below.

1. Software writes CMPDAT DATA1 at point 1.
2. Hardware loads CMPDAT DATA1 to CMPBUF at center of PWM period at point 2.
3. Software writes PERIOD DATA1 at point 3.
4. Hardware loads PERIOD DATA1 to PBUF at the end of PWM period at point 4.
5. Software writes CMPDAT DATA2 at point 5.
6. Hardware loads CMPDAT DATA2 to CMPBUF at center of PWM period at point 6.
7. Software writes PERIOD DATA2 at point 7.
8. Hardware loads PERIOD DATA2 to PBUF at the end of PWM period at point 8.

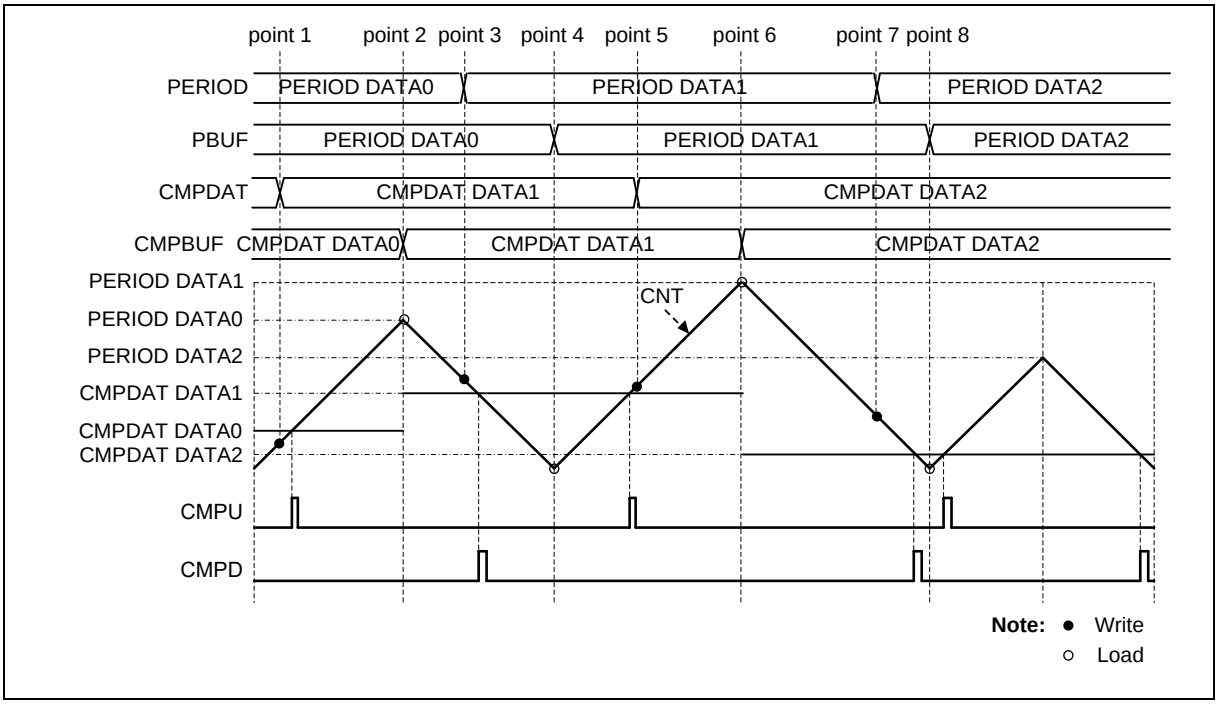


Figure 6.14-15 Center Loading in Up-Down-Count Mode

6.14.5.11 PWM Counter Operation Mode

The PWM counter supports Auto-reload mode.

In Auto-reload mode, CMPDAT and PERIOD registers should be written first and then set CNTENn (PWM_CNTEN[n], n = 0, 2, 4) bit to 1 to enable PWM prescaler and start to run counter. The value of CLKPSC (PWM_CLKPSCn_m[11:0], n = 0, 2, 4, m = 1, 3, 5), PERIOD (PWM_PERIODn[15:0], n = 0, 2, 4). PERIOD and CMP (PWM_CMPDATn[15:0], n = 0,1..5) will auto reload to their active buffer according different loading mode. If PERIOD is set to 0, PWM counter will be set to 0.

6.14.5.12 PWM Pulse Generator

The PWM pulse generator uses counter and comparator events to generate PWM pulse. The events are: zero point, period point in up counter type and down counter type, center point in up-down counter type and counter equal to comparator point in three types. As to up-down counter type, there are two counter equal comparator points, one at up count and the other at down count. Besides, Complementary mode has two comparators compared with counter, and thus comparing equal points will become four in up-down counter type and two for up or down counter type.

Each event point can decide PWM waveform to do nothing (X), set Low (L), set High (H) or toggle (T) by setting the PWM_WGCTL0 and PWM_WGCTL1 registers. Using these points can easily generate asymmetric PWM pulse or variant waveform as shown in Figure 6.14-16. In the figure, PWM is in complementary mode, there are two comparators n and m to generate PWM pulse. n denotes even channel number 0, 2, or 4, and m denotes odd channel number 1, 3, or 5. n channel and m channel are complementary paired. Complementary mode uses two channels (CH0 and CH1, CH2 and CH3, or CH4 and CH5) as a pair of PWM outputs to generate complement paired waveforms. CMPU denotes CNT (PWM_CNTn[15:0], n = 0, 2, 4) is equal to CMP (PWM_CMPDATn[15:0], n = 0,1..5) when counting up. CMPD denotes CNT bits is equal to CMP bits when counting down.

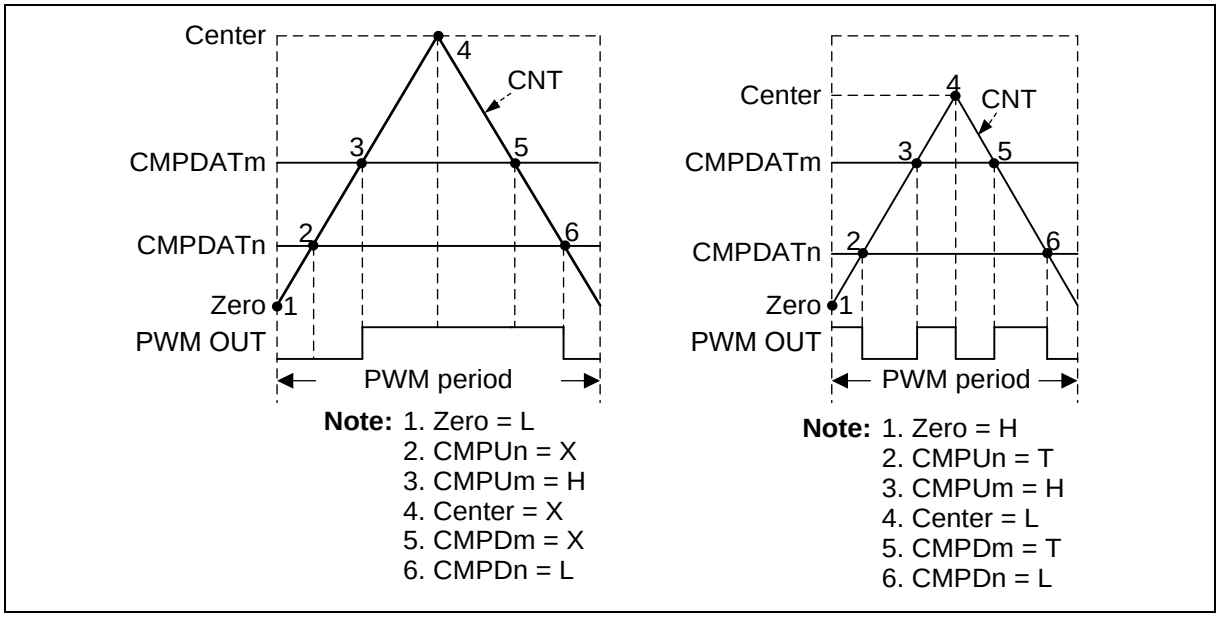


Figure 6.14-16 PWM Pulse Generation

The generation events may sometimes set to the same value, as the reason, events priority between different counter types are list below, up counter type (Table 6.14-2), down counter type (Table 6.14-3) and up-down counter type (Table 6.14-4). By using event priority, user can easily generate 0% to 100% duty pulse as shown in Figure 6.14-17.

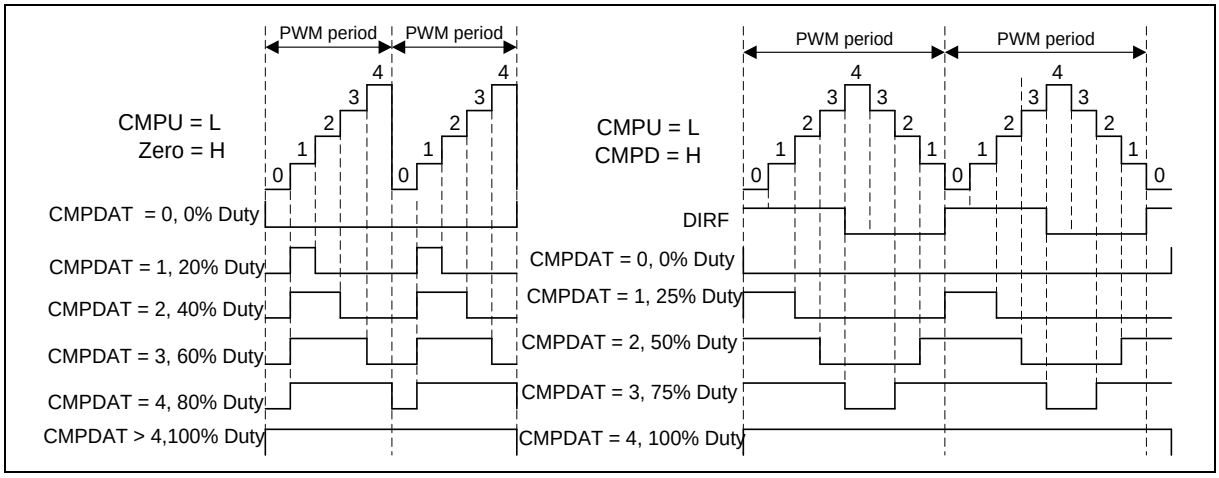


Figure 6.14-17 PWM 0% to 100% Pulse Generation

Priority	Up Event
1 (Highest)	Period event (CNT = PERIOD)
2	Compare up event of odd channel (CNT = CMPUm)
3	Compare up event of even channel (CNT = CMPUn)
4 (Lowest)	Zero event (CNT = 0)

Table 6.14-2 PWM Pulse Generation Event Priority for Up-Counter

Priority	Down Event
1 (Highest)	Zero event (CNT = 0)
2	Compare down event of odd channel (CNT = CMPDm)
3	Compare down event of even channel (CNT = CMPDn)
4 (Lowest)	Period event (CNT = PERIOD)

Table 6.14-3 PWM Pulse Generation Event Priority for Down-Counter

Priority	Up Event	Down Event
1 (Highest)	Compare up event of odd channel (CNT = CMPUm)	Compare down event of odd channel (CNT = CMPDm)
2	Compare up event of even channel (CNT = CMPUn)	Compare down event of even channel (CNT = CMPDn)
3 (Lowest)	Zero event (CNT = 0)	Period (center) event (CNT = PERIOD)

Table 6.14-4 PWM Pulse Generation Event Priority for Up-Down-Counter

6.14.5.13 PWM Output Mode

The PWM supports two output modes: Independent mode which may be applied to DC motor system, Complementary mode with dead-time insertion which may be used in the application of AC induction motor and permanent magnet synchronous motor.

6.14.5.14 Independent mode

By default, the PWM is operating in independent mode. Independent mode is enabled when pair channel's corresponding OUTMODEn (PWM_CTL1[26+n/2], n = 0, 2, 4) bit is set to 0. In this mode, three pair channels are running off their own period and six output channels are running off its own duty as shown in Figure 6.14-18

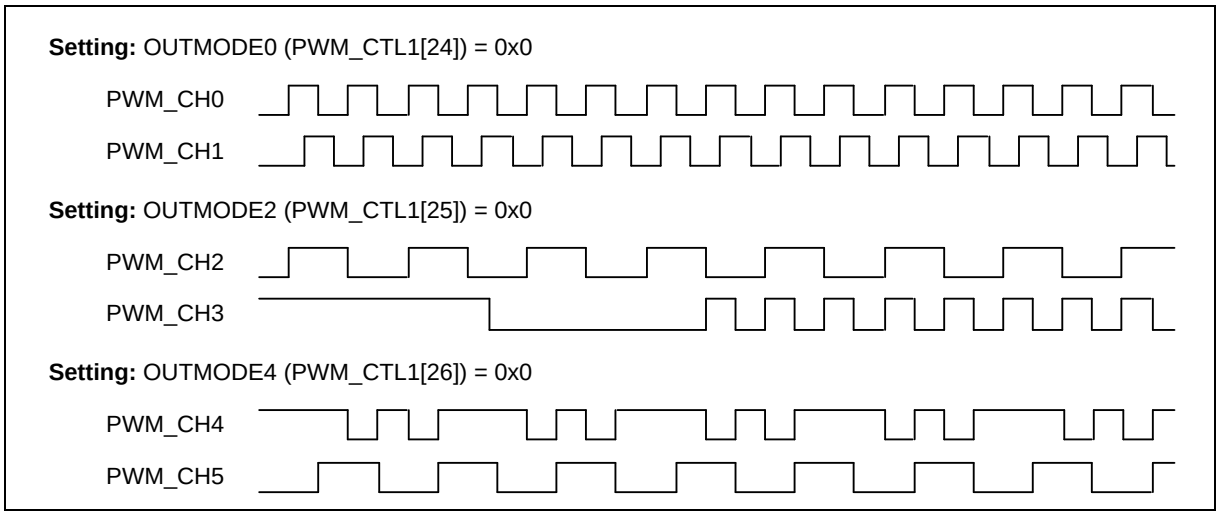


Figure 6.14-18 PWM Independent Mode Waveform

6.14.5.15 Complementary mode

Complementary mode is enabled when pair channel's corresponding OUTMODEn (PWM_CTL1[26+n/2], n = 0, 2, 4) bit set to 1. In this mode there are 3 PWM generators utilized for

complementary mode, with total of 3 PWM output paired pins in this module. In Complimentary modes, the internal odd PWM signal must always be the complement of the corresponding even PWM signal. PWM_CH1 will be the complement of PWM_CH0. PWM_CH3 will be the complement of PWM_CH2 and PWM_CH5 will be the complement of PWM_CH4 as shown in Figure 6.14-19.

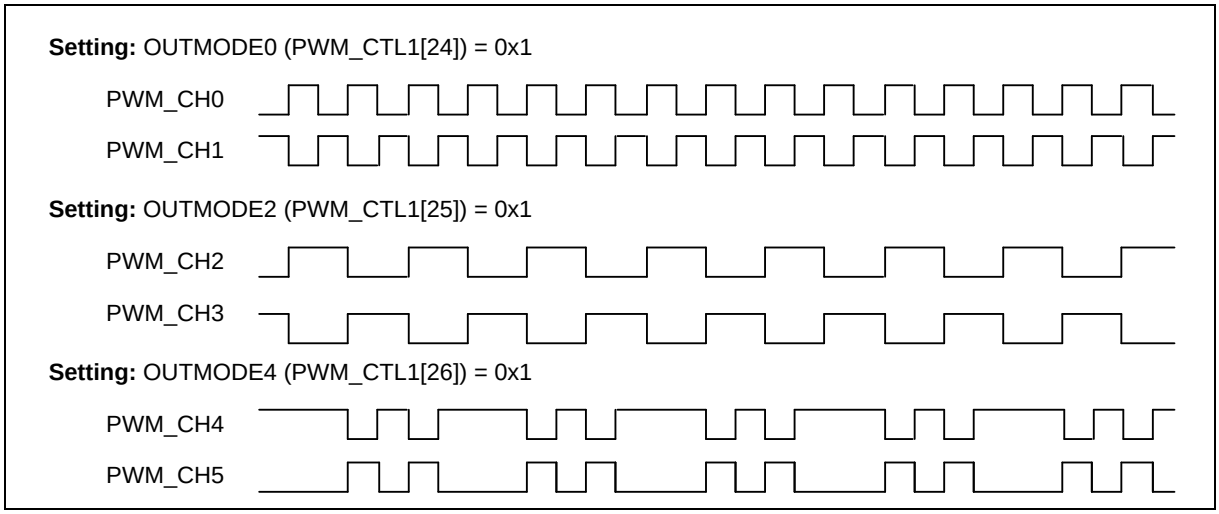


Figure 6.14-19 PWM Complementary Mode Waveform

6.14.5.16 PWM Output Control

After PWM pulse generation, there are four to six steps to control the output of PWM channels. In independent mode, there are Mask, Brake, Pin Polarity and Output Enable four steps as shown in Figure 6.14-20. In complementary mode, it needs two more steps to precede these four steps, Complementary channels and Dead-Time Insertion as shown in Figure 6.14-21.

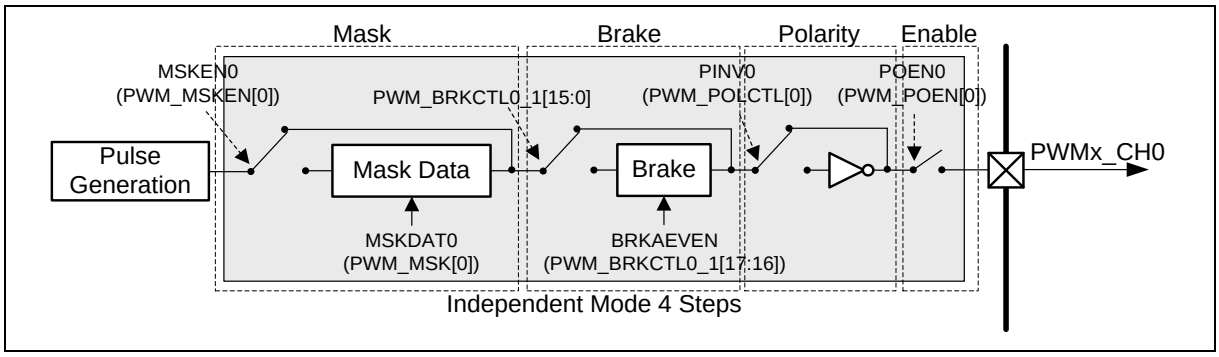


Figure 6.14-20 PWMx_CH0 Output Control in Independent Mode

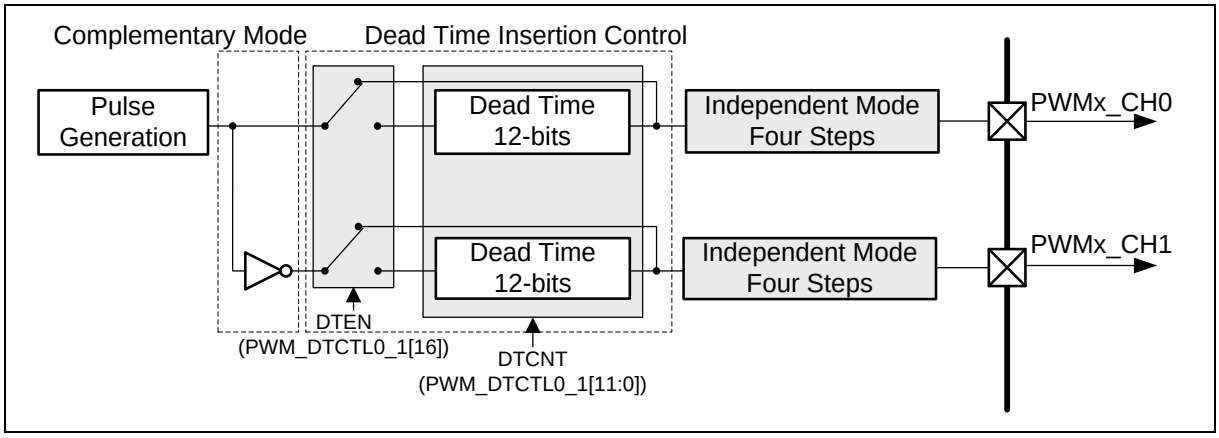


Figure 6.14-21 PWMx_CH0 and PWMx_CH1 Output Control in Complementary Mode

6.14.5.17 Dead-Time Insertion

In the complementary application, the complement channels may drive the external devices like power switches. The dead-time generator inserts a low level period called “dead-time” between complementary outputs to drive these devices safely and to prevent system or devices from the burn-out damage. Hence the dead-time control is a crucial mechanism to the proper operation of the complementary system. By setting pair channel's corresponding DTEN (PWM_DTCTLn_m[16]) bit to enable dead-time function and DTCNT (PWM_DTCTLn_m[11:0]) to control dead-time period, the dead-time can be calculated from the following formula:

Dead-time = (DTCNT (PWM_DTCTLn[11:0])+1) * PWMx_CLK period

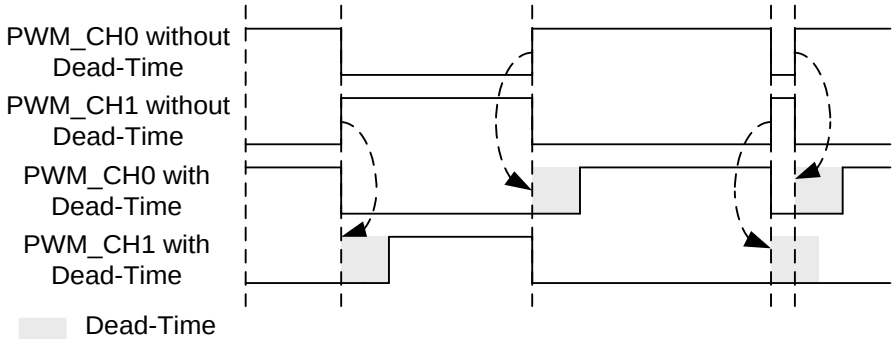
Dead-time insertion clock source can be selected from prescaler output by setting DTCKSEL (PWM_DTCTLn_m[24]) to 1. By default, clock source comes from PWM_CLK, which is prescaler input. Then the dead-time can be calculated from the following formula:

Dead-time = (DTCNT (PWM_DTCTLn[11:0])+1) *

(CLKPSC (PWM_CLKPSCn [11:0])+1)*PWMx_CLK period

Please note that the PWM_DTCTLn_m are write-protected registers.

Figure 6.14-22 indicates the dead-time insertion for one pair of PWM signals.



Effect of Dead-Time for complementary pairs

Figure 6.14-22 Dead-Time Insertion

6.14.5.18 PWM Mask Output Function

Each of the PWM channel output value can be manually overridden with the settings in PWM Mask Enable Register (PWM_MSKEN) and PWM Mask Data Register (PWM_MSK) With these settings, the PWM channel outputs can be assigned to specified logic states independent of the duty cycle comparison units. The PWM mask bits are useful when controlling various types of Electrically Commutated Motor (ECM) like a BLDC motor. The PWM_MSKEN register contains six bits, MSKENn(PWM_MSKEN[5:0]). If the MASKENn is set to active-high, the PWM channel n output will be overridden. The PWM_MSK register contains six bits, MSKDATn(PWM_MSK[5:0]). The bit value of the MSKDATn determines the state value of the PWM channel n output when the channel is overridden. Figure 6.14-23 shows an example of how PWM mask control can be used for the override feature.

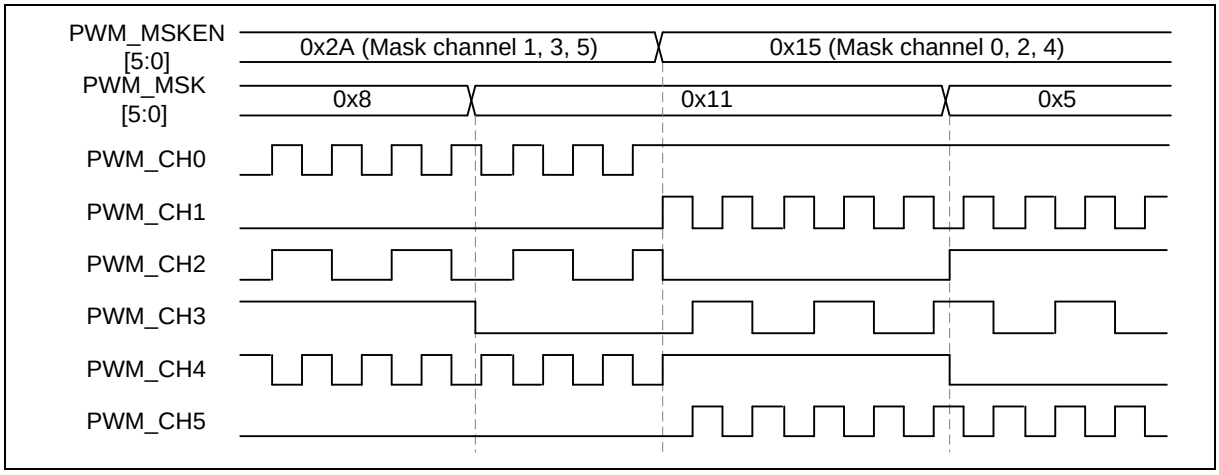


Figure 6.14-23 Illustration of Mask Control Waveform

6.14.5.19 PWM Brake

Each PWM module has two external input brake control signals. User can set BKySRC (PWM_BNF[16+8*y], y = 0, 1) to select the source of brake y is from PWMx_BRAKEy (x = 0, 1). The external signals will be filtered by a 3-bit noise filter. User can enable the noise filter function by BRKyNFEN (PWM_BNF[8*y], y = 0, 1), and noise filter sampling clock can be selected by setting BRKyNFSEL (PWM_BNF[3+8*y:1+8*y], y = 0, 1) to fit different noise properties. Moreover, by setting BRKyFCNT (PWM_BNF[6+8*y:4+8*y], y = 0, 1) bits, user can define by how many sampling clock cycles a filter will recognize the effective edge of the brake signal.

In addition, it can be inversed by setting BRKyPINV (PWM_BNF(7+8*y), y = 0 , 1) to realize the polarity setup for the brake control signals. Set BRKyPINV bit to 0, brake event will occurred when PWMx_BRAKEy pin status is from low to high; set BRKyPINV to 1, brake event will occurred when PWMx_BRAKEy pin status is from high to low.

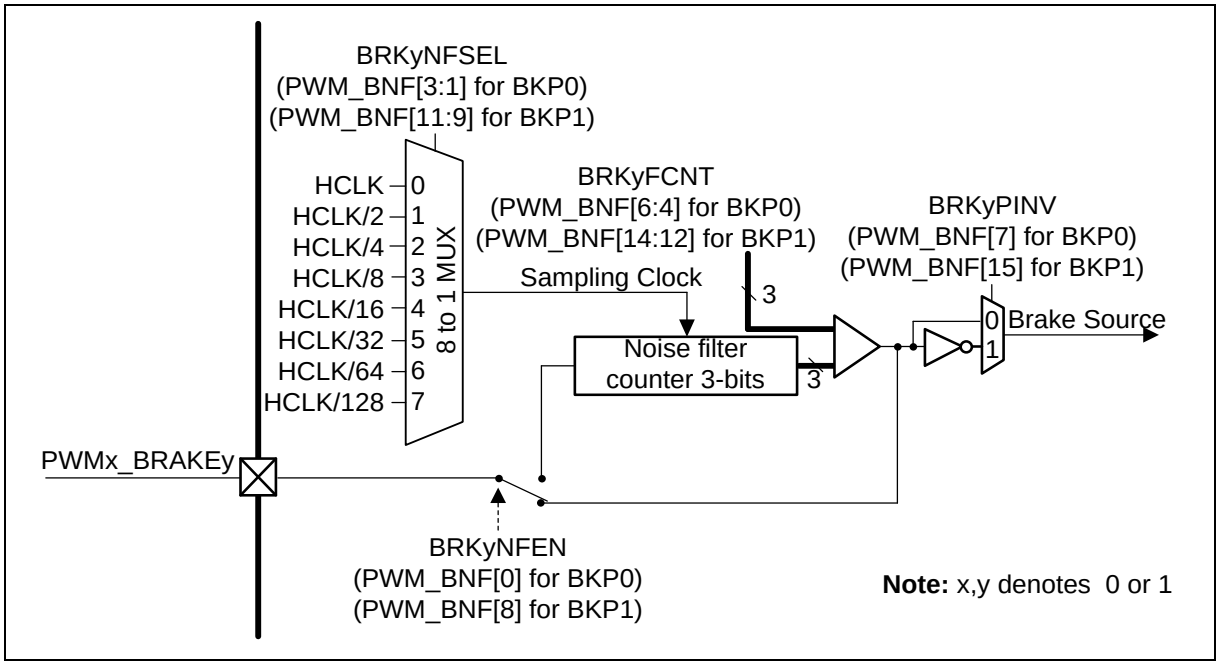


Figure 6.14-24 Brake Noise Filter Block Diagram

For Complementatry mode, it is often necessary to set a safe output state to the complement output pairs once the brake event occurs.

Each complementary channel pair shares a PWM brake function, as shown in Figure 6.14-25. To control paired channels to output safety state, user can setup BRKAEVEN (PWM_BRKCTL0_1[17:16]) for even channels and BRKAODD (PWM_BRKCTL0_1[19:18]) for odd channels when the fault brake event happens. There are two brake detectors: Edge detector and Level detector. When the edge detector detects the brake signal and BRKEIENn_m (PWM_INTEN1[2:0]) is enabled, the brake function generates BRK_INT. This interrupt needs software to clear, and the BRKESTSn (PWM_INTSTS1[21:16]) brake state will keep until the next PWM period starts after the interrupt cleared. The brake function can also operate in another way through the level detector. Once the level detector detects the brake signal and the BRKLIENn_m (PWM_INTEN1[10:8]) is also enabled, the brake function will generate BRK_INT, but BRKLSTSn (PWM_INTSTS1[29:24]) brake state will auto recovery to normal output while level brake source recovery to high level and pass through “Low Level Detection” at the PWM waveform period when brake condition removed without clear interrupt.

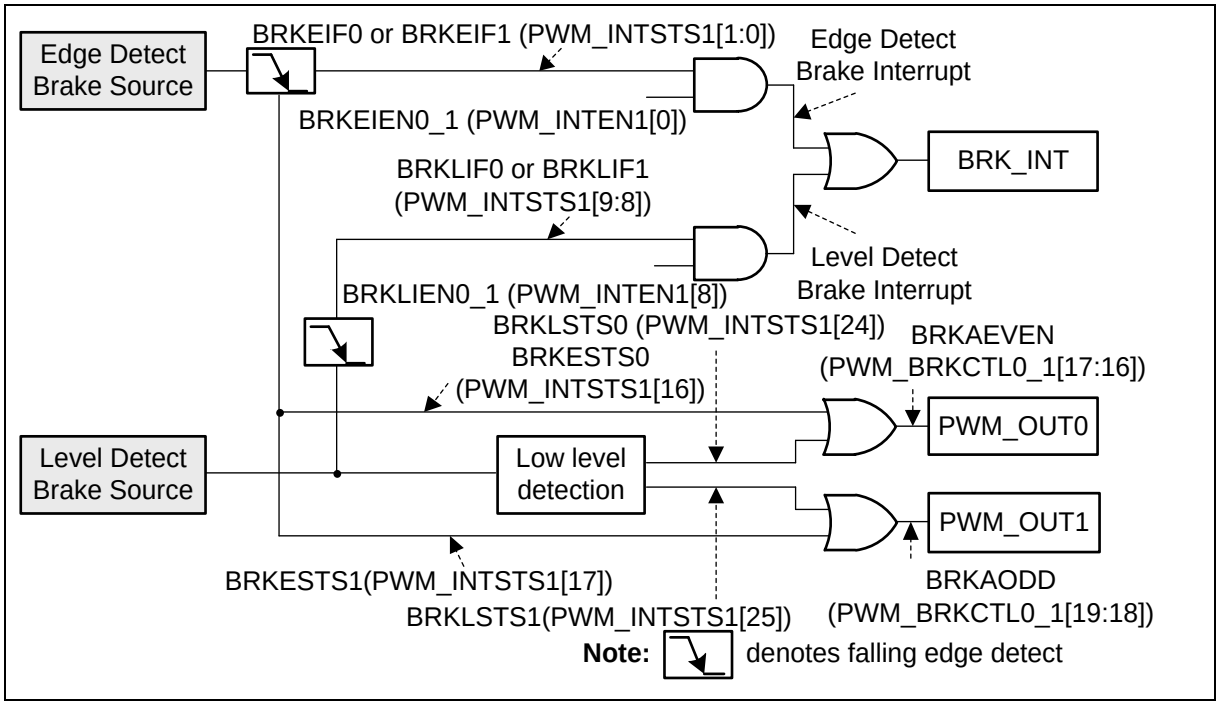


Figure 6.14-25 Brake Block Diagram for PWMx_CH0 and PWMx_CH1 Pair

Figure 6.14-26 illustrates the edge detector waveform for PWMx_CH0 and PWMx_CH1 pair. In this case, the edge detect brake source has occurred twice for the brake events. When the event occurs, both of the BRKEIF0 and BRKEIF1 flags are set and BRKESTS0 and BRKESTS1 bits are also set to indicate brake state of PWMx_CH0 and PWMx_CH1. For the first occurring event, software writes 1 to clear the BRKEIF0 flag. After that, the BRKESTS0 bit is cleared by hardware at the next start of the PWM period. At the same moment, the PWMx_CH0 outputs the normal waveform even though the brake event is still occurring. The second event also triggers the same flags, but at this time, software writes 1 to clear the BRKEIF1 flag. Afterward, PWMx_CH1 outputs normally at the next start of the PWM period.

As a contrast to the edge detector example, Figure 6.14-27 illustrates the level detector waveform for PWMx_CH0 and PWMx_CH1 pair. In this case, the BRKLIF0 and BRKLIF1 flags can only indicate the brake event having occurred. The BRKLSTS0 and BRKLSTS1 brake states will automatically recover at the start of the next PWM period no matter at what states the BRKLIF0 and BRKLIF1 flags are at that moment.

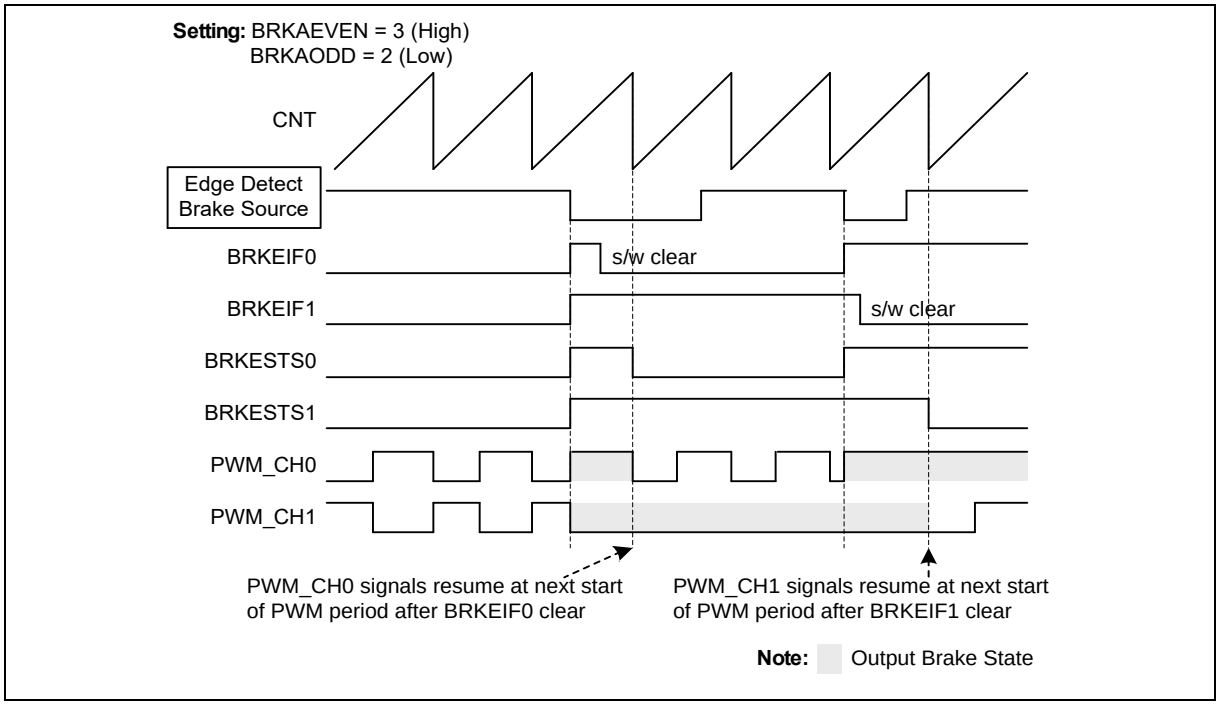


Figure 6.14-26 Edge Detector Waveform for PWMx_CH0 and PWMx_CH1 Pair

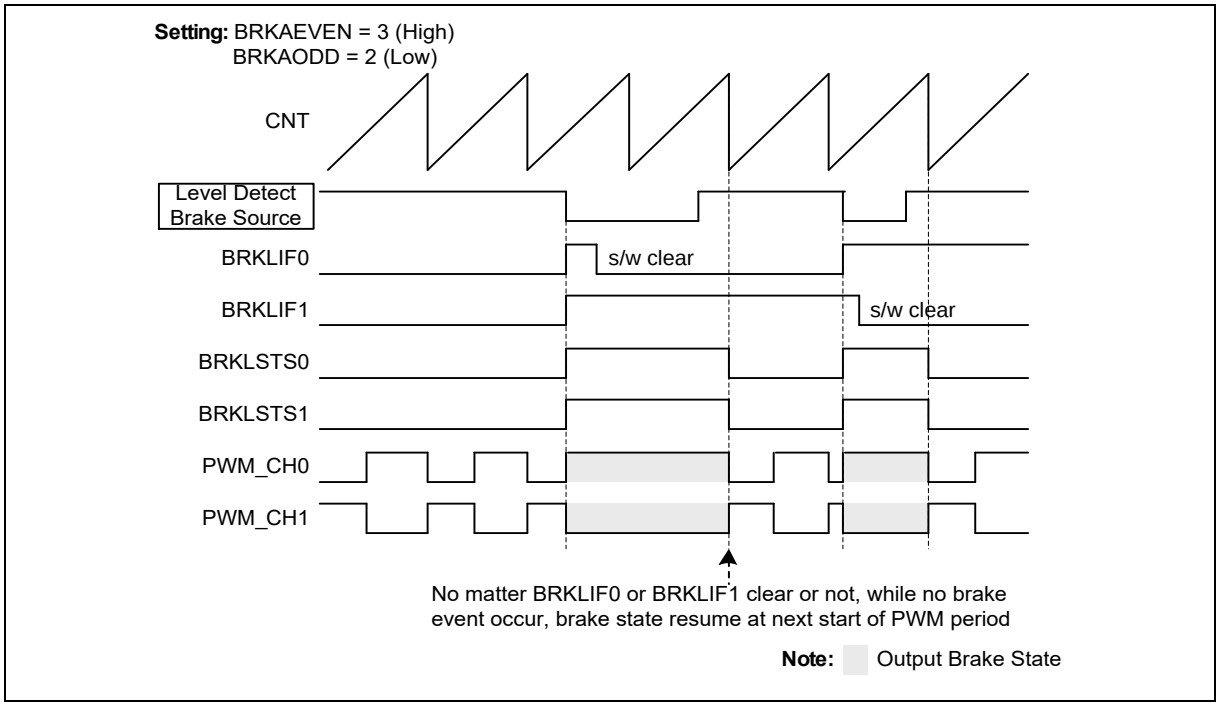


Figure 6.14-27 Level Detector Waveform for PWMx_CH0 and PWMx_CH1 Pair

The two kinds of detectors detect the same ten brake sources: two from external input signals, three from analog comparators(ACMP), one from EADC result monitor (EADCRM), one from system fail, one from software triggered, one from UTCPD0_VBUS_Source, and one from UTCPD0_VBUS_Sink, that are shown in Figure 6.14-28. ACMP brake sources will be detected only when internal ACMP0_O or ACMP1_O signal from low to high.

Among the above described brake sources, the brake source coming from system fail can still be specified to several different system fail conditions. These conditions include clock fail, Brown-out detect, SRAM parity check error, and Core lockup. Figure 6.14-29 shows that by setting corresponding enable bits, the enabled system fail condition can be one of the sources to issue the Brake system fail to the PWM brake.

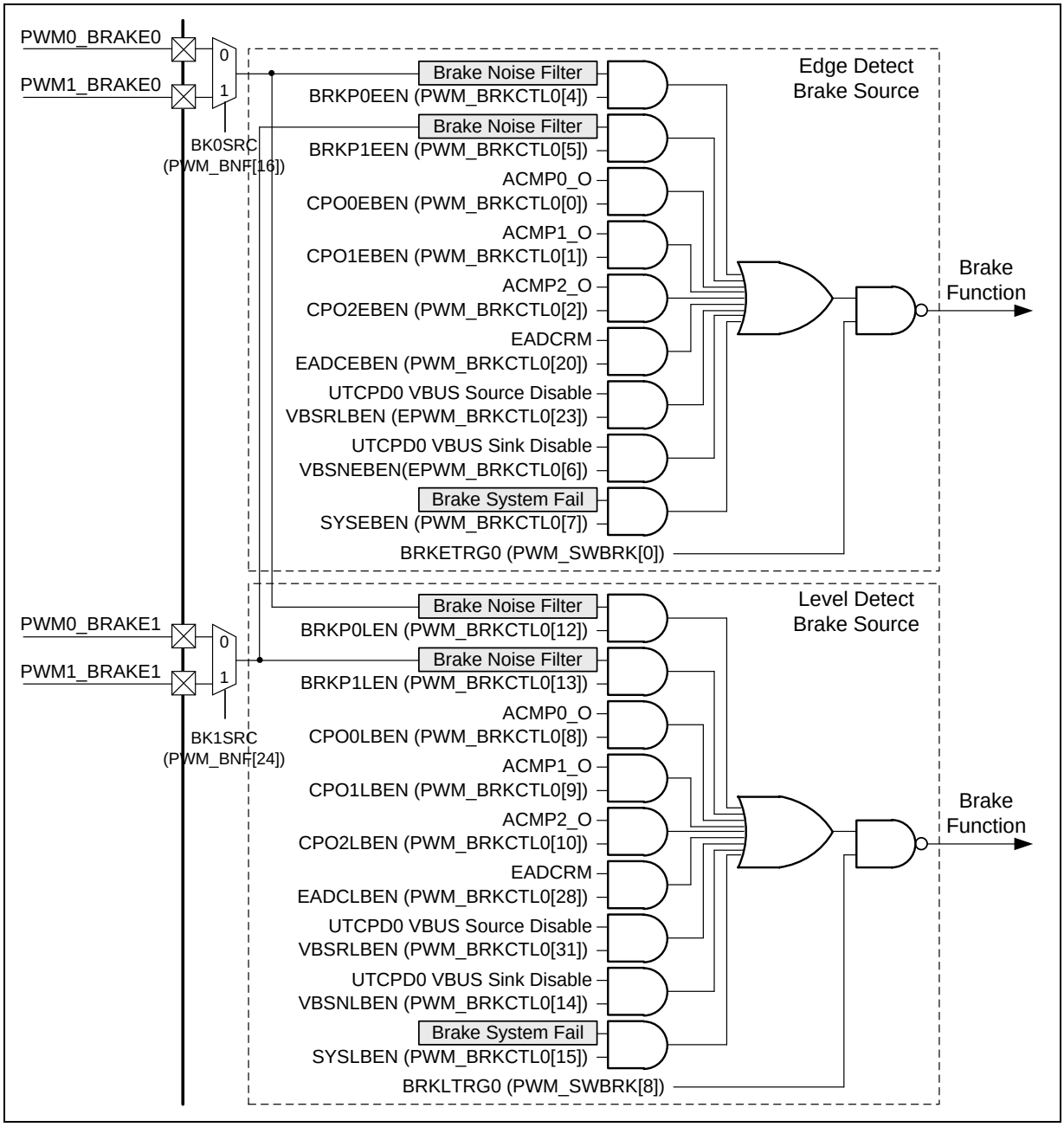


Figure 6.14-28 Brake Source Block Diagram

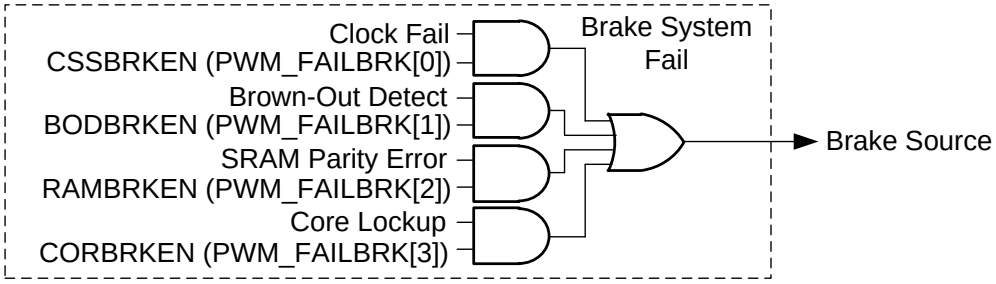
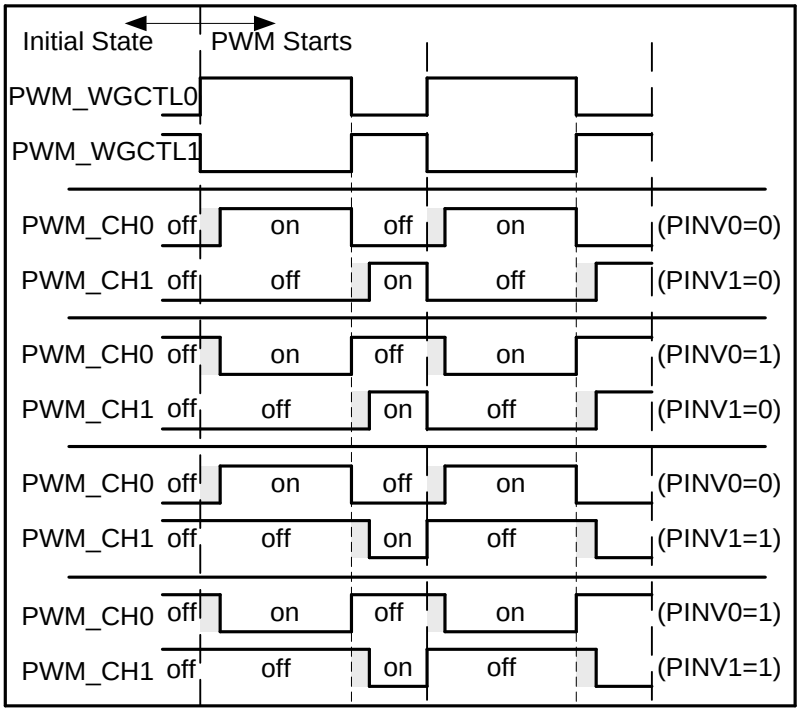


Figure 6.14-29 Brake System Fail Block Diagram

6.14.5.20 Polarity Control

Each PWM port, from PWM_CH0 to PWM_CH5, has an independent polarity control module to configure the polarity of the active state of the PWM output. By default, the PWM output is active high. This implies the PWM OFF state is low and ON state is high. This definition is variable through setting the PWM Pin Polar Inverse Control (PWM_POLCTL) register, for each individual PWM channel. Figure 6.14-30 shows the initial state before PWM starting with different polarity settings.



Dead-time insertion; It is only effective in complementary mode
Note: PINVx: Negative Polarity control bits; It controls the PWM output initial state and polarity, x denotes 0 or 1.

Figure 6.14-30 Initial State and Polarity Control with Rising Edge Dead-Time Insertion

6.14.5.21 Synchronous start function

The synchronous start function can be enabled when SSENn (PWM_SSCTL[n], n = 0, 2 ,4) is set. User can select synchronous source which is from EPWM0, EPWM1, PWM0 or PWM1 by SSRC

(PWM_SSCTL[9:8]). The selected PWM channels (include channel0 to channel5 of each PWM) will start counting at the same time once the synchronous start function is enabled and set CNTSEN (PWM_SSTRG[0]). It is noted that set CNTSEN bit will also set the counter enable bit (CNTENn, n = 0, 2, 4) to start counting. Because the source of synchronous start belongs to different PCLK clock domain, please make sure the APB0DIV (CLK_PCLKDIV[2:0]) and APB1DIV (CLK_PCLKDIV[6:4]) should be set the same divider value if the selected synchronous start source IP and destination IP are different APB clock domain.

6.14.5.22 PWM Interrupt Generator.

There are three independent interrupts for each PWM as shown in Figure 6.14-32.

The 1st PWM interrupt (PWM_INT) comes from PWM complementary pair events. The counter can generate the Zero point Interrupt Flag ZIFn (PWM_INTSTS0[n], n=0,2,4) and the Period point Interrupt Flag PIFn (PWM_INTSTS0[n+8], n=0,2,4). When PWM channel n's counter equals to the comparator value stored in PWM_CMPDATn, the different interrupt flags will be triggered depending on the counting direction. If the matching occurs at up-count direction, the Up Interrupt Flag CMPUIFn (PWM_INTSTS0[21:16]) is set and if matching at the opposite direction, the Down Interrupt Flag CMPDIFn (PWM_INTSTS0[29:24]) is set. If the corresponding interrupt enable bits are set, the trigger events will generates interrupt signals.

PWM_INT can use the PWM_IFAn (n=0,2,4) register to accumulate the number of times that the interrupt flags have been triggered for each channel. By setting one of IFAEN (PWM_IFAn[31], n=0,2,4) bit to 1 to enable accumulator, PWM_INT will switch interrupt source from every event trigger interrupt to trigger interrupt once every accumulate times.

By setting the IFASEL (PWM_IFAn[29:28], n=0,2,4) bits, user can select one of the 4 interrupt sources to accumulate interrupt flag times for each channel, and the number of times interrupt flags will compare with IFACNT (PWM_IFAn[7:0], n=0,2,4) bits. When interrupt accumulator equals IFACNT then set IFAIFn (PWM_AINTSTS[n], n=0,2,4) bits as PWM_INT signal if user enable IFAIENn (PWM_AINTEN[n], n=0,2,4) bits. Accumulator interrupt of each channel can also be as request source of PDMA. Figure 6.14-31 is an example of channel 0 using PWM_IFA0 register to output PWM_INT once every IFACNT0+1 times interrupt events occurred.

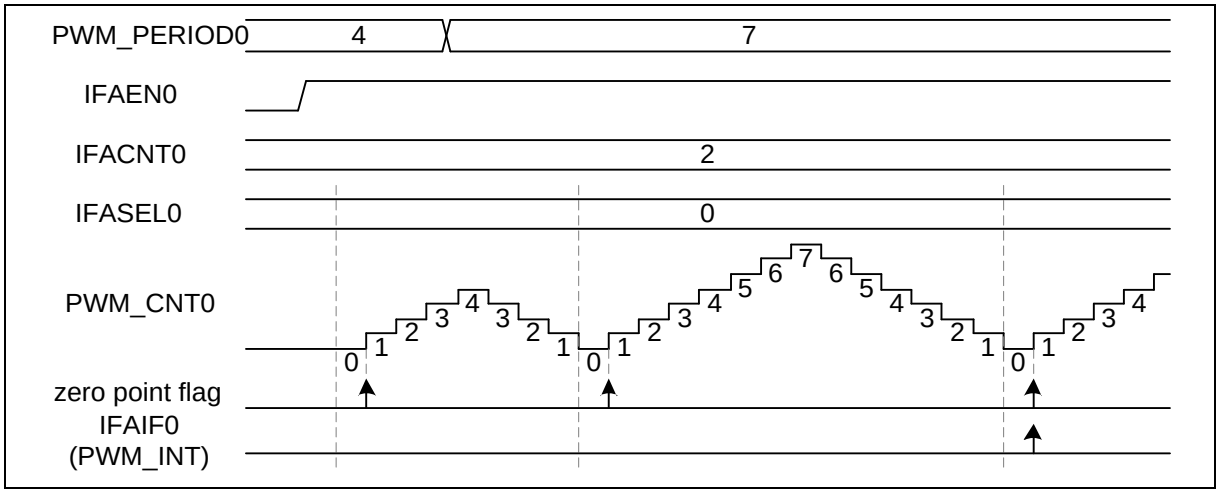


Figure 6.14-31 PWMx_CH0 Accumulate Interrupt Waveform

The 2nd interrupt is the capture interrupt (CAP_INT). It shares the PWM_INT vector in NVIC. The CAP_INT can be generated when the CRLIFn (PWM_CAPIF[5:0]) is triggered and the Capture Rising Interrupt Enable bit CAPRIENn (PWM_CAPIEN[5:0]) is set to 1. Or in the falling edge condition, the CAP_INT can be generated when the CFLIFn (PWM_CAPIF[13:8]) is triggered and the Capture Falling Interrupt Enable bit CAPFIENn (PWM_CAPIEN[13:8]) is set to 1.

The last one is the brake interrupt (BRK_INT). The detail of the BRK_INT is described in the PWM Brake

section.

Figure 6.14-32 demonstrates the architecture of the PWM interrupts.

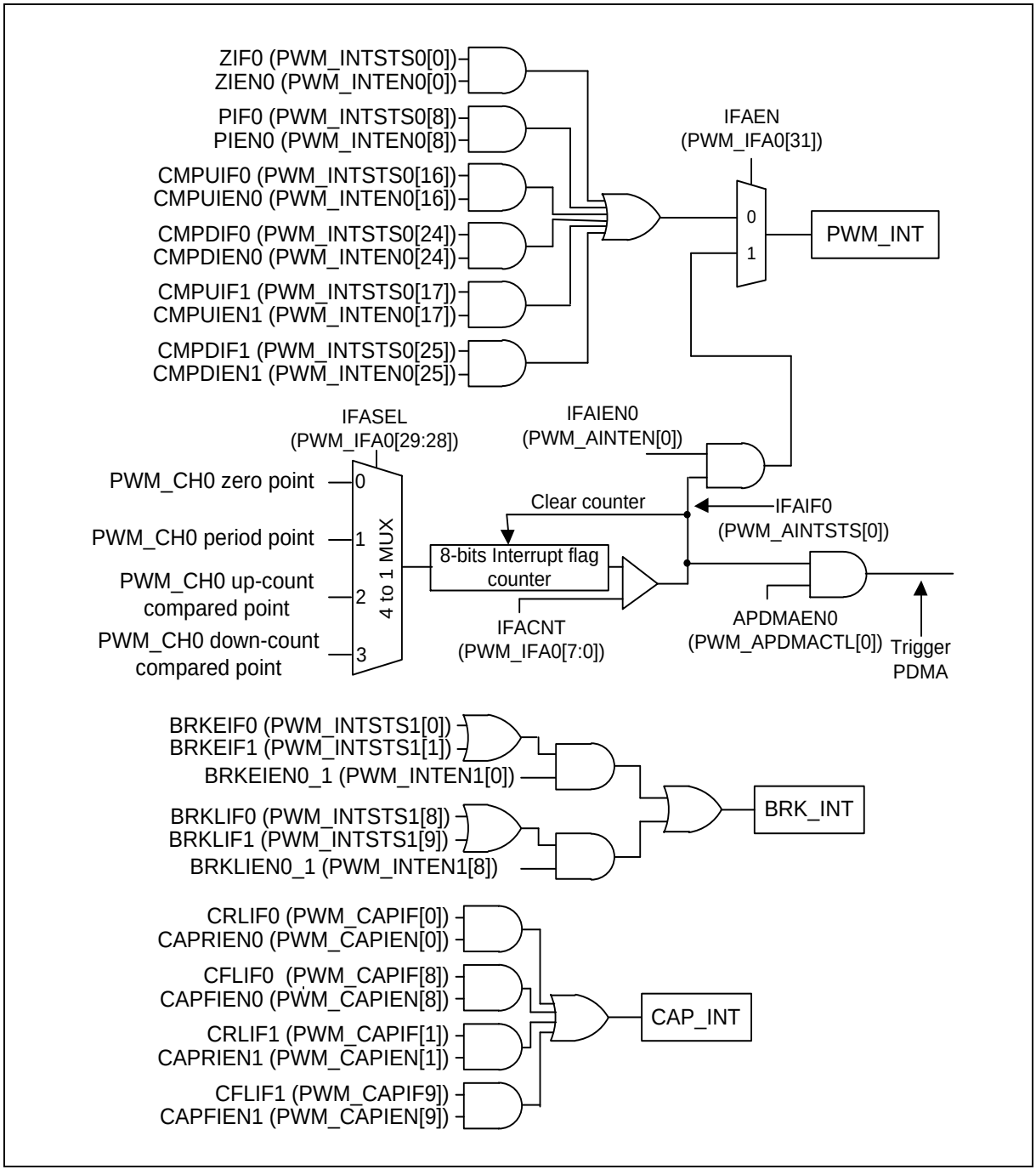


Figure 6.14-32 PWM_CH0 and PWM_CH1 Pair Interrupt Architecture Diagram

6.14.5.23 PWM Trigger EADC/LPADC Generator

PWM can be one of the EADC/LPADC conversion trigger source. Each PWM pair channels share the same trigger source. Setting TRGSELn is to select the trigger sources, where TRGSELn is TRGSEL0, TRGSEL1, …, and TRGSEL5, which are located in PWM_EADCTS0[3:0], PWM_EADCTS0[11:8],

PWM_EADCTS0[19:16], PWM_EADCTS0[27:24], PWM_EADCTS1[3:0] and PWM_EADCTS1[11:8], respectively. Setting TRGENn is to enable the trigger output to ADC, where TRGENn is TRGEN0, TRGEN1, …, TRGEN5, which are located in PWM_EADCTS0[7], PWM_EADCTS0[15], PWM_EADCTS0[23], PWM_EADCTS0[31], PWM_EADCTS1[7] and PWM_EADCTS1[15], respectively. The number n (n = 0,1, ..,5) denotes PWM channel number.

There are 7 PWM events can be selected as the trigger source for one pair of channels. Figure 6.14-33 is an example of PWM_CH0 and PWM_CH1. And any one of PWM trigger event can trigger LPADC if corrlated TRGEN is enabled. Figure 6.14-34 shows the PWM trigger LPADC source block diagram. PWM can trigger EADC/LPADC to start conversion in different timings by setting PERIOD and CMPDAT. Figure 6.14-35 is the trigger EADC/LPADC timing waveform in the up-down counter type.

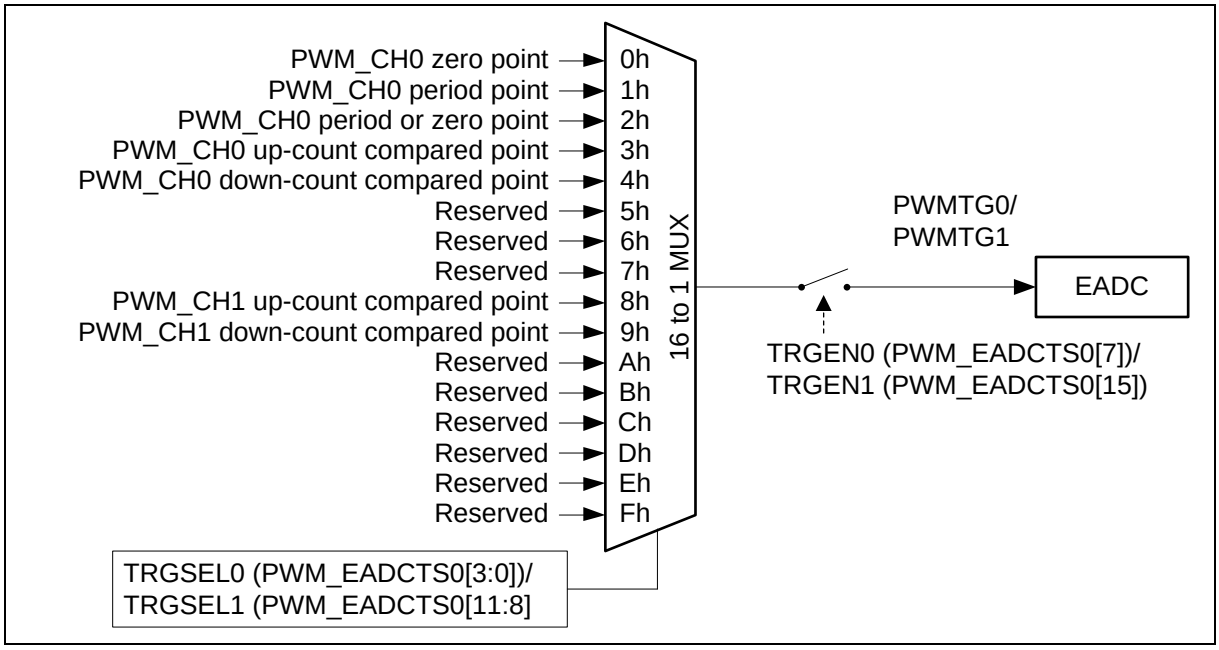


Figure 6.14-33 PWMx_CH0 and PWMx_CH1 Pair Trigger EADC Block Diagram

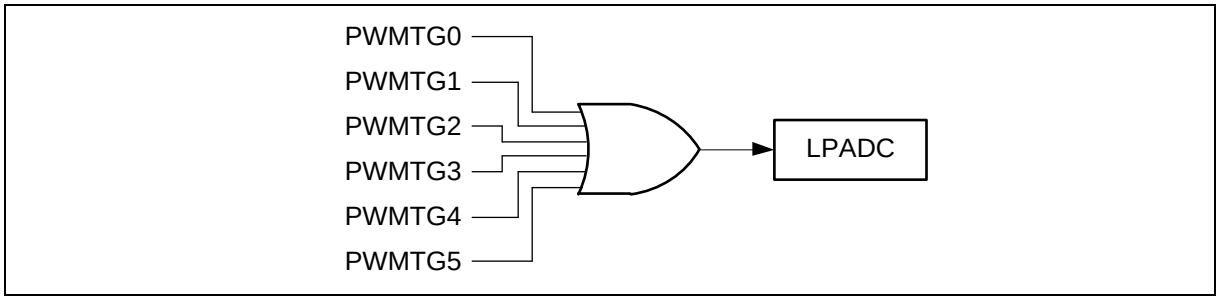


Figure 6.14-34 PWM Trigger LPADC Block Diagram

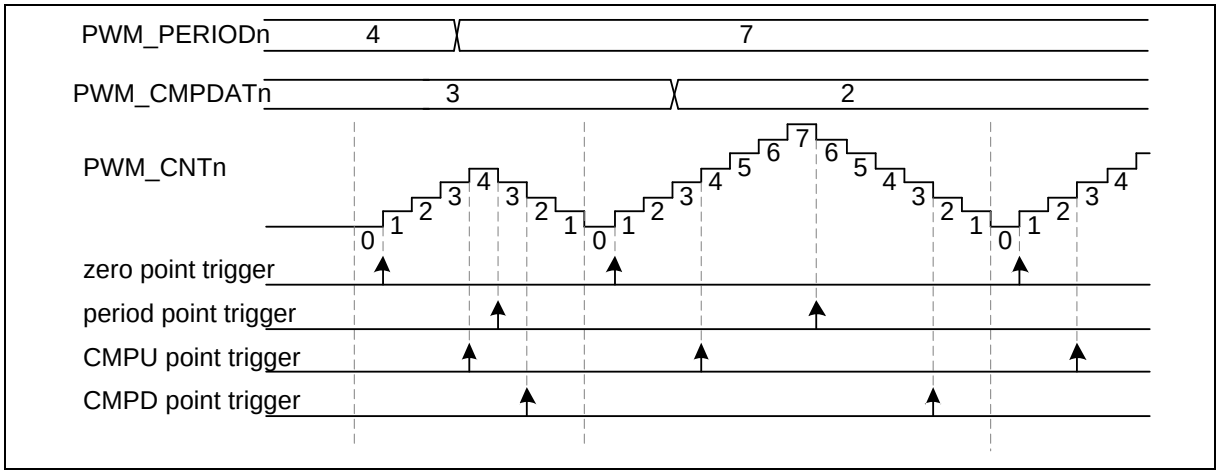


Figure 6.14-35 PWM Trigger EADC/LPADC in Up-Down Counter Type Timing Waveform

6.14.5.24 PWM Trigger DAC Generator

PWM can also be used to trigger DAC conversion. Each PWM pair channel (CH0 and CH1, CH2 and CH3, CH4 and CH5) generates a trigger signal. Using the PWM Trigger DAC Enable Register (PWM_DACTRGEN) can decide at which points to trigger DAC. The timing of the PWM triggering DAC is similar to those for triggering EADC.

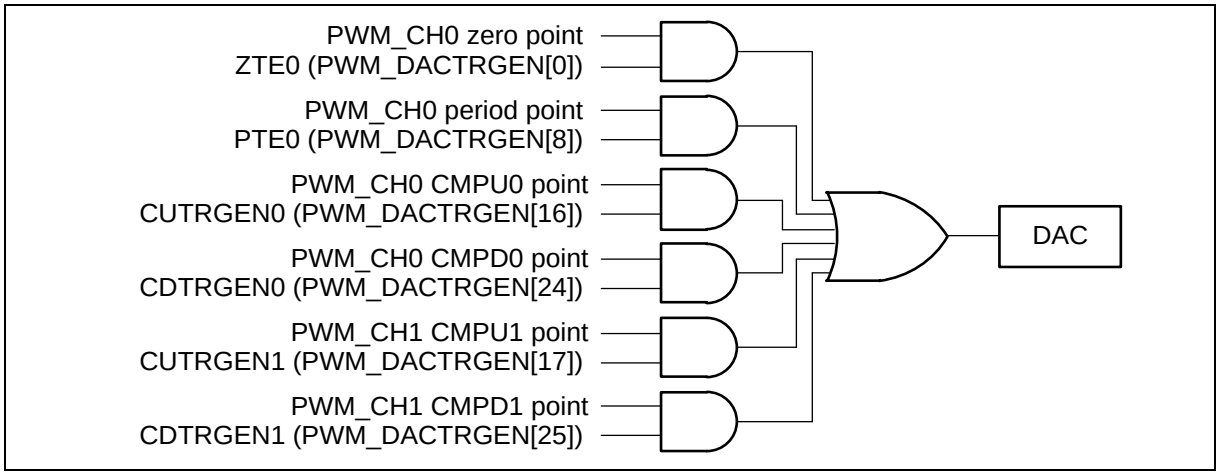


Figure 6.14-36 PWM_CH0 and PWM_CH1 Pair Trigger DAC Block Diagram

6.14.5.25 Capture Operation

The channels of the capture input and the PWM output share the same pin and counter. The counter can operating in up or down counter type. The capture input will be filtered by a 3-bit noise filter. User can enable the noise filter function by CAPNFEN (PWM_CAPNFn[0], n=0~5) reigster, and noise filter sampling clock can be selected by setting CAPNFSEL bits of (PWM_CAPNFn[6:4], n=0~5) register to fit different noise properties. Moreover, by setting the CAPNFCNT (PWM_CAPNFn[10:8], n=0~5) bits, user can define by how many sampling clock cycles a filter will recognize the effective edge of the capture input signal. The capture function will always latch the PWM counter to the RCAPDATn (PWM_RCAPDATn[15:0]) bits or the FCAPDATn (PWM_FCAPDATn[15:0]) bits, if the input channel has a rising transition or a falling transition, respectively. The capture function will also generate an interrupt CAP_INT (using PWM_INT vector) if the rising or falling latch occurs and the corresponding channel n's rising or falling interrupt enable bits are set, where the CAPRIENn (PWM_CAPIEN[5:0]) bit is for the rising edge and the CAPFIENn (PWM_CAPIEN[13:8]) bit is for the falling edge. When rising

or falling latch occurs, the corresponding PWM counter may be reloaded with the value of PWM_PERIODn register, depending on the setting of RCRLDENn or FCRLDENn bits (where RCRLDENn and FCRLDENn are located at PWM_CAPCTL[21:16] and PWM_CAPCTL[29:24], respectively). Note that the corresponding GPIO pins must be configured as the capture function by enable the CAPINENn (PWM_CAPINEN[5:0]) bits for the corresponding capture channel n. Figure 6.14-37 is the capture block diagram of channel 0.

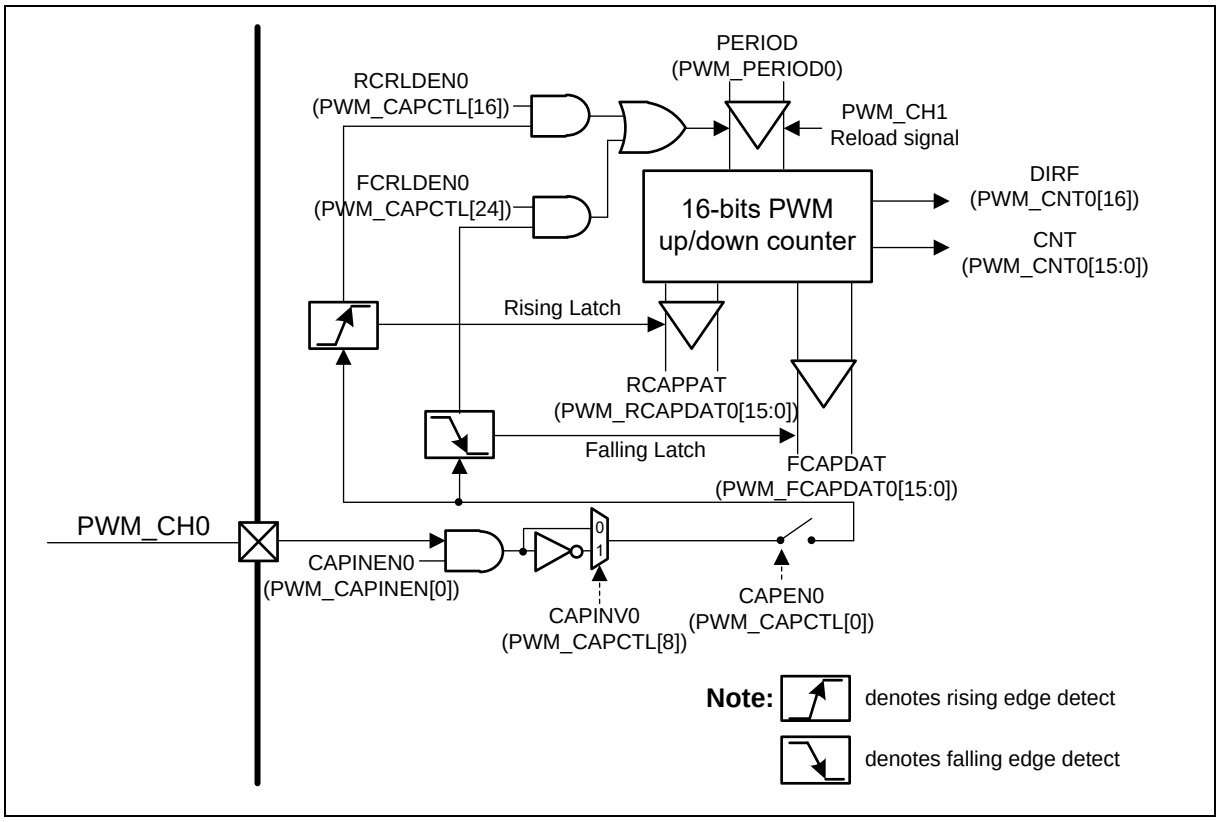


Figure 6.14-37 PWM_CH0 Capture Block Diagram

Figure 6.14-38 illustrates the capture function timing. In this case, the capture counter is set as PWM down counter type and the PERIOD is set to 8 so that the counter counts in the down direction, from 8 to 0. When detecting a falling edge at the capture input pin, the capture function latches the counter value to the PWM_FCAPDATn register. When detecting the rising edge, it latches the counter value to the PWM_RCAPDATn register. In this timing diagram, when the falling edge is detected at the first time, the capture function will reload the counter value from the PERIOD setting because the FCRLDENn bit is enabled. But at the second time, the falling edge does not result in a reload because of the disabled FCRLDENn bit. In this example, the counter also reloads at the rising edge of the capture input because the RCRLDENn bit is enabled, too.

Moreover, if the case is setup as the up counter type, the counter will reload the value zero and count up to the value PERIOD.

Figure 6.14-38 also illustrates the timing example for the interrupt and interrupt flag generation. When the rising edge at channel n is detected, the corresponding CRLIFn (PWM_CAPIF[5:0]) bit is set by hardware. Similarly, a falling edge detection at chnnel n causes the corresponding CFLIFn (PWM_CAPIF[13:8]) bit is set by hardware. CRLIFn and CFLIFn bits can be cleared by software by writing '1'. If the CRLIFn bit is set and the CAPRIENn bit is enabled, the capture function generates an interrupt. If the CFLIFn bit is set and the CAPFIENn is enabled, the interrupt also happens.

A condition which is not shown in Figure 6.14-38 is: if the rising latch happens again when the CRLIFn bit is already set, the overrun status CRLIFOVn (PWM_CAPSTS[5:0]) bit will be set to 1 by hardware to

indicate the CRLIF flag overrunning. Also, if the falling latch happens again, the same hardware operation occurs for the CFLIF interrupt flag and the overrun status CFLIFOVn (PWM_CAPSTS[13:8]).

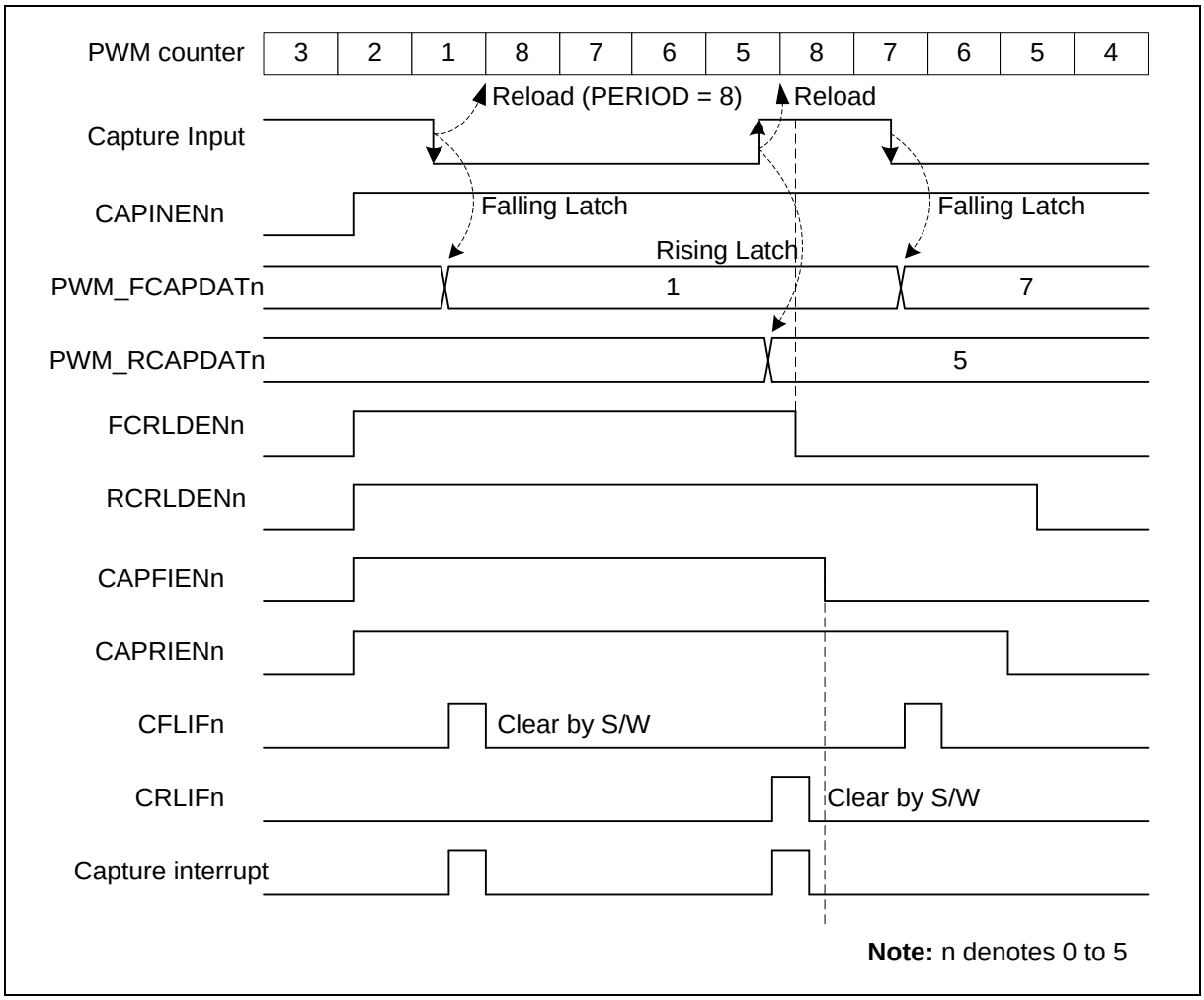


Figure 6.14-38 Capture Operation Waveform

The capture pulse width can be calculated according to the following formula:

For the negative pulse case, the channel low pulse width is calculated as (PWM_PERIODn + 1 - PWM_RCAPDATn) PWM counter time, where one PWM counter time is (CLKPSC+1) * PWMx_CLK clock time. In Figure 6.14-38, the low pulse width is 8+1-5 = 4 PWM counter time .

For the positive pulse case, the channel high pulse width is calculated as (PWM_PERIODn + 1 - PWM_FCAPDATn) PWM counter time, where one PWM counter time is (CLKPSC+1) * PWMx_CLK clock time. In Figure 6.14-38, the high pulse width is 8+1-7 = 2 PWM counter time.

6.14.5.26 Capture Trigger PDMA Function

The PWM module supports the PDMA transfer function when operating in the capture mode (**Note:** If the PDMA function is not supported, this bit field will become invalid). When the corresponding PDMA enable bit CHENn_m (CHEN0_1 at PWM_PDMACTL[0], CHEN2_3 at PWM_PDMACTL[8] and CHEN4_5 at PWM_PDMACTL[16], where n and m denote complement pair channels) is set, the capture module will issue a request to PDMA controller when the preceding capture event has happened. Figure 6.14-39 shows Capture Trigger PDMA Function Architecture Diagram for PWM_CH0 and PWM_CH1 Pair.

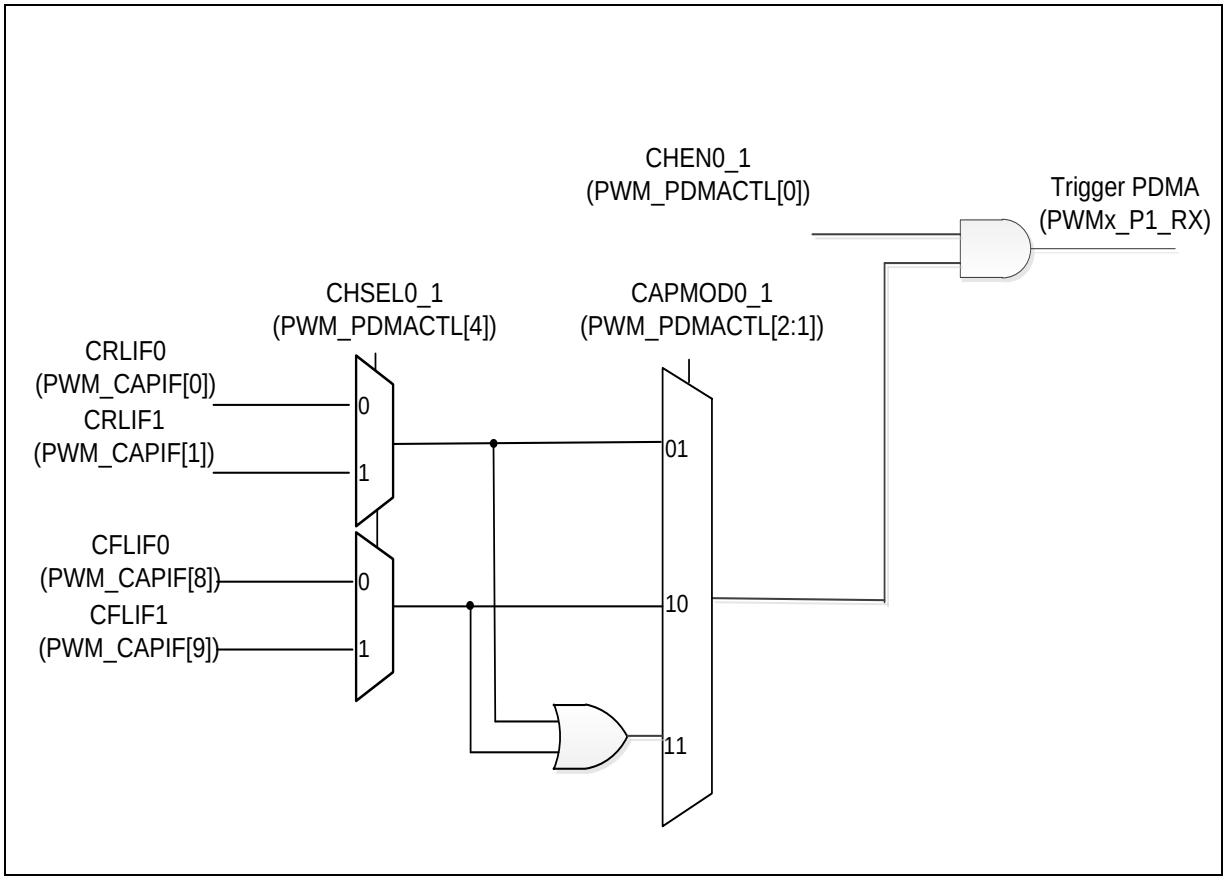


Figure 6.14-39 Capture Trigger PDMA Function Architecture Diagram for PWM_CH0 and PWM_CH1 Pair

The PDMA controller will issue an acknowledgement to the capture module after it has read back the CAPBUF (PWM_PDMACAPn_m[15:0], n, m denotes complement pair channels) register in the capture module and has sent the register value to the memory. By setting CAPMODn_m (CAPMOD0_1 at PWM_PDMACTL[2:1], CAPMOD2_3 at PWM_PDMACTL[10:9] and CAPMOD4_5 at PWM_PDMACTL[18:17]) bits, the PDMA can transfer the rising edge captured data or falling edge captured data or both of them to the memory. When using the PDMA to transfer both of the falling and rising edge data, remember to set CAPORDn_m (CAPORD0_1 at PWM_PDMACTL[3], CAPORD2_3 at PWM_PDMACTL[11] and CAPORD4_5 at PWM_PDMACTL[19]) bit to decide the order of the transferred data (falling edge captured is first or rising edge captured first). The complement pair channels share a PDMA channel. Therefore, a selection bit CHSELn_m (CHSEL0_1 (PWM_PDMACTL[4]), CHSEL2_3 (PWM_PDMACTL[12]) and CHSEL4_5 (PWM_PDMACTL[20])) bit is used to decide either channel n or channel m can be serviced by the PDMA channel.

Figure 6.14-40 is capture PDMA waveform. In this case, the CHSEL0_1 (PWM_PDMACTL[4]) bit is set to 0. Hence the PDMA will service channel 0 for the capture data transfer. CAPMOD0_1 (PWM_PDMACTL[2:1]) bits are set to 3. That means both of the rising and falling edge captured data will be transferred to the memory. The CAPORD0_1 (PWM_PDMACTL[3]) is set to 1, so the rising edge data will be the first data to transfer and following is the falling edge data to transfer. As shown in Figure 6.14-40, the last assertions of the CAPRIF0 CRLIF0 and CAPFIF0 CFLIF0 signal have some overlap. The PWM_RCAPDAT0 value 11 will be loaded to PWM_PDMACAP0_1 register to wait for transfer but not the PWM_FCAPDAT0 value 6. The PWM_PDMACAP0_1 register saves the data which will be transferred to the memory by PDMA. The HWDATA in this figure denotes the data which are being

transferred by PDMA.

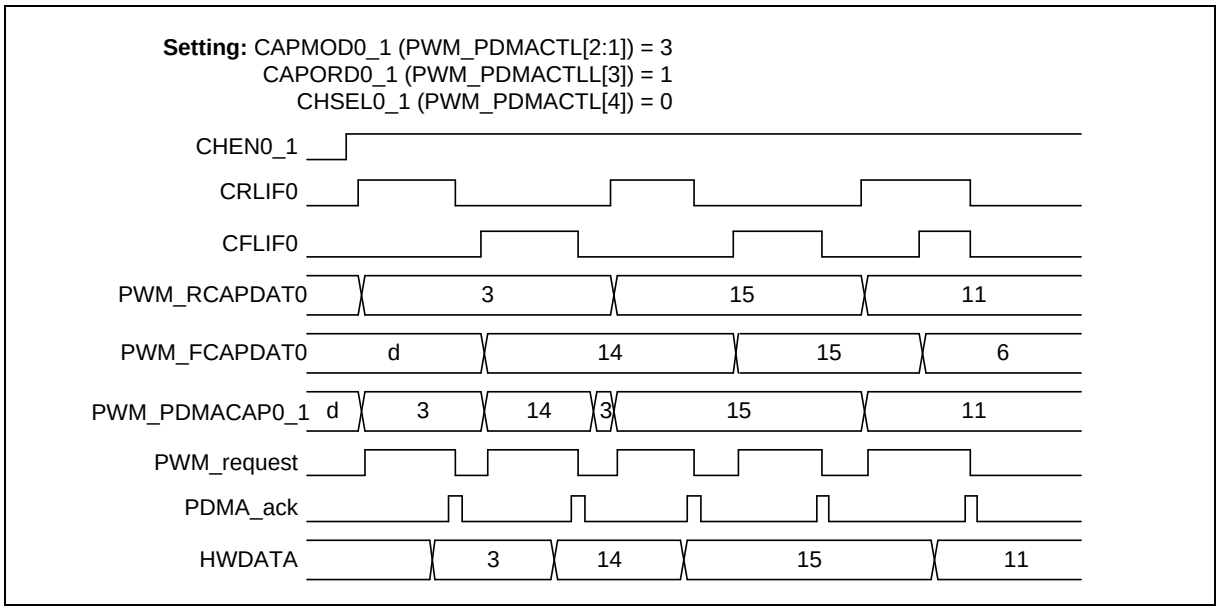


Figure 6.14-40 Capture PDMA Operation Waveform of Channel 0

6.14.5.27 Accumulator Trigger PDMA Function

The PWM module supports the PDMA transfer function when accumulator interrupt happened. Figure 6.14-41 shows accumulator Trigger PDMA function architecture diagram. When the corresponding PDMA enable bit APDMAENn (PWM_APDMACTL[n], n=0,2,4) is set, accumulator module will send a request to PDMA controller when accumulator interrupt has happened, meaning that IFAIFn (PWM_AINTSTS[n], n=0,2,4) is set 1. The PDMA controller will issue an acknowledge to accumulator after it has read memory data and send the data to the particular register (PWM_PERIOD, etc.). So, user can use this function to change accumulator interrupt frequency.

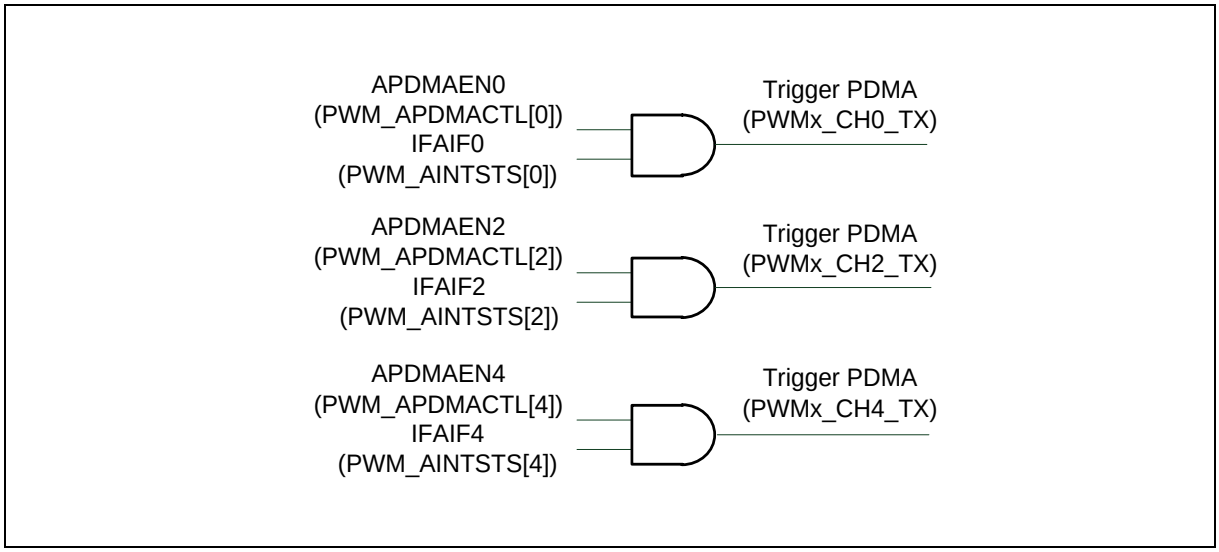


Figure 6.14-41 Accumulator Trigger PDMA Function Architecture Diagram

6.14.5.28 Accumulator Stop Mode

The PWM module supports the Accumulator Stop Mode to stop counting when accumulator interrupt happened. Figure 6.12-22 shows Accumulator Stop Mode waveform. When the corresponding

STPMOD (PWM_IFAn[24], n=0,2,4) is set, accumulator module will disable CNT_ENn (PWM_CNTEN[n], n=0,2,4) after counter finishs whole period.

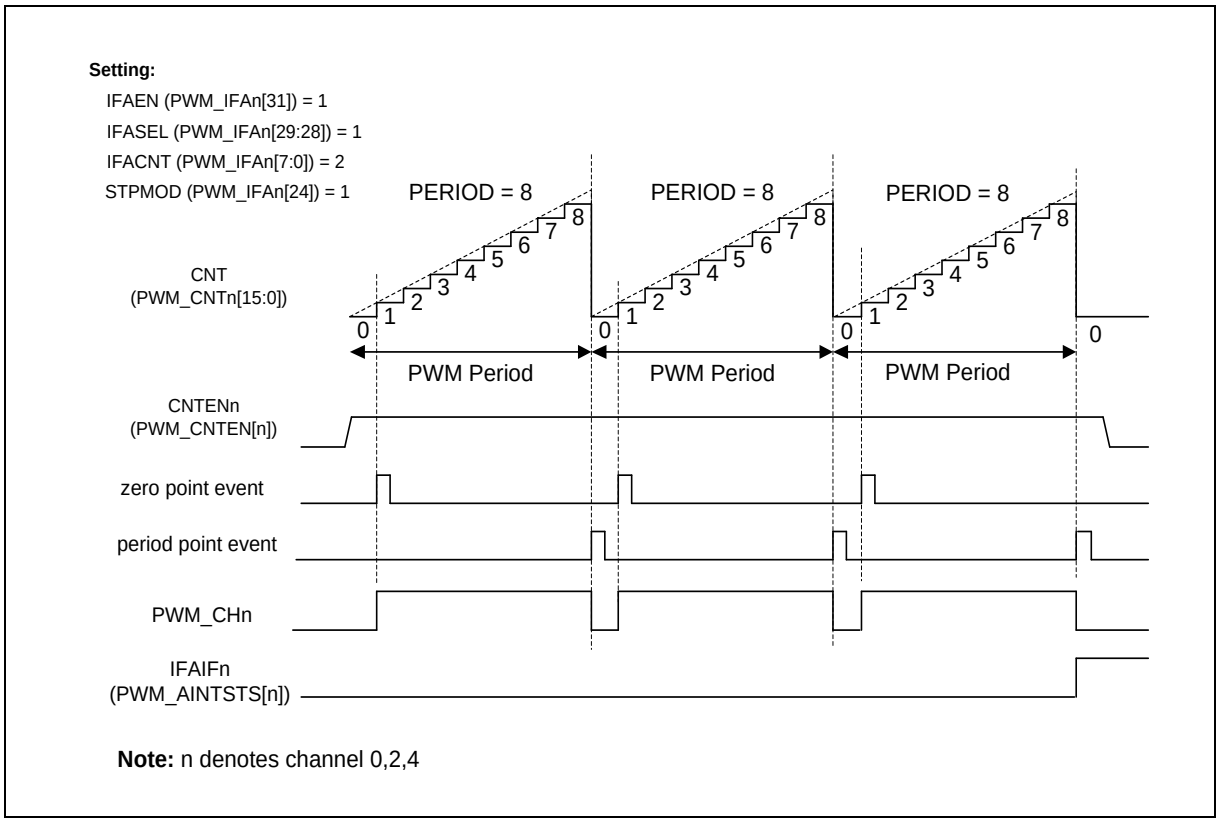


Figure 6.14-42 PWM Accumulator Stop Mode Waveform

6.14.5.29 External Pin Event Trigger

External pin event can trigger PWM counter. Set EXTETEN (PWM_EXTETCTL[0]) bit to 1 can enable this function. By setting CNTACTS (PWM_EXTETCTL[5:4]) can choose counter action: reset, start or reset and start. Setting EXTTRGS (PWM_EXTETCTL[11:8]) can select external trigger source from INT0~INT7.

Trigger event comes from detecting INT0~INT7 rising, falling or both of rising and falling. More detail setting is in GPIO chapter.

6.14.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
PWM Base Address: **PWM0_BA = 0x4005_C000** **PWM1_BA = 0x4005_D000**				
PWM_CTL0 **x=0, 1**	PWMx_BA+0x00	R/W	PWM Control Register 0	0x0000_0000
PWM_CTL1 **x=0, 1**	PWMx_BA+0x04	R/W	PWM Control Register 1	0x0000_0000
PWM_CLKSRC **x=0, 1**	PWMx_BA+0x10	R/W	PWM Clock Source Register	0x0000_0000
PWM_CLKPSC0_1 **x=0, 1**	PWMx_BA+0x14	R/W	PWM Clock Prescale Register 0/1	0x0000_0000
PWM_CLKPSC2_3 **x=0, 1**	PWMx_BA+0x18	R/W	PWM Clock Prescale Register 2/3	0x0000_0000
PWM_CLKPSC4_5 **x=0, 1**	PWMx_BA+0x1C	R/W	PWM Clock Prescale Register 4/5	0x0000_0000
PWM_CNTEN **x=0, 1**	PWMx_BA+0x20	R/W	PWM Counter Enable Register	0x0000_0000
PWM_CNTCLR **x=0, 1**	PWMx_BA+0x24	R/W	PWM Clear Counter Register	0x0000_0000
PWM_PERIOD0 **x=0, 1**	PWMx_BA+0x30	R/W	PWM Period Register 0	0x0000_0000
PWM_PERIOD2 **x=0, 1**	PWMx_BA+0x38	R/W	PWM Period Register 2	0x0000_0000
PWM_PERIOD4 **x=0, 1**	PWMx_BA+0x40	R/W	PWM Period Register 4	0x0000_0000
PWM_CMPDAT0 **x=0, 1**	PWMx_BA+0x50	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1 **x=0, 1**	PWMx_BA+0x54	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2 **x=0, 1**	PWMx_BA+0x58	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3 **x=0, 1**	PWMx_BA+0x5C	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CMPDAT4 **x=0, 1**	PWMx_BA+0x60	R/W	PWM Comparator Register 4	0x0000_0000
PWM_CMPDAT5 **x=0, 1**	PWMx_BA+0x64	R/W	PWM Comparator Register 5	0x0000_0000
PWM_DTCTL0_1	PWMx_BA+0x70	R/W	PWM Dead-time Control Register 0/1	0x0000_0000

x=0, 1				
PWM_DTCTL2_3 **x=0, 1**	PWMx_BA+0x74	R/W	PWM Dead-time Control Register 2/3	0x0000_0000
PWM_DTCTL4_5 **x=0, 1**	PWMx_BA+0x78	R/W	PWM Dead-time Control Register 4/5	0x0000_0000
PWM_CNT0 **x=0, 1**	PWMx_BA+0x90	R	PWM Counter Register 0	0x0000_0000
PWM_CNT2 **x=0, 1**	PWMx_BA+0x98	R	PWM Counter Register 2	0x0000_0000
PWM_CNT4 **x=0, 1**	PWMx_BA+0xA0	R	PWM Counter Register 4	0x0000_0000
PWM_WGCTL0 **x=0, 1**	PWMx_BA+0xB0	R/W	PWM Generation Register 0	0x0000_0000
PWM_WGCTL1 **x=0, 1**	PWMx_BA+0xB4	R/W	PWM Generation Register 1	0x0000_0000
PWM_MSKEN **x=0, 1**	PWMx_BA+0xB8	R/W	PWM Mask Enable Register	0x0000_0000
PWM_MSK **x=0, 1**	PWMx_BA+0xBC	R/W	PWM Mask Data Register	0x0000_0000
PWM_BNF **x=0, 1**	PWMx_BA+0xC0	R/W	PWM Brake Noise Filter Register	0x0000_0000
PWM_FAILBRK **x=0, 1**	PWMx_BA+0xC4	R/W	PWM System Fail Brake Control Register	0x0000_0000
PWM_BRKCTL0_1 **x=0, 1**	PWMx_BA+0xC8	R/W	PWM Brake Edge Detect Control Register 0/1	0x0000_0000
PWM_BRKCTL2_3 **x=0, 1**	PWMx_BA+0xCC	R/W	PWM Brake Edge Detect Control Register 2/3	0x0000_0000
PWM_BRKCTL4_5 **x=0, 1**	PWMx_BA+0xD0	R/W	PWM Brake Edge Detect Control Register 4/5	0x0000_0000
PWM_POLCTL **x=0, 1**	PWMx_BA+0xD4	R/W	PWM Pin Polar Inverse Register	0x0000_0000
PWM_POEN **x=0, 1**	PWMx_BA+0xD8	R/W	PWM Output Enable Register	0x0000_0000
PWM_SWBRK **x=0, 1**	PWMx_BA+0xDC	W	PWM Software Brake Control Register	0x0000_0000
PWM_INTEN0 **x=0, 1**	PWMx_BA+0xE0	R/W	PWM Interrupt Enable Register 0	0x0000_0000
PWM_INTEN1 **x=0, 1**	PWMx_BA+0xE4	R/W	PWM Interrupt Enable Register 1	0x0000_0000
PWM_INTSTS0 **x=0, 1**	PWMx_BA+0xE8	R/W	PWM Interrupt Flag Register 0	0x0000_0000

PWM_INTSTS1 **x=0, 1**	PWMx_BA+0xEC	R/W	PWM Interrupt Flag Register 1	0x0000_0000
PWM_DACTRGEN **x=0,1**	PWMx_BA+0xF4	R/W	PWM Trigger DAC Enable Register	0x0000_0000
PWM_EADCTS0 **x=0, 1**	PWMx_BA+0xF8	R/W	PWM Trigger EADC Source Select Register 0	0x0000_0000
PWM_EADCTS1 **x=0, 1**	PWMx_BA+0xFC	R/W	PWM Trigger EADC Source Select Register 1	0x0000_0000
PWM_SSCTL **x=0, 1**	PWMx_BA+0x110	R/W	PWM Synchronous Start Control Register	0x0000_0000
PWM_SSTRG **x=0, 1**	PWMx_BA+0x114	W	PWM Synchronous Start Trigger Register	0x0000_0000
PWM_STATUS **x=0, 1**	PWMx_BA+0x120	R/W	PWM Status Register	0x0000_0000
PWM_IFA0 **x=0,1**	PWMx_BA+0x130	R/W	PWM Interrupt Flag Accumulator Register 0	0x0000_0000
PWM_IFA2 **x=0,1**	PWMx_BA+0x138	R/W	PWM Interrupt Flag Accumulator Register 2	0x0000_0000
PWM_IFA4 **x=0,1**	PWMx_BA+0x140	R/W	PWM Interrupt Flag Accumulator Register 4	0x0000_0000
PWM_AINTSTS **x=0,1**	PWMx_BA+0x150	R/W	PWM Accumulator Interrupt Flag Register	0x0000_0000
PWM_AINTEN **x=0,1**	PWMx_BA+0x154	R/W	PWM Accumulator Interrupt Enable Register	0x0000_0000
PWM_APDMACTL **x=0,1**	PWMx_BA+0x158	R/W	PWM Accumulator PDMA Control Register	0x0000_0000
PWM_CAPINEN **x=0, 1**	PWMx_BA+0x200	R/W	PWM Capture Input Enable Register	0x0000_0000
PWM_CAPCTL **x=0, 1**	PWMx_BA+0x204	R/W	PWM Capture Control Register	0x0000_0000
PWM_CAPSTS **x=0, 1**	PWMx_BA+0x208	R	PWM Capture Status Register	0x0000_0000
PWM_RCAPDAT0 **x=0, 1**	PWMx_BA+0x20C	R	PWM Rising Capture Data Register 0	0x0000_0000
PWM_FCAPDAT0 **x=0, 1**	PWMx_BA+0x210	R	PWM Falling Capture Data Register 0	0x0000_0000
PWM_RCAPDAT1 **x=0, 1**	PWMx_BA+0x214	R	PWM Rising Capture Data Register 1	0x0000_0000
PWM_FCAPDAT1 **x=0, 1**	PWMx_BA+0x218	R	PWM Falling Capture Data Register 1	0x0000_0000
PWM_RCAPDAT2	PWMx_BA+0x21C	R	PWM Rising Capture Data Register 2	0x0000_0000

x=0, 1				
PWM_FCAPDAT2 **x=0, 1**	PWMx_BA+0x220	R	PWM Falling Capture Data Register 2	0x0000_0000
PWM_RCAPDAT3 **x=0, 1**	PWMx_BA+0x224	R	PWM Rising Capture Data Register 3	0x0000_0000
PWM_FCAPDAT3 **x=0, 1**	PWMx_BA+0x228	R	PWM Falling Capture Data Register 3	0x0000_0000
PWM_RCAPDAT4 **x=0, 1**	PWMx_BA+0x22C	R	PWM Rising Capture Data Register 4	0x0000_0000
PWM_FCAPDAT4 **x=0, 1**	PWMx_BA+0x230	R	PWM Falling Capture Data Register 4	0x0000_0000
PWM_RCAPDAT5 **x=0, 1**	PWMx_BA+0x234	R	PWM Rising Capture Data Register 5	0x0000_0000
PWM_FCAPDAT5 **x=0, 1**	PWMx_BA+0x238	R	PWM Falling Capture Data Register 5	0x0000_0000
PWM_PDMACTL **x=0, 1**	PWMx_BA+0x23C	R/W	PWM PDMA Control Register	0x0000_0000
PWM_PDMACAP0_1 **x=0, 1**	PWMx_BA+0x240	R	PWM Capture Channel 01 PDMA Register	0x0000_0000
PWM_PDMACAP2_3 **x=0, 1**	PWMx_BA+0x244	R	PWM Capture Channel 23 PDMA Register	0x0000_0000
PWM_PDMACAP4_5 **x=0, 1**	PWMx_BA+0x248	R	PWM Capture Channel 45 PDMA Register	0x0000_0000
PWM_CAPIEN **x=0, 1**	PWMx_BA+0x250	R/W	PWM Capture Interrupt Enable Register	0x0000_0000
PWM_CAPIF **x=0, 1**	PWMx_BA+0x254	R/W	PWM Capture Interrupt Flag Register	0x0000_0000
PWM_CAPNF0 **x=0,1**	PWMx_BA+0x258	R/W	PWM Capture Input Noise Filter Register 0	0x0000_0000
PWM_CAPNF1 **x=0,1**	PWMx_BA+0x25C	R/W	PWM Capture Input Noise Filter Register 1	0x0000_0000
PWM_CAPNF2 **x=0,1**	PWMx_BA+0x260	R/W	PWM Capture Input Noise Filter Register 2	0x0000_0000
PWM_CAPNF3 **x=0,1**	PWMx_BA+0x264	R/W	PWM Capture Input Noise Filter Register 3	0x0000_0000
PWM_CAPNF4 **x=0,1**	PWMx_BA+0x268	R/W	PWM Capture Input Noise Filter Register 4	0x0000_0000
PWM_CAPNF5 **x=0,1**	PWMx_BA+0x26C	R/W	PWM Capture Input Noise Filter Register 5	0x0000_0000
PWM_EXTETCTL0 **x=0,1**	PWMx_BA+0x270	R/W	PWM External Event Trigger Control Register 0	0x0000_0000

PWM_EXTETCTL2 **x=0,1**	PWMx_BA+0x278	R/W	PWM External Event Trigger Control Register 2	0x0000_0000
PWM_EXTETCTL4 **x=0,1**	PWMx_BA+0x280	R/W	PWM External Event Trigger Control Register 4	0x0000_0000
PWM_SWEOFCTL **x=0,1**	PWMx_BA+0x288	R/W	PWM Software Event Output Force Control Register	0x0000_0000
PWM_SWEOFTRG **x=0,1**	PWMx_BA+0x28C	R/W	PWM Software Event Output Force Trigger Register	0x0000_0000
PWM_PBUF0 **x=0, 1**	PWMx_BA+0x304	R	PWM PERIOD0 Buffer	0x0000_0000
PWM_PBUF2 **x=0, 1**	PWMx_BA+0x30C	R	PWM PERIOD2 Buffer	0x0000_0000
PWM_PBUF4 **x=0, 1**	PWMx_BA+0x314	R	PWM PERIOD4 Buffer	0x0000_0000
PWM_CMPBUF0 **x=0, 1**	PWMx_BA+0x31C	R	PWM CMPDAT0 Buffer	0x0000_0000
PWM_CMPBUF1 **x=0, 1**	PWMx_BA+0x320	R	PWM CMPDAT1 Buffer	0x0000_0000
PWM_CMPBUF2 **x=0, 1**	PWMx_BA+0x324	R	PWM CMPDAT2 Buffer	0x0000_0000
PWM_CMPBUF3 **x=0, 1**	PWMx_BA+0x328	R	PWM CMPDAT3 Buffer	0x0000_0000
PWM_CMPBUF4 **x=0, 1**	PWMx_BA+0x32C	R	PWM CMPDAT4 Buffer	0x0000_0000
PWM_CMPBUF5 **x=0, 1**	PWMx_BA+0x330	R	PWM CMPDAT5 Buffer	0x0000_0000

6.14.7 Register Description

PWM Control Register 0 (PWM_CTL0)

Register	Offset	R/W	Description	Reset Value
PWM_CTL0	PWMx_BA+0x00	R/W	PWM Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	**DBGHALT**	**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**IMMLDEN5**	**IMMLDEN4**	**IMMLDEN3**	**IMMLDEN2**	**IMMLDEN1**	**IMMLDEN0**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CTRLD5**	**CTRLD4**	**CTRLD3**	**CTRLD2**	**CTRLD1**	**CTRLD0**

Bits	Description	
[31]	**DBGTRIOFF**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects PWM output. PWM pin will be forced as tri-state while ICE debug mode acknowledged. 1 = ICE debug mode acknowledgement disabled. PWM pin will keep output no matter ICE debug mode acknowledged or not. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[30]	**DBGHALT**	**ICE Debug Mode Counter Halt (Write Protect)** If counter halt is enabled, PWM all counters will keep current value until exit ICE debug mode. 0 = ICE debug mode counter halt Disable. 1 = ICE debug mode counter halt Enable. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[29:22]	**Reserved**	Reserved.
[n+16] n=0,1…5	**IMMLDENn**	**Immediately Load Enable Bits** 0 = PERIOD will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the end point or center point of each period by setting CTRLD bit. 1 = PERIOD/CMPDAT will load to PBUF and CMPBUF immediately when software update PERIOD/CMPDAT. **Note:** If IMMLDENn is enabled, WINLDENn and CTRLDn will be invalid.
[15:6]	**Reserved**	Reserved.
[n] n=0,1…5	**CTRLDn**	**Center Load Enable Bits** In up-down counter type, PERIOD will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the center point of a period.

PWM Control Register 1 (PWM_CTL1)

Register	Offset	R/W	Description	Reset Value
PWM_CTL1	PWMx_BA+0x04	R/W	PWM Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					OUTMODE4	OUTMODE2	OUTMODE0
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						CNTTYPE4	
7	6	5	4	3	2	1	0
Reserved		CNTTYPE2		Reserved		CNTTYPE0	

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:24]	**OUTMODEn**	**PWM Output Mode** Each bit n controls the output mode of corresponding PWM channel n. 0 = PWM independent mode. 1 = PWM complementary mode. **Note:** When operating in group function, these bits must all set to the same mode.
[23:10]	**Reserved**	Reserved.
[9:8]	**CNTTYPE4**	**PWM Counter Behavior Type 4** The two bits control channel5 and channel4 00 = Up counter type (supported in capture mode). 01 = Down count type (supported in capture mode). 10 = Up-down counter type. 11 = Reserved.
[7:6]	**Reserved**	Reserved.
[5:4]	**CNTTYPE2**	**PWM Counter Behavior Type 2** The two bits control channel3 and channel2 00 = Up counter type (supported in capture mode). 01 = Down count type (supported in capture mode). 10 = Up-down counter type. 11 = Reserved.
[3:2]	**Reserved**	Reserved.
[1:0]	**CNTTYPE0**	**PWM Counter Behavior Type 0** The two bits control channel1 and channel0 00 = Up counter type (supported in capture mode). 01 = Down count type (supported in capture mode). 10 = Up-down counter type.

		11 = Reserved.

PWM Clock Source Register (PWM_CLKSRC)

Register	Offset	R/W	Description	Reset Value
PWM_CLKSRC	PWMx_BA+0x10	R/W	PWM Clock Source Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**ECLKSRC4**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**ECLKSRC2**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ECLKSRC0**		

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18:16]	**ECLKSRC4**	**PWM_CH45 External Clock Source Select** 000 = PWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.
[15:11]	**Reserved**	Reserved.
[10:8]	**ECLKSRC2**	**PWM_CH23 External Clock Source Select** 000 = PWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.
[7:3]	**Reserved**	Reserved.
[2:0]	**ECLKSRC0**	**PWM_CH01 External Clock Source Select** 000 = PWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.

PWM Clock Pre-scale Register 0_1, 2_3, 4_5 (PWM_CLKPSC0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_CLKPSC0_1	PWMx_BA+0x14	R/W	PWM Clock Prescale Register 0/1	0x0000_0000
PWM_CLKPSC2_3	PWMx_BA+0x18	R/W	PWM Clock Prescale Register 2/3	0x0000_0000
PWM_CLKPSC4_5	PWMx_BA+0x1C	R/W	PWM Clock Prescale Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CLKPSC**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**CLKPSC**	**PWM Counter Clock Prescale** The clock of PWM counter is decided by clock prescaler. Each PWM pair share one PWM counter clock prescaler. The clock of PWM counter is divided by (CLKPSC+ 1).

PWM Counter Enable Register (PWM_CNTEN)

Register	Offset	R/W	Description	Reset Value
PWM_CNTEN	PWMx_BA+0x20	R/W	PWM Counter Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**CNTEN4**	**Reserved**	**CNTEN2**	**Reserved**	**CNTEN0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**CNTEN4**	**PWM Counter Enable Bit 4** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[3]	**Reserved**	Reserved.
[2]	**CNTEN2**	**PWM Counter Enable Bit 2** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[1]	**Reserved**	Reserved.
[0]	**CNTEN0**	**PWM Counter Enable Bit 0** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.

PWM Clear Counter Register (PWM_CNTCLR)

Register	Offset	R/W	Description	Reset Value
PWM_CNTCLR	PWMx_BA+0x24	R/W	PWM Clear Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**CNTCLR4**	**Reserved**	**CNTCLR2**	**Reserved**	**CNTCLR0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**CNTCLR4**	**Clear PWM Counter Control Bit 4** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[3]	**Reserved**	Reserved.
[2]	**CNTCLR2**	**Clear PWM Counter Control Bit 2** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[1]	**Reserved**	Reserved.
[0]	**CNTCLR0**	**Clear PWM Counter Control Bit 0** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.

PWM Period Register 0, 2, 4 (PWM_PERIOD0, 2, 4)

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD0	PWMx_BA+0x30	R/W	PWM Period Register 0	0x0000_0000
PWM_PERIOD2	PWMx_BA+0x38	R/W	PWM Period Register 2	0x0000_0000
PWM_PERIOD4	PWMx_BA+0x40	R/W	PWM Period Register 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**PWM Period Register** Up-Count mode: In this mode, PWM counter counts from 0 to PERIOD, and restarts from 0. Down-Count mode: In this mode, PWM counter counts from PERIOD to 0, and restarts from PERIOD. PWM period time = (PERIOD+1) * PWM_CLK period. Up-Down-Count mode: In this mode, PWM counter counts from 0 to PERIOD, then decrements to 0 and repeats again. PWM period time = 2 * PERIOD * PWM_CLK period.

PWM Comparator Register 0~5 (PWM_CMPDAT0~5)

Register	Offset	R/W	Description	Reset Value
PWM_CMPDAT0	PWMx_BA+0x50	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1	PWMx_BA+0x54	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2	PWMx_BA+0x58	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3	PWMx_BA+0x5C	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CMPDAT4	PWMx_BA+0x60	R/W	PWM Comparator Register 4	0x0000_0000
PWM_CMPDAT5	PWMx_BA+0x64	R/W	PWM Comparator Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMP**	**PWM Comparator Register** CMP is used to compare with CNT to generate PWM waveform, interrupt and trigger EADC/LPADC/DAC. In independent mode, PWM_CMPDAT0~5 denote as 6 independent PWM_CH0~5 compared point. In complementary mode, PWM_CMPDAT0, 2, 4 denote as first compared point, and PWM_CMPDAT1, 3, 5 denote as second compared point for the corresponding 3 complementary pairs PWM_CH0 and PWM_CH1, PWM_CH2 and PWM_CH3, PWM_CH4 and PWM_CH5.

PWM Dead-time Control Register 0_1, 2_3, 4_5 (PWM_DTCTL0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_DTCTL0_1	PWMx_BA+0x70	R/W	PWM Dead-time Control Register 0/1	0x0000_0000
PWM_DTCTL2_3	PWMx_BA+0x74	R/W	PWM Dead-time Control Register 2/3	0x0000_0000
PWM_DTCTL4_5	PWMx_BA+0x78	R/W	PWM Dead-time Control Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							DTCKSEL
23	22	21	20	19	18	17	16
Reserved							DTEN
15	14	13	12	11	10	9	8
Reserved				DTCNT			
7	6	5	4	3	2	1	0
DTCNT							

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**DTCKSEL**	**Dead-time Clock Select (Write Protect)** 0 = Dead-time clock source from PWM_CLK. 1 = Dead-time clock source from prescaler output. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[23:17]	**Reserved**	Reserved.
[16]	**DTEN**	**Enable Dead-time Insertion for PWM Pair (Write Protect)** Dead-time insertion is only active when this pair of complementary PWM is enabled. If dead- time insertion is inactive, the outputs of pin pair are complementary without any delay. 0 = Dead-time insertion Disabled on the pin pair. 1 = Dead-time insertion Enabled on the pin pair. **Note:** This bit is write protected. Refer to SYS_REGLCTL register. **Note:** PWM pair means PWM_CH0, PWM_CH1 are a pair, PWM_CH2, PWM_CH3 are a pair and PWM_CH4, PWM_CH5 are a pair.
[15:12]	**Reserved**	Reserved.
[11:0]	**DTCNT**	**Dead-time Counter (Write Protect)** The dead-time can be calculated from the following formula: DTCKSEL=0: Dead-time = (DTCNT[11:0]+1) * PWM_CLK period. DTCKSEL=1: Dead-time = (DTCNT[11:0]+1) * PWM_CLK period * (CLKPSC+1). **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

PWM Counter Register 0, 2, 4 (PWM_CNT0, 2, 4)

Register	Offset	R/W	Description	Reset Value
PWM_CNT0	PWMx_BA+0x90	R	PWM Counter Register 0	0x0000_0000
PWM_CNT2	PWMx_BA+0x98	R	PWM Counter Register 2	0x0000_0000
PWM_CNT4	PWMx_BA+0xA0	R	PWM Counter Register 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**DIRF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**DIRF**	**PWM Direction Indicator Flag (Read Only)** 0 = Counter is counting down. 1 = Counter is counting up.
[15:0]	**CNT**	**PWM Data Register (Read Only)** User can monitor CNT to know the current value in 16-bit period counter.

PWM Generation Register 0 (PWM_WGCTL0)

Register	Offset	R/W	Description	Reset Value
PWM_WGCTL0	PWMx_BA+0xB0	R/W	PWM Generation Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**PRDPCTL5**		**PRDPCTL4**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRDPCTL3		**PRDPCTL2**		**PRDPCTL1**		**PRDPCTL0**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ZPCTL5**		**ZPCTL4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ZPCTL3		**ZPCTL2**	**ZPCTL1**			**ZPCTL0**	

Bits	Description	
[31:28]	**Reserved**	Reserved.
[17+2n:16+2n] n=0,1..5	**PRDPCTLn**	**PWM Period/Center Point Control** 00 = Do nothing. 01 = PWM period/center point output Low. 10 = PWM period/center point output High. 11 = PWM period/center point output Toggle. **Note 1:** PWM can control output level when PWM counter counts to (PERIODn+1). **Note 2:** This bit is center point control when PWM counter operating in up-down counter type.
[15:12]	**Reserved**	Reserved.
[1+2n:2n] n=0,1..5	**ZPCTLn**	**PWM Zero Point Control** 00 = Do nothing. 01 = PWM zero point output Low. 10 = PWM zero point output High. 11 = PWM zero point output Toggle. **Note:** PWM can control output level when PWM counter counts to 0.

PWM Generation Register 1 (PWM_WGCTL1)

Register	Offset	R/W	Description	Reset Value
PWM_WGCTL1	PWMx_BA+0xB4	R/W	PWM Generation Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**CMPDCTL5**		**CMPDCTL4**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPDCTL3		**CMPDCTL2**		**CMPDCTL1**		**CMPDCTL0**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CMPUCTL5**		**CMPUCTL4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPUCTL3		**CMPUCTL2**		**CMPUCTL1**		**CMPUCTL0**	

Bits	Description	
[31:28]	**Reserved**	Reserved.
[17+2n:16+2n] n=0,1..5	**CMPDCTLn**	**PWM Compare Down Point Control** 00 = Do nothing. 01 = PWM compare down point output Low. 10 = PWM compare down point output High. 11 = PWM compare down point output Toggle. **Note 1:** PWM can control output level when PWM counter counts down to CMPDAT. **Note 2:** In complementary mode, CMPDCTL1, 3, 5 is used as another CMPDCTL for channel 0, 2, 4.
[15:12]	**Reserved**	Reserved.
[1+2n:2n] n=0,1..5	**CMPUCTLn**	**PWM Compare Up Point Control** 00 = Do nothing. 01 = PWM compare up point output Low. 10 = PWM compare up point output High. 11 = PWM compare up point output Toggle. **Note 1:** PWM can control output level when PWM counter counts up to CMPDAT. **Note 2:** In complementary mode, CMPUCTL1, 3, 5 is used as another CMPUCTL for channel 0, 2, 4.

PWM Mask Enable Register (PWM_MSKEN)

Register	Offset	R/W	Description	Reset Value
PWM_MSKEN	PWMx_BA+0xB8	R/W	PWM Mask Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**MSKEN5**	**MSKEN4**	**MSKEN3**	**MSKEN2**	**MSKEN1**	**MSKEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**MSKENn**	**PWM Mask Enable Bits** The PWM output signal will be masked when this bit is enabled. The corresponding PWM channel n will output MSKDATn (PWM_MSK[5:0]) data. 0 = PWM output signal is non-masked. 1 = PWM output signal is masked and output MSKDATn data.

PWM Mask DATA Register (PWM_MSK)

Register	Offset	R/W	Description	Reset Value
PWM_MSK	PWMx_BA+0xBC	R/W	PWM Mask Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**MSKDAT5**	**MSKDAT4**	**MSKDAT3**	**MSKDAT2**	**MSKDAT1**	**MSKDAT0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**MSKDATn**	**PWM Mask Data Bit** This data bit control the state of PWMn output pin, if corresponding mask function is enabled. Each bit n controls the corresponding PWM channel n. 0 = Output logic low to PWM channel n. 1 = Output logic high to PWM channel n.

PWM Brake Noise Filter Register (PWM_BNF)

Register	Offset	R/W	Description	Reset Value
PWM_BNF	PWMx_BA+0xC0	R/W	PWM Brake Noise Filter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							**BK1SRC**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**BK0SRC**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BRK1PINV	**BRK1FCNT**			**BRK1FSEL**			**BRK1NFEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BRK0PINV	**BRK0FCNT**			**BRK0FSEL**			**BRK0NFEN**

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**BK1SRC**	**Brake 1 Pin Source Select** For PWM0 setting: 0 = Brake 1 pin source come from PWM0_BRAKE1. 1 = Brake 1 pin source come from PWM1_BRAKE1. For PWM1 setting: 0 = Brake 1 pin source come from PWM1_BRAKE1. 1 = Brake 1 pin source come from PWM0_BRAKE1.
[23:17]	**Reserved**	Reserved.
[16]	**BK0SRC**	**Brake 0 Pin Source Select** For PWM0 setting: 0 = Brake 0 pin source come from PWM0_BRAKE0. 1 = Brake 0 pin source come from PWM1_BRAKE0. For PWM1 setting: 0 = Brake 0 pin source come from PWM1_BRAKE0. 1 = Brake 0 pin source come from PWM0_BRAKE0.
[15]	**BRK1PINV**	**Brake 1 Pin Inverse** 0 **=** Brake pin event will be detected if PWMx_BRAKE1 pin status transfer from low to high in edge-detect, or pin status is high in level-detect. 1 = Brake pin event will be detected if PWMx_BRAKE1 pin status transfer from high to low in edge-detect, or pin status is low in level-detect.
[14:12]	**BRK1FCNT**	**Brake 1 Edge Detector Filter Count** The register bits control the Brake1 filter counter to count from 0 to BRK1FCNT.
[11:9]	**BRK1FSEL**	**Brake 1 Edge Detector Filter Clock Selection** 000 = Filter clock = HCLK. 001 = Filter clock = HCLK/2. 010 = Filter clock = HCLK/4.

		011 = Filter clock = HCLK/8. 100 = Filter clock = HCLK/16. 101 = Filter clock = HCLK/32. 110 = Filter clock = HCLK/64. 111 = Filter clock = HCLK/128.
[8]	**BRK1NFEN**	**PWM Brake 1 Noise Filter Enable Bit** 0 = Noise filter of PWM Brake 1 Disabled. 1 = Noise filter of PWM Brake 1 Enabled.
[7]	**BRK0PINV**	**Brake 0 Pin Inverse** 0 **=** brake pin event will be detected if PWMx_BRAKE0 pin status transfer from low to high in edge-detect, or pin status is high in level-detect. 1 = brake pin event will be detected if PWMx_BRAKE0 pin status transfer from high to low in edge-detect, or pin status is low in level-detect.
[6:4]	**BRK0FCNT**	**Brake 0 Edge Detector Filter Count** The register bits control the Brake0 filter counter to count from 0 to BRK1FCNT.
[3:1]	**BRK0FSEL**	**Brake 0 Edge Detector Filter Clock Selection** 000 = Filter clock = HCLK. 001 = Filter clock = HCLK/2. 010 = Filter clock = HCLK/4. 011 = Filter clock = HCLK/8. 100 = Filter clock = HCLK/16. 101 = Filter clock = HCLK/32. 110 = Filter clock = HCLK/64. 111 = Filter clock = HCLK/128.
[0]	**BRK0NFEN**	**PWM Brake 0 Noise Filter Enable Bit** 0 = Noise filter of PWM Brake 0 Disabled. 1 = Noise filter of PWM Brake 0 Enabled.

PWM System Fail Brake Control Register (PWM_FAILBRK)

Register	Offset	R/W	Description	Reset Value
PWM_FAILBRK	PWMx_BA+0xC4	R/W	PWM System Fail Brake Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**CORBRKEN**	**RAMBRKEN**	**BODBRKEN**	**CSSBRKEN**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**CORBRKEN**	**Core Lockup Detection Trigger PWM Brake Function Enable Bit** 0 = Brake Function triggered by Core lockup detection Disabled. 1 = Brake Function triggered by Core lockup detection Enabled.
[2]	**RAMBRKEN**	**SRAM Parity Error Detection Trigger PWM Brake Function Enable Bit** 0 = Brake Function triggered by SRAM parity error detection Disabled. 1 = Brake Function triggered by SRAM parity error detection Enabled.
[1]	**BODBRKEN**	**Brown-out Detection Trigger PWM Brake Function Enable Bit** 0 = Brake Function triggered by BOD Disabled. 1 = Brake Function triggered by BOD Enabled.
[0]	**CSSBRKEN**	**Clock Security System Detection Trigger PWM Brake Function Enable Bit** 0 = Brake Function triggered by CSS detection Disabled. 1 = Brake Function triggered by CSS detection Enabled.

PWM Brake Edge Detect Control Register 0_1, 2_3, 4_5 (PWM_BRKCTL0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_BRKCTL0_1	PWMx_BA+0xC8	R/W	PWM Brake Edge Detect Control Register 0/1	0x0000_0000
PWM_BRKCTL2_3	PWMx_BA+0xCC	R/W	PWM Brake Edge Detect Control Register 2/3	0x0000_0000
PWM_BRKCTL4_5	PWMx_BA+0xD0	R/W	PWM Brake Edge Detect Control Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
VBSRLBEN	Reserved		EADC0LBEN	Reserved			
23	22	21	20	19	18	17	16
VBSREBEN	Reserved		EADC0EBEN	BRKAODD		BRKAEVEN	
15	14	13	12	11	10	9	8
SYSLBEN	VBSNLBEN	BRKP1LEN	BRKP0LEN	Reserved	CPO2LBEN	CPO1LBEN	CPO0LBEN
7	6	5	4	3	2	1	0
SYSEBEN	VBSNEBEN	BRKP1EEN	BRKP0EEN	Reserved	CPO2EBEN	CPO1EBEN	CPO0EBEN

Bits	Description	
[31]	**VBSRLBEN**	**UTCPD0 V_{BUS} Source Disable As Level-detect Brake Source (Write Protect)** 0 = UTCPD0 V_{BUS} source disable as level-detect brake source Disabled. 1 = UTCPD0 V_{BUS} source disable as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[30:29]	**Reserved**	Reserved.
[28]	**EADC0LBEN**	**Enable EADC0 Result Monitor as Level-detect Brake Source (Write Protect)** 0 = EADC0RM as level-detect brake source Disabled. 1 = EADC0RM as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[27:24]	**Reserved**	Reserved.
[23]	**VBSREBEN**	**UTCPD0 V_{BUS} Source Disable As Edge-detect Brake Source (Write Protect)** 0 = UTCPD0 V_{BUS} source disable as edge-detect brake source Disabled. 1 = UTCPD0 V_{BUS} source disable as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[22:21]	**Reserved**	Reserved.
[20]	**EADC0EBEN**	**Enable EADC0 Result Monitor as Edge-detect Brake Source (Write Protect)** 0 = EADC0RM as edge-detect brake source Disabled. 1 = EADC0RM as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[19:18]	**BRKAODD**	**PWM Brake Action Select for Odd Channel (Write Protect)** 00 = PWM odd channel level-detect or edge-detect brake function not affect channel output. 01 = PWM odd channel output tri-state when level-detect or edge-detect brake happened.

		10 = PWM odd channel output low level when level-detect or edge-detect brake happened. 11 = PWM odd channel output high level when level-detect or edge-detect brake happened. **Note:** These bits are write protected. Refer to SYS_REGLCTL register.
[17:16]	**BRKAEVEN**	**PWM Brake Action Select for Even Channel (Write Protect)** 00 = PWM even channel level-detect or edge-detect brake function not affect channel output. 01 = PWM even channel output tri-state when level-detect or edge-detect brake happened. 10 = PWM even channel output low level when level-detect or edge-detect brake happened. 11 = PWM even channel output high level when level-detect or edge-detect brake happened. **Note:** These bits are write protected. Refer to SYS_REGLCTL register.
[15]	**SYSLBEN**	**Enable System Fail As Level-detect Brake Source (Write Protect)** 0 = System Fail condition as level-detect brake source Disabled. 1 = System Fail condition as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[14]	**VBSNLBEN**	**UTCPD0 V_{BUS} Sink Disable As Level-detect Brake Source (Write Protect)** 0 = UTCPD0 V_{BUS} sink disable as level-detect brake source Disabled. 1 = UTCPD0 V_{BUS} sink disable as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[13]	**BRKP1LEN**	**Enable BKP1 Pin As Level-detect Brake Source (Write Protect)** 0 = PWMx_BRAKE1 pin as level-detect brake source Disabled. 1 = PWMx_BRAKE1 pin as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[12]	**BRKP0LEN**	**Enable BKP0 Pin As Level-detect Brake Source (Write Protect)** 0 = PWMx_BRAKE0 pin as level-detect brake source Disabled. 1 = PWMx_BRAKE0 pin as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[11]	**Reserved**	Reserved.
[10]	**CPO2LBEN**	**Enable ACMP2_O Digital Output As Level-detect Brake Source (Write Protect)** 0 = ACMP2_O as level-detect brake source Disabled. 1 = ACMP2_O as level-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[9]	**CPO1LBEN**	**Enable ACMP1_O Digital Output As Level-detect Brake Source (Write Protect)** 0 = ACMP1_O as level-detect brake source Disabled. 1 = ACMP1_O as level-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[8]	**CPO0LBEN**	**Enable ACMP0_O Digital Output As Level-detect Brake Source (Write Protect)** 0 = ACMP0_O as level-detect brake source Disabled. 1 = ACMP0_O as level-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[7]	**SYSEBEN**	**Enable System Fail As Edge-detect Brake Source (Write Protect)** 0 = System Fail condition as edge-detect brake source Disabled. 1 = System Fail condition as edge-detect brake source Enabled.

		Note: This bit is write protected. Refer to SYS_REGLCTL register.
[6]	**VBSNEBEN**	**UTCPD0 V_{BUS} Sink Disable As Edge-detect Brake Source (Write Protect)** 0 = UTCPD0 V_{BUS} sink disable as edge-detect brake source Disabled. 1 = UTCPD0 V_{BUS} sink disable as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[5]	**BRKP1EEN**	**Enable PWMx_BRAKE1 Pin As Edge-detect Brake Source (Write Protect)** 0 = BKP1 pin as edge-detect brake source Disabled. 1 = BKP1 pin as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[4]	**BRKP0EEN**	**Enable PWMx_BRAKE0 Pin As Edge-detect Brake Source (Write Protect)** 0 = BKP0 pin as edge-detect brake source Disabled. 1 = BKP0 pin as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[3]	**Reserved**	Reserved.
[2]	**CPO2EBEN**	**Enable ACMP2_O Digital Output As Edge-detect Brake Source (Write Protect)** 0 = ACMP2_O as edge-detect brake source Disabled. 1 = ACMP2_O as edge-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[1]	**CPO1EBEN**	**Enable ACMP1_O Digital Output As Edge-detect Brake Source (Write Protect)** 0 = ACMP1_O as edge-detect brake source Disabled. 1 = ACMP1_O as edge-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[0]	**CPO0EBEN**	**Enable ACMP0_O Digital Output As Edge-detect Brake Source (Write Protect)** 0 = ACMP0_O as edge-detect brake source Disabled. 1 = ACMP0_O as edge-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.

PWM Pin Polar Inverse Control (PWM_POLCTL)

Register	Offset	R/W	Description	Reset Value
PWM_POLCTL	PWMx_BA+0xD4	R/W	PWM Pin Polar Inverse Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**PINV5**	**PINV4**	**PINV3**	**PINV2**	**PINV1**	**PINV0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**PINVn**	**PWM PIN Polar Inverse Control** The register controls polarity state of PWM output. 0 = PWM output polar inverse Disabled. 1 = PWM output polar inverse Enabled.

PWM Output Enable Register (PWM_POEN)

Register	Offset	R/W	Description	Reset Value
PWM_POEN	PWMx_BA+0xD8	R/W	PWM Output Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**POEN5**	**POEN4**	**POEN3**	**POEN2**	**POEN1**	**POEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**POENn**	**PWM Pin Output Enable Bits** 0 = PWM pin at tri-state. 1 = PWM pin in output mode.

PWM Software Brake Control Register (PWM_SWBRK)

Register	Offset	R/W	Description	Reset Value
PWM_SWBRK	PWMx_BA+0xDC	W	PWM Software Brake Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**BRKLTRG4**	**BRKLTRG2**	**BRKLTRG0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**BRKETRG4**	**BRKETRG2**	**BRKETRG0**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[8+n/2] n=0,2,4	**BRKLTRGn**	**PWM Level Brake Software Trigger (Write Only) (Write Protect)** Write 1 to this bit will trigger level brake, and set BRKLIFn to 1 in PWM_INTSTS1 register. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**BRKETRGn**	**PWM Edge Brake Software Trigger (Write Only) (Write Protect)** Write 1 to this bit will trigger Edge brake, and set BRKEIFn to 1 in PWM_INTSTS1 register. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

PWM Interrupt Enable Register 0 (PWM_INTEN0)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN0	PWMx_BA+0xE0	R/W	PWM Interrupt Enable Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIEN5	CMPDIEN4	CMPDIEN3	CMPDIEN2	CMPDIEN1	CMPDIEN0
23	22	21	20	19	18	17	16
Reserved		CMPUIEN5	CMPUIEN4	CMPUIEN3	CMPUIEN2	CMPUIEN1	CMPUIEN0
15	14	13	12	11	10	9	8
Reserved			PIEN4	Reserved	PIEN2	Reserved	PIEN0
7	6	5	4	3	2	1	0
Reserved			ZIEN4	Reserved	ZIEN2	Reserved	ZIEN0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CMPDIENn**	**PWM Compare Down Count Interrupt Enable Bits** 0 = Compare down count interrupt Disabled. 1 = Compare down count interrupt Enabled. **Note:** In complementary mode, CMPDIEN1, 3, 5 is used as another CMPDIEN for channel 0, 2, 4.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CMPUIENn**	**PWM Compare Up Count Interrupt Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = Compare up count interrupt Disabled. 1 = Compare up count interrupt Enabled. **Note:** In complementary mode, CMPUIEN1, 3, 5 is used as another CMPUIEN for channel 0, 2, 4.
[15:13]	**Reserved**	Reserved.
[12]	**PIEN4**	**PWM Period Point Interrupt Enable Bit 4** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When up-down counter type, period point means center point.
[11]	**Reserved**	Reserved.
[10]	**PIEN2**	**PWM Period Point Interrupt Enable Bit 2** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When up-down counter type, period point means center point.
[9]	**Reserved**	Reserved.
[8]	**PIEN0**	**PWM Period Point Interrupt Enable Bit 0** 0 = Period point interrupt Disabled.

		1 = Period point interrupt Enabled. **Note:** When up-down counter type, period point means center point.
[7:5]	**Reserved**	Reserved.
[4]	**ZIEN4**	**PWM Zero Point Interrupt Enable Bit 4** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.
[3]	**Reserved**	Reserved.
[2]	**ZIEN2**	**PWM Zero Point Interrupt Enable Bit 2** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.
[1]	**Reserved**	Reserved.
[0]	**ZIEN0**	**PWM Zero Point Interrupt Enable Bit 0** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.

PWM Interrupt Enable Register 1 (PWM_INTEN1)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN1	PWMx_BA+0xE4	R/W	PWM Interrupt Enable Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**BRKLIEN4_5**	**BRKLIEN2_3**	**BRKLIEN0_1**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**BRKEIEN4_5**	**BRKEIEN2_3**	**BRKEIEN0_1**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**BRKLIEN4_5**	**PWM Level-detect Brake Interrupt Enable for Channel4/5 (Write Protect)** 0 = Level-detect Brake interrupt for channel4/5 Disabled. 1 = Level-detect Brake interrupt for channel4/5 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[9]	**BRKLIEN2_3**	**PWM Level-detect Brake Interrupt Enable for Channel2/3 (Write Protect)** 0 = Level-detect Brake interrupt for channel2/3 Disabled. 1 = Level-detect Brake interrupt for channel2/3 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[8]	**BRKLIEN0_1**	**PWM Level-detect Brake Interrupt Enable for Channel0/1 (Write Protect)** 0 = Level-detect Brake interrupt for channel0/1 Disabled. 1 = Level-detect Brake interrupt for channel0/1 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[2]	**BRKEIEN4_5**	**PWM Edge-detect Brake Interrupt Enable for Channel4/5 (Write Protect)** 0 = Edge-detect Brake interrupt for channel4/5 Disabled. 1 = Edge-detect Brake interrupt for channel4/5 Enabled. **Note:** This bitr is write protected. Refer to SYS_REGLCTL register.
[1]	**BRKEIEN2_3**	**PWM Edge-detect Brake Interrupt Enable for Channel2/3 (Write Protect)** 0 = Edge-detect Brake interrupt for channel2/3 Disabled. 1 = Edge-detect Brake interrupt for channel2/3 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[0]	**BRKEIEN0_1**	**PWM Edge-detect Brake Interrupt Enable for Channel0/1 (Write Protect)** 0 = Edge-detect Brake interrupt for channel0/1 Disabled. 1 = Edge-detect Brake interrupt for channel0/1 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

PWM Interrupt Flag Register 0 (PWM_INTSTS0)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS0	PWMx_BA+0xE8	R/W	PWM Interrupt Flag Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	22	21	20	19	18	17	16
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
15	14	13	12	11	10	9	8
Reserved			PIF4	Reserved	PIF2	Reserved	PIF0
7	6	5	4	3	2	1	0
Reserved			ZIF4	Reserved	ZIF2	Reserved	ZIF0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CMPDIFn**	**PWM Compare Down Count Interrupt Flag** Flag is set by hardware when PWM counter down count and reaches PWM_CMPDATn, software can clear this bit by writing 1 to it. **Note:** In complementary mode, CMPDIF1, 3, 5 is used as another CMPDIF for channel 0, 2, 4.
[23:22]	**Reserved**	Reserved.
[21:16]	**CMPUIFn**	**PWM Compare Up Count Interrupt Flag** Flag is set by hardware when PWM counter up count and reaches PWM_CMPDATn, software can clear this bit by writing 1 to it. **Note:** In complementary mode, CMPUIF1, 3, 5 is used as another CMPUIF for channel 0, 2, 4.
[15:13]	**Reserved**	Reserved.
[12]	**PIF4**	**PWM Period Point Interrupt Flag 4** This bit is set by hardware when PWM_CH4 counter reaches PWM_PERIOD4. **Note:** This bit can be cleared to 0 by software writing 1.
[11]	**Reserved**	Reserved.
[10]	**PIF2**	**PWM Period Point Interrupt Flag 2** This bit is set by hardware when PWM_CH2 counter reaches PWM_PERIOD2. **Note:** This bit can be cleared to 0 by software writing 1.
[9]	**Reserved**	Reserved.
[8]	**PIF0**	**PWM Period Point Interrupt Flag 0** This bit is set by hardware when PWM_CH0 counter reaches PWM_PERIOD0. **Note:** This bit can be cleared to 0 by software writing 1.
[7:5]	**Reserved**	Reserved.
[4]	**ZIF4**	**PWM Zero Point Interrupt Flag 4**

		This bit is set by hardware when PWM_CH4 counter reaches 0. **Note:** This bit can be cleared to 0 by software writing 1.
[3]	**Reserved**	Reserved.
[2]	**ZIF2**	**PWM Zero Point Interrupt Flag 2** This bit is set by hardware when PWM_CH2 counter reaches 0. **Note:** This bit can be cleared to 0 by software writing 1.
[1]	**Reserved**	Reserved.
[0]	**ZIF0**	**PWM Zero Point Interrupt Flag 0** This bit is set by hardware when PWM_CH0 counter reaches 0. **Note:** This bit can be cleared to 0 by software writing 1.

PWM Interrupt Flag Register 1 (PWM_INTSTS1)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS1	PWMx_BA+0xEC	R/W	PWM Interrupt Flag Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		BRKLSTS5	BRKLSTS4	BRKLSTS3	BRKLSTS2	BRKLSTS1	BRKLSTS0
23	22	21	20	19	18	17	16
Reserved		BRKESTS5	BRKESTS4	BRKESTS3	BRKESTS2	BRKESTS1	BRKESTS0
15	14	13	12	11	10	9	8
Reserved		BRKLIF5	BRKLIF4	BRKLIF3	BRKLIF2	BRKLIF1	BRKLIF0
7	6	5	4	3	2	1	0
Reserved		BRKEIF5	BRKEIF4	BRKEIF3	BRKEIF2	BRKEIF1	BRKEIF0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**BRKLSTSn**	**PWM Channel n Level-detect Brake Status (Read Only)** 0 = PWM channel n level-detect brake state is released. 1 = When PWM channel n level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel n at brake state. **Note:** This bit is read only and auto cleared by hardware. When enabled brake source return to high level, PWM will release brake state until current PWM period finished. The PWM waveform will start output from next full PWM period.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**BRKESTSn**	**PWM Channel n Edge-detect Brake Status (Read Only)** 0 = PWM channel n edge-detect brake state is released. 1 = When PWM channel n edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel n at brake state. **Note:** This bit is read only and auto cleared by hardware. When edge-detect brake interrupt flag is cleared, PWM will release brake state until current PWM period finished. The PWM waveform will start output from next full PWM period.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**BRKLIFn**	**PWM Channel n Level-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel n level-detect brake event do not happened. 1 = When PWM channel n level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**BRKEIFn**	**PWM Channel n Edge-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel n edge-detect brake event do not happened. 1 = When PWM channel n edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

PWM Trigger DAC Enable Register (PWM_DACTRGEN)

Register	Offset	R/W	Description	Reset Value
PWM_DACTRGEN	PWMx_BA+0xF4	R/W	PWM Trigger DAC Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CDTRGEN5	CDTRGEN4	CDTRGEN3	CDTRGEN2	CDTRGEN1	CDTRGEN0
23	22	21	20	19	18	17	16
Reserved		CUTRGEN5	CUTRGEN4	CUTRGEN3	CUTRGEN2	CUTRGEN1	CUTRGEN0
15	14	13	12	11	10	9	8
Reserved			PTE4	Reserved	PTE2	Reserved	PTE0
7	6	5	4	3	2	1	0
Reserved			ZTE4	Reserved	ZTE2	Reserved	ZTE0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CDTRGENn**	**PWM Compare Down Count Point Trigger DAC Enable Bits** PWM can trigger DAC to start action when PWM counter down count to CMP if this bit is set to1. 0 = PWM Compare Down count point trigger DAC function Disabled. 1 = PWM Compare Down count point trigger DAC function Enabled. **Note 1:** This bit should keep at 0 when PWM counter operating in up counter type. **Note 2:** In complementary mode, CDTRGEN1, 3, 5 is used as another CDTRGEN for channel 0, 2, 4.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CUTRGENn**	**PWM Compare Up Count Point Trigger DAC Enable Bits** PWM can trigger DAC to start action when PWM counter counts up to CMP if this bit is set to1. 0 = PWM Compare Up point trigger DAC function Disabled. 1 = PWM Compare Up point trigger DAC function Enabled. **Note 1:** This bit should keep at 0 when PWM counter operating in down counter type. **Note 2:** In complementary mode, CUTRGEN1, 3, 5 is used as another CUTRGEN for channel 0, 2, 4.
[15:13]	**Reserved**	Reserved.
[12]	**PTE4**	**PWM Period Point Trigger DAC Enable Bit 4** PWM can trigger DAC to start action when PWM counter counts up to (PERIODn+1) if this bit is set to1. 0 = PWM period point trigger DAC function Disabled. 1 = PWM period point trigger DAC function Enabled.
[11]	**Reserved**	Reserved.
[10]	**PTE2**	**PWM Period Point Trigger DAC Enable Bit 2** PWM can trigger DAC to start action when PWM counter counts up to (PERIODn+1) if this bit is set to1.

		0 = PWM period point trigger DAC function Disabled. 1 = PWM period point trigger DAC function Enabled.
[9]	**Reserved**	Reserved.
[8]	**PTE0**	**PWM Period Point Trigger DAC Enable Bit 0** PWM can trigger DAC to start action when PWM counter counts up to (PERIODn+1) if this bit is set to1. 0 = PWM period point trigger DAC function Disabled. 1 = PWM period point trigger DAC function Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**ZTE4**	**PWM Zero Point Trigger DAC Enable Bit 4** PWM can trigger EADC/DAC/DMA to start action when PWM counter down count to zero if this bit is set to1. 0 = PWM period point trigger DAC function Disabled. 1 = PWM period point trigger DAC function Enabled.
[3]	**Reserved**	Reserved.
[2]	**ZTE2**	**PWM Zero Point Trigger DAC Enable Bit 2** PWM can trigger EADC/DAC/DMA to start action when PWM counter down count to zero if this bit is set to1. 0 = PWM period point trigger DAC function Disabled. 1 = PWM period point trigger DAC function Enabled.
[1]	**Reserved**	Reserved.
[0]	**ZTE0**	**PWM Zero Point Trigger DAC Enable Bit 0** PWM can trigger EADC/DAC/DMA to start action when PWM counter down count to zero if this bit is set to1. 0 = PWM period point trigger DAC function Disabled. 1 = PWM period point trigger DAC function Enabled.

PWM Trigger EADC Source Select Register 0 (PWM_EADCTS0)

Register	Offset	R/W	Description	Reset Value
PWM_EADCTS0	PWMx_BA+0xF8	R/W	PWM Trigger EADC Source Select Register 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved			TRGSEL3			
23	22	21	20	19	18	17	16
TRGEN2	Reserved			TRGSEL2			
15	14	13	12	11	10	9	8
TRGEN1	Reserved			TRGSEL1			
7	6	5	4	3	2	1	0
TRGEN0	Reserved			TRGSEL0			

Bits	Description	
[31]	**TRGEN3**	**PWM_CH3 Trigger EADC Enable Bit** 0 = PWM_CH3 Trigger EADC function Disabled. 1 = PWM_CH3 Trigger EADC function Enabled.
[30:28]	**Reserved**	Reserved.
[27:24]	**TRGSEL3**	**PWM_CH3 Trigger EADC Source Select** 0000 = PWM_CH2 zero point. 0001 = PWM_CH2 period point. 0010 = PWM_CH2 zero or period point. 0011 = PWM_CH2 up-count CMPDAT point. 0100 = PWM_CH2 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH3 up-count CMPDAT point. 1001 = PWM_CH3 down-count CMPDAT point. Others = reserved.
[23]	**TRGEN2**	**PWM_CH2 Trigger EADC Enable Bit** 0 = PWM_CH2 Trigger EADC function Disabled. 1 = PWM_CH2 Trigger EADC function Enabled.
[22:20]	**Reserved**	Reserved.
[19:16]	**TRGSEL2**	**PWM_CH2 Trigger EADC Source Select** 0000 = PWM_CH2 zero point. 0001 = PWM_CH2 period point. 0010 = PWM_CH2 zero or period point. 0011 = PWM_CH2 up-count CMPDAT point. 0100 = PWM_CH2 down-count CMPDAT point. 0101 = Reserved.

		0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH3 up-count CMPDAT point. 1001 = PWM_CH3 down-count CMPDAT point. Others = reserved.
[15]	**TRGEN1**	**PWM_CH1 Trigger EADC Enable Bit** 0 = PWM_CH1 Trigger EADC function Disabled. 1 = PWM_CH1 Trigger EADC function Enabled.
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL1**	**PWM_CH1 Trigger EADC Source Select** 0000 = PWM_CH0 zero point. 0001 = PWM_CH0 period point. 0010 = PWM_CH0 zero or period point. 0011 = PWM_CH0 up-count CMPDAT point. 0100 = PWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH1 up-count CMPDAT point. 1001 = PWM_CH1 down-count CMPDAT point. Others = reserved.
[7]	**TRGEN0**	**PWM_CH0 Trigger EADC Enable Bit** 0 = PWM_CH0 Trigger EADC function Disabled. 1 = PWM_CH0 Trigger EADC function Enabled.
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL0**	**PWM_CH0 Trigger EADC Source Select** 0000 = PWM_CH0 zero point. 0001 = PWM_CH0 period point. 0010 = PWM_CH0 zero or period point. 0011 = PWM_CH0 up-count CMPDAT point. 0100 = PWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH1 up-count CMPDAT point. 1001 = PWM_CH1 down-count CMPDAT point. Others = reserved.

PWM Trigger EADC Source Select Register 1 (PWM_EADCTS1)

Register	Offset	R/W	Description	Reset Value
PWM_EADCTS1	PWMx_BA+0xFC	R/W	PWM Trigger EADC Source Select Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TRGEN5	Reserved			TRGSEL5			
7	6	5	4	3	2	1	0
TRGEN4	Reserved			TRGSEL4			

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**TRGEN5**	**PWM_CH5 Trigger EADC Enable Bit** 0 = PWM_CH5 Trigger EADC function Disabled. 1 = PWM_CH5 Trigger EADC function Enabled.
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL5**	**PWM_CH5 Trigger EADC Source Select** 0000 = PWM_CH4 zero point. 0001 = PWM_CH4 period point. 0010 = PWM_CH4 zero or period point. 0011 = PWM_CH4 up-count CMPDAT point. 0100 = PWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH5 up-count CMPDAT point. 1001 = PWM_CH5 down-count CMPDAT point. Others = reserved.
[7]	**TRGEN4**	**PWM_CH4 Trigger EADC Enable Bit** 0 = PWM_CH4 Trigger EADC function Disabled. 1 = PWM_CH4 Trigger EADC function Enabled.
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL4**	**PWM_CH4 Trigger EADC Source Select** 0000 = PWM_CH4 zero point. 0001 = PWM_CH4 period point. 0010 = PWM_CH4 zero or period point. 0011 = PWM_CH4 up-count CMPDAT point.

		0100 = PWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH5 up-count CMPDAT point. 1001 = PWM_CH5 down-count CMPDAT point. Others = reserved.

PWM Synchronous Start Control Register (PWM_SSCTL)

Register	Offset	R/W	Description	Reset Value
PWM_SSCTL	PWMx_BA+0x110	R/W	PWM Synchronous Start Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						SSRC	
7	6	5	4	3	2	1	0
Reserved			SSEN4	Reserved	SSEN2	Reserved	SSEN0

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**SSRC**	**PWM Synchronous Start Source Select Bits** 00 = Synchronous start source come from EPWM0. 01 = Synchronous start source come from EPWM1. 10 = Synchronous start source come from PWM0. 11 = Synchronous start source come from PWM1.
[7:5]	**Reserved**	Reserved.
[4]	**SSEN4**	**PWM Synchronous Start Function Enable Bit 4** When synchronous start function is enabled, the PWM_CH4 counter enable bit (CNTEN4) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.
[3]	**Reserved**	Reserved.
[2]	**SSEN2**	**PWM Synchronous Start Function Enable Bit 2** When synchronous start function is enabled, the PWM_CH2 counter enable bit (CNTEN2) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.
[1]	**Reserved**	Reserved.
[0]	**SSEN0**	**PWM Synchronous Start Function Enable Bit 0** When synchronous start function is enabled, the PWM_CH0 counter enable bit (CNTEN0) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.

PWM Synchronous Start Trigger Register (PWM_SSTRG)

Register	Offset	R/W	Description	Reset Value
PWM_SSTRG	PWMx_BA+0x114	W	PWM Synchronous Start Trigger Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**CNTSEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTSEN**	**PWM Counter Synchronous Start Enable (Write Only)** PWM counter synchronous enable function is used to make selected PWM channels (include PWM0_CHx and PWM1_CHx) start counting at the same time. Writing this bit to 1 will also set the counter enable bit (CNTENn, n = 0, 2, 4) if correlated PWM channel counter synchronous start function is enabled.

PWM Status Register (PWM_STATUS)

Register	Offset	R/W	Description	Reset Value
PWM_STATUS	PWMx_BA+0x120	R/W	PWM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							DACTRGF
23	22	21	20	19	18	17	16
LPADCTRG	Reserved	EADCTRG5	EADCTRG4	EADCTRG3	EADCTRG2	EADCTRG1	EADCTRG0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			CNTMAX4	Reserved	CNTMAX2	Reserved	CNTMAX0

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**DACTRGF**	**DAC Start of Conversion Flag** 0 = No DAC start of conversion trigger event has occurred. 1 = A DAC start of conversion trigger event has occurred. **Note:** This bit can be cleared by software writing 1.
[23]	**LPADCTRG**	**LPADC Start of Conversion Status** 0 = Indicates no LPADC start of conversion trigger event has occurred. 1 = An LPADC start of conversion trigger event has occurred. **Note:** This bit can be cleared by software writing 1.
[22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**EADCTRGn**	**EADC Start of Conversion Status** 0 = Indicates no EADC start of conversion trigger event has occurred. 1 = An EADC start of conversion trigger event has occurred. **Note:** This bit can be cleared by software writing 1.
[15:5]	**Reserved**	Reserved.
[4]	**CNTMAX4**	**Time-base Counter 4 Equal to 0xFFFF Latched Flag** 0 = The time-base counter never reached its maximum value 0xFFFF. 1 = The time-base counter reached its maximum value. **Note:** This bit can be cleared by software writing 1.
[3]	**Reserved**	Reserved.
[2]	**CNTMAX2**	**Time-base Counter 2 Equal to 0xFFFF Latched Flag** 0 = indicates the time-base counter never reached its maximum value 0xFFFF. 1 = indicates the time-base counter reached its maximum value. **Note:** This bit can be cleared by software writing 1.
[1]	**Reserved**	Reserved.

[0]	**CNTMAX0**	**Time-base Counter 0 Equal to 0xFFFF Latched Flag** 0 = indicates the time-base counter never reached its maximum value 0xFFFF. 1 = indicates the time-base counter reached its maximum value. **Note:** This bit can be cleared by software writing 1.

PWM Interrupt Flag Accumulator Register (PWM_IFAn)

Register	Offset	R/W	Description	Reset Value
PWM_IFA0	PWMx_BA+0x130	R/W	PWM Interrupt Flag Accumulator Register 0	0x0000_0000
PWM_IFA2	PWMx_BA+0x138	R/W	PWM Interrupt Flag Accumulator Register 2	0x0000_0000
PWM_IFA4	PWMx_BA+0x140	R/W	PWM Interrupt Flag Accumulator Register 4	0x0000_0000

31	30	29	28	27	26	25	24
IFAEN	Reserved	IFASEL		Reserved			STPMOD
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
IFACNT							

Bits	Description	
[31]	**IFAEN**	**PWM_CHn Interrupt Flag Accumulator Enable Bits** 0 = PWM_CHn interrupt flag accumulator Disabled. 1 = PWM_CHn interrupt flag accumulator Enabled.
[30]	**Reserved**	Reserved.
[29:28]	**IFASEL**	**PWM_CHn Interrupt Flag Accumulator Source Select** 00 = PWM_CHn zero point. 01 = PWM_CHn period in channel n. 10 = PWM_CHn up-count compared point. 11 = PWM_CHn down-count compared point.
[27:25]	**Reserved**	Reserved.
[24]	**STPMOD**	**PWM_CHn Accumulator Stop Mode Enable Bits** 0 = PWM_CHn Stop Mode Disable. 1 = PWM_CHn Stop Mode Enable.
[23:8]	**Reserved**	Reserved.
[7:0]	**IFACNT**	**PWM_CHn Interrupt Flag Counter** The register sets the count number which defines (IFACNT+1) times of PWM_CHn period occurs to set bit IFAIFn to request the PWM period interrupt. PWM flag will be set in every IFACNT[15:0] times of PWM period.

PWM Accumulator Interrupt Flag Register (PWM_AINTSTS)

Register	Offset	R/W	Description	Reset Value
PWM_AINTSTS	PWMx_BA+0x150	R/W	PWM Accumulator Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**IFAIF4**	**Reserved**	**IFAIF2**	**Reserved**	**IFAIF0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**IFAIF4**	**PWM_CH4 Interrupt Flag Accumulator Interrupt Flag** Flag is set by hardware when condition match IFASEL in PWM_IFAn register, software can clear this bit by writing 1 to it.
[3]	**Reserved**	Reserved.
[2]	**IFAIF2**	**PWM_CH2 Interrupt Flag Accumulator Interrupt Flag** Flag is set by hardware when condition match IFASEL in PWM_IFAn register, software can clear this bit by writing 1 to it.
[1]	**Reserved**	Reserved.
[0]	**IFAIF0**	**PWM_CH0 Interrupt Flag Accumulator Interrupt Flag** Flag is set by hardware when condition match IFASEL in PWM_IFAn register, software can clear this bit by writing 1 to it.

PWM Accumulator Interrupt Enable Register (PWM_AINTEN)

Register	Offset	R/W	Description	Reset Value
PWM_AINTEN	PWMx_BA+0x154	R/W	PWM Accumulator Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**IFAIEN4**	**Reserved**	**IFAIEN2**	**Reserved**	**IFAIEN0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**IFAIEN4**	**PWM_CH4 Interrupt Flag Accumulator Interrupt Enable Bits** 0 = Interrupt Flag accumulator interrupt Disabled. 1 = Interrupt Flag accumulator interrupt Enabled.
[3]	**Reserved**	Reserved.
[2]	**IFAIEN2**	**PWM_CH2 Interrupt Flag Accumulator Interrupt Enable Bits** 0 = Interrupt Flag accumulator interrupt Disabled. 1 = Interrupt Flag accumulator interrupt Enabled.
[1]	**Reserved**	Reserved.
[0]	**IFAIEN0**	**PWM_CH0 Interrupt Flag Accumulator Interrupt Enable Bits** 0 = Interrupt Flag accumulator interrupt Disabled. 1 = Interrupt Flag accumulator interrupt Enabled.

PWM Accumulator PDMA Control Register (PWM_APDMACTL)

Register	Offset	R/W	Description	Reset Value
PWM_APDMACTL	PWMx_BA+0x158	R/W	PWM Accumulator PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**APDMAEN4**	**Reserved**	**APDMAEN2**	**Reserved**	**APDMAEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**APDMAENn**	**Channel n Accumulator PDMA Enable Bits** 0 = Channel n PDMA function Disabled. 1 = Channel n PDMA function Enabled for the channel n to trigger PDMA to transfer memory data to register.

PWM Capture Input Enable Register (PWM_CAPINEN)

Register	Offset	R/W	Description	Reset Value
PWM_CAPINEN	PWMx_BA+0x200	R/W	PWM Capture Input Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CAPINEN5**	**CAPINEN4**	**CAPINEN3**	**CAPINEN2**	**CAPINEN1**	**CAPINEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPINENn**	**Capture Input Enable Bits** 0 = PWM Channel capture input path Disabled. The input of PWM channel capture function is always regarded as 0. 1 = PWM Channel capture input path Enabled. The input of PWM channel capture function comes from correlative multifunction pin.

PWM Capture Control Register (PWM_CAPCTL)

Register	Offset	R/W	Description	Reset Value
PWM_CAPCTL	PWMx_BA+0x204	R/W	PWM Capture Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		FCRLDEN5	FCRLDEN4	FCRLDEN3	FCRLDEN2	FCRLDEN1	FCRLDEN0
23	22	21	20	19	18	17	16
Reserved		RCRLDEN5	RCRLDEN4	RCRLDEN3	RCRLDEN2	RCRLDEN1	RCRLDEN0
15	14	13	12	11	10	9	8
Reserved		CAPINV5	CAPINV4	CAPINV3	CAPINV2	CAPINV1	CAPINV0
7	6	5	4	3	2	1	0
Reserved		CAPEN5	CAPEN4	CAPEN3	CAPEN2	CAPEN1	CAPEN0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**FCRLDENn**	**Falling Capture Reload Enable Bits** 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**RCRLDENn**	**Rising Capture Reload Enable Bits** 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CAPINVn**	**Capture Inverter Enable Bits** 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPENn**	**Capture Function Enable Bits** 0 = Capture function Disabled. RCAPDAT/FCAPDAT register will not be updated. 1 = Capture function Enabled. Capture latched the PWM counter value when detected rising or falling edge of input signal and saved to RCAPDAT (Rising latch) and FCAPDAT (Falling latch).

PWM Capture Status Register (PWM_CAPSTS)

Register	Offset	R/W	Description	Reset Value
PWM_CAPSTS	PWMx_BA+0x208	R	PWM Capture Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CFLIFOV5**	**CFLIFOV4**	**CFLIFOV3**	**CFLIFOV2**	**CFLIFOV1**	**CFLIFOV0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CRLIFOV5**	**CRLIFOV4**	**CRLIFOV3**	**CRLIFOV2**	**CRLIFOV1**	**CRLIFOV0**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CFLIFOVn**	**Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CFLIF.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CRLIFOVn**	**Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CRLIF.

PWM Rising Capture Data Register 0~5 (PWM_RCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
PWM_RCAPDAT0	PWMx_BA+0x20C	R	PWM Rising Capture Data Register 0	0x0000_0000
PWM_RCAPDAT1	PWMx_BA+0x214	R	PWM Rising Capture Data Register 1	0x0000_0000
PWM_RCAPDAT2	PWMx_BA+0x21C	R	PWM Rising Capture Data Register 2	0x0000_0000
PWM_RCAPDAT3	PWMx_BA+0x224	R	PWM Rising Capture Data Register 3	0x0000_0000
PWM_RCAPDAT4	PWMx_BA+0x22C	R	PWM Rising Capture Data Register 4	0x0000_0000
PWM_RCAPDAT5	PWMx_BA+0x234	R	PWM Rising Capture Data Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RCAPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RCAPDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RCAPDAT**	**PWM Rising Capture Data Register (Read Only)** When rising capture condition happened, the PWM counter value will be saved in this register.

PWM Falling Capture Data Register 0~5 (PWM_FCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
PWM_FCAPDAT0	PWMx_BA+0x210	R	PWM Falling Capture Data Register 0	0x0000_0000
PWM_FCAPDAT1	PWMx_BA+0x218	R	PWM Falling Capture Data Register 1	0x0000_0000
PWM_FCAPDAT2	PWMx_BA+0x220	R	PWM Falling Capture Data Register 2	0x0000_0000
PWM_FCAPDAT3	PWMx_BA+0x228	R	PWM Falling Capture Data Register 3	0x0000_0000
PWM_FCAPDAT4	PWMx_BA+0x230	R	PWM Falling Capture Data Register 4	0x0000_0000
PWM_FCAPDAT5	PWMx_BA+0x238	R	PWM Falling Capture Data Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FCAPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FCAPDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FCAPDAT**	**PWM Falling Capture Data Register (Read Only)** When falling capture condition happened, the PWM counter value will be saved in this register.

PWM PDMA Control Register (PWM_PDMACTL)

Register	Offset	R/W	Description	Reset Value
PWM_PDMACTL	PWMx_BA+0x23C	R/W	PWM PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			CHSEL4_5	CAPORD4_5	CAPMOD4_5		CHEN4_5
15	14	13	12	11	10	9	8
Reserved			CHSEL2_3	CAPORD2_3	CAPMOD2_3		CHEN2_3
7	6	5	4	3	2	1	0
Reserved			CHSEL0_1	CAPORD0_1	CAPMOD0_1		CHEN0_1

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**CHSEL4_5**	**Select Channel 4/5 to Do PDMA Transfer** 0 = Channel4. 1 = Channel5. **Note:** If the PDMA function is not supported, this bit field will become invalid.
[19]	**CAPORD4_5**	**Capture Channel 4/5 Rising/Falling Order** Set this bit to determine whether the PWM_RCAPDAT4/5 or PWM_FCAPDAT4/5 is the first captured data transferred to memory through PDMA when CAPMOD4_5 =11. 0 = PWM_FCAPDAT4/5 is the first captured data to memory. 1 = PWM_RCAPDAT4/5 is the first captured data to memory. **Note:** If the PDMA function is not supported, this bit field will become invalid.
[18:17]	**CAPMOD4_5**	**Select PWM_RCAPDAT4/5 or PWM_FCAPDAT4/5 to Do PDMA Transfer** 00 = Reserved. 01 = PWM_RCAPDAT4/5. 10 = PWM_FCAPDAT4/5. 11 = Both PWM_RCAPDAT4/5 and PWM_FCAPDAT4/5. **Note:** If the PDMA function is not supported, this bit field will become invalid.
[16]	**CHEN4_5**	**Channel 4/5 PDMA Enable Bit** 0 = Channel 4/5 PDMA function Disabled. 1 = Channel 4/5 PDMA function Enabled for the channel 4/5 captured data and transfer to memory. **Note:** If the PDMA function is not supported, this bit field will become invalid.
[15:13]	**Reserved**	Reserved.
[12]	**CHSEL2_3**	**Select Channel 2/3 to Do PDMA Transfer** 0 = Channel2. 1 = Channel3.

		Note: If the PDMA function is not supported, this bit field will become invalid.
[11]	**CAPORD2_3**	**Capture Channel 2/3 Rising/Falling Order** Set this bit to determine whether the PWM_RCAPDAT2/3 or PWM_FCAPDAT2/3 is the first captured data transferred to memory through PDMA when CAPMOD2_3 =11. 0 = PWM_FCAPDAT2/3 is the first captured data to memory. 1 = PWM_RCAPDAT2/3 is the first captured data to memory. **Note:** If the PDMA function is not supported, this bit field will become invalid.
[10:9]	**CAPMOD2_3**	**Select PWM_RCAPDAT2/3 or PWM_FCAODAT2/3 to Do PDMA Transfer** 00 = Reserved. 01 = PWM_RCAPDAT2/3. 10 = PWM_FCAPDAT2/3. 11 = Both PWM_RCAPDAT2/3 and PWM_FCAPDAT2/3. **Note:** If the PDMA function is not supported, this bit field will become invalid.
[8]	**CHEN2_3**	**Channel 2/3 PDMA Enable Bit** 0 = Channel 2/3 PDMA function Disabled. 1 = Channel 2/3 PDMA function Enabled for the channel 2/3 captured data and transfer to memory. **Note:** If the PDMA function is not supported, this bit field will become invalid.
[7:5]	**Reserved**	Reserved.
[4]	**CHSEL0_1**	**Select Channel 0/1 to Do PDMA Transfer** 0 = Channel0. 1 = Channel1. **Note:** If the PDMA function is not supported, this bit field will become invalid.
[3]	**CAPORD0_1**	**Capture Channel 0/1 Rising/Falling Order** Set this bit to determine whether the PWM_RCAPDAT0/1 or PWM_FCAPDAT0/1 is the first captured data transferred to memory through PDMA when CAPMOD0_1 =11. 0 = PWM_FCAPDAT0/1 is the first captured data to memory. 1 = PWM_RCAPDAT0/1 is the first captured data to memory. **Note:** If the PDMA function is not supported, this bit field will become invalid.
[2:1]	**CAPMOD0_1**	**Select PWM_RCAPDAT0/1 or PWM_FCAPDAT0/1 to Do PDMA Transfer** 00 = Reserved. 01 = PWM_RCAPDAT0/1. 10 = PWM_FCAPDAT0/1. 11 = Both PWM_RCAPDAT0/1 and PWM_FCAPDAT0/1. **Note:** If the PDMA function is not supported, this bit field will become invalid.
[0]	**CHEN0_1**	**Channel 0/1 PDMA Enable Bit** 0 = Channel 0/1 PDMA function Disabled. 1 = Channel 0/1 PDMA function Enabled for the channel 0/1 captured data and transfer to memory. **Note:** If the PDMA function is not supported, this bit field will become invalid.

PWM Capture Channel 0_1, 2_3, 4_5 PDMA Register (PWM_PDMACAP 0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_PDMACAP0_1	PWMx_BA+0x240	R	PWM Capture Channel 01 PDMA Register	0x0000_0000
PWM_PDMACAP2_3	PWMx_BA+0x244	R	PWM Capture Channel 23 PDMA Register	0x0000_0000
PWM_PDMACAP4_5	PWMx_BA+0x248	R	PWM Capture Channel 45 PDMA Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CAPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CAPBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CAPBUF**	**PWM Capture PDMA Register (Read Only)** This register is used as a buffer to transfer PWM capture rising or falling data to memory by PDMA. **Note:** If the PDMA function is not supported, this bit field will become invalid.

PWM Capture Interrupt Enable Register (PWM_CAPIEN)

Register	Offset	R/W	Description	Reset Value
PWM_CAPIEN	PWMx_BA+0x250	R/W	PWM Capture Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CAPFIEN5**	**CAPFIEN4**	**CAPFIEN3**	**CAPFIEN2**	**CAPFIEN1**	**CAPFIEN0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CAPRIEN5**	**CAPRIEN4**	**CAPRIEN3**	**CAPRIEN2**	**CAPRIEN1**	**CAPRIEN0**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CAPFIENn**	**PWM Capture Falling Latch Interrupt Enable Bits** 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPRIENn**	**PWM Capture Rising Latch Interrupt Enable Bits** 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled.

PWM Capture Interrupt Flag Register (PWM_CAPIF)

Register	Offset	R/W	Description	Reset Value
PWM_CAPIF	PWMx_BA+0x254	R/W	PWM Capture Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CFLIF5**	**CFLIF4**	**CFLIF3**	**CFLIF2**	**CFLIF1**	**CFLIF0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CRLIF5**	**CRLIF4**	**CRLIF3**	**CRLIF2**	**CRLIF1**	**CRLIF0**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CFLIFn**	**PWM Capture Falling Latch Interrupt Flag** 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high. **Note 1:** When Capture with PDMA operating, CAPIF corresponding channel CFLIF will be cleared by hardware after PDMA transfer data. **Note 2:** This bit is cleared by writing 1 to it.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CRLIFn**	**PWM Capture Rising Latch Interrupt Flag** 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high. **Note 1:** When Capture with PDMA operating, CAPIF corresponding channel CRLIF will be cleared by hardware after PDMA transfer data. **Note 2:** This bit is cleared by writing 1 to it.

PWM Capture Input Noise Filter Register (PWM_CAPNF0~5)

Register	Offset	R/W	Description	Reset Value
PWM_CAPNF0	PWMx_BA+0x258	R/W	PWM Capture Input Noise Filter Register 0	0x0000_0000
PWM_CAPNF1	PWMx_BA+0x25C	R/W	PWM Capture Input Noise Filter Register 1	0x0000_0000
PWM_CAPNF2	PWMx_BA+0x260	R/W	PWM Capture Input Noise Filter Register 2	0x0000_0000
PWM_CAPNF3	PWMx_BA+0x264	R/W	PWM Capture Input Noise Filter Register 3	0x0000_0000
PWM_CAPNF4	PWMx_BA+0x268	R/W	PWM Capture Input Noise Filter Register 4	0x0000_0000
PWM_CAPNF5	PWMx_BA+0x26C	R/W	PWM Capture Input Noise Filter Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					CAPNFCNT		
7	6	5	4	3	2	1	0
Reserved	CAPNFSEL			Reserved			CAPNFEN

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**CAPNFCNT**	**Capture Edge Detector Noise Filter Count** The register bits control the capture filter counter to count from 0 to CAPNFCNT.
[7]	**Reserved**	Reserved.
[6:4]	**CAPNFSEL**	**Capture Edge Detector Noise Filter Clock Selection** 000 = Filter clock = PCLK. 001 = Filter clock = PCLK/2. 010 = Filter clock = PCLK/4. 011 = Filter clock = PCLK/8. 100 = Filter clock = PCLK/16. 101 = Filter clock = PCLK/32. 110 = Filter clock = PCLK/64. 111 = Filter clock = PCLK/128.
[3:1]	**Reserved**	Reserved.
[0]	**CAPNFEN**	**Capture Noise Filter Enable Bit** 0 = Capture Noise Filter function Disabled. 1 = Capture Noise Filter function Enabled.

PWM External Event Trigger Control Register 0~5 (PWMx_EXTETCTL0~5)

Register	Offset	R/W	Description	Reset Value
PWM_EXTETCTL0	PWMx_BA+0x270	R/W	PWM External Event Trigger Control Register 0	0x0000_0000
PWM_EXTETCTL2	PWMx_BA+0x278	R/W	PWM External Event Trigger Control Register 2	0x0000_0000
PWM_EXTETCTL4	PWMx_BA+0x280	R/W	PWM External Event Trigger Control Register 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				EXTTRGS			
7	6	5	4	3	2	1	0
Reserved		CNTACTS		Reserved			EXTETEN

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**EXTTRGS**	**External Trigger Selection** 0000 = INT0. 0001 = INT1. 0010 = INT2. 0011 = INT3. 0100 = INT4. 0101 = INT5. 0110 = INT6. 0111 = INT7. Other = Resrved.
[7:6]	**Reserved**	Reserved.
[5:4]	**CNTACTS**	**Counter Action Selection** 00 = Counter reset. 01 = Counter start. 10 = Counter reset and start. 11 = Reseved.
[3:1]	**Reserved**	Reserved.
[0]	**EXTETEN**	**External Event Trigger Enable Bit** 0 = External Event Trigger function Disabled. 1 = External Event Trigger function Enabled.

PWM Software Event Output Force Control Register (PWMx_SWEOFCTL)

Register	Offset	R/W	Description	Reset Value
PWM_SWEOFCTL	PWMx_BA+0x288	R/W	PWM Software Event Output Force Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**OUTACTS5**		**OUTACTS4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OUTACTS3		**OUTACTS2**		**OUTACTS1**		**OUTACTS0**	

Bits	Description	
[31:12]	**Reserved**	Reserved.
[2n+1:2n] n=0,1..5	**OUTACTSn**	**Output Action Selection** 00 = Do nothing. 01 = PWM output Low. 10 = PWM output High. 11 = PWM output Toggle.

PWM Software Event Output Force Trigger Register (PWMx_SWEOFTRG)

Register	Offset	R/W	Description	Reset Value
PWM_SWEOFTRG	PWMx_BA+0x28C	R/W	PWM Software Event Output Force Trigger Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**SWETRG5**	**SWETRG4**	**SWETRG3**	**SWETRG2**	**SWETRG1**	**SWETRG0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**SWETRGn**	**Software Event Trigger** Write 1 to this bit will change PWM output status according to OUTACTSn in PWMx_SWEOFCTL setting. **Note:** This bit will auto cleared by hardware.

PWM Period Register Buffer 0, 2, 4 (PWM_PBUF0, 2, 4)

Register	Offset	R/W	Description	Reset Value
PWM_PBUF0	PWMx_BA+0x304	R	PWM PERIOD0 Buffer	0x0000_0000
PWM_PBUF2	PWMx_BA+0x30C	R	PWM PERIOD2 Buffer	0x0000_0000
PWM_PBUF4	PWMx_BA+0x314	R	PWM PERIOD4 Buffer	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PBUF**	**PWM Period Register Buffer (Read Only)** Used as PERIOD active register.

PWM Comparator Register Buffer 0~5 (PWM_CMPBUF0~5)

Register	Offset	R/W	Description	Reset Value
PWM_CMPBUF0	PWMx_BA+0x31C	R	PWM CMPDAT0 Buffer	0x0000_0000
PWM_CMPBUF1	PWMx_BA+0x320	R	PWM CMPDAT1 Buffer	0x0000_0000
PWM_CMPBUF2	PWMx_BA+0x324	R	PWM CMPDAT2 Buffer	0x0000_0000
PWM_CMPBUF3	PWMx_BA+0x328	R	PWM CMPDAT3 Buffer	0x0000_0000
PWM_CMPBUF4	PWMx_BA+0x32C	R	PWM CMPDAT4 Buffer	0x0000_0000
PWM_CMPBUF5	PWMx_BA+0x330	R	PWM CMPDAT5 Buffer	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPBUF**	**PWM Comparator Register Buffer (Read Only)** Used as CMP active register.

6.15 EPWM Generator and Capture Timer (EPWM)

6.15.1 Overview

The chip provides two EPWM generators — EPWM0 and EPWM1. Each EPWM supports 6 channels of EPWM output or input capture. There is a 12-bit prescaler to support flexible clock to the 16-bit EPWM counter with 16-bit comparator. The EPWM counter supports up, down and up-down counter types. EPWM uses comparator compared with counter to generate events. These events are used to generate EPWM pulse, interrupt and trigger signal for EADC, LPADC and DAC to start conversion.

The EPWM generator supports two standard EPWM output modes: Independent mode and Complementary mode, they have difference architecture. There are two output functions based on standard output modes: Group function and Synchronous function. Group function can be enabled under Independent mode or complementary mode. Synchronous function only enabled under complementary mode. Complementary mode has two comparators to generate various EPWM pulse with 12-bit dead-time generator and another free trigger comparator to generate trigger signal for EADC/LPADC. For EPWM output control unit, it supports polarity output, independent pin mask and brake functions.

The EPWM generator also supports input capture function. It supports latch EPWM counter value to corresponding register when input channel has a rising transition, falling transition or both transition is happened. Capture function also supports PDMA to transfer captured data to memory.

6.15.2 Features

6.15.2.1 EPWM Function Features

- Supports maximum clock frequency up to maximum PLL frequency
- Supports up to two EPWM modules, each module provides 6 output channels
- Supports independent mode for EPWM output/Capture input channel
- Supports complementary mode for 3 complementary paired EPWM output channel
  - Dead-time insertion independent control with 12-bit resolution
  - Synchronous function for phase control
  - Two compared values during one period
- Supports 12-bit prescaler from 1 to 4096
- Supports 16-bit resolution EPWM counter
  - Up, down and up/down counter operation type
- Supports one-shot or auto-reload counter operation mode
- Supports group function
- Supports synchronous function
- Supports mask function and tri-state enable for each EPWM pin
- Supports brake function
  - Brake source from pin, analog comparator, UTCPD, EADC0 result monitor and system safety events (clock failed, SRAM parity error, Brown-out detection and CPU lockup).
  - Noise filter for brake source from pin
  - Leading edge blanking (LEB) function for brake source from analog comparator
  - Edge detect brake source to control brake state until brake interrupt cleared
  - Level detect brake source to auto recover function after brake condition removed

- Supports interrupt on the following events:
  - EPWM counter matches 0, period value or compared value
  - Brake condition happened
- Supports trigger EADC/LPADC/DAC on the following events:
  - EPWM counter matches 0, period value or compared value
  - EPWM counter matches free trigger comparator compared value (only for EADC/LPADC)
  - Supports EPWM trigger EADC/LPADC event prescaler feature
- Supports PDMA transfer for Interrupt Flag Accumulator Function
- Supports EPWM output accumulator stop counter mode
- Supports Fault Detect function
- Supports External Pin Trigger Function

6.15.2.2 Capture Function Features

- Supports 3-bit capture input noise filter
- Supports up to 12 capture input channels with 16-bit resolution
- Supports rising or falling capture condition
- Supports input rising/falling capture interrupt
- Supports rising/falling capture with counter reload option
- Supports PDMA transfer function for EPWM all channels

6.15.3 Block Diagram

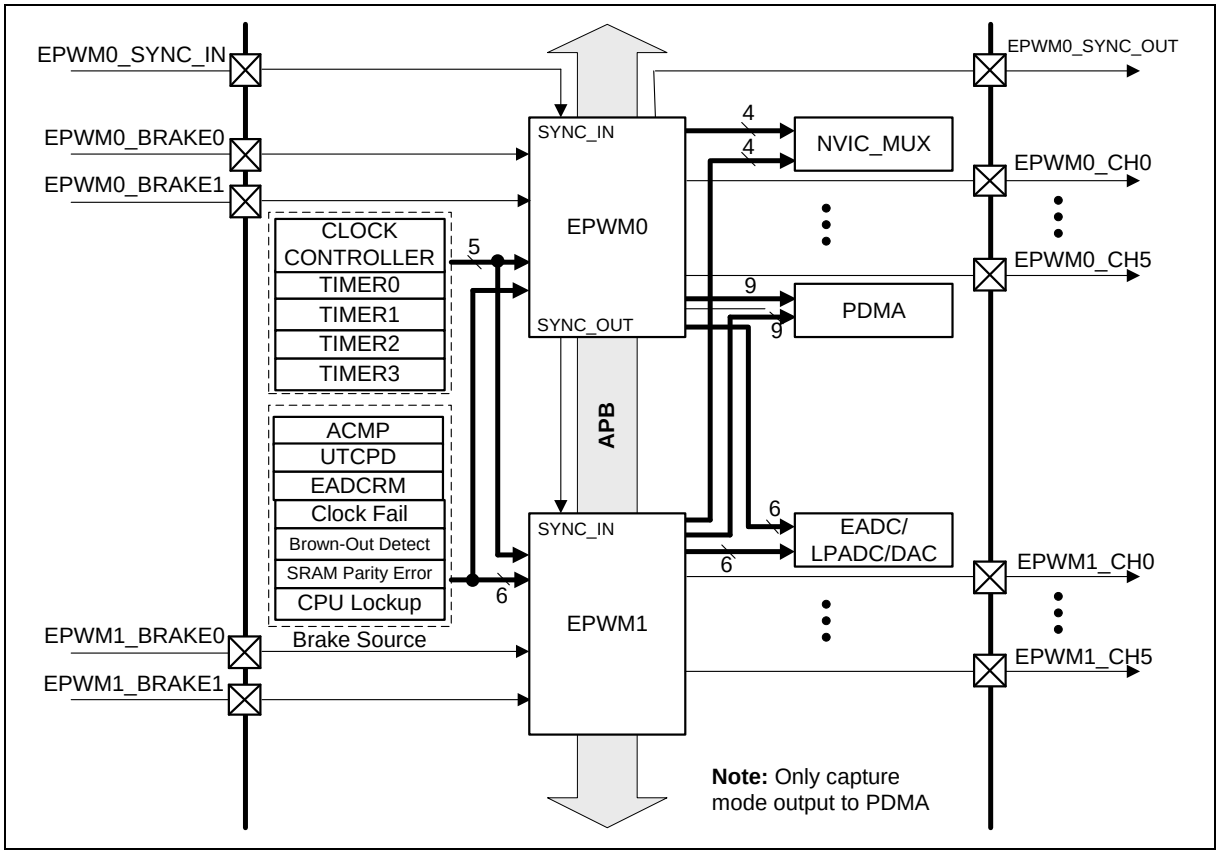


Figure 6.15-1 EPWM Generator Overview Block Diagram

EPWM Clock frequency can be set equal or double to PCLK frequency as Figure 6.15-2. For the detailed register setting, please refer to Table 6.15-1. Each EPWM generator has three clock source inputs, each clock source can be selected from EPWM Clock or four TIMER trigger EPWM outputs as Figure 6.15-3 by ECLKSRC0 (EPWM_CLKSRC[2:0]) for EPWM_CLK0, ECLKSRC2 (EPWM_CLKSRC[10:8]) for EPWM_CLK2 and ECLKSRC4 (EPWM_CLKSRC[18:16]) for EPWM_CLK4.

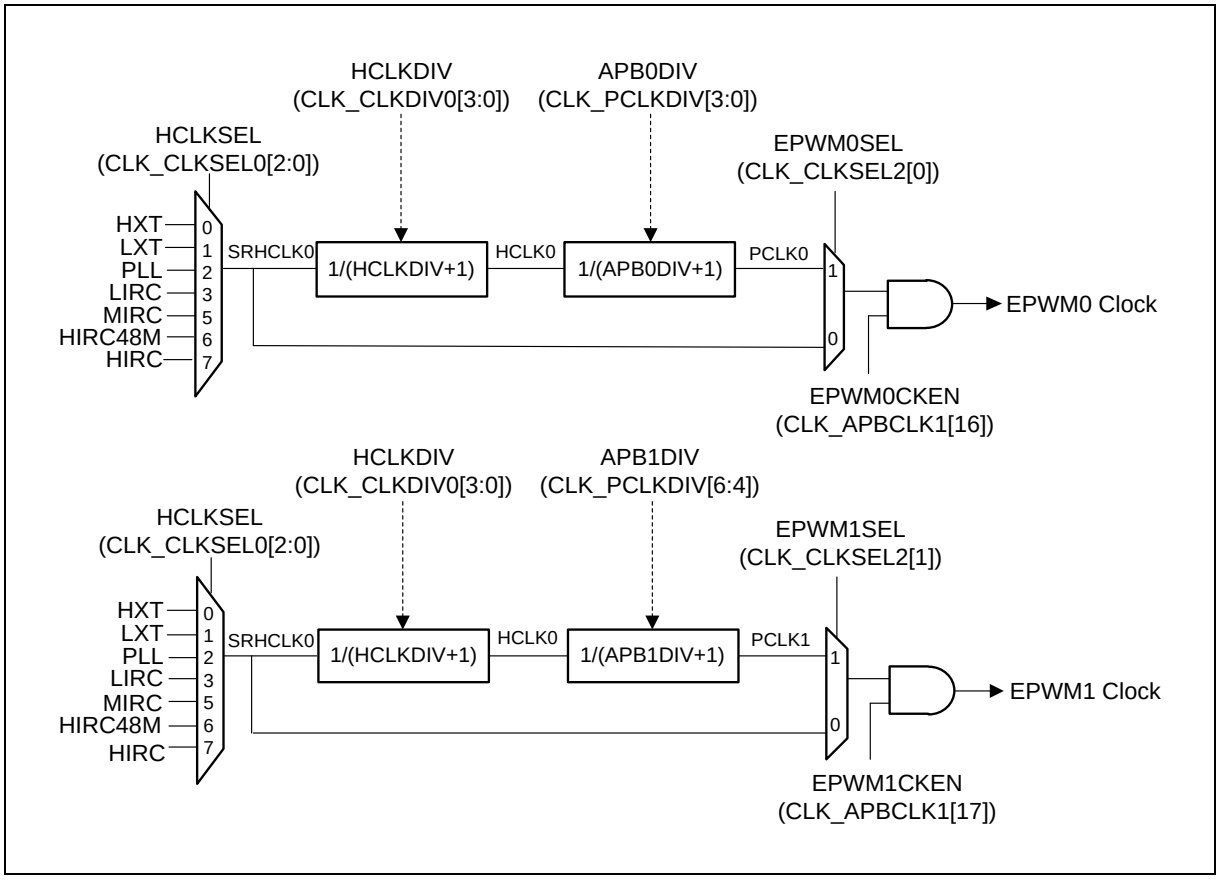


Figure 6.15-2 EPWM Clock Source Control

Frequency Ratio PCLK:EPWM Clock	HCLK	PCLK	EPWM Clock	HCLK0SEL CLK_CLKSEL0[2:0]	HCLKDIV CLK_CLKDIV0[3:0]	APBnDIV (CLK_PCLKDIV [2+4n:4n]), n denotes 0 or 1	EPWMnSEL (CLK_CLKSEL2[n]) , n denotes 0 or 1
1:1	HCLK	PCLK	PCLK	Don't care	Don't care	Don't care	1
1:2	PLL	PLL/ 2	SRHCLK 0	2	0	1	0
1:2	PLL/ 2	PLL/ 2	SRHCLK 0	2	1	0	0

Table 6.15-1 EPWM Clock Source Control Registers Setting Table

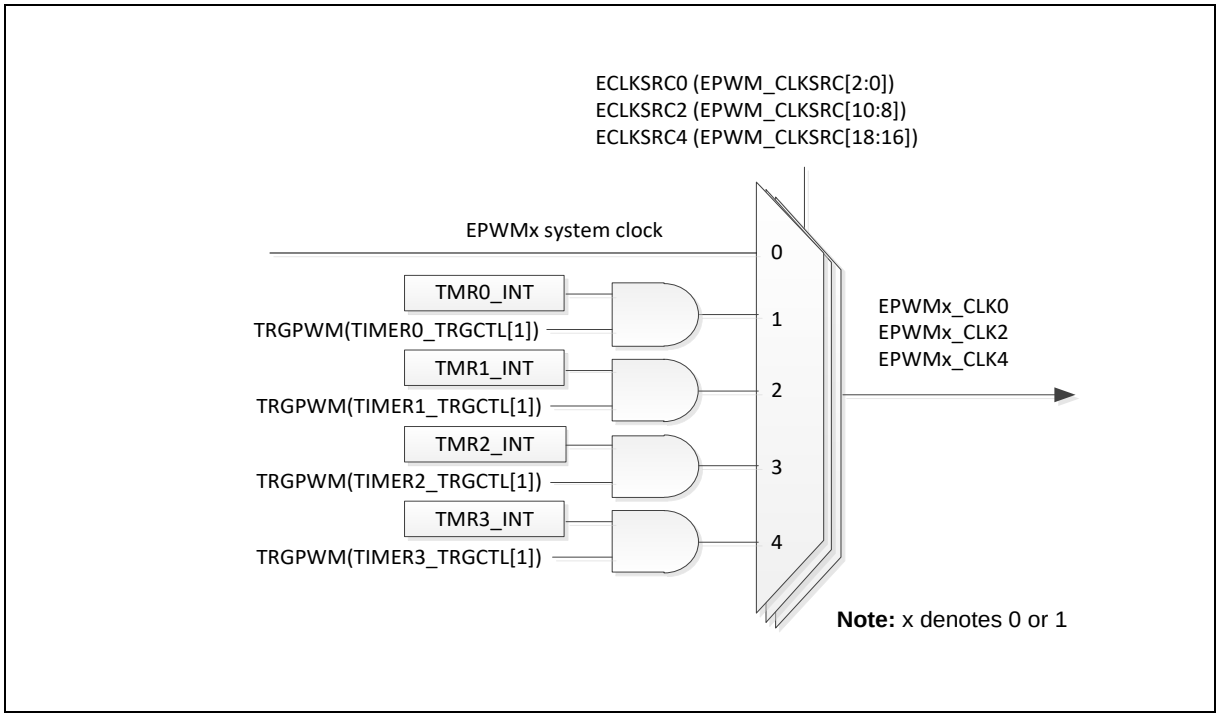


Figure 6.15-3 EPWM Clock Source Control

Figure 6.15-4 and Figure 6.15-5 illustrate the architecture of EPWM independent mode and complementary mode. In complementary mode, paired channels' (EPWM_CH0 and EPWM_CH1, EPWM_CH2 and EPWM_CH3, EPWM_CH4 and EPWM_CH5) counters both come from the same clock source and prescaler. In independent mode, paired channel's counters come from the same clock source, but each channel has its own prescaler. When counter count to 0, PERIOD (EPWM_PERIODn[15:0]) or equal to comparator, events will be generated. These events are passed to corresponding generators to generate EPWM pulse, interrupt signal and trigger signal for EADC/LPADC/DAC to start conversion. Output control is used to change EPWM pulse output state; brake function in output control also generates interrupt events. In complementary mode, synchronize function is available and even channel use odd channel comparator to generate events, free trigger comparator events only use to generate trigger EADC/LPADC signals.

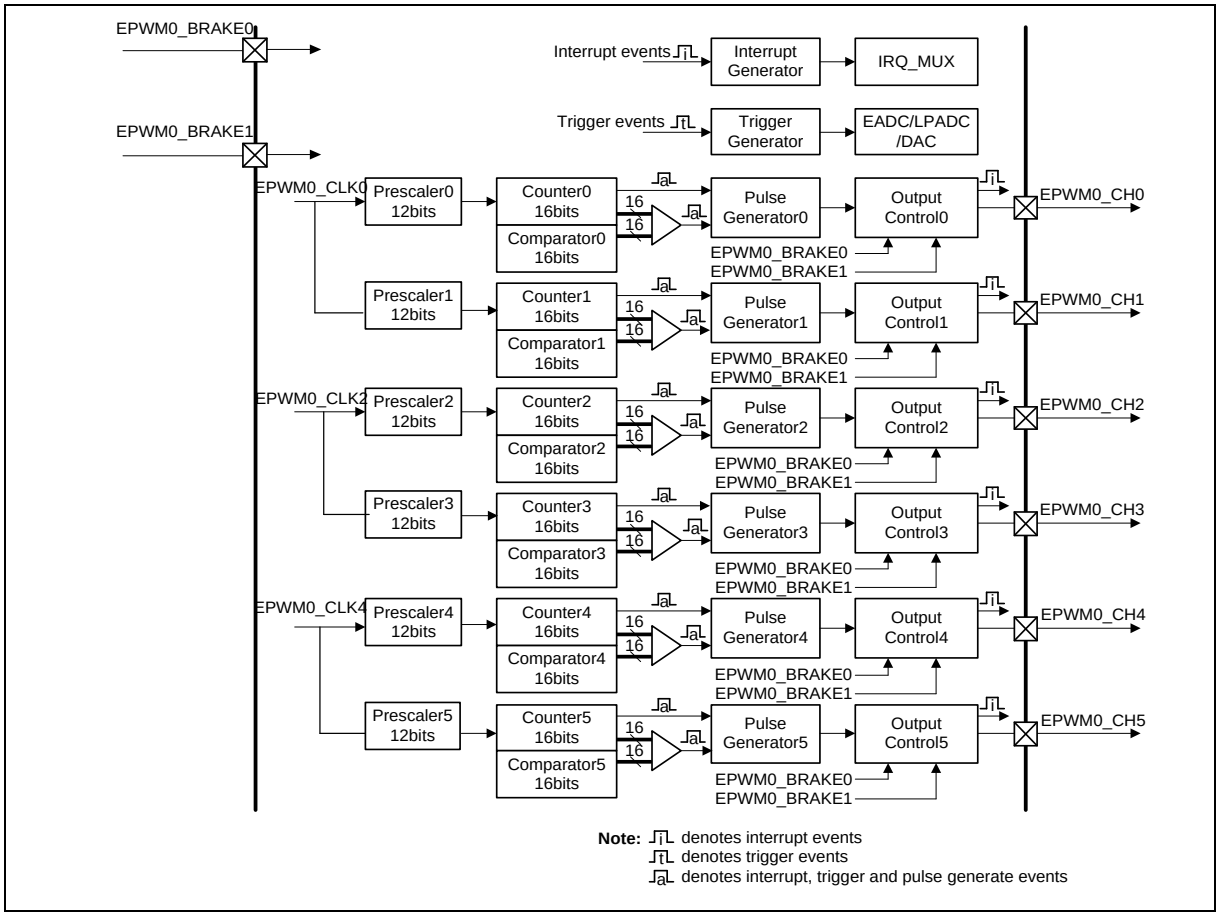


Figure 6.15-4 EPWM Independent Mode Architecture Diagram

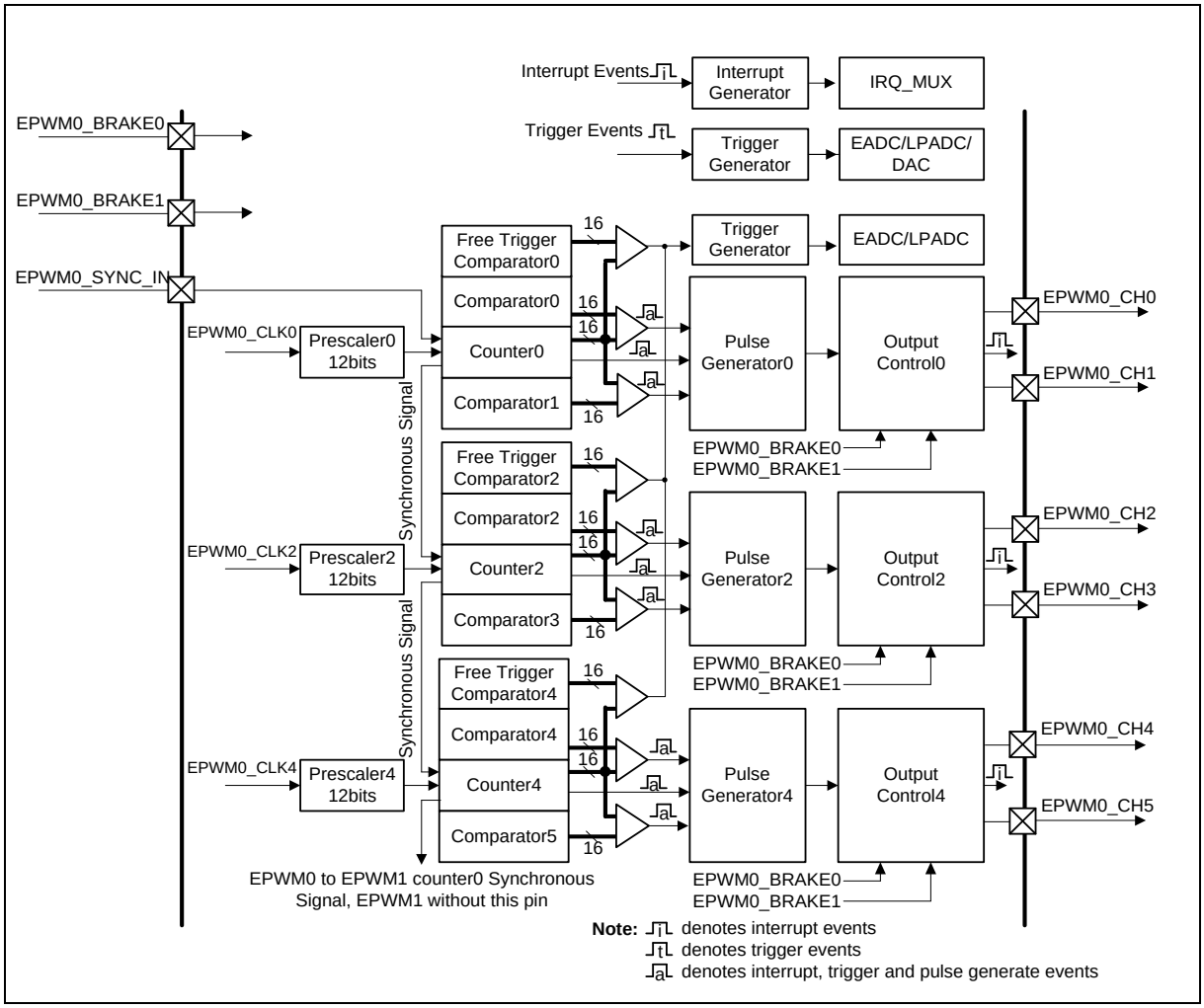


Figure 6.15-5 EPWM Complementary Mode Architecture Diagram

6.15.4 Basic Configuration

6.15.4.1 EPWM0 Basic Configuration

- Clock Source Configuration
  - Select the source of EPWM0 peripheral clock on EPWM0SEL (CLK_CLKSEL2[0])
  - Enable EPWM0 peripheral clock in EPWM0CKEN (CLK_APBCLK1[16]).
- Reset Configuration
  - Reset EPWM0 in EPWM0RST SYS_IPRST2[16]
-

6.15.4.2 EPWM1 Basic Configuration

- Clock Source Configuration
  - Select the source of EPWM1 peripheral clock on EPWM1SEL (CLK_CLKSEL2[1])
  - Enable EPWM1 peripheral clock in EPWM1CKEN (CLK_APBCLK1[17]).
- Reset Configuration
  - Reset EPWM1 in EPWM1RST (SYS_IPRST2[17])

6.15.5 Functional Description

6.15.5.1 EPWM Prescaler

The EPWM prescaler is used to divide clock source, prescaler counting CLKPSC+1 times, EPWM counter only count once. The prescale double buffer is setting by CLKPSC (EPWM_CLKPSCn[11:0], n = 0,1..5) bits. Figure 6.15-6 is an example of EPWM channel 0 prescale waveform. The prescale counter will reload CLKPSC at the begin of the next prescale counter down-count.

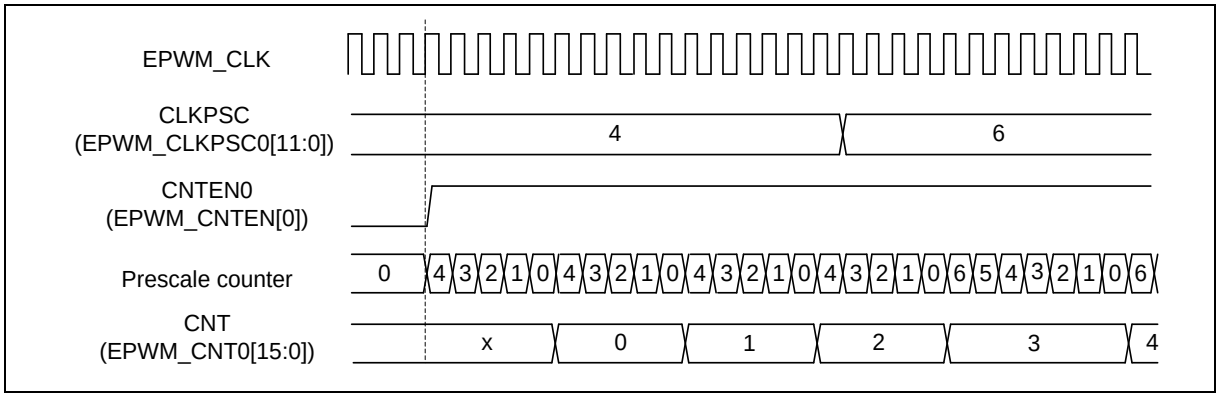


Figure 6.15-6 EPWM_CH0 Prescaler Waveform in Up Counter Type

6.15.5.2 EPWM Counter

The EPWM supports 3 counter types operation: Up Counter, Down Counter and Up-Down Counter types.

For EPWM channel0, CNT(EPWM_CNT0[15:0]) can clear to 0x00 by CNTCLR0 (EPWM_CNTCLR[0]). CNT will be cleared when prescale counter count to 0, and CNTCLR0 will be set 0 by hardware automatically.

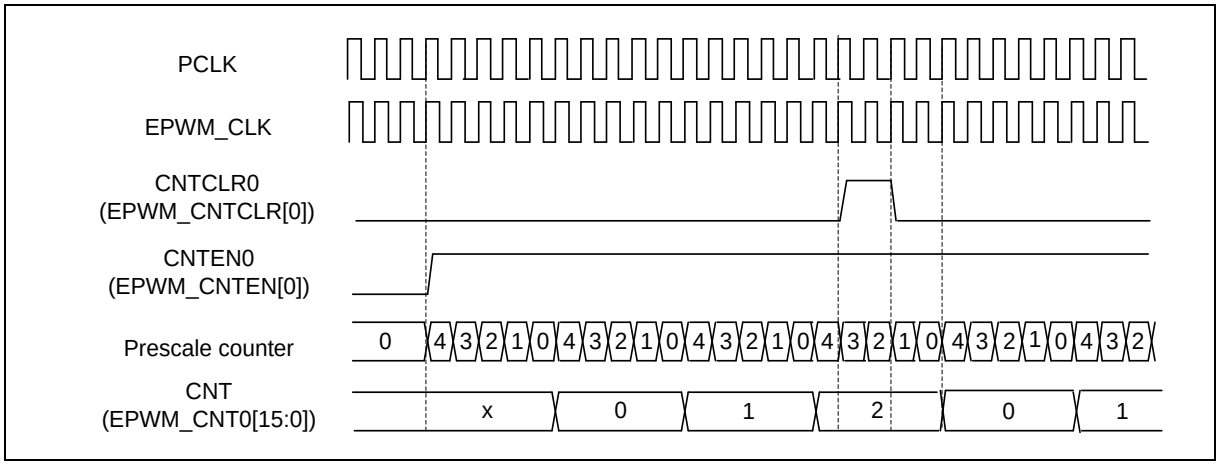


Figure 6.15-7 EPWM Counter Waveform when Setting Clear Counter

6.15.5.3 Up Counter Type

When EPWM counter is set to up counter type, CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) is 0x0, it starts up-counting from 0 to PERIOD (EPWM_PERIODn[15:0], where n denotes channel number) to complete a EPWM period. The current counter value can be read from CNT (EPWM_CNTn[15:0]) bits. EPWM generates zero point event when the counter counts to 0 and prescale counts to 0. EPWM generates period point event when the counter counts to PERIOD and prescale counts to 0. Figure

6.15-8 shows an example of up counter, wherein

EPWM period time = (PERIOD+1) *(CLKPSC+1)* EPWMx_CLK.

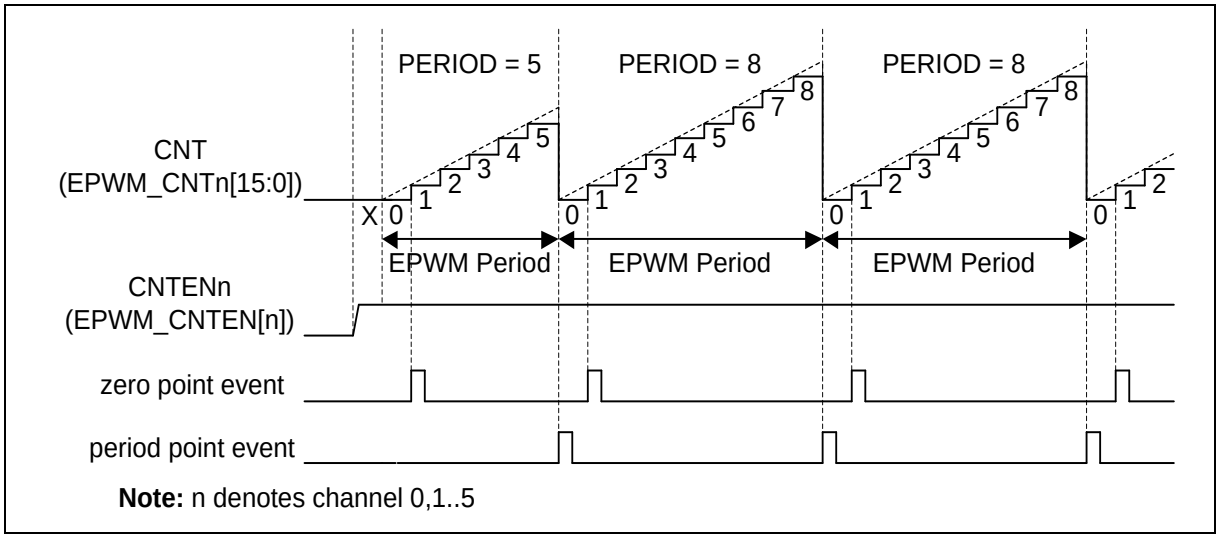


Figure 6.15-8 EPWM Up Counter Type

6.15.5.4 Down Counter Type

When EPWM counter is set to down counter type, CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) is 0x1, it starts down-counting from PERIOD to 0 to complete a EPWM period. The current counter value can be read from CNT (EPWM_CNTn[15:0]) bits. EPWM generates zero point event when the counter counts to 0 and prescale counts to 0. EPWM generates period point event when the counter counts to PERIOD and prescale counts to 0. Figure 6.14-9 shows an example of down counter, wherein

EPWM period time = (PERIOD+1) * (CLKPSC+1) * EPWMx_CLK.

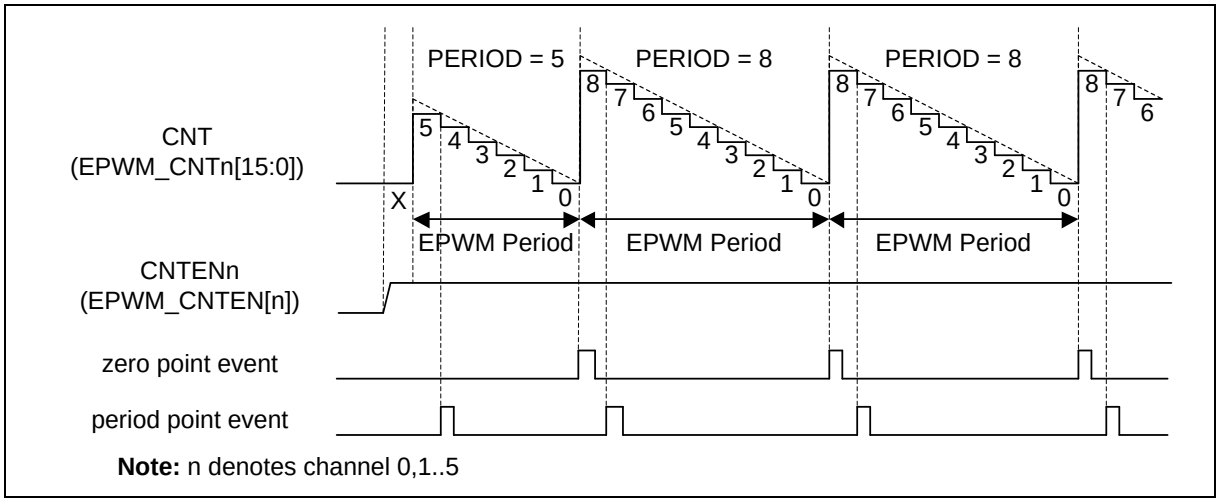


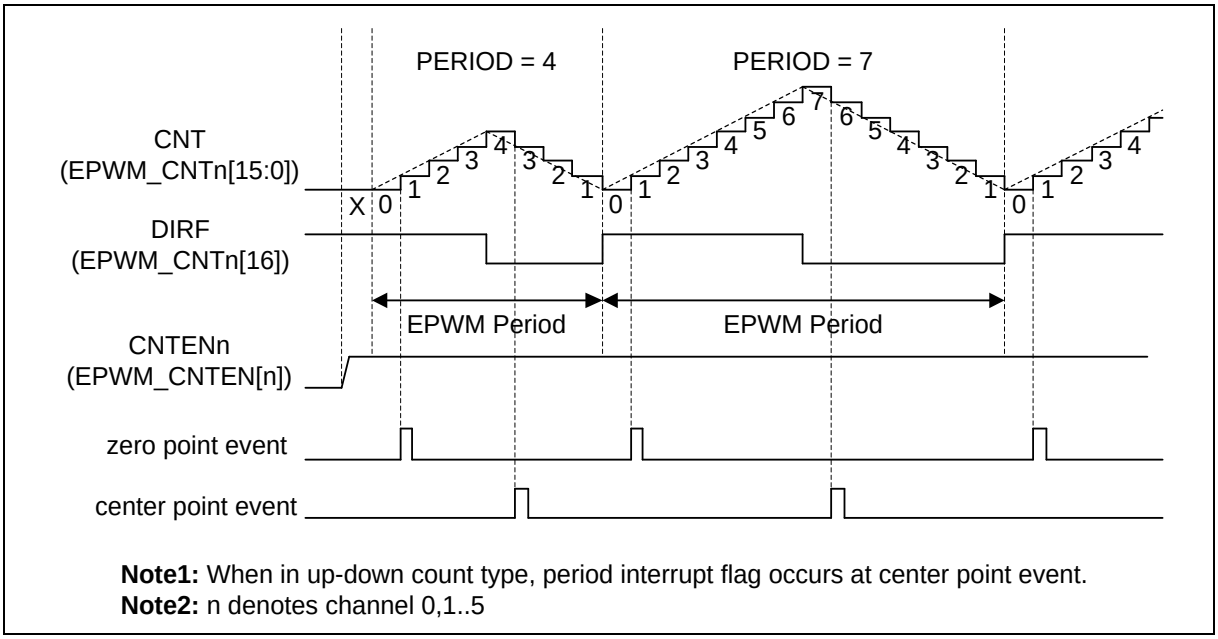
Figure 6.15-9 EPWM Down Counter Type

6.15.5.5 Up-Down Counter Type

When EPWM counter is set to up-down count type, CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) is 0x2, it starts counting-up from 0 to PERIOD and then starts counting down to 0 to complete a EPWM period. The current counter value can be read from CNT (EPWM_CNTn[15:0]) bits. EPWM generates zero point event when the counter counts to 0 and prescale counts to 0. EPWM generates center point event which is equal to period point event when the counter counts to PERIOD. Figure 6.14-10 shows an example of up-down counter, wherein

EPWM period time =(2*PERIOD) * (CLKPSC+1) * EPWMx_CLK.

The DIRF (EPWM_CNTn[16]) bit is counter direction indicator flag, where high is up counting, and low is down counting.

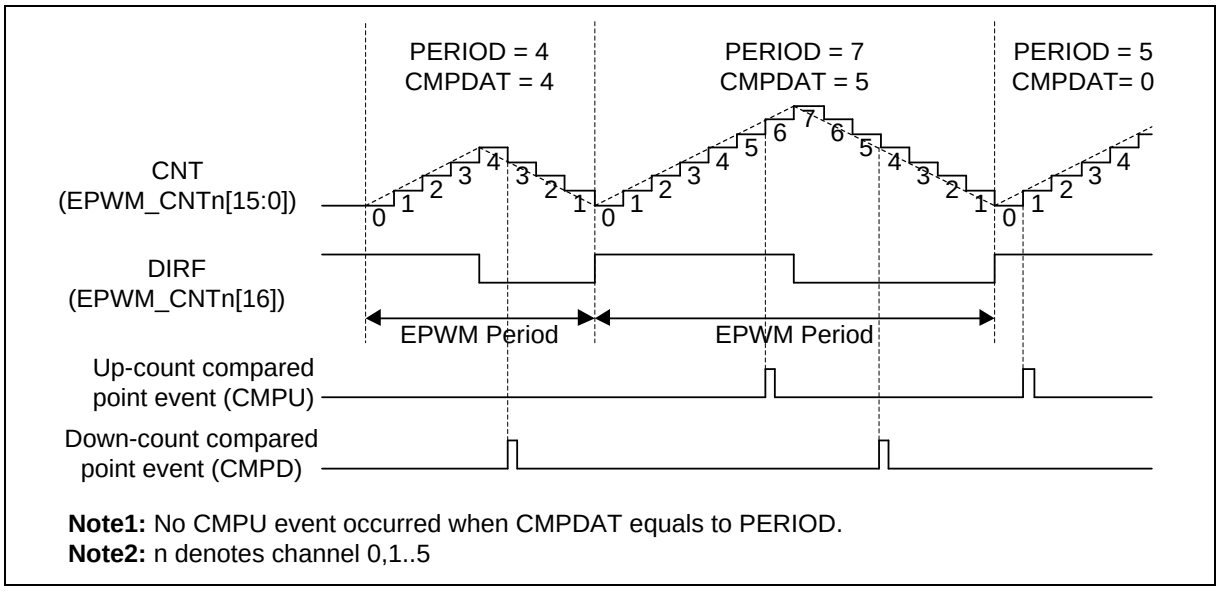


Note1: When in up-down count type, period interrupt flag occurs at center point event.
Note2: n denotes channel 0,1..5

Figure 6.15-10 EPWM Up-Down Counter Type

6.15.5.6 EPWM Comparator

There are two kinds of comparator registers : one is EPWM_CMPDATn(n = 0,1..5), and the other is EPWM_FTCMPDATn_m(n = 0,2,4, m = 1,3,5) register. EPWM_CMPDATn is a basic comparator register of EPWM channel n; In Independent mode each channel only has one comparator, the value of EPWM_CMPDATn register is continuously compared to the corresponding channel's counter value. In Complementary mode each paired channels has two comparators, and the value of EPWM_CMPDATn and EPWM_CMPDATm (n = 0,2,4, m = 1,3,5) registers are continuously compared to the complementary even channel's counter value, because of odd channel's counter is useless. For example, channel 0 and channel 1 are complementary channels, in Complementary mode, channel 1's comparator is continuously compared to channel 0's counter, but not channel 1's. When the counter is equal to value of EPWM_CMPDAT0 register, EPWM generates a compared point event and uses the event to generate EPWM pulse, interrupt or use to trigger EADC/LPADC/DAC. In up-down counter type, two events will be generated in a EPWM period as shown in Figure 6.14-11. The CMPU is up count compared point event and CMPD is down count compared point event.



Note1: No CMPU event occurred when CMPDAT equals to PERIOD.
Note2: n denotes channel 0,1..5

Figure 6.15-11 EPWM Compared point Events in Up-Down Counter Type

EPWM_FTCMPDATn_m is a free trigger comparator register. Each complementary paired channel only supports one free trigger comparator. The value of EPWM_FTCMPDATn_m(n = 0,2,4, m = 1,3,5) register is continuously compared to even channel's counter value. When counter is equal to the value of EPWM_FTCMPDATn_m register, FTCMDn (EPWM_FTCI[8+n/2], n = 0,2,4) indicator is set in down count type and FTCMUn (EPWM_FTCI[n/2], n = 0,2,4) indicator is set in up count type. In addition, EPWM generates an event and only uses to trigger EADC/LPADC.

6.15.5.7 EPWM Double Buffering

The double buffering uses double buffers to separate software writing and hardware action operation timing. There are four loading modes for loading values to buffer: period loading mode, immediately loading mode, window loading mode and center loading mode. After registers are modified through software, hardware will load register value to the buffer register according to the loading mode timing. The hardware action is based on the buffer value. This can prevent asynchronously operation problem due to software and hardware asynchronism.

The EPWM provides PBUF (EPWM_PBUFn[15:0]) as the active EPWM_PERIODn buffer register, CMPBUF (EPWM_CMPBUFn[15:0]) as the active EPWM_CMPDATn buffer register, FTCMPBUF (EPWM_FTCBUFn_m[15:0]) as the active EPWM_FTCMPDATn_m buffer register and CPSCBUF (EPWM_CPSCBUFn[11:0]) as the active EPWM_CLKPSCn buffer register. The concept of double buffering is used in loading modes, which are described in the following sections. For example, as shown Figure 6.14-12, in period loading mode, writing PERIOD(EPWM_PERIODn[15:0]), CMP(EPWM_CMPDATn[15:0]) and FTCMP(EPWM_FTCMPDATn_m[15:0]) buffer through software, EPWM will load new values to their buffer PBUF(EPWM_PBUFn[15:0]), CMPBUF (EPWM_CMPBUFn[15:0]) and FTCMPBUF (EPWM_FTCBUFn_m[15:0]) at start of the next period without affecting the current period counter operation. FTCMPU denotes up-count free trigger compared point event and FTCMPD denotes down-count free trigger compared event.

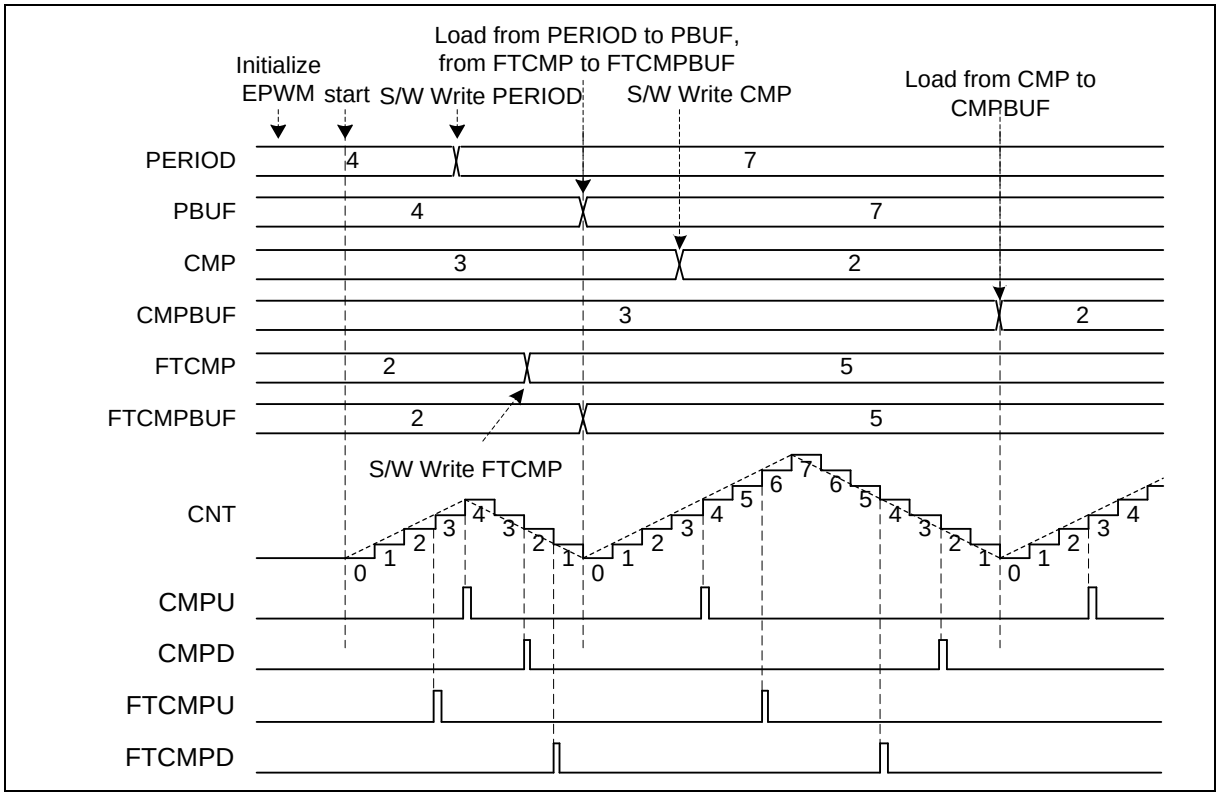


Figure 6.15-12 EPWM Double Buffering Illustration

6.15.5.8 Period Loading Mode

When immediately loading mode, window loading mode and center loading mode are disabled that IMMLDENn bits, WINLDENn bits and CTRLDn bits of EPWM_CTL0 register are set to 0, EPWM operates at period Loading mode. In period Loading mode, CLKPSC(EPWM_CLKPSCn[11:0]), PERIOD(EPWM_PERIODn[15:0]), CMP(EPWM_CMPDATn[15:0]) and FTCMPDAT (EPWM_FTCMPDATn_m[15:0]) will all load to their active CPSCBUF(EPWM_CPSCBUFn[11:0]), PBUF(EPWM_PBUFn[15:0]), CMPBUF(EPWM_CMPBUFn[15:0]) and FTCMPBUF(EPWM_FTCBUFn_m[15:0]) buffer while each period is completed. For example, after EPWM counter up counts from 0 to PERIOD in the up-counter operation or down counts from PERIOD to 0 in the down-counter operation or counts up from 0 to PERIOD and then counts down to 0 in the up-down counter operation.

Figure 6.14-13 shows period loading timing of up-count operation, where PERIOD DATA0 denotes the initial data of PERIOD, PERIOD DATA1 denotes the first updated PERIOD data by software and so on. CMP also follows this rule. The following describes steps sequence of Figure 6.14-13. User can know the PERIOD and CMP update condition, by watching EPWM period and CMPU event.

1. Software writes CMP DATA1 to CMP at point 1.
2. Hardware loads CMP DATA1 to CMPBUF at the end of EPWM period at point 2.
3. Software writes PERIOD DATA1 to PERIOD at point 3.
4. Hardware loads PERIOD DATA1 to PBUF at the end of EPWM period at point 4.
5. Software writes PERIOD DATA2 to PERIOD at point 5.
6. Hardware loads PERIOD DATA2 to PBUF at the end of EPWM period at point 6.

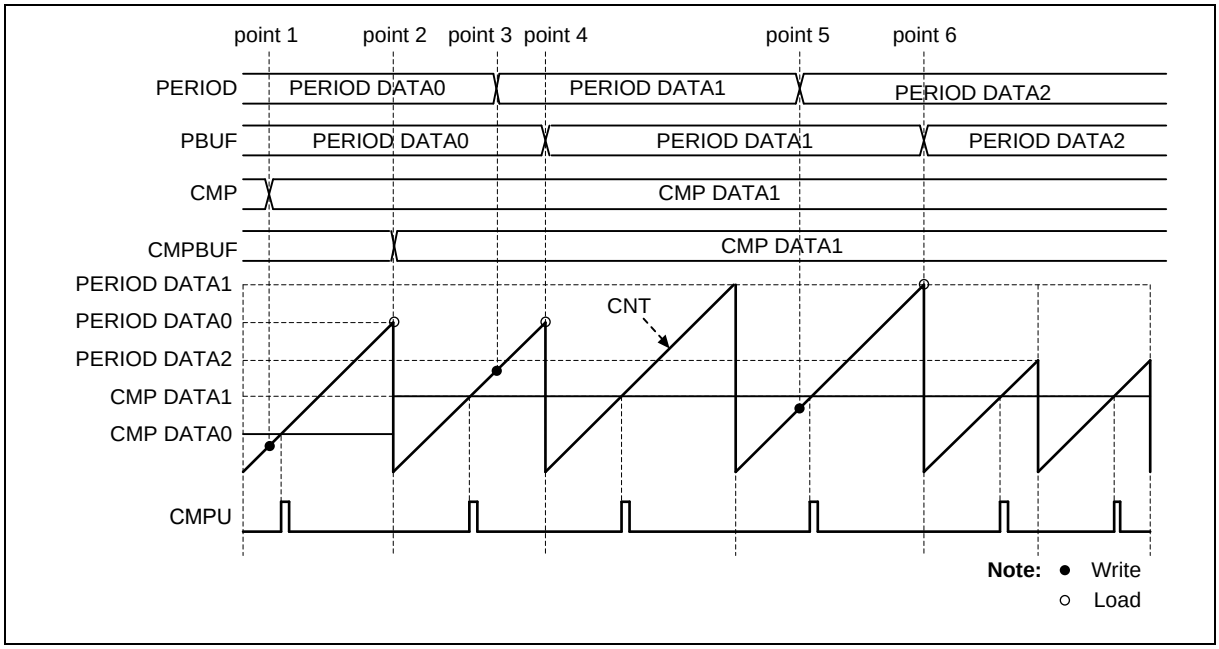


Figure 6.15-13 Period Loading in Up-Count Mode

6.15.5.9 Immediately Loading Mode

If the IMMLDENn (EPWM_CTL0[21:16]) bit is set to 1, EPWM operates at immediately loading mode. In immediately loading mode, when user update CLKPSC(EPWM_CLKPSCn[11:0]), PERIOD(EPWM_PERIODn[15:0]), CMP(EPWM_CMPDATn[15:0]) or FTCMPDAT (EPWM_FTCMPDATn_m[15:0]), CLKPSC, PERIOD, CMP or FTCMPDAT will be load to active CPSCBUF (EPWM_CPSCBUFn[11:0]), PBUF (EPWM_PBUFn[15:0]), CMPBUF (EPWM_CMPBUFn[15:0]) or FTCMPBUF (EPWM_FTCBUFn_m[15:0]) after current counter count is completed. If the updated PERIOD value is less than current counter value, counter will count to 0xFFFF, when counter count to 0xFFFF and prescale count to 0, the flag CNTMAXFn(EPWMx_STATUS[5:0]) will raise, and then counter will count wraparound. Immediately loading mode has the highest priority. If IMMLDENn has been set, other loading mode for channel n will become invalid. Figure 6.14-14 shows an example and its steps sequence is described below.

1. Software writes CMP DATA1 and hardware immediately loading CMP DATA1 to CMPBUF at point 1.
2. Software writes PERIOD DATA1 which is greater than current counter value at point 2; counter will continue counting until equal to PERIOD DATA1 to finish a period loading.
3. Software writes PERIOD DATA2 which is less than the current counter value at point 3; counter will continue counting to its maximum value 0xFFFF and count wraparound from 0 to PERIOD DATA2 to finish this period loading.

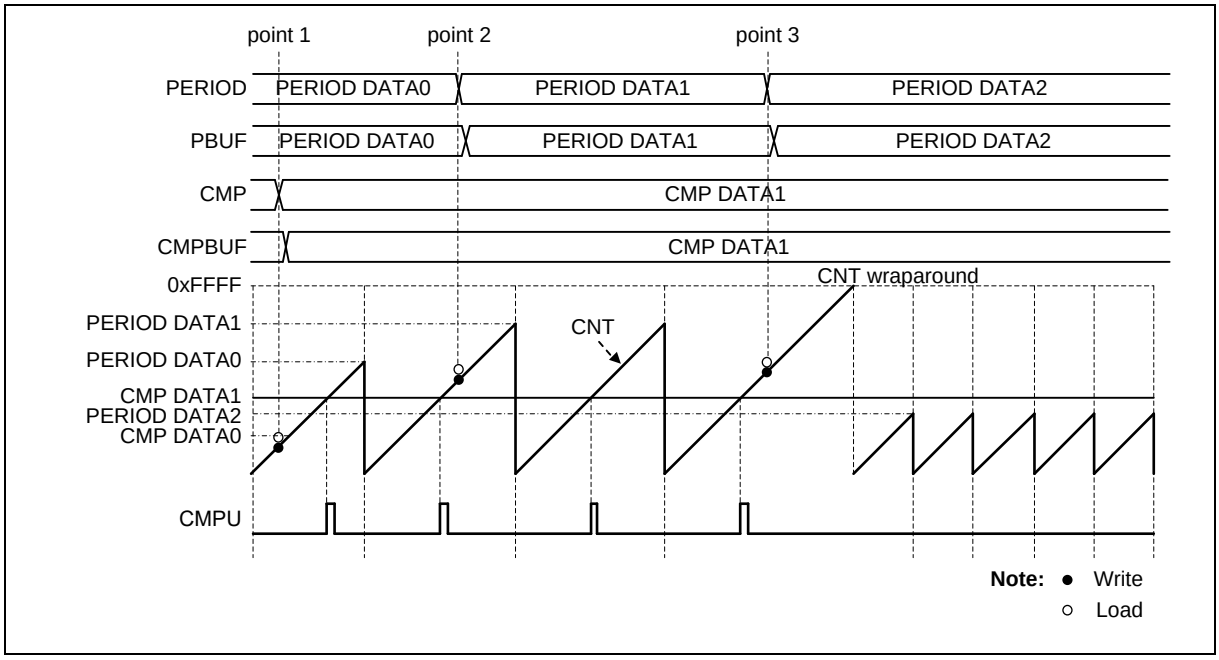


Figure 6.15-14 Immediately Loading in Up-Count Mode

6.15.5.10 Window Loading Mode

When the WINLDENn (EPWM_CTL0[8+n], n = 0,1..5) bit is set to 1, EPWM operates at window loading mode. In Window loading mode, CLKPSC(EPWM_CLKPSCn[11:0]), PERIOD(EPWM_PERIODn[15:0]), CMP(EPWM_CMPDATn[15:0]) and FTCMP(EPWM_FTCMPDATn_m[15:0]) will all load to their active CPSCBUF(EPWM_CPSCBUFn[11:0]), PBUF(EPWM_PBUFn[15:0]), CMPBUF(EPWM_CMPBUFn[15:0]) and FTCMPBUF(EPWM_FTCBUFn_m[15:0]) buffers while each period is completed, but CMPBUF loading are valid only when load window is opened. Every channel n's load window is opened by setting the corresponding LOADn (EPWM_LOAD[n], n = 0,1..5) to 1, and hardware will close the window at the end of EPWM period. Figure 6.15-15 shows an example and its steps sequence is described below.

1. Software writes CMP DATA1 at point 1, and the load window is not opened at this period so CMP will not load to CMPBUF.
2. Software writes LOAD to open the load window at point2.
3. Software writes CLKPSC DATA1 and PERIOD DATA1 at point 3.
4. At point 4, load window has been opened, hardware loads CLKPSC DATA1, PERIOD DATA1 and CMP DATA1 to their buffer and closes the load window at the end of EPWM period.
5. Software writes CLKPSC DATA2 and PERIOD DATA2 at point 5.
6. Hardware loads CLKPSC DATA2 and PERIOD DATA2 to their buffer at the end of EPWM period at point 6.
7. Software writes CLKPSC DATA3 and PERIOD DATA3 at point 7.
8. Software writes LOAD to open the load window at point8.
9. Hardware loads CLKPSC DATA3 and PERIOD DATA3 to their buffer and closes the load window at the end of EPWM period at point 9.

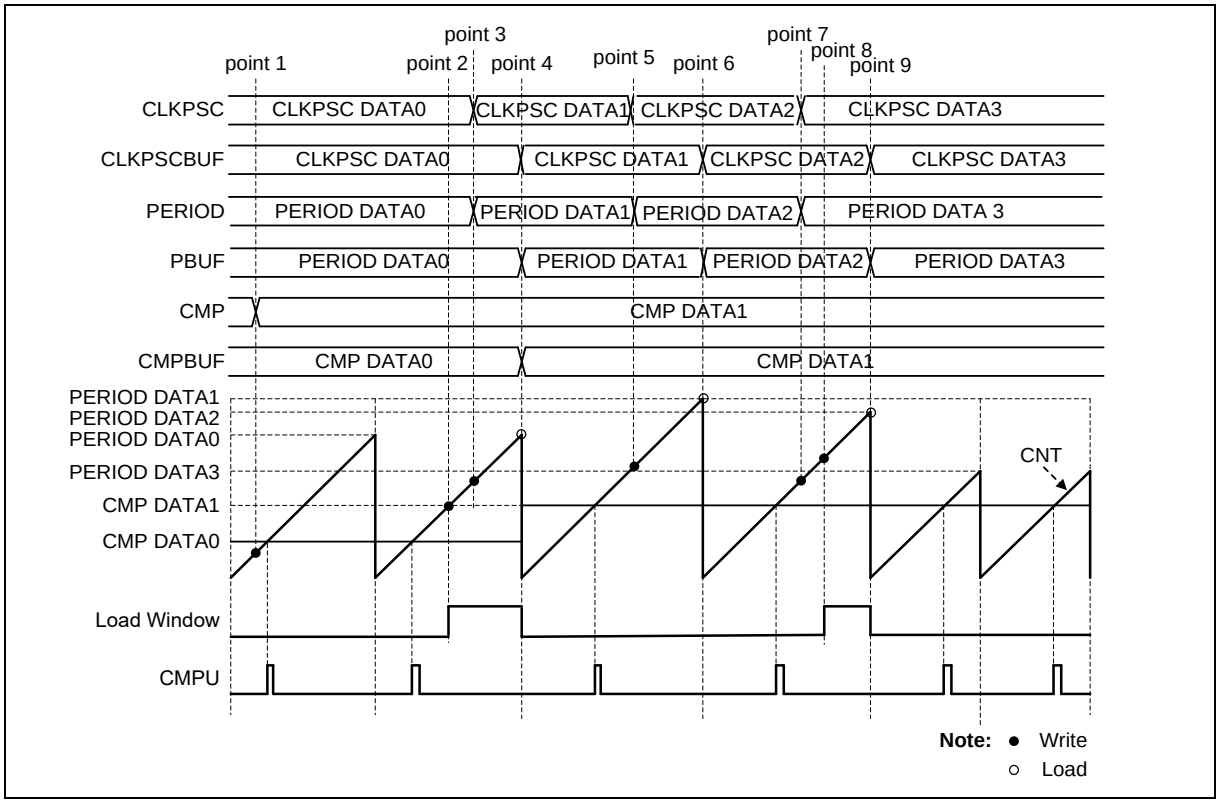


Figure 6.15-15 Window Loading in Up-Count Mode

6.15.5.11 Center Loading Mode

When the CTRLDn (EPWM_CTL0[n], n = 0,1..5) bit is set to 1 and EPWM counter is set to up-down count type, CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) is 0x2, EPWM operates at center loading mode. In center loading mode, CMP(EPWM_CMPDATn[15:0]) and FTCMPDAT (EPWM_FTCMPDATn_m[15:0]) will load to active CMPBUF(EPWM_CMPBUFn[15:0]) and FTCMPBUF(EPWM_FTCBUFn_m[15:0]) buffers in center of each period, that is, counter counts to PERIOD. CLKPSC(EPWM_CLKPSCn[11:0]) and PERIOD(EPWM_PERIODn[15:0]) will all load to their active CPSCBUF(EPWM_CPSCBUFn[11:0]) and PBUF(EPWM_PBUFn[15:0]) buffers while each period is completed. Center loading mode can work with window loading mode, the CMP(EPWM_CMPDATn[15:0]) will load to active CMPBUF(EPWM_CMPBUFn[15:0]) buffer in center of each period, but it is valid only at the interval of load window. Figure 6.14-15 shows an example and its steps sequence is described below.

1. Software writes CMP DATA1 at point 1.
2. Hardware loads CMP DATA1 to CMPBUF at center of EPWM period at point 2.
3. Software writes PERIOD DATA1 at point 3.
4. Hardware loads PERIOD DATA1 to PBUF at the end of EPWM period at point 4.
5. Software writes CMP DATA2 at point 5.
6. Hardware loads CMP DATA2 to CMPBUF at center of EPWM period at point 6.
7. Software writes PERIOD DATA2 at point 7.
8. Hardware loads PERIOD DATA2 to PBUF at the end of EPWM period at point 8.

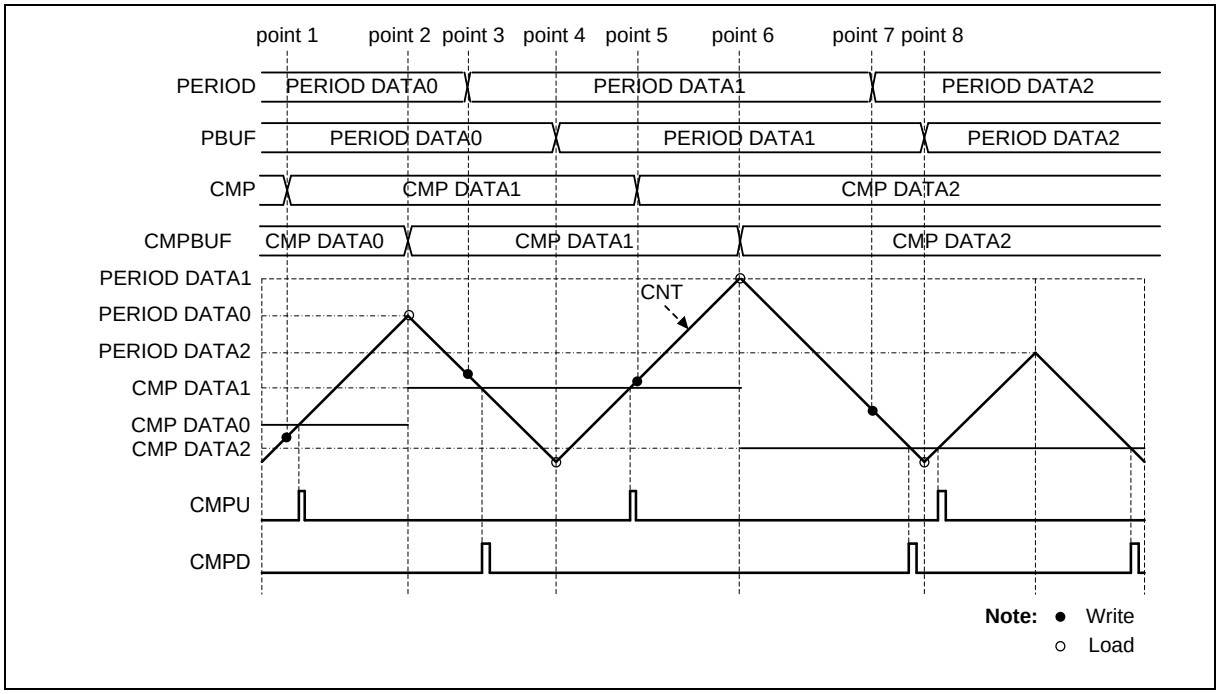


Figure 6.15-16 Center Loading in Up-Down-Count Mode

6.15.5.12 EPWM Counter Operation Mode

The EPWM counter supports two operation modes: One-shot mode and Auto-reload mode. EPWM counter will operate in One-shot mode if CNTMODEn (EPWM_CTL1[16+n], n = 0,1..5) bit is set to 1, and operate in Auto-reload mode if set to 0.

In One-shot mode, EPWM_CMPDATn and EPWM_PERIODn registers should be written first and then set CNTENn (EPWM_CNTEN[n], n = 0,1..5) bit to 1 to enable EPWM prescaler and counter start running. After EPWM counter counted a period, counter value will keep 0.

User can re-start next one-shot by writing new value to CMP(EPWM_CMPDATn[15:0]) bits. If one-shot counter still running, to update EPWM_CMPDATn register will cause next one-shot as continuous one-shot. Besides, to write EPWM_CMPDATn register twice under continuous one-shot operation, latest value in EPWM_CMPDATn register is valid at next one-shot period and only generate one-shot pulse once. Moreover, if user wants to clear counter within one-shot operation and starts next one-shot, user should monitor counter value to check counter has cleared and then writes EPWM_CMPDATn register. Figure 6.15-17 is an example and following is steps sequence.

1. Software writes PERIOD DATA1 and hardware immediately loading PERIOD DATA1 to PBUF at point 1.
2. Software writes CMP DATA1 which is equal to CMP DATA0 at point 2 and hardware immediately loading CMPDAT DATA1 to CMPBUF, this event also trigger one-shot.
3. Software writes CMP DATA2 and re-trigger next one-shot (continuous one-shot) at point 3.
4. Software writes CMP DATA3 to cover CMP DATA2 and re-trigger next one-shot at point 4.
5. Period loading CMP DATA3 to CMPBUF at point 5.
6. There are no new CMP write in the previous period, and the counter value is kept as 0 at point 6

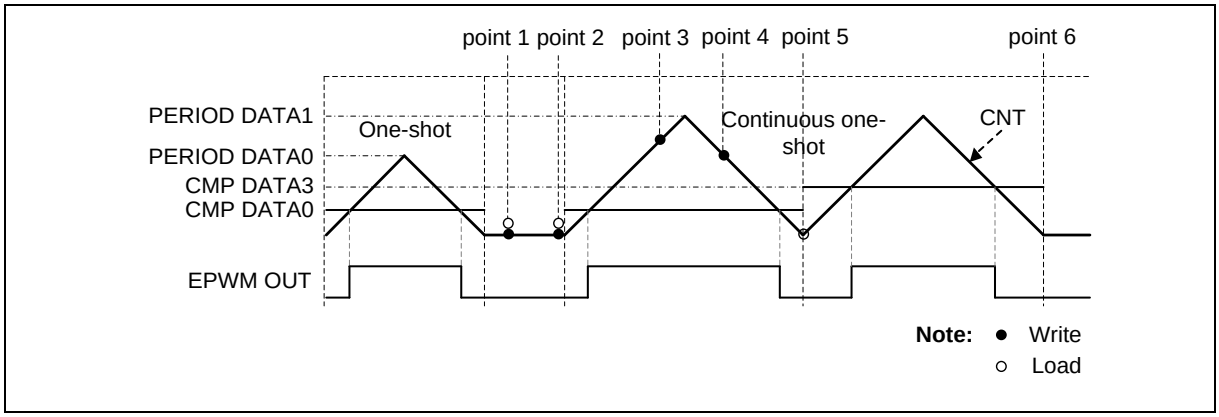


Figure 6.15-17 EPWM One-shot Mode Output Waveform

In Auto-reload mode, EPWM_CMPDATn and EPWM_PERIODn registers should be written first and then the CNTENn(EPWM_CNTEN[n], n = 0,1..5) bit is set to 1 to enable EPWM prescaler and start to run counter. The value of CLKPSC(EPWM_CLKPSCn[11:0]), PERIOD(EPWM_PERIODn[15:0]) and CMP(EPWM_CMPDATn[15:0]) will auto reload to their active buffer according different loading mode. If PERIOD(EPWM_PERIODn[15:0]) is set to 0, EPWM counter will be set to 0.

6.15.5.13 EPWM Pulse Generator

The EPWM pulse generator uses counter and comparator events to generate EPWM pulse. The events are: zero point, period point in up counter type and down counter type, center point in up-down counter type and counter equal to comparator point in three types. As to up-down counter type, there are two counter equal comparator points, one at up count and the other at down count. Besides, Complementary mode has two comparators compared with counter, and thus comparing equal points will become four in up-down counter type and two for up or down counter type.

Each event point can decide EPWM waveform to do nothing (X), set Low (L), set High (H) or toggle (T) by setting the EPWM_WGCTL0 and EPWM_WGCTL1 registers. Using these points can easily generate asymmetric EPWM pulse or variant waveform as shown in Figure 6.14-16. In the figure, EPWM is in complementary mode, there are two comparators n and m to generate EPWM pulse. n denotes even channel number 0, 2, or 4, and m denotes odd channel number 1, 3, or 5. n channel and m channel are complementary paired. Complementary mode uses two channels (CH0 and CH1, CH2 and CH3, or CH4 and CH5) as a pair of EPWM outputs to generate complement paired waveforms. CMPU denotes CNT(EPWM_CNTn[15:0]) is equal to CMP(EPWM_CMPDATn[15:0]) when counting up. CMPD denotes CNT bits is equal to CMP bits when counting down.

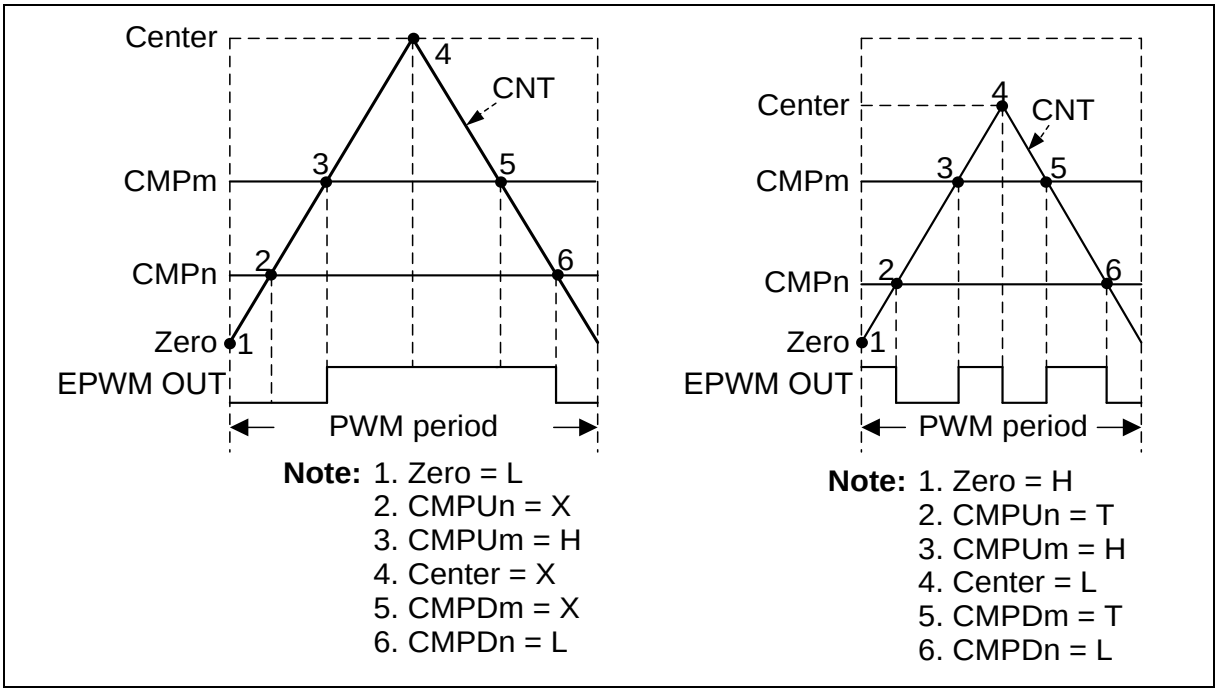


Figure 6.15-18 EPWM Pulse Generation

User can also use software event to force EPWM output. Software trigger SWETRGn (EPWM_SWEOFTRG[n], n = 0,1..5) event can decide EPWM waveform to do nothing (X), set Low (L), set High (H) or toggle (T) by setting OUTACTSn (EPWM_SWEOFCTL[2n+1:2n], n = 0,1..5).

Figure 6.15-19 shows software event effect EPWM output according to OUTACTS0 setting. Besides, software events have the highest priority. If other events and software events happen in the same time, software event will be serviced first.

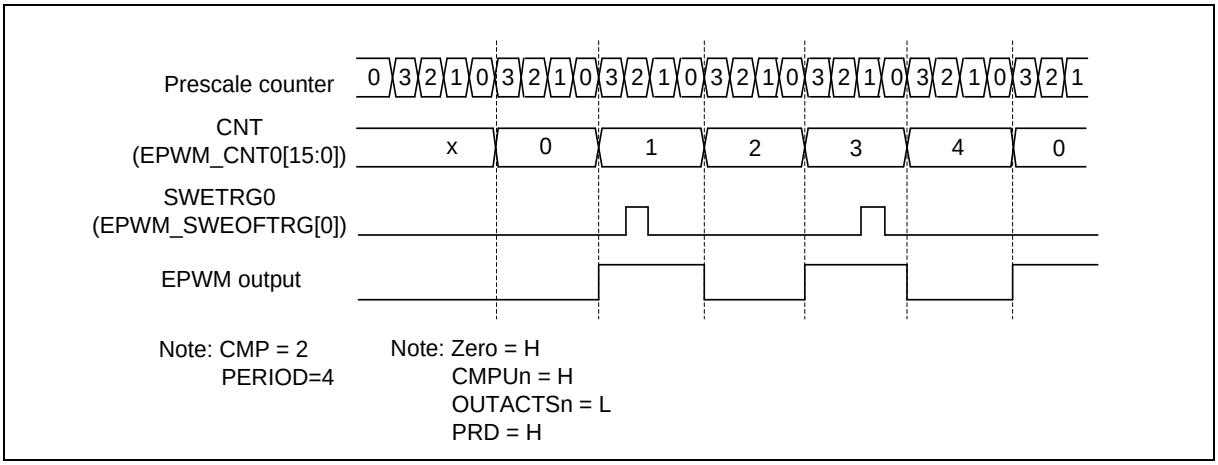


Figure 6.15-19 Software Event Output Force

The generation events may sometimes set to the same value, as the reason, events priority between different counter types are list below, up counter type (Table 6.14-2), down counter type (Table 6.14-3) and up-down counter type (Table 6.14-4). By using event priority, user can easily generate 0% to 100% duty pulse as shown in Figure 6.14-17.

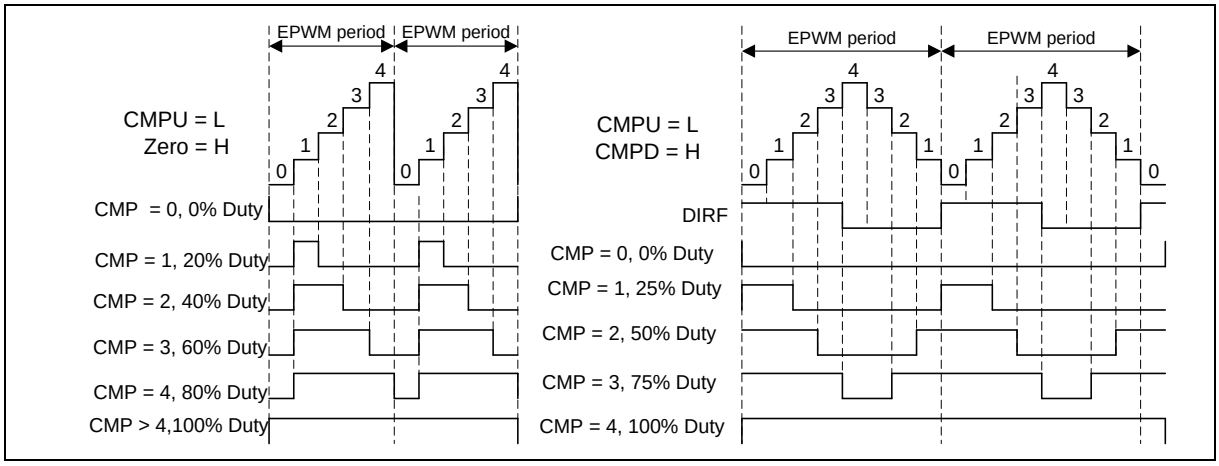


Figure 6.15-20 EPWM 0% to 100% Pulse Generation

Priority	Up Event
1 (Highest)	Period event (CNT = PERIOD)
2	Compare up event of odd channel (CNT = CMPUm)
3	Compare up event of even channel (CNT = CMPUn)
4 (Lowest)	Zero event (CNT = 0)

Table 6.15-2 EPWM Pulse Generation Event Priority for Up-Counter

Priority	Down Event
1 (Highest)	Zero event (CNT = 0)
2	Compare down event of odd channel (CNT = CMPDm)
3	Compare down event of even channel (CNT = CMPDn)
4 (Lowest)	Period event (CNT = PERIOD)

Table 6.15-3 EPWM Pulse Generation Event Priority for Down-Counter

Priority	Up Event	Down Event
1 (Highest)	Compare up event of odd channel (CNT = CMPUm	Compare down event of odd channel (CNT = CMPDm)
2	Compare up event of even channel (CNT = CMPUn)	Compare down event of even channel (CNT = CMPDn)
3 (Lowest)	Zero event (CNT = 0)	Period (center) event (CNT = PERIOD)

Table 6.15-4 EPWM Pulse Generation Event Priority for Up-Down-Counter

6.15.5.14 EPWM Output Mode

The EPWM supports two output modes: Independent mode which may be applied to DC motor system, Complementary mode with dead-time insertion which may be used in the application of AC induction motor and permanent magnet synchronous motor.

6.15.5.15 Independent mode

By default, the EPWM is operating in independent mode. Independent mode is enabled when pair channel's corresponding OUTMODEn (EPWM_CTL1[24+n/2], n = 0,2,4) bit is set to 0. In this mode, six EPWM channels: EPWM_CH0, EPWM_CH1, EPWM_CH2, EPWM_CH3, EPWM_CH4 and EPWM_CH5 are running off its own period and duty as shown in Figure 6.15-21.

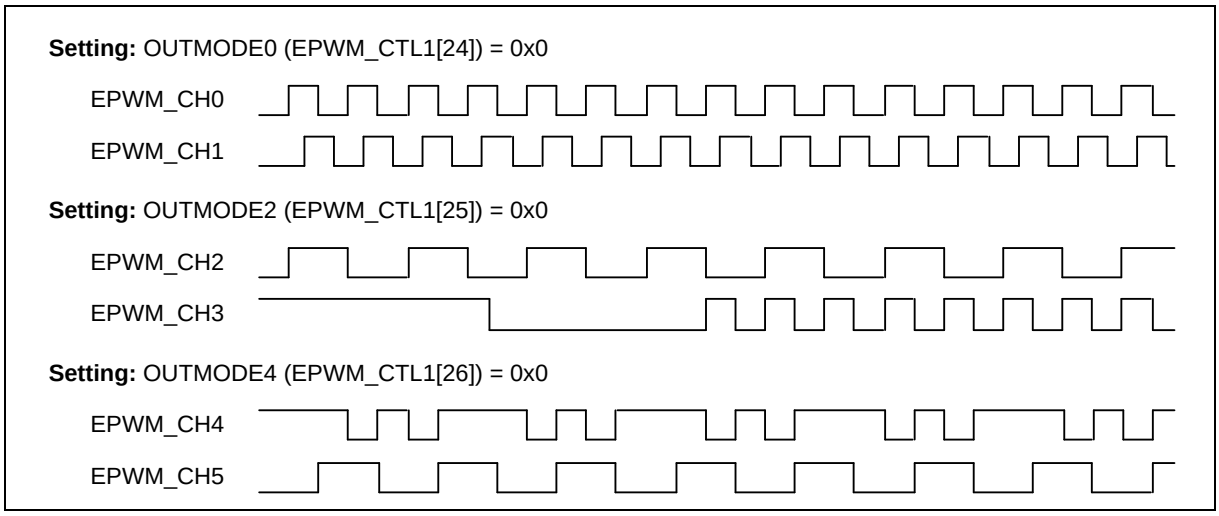


Figure 6.15-21 EPWM Independent Mode Waveform

6.15.5.16 Complementary Mode

Complementary mode is enabled when the pair channel's corresponding OUTMODEn (EPWM_CTL1[24+n/2], n=0,2,4) bit set to 1. In this mode there are 3 EPWM generators utilized for complementary mode, with total of 3 EPWM output paired pins in this module. In Complimentary modes, the internal odd EPWM signal must always be the complement of the corresponding even EPWM signal. EPWM_CH1 will be the complement of EPWM_CH0. EPWM_CH3 will be the complement of EPWM_CH2 and EPWM_CH5 will be the complement of EPWM_CH4 as shown in Figure 6.14-19.

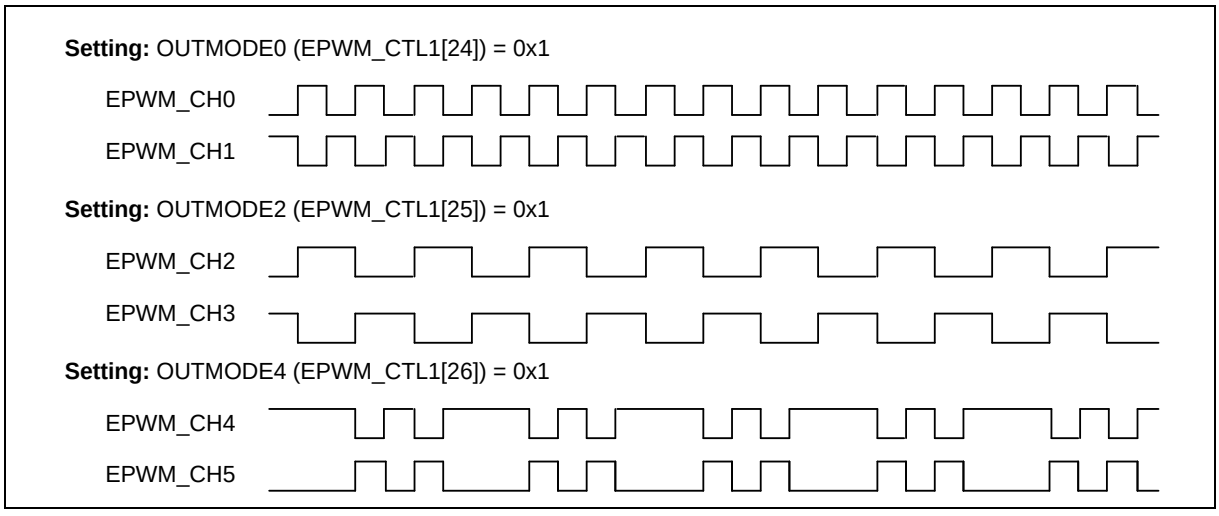


Figure 6.15-22 EPWM Complementary Mode Waveform

6.15.5.17 EPWM Output Function

Based on the output mode, there are two output functions: group and synchronous functions for

advanced output control. Group function, forces the EPWM_CH2 and EPWM_CH4 synchronous with EPWM_CH0 generator and forces the EPWM_CH3 and EPWM_CH5 synchronous with EPWM_CH1, may simplify updating duty control in DC and BLDC motor applications. Besides, Synchronous function makes any channel of EPWM0 and EPWM1 in phase, user can control phase value and direction.

6.15.5.18 Group Function

Group function is enabled when GROUPEN (EPWM_CTL0[24]) is set to 1, no matter in independent or complementary mode. This control allows all even EPWM channels output to be controllable by EPWM_PERIOD0 and EPWM_CMPDAT0 registers and all odd EPWM channels output to be controllable by EPWM_PERIOD1 and EPWM_CMPDAT1 registers. That is, user only needs to set EPWM_CH0 to get EPWM_CH0, EPWM_CH2 and EPWM_CH4 output the same pulse, and set EPWM_CH1 to get EPWM_CH1, EPWM_CH3 and EPWM_CH5 output the same pulse, as shown in Figure 6.15-23. When operating group function, OUTMODE0, OUTMODE2 and OUTMODE4 bits of CTL1 register must all set to 0 for independent mode or all set to 1 for complementary mode.

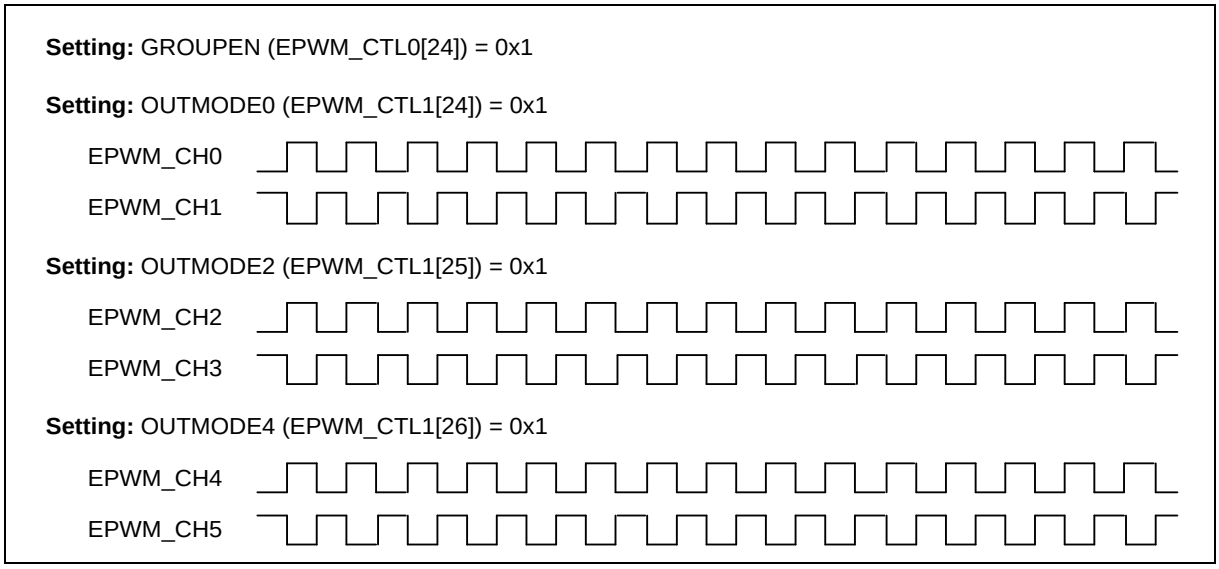


Figure 6.15-23 EPWM Group Function Waveform

6.15.5.19 Synchronous Function

Synchronous function can only be enabled when complementary mode is enabled. Each complementary pair has one input SYNC_IN and output SYNC_OUT signal. The first pair is the EPWM0_CH0/1, and the last one is EPWM1_CH4/5. The signal source of SYNC_IN comes from the SYNC_OUT signal of the previous pair as shown in Figure 6.15-24. Specifically, the first SYNC_IN signal is from the external EPWM0_SYNC_IN pin, and the first SYNC_OUT signal cannot only be the signal source of the next SYNC_IN but also transmitted to the external EPWM0_SYNC_OUT pin.

The input signal from EPWM0_SYNC_IN pin will be filtered by a 3-bit noise filter as Figure 6.15-25. In addition, it can be inversed by setting the bit SINPINV (EPWM_SYNC[23]) to realize the polarity setup for the input signal. The noise filter sampling clock can be selected by setting bits SFLTCSEL (EPWM_SYNC[19:17]) to fit different noise properties. Moreover, by setting the bits SFLTCNT (EPWM_SYNC[22:20]), user can define by how many sampling clock cycles a filter will recognize the effective edge of the SYNC_IN signal. Configuring the SNFLTEN (EPWM_SYNC[16]) will enable the noise filter function. By default, it is disabled.

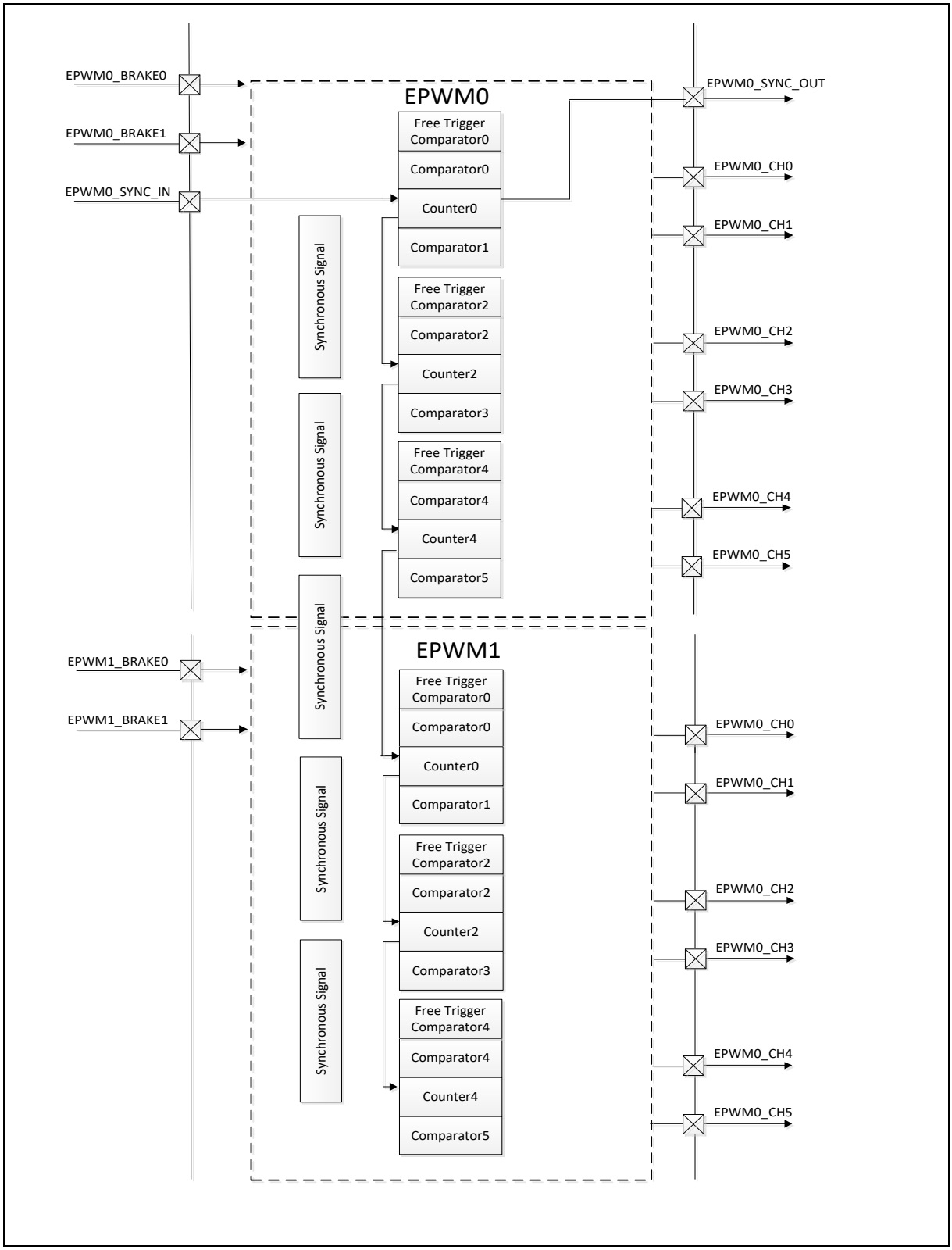


Figure 6.15-24 EPWM SYNC_IN and SYNC_OUT Signal Block Diagram

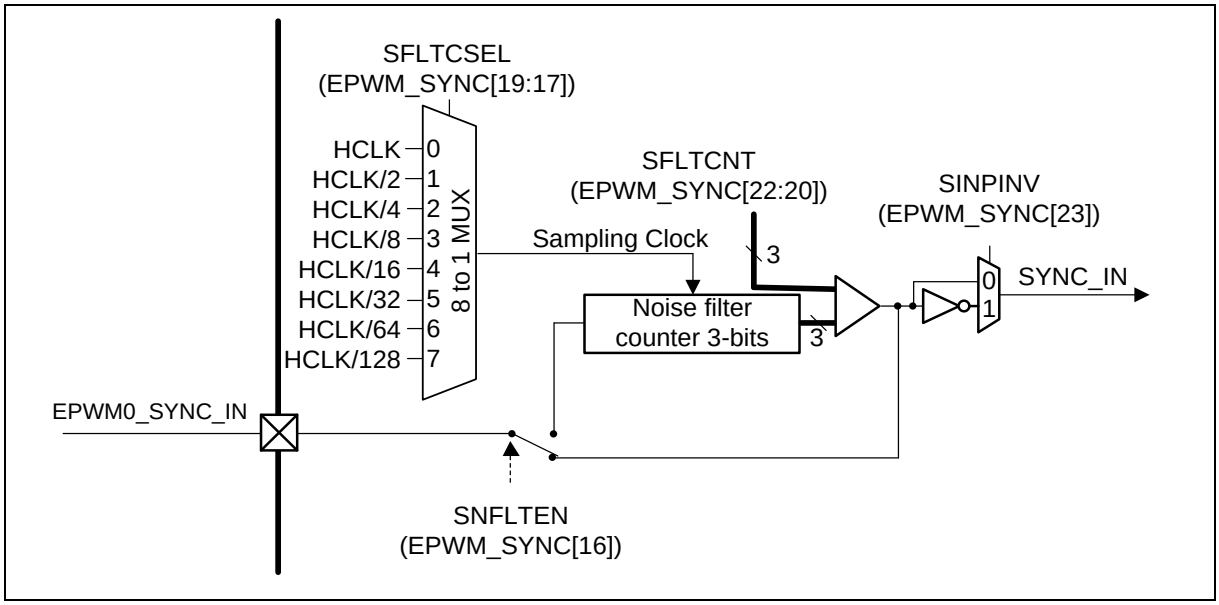


Figure 6.15-25 EPWM SYNC_IN Noise Filter Block Diagram

User can use SINSRCn (EPWM_SYNC[9+n:8+n], n=0,2,4) bits to select the synchronize source. When SINSRCn bits is set to 0, user can generate SYNC_IN signal for the next counter's synchronization when EPWM0_SYNC_IN pin is high or setting SWSYNCn (EPWM_SWSYNC[n/2], n=0,2,4) to 1. Synchronizing source can also be selected as CNT = 0 or CNT = EPWM_CMPDATm register (if being the up-down counter type, it will synchronize twice in a EPWM period) to trigger a sync event or to disable SYNC_OUT signal.

Figure 6.15-26 is EPWM counter synchronous function block diagram. When the PHSENn (EPWM_SYNC[n/2], n=0,2,4) is enabled and the synchronous source has a happening event, the counter will load a value from the PHS (EPWM_PHSn_m[15:0]) register. This method synchronizes counters to different phase in the same time. In the up-down counter type, user can set the value in PHSDIRn (EPWM_SYNC[24+n/2], n=0,2,4) to control the counter direction after synchronization. Although the Synchronous function can synchronize channels in phase, it can't work from the beginning of EPWM enable. To start EPWM and PWM counters in the same time, user have to set the EPWM Synchronous Start Control Register SSENn (EPWM_SSCTL[n], n=0,1..5) to enable the channel counters which are planned to start counting together, and select the SSRC(EPWM_SSCTL[9:8]) to choose the Synchronous Start source, followed by setting the EPWM Synchronous Start Trigger Register CNTSEN (EPWM_SSTRG[0]). Because the source of synchronous start belongs to different PCLK clock domain, please make sure the APB0DIV (CLK_PCLKDIV[2:0]) and APB1DIV (CLK_PCLKDIV[6:4]) set the same divider value if the selected synchronous start source IP and destination IP are different APB clock domain.

For applications, please do not use Group and Synchronous function simultaneously because the Synchronous function will be inactive.

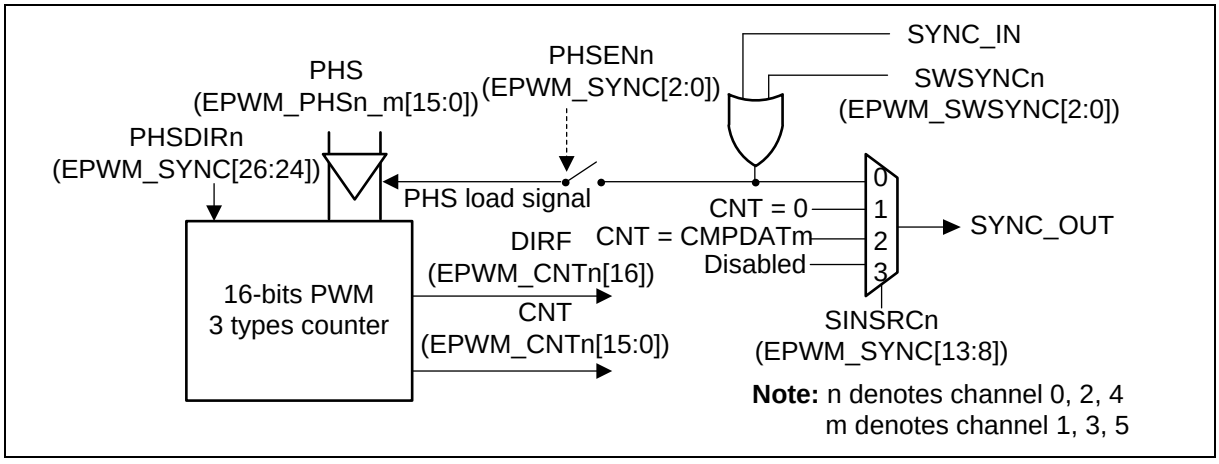


Figure 6.15-26 EPWM Counter Synchronous Function Block Diagram

Figure 6.15-27 is an example of the synchronous function in the up-down counter type. In the example, synchronizing source comes from the external EPWM SYNC_IN signal. At the beginning, the output waveform of EPWM_CH0, EPWM_CH2 and EPWM_CH4 are in the same phase. Then at Point A, the EPWM SYNC input signal comes as a sync event, resulting in phase shifts and counting direction changes for all of the counters. To realize the altered counter behaviors before the sync event coming, user has to setup the corresponding phase value in the PHS (EPWM_PHSn_m[15:0]) as well as the counting direction in the PHSDIRn (EPWM_SYNC[24+n/2], n=0,2,4). In this case, one third of phase shifts are made. by setting the corresponding channel n's counter counting direction after synchronizing, as illustrated around the left side of Figure 6.15-27.

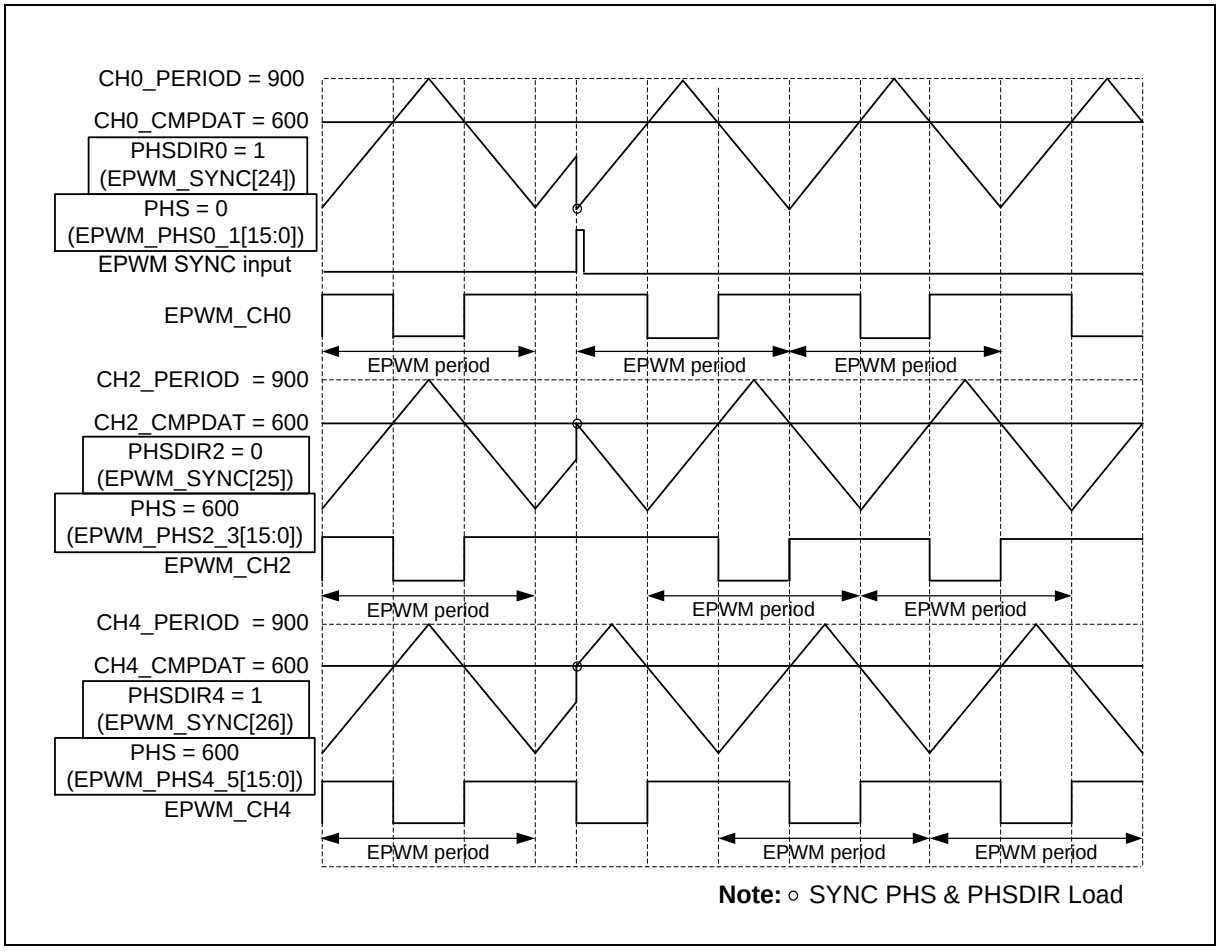


Figure 6.15-27 EPWM Synchronous Function with Synchronize source from SYNC_IN Signal

6.15.5.20 EPWM Output Control

After EPWM pulse generation, there are four to six steps to control the output of EPWM channels. In independent mode, there are four steps: Mask, Brake, Pin Polarity and Output Enable as shown in Figure 6.14-20. In complementary mode, it needs two more steps to precede these four steps, Complementary channels and Dead-Time Insertion as shown in Figure 6.14-21.

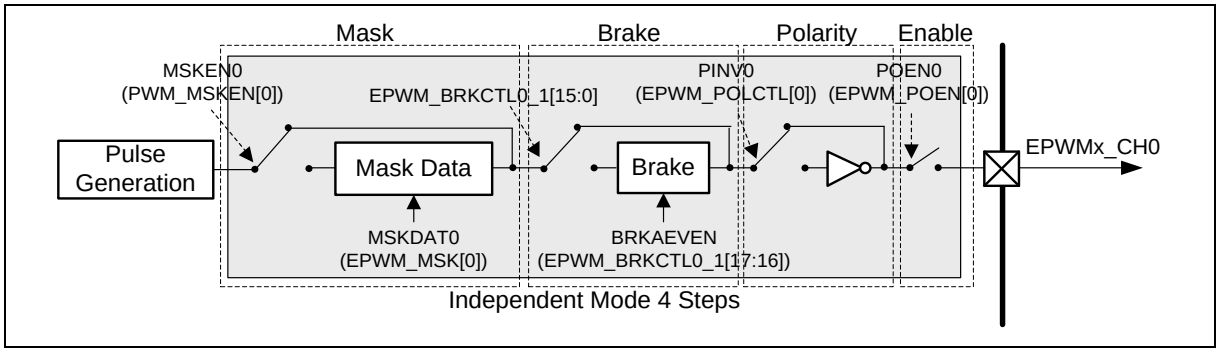


Figure 6.15-28 EPWMx_CH0 Output Control in Independent Mode

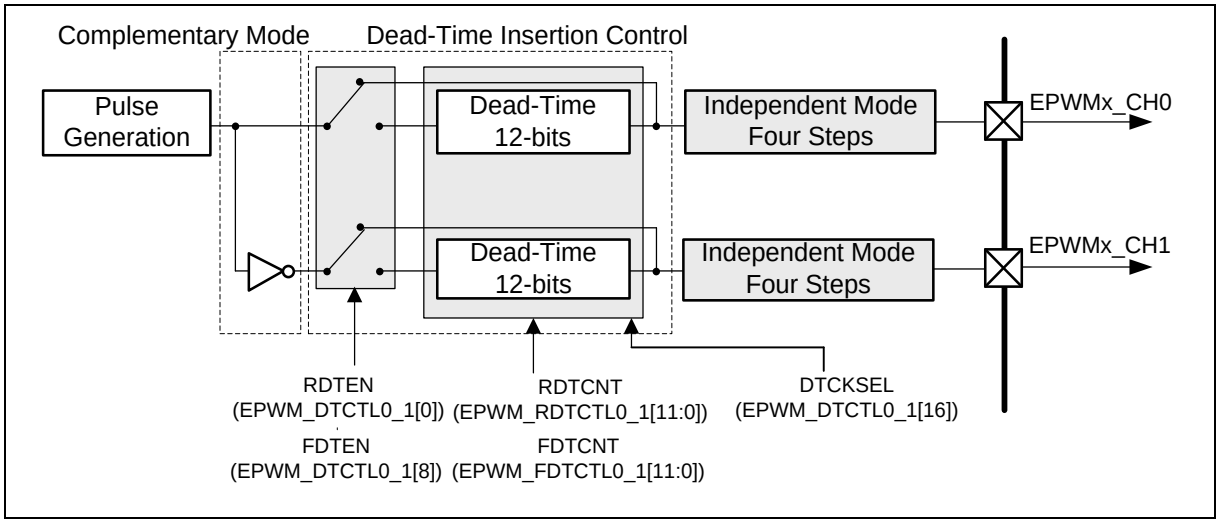


Figure 6.15-29 EPWMx_CH0 and EPWMx_CH1 Output Control in Complementary Mode

6.15.5.21 Dead-Time Insertion

In the complementary application, the complement channels may drive the external devices like power switches. The dead-time generator inserts a low level period called “dead-time” between complementary outputs to drive these devices safely and to prevent system or devices from the burn-out damage. Hence the dead-time control is a crucial mechanism to the proper operation of the complementary system. Set the corresponding channel n RDTEN (EPWM_DTCTL[n/2]) bit to enable rising dead-time function and FDTENn (EPWM_DTCTL[8+n/2], n=0,2,4) bit to enable falling dead-time function. Set RDTCNT (EPWM_RDTCNTn_m[11:0], n=0,2,4, m=n+1) and FDTCNT (EPWM_FDTCNTn_m[11:0], n=0,2,4, m=n+1) to control dead-time rising and falling period. Dead-time can be calculated from the following formula:

Dead-time = (RDTCNT (EPWM_RDTCNTn_m[11:0])+1) * EPWMx_CLK period

Dead-time insertion clock source can be selected from prescaler output by setting DTCKSELn (EPWM_DTCTL[16+n/2], n=0,2,4) to 1. By default, clock source comes from EPWM_CLK, which is prescaler input. Then the dead-time can be calculated from the following formula:

Dead-time = (RDTCNT (EPWM_RDTCNTn_m[11:0])+1) *

(CLKPSC (EPWM_CLKPSCn[11:0])+1)*EPWMx_CLK period

Please note that the EPWM_RDTCNTn_m and EPWM_DTCTL are write-protected registers.

Figure 6.14-22 indicates the dead-time insertion for one pair of EPWM signals.

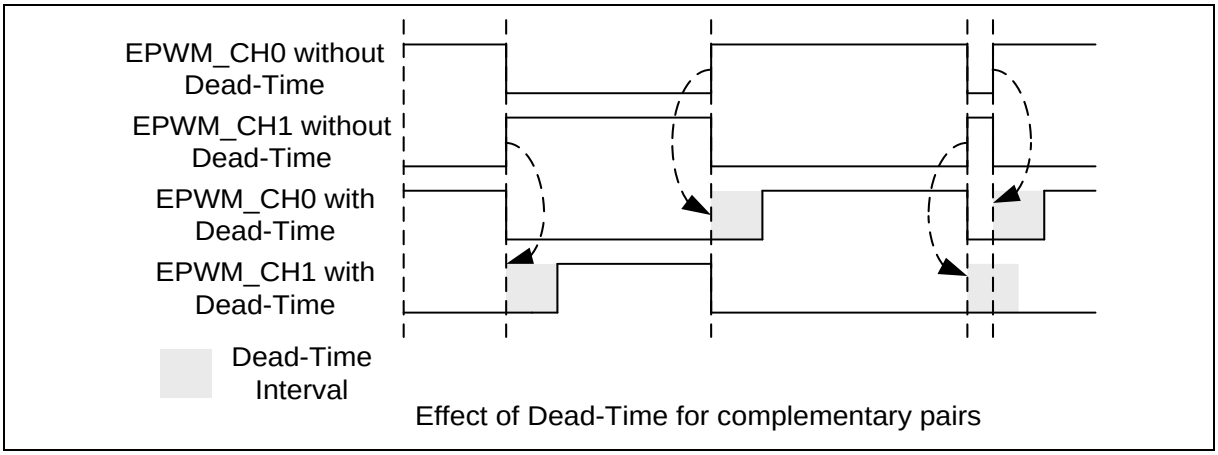


Figure 6.15-30 Dead-Time Insertion

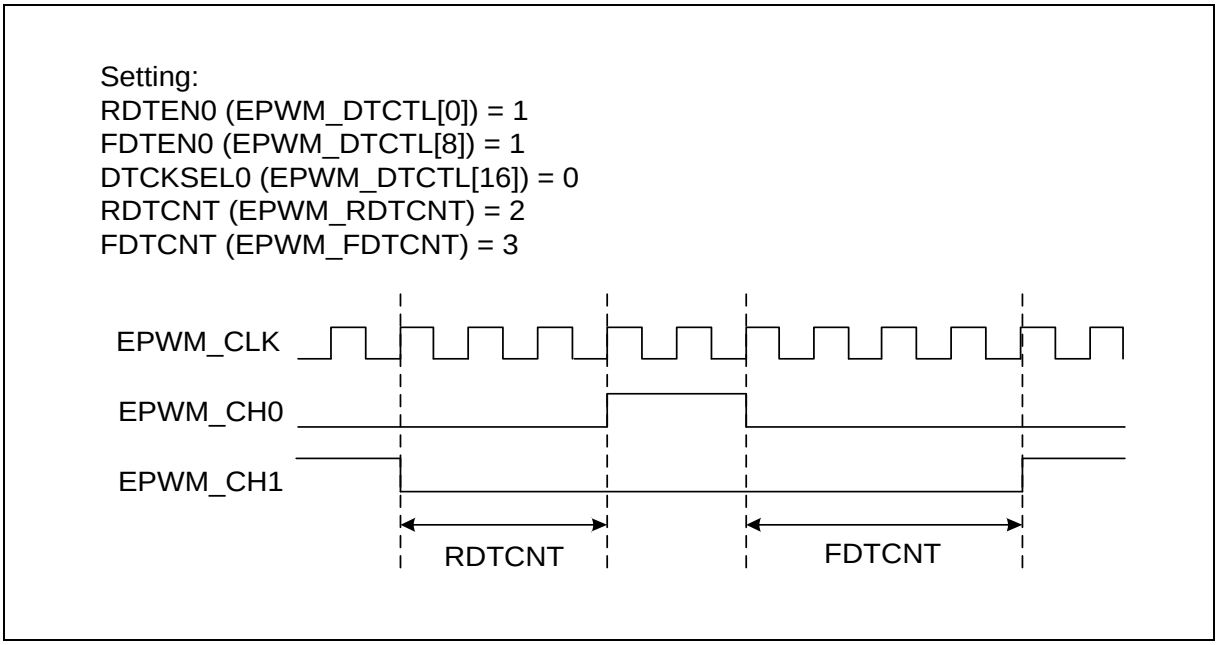


Figure 6.15-31 Dead-Time Independent Control

6.15.5.22 EPWM Mask Output Function

Each of the EPWM channel output value can be manually overridden with the settings in EPWM Mask Enable Register (EPWM_MSKEN) and EPWM Mask Data Register (EPWM_MSK) With these settings, the EPWM channel outputs can be assigned to specified logic states independent of the duty cycle comparison units. The EPWM mask bits are useful when controlling various types of Electrically Commutated Motor (ECM) like a BLDC motor. The EPWM_MSKEN register contains six bits, MSKENn(EPWM_MSKEN[5:0]). If the MSKENn is set to active-high, the EPWM channel n output will be overridden. The EPWM_MSK register contains six bits, MSKDATn(EPWM_MSK[5:0]). The bit value of the MSKDATn determines the state value of the EPWM channel n output when the channel is overridden. Figure 6.14-23 shows an example of how EPWM mask control can be used for the override feature.

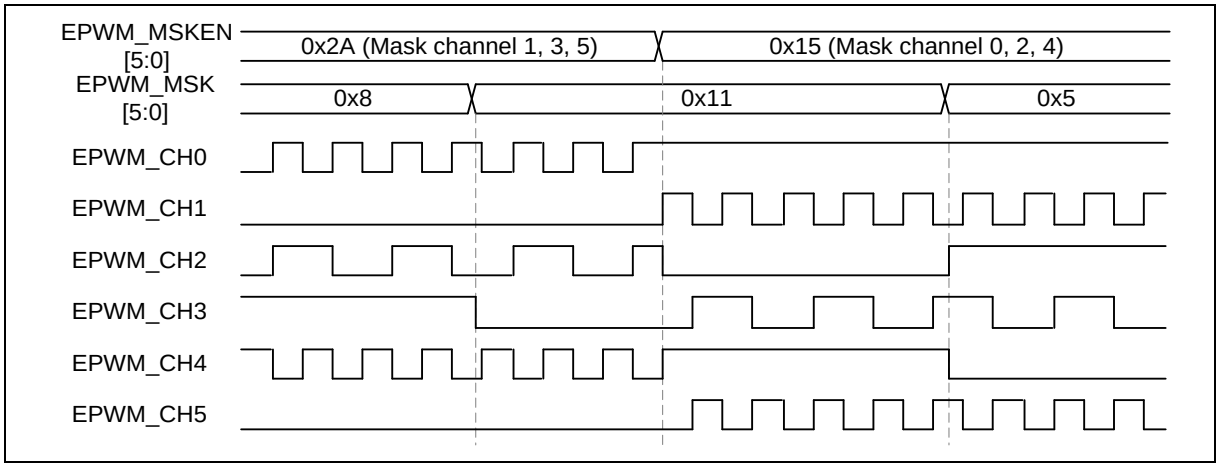


Figure 6.15-32 Illustration of Mask Control Waveform

6.15.5.23 EPWM Brake

Each EPWM module has two external input brake control signals. User can set BKySRC (PWM_BNF[16+8*y], y = 0, 1) to select the source of brake y is from EPWMx_BRAKEy (x = 0, 1). The external signals will be filtered by a 3-bit noise filter. User can enable the noise filter function by BRKyNFEN (EPWM_BNF[8*y], y = 0, 1), and noise filter sampling clock can be selected by setting BRKyNFSEL (EPWM_BNF[3+8*y:1+8*y], y = 0, 1) to fit different noise properties. Moreover, by setting BRKyFCNT (EPWM_BNF[6+8*y:4+8*y], y = 0, 1) bits, user can define by how many sampling clock cycles a filter will recognize the effective edge of the brake signal.

In addition, it can be inversed by setting BRKyPINV (EPWM_BNF(7+8*y), y = 0 , 1) to realize the polarity setup for the brake control signals. Set BRKyPINV bit to 0, brake event will occurred when EPWMx_BRAKEy pin status is from low to high; set BRKyPINV to 1, brake event will occurred when EPWMx_BRAKEy pin status is from high to low.

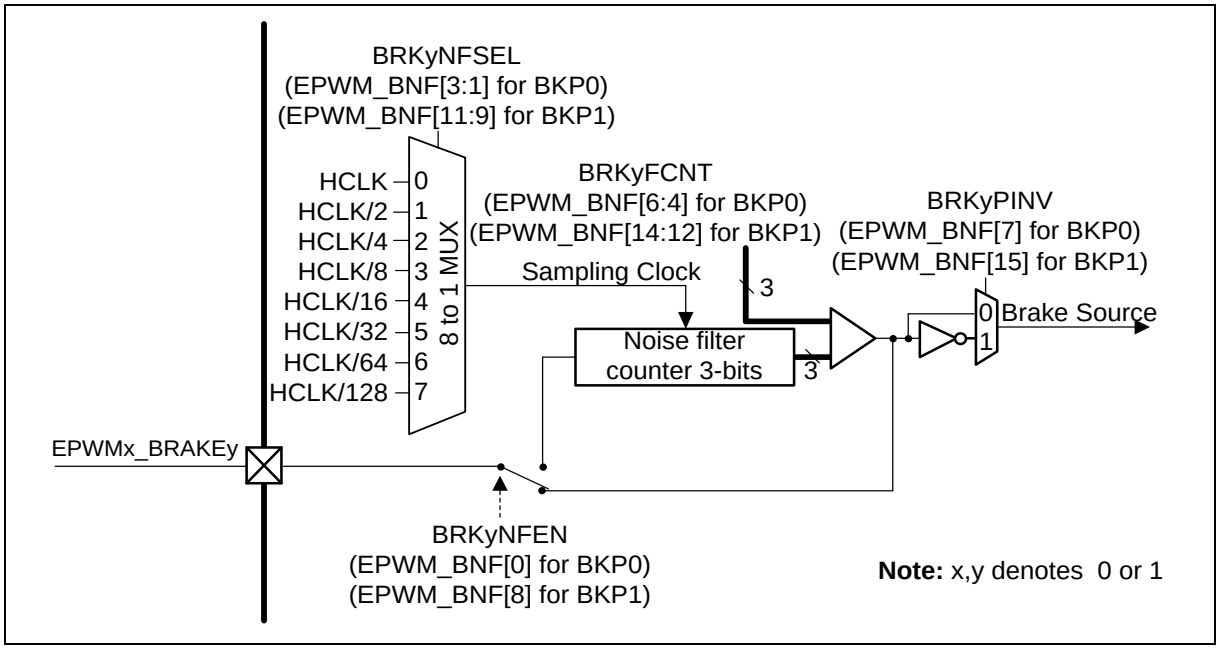


Figure 6.15-33 Brake Noise Filter Block Diagram

For Complementary mode, it is often necessary to set a safe output state to the complement output pairs once the brake event occurs.

Each complementary channel pair shares a EPWM brake function, as shown in Figure 6.14-25. To control paired channels to output safety state, user can setup BRKAEVEN (EPWM_BRKCTL0_1[17:16]) for even channels and BRKAODD (EPWM_BRKCTL0_1[19:18]) for odd channels when the fault brake event happens. There are two brake detectors: Edge detector and Level detector. When the edge detector detects the brake signal and BRKEIENn_m (EPWM_INTEN1[2:0]) is enabled, the brake function generates BRK_INT. This interrupt needs software to clear, and the BRKESTSn (EPWM_INTSTS1[21:16]) brake state will keep until the next EPWM period starts after the interrupt cleared. The brake function can also operate in another way through the level detector. Once the level detector detects the brake signal and the BRKLIENn_m (EPWM_INTEN1[10:8]) is also enabled, the brake function will generate BRK_INT, but BRKLSTSn (EPWM_INTSTS1[29:24]) brake state will auto recovery to normal output while level brake source recovery to high level and pass through “Low Level Detection” at the EPWM waveform period when brake condition removed without clear interrupt.

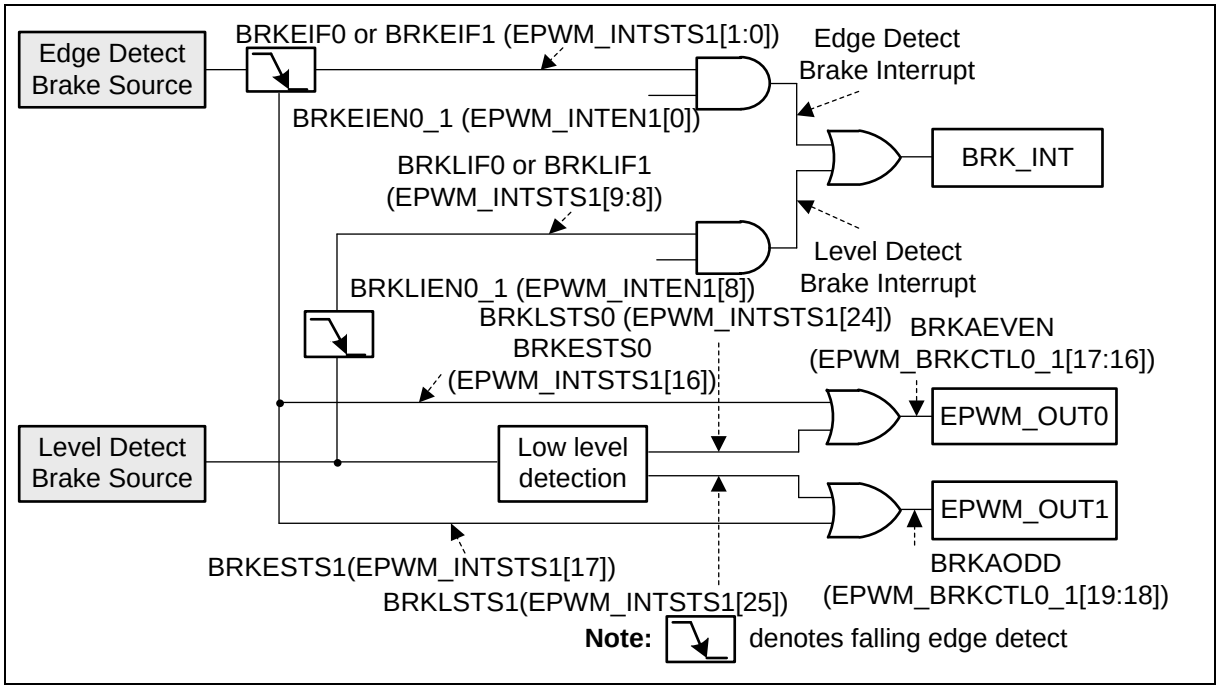


Figure 6.15-34 Brake Block Diagram for EPWMx_CH0 and EPWMx_CH1 Pair

Figure 6.14-26 illustrates the edge detector waveform for EPWMx_CH0 and EPWMx_CH1 pair. In this case, the edge detect brake source has occurred twice for the brake events. When the event occurs, both of the BRKEIF0 and BRKEIF1 flags are set and BRKESTS0 and BRKESTS1 bits are also set to indicate brake state of EPWMx_CH0 and EPWMx_CH1. For the first occurring event, software writes 1 to clear the BRKEIF0 flag. After that, the BRKESTS0 bit is cleared by hardware at the next start of the EPWM period. At the same moment, the EPWMx_CH0 outputs the normal waveform even though the brake event is still occurring. The second event also triggers the same flags, but at this time, software writes 1 to clear the BRKEIF1 flag. Afterward, EPWMx_CH1 outputs normally at the next start of the EPWM period.

As a contrast to the edge detector example, Figure 6.14-27 illustrates the level detector waveform for EPWMx_CH0 and EPWMx_CH1 pair. In this case, the BRKLIF0 and BRKLIF1 flags can only indicate the brake event having occurred. The BRKLSTS0 and BRKLSTS1 brake states will automatically recover at the start of the next EPWM period no matter at what states the BRKLIF0 and BRKLIF1 flags are at that moment.

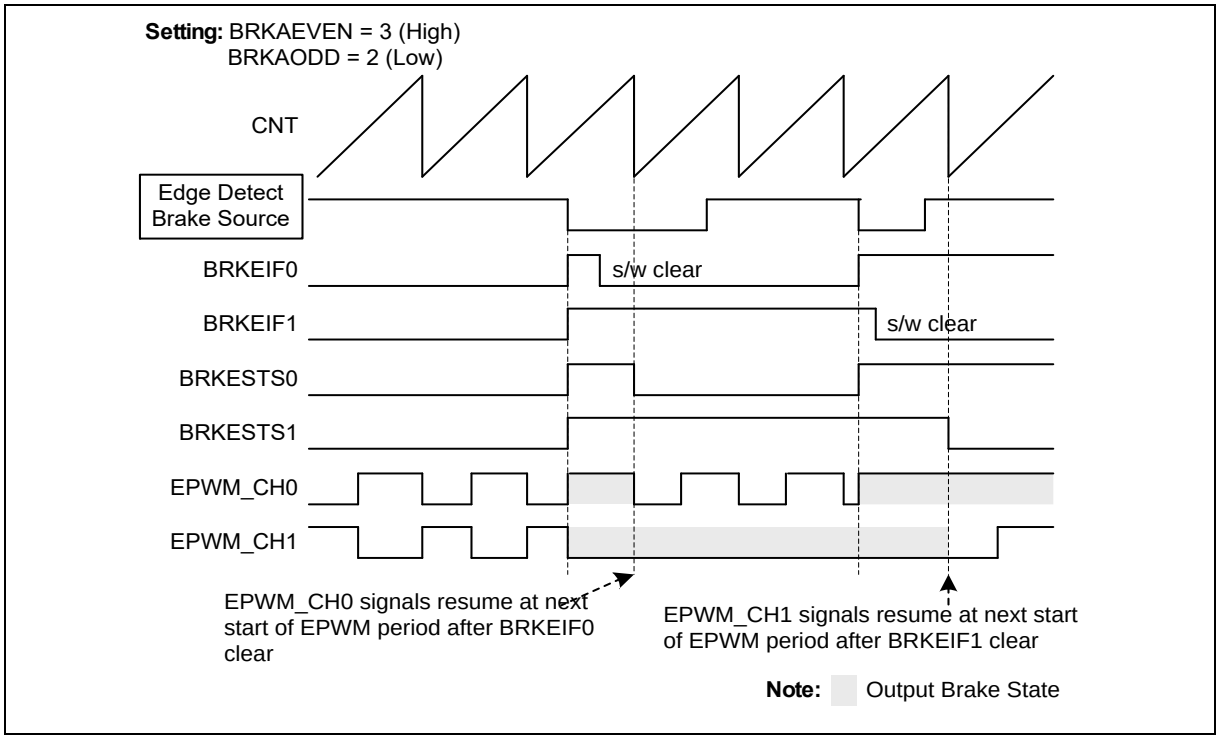


Figure 6.15-35 Edge Detector Waveform for EPWMx_CH0 and EPWMx_CH1 Pair

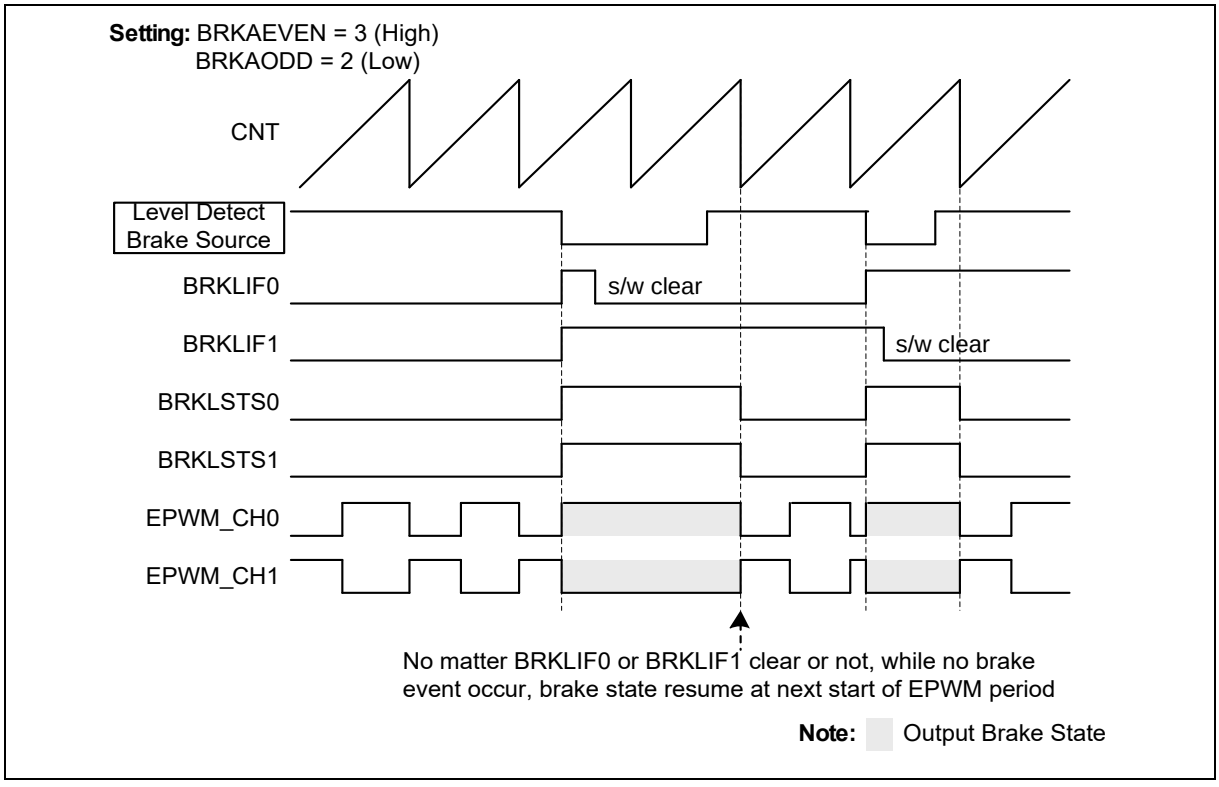


Figure 6.15-36 Level Detector Waveform for EPWMx_CH0 and EPWMx_CH1 Pair

The two kinds of detectors detect the same ten brake sources: two from external input signals, three from analog comparators(ACMP), one from EADC result monitor (EADCRM), one from system fail, one

from software triggered, one from UTCPD0_VBUS_Source, and one from UTCPD0_VBUS_Sink, which are shown in Figure 6.14-28. ACMP brake sources will be detected only when internal ACMP0_O or ACMP1_O or ACMP2_O signal from low to high.

Among the above described brake sources, the brake source coming from system fail can still be specified to four different system fail conditions. These conditions include clock fail, Brown-out detect, SRAM parity check error and Core lockup. Figure 6.14-29 shows that by setting corresponding enable bits, the enabled system fail condition can be one of the sources to issue the Brake system fail to the EPWM brake.

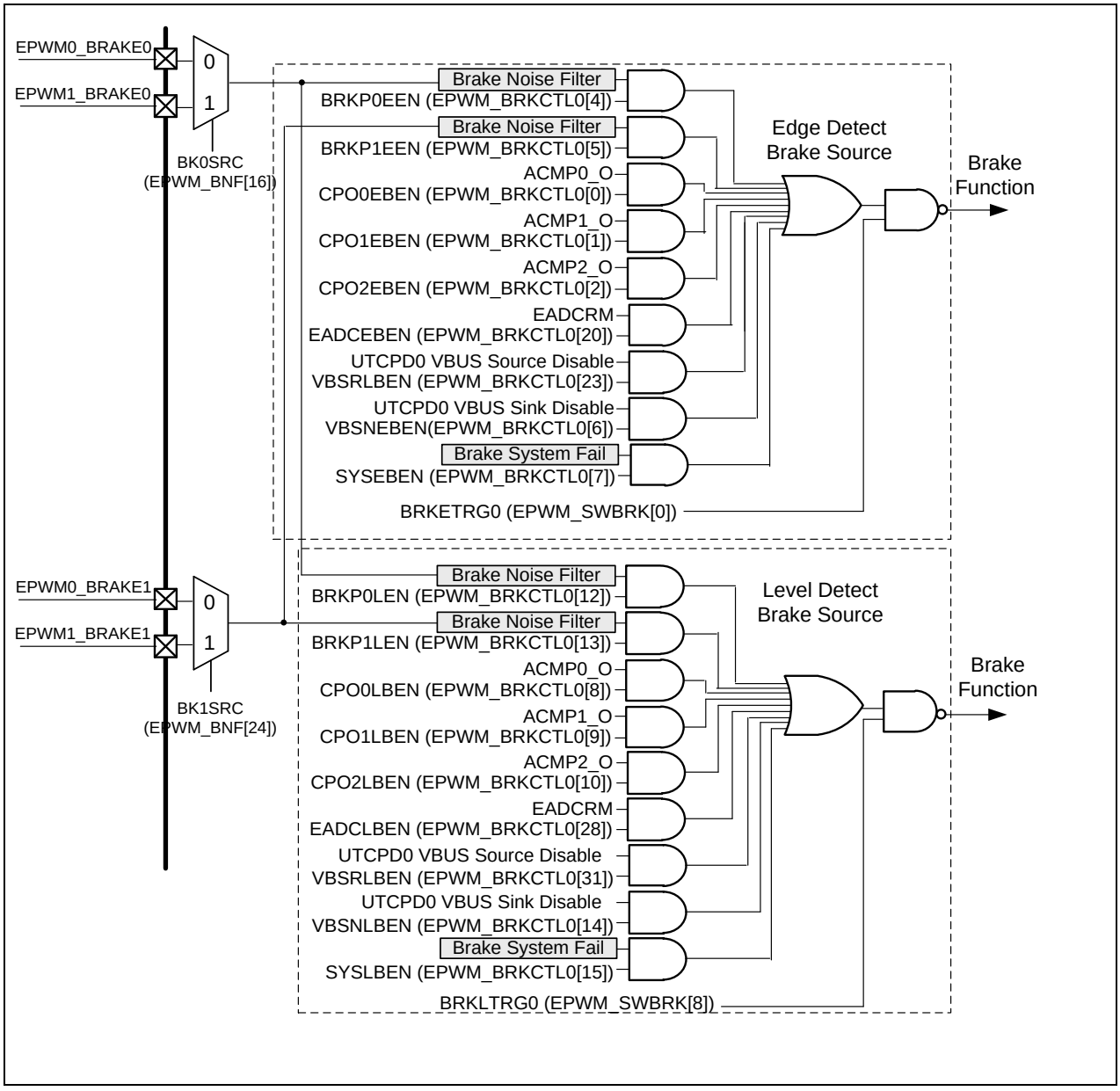


Figure 6.15-37 Brake Source Block Diagram

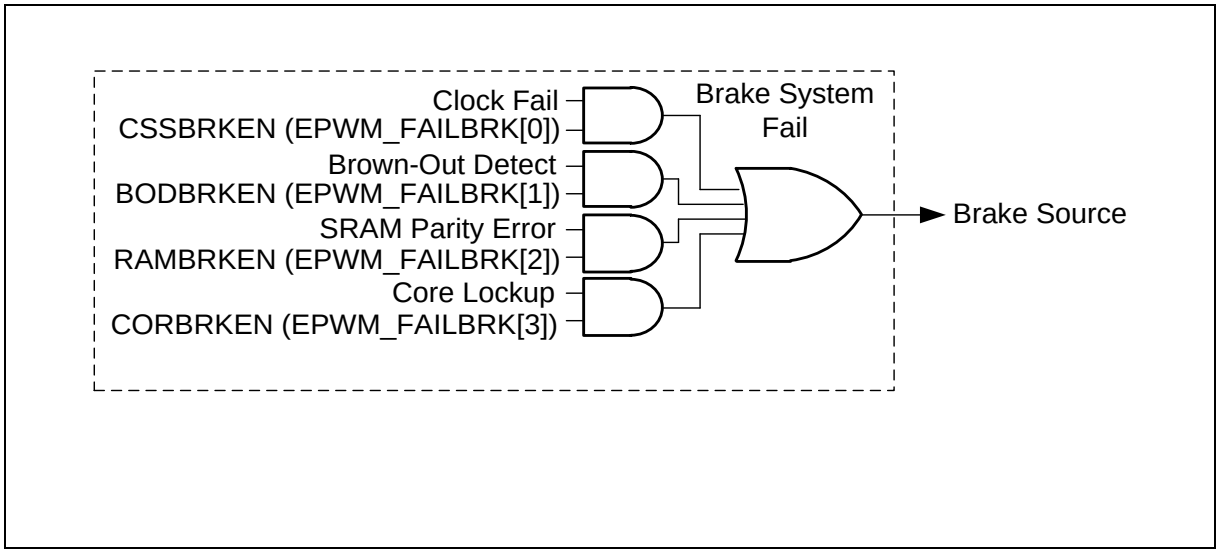


Figure 6.15-38 Brake System Fail Block Diagram

6.15.5.24 LEB Function

Leading edge blanking (LEB) function is used to blank the false trigger from brake source ACMP which may be caused by EPWM output transition. Set LEBEN (EPWM_LEBCTL[0]) to enable this function. LEB source comes from EPWM_CH0, EPWM_CH2 and EPWM_CH4, use SRCENn (EPWM_LEBCTL[10:8]) as input source enable. LEB function blanking time is decided by LEBCNT (EPWM_LEBCNT[8:0]), when LEB detected trigger edge, then blanking time will count from LEBCNT+1 to 0, the counter clock base is EPWM_CLK. If a new trigger event occurs, blanking counter will reset to LEBCNT and down count again. LEB trigger edge can be rising, falling or both rising and falling edge by setting TRGTYPE (EPWM_LEBCTL[17:16]). Figure 6.15-39 shows that LEB will blanking leading edge caused by EPWM_CH0 and EPWM_CH2.

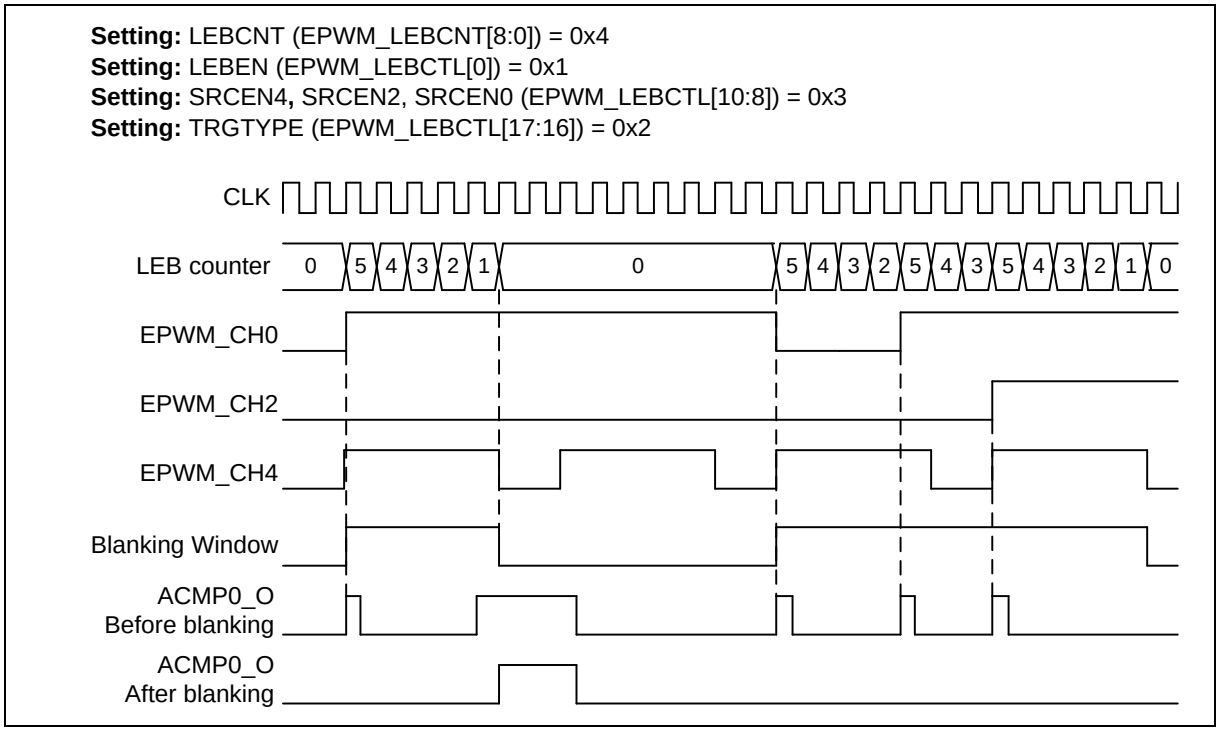
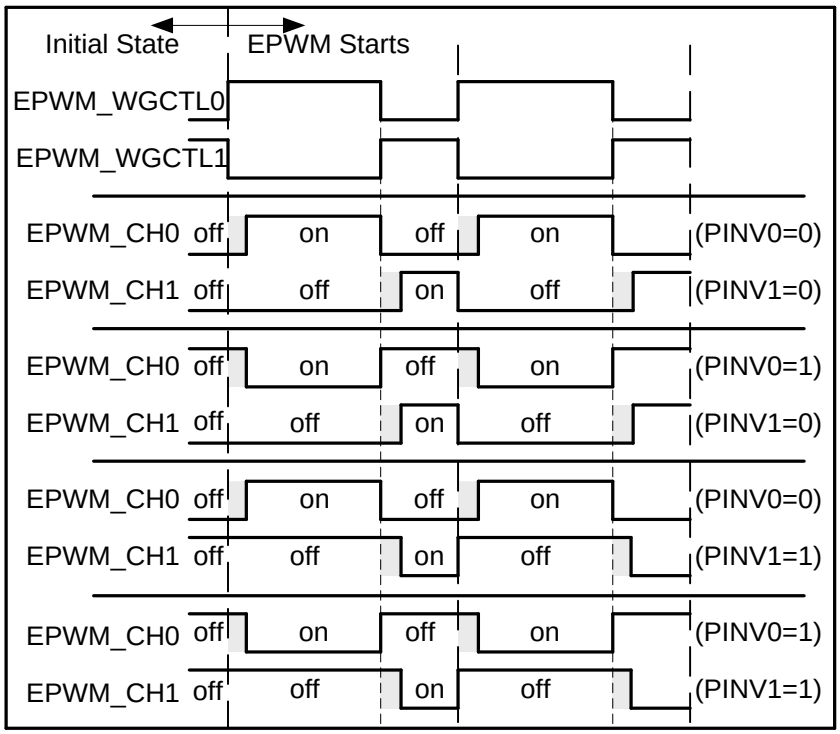


Figure 6.15-39 EPWM LEB Function Waveform

6.15.5.25 Polarity Control

Each EPWM port, from EPWM_CH0 to EPWM_CH5, has an independent polarity control module to configure the polarity of the active state of the EPWM output. By default, the EPWM output is active high. This implies the EPWM OFF state is low and ON state is high. This definition is variable through setting the EPWM Pin Polar Inverse Control (PWM_POLCTL) register, for each individual EPWM channel. Figure 6.14-30 shows the initial state before EPWM starting with different polarity settings.



Dead-time insertion; It is only effective in complementary mode

Note: PINVx: Negative Polarity control bits; It controls the EPWM output initial state and polarity, x denotes 0 or 1.

Figure 6.15-40 Initial State and Polarity Control with Rising Edge Dead-Time Insertion

6.15.5.26 EPWM Interrupt Generator

Interrupts for each EPWM show in Figure 6.15-42 and Figure 6.15-43.

The 1st EPWM interrupt (EPWM_INT) comes from EPWM complementary pair events. The counter can generate the Zero point Interrupt Flag ZIFn (EPWM_INTSTS0[n], n=0,1..5) and the Period point Interrupt Flag PIFn (EPWM_INTSTS0[8+n], n=0,1..5). When EPWM channel n's counter equals to the comparator value stored in EPWM_CMPDATn register, the different interrupt flags will be triggered depending on the counting direction. If the matching occurs at up–count direction, the Up Interrupt Flag CMPUIFn (EPWM_INTSTS0[16+n], n=0,1..5) is set and if matching at the opposite direction, the Down Interrupt Flag CMPDIFn (EPWM_INTSTS0[24+n], n=0,1..5) is set. If the corresponding interrupt enable bits are set, the trigger events will generate interrupt signals.

EPWM_INT can use the EPWM_IFAn (n=0,1..5) register to accumulate the number of times that the interrupt flags have been triggered for each channel. By setting one of IFAEN (EPWM_IFAn[31], n=0,1..5) bit to 1 to enable accumulator, EPWM_INT will switch interrupt source from every event trigger interrupt to trigger interrupt once every accumulate times.

By setting the IFASEL (EPWM_IFAn[29:28], n=0,1..5) bits, user can select one of the 4 interrupt sources to accumulate interrupt flag times for each channel, and the number of times interrupt flags will compare with IFACNT (EPWM_IFAn[15:0], n=0,1..5) bits. When interrupt accumulator equals IFACNT then set IFAIFn (EPWM_AINTSTS[n], n=0,1..5) bits as EPWM_INT signal if user enable IFAIENn (EPWM_AINTEN[n], n=0,1..5) bits. Accumulator interrupt of each channel can also be as request source of PDMA. Figure 6.14-31 is an example of channel 0 using EPWM_IFA0 register to output EPWM_INT once every IFACNT0+1 times interrupt events occurred.

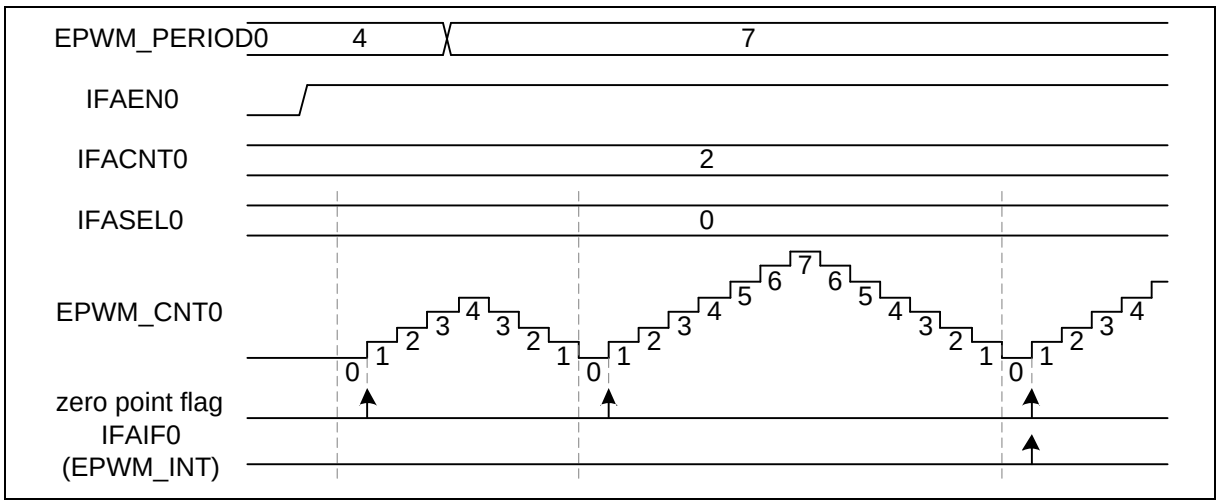


Figure 6.15-41 EPWMx_CH0 Accumulate Interrupt Waveform

The 2nd interrupt is the capture interrupt (CAP_INT). It shares the EPWM_INT vector in NVIC. The CAP_INT can be generated when the CRLIFn (EPWM_CAPIF[5:0]) flag is triggered and the Capture Rising Interrupt Enable bit CAPRIENn (EPWM_CAPIEN[5:0]) is set to 1. Or in the falling edge condition, the CAP_INT can be generated when the CFLIFn (EPWM_CAPIF[13:8]) is triggered and the Capture Falling Interrupt Enable bit CAPFIENn (EPWM_CAPIEN[13:8]) is set to 1.

The 3rd interrupt is the brake interrupt (BRK_INT). The details of the BRK_INT is described in the EPWM Brake section. Figure 6.15-42 demonstrates the architecture of the EPWM interrupts.

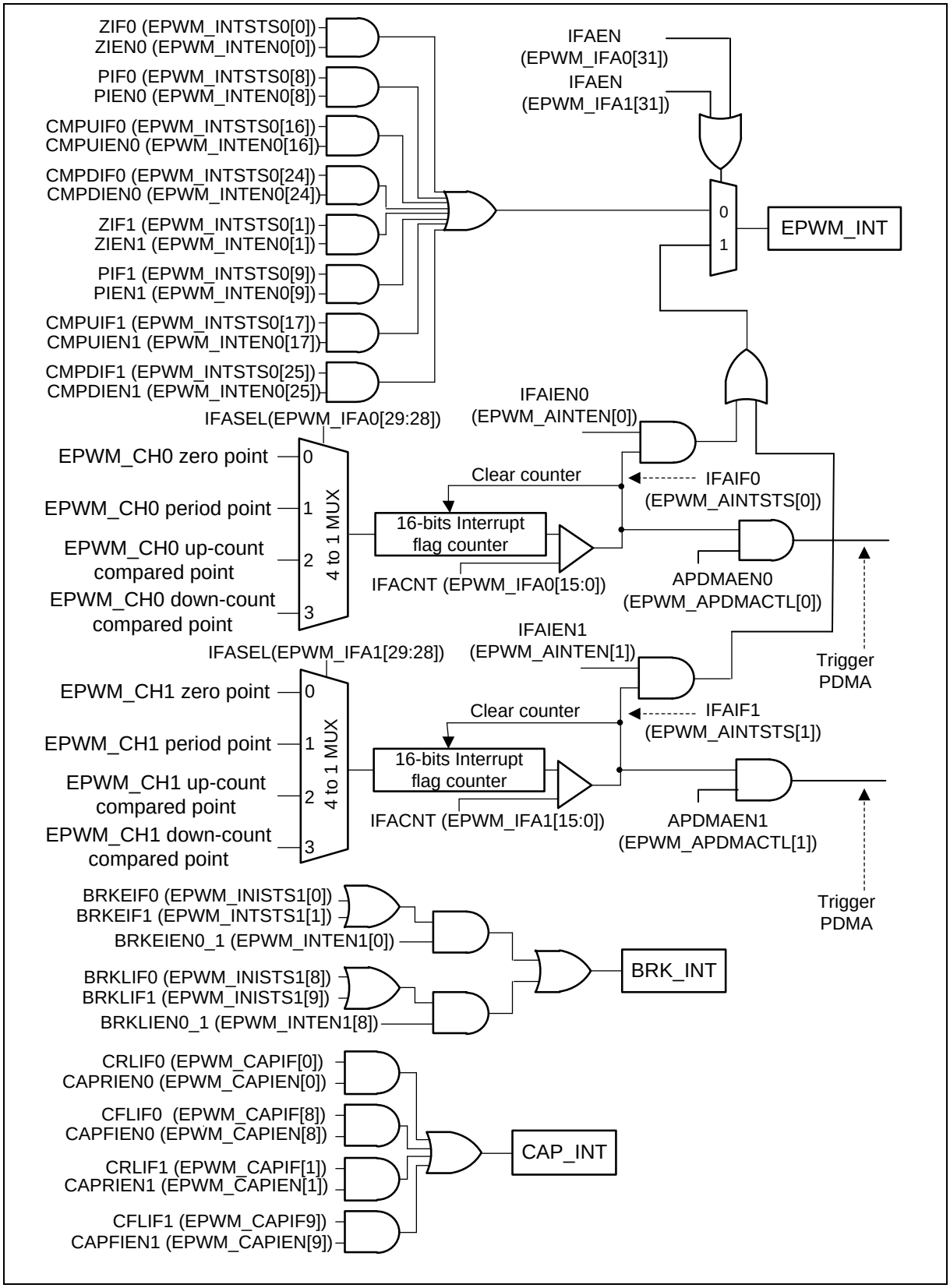


Figure 6.15-42 EPWMx_CH0 and EPWMx_CH1 Pair Interrupt Architecture Diagram

The 4th interrupt is Fault Detect Function interrupt (FLT_INT). It shares the EPWM_INT vector in NVIC. Figure 6.15-43 shows the FLT_INT can be generate when FDIENn (EPWM_FDIEN[n], n=0,1..5) is set to 1 and FDIFn (EPWM_FDSTS[n], n=0,1..5) is set to 1 by detecting output open short.

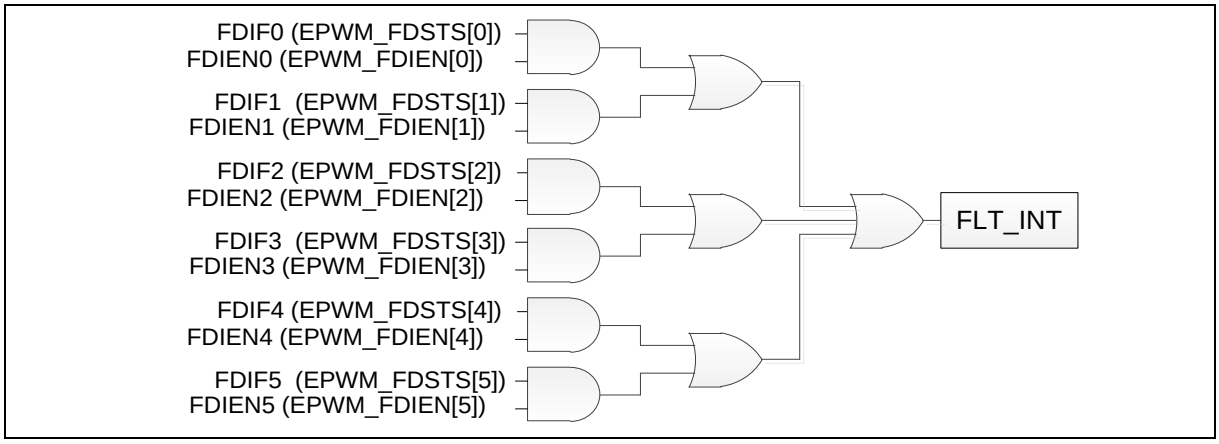


Figure 6.15-43 Fault Detect Function Interrupt Architecture Diagram

6.15.5.27 EPWM Trigger EADC/LPADC/DAC Generator

EPWM can be one of the EADC/LPADC conversion trigger source. Each EPWM pair channels share the same trigger source. Setting TRGSELn bit of EPWM_EADCTS0 and EPWM_EADCTS1 registers is to select the trigger sources, where TRGSELn bit is TRGSEL0, TRGSEL1, …, and TRGSEL5, which are located in EPWM_EADCTS0[4:0], EPWM_EADCTS0[12:8], EPWM_EADCTS0[20:16], EPWM_EADCTS0[28:24], EPWM_EADCTS1[4:0] and EPWM_EADCTS1[12:8], respectively. Setting TRGENn bit of EPWM_EADCTS0 and EPWM_EADCTS1 registers is to enable the trigger output to EADC, where TRGENn bit is TRGEN0, TRGEN1, …, TRGEN5, which are located in EPWM_EADCTS0[7], EPWM_EADCTS0[15], EPWM_EADCTS0[23], EPWM_EADCTS0[31], EPWM_EADCTS1[7] and EPWM_EADCTS1[15], respectively. The number n (n = 0,1..5) denotes EPWM channel number. Figure 6.14-33 is an example of EPWM_CH0 and EPWM_CH1. Any one of EPWM trigger event can trigger LPADC if correlated TRGEN is enabled. Figure 6.15-45 shows the EPWM trigger LPADC source block diagram. EPWM can trigger EADC/LPADC to start conversion in different timings by setting PERIOD and CMPDAT. Figure 6.14-35 is the trigger EADC/LPADC timing waveform in the up-down counter type.

There are 18 EPWM events can be selected as the trigger source for one pair of channels which shown in Figure 6.14-33. Figure 6.15-46 shows trigger EADC block diagram with prescaler. By setting PSCENn (EPWM_EADCPSCCTR[n], n=0,1..5) bit to 1, EPWM will trigger EADC when the number of trigger events is equal to EADCPSCn (n=0,1..5). Reading PSCNTn (n=0,1..5) can know how many events happened. If PSCENn (n=0,1..5) is 0, user also can write initial data to PSCNTn (n=0,1..5) and pre-scale counter will start from this data. If PSCENn (n=0,1..5) is set from 1 to 0, PSCNTn (n=0,1..5) will reset to 0. Figure 6.14-35 is the trigger EADC timing waveform in the up-down counter type.

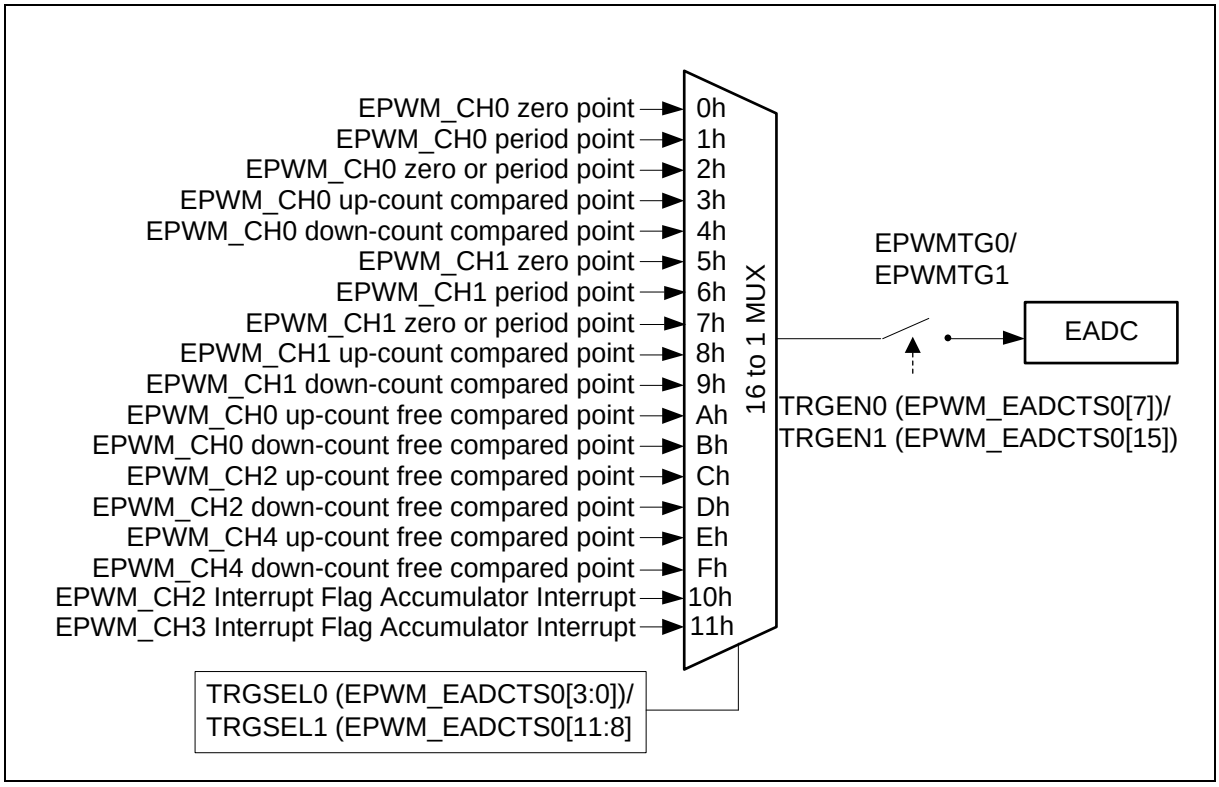


Figure 6.15-44 EPWMx_CH0 and EPWMx_CH1 Pair Trigger EADC Events

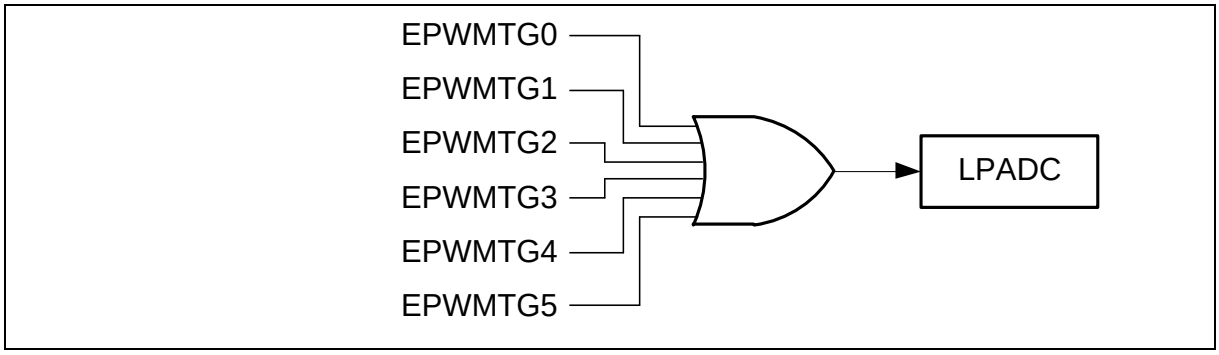


Figure 6.15-45 EPWM Trigger LPADC Block Diagram

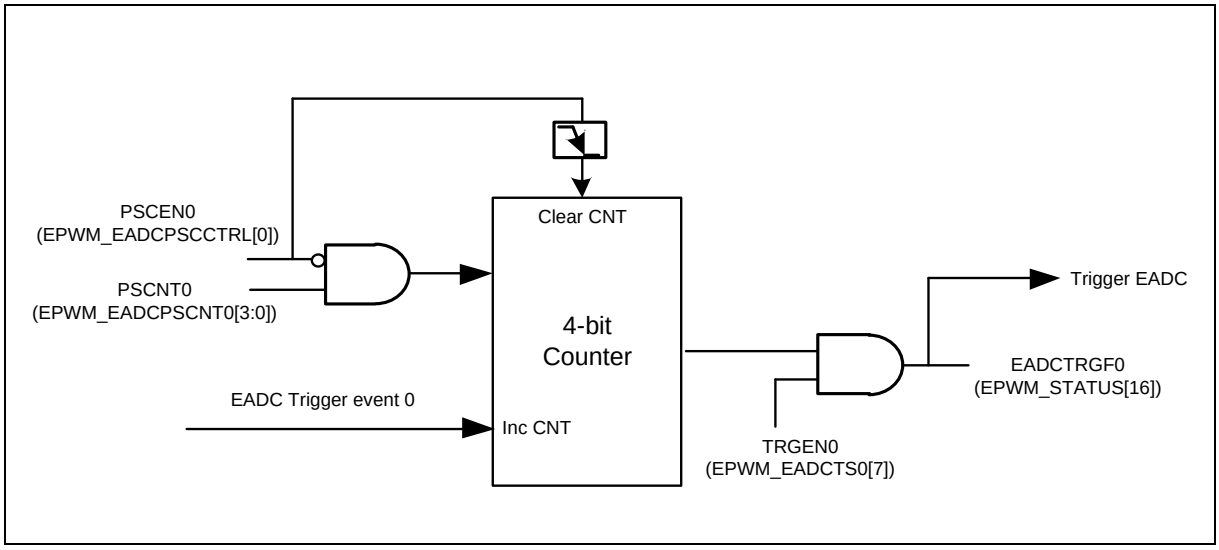


Figure 6.15-46 EPWMx_CH0 Trigger EADC/LPADC Block Diagram

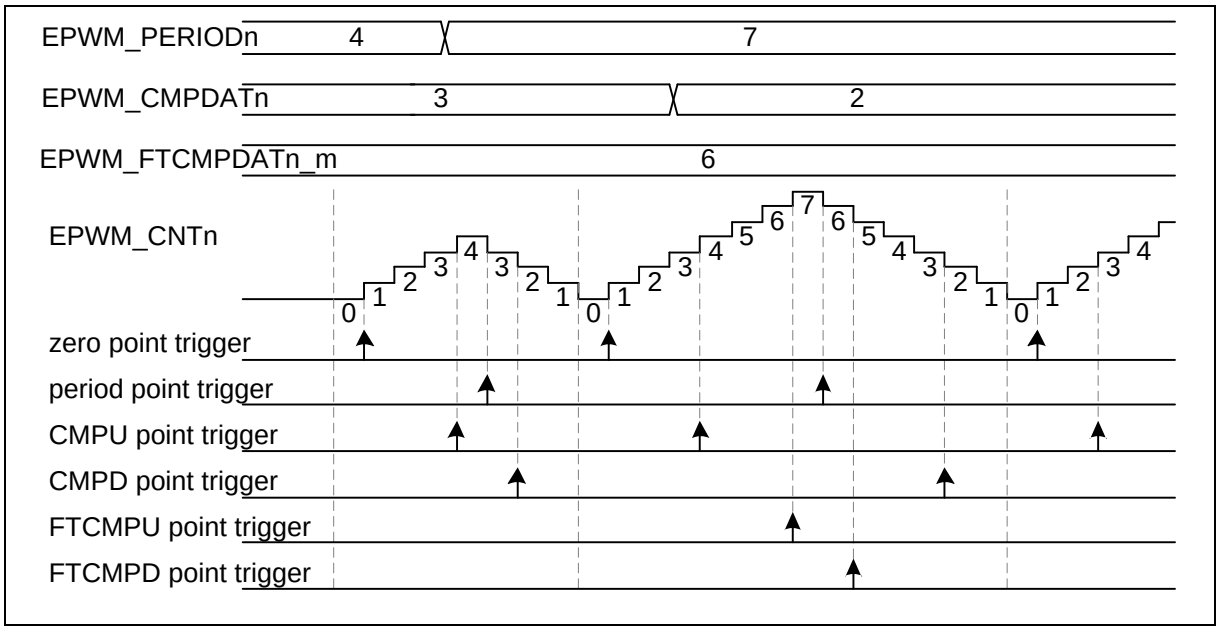


Figure 6.15-47 EPWM Trigger EADC/LPADC in Up-Down Counter Type Timing Waveform

EPWM can also be used to trigger DAC conversion. Each EPWM pair channel (CH0 and CH1, CH2 and CH3, CH4 and CH5) generates a trigger signal. Using the EPWM Trigger DAC Enable Register (EPWM_DACTRGEN) can decide at which points to trigger DAC. The timing of the EPWM triggering DAC is similar to those for triggering EADC. However, DAC triggering function does not include the triggering events from comparison with FTCMPDAT, that is, there are no trigger points the same as FTCMPU and FTCMPD which are shown in EADC triggering.

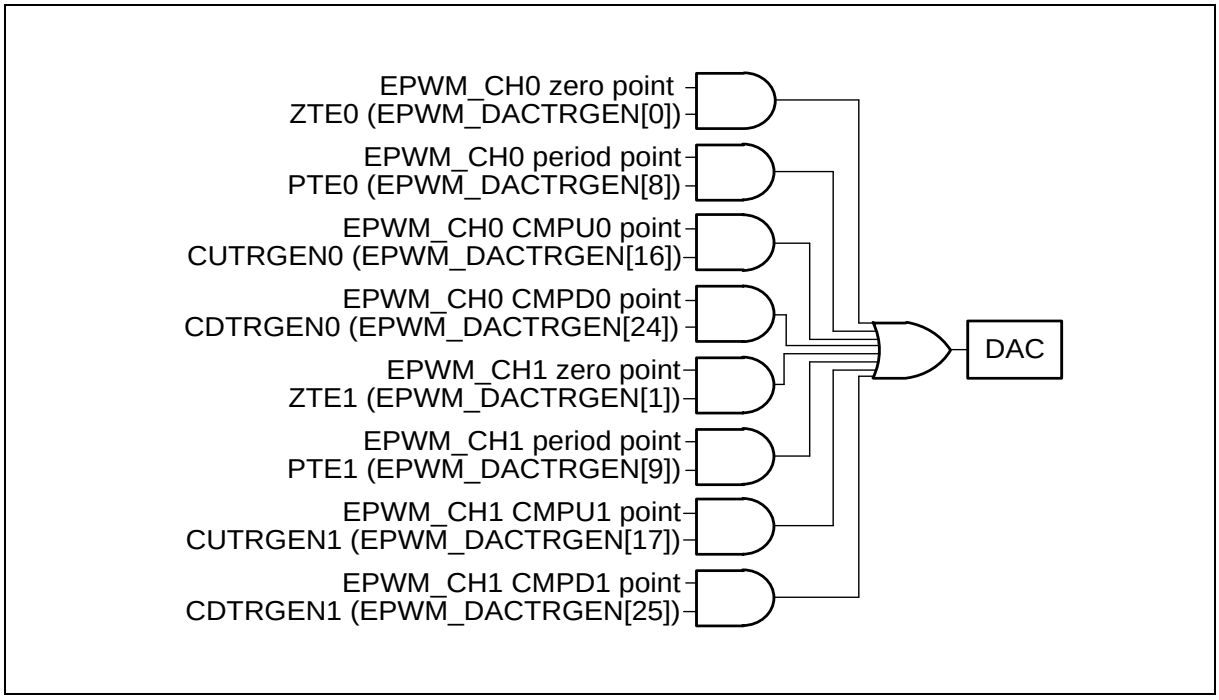


Figure 6.15-48 EPWM_CH0 and EPWM_CH1 Pair Trigger DAC Block Diagram

6.15.5.28 Capture Operation

The channels of the capture input and the EPWM output share the same pin and counter. The counter can operating in up or down counter type. The capture input will be filtered by a 3-bit noise filter. User can enable the noise filter function by CAPNFEN (EPWM_CAPNFn[0], n=0~5) reigster, and noise filter sampling clock can be selected by setting CAPNFSEL bits of (EPWM_CAPNFn[6:4], n=0~5) register to fit different noise properties. Moreover, by setting the CAPNFCNT (EPWM_CAPNFn[10:8], n=0~5) bits, user can define how many sampling clock cycles a filter will recognize the effective edge of the capture input signal.The capture function will always latch the EPWM counter to the RCAPDATn (EPWM_RCAPDATn[15:0]) bits or the FCAPDATn (EPWM_FCAPDATn[15:0]) bits, if the input channel has a rising transition or a falling transition, respectively. The capture function will also generate an interrupt CAP_INT (using EPWM_INT vector) if the rising or falling latch occurs and the corresponding channel n's rising or falling interrupt enable bits are set, where the CAPRIENn (EPWM_CAPIEN[5:0]) bit is for the rising edge and the CAPFIENn (EPWM_CAPIEN[13:8]) bit is for the falling edge. When rising or falling latch occurs, the corresponding EPWM counter may be reloaded with the value of EPWM_PERIODn register, depending on the setting of RCRLDENn or FCRLDENn bits (where RCRLDENn and FCRLDENn are located at EPWM_CAPCTL[21:16] and EPWM_CAPCTL[29:24], respectively). Note that the corresponding GPIO pins must be configured as the capture function by enable the CAPINENn (EPWM_CAPINEN[5:0]) bits for the corresponding capture channel n. Figure 6.15-49 is the capture block diagram of channel 0.

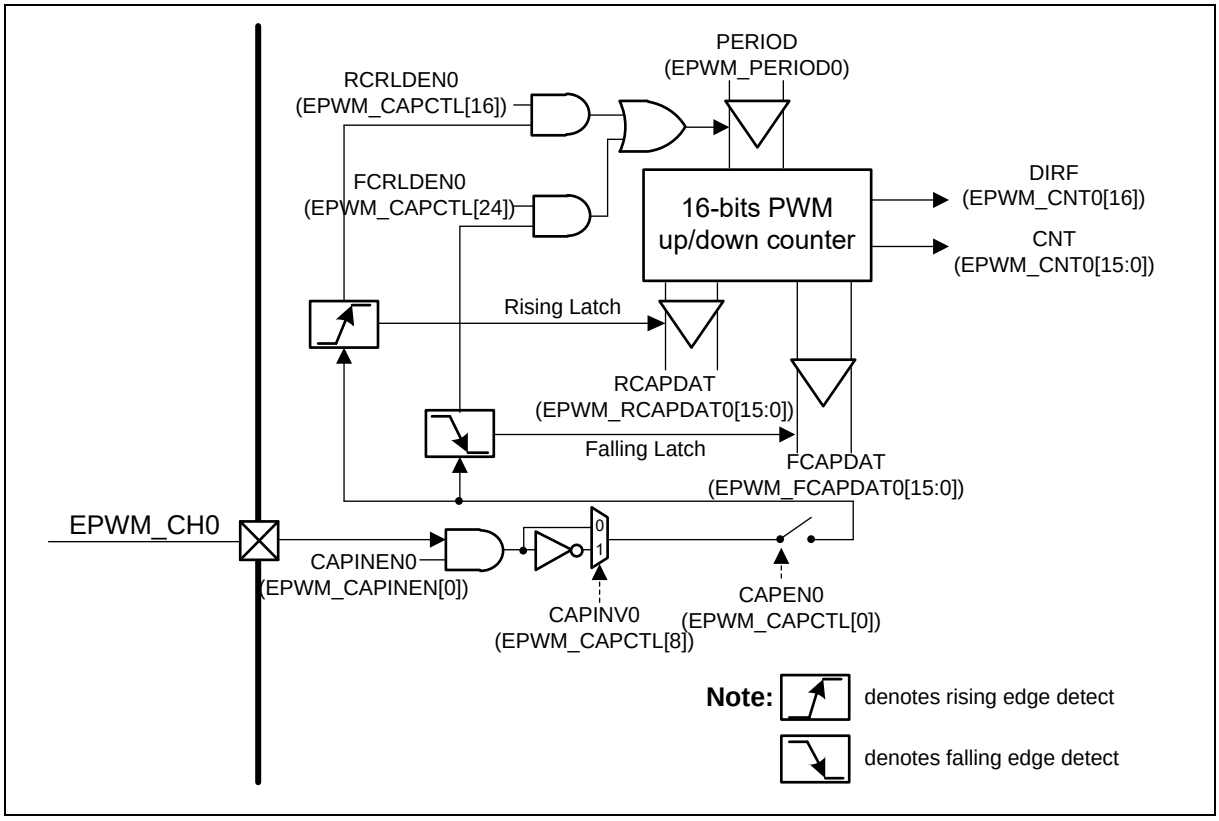


Figure 6.15-49 EPWM_CH0 Capture Block Diagram

Figure 6.15-50 illustrates the capture function timing. In this case, the capture counter is set as EPWM down counter type and the PERIOD is set to 8 so that the counter counts in the down direction, from 8 to 0. When detecting a falling edge at the capture input pin, the capture function latches the counter value to the EPWM_FCAPDATn register. When detecting the rising edge, it latches the counter value to the EPWM_RCAPDATn register. In this timing diagram, when the falling edge is detected at the first time, the capture function will reload the counter value from the PERIOD setting because the FCRLDENn bit is enabled. But at the second time, the falling edge does not result in a reload because of the disabled FCRLDENn bit. In this example, the counter also reloads at the rising edge of the capture input because the RCRLDENn bit is enabled, too.

Moreover, if the case is setup as the up counter type, the counter will reload the value zero and count up to the value PERIOD.

Figure 6.15-50 also illustrates the timing example for the interrupt and interrupt flag generation. When the rising edge at channel n is detected, the corresponding CRLIFn (EPWM_CAPIF[5:0]) bit is set by hardware. Similarly, a falling edge detection at chnnel n causes the corresponding CFLIFn (EPWM_CAPIF[13:8]) bit is set by hardware. CRLIFn and CFLIFn bits can be cleared by software by writing ‘1’. If the CRLIFn bit is set and the CAPRIENn bit is enabled, the capture function generates an interrupt. If the CFLIFn bit is set and the CAPFIENn bit is enabled, the interrupt also happens.

A condition which is not shown in Figure 6.15-50 is: if the rising latch happens again when the CRLIFn bit is already set, the overrun status CRLIFOVn (EPWM_CAPSTS[5:0]) bit will be set to 1 by hardware to indicate the CRLIF flag overrunning. Also, if the falling latch happens again, the same hardware operation occurs for the CFLIF interrupt flag and the overrun status CFLIFOVn (EPWM_CAPSTS[13:8]).

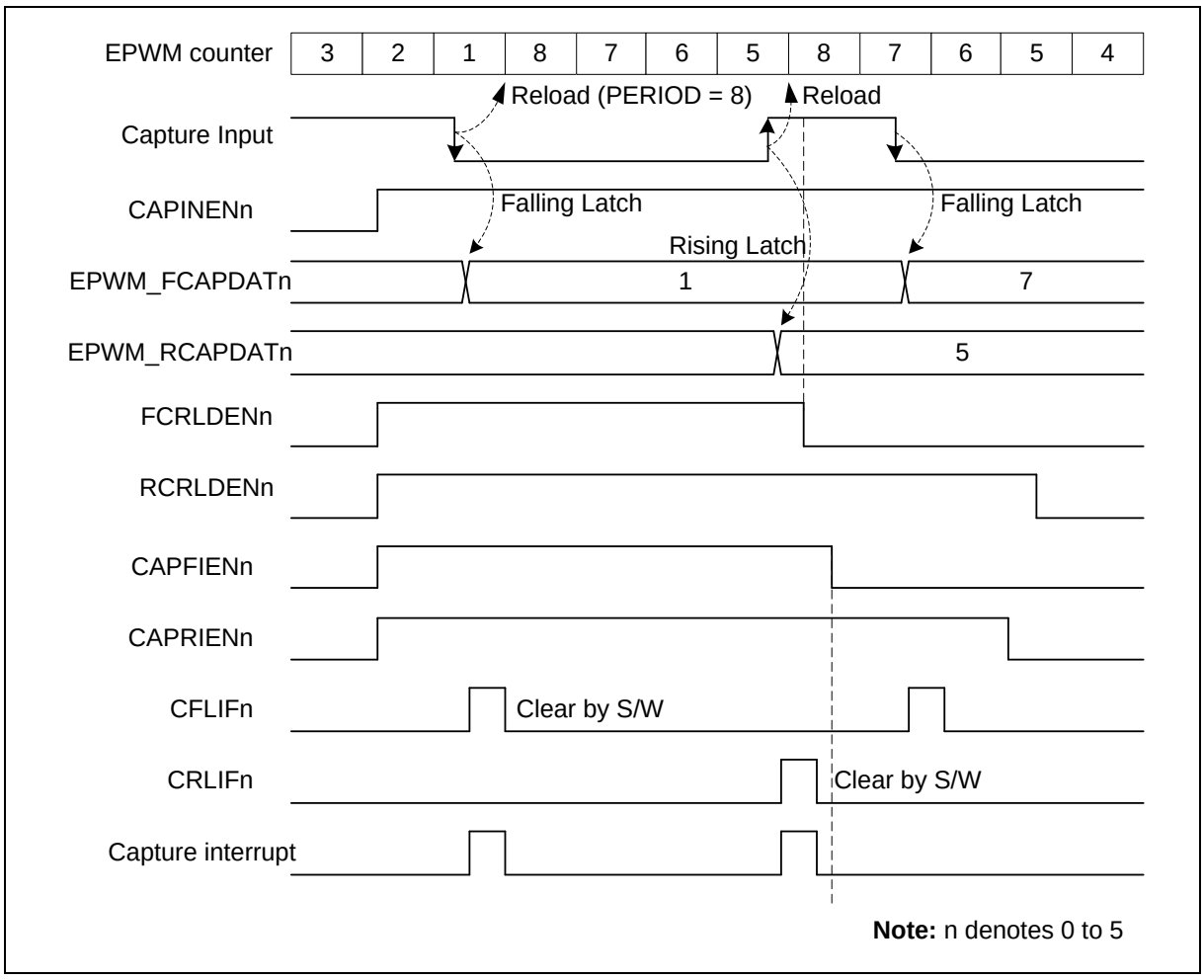


Figure 6.15-50 Capture Operation Waveform

The capture pulse width meeting the following conditions can be calculated according to the formula.

1. The capture positive or negative pulse width is shorter than a counter period.
2. The counter operates in down counter type.
3. The counter can be reloaded by both falling and rising capture events through setting FCRLDENn and RCRLDENn bits of EPWM_CAPCTL register to 1.

For the negative pulse case, the channel low pulse width is calculated as (EPWM_PERIODn + 1 - EPWM_RCAPDATn) EPWM counter time, where one EPWM counter time is (CLKPSC+1) * EPWMx_CLK clock time. In Figure 6.15-50, the low pulse width is 8+1-5 = 4 EPWM counter time.

For the positive pulse case, the channel high pulse width is calculated as (EPWM_PERIODn + 1 - EPWM_FCAPDATn) EPWM counter time, where one EPWM counter time is (CLKPSC+1) * EPWMx_CLK clock time. In Figure 6.15-50, the high pulse width is 8+1-7 = 2 EPWM counter time.

6.15.5.29 Capture Trigger PDMA Function

The EPWM module supports the PDMA transfer function when operating in the capture mode. When the corresponding PDMA enable bit CHENn_m (CHEN0_1 at EPWM_PDMACTL[0], CHEN2_3 at EPWM_PDMACTL[8] and CHEN4_5 at EPWM_PDMACTL[16], where n and m denote complement pair channels) is set, the capture module will issue a request to PDMA controller when the preceding capture event has happened. Figure 6.14-39 shows Capture Trigger PDMA Function Architecture Diagram for EPWM_CH0 and EPWM_CH1 Pair.

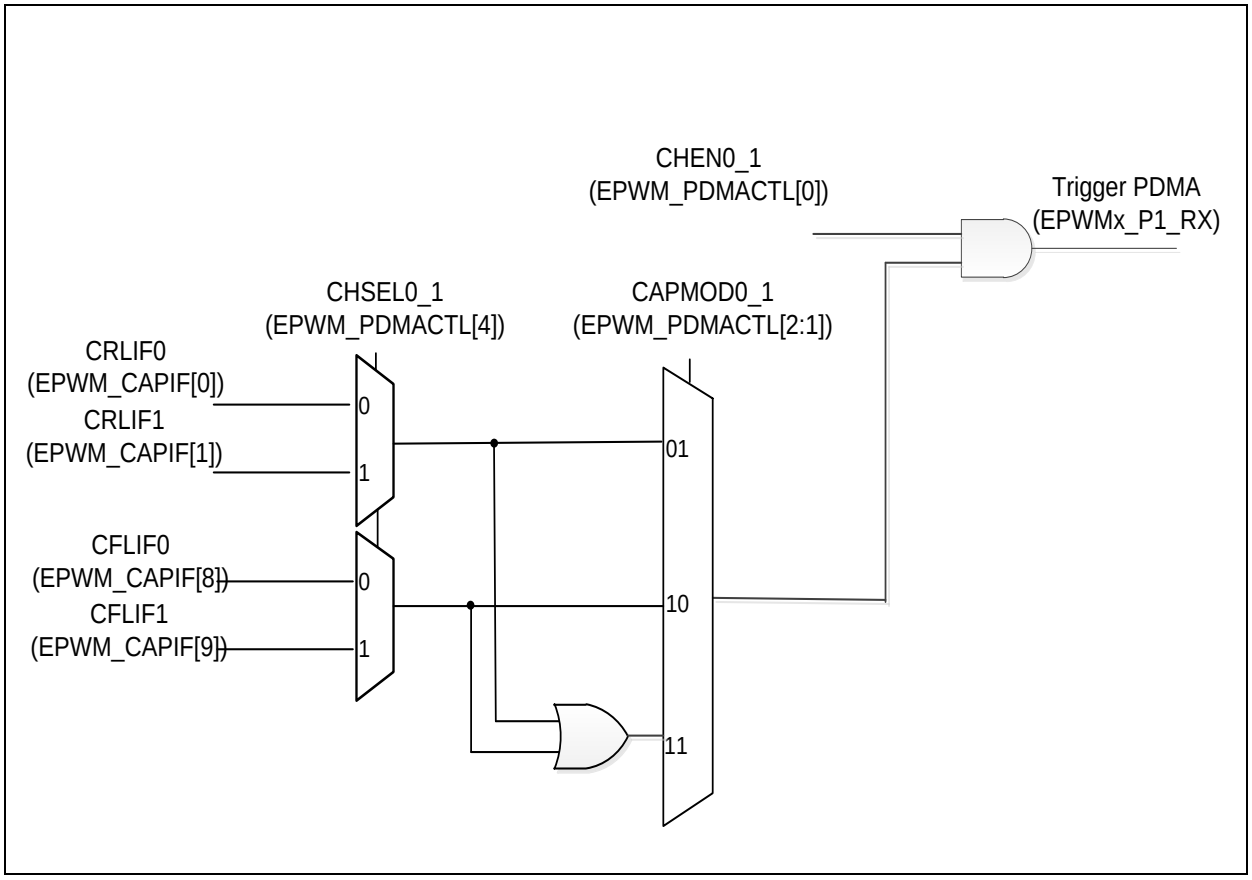


Figure 6.15-51 Capture Trigger PDMA Function Architecture Diagram for EPWM_CH0 and EPWM_CH1 Pair

The PDMA controller will issue an acknowledgement to the capture module after it has read back the CAPBUF (EPWM_PDMACAPn_m[15:0], n, m denotes complement pair channels) register in the capture module and has sent the register value to the memory. By setting CAPMODn_m (CAPMOD0_1 at EPWM_PDMACTL[2:1], CAPMOD2_3 at EPWM_PDMACTL[10:9] and CAPMOD4_5 at EPWM_PDMACTL[18:17]) bits, the PDMA can transfer the rising edge captured data or falling edge captured data or both of them to the memory. When using the PDMA to transfer both of the falling and rising edge data, remember to set CAPORDn_m (CAPORD0_1 at EPWM_PDMACTL[3], CAPORD2_3 at EPWM_PDMACTL[11] and CAPORD4_5 at EPWM_PDMACTL[19]) bit to decide the order of the transferred data (falling edge captured is first or rising edge captured first). The complement pair channels share a PDMA channel. Therefore, a selection bit CHSELn_m (CHSEL0_1 (EPWM_PDMACTL[4]), CHSEL2_3 (EPWM_PDMACTL[12]) and CHSEL4_5 (EPWM_PDMACTL[20])) bit is used to decide either channel n or channel m can be serviced by the PDMA channel.

Figure 6.14-40 is capture PDMA waveform. In this case, the CHSEL0_1 (EPWM_PDMACTL[4]) bit is set to 0. Hence the PDMA will service channel 0 for the capture data transfer. CAPMOD0_1 (EPWM_PDMACTL[2:1]) bits are set to 3. That means both of the rising and falling edge captured data will be transferred to the memory. The CAPORD0_1 (EPWM_PDMACTL[3]) bit is set to 1, so the rising edge data will be the first data to transfer and following is the falling edge data to transfer. As shown in Figure 6.14-40, the last assertions of the CRLIF0 and CFLIF0 signal have some overlap. The value of EPWM_RCAPDAT0 register is 11 will be loaded to EPWM_PDMACAP0_1 register to wait for transfer but not the EPWM_FCAPDAT0 value. The EPWM_PDMACAP0_1 register saves the data which will be transferred to the memory by PDMA. The HWDATA in this figure denotes the data which are being

transferred by PDMA.

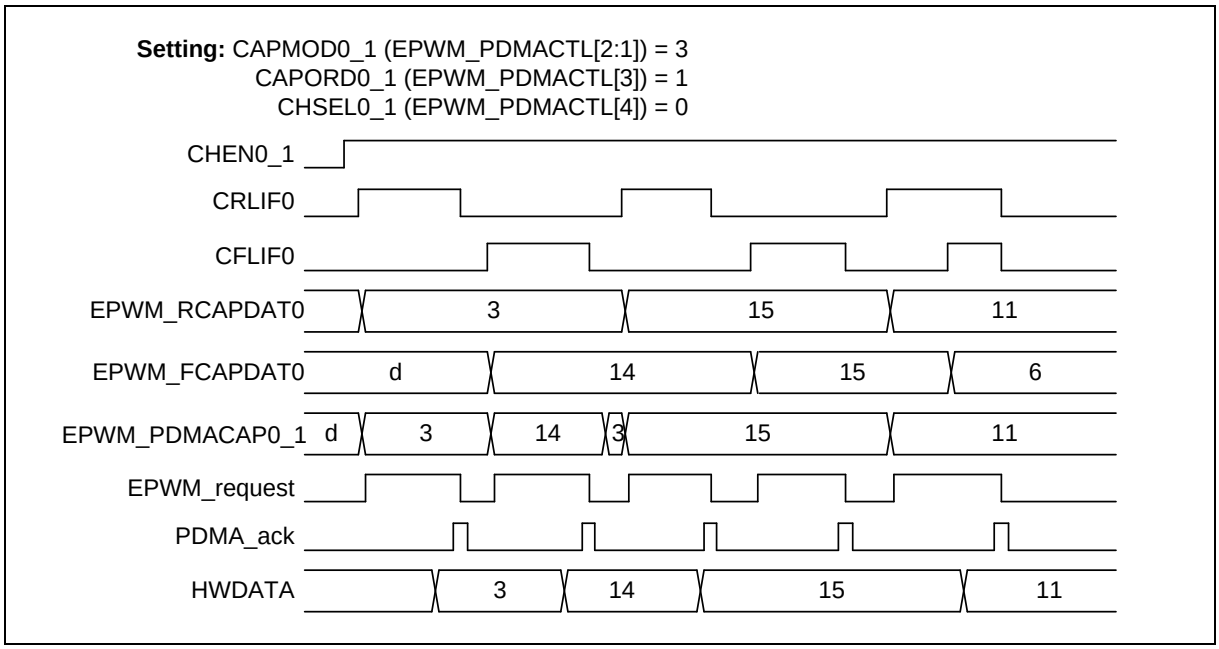


Figure 6.15-52 Capture PDMA Operation Waveform of Channel 0

6.15.5.30 Accumulator Trigger PDMA Function

The EPWM module supports the PDMA transfer function when accumulator interrupt happened. Figure 6.14-41 shows accumulator trigger PDMA function architecture diagram. When the corresponding PDMA enable bit APDMAENn (EPWM_APDMACTL[n], n=0,1..5) is set, accumulator module will send a request to PDMA controller when accumulator interrupt has happened, meaning that IFAIFn (EPWM_AINTSTS[n], n=0,1..5) is set 1. The PDMA controller will issue an acknowledge to accumulator after it has read memory data and send the data to the particular register (EPWM_PERIOD, etc.). So, user can use this function to change accumulator interrupt frequency.

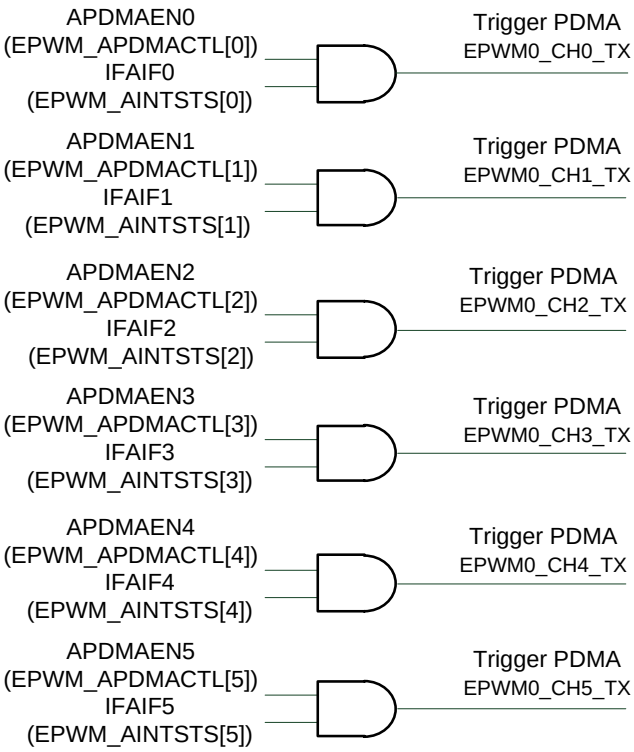


Figure 6.15-53 Accumulator Trigger PDMA Function Architecture Diagram

6.15.5.31 Accumulator Stop Mode

The EPWM module supports the Accumulator Stop Mode to stop counting when accumulator interrupt happened. Figure 6.12-22 shows Accumulator Stop Mode waveform. When the corresponding STPMOD (EPWM_IFAn[24], n=0,1..5) is set, accumulator module will disable CNTENn (EPWM_CNTEN[n], n= 0,1..5) after counter finishes whole period.

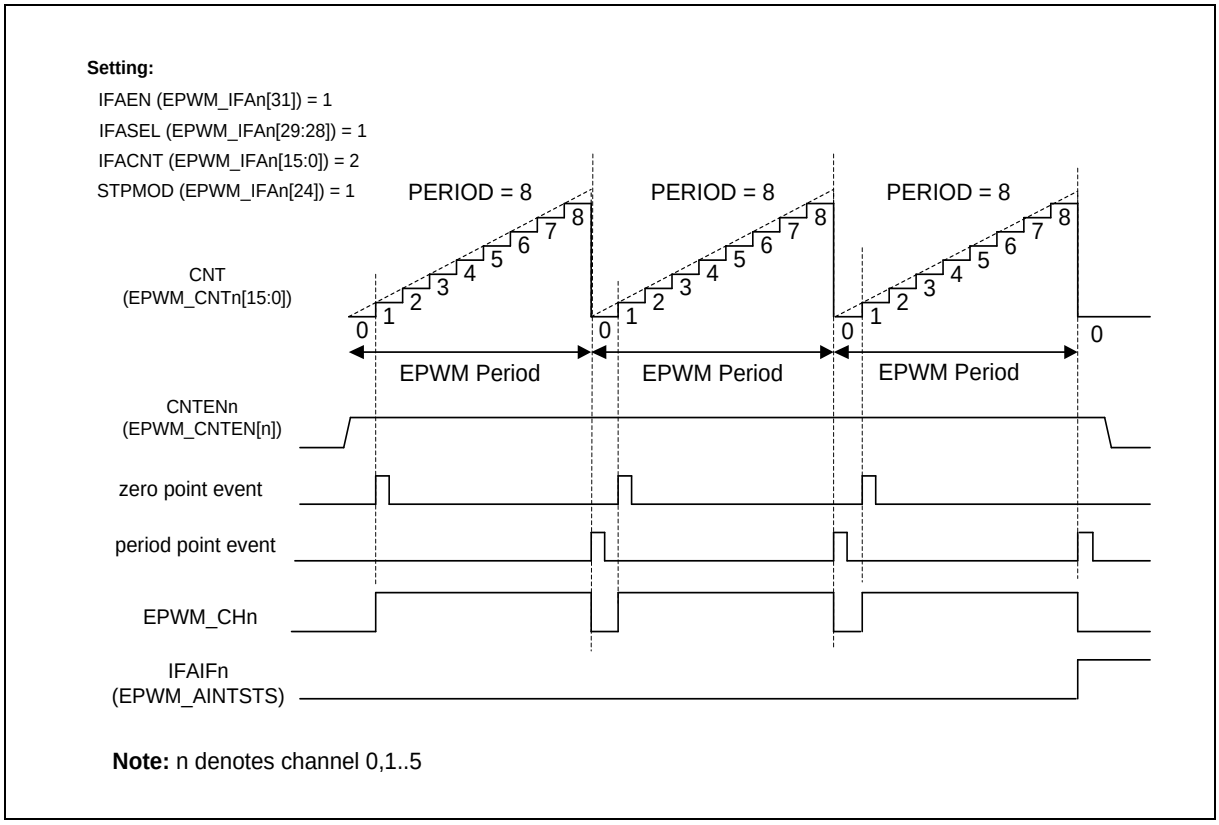


Figure 6.15-54 EPWM Accumulator Stop Mode Waveform

6.15.5.32 Fault Detect Function

The EPWM module supports Fault Detect Function to detect output short condition. Figure 6.15-55 shows fault detect function architecture. When the corresponding channel fault detect enable bit FDENn (EPWM_FDEN[n], n=0,1..5) is set to 1, two statement will start detection. One is that EPWM output becomes from low to high or from high to low. The other is that POENn (EPWM_POEN[n], n= 0,1..5) becomes from 0 to 1 or from 1 to 0.

6.15.5.33 External Pin Event Trigger

External pin event can trigger EPWM counter. Setting EXTETEN (EPWM_EXTETCTLn[0], n=0,1..5) bit to 1 can enable this function. Setting CNTACTS (EPWM_EXTETCTLn[5:4], n=0,1..5) can choose counter action: reset, start or reset and start. Setting EXTTRGS (EPWM_EXTETCTLn[11:8], n=0,1..5) can select external trigger source from INT0~INT7. Both CNTACTS (EPWM_EXTETCTLn[5:4], n= 0,1..5) and EXTTRGS (EPWM_EXTETCTLn[11:8], n=0,1..5) need to be set before setting EXTETEN (EPWM_EXTETCTLn[0], n=0,1..5) to 1.

Trigger event comes from detecting INT0~INT7 rising, falling or both of rising and falling. More detailed setting can be found in “INT Edge Detect and Trigger Function” section of the GPIO chapter.

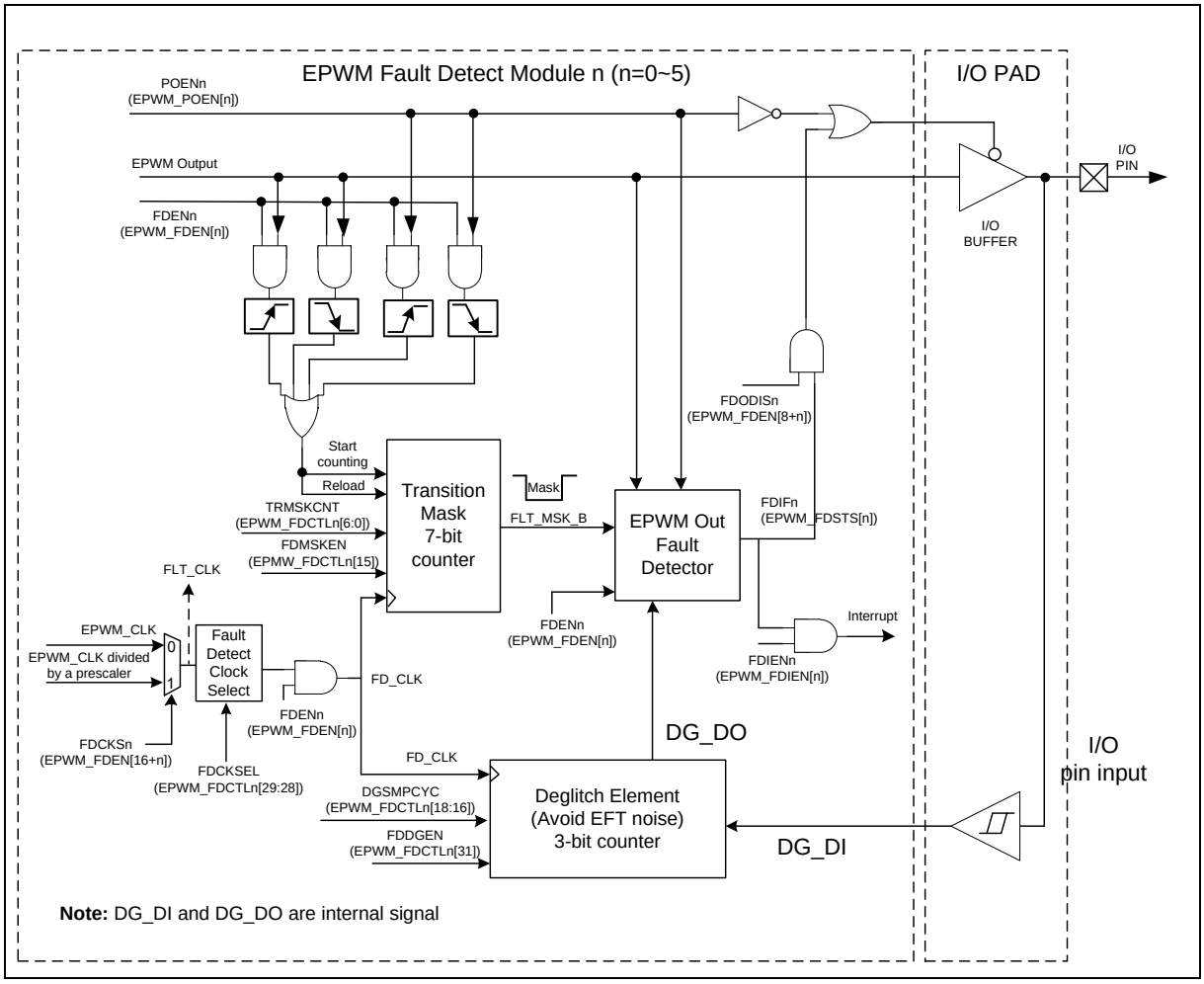


Figure 6.15-55 Fault Detect Function Architecture

Figure 6.15-56 shows Mask Function waveform example. Setting FDMSKEN (EPWM_FDCTLn[15], n=0,1..5) will enable the MASK Function. The detection will disable in the beginning until mask counter count from 0 to TRMSKCNT (EPWM_FDCTLn[6:0], n=0,1..5).

Figure 6.15-57 shows Deglitch Function waveform example. Setting FDDGEN (EPWM_FDCTLn[31], n=0,1..5) will enable Deglitch Function. The feedback signal will be sampled DGSMPCYC (EPWM_FDCTLn[18:16], n=0,1..5) +1 times and become valid input to be used for detection. FDCKSn (EPWM_FDEN[n], n=0,1..5) and FDCKSEL (EPWM_FDCTLn[29:28], n=0,1..5) are clock setting for MASK and Deglitch Function. For more detail, please see the register description.

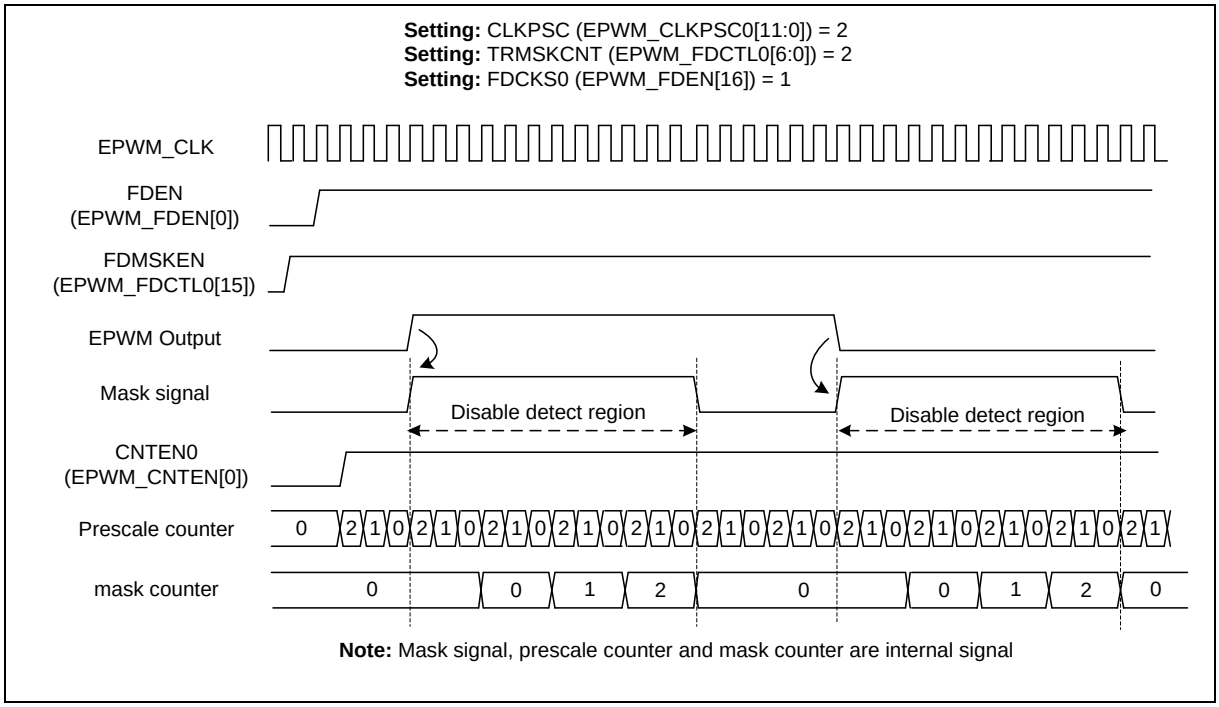


Figure 6.15-56 EPWM Mask Function Waveform

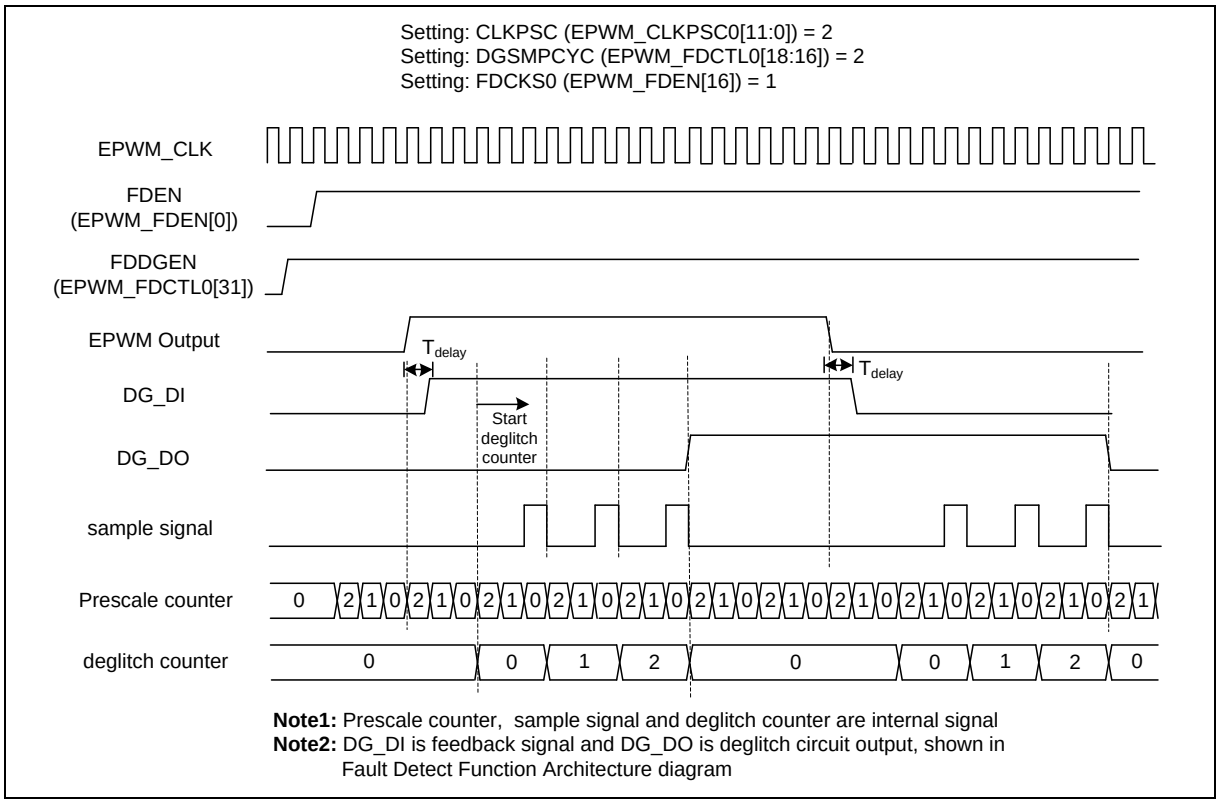


Figure 6.15-57 EPWM Deglitch Function Waveform

6.15.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
EPWM Base Address: **EPWM0_BA = 0x4005_8000** **EPWM1_BA = 0x4005_9000** **x=0,1**				
EPWM_CTL0 **x=0,1**	EPWMx_BA+0x00	R/W	EPWM Control Register 0	0x0000_0000
EPWM_CTL1 **x=0,1**	EPWMx_BA+0x04	R/W	EPWM Control Register 1	0x0000_0000
EPWM_SYNC **x=0,1**	EPWMx_BA+0x08	R/W	EPWM Synchronization Register	0x0000_0000
EPWM_SWSYNC **x=0,1**	EPWMx_BA+0x0C	W	EPWM Software Control Synchronization Register	0x0000_0000
EPWM_CLKSRC **x=0,1**	EPWMx_BA+0x10	R/W	EPWM Clock Source Register	0x0000_0000
EPWM_CNTEN **x=0,1**	EPWMx_BA+0x20	R/W	EPWM Counter Enable Register	0x0000_0000
EPWM_CNTCLR **x=0,1**	EPWMx_BA+0x24	R/W	EPWM Clear Counter Register	0x0000_0000
EPWM_LOAD **x=0,1**	EPWMx_BA+0x28	R/W	EPWM Load Register	0x0000_0000
EPWM_PERIOD0 **x=0,1**	EPWMx_BA+0x30	R/W	EPWM Period Register 0	0x0000_0000
EPWM_PERIOD1 **x=0,1**	EPWMx_BA+0x34	R/W	EPWM Period Register 1	0x0000_0000
EPWM_PERIOD2 **x=0,1**	EPWMx_BA+0x38	R/W	EPWM Period Register 2	0x0000_0000
EPWM_PERIOD3 **x=0,1**	EPWMx_BA+0x3C	R/W	EPWM Period Register 3	0x0000_0000
EPWM_PERIOD4 **x=0,1**	EPWMx_BA+0x40	R/W	EPWM Period Register 4	0x0000_0000
EPWM_PERIOD5 **x=0,1**	EPWMx_BA+0x44	R/W	EPWM Period Register 5	0x0000_0000
EPWM_CMPDAT0 **x=0,1**	EPWMx_BA+0x50	R/W	EPWM Comparator Register 0	0x0000_0000
EPWM_CMPDAT1 **x=0,1**	EPWMx_BA+0x54	R/W	EPWM Comparator Register 1	0x0000_0000
EPWM_CMPDAT2 **x=0,1**	EPWMx_BA+0x58	R/W	EPWM Comparator Register 2	0x0000_0000

EPWM_CMPDAT3 x=0,1	EPWMx_BA+0x5C	R/W	EPWM Comparator Register 3	0x0000_0000
EPWM_CMPDAT4 x=0,1	EPWMx_BA+0x60	R/W	EPWM Comparator Register 4	0x0000_0000
EPWM_CMPDAT5 x=0,1	EPWMx_BA+0x64	R/W	EPWM Comparator Register 5	0x0000_0000
EPWM_PHS0_1 x=0,1	EPWMx_BA+0x80	R/W	EPWM Counter Phase Register 0/1	0x0000_0000
EPWM_PHS2_3 x=0,1	EPWMx_BA+0x84	R/W	EPWM Counter Phase Register 2/3	0x0000_0000
EPWM_PHS4_5 x=0,1	EPWMx_BA+0x88	R/W	EPWM Counter Phase Register 4/5	0x0000_0000
EPWM_CNT0 x=0,1	EPWMx_BA+0x90	R	EPWM Counter Register 0	0x0000_0000
EPWM_CNT1 x=0,1	EPWMx_BA+0x94	R	EPWM Counter Register 1	0x0000_0000
EPWM_CNT2 x=0,1	EPWMx_BA+0x98	R	EPWM Counter Register 2	0x0000_0000
EPWM_CNT3 x=0,1	EPWMx_BA+0x9C	R	EPWM Counter Register 3	0x0000_0000
EPWM_CNT4 x=0,1	EPWMx_BA+0xA0	R	EPWM Counter Register 4	0x0000_0000
EPWM_CNT5 x=0,1	EPWMx_BA+0xA4	R	EPWM Counter Register 5	0x0000_0000
EPWM_WGCTL0 x=0,1	EPWMx_BA+0xB0	R/W	EPWM Generation Register 0	0x0000_0000
EPWM_WGCTL1 x=0,1	EPWMx_BA+0xB4	R/W	EPWM Generation Register 1	0x0000_0000
EPWM_MSKEN x=0,1	EPWMx_BA+0xB8	R/W	EPWM Mask Enable Register	0x0000_0000
EPWM_MSK x=0,1	EPWMx_BA+0xBC	R/W	EPWM Mask Data Register	0x0000_0000
EPWM_BNF x=0,1	EPWMx_BA+0xC0	R/W	EPWM Brake Noise Filter Register	0x0000_0000
EPWM_FAILBRK x=0,1	EPWMx_BA+0xC4	R/W	EPWM System Fail Brake Control Register	0x0000_0000
EPWM_BRKCTL0_1 x=0,1	EPWMx_BA+0xC8	R/W	EPWM Brake Edge Detect Control Register 0/1	0x0000_0000
EPWM_BRKCTL2_3 x=0,1	EPWMx_BA+0xCC	R/W	EPWM Brake Edge Detect Control Register 2/3	0x0000_0000
EPWM_BRKCTL4_5	EPWMx_BA+0xD0	R/W	EPWM Brake Edge Detect Control Register 4/5	0x0000_0000

x=0,1				
EPWM_POLCTL **x=0,1**	EPWMx_BA+0xD4	R/W	EPWM Pin Polar Inverse Register	0x0000_0000
EPWM_POEN **x=0,1**	EPWMx_BA+0xD8	R/W	EPWM Output Enable Register	0x0000_0000
EPWM_SWBRK **x=0,1**	EPWMx_BA+0xDC	W	EPWM Software Brake Control Register	0x0000_0000
EPWM_INTEN0 **x=0,1**	EPWMx_BA+0xE0	R/W	EPWM Interrupt Enable Register 0	0x0000_0000
EPWM_INTEN1 **x=0,1**	EPWMx_BA+0xE4	R/W	EPWM Interrupt Enable Register 1	0x0000_0000
EPWM_INTSTS0 **x=0,1**	EPWMx_BA+0xE8	R/W	EPWM Interrupt Flag Register 0	0x0000_0000
EPWM_INTSTS1 **x=0,1**	EPWMx_BA+0xEC	R/W	EPWM Interrupt Flag Register 1	0x0000_0000
EPWM_DACTRGEN **x=0,1**	EPWMx_BA+0xF4	R/W	EPWM Trigger DAC Enable Register	0x0000_0000
EPWM_EADCTS0 **x=0,1**	EPWMx_BA+0xF8	R/W	EPWM Trigger EADC Source Select Register 0	0x0000_0000
EPWM_EADCTS1 **x=0,1**	EPWMx_BA+0xFC	R/W	EPWM Trigger EADC Source Select Register 1	0x0000_0000
EPWM_FTCMPDAT0_1 **x=0,1**	EPWMx_BA+0x100	R/W	EPWM Free Trigger Compare Register 0/1	0x0000_0000
EPWM_FTCMPDAT2_3 **x=0,1**	EPWMx_BA+0x104	R/W	EPWM Free Trigger Compare Register 2/3	0x0000_0000
EPWM_FTCMPDAT4_5 **x=0,1**	EPWMx_BA+0x108	R/W	EPWM Free Trigger Compare Register 4/5	0x0000_0000
EPWM_SSCTL **x=0,1**	EPWMx_BA+0x110	R/W	EPWM Synchronous Start Control Register	0x0000_0000
EPWM_SSTRG **x=0,1**	EPWMx_BA+0x114	W	EPWM Synchronous Start Trigger Register	0x0000_0000
EPWM_LEBCTL **x=0,1**	EPWMx_BA+0x118	R/W	EPWM Leading Edge Blanking Control Register	0x0000_0000
EPWM_LEBCNT **x=0,1**	EPWMx_BA+0x11C	R/W	EPWM Leading Edge Blanking Counter Register	0x0000_0000
EPWM_STATUS **x=0,1**	EPWMx_BA+0x120	R/W	EPWM Status Register	0x0000_0000
EPWM_IFA0 **x=0,1**	EPWMx_BA+0x130	R/W	EPWM Interrupt Flag Accumulator Register 0	0x0000_0000
EPWM_IFA1 **x=0,1**	EPWMx_BA+0x134	R/W	EPWM Interrupt Flag Accumulator Register 1	0x0000_0000

EPWM_IFA2 **x=0,1**	EPWMx_BA+0x138	R/W	EPWM Interrupt Flag Accumulator Register 2	0x0000_0000
EPWM_IFA3 **x=0,1**	EPWMx_BA+0x13C	R/W	EPWM Interrupt Flag Accumulator Register 3	0x0000_0000
EPWM_IFA4 **x=0,1**	EPWMx_BA+0x140	R/W	EPWM Interrupt Flag Accumulator Register 4	0x0000_0000
EPWM_IFA5 **x=0,1**	EPWMx_BA+0x144	R/W	EPWM Interrupt Flag Accumulator Register 5	0x0000_0000
EPWM_AINTSTS **x=0,1**	EPWMx_BA+0x150	R/W	EPWM Accumulator Interrupt Flag Register	0x0000_0000
EPWM_AINTEN **x=0,1**	EPWMx_BA+0x154	R/W	EPWM Accumulator Interrupt Enable Register	0x0000_0000
EPWM_APDMACTL **x=0,1**	EPWMx_BA+0x158	R/W	EPWM Accumulator PDMA Control Register	0x0000_0000
EPWM_FDEN **x=0,1**	EPWMx_BA+0x160	R/W	EPWM Fault Detect Enable Register	0x0000_0000
EPWM_FDCTL0 **x=0,1**	EPWMx_BA+0x164	R/W	EPWM Fault Detect Control Register 0	0x0000_0000
EPWM_FDCTL1 **x=0,1**	EPWMx_BA+0x168	R/W	EPWM Fault Detect Control Register 1	0x0000_0000
EPWM_FDCTL2 **x=0,1**	EPWMx_BA+0x16C	R/W	EPWM Fault Detect Control Register 2	0x0000_0000
EPWM_FDCTL3 **x=0,1**	EPWMx_BA+0x170	R/W	EPWM Fault Detect Control Register 3	0x0000_0000
EPWM_FDCTL4 **x=0,1**	EPWMx_BA+0x174	R/W	EPWM Fault Detect Control Register 4	0x0000_0000
EPWM_FDCTL5 **x=0,1**	EPWMx_BA+0x178	R/W	EPWM Fault Detect Control Register 5	0x0000_0000
EPWM_FDIEN **x=0,1**	EPWMx_BA+0x17C	R/W	EPWM Fault Detect Interrupt Enable Register	0x0000_0000
EPWM_FDSTS **x=0,1**	EPWMx_BA+0x180	R/W	EPWM Fault Detect Interrupt Flag Register	0x0000_0000
EPWM_EADCPSCCTL **x=0,1**	EPWMx_BA+0x184	R/W	EPWM Trigger EADC Prescale Control Register	0x0000_0000
EPWM_EADCPSC0 **x=0,1**	EPWMx_BA+0x188	R/W	EPWM Trigger EADC Prescale Register 0	0x0000_0000
EPWM_EADCPSC1 **x=0,1**	EPWMx_BA+0x18C	R/W	EPWM Trigger EADC Prescale Register 1	0x0000_0000
EPWM_EADCPSCNT0 **x=0,1**	EPWMx_BA+0x190	R/W	EPWM Trigger EADC Prescale Counter Register 0	0x0000_0000
EPWM_EADCPSCNT1	EPWMx_BA+0x194	R/W	EPWM Trigger EADC Prescale Counter Register 1	0x0000_0000

x=0,1				
EPWM_CAPINEN **x=0,1**	EPWMx_BA+0x200	R/W	EPWM Capture Input Enable Register	0x0000_0000
EPWM_CAPCTL **x=0,1**	EPWMx_BA+0x204	R/W	EPWM Capture Control Register	0x0000_0000
EPWM_CAPSTS **x=0,1**	EPWMx_BA+0x208	R	EPWM Capture Status Register	0x0000_0000
EPWM_RCAPDAT0 **x=0,1**	EPWMx_BA+0x20C	R	EPWM Rising Capture Data Register 0	0x0000_0000
EPWM_FCAPDAT0 **x=0,1**	EPWMx_BA+0x210	R	EPWM Falling Capture Data Register 0	0x0000_0000
EPWM_RCAPDAT1 **x=0,1**	EPWMx_BA+0x214	R	EPWM Rising Capture Data Register 1	0x0000_0000
EPWM_FCAPDAT1 **x=0,1**	EPWMx_BA+0x218	R	EPWM Falling Capture Data Register 1	0x0000_0000
EPWM_RCAPDAT2 **x=0,1**	EPWMx_BA+0x21C	R	EPWM Rising Capture Data Register 2	0x0000_0000
EPWM_FCAPDAT2 **x=0,1**	EPWMx_BA+0x220	R	EPWM Falling Capture Data Register 2	0x0000_0000
EPWM_RCAPDAT3 **x=0,1**	EPWMx_BA+0x224	R	EPWM Rising Capture Data Register 3	0x0000_0000
EPWM_FCAPDAT3 **x=0,1**	EPWMx_BA+0x228	R	EPWM Falling Capture Data Register 3	0x0000_0000
EPWM_RCAPDAT4 **x=0,1**	EPWMx_BA+0x22C	R	EPWM Rising Capture Data Register 4	0x0000_0000
EPWM_FCAPDAT4 **x=0,1**	EPWMx_BA+0x230	R	EPWM Falling Capture Data Register 4	0x0000_0000
EPWM_RCAPDAT5 **x=0,1**	EPWMx_BA+0x234	R	EPWM Rising Capture Data Register 5	0x0000_0000
EPWM_FCAPDAT5 **x=0,1**	EPWMx_BA+0x238	R	EPWM Falling Capture Data Register 5	0x0000_0000
EPWM_PDMACTL **x=0,1**	EPWMx_BA+0x23C	R/W	EPWM PDMA Control Register	0x0000_0000
EPWM_PDMACAP0_1 **x=0,1**	EPWMx_BA+0x240	R	EPWM Capture Channel 01 PDMA Register	0x0000_0000
EPWM_PDMACAP2_3 **x=0,1**	EPWMx_BA+0x244	R	EPWM Capture Channel 23 PDMA Register	0x0000_0000
EPWM_PDMACAP4_5 **x=0,1**	EPWMx_BA+0x248	R	EPWM Capture Channel 45 PDMA Register	0x0000_0000
EPWM_CAPIEN **x=0,1**	EPWMx_BA+0x250	R/W	EPWM Capture Interrupt Enable Register	0x0000_0000

EPWM_CAPIF x=0,1	EPWMx_BA+0x254	R/W	EPWM Capture Interrupt Flag Register	0x0000_0000
EPWM_CAPNF0 x=0,1	EPWMx_BA+0x258	R/W	EPWM Capture Input Noise Filter Register 0	0x0000_0000
EPWM_CAPNF1 x=0,1	EPWMx_BA+0x25C	R/W	EPWM Capture Input Noise Filter Register 1	0x0000_0000
EPWM_CAPNF2 x=0,1	EPWMx_BA+0x260	R/W	EPWM Capture Input Noise Filter Register 2	0x0000_0000
EPWM_CAPNF3 x=0,1	EPWMx_BA+0x264	R/W	EPWM Capture Input Noise Filter Register 3	0x0000_0000
EPWM_CAPNF4 x=0,1	EPWMx_BA+0x268	R/W	EPWM Capture Input Noise Filter Register 4	0x0000_0000
EPWM_CAPNF5 x=0,1	EPWMx_BA+0x26C	R/W	EPWM Capture Input Noise Filter Register 5	0x0000_0000
EPWM_EXTETCTL0 x=0,1	EPWMx_BA+0x270	R/W	EPWM External Event Trigger Control Register 0	0x0000_0000
EPWM_EXTETCTL1 x=0,1	EPWMx_BA+0x274	R/W	EPWM External Event Trigger Control Register 1	0x0000_0000
EPWM_EXTETCTL2 x=0,1	EPWMx_BA+0x278	R/W	EPWM External Event Trigger Control Register 2	0x0000_0000
EPWM_EXTETCTL3 x=0,1	EPWMx_BA+0x27C	R/W	EPWM External Event Trigger Control Register 3	0x0000_0000
EPWM_EXTETCTL4 x=0,1	EPWMx_BA+0x280	R/W	EPWM External Event Trigger Control Register 4	0x0000_0000
EPWM_EXTETCTL5 x=0,1	EPWMx_BA+0x284	R/W	EPWM External Event Trigger Control Register 5	0x0000_0000
EPWM_SWEOFCTL x=0,1	EPWMx_BA+0x288	R/W	EPWM Software Event Output Force Control Register	0x0000_0000
EPWM_SWEOFTRG x=0,1	EPWMx_BA+0x28C	R/W	EPWM Software Event Output Force Trigger Register	0x0000_0000
EPWM_CLKPSC0 x=0,1	EPWMx_BA+0x290	R/W	EPWM Clock Prescale Register 0	0x0000_0000
EPWM_CLKPSC1 x=0,1	EPWMx_BA+0x294	R/W	EPWM Clock Prescale Register 1	0x0000_0000
EPWM_CLKPSC2 x=0,1	EPWMx_BA+0x298	R/W	EPWM Clock Prescale Register 2	0x0000_0000
EPWM_CLKPSC3 x=0,1	EPWMx_BA+0x29C	R/W	EPWM Clock Prescale Register 3	0x0000_0000
EPWM_CLKPSC4 x=0,1	EPWMx_BA+0x2A0	R/W	EPWM Clock Prescale Register 4	0x0000_0000
EPWM_CLKPSC5	EPWMx_BA+0x2A4	R/W	EPWM Clock Prescale Register 5	0x0000_0000

x=0,1				
EPWM_RDTCNT0_1 **x=0,1**	EPWMx_BA+0x2A8	R/W	EPWM Rising Dead-time Counter Register 0/1	0x0000_0000
EPWM_RDTCNT2_3 **x=0,1**	EPWMx_BA+0x2AC	R/W	EPWM Rising Dead-time Counter Register 2/3	0x0000_0000
EPWM_RDTCNT4_5 **x=0,1**	EPWMx_BA+0x2B0	R/W	EPWM Rising Dead-time Counter Register 4/5	0x0000_0000
EPWM_FDTCNT0_1 **x=0,1**	EPWMx_BA+0x2B4	R/W	EPWM Falling Dead-time Counter Register 0/1	0x0000_0000
EPWM_FDTCNT2_3 **x=0,1**	EPWMx_BA+0x2B8	R/W	EPWM Falling Dead-time Counter Register 2/3	0x0000_0000
EPWM_FDTCNT4_5 **x=0,1**	EPWMx_BA+0x2BC	R/W	EPWM Falling Dead-time Counter Register 4/5	0x0000_0000
EPWM_DTCTL **x=0,1**	EPWMx_BA+0x2C0	R/W	EPWM Dead-time Control Register	0x0000_0000
EPWM_PBUF0 **x=0,1**	EPWMx_BA+0x304	R	EPWM PERIOD0 Buffer	0x0000_0000
EPWM_PBUF1 **x=0,1**	EPWMx_BA+0x308	R	EPWM PERIOD1 Buffer	0x0000_0000
EPWM_PBUF2 **x=0,1**	EPWMx_BA+0x30C	R	EPWM PERIOD2 Buffer	0x0000_0000
EPWM_PBUF3 **x=0,1**	EPWMx_BA+0x310	R	EPWM PERIOD3 Buffer	0x0000_0000
EPWM_PBUF4 **x=0,1**	EPWMx_BA+0x314	R	EPWM PERIOD4 Buffer	0x0000_0000
EPWM_PBUF5 **x=0,1**	EPWMx_BA+0x318	R	EPWM PERIOD5 Buffer	0x0000_0000
EPWM_CMPBUF0 **x=0,1**	EPWMx_BA+0x31C	R	EPWM CMPDAT0 Buffer	0x0000_0000
EPWM_CMPBUF1 **x=0,1**	EPWMx_BA+0x320	R	EPWM CMPDAT1 Buffer	0x0000_0000
EPWM_CMPBUF2 **x=0,1**	EPWMx_BA+0x324	R	EPWM CMPDAT2 Buffer	0x0000_0000
EPWM_CMPBUF3 **x=0,1**	EPWMx_BA+0x328	R	EPWM CMPDAT3 Buffer	0x0000_0000
EPWM_CMPBUF4 **x=0,1**	EPWMx_BA+0x32C	R	EPWM CMPDAT4 Buffer	0x0000_0000
EPWM_CMPBUF5 **x=0,1**	EPWMx_BA+0x330	R	EPWM CMPDAT5 Buffer	0x0000_0000
EPWM_FTCBUF0_1 **x=0,1**	EPWMx_BA+0x340	R	EPWM FTCMPDAT0_1 Buffer	0x0000_0000

EPWM_FTCBUF2_3 x=0,1	EPWMx_BA+0x344	R	EPWM FTCMPDAT2_3 Buffer	0x0000_0000
EPWM_FTCBUF4_5 x=0,1	EPWMx_BA+0x348	R	EPWM FTCMPDAT4_5 Buffer	0x0000_0000
EPWM_FTCI x=0,1	EPWMx_BA+0x34C	R/W	EPWM FTCMPDAT Indicator Register	0x0000_0000
EPWM_CPSCBUF0 x=0,1	EPWMx_BA+0x350	R	EPWM CLKPSC0 Buffer	0x0000_0000
EPWM_CPSCBUF1 x=0,1	EPWMx_BA+0x354	R	EPWM CLKPSC1 Buffer	0x0000_0000
EPWM_CPSCBUF2 x=0,1	EPWMx_BA+0x358	R	EPWM CLKPSC2 Buffer	0x0000_0000
EPWM_CPSCBUF3 x=0,1	EPWMx_BA+0x35C	R	EPWM CLKPSC3 Buffer	0x0000_0000
EPWM_CPSCBUF4 x=0,1	EPWMx_BA+0x360	R	EPWM CLKPSC4 Buffer	0x0000_0000
EPWM_CPSCBUF5 x=0,1	EPWMx_BA+0x364	R	EPWM CLKPSC5 Buffer	0x0000_0000
EPWM_IFACNT0 x=0,1	EPWMx_BA+0x368	R	EPWM Interrupt Flag Accumulator Counter 0	0x0000_0000
EPWM_IFACNT1 x=0,1	EPWMx_BA+0x36C	R	EPWM Interrupt Flag Accumulator Counter 1	0x0000_0000
EPWM_IFACNT2 x=0,1	EPWMx_BA+0x370	R	EPWM Interrupt Flag Accumulator Counter 2	0x0000_0000
EPWM_IFACNT3 x=0,1	EPWMx_BA+0x374	R	EPWM Interrupt Flag Accumulator Counter 3	0x0000_0000
EPWM_IFACNT4 x=0,1	EPWMx_BA+0x378	R	EPWM Interrupt Flag Accumulator Counter 4	0x0000_0000
EPWM_IFACNT5 x=0,1	EPWMx_BA+0x37C	R	EPWM Interrupt Flag Accumulator Counter 5	0x0000_0000

6.15.7 Register Description

EPWM Control Register 0 (EPWM_CTL0)

Register	Offset	R/W	Description	Reset Value
EPWM_CTL0	EPWMx_BA+0x00	R/W	EPWM Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	DBGHALT	Reserved					GROUPEN
23	22	21	20	19	18	17	16
Reserved		IMMLDEN5	IMMLDEN4	IMMLDEN3	IMMLDEN2	IMMLDEN1	IMMLDEN0
15	14	13	12	11	10	9	8
Reserved		WINLDEN5	WINLDEN4	WINLDEN3	WINLDEN2	WINLDEN1	WINLDEN0
7	6	5	4	3	2	1	0
Reserved		CTRLD5	CTRLD4	CTRLD3	CTRLD2	CTRLD1	CTRLD0

Bits	Description	
[31]	**DBGTRIOFF**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects EPWM output. EPWM pin will be forced as tri-state while ICE debug mode acknowledged. 1 = ICE debug mode acknowledgement disabled. EPWM pin will keep output no matter ICE debug mode acknowledged or not. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[30]	**DBGHALT**	**ICE Debug Mode Counter Halt (Write Protect)** If counter halt is enabled, EPWM all counters will keep current value until exit ICE debug mode. 0 = ICE debug mode counter halt Disabled. 1 = ICE debug mode counter halt Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[29:25]	**Reserved**	Reserved.
[24]	**GROUPEN**	**Group Function Enable Bit** 0 = The output waveform of each EPWM channel are independent. 1 = Unify the EPWM_CH2 and EPWM_CH4 to output the same waveform as EPWM_CH0 and unify the EPWM_CH3 and EPWM_CH5 to output the same waveform as EPWM_CH1.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**IMMLDENn**	**Immediately Load Enable Bits** 0 = PERIOD will load to PBUF at the end point of each period. CMP will load to CMPBUF at the end point or center point of each period by setting CTRLD bit. 1 = PERIOD/CMP will load to PBUF and CMPBUF immediately when software update PERIOD/CMP. **Note:** If IMMLDENn is enabled, WINLDENn and CTRLDn will be invalid.
[15:14]	**Reserved**	Reserved.

[8+n] n=0,1..5	**WINLDENn**	**Window Load Enable Bits** 0 = PERIOD will load to PBUF at the end point of each period. CMP will load to CMPBUF at the end point or center point of each period by setting CTRLD bit. 1 = PERIOD will load to PBUF at the end point of each period. CMP will load to CMPBUF at the end point of each period when valid reload window is set. The valid reload window is set by software write 1 to EPWM_LOAD register and cleared by hardware after load success.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CTRLDn**	**Center Re-load** In up-down counter type, PERIOD will load to PBUF at the end point of each period. CMP will load to CMPBUF at the center point of a period.

EPWM Control Register 1 (EPWM_CTL1)

Register	Offset	R/W	Description	Reset Value
EPWM_CTL1	EPWMx_BA+0x04	R/W	EPWM Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					OUTMODE4	OUTMODE2	OUTMODE0
23	22	21	20	19	18	17	16
Reserved		CNTMODE5	CNTMODE4	CNTMODE3	CNTMODE2	CNTMODE1	CNTMODE0
15	14	13	12	11	10	9	8
Reserved				CNTTYPE5		CNTTYPE4	
7	6	5	4	3	2	1	0
CNTTYPE3		CNTTYPE2		CNTTYPE1		CNTTYPE0	

Bits	Description	
[31:27]	**Reserved**	Reserved.
[24+n/2] n=0,2,4	**OUTMODEn**	**EPWM Output Mode** Each bit n controls the output mode of corresponding EPWM channel n. 0 = EPWM independent mode. 1 = EPWM complementary mode. **Note:** When operating in group function, these bits must all set to the same mode.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CNTMODEn**	**EPWM Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.
[15:12]	**Reserved**	Reserved.
[2n+1:2n] n=0,1..5	**CNTTYPEn**	**EPWM Counter Behavior Type** 00 = Up counter type (supported in capture mode). 01 = Down count type (supported in capture mode). 10 = Up-down counter type. 11 = Reserved.

EPWM Synchronization Register (EPWM_SYNC)

Register	Offset	R/W	Description	Reset Value
EPWM_SYNC	EPWMx_BA+0x08	R/W	EPWM Synchronization Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					PHSDIR4	PHSDIR2	PHSDIR0
23	22	21	20	19	18	17	16
SINPINV	SFLTCNT			SFLTCSEL			SNFLTEN
15	14	13	12	11	10	9	8
Reserved		SINSRC4		SINSRC2		SINSRC0	
7	6	5	4	3	2	1	0
Reserved					PHSEN4	PHSEN2	PHSEN0

Bits	Description	
[31:27]	**Reserved**	Reserved.
[24+n/2] n=0,2,4	**PHSDIRn**	**EPWM Phase Direction Control** 0 = Control EPWM counter count decrement after synchronizing. 1 = Control EPWM counter count increment after synchronizing.
[23]	**SINPINV**	**SYNC Input Pin Inverse** 0 = The state of pin SYNC is passed to the negative edge detector. 1 = The inversed state of pin SYNC is passed to the negative edge detector.
[22:20]	**SFLTCNT**	**SYNC Edge Detector Filter Count** The register bits control the counter number of edge detector.
[19:17]	**SFLTCSEL**	**SYNC Edge Detector Filter Clock Selection** 000 = Filter clock = HCLK. 001 = Filter clock = HCLK/2. 010 = Filter clock = HCLK/4. 011 = Filter clock = HCLK/8. 100 = Filter clock = HCLK/16. 101 = Filter clock = HCLK/32. 110 = Filter clock = HCLK/64. 111 = Filter clock = HCLK/128.
[16]	**SNFLTEN**	**EPWM0_SYNC_IN Noise Filter Enable Bits** 0 = Noise filter of input pin EPWM0_SYNC_IN Disabled. 1 = Noise filter of input pin EPWM0_SYNC_IN Enabled.
[15:14]	**Reserved**	Reserved.
[9+n:8+n] n=0,2,4	**SINSRCn**	**EPWM0_SYNC_IN Source Selection** 00 = Synchronize source from SYNC_IN or SWSYNC. 01 = Counter equal to 0. 10 = Counter equal to EPWM_CMPDATm, m denotes 1, 3, 5.

		11 = SYNC_OUT will not be generated.
[7:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**PHSENn**	**SYNC Phase Enable Bits** 0 = EPWM counter disable to load PHS value. 1 = EPWM counter enable to load PHS value.

EPWM Software Control Synchronization Register (EPWM_SWSYNC)

Register	Offset	R/W	Description	Reset Value
EPWM_SWSYNC	EPWMx_BA+0x0C	W	EPWM Software Control Synchronization Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					SWSYNC4	SWSYNC2	SWSYNC0

Bits	Description	
[31:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**SWSYNCn**	**Software SYNC Function (Write Only)** When SINSRCn (EPWM_SYNC[13:8]) is selected to 0, SYNC_OUT source comes from SYNC_IN or this bit.

EPWM Clock Source Register (EPWM_CLKSRC)

Register	Offset	R/W	Description	Reset Value
EPWM_CLKSRC	EPWMx_BA+0x10	R/W	EPWM Clock Source Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**ECLKSRC4**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**ECLKSRC2**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ECLKSRC0**		

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18:16]	**ECLKSRC4**	**EPWM_CH45 External Clock Source Select** 000 = EPWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.
[15:11]	**Reserved**	Reserved.
[10:8]	**ECLKSRC2**	**EPWM_CH23 External Clock Source Select** 000 = EPWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.
[7:3]	**Reserved**	Reserved.
[2:0]	**ECLKSRC0**	**EPWM_CH01 External Clock Source Select** 000 = EPWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.

EPWM Counter Enable Register (EPWM_CNTEN)

Register	Offset	R/W	Description	Reset Value
EPWM_CNTEN	EPWMx_BA+0x20	R/W	EPWM Counter Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CNTEN5**	**CNTEN4**	**CNTEN3**	**CNTEN2**	**CNTEN1**	**CNTEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CNTENn**	**EPWM Counter Enable Bits** 0 = EPWM Counter and clock prescaler stop running. 1 = EPWM Counter and clock prescaler start running.

EPWM Clear Counter Register (EPWM_CNTCLR)

Register	Offset	R/W	Description	Reset Value
EPWM_CNTCLR	EPWMx_BA+0x24	R/W	EPWM Clear Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CNTCLR5**	**CNTCLR4**	**CNTCLR3**	**CNTCLR2**	**CNTCLR1**	**CNTCLR0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CNTCLRn**	**Clear EPWM Counter Control Bit** It is automatically cleared by hardware. Each bit n controls the corresponding EPWM channel n. 0 = No effect. 1 = Clear 16-bit EPWM counter to 0000H.

EPWM Load Register (EPWM_LOAD)

Register	Offset	R/W	Description	Reset Value
EPWM_LOAD	EPWMx_BA+0x28	R/W	EPWM Load Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**LOAD5**	**LOAD4**	**LOAD3**	**LOAD2**	**LOAD1**	**LOAD0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**LOADn**	**Re-load EPWM Comparator Register Control Bit** This bit is software write to reload EPWM_CMPDATn, n=0~5. Hardware clear when current EPWM period end. Write Operation: 0 = No effect. 1 = Set load window of window loading mode. Read Operation: 0 = No load window is set. 1 = Load window is set. **Note:** This bit only use in window loading mode, WINLDENn(EPWM_CTL0[13:8]) = 1.

EPWM Period Register 0~5 (EPWM_PERIOD0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_PERIOD0	EPWMx_BA+0x30	R/W	EPWM Period Register 0	0x0000_0000
EPWM_PERIOD1	EPWMx_BA+0x34	R/W	EPWM Period Register 1	0x0000_0000
EPWM_PERIOD2	EPWMx_BA+0x38	R/W	EPWM Period Register 2	0x0000_0000
EPWM_PERIOD3	EPWMx_BA+0x3C	R/W	EPWM Period Register 3	0x0000_0000
EPWM_PERIOD4	EPWMx_BA+0x40	R/W	EPWM Period Register 4	0x0000_0000
EPWM_PERIOD5	EPWMx_BA+0x44	R/W	EPWM Period Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**EPWM Period Register** Up-Count mode: In this mode, EPWM counter counts from 0 to PERIOD, and restarts from 0. EPWM period time = (PERIOD+1) * (CLKPSC+1) * EPWM_CLK . Down-Count mode: In this mode, EPWM counter counts from PERIOD to 0, and restarts from PERIOD. EPWM period time = (PERIOD+1) * (CLKPSC+1) * EPWM_CLK . Up-Down-Count mode: In this mode, EPWM counter counts from 0 to PERIOD, then decrements to 0 and repeats again. EPWM period time = 2 * PERIOD * (CLKPSC+1) * EPWM_CLK.

EPWM Comparator Register 0~5 (EPWM_CMPDAT0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CMPDAT0	EPWMx_BA+0x50	R/W	EPWM Comparator Register 0	0x0000_0000
EPWM_CMPDAT1	EPWMx_BA+0x54	R/W	EPWM Comparator Register 1	0x0000_0000
EPWM_CMPDAT2	EPWMx_BA+0x58	R/W	EPWM Comparator Register 2	0x0000_0000
EPWM_CMPDAT3	EPWMx_BA+0x5C	R/W	EPWM Comparator Register 3	0x0000_0000
EPWM_CMPDAT4	EPWMx_BA+0x60	R/W	EPWM Comparator Register 4	0x0000_0000
EPWM_CMPDAT5	EPWMx_BA+0x64	R/W	EPWM Comparator Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMP**	**EPWM Comparator Register** CMP is used to compare with CNT (EPWM_CNTn[15:0]) bits to generate EPWM waveform, interrupt and trigger EADC/LPADC/DAC. In independent mode, EPWM_CMPDATn, n=0,1..5 denote as 6 independent EPWM_CH0~5 compared point. In complementary mode, EPWM_CMPDAT0, EPWM_CMPDAT2, EPWM_CMPDAT4 denote as first compared point, and EPWM_CMPDAT1, EPWM_CMPDAT3, EPWM_CMPDAT5 denote as second compared point for the corresponding 3 complementary pairs EPWM_CH0 and EPWM_CH1, EPWM_CH2 and EPWM_CH3, EPWM_CH4 and EPWM_CH5.

EPWM Counter Phase Register 0_1, 2_3, 4_5 (EPWM_PHS0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_PHS0_1	EPWMx_BA+0x80	R/W	EPWM Counter Phase Register 0/1	0x0000_0000
EPWM_PHS2_3	EPWMx_BA+0x84	R/W	EPWM Counter Phase Register 2/3	0x0000_0000
EPWM_PHS4_5	EPWMx_BA+0x88	R/W	EPWM Counter Phase Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PHS							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PHS							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PHS**	**EPWM Synchronous Start Phase Bits** PHS determines the EPWM synchronous start phase value. These bits only use in synchronous function.

EPWM Counter Register 0~5 (EPWM_CNT0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CNT0	EPWMx_BA+0x90	R	EPWM Counter Register 0	0x0000_0000
EPWM_CNT1	EPWMx_BA+0x94	R	EPWM Counter Register 1	0x0000_0000
EPWM_CNT2	EPWMx_BA+0x98	R	EPWM Counter Register 2	0x0000_0000
EPWM_CNT3	EPWMx_BA+0x9C	R	EPWM Counter Register 3	0x0000_0000
EPWM_CNT4	EPWMx_BA+0xA0	R	EPWM Counter Register 4	0x0000_0000
EPWM_CNT5	EPWMx_BA+0xA4	R	EPWM Counter Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**DIRF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**DIRF**	**EPWM Direction Indicator Flag (Read Only)** 0 = Counter is counting down. 1 = Counter is counting up.
[15:0]	**CNT**	**EPWM Data Register (Read Only)** User can monitor CNT to know the current value in 16-bit period counter.

EPWM Generation Register 0 (EPWM_WGCTL0)

Register	Offset	R/W	Description	Reset Value
EPWM_WGCTL0	EPWMx_BA+0xB0	R/W	EPWM Generation Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**PRDPCTL5**		**PRDPCTL4**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRDPCTL3		**PRDPCTL2**		**PRDPCTL1**		**PRDPCTL0**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ZPCTL5**		**ZPCTL4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ZPCTL3		**ZPCTL2**		**ZPCTL1**		**ZPCTL0**	

Bits	Description	
[31:28]	**Reserved**	Reserved.
[17+2n:16+2n] n=0,1..5	**PRDPCTLn**	**EPWM Period or Center Point Control** EPWM can control output level when EPWM counter counts to (PERIODn+1). 00 = Do nothing. 01 = EPWM period (center) point output Low. 10 = EPWM period (center) point output High. 11 = EPWM period (center) point output Toggle. **Note:** This bit is center point control when EPWM counter operating in up-down counter type.
[15:12]	**Reserved**	Reserved.
[1+2n:2n] n=0,1..5	**ZPCTLn**	**EPWM Zero Point Control** EPWM can control output level when EPWM counter counts to 0. 00 = Do nothing. 01 = EPWM zero point output Low. 10 = EPWM zero point output High. 11 = EPWM zero point output Toggle.

EPWM Generation Register 1 (EPWM_WGCTL1)

Register	Offset	R/W	Description	Reset Value
EPWM_WGCTL1	EPWMx_BA+0xB4	R/W	EPWM Generation Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				CMPDCTL5		CMPDCTL4	
23	22	21	20	19	18	17	16
CMPDCTL3		CMPDCTL2		CMPDCTL1		CMPDCTL0	
15	14	13	12	11	10	9	8
Reserved				CMPUCTL5		CMPUCTL4	
7	6	5	4	3	2	1	0
CMPUCTL3		CMPUCTL2		CMPUCTL1		CMPUCTL0	

Bits	Description	
[31:28]	**Reserved**	Reserved.
[17+2n:16+2n] n=0,1..5	**CMPDCTLn**	**EPWM Compare Down Point Control** EPWM can control output level when EPWM counter counts down to CMP. 00 = Do nothing. 01 = EPWM compare down point output Low. 10 = EPWM compare down point output High. 11 = EPWM compare down point output Toggle. **Note:** In complementary mode, CMPDCTL1, 3, 5 is used as another CMPDCTL for channel 0, 2, 4.
[15:12]	**Reserved**	Reserved.
[1+2n:2n] n=0,1..5	**CMPUCTLn**	**EPWM Compare Up Point Control** EPWM can control output level when EPWM counter counts up to CMP. 00 = Do nothing. 01 = EPWM compare up point output Low. 10 = EPWM compare up point output High. 11 = EPWM compare up point output Toggle. **Note:** In complementary mode, CMPUCTL1, 3, 5 is used as another CMPUCTL for channel 0, 2, 4.

EPWM Mask Enable Register (EPWM_MSKEN)

Register	Offset	R/W	Description	Reset Value
EPWM_MSKEN	EPWMx_BA+0xB8	R/W	EPWM Mask Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**MSKEN5**	**MSKEN4**	**MSKEN3**	**MSKEN2**	**MSKEN1**	**MSKEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**MSKENn**	**EPWM Mask Enable Bits** The EPWM output signal will be masked when this bit is enabled. The corresponding EPWM channel n will output MSKDATn (EPWM_MSK[5:0]) data. 0 = EPWM output signal is non-masked. 1 = EPWM output signal is masked and output MSKDATn data.

EPWM Mask DATA Register (EPWM_MSK)

Register	Offset	R/W	Description	Reset Value
EPWM_MSK	EPWMx_BA+0xBC	R/W	EPWM Mask Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MSKDAT5	MSKDAT4	MSKDAT3	MSKDAT2	MSKDAT1	MSKDAT0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**MSKDATn**	**EPWM Mask Data Bit** This data bit control the state of EPWMn output pin, if corresponding mask function is enabled. 0 = Output logic low to EPWM channel n. 1 = Output logic high to EPWM channel n.

EPWM Brake Noise Filter Register (EPWM_BNF)

Register	Offset	R/W	Description	Reset Value
EPWM_BNF	EPWMx_BA+0xC0	R/W	EPWM Brake Noise Filter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							**BK1SRC**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**BK0SRC**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BRK1PINV	**BRK1FCNT**			**BRK1NFSEL**			**BRK1NFEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BRK0PINV	**BRK0FCNT**			**BRK0NFSEL**			**BRK0NFEN**

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**BK1SRC**	**Brake 1 Pin Source Select** For EPWM0 setting: 0 = Brake 1 pin source come from EPWM0_BRAKE1. 1 = Brake 1 pin source come from EPWM1_BRAKE1. For EPWM1 setting: 0 = Brake 1 pin source come from EPWM1_BRAKE1. 1 = Brake 1 pin source come from EPWM0_BRAKE1.
[23:17]	**Reserved**	Reserved.
[16]	**BK0SRC**	**Brake 0 Pin Source Select** For EPWM0 setting: 0 = Brake 0 pin source come from EPWM0_BRAKE0. 1 = Brake 0 pin source come from EPWM1_BRAKE0. For EPWM1 setting: 0 = Brake 0 pin source come from EPWM1_BRAKE0. 1 = Brake 0 pin source come from EPWM0_BRAKE0.
[15]	**BRK1PINV**	**Brake 1 Pin Inverse** 0 **=** brake pin event will be detected if EPWMx_BRAKE1 pin status transfer from low to high in edge-detect, or pin status is high in level-detect. 1 = brake pin event will be detected if EPWMx_BRAKE1 pin status transfer from high to low in edge-detect, or pin status is low in level-detect.
[14:12]	**BRK1FCNT**	**Brake 1 Edge Detector Filter Count** The register bits control the Brake1 filter counter to count from 0 to BRK1FCNT.
[11:9]	**BRK1NFSEL**	**Brake 1 Edge Detector Filter Clock Selection** 000 = Filter clock = HCLK. 001 = Filter clock = HCLK/2. 010 = Filter clock = HCLK/4.

		011 = Filter clock = HCLK/8. 100 = Filter clock = HCLK/16. 101 = Filter clock = HCLK/32. 110 = Filter clock = HCLK/64. 111 = Filter clock = HCLK/128.
[8]	**BRK1NFEN**	**EPWM Brake 1 Noise Filter Enable Bit** 0 = Noise filter of EPWM Brake 1 Disabled. 1 = Noise filter of EPWM Brake 1 Enabled.
[7]	**BRK0PINV**	**Brake 0 Pin Inverse** 0 **=** brake pin event will be detected if EPWMx_BRAKE0 pin status transfer from low to high in edge-detect, or pin status is high in level-detect. 1 = brake pin event will be detected if EPWMx_BRAKE0 pin status transfer from high to low in edge-detect, or pin status is low in level-detect.
[6:4]	**BRK0FCNT**	**Brake 0 Edge Detector Filter Count** The register bits control the Brake0 filter counter to count from 0 to BRK0FCNT.
[3:1]	**BRK0NFSEL**	**Brake 0 Edge Detector Filter Clock Selection** 000 = Filter clock = HCLK. 001 = Filter clock = HCLK/2. 010 = Filter clock = HCLK/4. 011 = Filter clock = HCLK/8. 100 = Filter clock = HCLK/16. 101 = Filter clock = HCLK/32. 110 = Filter clock = HCLK/64. 111 = Filter clock = HCLK/128.
[0]	**BRK0NFEN**	**EPWM Brake 0 Noise Filter Enable Bit** 0 = Noise filter of EPWM Brake 0 Disabled. 1 = Noise filter of EPWM Brake 0 Enabled.

EPWM System Fail Brake Control Register (EPWM_FAILBRK)

Register	Offset	R/W	Description	Reset Value
EPWM_FAILBRK	EPWMx_BA+0xC4	R/W	EPWM System Fail Brake Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**Reserved**	**Reserved**	**CORBRKEN**	**RAMBRKEN**	**BODBRKEN**	**CSSBRKEN**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**CORBRKEN**	**Core Lockup Detection Trigger EPWM Brake Function Enable Bit** 0 = Brake Function triggered by Core lockup detection Disabled. 1 = Brake Function triggered by Core lockup detection Enabled.
[2]	**RAMBRKEN**	**SRAM Parity Error Detection Trigger EPWM Brake Function Enable Bit** 0 = Brake Function triggered by SRAM parity error detection Disabled. 1 = Brake Function triggered by SRAM parity error detection Enabled.
[1]	**BODBRKEN**	**Brown-out Detection Trigger EPWM Brake Function Enable Bit** 0 = Brake Function triggered by BOD Disabled. 1 = Brake Function triggered by BOD Enabled.
[0]	**CSSBRKEN**	**Clock Security System Detection Trigger EPWM Brake Function Enable Bit** 0 = Brake Function triggered by CSS detection Disabled. 1 = Brake Function triggered by CSS detection Enabled.

EPWM Brake Edge Detect Control Register 0_1, 2_3, 4_5 (EPWM_BRKCTL0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_BRKCTL0_1	EPWMx_BA+0xC8	R/W	EPWM Brake Edge Detect Control Register 0/1	0x0000_0000
EPWM_BRKCTL2_3	EPWMx_BA+0xCC	R/W	EPWM Brake Edge Detect Control Register 2/3	0x0000_0000
EPWM_BRKCTL4_5	EPWMx_BA+0xD0	R/W	EPWM Brake Edge Detect Control Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
VBSRLBEN	Reserved	Reserved	EADC0LBEN	Reserved			
23	22	21	20	19	18	17	16
VBSREBEN	Reserved	Reserved	EADC0EBEN	BRKAODD		BRKAEVEN	
15	14	13	12	11	10	9	8
SYSLBEN	VBSNLBEN	BRKP1LEN	BRKP0LEN	Reserved	CPO2LBEN	CPO1LBEN	CPO0LBEN
7	6	5	4	3	2	1	0
SYSEBEN	VBSNEBEN	BRKP1EEN	BRKP0EEN	Reserved	CPO2EBEN	CPO1EBEN	CPO0EBEN

Bits	Description	
[31]	**VBSRLBEN**	**UTCPD0 V_{BUS} Source Disable As Level-detect Brake Source (Write Protect)** 0 = UTCPD0 V_{BUS} source disable as level-detect brake source Disabled. 1 = UTCPD0 V_{BUS} source disable as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[30]	**Reserved**	Reserved.
[29]	**Reserved**	Reserved.
[28]	**EADC0LBEN**	**Enable EADC0 Result Monitor as Level-detect Brake Source (Write Protect)** 0 = EADC0RM as level-detect brake source Disabled. 1 = EADC0RM as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[27:24]	**Reserved**	Reserved.
[23]	**VBSREBEN**	**UTCPD0 V_{BUS} Source Disable As Edge-detect Brake Source (Write Protect)** 0 = UTCPD0 V_{BUS} source disable as edge-detect brake source Disabled. 1 = UTCPD0 V_{BUS} source disable as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[22]	**Reserved**	Reserved.
[21]	**Reserved**	Reserved.
[20]	**EADC0EBEN**	**Enable EADC0 Result Monitor as Edge-detect Brake Source (Write Protect)** 0 = EADC0RM as edge-detect brake source Disabled. 1 = EADC0RM as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[19:18]	**BRKAODD**	**EPWM Brake Action Select for Odd Channel (Write Protect)**

		00 = EPWMx brake event will not affect odd channels output. 01 = EPWM odd channel output tri-state when EPWMx brake event happened. 10 = EPWM odd channel output low level when EPWMx brake event happened. 11 = EPWM odd channel output high level when EPWMx brake event happened. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[17:16]	**BRKAEVEN**	**EPWM Brake Action Select for Even Channel (Write Protect)** 00 = EPWMx brake event will not affect even channels output. 01 = EPWM even channel output tri-state when EPWMx brake event happened. 10 = EPWM even channel output low level when EPWMx brake event happened. 11 = EPWM even channel output high level when EPWMx brake event happened. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[15]	**SYSLBEN**	**Enable System Fail As Level-detect Brake Source (Write Protect)** 0 = System Fail condition as level-detect brake source Disabled. 1 = System Fail condition as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[14]	**VBSNLBEN**	**UTCPD0 V_{BUS} Sink Disable As Level-detect Brake Source (Write Protect)** 0 = UTCPD0 V_{BUS} sink disable as level-detect brake source Disabled. 1 = UTCPD0 V_{BUS} sink disable as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[13]	**BRKP1LEN**	**Enable BKP1 Pin As Level-detect Brake Source (Write Protect)** 0 = EPWMx_BRAKE1 pin as level-detect brake source Disabled. 1 = EPWMx_BRAKE1 pin as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[12]	**BRKP0LEN**	**Enable BKP0 Pin As Level-detect Brake Source (Write Protect)** 0 = EPWMx_BRAKE0 pin as level-detect brake source Disabled. 1 = EPWMx_BRAKE0 pin as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[11]	**Reserved**	Reserved.
[10]	**CPO2LBEN**	**Enable ACMP2_O Digital Output As Level-detect Brake Source (Write Protect)** 0 = ACMP2_O as level-detect brake source Disabled. 1 = ACMP2_O as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[9]	**CPO1LBEN**	**Enable ACMP1_O Digital Output As Level-detect Brake Source (Write Protect)** 0 = ACMP1_O as level-detect brake source Disabled. 1 = ACMP1_O as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[8]	**CPO0LBEN**	**Enable ACMP0_O Digital Output As Level-detect Brake Source (Write Protect)** 0 = ACMP0_O as level-detect brake source Disabled. 1 = ACMP0_O as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7]	**SYSEBEN**	**Enable System Fail As Edge-detect Brake Source (Write Protect)** 0 = System Fail condition as edge-detect brake source Disabled. 1 = System Fail condition as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[6]	**VBSNEBEN**	**UTCPD0 V_{BUS} Sink Disable As Edge-detect Brake Source (Write Protect)**

		0 = UTCPD0 V_{BUS} sink disable as edge-detect brake source Disabled. 1 = UTCPD0 V_{BUS} sink disable as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[5]	**BRKP1EEN**	**Enable EPWMx_BRAKE1 Pin As Edge-detect Brake Source (Write Protect)** 0 = EPWMx_BRAKE1 pin as edge-detect brake source Disabled. 1 = EPWMx_BRAKE1 pin as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[4]	**BRKP0EEN**	**Enable EPWMx_BRAKE0 Pin As Edge-detect Brake Source (Write Protect)** 0 = EPWMx_BRAKE0 pin as edge-detect brake source Disabled. 1 = EPWMx_BRAKE0 pin as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[3]	**Reserved**	Reserved.
[2]	**CPO2EBEN**	**Enable ACMP2_O Digital Output As Edge-detect Brake Source (Write Protect)** 0 = ACMP2_O as edge-detect brake source Disabled. 1 = ACMP2_O as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[1]	**CPO1EBEN**	**Enable ACMP1_O Digital Output As Edge-detect Brake Source (Write Protect)** 0 = ACMP1_O as edge-detect brake source Disabled. 1 = ACMP1_O as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[0]	**CPO0EBEN**	**Enable ACMP0_O Digital Output As Edge-detect Brake Source (Write Protect)** 0 = ACMP0_O as edge-detect brake source Disabled. 1 = ACMP0_O as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Pin Polar Inverse Control (EPWM_POLCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_POLCTL	EPWMx_BA+0xD4	R/W	EPWM Pin Polar Inverse Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**PINV5**	**PINV4**	**PINV3**	**PINV2**	**PINV1**	**PINV0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**PINVn**	**EPWM PIN Polar Inverse Control** The register controls polarity state of EPWMx_CHn output pin. 0 = EPWMx_CHn output pin polar inverse Disabled. 1 = EPWMx_CHn output pin polar inverse Enabled.

EPWM Output Enable Register (EPWM_POEN)

Register	Offset	R/W	Description	Reset Value
EPWM_POEN	EPWMx_BA+0xD8	R/W	EPWM Output Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**POEN5**	**POEN4**	**POEN3**	**POEN2**	**POEN1**	**POEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**POENn**	**EPWM Pin Output Enable Bits** 0 = EPWMx_CHn pin at tri-state. 1 = EPWMx_CHn pin in output mode.

EPWM Software Brake Control Register (EPWM_SWBRK)

Register	Offset	R/W	Description	Reset Value
EPWM_SWBRK	EPWMx_BA+0xDC	W	EPWM Software Brake Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**BRKLTRG4**	**BRKLTRG2**	**BRKLTRG0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**BRKETRG4**	**BRKETRG2**	**BRKETRG0**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[8+n/2] n=0,2,4	**BRKLTRGn**	**EPWM Level Brake Software Trigger (Write Only) (Write Protect)** Write 1 to this bit will trigger level brake, and set BRKLIFn to 1 in EPWM_INTSTS1 register. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**BRKETRGn**	**EPWM Edge Brake Software Trigger (Write Only) (Write Protect)** Write 1 to this bit will trigger edge brake, and set BRKEIFn to 1 in EPWM_INTSTS1 register. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Interrupt Enable Register 0 (EPWM_INTEN0)

Register	Offset	R/W	Description	Reset Value
EPWM_INTEN0	EPWMx_BA+0xE0	R/W	EPWM Interrupt Enable Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIEN5	CMPDIEN4	CMPDIEN3	CMPDIEN2	CMPDIEN1	CMPDIEN0
23	22	21	20	19	18	17	16
Reserved		CMPUIEN5	CMPUIEN4	CMPUIEN3	CMPUIEN2	CMPUIEN1	CMPUIEN0
15	14	13	12	11	10	9	8
Reserved		PIEN5	PIEN4	PIEN3	PIEN2	PIEN1	PIEN0
7	6	5	4	3	2	1	0
Reserved		ZIEN5	ZIEN4	ZIEN3	ZIEN2	ZIEN1	ZIEN0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CMPDIENn**	**EPWM Compare Down Count Interrupt Enable Bits** 0 = Compare down count interrupt Disabled. 1 = Compare down count interrupt Enabled. **Note:** In complementary mode, CMPDIEN1, 3, 5 is used as another CMPDIEN for channel 0, 2, 4.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CMPUIENn**	**EPWM Compare Up Count Interrupt Enable Bits** 0 = Compare up count interrupt Disabled. 1 = Compare up count interrupt Enabled. **Note:** In complementary mode, CMPUIEN1, 3, 5 is used as another CMPUIEN for channel 0, 2, 4.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**PIENn**	**EPWM Period Point Interrupt Enable Bits** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note 1:** When up-down counter type period point means center point. **Note 2:** Odd channels will read always 0 at complementary mode.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**ZIENn**	**EPWM Zero Point Interrupt Enable Bits** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.

EPWM Interrupt Enable Register 1 (EPWM_INTEN1)

Register	Offset	R/W	Description	Reset Value
EPWM_INTEN1	EPWMx_BA+0xE4	R/W	EPWM Interrupt Enable Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**BRKLIEN4_5**	**BRKLIEN2_3**	**BRKLIEN0_1**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**BRKEIEN4_5**	**BRKEIEN2_3**	**BRKEIEN0_1**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**BRKLIEN4_5**	**EPWM Level-detect Brake Interrupt Enable for Channel4/5 (Write Protect)** 0 = Level-detect Brake interrupt for channel4/5 Disabled. 1 = Level-detect Brake interrupt for channel4/5 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[9]	**BRKLIEN2_3**	**EPWM Level-detect Brake Interrupt Enable for Channel2/3 (Write Protect)** 0 = Level-detect Brake interrupt for channel2/3 Disabled. 1 = Level-detect Brake interrupt for channel2/3 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[8]	**BRKLIEN0_1**	**EPWM Level-detect Brake Interrupt Enable for Channel0/1 (Write Protect)** 0 = Level-detect Brake interrupt for channel0/1 Disabled. 1 = Level-detect Brake interrupt for channel0/1 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[2]	**BRKEIEN4_5**	**EPWM Edge-detect Brake Interrupt Enable for Channel4/5 (Write Protect)** 0 = Edge-detect Brake interrupt for channel4/5 Disabled. 1 = Edge-detect Brake interrupt for channel4/5 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[1]	**BRKEIEN2_3**	**EPWM Edge-detect Brake Interrupt Enable for Channel2/3 (Write Protect)** 0 = Edge-detect Brake interrupt for channel2/3 Disabled. 1 = Edge-detect Brake interrupt for channel2/3 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[0]	**BRKEIEN0_1**	**EPWM Edge-detect Brake Interrupt Enable for Channel0/1 (Write Protect)** 0 = Edge-detect Brake interrupt for channel0/1 Disabled. 1 = Edge-detect Brake interrupt for channel0/1 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Interrupt Flag Register 0 (EPWM_INTSTS0)

Register	Offset	R/W	Description	Reset Value
EPWM_INTSTS0	EPWMx_BA+0xE8	R/W	EPWM Interrupt Flag Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	22	21	20	19	18	17	16
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
15	14	13	12	11	10	9	8
Reserved		PIF5	PIF4	PIF3	PIF2	PIF1	PIF0
7	6	5	4	3	2	1	0
Reserved		ZIF5	ZIF4	ZIF3	ZIF2	ZIF1	ZIF0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CMPDIFn**	**EPWM Compare Down Count Interrupt Flag** Flag is set by hardware when EPWM counter down count and reaches EPWM_CMPDATn, software can clear this bit by writing 1 to it. **Note:** In complementary mode, CMPDIF1, 3, 5 is used as another CMPDIF for channel 0, 2, 4.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CMPUIFn**	**EPWM Compare Up Count Interrupt Flag** Flag is set by hardware when EPWM counter up count and reaches EPWM_CMPDATn, software can clear this bit by writing 1 to it. **Note:** In complementary mode, CMPUIF1, 3, 5 is used as another CMPUIF for channel 0, 2, 4.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**PIFn**	**EPWM Period Point Interrupt Flag** This bit is set by hardware when EPWM counter reaches EPWM_PERIODn. **Note:** This bit can be cleared to 0 by software writing 1.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**ZIFn**	**EPWM Zero Point Interrupt Flag** This bit is set by hardware when EPWM counter reaches 0. **Note:** This bit can be cleared to 0 by software writing 1

EPWM Interrupt Flag Register 1 (EPWM_INTSTS1)

Register	Offset	R/W	Description	Reset Value
EPWM_INTSTS1	EPWMx_BA+0xEC	R/W	EPWM Interrupt Flag Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		BRKLSTS5	BRKLSTS4	BRKLSTS3	BRKLSTS2	BRKLSTS1	BRKLSTS0
23	22	21	20	19	18	17	16
Reserved		BRKESTS5	BRKESTS4	BRKESTS3	BRKESTS2	BRKESTS1	BRKESTS0
15	14	13	12	11	10	9	8
Reserved		BRKLIF5	BRKLIF4	BRKLIF3	BRKLIF2	BRKLIF1	BRKLIF0
7	6	5	4	3	2	1	0
Reserved		BRKEIF5	BRKEIF4	BRKEIF3	BRKEIF2	BRKEIF1	BRKEIF0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**BRKLSTSn**	**EPWM Channel N Level-detect Brake Status (Read Only)** 0 = EPWM channel n level-detect brake state is released. 1 = When EPWM channel n level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the EPWM channel n at brake state. **Note:** This bit is read only and auto cleared by hardware. When enabled brake source return to high level, EPWM will release brake state until current EPWM period finished. The EPWM waveform will start output from next full EPWM period.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**BRKESTSn**	**EPWM Channel N Edge-detect Brake Status (Read Only)** 0 = EPWM channel n edge-detect brake state is released. 1 = When EPWM channel n edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the EPWM channel n at brake state. **Note:** This bit is read only and auto cleared by hardware. When edge-detect brake interrupt flag is cleared, EPWM will release brake state until current EPWM period finished. The EPWM waveform will start output from next full EPWM period.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**BRKLIFn**	**EPWM Channel N Level-detect Brake Interrupt Flag (Write Protect)** 0 = EPWM channel n level-detect brake event do not happened. 1 = When EPWM channel n level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**BRKEIFn**	**EPWM Channel N Edge-detect Brake Interrupt Flag (Write Protect)** 0 = EPWM channel n edge-detect brake event do not happened. 1 = When EPWM channel n edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Trigger DAC Enable Register (EPWM_DACTRGEN)

Register	Offset	R/W	Description	Reset Value
EPWM_DACTRGEN	EPWMx_BA+0xF4	R/W	EPWM Trigger DAC Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CDTRGEN5	CDTRGEN4	CDTRGEN3	CDTRGEN2	CDTRGEN1	CDTRGEN0
23	22	21	20	19	18	17	16
Reserved		CUTRGEN5	CUTRGEN4	CUTRGEN3	CUTRGEN2	CUTRGEN1	CUTRGEN0
15	14	13	12	11	10	9	8
Reserved		PTE5	PTE4	PTE3	PTE2	PTE1	PTE0
7	6	5	4	3	2	1	0
Reserved		ZTE5	ZTE4	ZTE3	ZTE2	ZTE1	ZTE0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CDTRGENn**	**EPWM Compare Down Count Point Trigger DAC Enable Bits** EPWM can trigger DAC to start action when EPWM counter down count to CMP if this bit is set to1. 0 = EPWM Compare Down count point trigger DAC function Disabled. 1 = EPWM Compare Down count point trigger DAC function Enabled. **Note 1:** This bit should keep at 0 when EPWM counter operating in up counter type. **Note 2:** In complementary mode, CDTRGEN1, 3, 5 is used as another CDTRGEN for channel 0, 2, 4.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CUTRGENn**	**EPWM Compare Up Count Point Trigger DAC Enable Bits** EPWM can trigger DAC to start action when EPWM counter counts up to CMP if this bit is set to1. 0 = EPWM Compare Up point trigger DAC function Disabled. 1 = EPWM Compare Up point trigger DAC function Enabled. **Note 1:** This bit should keep at 0 when EPWM counter operating in down counter type. **Note 2:** In complementary mode, CUTRGEN1, 3, 5 is used as another CUTRGEN for channel 0, 2, 4.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**PTEn**	**EPWM Period Point Trigger DAC Enable Bits** EPWM can trigger DAC to start action when EPWM counter counts up to (PERIODn+1) if this bit is set to1. 0 = EPWM period point trigger DAC function Disabled. 1 = EPWM period point trigger DAC function Enabled.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**ZTEn**	**EPWM Zero Point Trigger DAC Enable Bits** EPWM can trigger EADC/DAC/DMA to start action when EPWM counter down count to zero if this bit is set to1.

		0 = EPWM period point trigger DAC function Disabled. 1 = EPWM period point trigger DAC function Enabled.

EPWM Trigger EADC Source Select Register 0 (EPWM_EADCTS0)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCTS0	EPWMx_BA+0xF8	R/W	EPWM Trigger EADC Source Select Register 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved		TRGSEL3				
23	22	21	20	19	18	17	16
TRGEN2	Reserved		TRGSEL2				
15	14	13	12	11	10	9	8
TRGEN1	Reserved		TRGSEL1				
7	6	5	4	3	2	1	0
TRGEN0	Reserved		TRGSEL0				

Bits	Description	
[31]	**TRGEN3**	**EPWM_CH3 Trigger EADC Enable Bit** 0 = EPWM_CH3 Trigger EADC function Disabled. 1 = EPWM_CH3 Trigger EADC function Enabled.
[30:29]	**Reserved**	Reserved.
[28:24]	**TRGSEL3**	**EPWM_CH3 Trigger EADC Source Select** 00000 = EPWM_CH2 zero point. 00001 = EPWM_CH2 period point. 00010 = EPWM_CH2 zero or period point. 00011 = EPWM_CH2 up-count compared point. 00100 = EPWM_CH2 down-count compared point. 00101 = EPWM_CH3 zero point. 00110 = EPWM_CH3 period point. 00111 = EPWM_CH3 zero or period point. 01000 = EPWM_CH3 up-count compared point. 01001 = EPWM_CH3 down-count compared point. 01010 = EPWM_CH0 up-count free trigger compared point. 01011 = EPWM_CH0 down-count free trigger compared point. 01100 = EPWM_CH2 up-count free trigger compared point. 01101 = EPWM_CH2 down-count free trigger compared point. 01110 = EPWM_CH4 up-count free trigger compared point. 01111 = EPWM_CH4 down-count free trigger compared point. 10000 = EPWM_CH2 Interrupt Flag Accumulator Interrupt. 10001 = EPWM_CH3 Interrupt Flag Accumulator Interrupt.
[23]	**TRGEN2**	**EPWM_CH2 Trigger EADC Enable Bit** 0 = EPWM_CH2 Trigger EADC function Disabled. 1 = EPWM_CH2 Trigger EADC function Enabled.
[22:21]	**Reserved**	Reserved.

[20:16]	**TRGSEL2**	**EPWM_CH2 Trigger EADC Source Select** 00000 = EPWM_CH2 zero point. 00001 = EPWM_CH2 period point. 00010 = EPWM_CH2 zero or period point. 00011 = EPWM_CH2 up-count compared point. 00100 = EPWM_CH2 down-count compared point. 00101 = EPWM_CH3 zero point. 00110 = EPWM_CH3 period point. 00111 = EPWM_CH3 zero or period point. 01000 = EPWM_CH3 up-count compared point. 01001 = EPWM_CH3 down-count compared point. 01010 = EPWM_CH0 up-count free trigger compared point. 01011 = EPWM_CH0 down-count free trigger compared point. 01100 = EPWM_CH2 up-count free trigger compared point. 01101 = EPWM_CH2 down-count free trigger compared point. 01110 = EPWM_CH4 up-count free trigger compared point. 01111 = EPWM_CH4 down-count free trigger compared point. 10000 = EPWM_CH2 Interrupt Flag Accumulator Interrupt. 10001 = EPWM_CH3 Interrupt Flag Accumulator Interrupt.
[15]	**TRGEN1**	**EPWM_CH1 Trigger EADC Enable Bit** 0 = EPWM_CH1 Trigger EADC function Disabled. 1 = EPWM_CH1 Trigger EADC function Enabled.
[14:13]	**Reserved**	Reserved.
[12:8]	**TRGSEL1**	**EPWM_CH1 Trigger EADC Source Select** 00000 = EPWM_CH0 zero point. 00001 = EPWM_CH0 period point. 00010 = EPWM_CH0 zero or period point. 00011 = EPWM_CH0 up-count compared point. 00100 = EPWM_CH0 down-count compared point. 00101 = EPWM_CH1 zero point. 00110 = EPWM_CH1 period point. 00111 = EPWM_CH1 zero or period point. 01000 = EPWM_CH1 up-count compared point. 01001 = EPWM_CH1 down-count compared point. 01010 = EPWM_CH0 up-count free trigger compared point. 01011 = EPWM_CH0 down-count free trigger compared point. 01100 = EPWM_CH2 up-count free trigger compared point. 01101 = EPWM_CH2 down-count free trigger compared point. 01110 = EPWM_CH4 up-count free trigger compared point. 01111 = EPWM_CH4 down-count free trigger compared point. 10000 = EPWM_CH0 Interrupt Flag Accumulator Interrupt. 10001 = EPWM_CH1 Interrupt Flag Accumulator Interrupt.
[7]	**TRGEN0**	**EPWM_CH0 Trigger EADC Enable Bit** 0 = EPWM_CH0 Trigger EADC function Disabled. 1 = EPWM_CH0 Trigger EADC function Enabled.
[6:5]	**Reserved**	Reserved.

[4:0]	**TRGSEL0**	**EPWM_CH0 Trigger EADC Source Select** 00000 = EPWM_CH0 zero point. 00001 = EPWM_CH0 period point. 00010 = EPWM_CH0 zero or period point. 00011 = EPWM_CH0 up-count compared point. 00100 = EPWM_CH0 down-count compared point. 00101 = EPWM_CH1 zero point. 00110 = EPWM_CH1 period point. 00111 = EPWM_CH1 zero or period point. 01000 = EPWM_CH1 up-count compared point. 01001 = EPWM_CH1 down-count compared point. 01010 = EPWM_CH0 up-count free trigger compared point. 01011 = EPWM_CH0 down-count free trigger compared point. 01100 = EPWM_CH2 up-count free trigger compared point. 01101 = EPWM_CH2 down-count free trigger compared point. 01110 = EPWM_CH4 up-count free trigger compared point. 01111 = EPWM_CH4 down-count free trigger compared point. 10000 = EPWM_CH0 Interrupt Flag Accumulator Interrupt. 10001 = EPWM_CH1 Interrupt Flag Accumulator Interrupt.

EPWM Trigger EADC Source Select Register 1 (EPWM_EADCTS1)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCTS1	EPWMx_BA+0xFC	R/W	EPWM Trigger EADC Source Select Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TRGEN5	Reserved		TRGSEL5				
7	6	5	4	3	2	1	0
TRGEN4	Reserved		TRGSEL4				

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**TRGEN5**	**EPWM_CH5 Trigger EADC Enable Bit** 0 = EPWM_CH5 Trigger EADC function Disabled. 1 = EPWM_CH5 Trigger EADC function Enabled.
[14:13]	**Reserved**	Reserved.
[12:8]	**TRGSEL5**	**EPWM_CH5 Trigger EADC Source Select** 00000 = EPWM_CH4 zero point. 00001 = EPWM_CH4 period point. 00010 = EPWM_CH4 zero or period point. 00011 = EPWM_CH4 up-count compared point. 00100 = EPWM_CH4 down-count compared point. 00101 = EPWM_CH5 zero point. 00110 = EPWM_CH5 period point. 00111 = EPWM_CH5 zero or period point. 01000 = EPWM_CH5 up-count compared point. 01001 = EPWM_CH5 down-count compared point. 01010 = EPWM_CH0 up-count free trigger compared point. 01011 = EPWM_CH0 down-count free trigger compared point. 01100 = EPWM_CH2 up-count free trigger compared point. 01101 = EPWM_CH2 down-count free trigger compared point. 01110 = EPWM_CH4 up-count free trigger compared point. 01111 = EPWM_CH4 down-count free trigger compared point. 10000 = EPWM_CH4 Interrupt Flag Accumulator Interrupt. 10001 = EPWM_CH5 Interrupt Flag Accumulator Interrupt.
[7]	**TRGEN4**	**EPWM_CH4 Trigger EADC Enable Bit** 0 = EPWM_CH4 Trigger EADC function Disabled. 1 = EPWM_CH4 Trigger EADC function Enabled.

[6:5]	**Reserved**	Reserved.
[4:0]	**TRGSEL4**	**EPWM_CH4 Trigger EADC Source Select** 00000 = EPWM_CH4 zero point. 00001 = EPWM_CH4 period point. 00010 = EPWM_CH4 zero or period point. 00011 = EPWM_CH4 up-count compared point. 00100 = EPWM_CH4 down-count compared point. 00101 = EPWM_CH5 zero point. 00110 = EPWM_CH5 period point. 00111 = EPWM_CH5 zero or period point. 01000 = EPWM_CH5 up-count compared point. 01001 = EPWM_CH5 down-count compared point. 01010 = EPWM_CH0 up-count free trigger compared point. 01011 = EPWM_CH0 down-count free trigger compared point. 01100 = EPWM_CH2 up-count free trigger compared point. 01101 = EPWM_CH2 down-count free trigger compared point. 01110 = EPWM_CH4 up-count free trigger compared point. 01111 = EPWM_CH4 down-count free trigger compared point. 10000 = EPWM_CH4 Interrupt Flag Accumulator Interrupt. 10001 = EPWM_CH5 Interrupt Flag Accumulator Interrupt.

EPWM Free Trigger Compare Register 0_1, 2_3, 4_5 (EPWM_FTCMPDAT0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_FTCMPDAT0_1	EPWMx_BA+0x100	R/W	EPWM Free Trigger Compare Register 0/1	0x0000_0000
EPWM_FTCMPDAT2_3	EPWMx_BA+0x104	R/W	EPWM Free Trigger Compare Register 2/3	0x0000_0000
EPWM_FTCMPDAT4_5	EPWMx_BA+0x108	R/W	EPWM Free Trigger Compare Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FTCMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FTCMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FTCMP**	**EPWM Free Trigger Compare Register** FTCMP use to compare with even CNT (EPWM_CNTm[15:0], m=0,2,4) to trigger EADC. EPWM_FTCMPDAT0_1, EPWM_FTCMPDAT2_3, EPWM_FTCMPDAT4_5 corresponding complementary pairs EPWM_CH0 and EPWM_CH1, EPWM_CH2 and EPWM_CH3, EPWM_CH4 and EPWM_CH5.

EPWM Synchronous Start Control Register (EPWM_SSCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_SSCTL	EPWMx_BA+0x110	R/W	EPWM Synchronous Start Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SSRC**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**SSEN5**	**SSEN4**	**SSEN3**	**SSEN2**	**SSEN1**	**SSEN0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**SSRC**	**EPWM Synchronous Start Source Select Bits** 00 = Synchronous start source come from EPWM0. 01 = Synchronous start source come from EPWM1. 10 = Synchronous start source come from PWM0. 11 = Synchronous start source come from PWM1.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**SSENn**	**EPWM Synchronous Start Function Enable Bits** When synchronous start function is enabled, the EPWM counter enable register (EPWM_CNTEN) can be enabled by writing EPWM synchronous start trigger bit (CNTSEN). 0 = EPWM synchronous start function Disabled. 1 = EPWM synchronous start function Enabled.

EPWM Synchronous Start Trigger Register (EPWM_SSTRG)

Register	Offset	R/W	Description	Reset Value
EPWM_SSTRG	EPWMx_BA+0x114	W	EPWM Synchronous Start Trigger Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**CNTSEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTSEN**	**EPWM Counter Synchronous Start Enable (Write Only)** PMW counter synchronous enable function is used to make selected EPWM channels (include EPWM0_CHx and EPWM1_CHx) start counting at the same time. Writing this bit to 1 will also set the counter enable bit (CNTENn, n denotes channel 0 to 5) if correlated EPWM channel counter synchronous start function is enabled.

EPWM Leading Edge Blanking Control Register (EPWM_LEBCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_LEBCTL	EPWMx_BA+0x118	R/W	EPWM Leading Edge Blanking Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						TRGTYPE	
15	14	13	12	11	10	9	8
Reserved					SRCEN4	SRCEN2	SRCEN0
7	6	5	4	3	2	1	0
Reserved							LEBEN

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**TRGTYPE**	**EPWM Leading Edge Blanking Trigger Type** 0 = When detect leading edge blanking source rising edge, blanking counter start counting. 1 = When detect leading edge blanking source falling edge, blanking counter start counting. 2 = When detect leading edge blanking source rising or falling edge, blanking counter start counting. 3 = Reserved.
[15:11]	**Reserved**	Reserved.
[10]	**SRCEN4**	**EPWM Leading Edge Blanking Source From EPWM_CH4 Enable Bit** 0 = EPWM Leading Edge Blanking Source from EPWM_CH4 Disabled. 1 = EPWM Leading Edge Blanking Source from EPWM_CH4 Enabled.
[9]	**SRCEN2**	**EPWM Leading Edge Blanking Source From EPWM_CH2 Enable Bit** 0 = EPWM Leading Edge Blanking Source from EPWM_CH2 Disabled. 1 = EPWM Leading Edge Blanking Source from EPWM_CH2 Enabled.
[8]	**SRCEN0**	**EPWM Leading Edge Blanking Source From EPWM_CH0 Enable Bit** 0 = EPWM Leading Edge Blanking Source from EPWM_CH0 Disabled. 1 = EPWM Leading Edge Blanking Source from EPWM_CH0 Enabled.
[7:1]	**Reserved**	Reserved.
[0]	**LEBEN**	**EPWM Leading Edge Blanking Enable Bit** 0 = EPWM Leading Edge Blanking Disabled. 1 = EPWM Leading Edge Blanking Enabled.

EPWM Leading Edge Blanking Counter Register (EPWM_LEBCNT)

Register	Offset	R/W	Description	Reset Value
EPWM_LEBCNT	EPWMx_BA+0x11C	R/W	EPWM Leading Edge Blanking Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**LEBCNT**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LEBCNT							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**LEBCNT**	**EPWM Leading Edge Blanking Counter** This counter value decides leading edge blanking window size. Blanking window size = LEBCNT+1, and LEB counter clock base is EPWM_CLK.

EPWM Status Register (EPWM_STATUS)

Register	Offset	R/W	Description	Reset Value
EPWM_STATUS	EPWMx_BA+0x120	R/W	EPWM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							DACTRGF
23	22	21	20	19	18	17	16
LPADCTRGF	Reserved	EADCTRGF5	EADCTRGF4	EADCTRGF3	EADCTRGF2	EADCTRGF1	EADCTRGF0
15	14	13	12	11	10	9	8
Reserved					SYNCINF4	SYNCINF2	SYNCINF0
7	6	5	4	3	2	1	0
Reserved		CNTMAXF5	CNTMAXF4	CNTMAXF3	CNTMAXF2	CNTMAXF1	CNTMAXF0

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**DACTRGF**	**DAC Start of Conversion Flag** 0 = No DAC start of conversion trigger event has occurred. 1 = A DAC start of conversion trigger event has occurred. **Note:** This bit can be cleared by software writing 1.
[23]	**LPADCTRGF**	**LPADC Start of Conversion Flag** 0 = No LPADC start of conversion trigger event has occurred. 1 = A LPADC start of conversion trigger event has occurred. **Note:** This bit can be cleared by software writing 1.
[22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**EADCTRGFn**	**EADC Start of Conversion Flag** 0 = No EADC start of conversion trigger event has occurred. 1 = An EADC start of conversion trigger event has occurred. **Note:** This bit can be cleared by software writing 1.
[15:11]	**Reserved**	Reserved.
[8+n/2] n=0,2,4	**SYNCINFn**	**Input Synchronization Latched Flag** 0 = No SYNC_IN event has occurred. 1 = A SYNC_IN event has occurred. **Note:** This bit can be cleared by software writing 1.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CNTMAXFn**	**Time-base Counter Equal to 0xFFFF Latched Flag** 0 = The time-base counter never reached its maximum value 0xFFFF. 1 = The time-base counter reached its maximum value. **Note:** This bit can be cleared by software writing 1.

EPWM Interrupt Flag Accumulator Register (EPWM_IFAn)

Register	Offset	R/W	Description	Reset Value
EPWM_IFA0	EPWMx_BA+0x130	R/W	EPWM Interrupt Flag Accumulator Register 0	0x0000_0000
EPWM_IFA1	EPWMx_BA+0x134	R/W	EPWM Interrupt Flag Accumulator Register 1	0x0000_0000
EPWM_IFA2	EPWMx_BA+0x138	R/W	EPWM Interrupt Flag Accumulator Register 2	0x0000_0000
EPWM_IFA3	EPWMx_BA+0x13C	R/W	EPWM Interrupt Flag Accumulator Register 3	0x0000_0000
EPWM_IFA4	EPWMx_BA+0x140	R/W	EPWM Interrupt Flag Accumulator Register 4	0x0000_0000
EPWM_IFA5	EPWMx_BA+0x144	R/W	EPWM Interrupt Flag Accumulator Register 5	0x0000_0000

31	30	29	28	27	26	25	24
IFAEN	Reserved	IFASEL		Reserved			STPMOD
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
IFACNT							
7	6	5	4	3	2	1	0
IFACNT							

Bits	Description	
[31]	**IFAEN**	**EPWM_CHn Interrupt Flag Accumulator Enable Bits** 0 = EPWM_CHn interrupt flag accumulator Disabled. 1 = EPWM_CHn interrupt flag accumulator Enabled.
[30]	**Reserved**	Reserved.
[29:28]	**IFASEL**	**EPWM_CHn Interrupt Flag Accumulator Source Select** 00 = EPWM_CHn zero point. 01 = EPWM_CHn period in channel n. 10 = EPWM_CHn up-count compared point. 11 = EPWM_CHn down-count compared point.
[27:25]	**Reserved**	Reserved.
[24]	**STPMOD**	**EPWM_CHn Accumulator Stop Mode Enable Bits** 0 = EPWM_CHn Stop Mode Disable. 1 = EPWM_CHn Stop Mode Enable.
[23:16]	**Reserved**	Reserved.
[15:0]	**IFACNT**	**EPWM_CHn Interrupt Flag Counter** The register sets the count number which defines (IFACNT+1) times of EPWM_CHn period occurs to set bit IFAIFn to request the EPWM period interrupt. EPWM flag will be set in every IFACNT[15:0] times of EPWM period.

EPWM Accumulator Interrupt Flag Register (EPWM_AINTSTS)

Register	Offset	R/W	Description	Reset Value
EPWM_AINTSTS	EPWMx_BA+0x150	R/W	EPWM Accumulator Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**IFAIF5**	**IFAIF4**	**IFAIF3**	**IFAIF2**	**IFAIF1**	**IFAIF0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**IFAIFn**	**EPWM_CHn Interrupt Flag Accumulator Interrupt Flag** Flag is set by hardware when condition match IFASEL in EPWM_IFAn register, software can clear this bit by writing 1 to it.

EPWM Accumulator Interrupt Enable Register (EPWM_AINTEN)

Register	Offset	R/W	Description	Reset Value
EPWM_AINTEN	EPWMx_BA+0x154	R/W	EPWM Accumulator Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**IFAIEN5**	**IFAIEN4**	**IFAIEN3**	**IFAIEN2**	**IFAIEN1**	**IFAIEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**IFAIENn**	**EPWM_CHn Interrupt Flag Accumulator Interrupt Enable Bits** 0 = Interrupt Flag accumulator interrupt Disabled. 1 = Interrupt Flag accumulator interrupt Enabled.

EPWM Accumulator PDMA Control Register (EPWM_APDMACTL)

Register	Offset	R/W	Description	Reset Value
EPWM_APDMACTL	EPWMx_BA+0x158	R/W	EPWM Accumulator PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		APDMAEN5	APDMAEN4	APDMAEN3	APDMAEN2	APDMAEN1	APDMAEN0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**APDMAENn**	**Channel n Accumulator PDMA Enable Bits** 0 = Channel n PDMA function Disabled. 1 = Channel n PDMA function Enabled for the channel n to trigger PDMA to transfer memory data to register.

EPWM Fault Detect Enable Register (EPWM_FDEN)

Register	Offset	R/W	Description	Reset Value
EPWM_FDEN	EPWMx_BA+0x160	R/W	EPWM Fault Detect Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**FDCKS5**	**FDCKS4**	**FDCKS3**	**FDCKS2**	**FDCKS1**	**FDCKS0**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**FDODIS5**	**FDODIS4**	**FDODIS3**	**FDODIS2**	**FDODIS1**	**FDODIS0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**FDEN5**	**FDEN4**	**FDEN3**	**FDEN2**	**FDEN1**	**FDEN0**

Bits	Description	
[31:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**FDCKSn**	**EPWM Channel n Fault Detect Clock Source Select Bit** 0 = EPWMx_CLK, x denotes 0 or 1. 1 = EPWMx_CLK divide by prescaler, x denotes 0 or 1.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**FDODISn**	**EPWM Channel n Output Fault Detect Disable Bit** 0 = EPWM detect fault and output Enable. 1 = EPWM detect fault and output Disable.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**FDENn**	**EPWM Fault Detect Function Enable Bit** 0 = Fault detect function Disable. 1 = Fault detect function Enable.

EPWM Fault Detect Control Register 0~5 (EPWM_FDCTL0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_FDCTL0	EPWMx_BA+0x164	R/W	EPWM Fault Detect Control Register 0	0x0000_0000
EPWM_FDCTL1	EPWMx_BA+0x168	R/W	EPWM Fault Detect Control Register 1	0x0000_0000
EPWM_FDCTL2	EPWMx_BA+0x16C	R/W	EPWM Fault Detect Control Register 2	0x0000_0000
EPWM_FDCTL3	EPWMx_BA+0x170	R/W	EPWM Fault Detect Control Register 3	0x0000_0000
EPWM_FDCTL4	EPWMx_BA+0x174	R/W	EPWM Fault Detect Control Register 4	0x0000_0000
EPWM_FDCTL5	EPWMx_BA+0x178	R/W	EPWM Fault Detect Control Register 5	0x0000_0000

31	30	29	28	27	26	25	24
FDDGEN	**Reserved**	**FDCKSEL**		**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**DGSMPCYC**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FDMSKEN	**Reserved**						
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**TRMSKCNT**						

Bits	Description	
[31]	**FDDGEN**	**Fault Detect Deglitch Enable Bit** 0 = Fault detect deglitch function Disable. 1 = Fault detect deglitch function Enable.
[30]	**Reserved**	Reserved.
[29:28]	**FDCKSEL**	**EPWM Channel Fault Detect Clock Select** 00 = FLT_CLK/1. 01 = FLT_CLK/2. 10 = FLT_CLK/4. 11 = FLT_CLK/8. **Note:** FLT_CLK is FDCKSn (EPWM_FDENn[16+n], n=0,1..5) selected clock.
[27:19]	**Reserved**	Reserved.
[18:16]	**DGSMPCYC**	**Deglitch Sampling Cycle** 1. FDCKS is set to 0: Sampling detect signal each EPWMx_CLK * (2^FDCKSEL) period and detect DGSMPCYC+1 times 2. FDCKS is set to 1: Sampling detect signal each EPWMx_CLK * CLKPSC * (2^FDCKSEL) period and detect DGSMPCYC+1 times **Note:**

		CLKPSC (EPWM_CLKPSCn[11:0]) is 0: TRMSKCNT >= DGSMPCYC + 2. FDCKS is 1 and CLKPSC (EPWM_CLKPSCn[11:0]) is 1: TRMSKCNT >= DGSMPCYC + 1. FDCKS is 1 and CLKPSC (EPWM_CLKPSCn[11:0]) is 2: TRMSKCNT >= DGSMPCYC.
[15]	**FDMSKEN**	**Fault Detect Mask Enable Bit** 0 = Fault detect mask function Disable. 1 = Fault detect mask function Enable.
[14:7]	**Reserved**	Reserved.
[6:0]	**TRMSKCNT**	**Transition Mask Counter** The fault detect result will be masked before counter count from 0 to TRMSKCNT. 1. FDCKS is set to 0: Mask time is EPWMx_CLK * (2^FDCKSEL) * (TRMSKCNT+2) 2. FDCKS is set to 1: Mask time EPWMx_CLK * CLKPSC * (2^FDCKSEL) * (TRMSKCNT+2) **Note:** CLKPSC (EPWM_CLKPSCn_m[11:0]) is 0: TRMSKCNT >= DGSMPCYC + 2. FDCKS is 1 and CLKPSC (EPWM_CLKPSCn[11:0]) is 1: TRMSKCNT >= DGSMPCYC + 1. FDCKS is 1 and CLKPSC (EPWM_CLKPSCn[11:0]) is 2: TRMSKCNT >= DGSMPCYC.

EPWM Fault Detect Interrupt Enable Register (EPWM_FDIEN)

Register	Offset	R/W	Description	Reset Value
EPWM_FDIEN	EPWMx_BA+0x17C	R/W	EPWM Fault Detect Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**FDIEN5**	**FDIEN4**	**FDIEN3**	**FDIEN2**	**FDIEN1**	**FDIEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**FDIENn**	**EPWM Channel n Fault Detect Interrupt Enable Bit** 0 = EPWM Channel n Fault Detect Interrupt Disable. 1 = EPWM Channel n Fault Detect Interrupt Enable.

EPWM Fault Detect Interrupt Flag Register (EPWM_FDSTS)

Register	Offset	R/W	Description	Reset Value
EPWM_FDSTS	EPWMx_BA+0x180	R/W	EPWM Fault Detect Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		FDIF5	FDIF4	FDIF3	FDIF2	FDIF1	FDIF0

Bits	Description	
[31:6]	Reserved	Reserved.
[5:0]	FDIFn	**EPWM Channel n Fault Detect Interrupt Flag Bit** Fault Detect Interrupt Flag will be set when detect EPWM output short condition. Software can clear this bit by writing 1 to it.

EPWM Trigger EADC Prescale Control Register (EPWM_EADCPSCCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCPSCCTL	EPWMx_BA+0x184	R/W	EPWM Trigger EADC Prescale Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		PSCEN5	PSCEN4	PSCEN3	PSCEN2	PSCEN1	PSCEN0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**PSCENn**	**EPWM Trigger EADC Pre-scale Function Enable Bits** 0 = EPWM Trigger EADC Pre-scale Function Disable. 1 = EPWM Trigger EADC Pre-scale Function Enable.

EPWM Trigger EADC Prescale Register 0 (EPWM_EADCPSC0)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCPSC0	EPWMx_BA+0x188	R/W	EPWM Trigger EADC Prescale Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**EADCPSC3**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**EADCPSC2**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**EADCPSC1**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**EADCPSC0**			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**EADCPSC3**	**EPWM Channel 3 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC3+1) times of EPWM_CH3 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF3.
[23:20]	**Reserved**	Reserved.
[19:16]	**EADCPSC2**	**EPWM Channel 2 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC2+1) times of EPWM_CH2 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF2.
[15:12]	**Reserved**	Reserved.
[11:8]	**EADCPSC1**	**EPWM Channel 1 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC1+1) times of EPWM_CH1 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF1.
[7:4]	**Reserved**	Reserved.
[3:0]	**EADCPSC0**	**EPWM Channel 0 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC0+1) times of EPWM_CH0 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF0.

EPWM Trigger EADC Prescale Register 1 (EPWM_EADCPSC1)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCPSC1	EPWMx_BA+0x18C	R/W	EPWM Trigger EADC Prescale Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**EADCPSC5**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**EADCPSC4**			

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**EADCPSC5**	**EPWM Channel 5 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC5+1) times of EPWM_CH5 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF5.
[7:4]	**Reserved**	Reserved.
[3:0]	**EADCPSC4**	**EPWM Channel 4 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC4+1) times of EPWM_CH4 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF4.

EPWM Trigger EADC Prescale Counter Register 0 (EPWM_EADCPSCNT0)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCPSCNT0	EPWMx_BA+0x190	R/W	EPWM Trigger EADC Prescale Counter Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**PSCNT3**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**PSCNT2**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**PSCNT1**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PSCNT0**			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**PSCNT3**	**EPWM Trigger EADC Prescale Counter 3** User can monitor PSCNT3 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN3 is 0. **Note 2:** Write data limitation: PSCNT3 < EADCPSC3.
[23:20]	**Reserved**	Reserved.
[19:16]	**PSCNT2**	**EPWM Trigger EADC Prescale Counter 2** User can monitor PSCNT2 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN2 is 0. **Note 2:** Write data limitation: PSCNT2 < EADCPSC2.
[15:12]	**Reserved**	Reserved.
[11:8]	**PSCNT1**	**EPWM Trigger EADC Prescale Counter 1** User can monitor PSCNT1 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN1 is 0. **Note 2:** Write data limitation: PSCNT1 < EADCPSC1.
[7:4]	**Reserved**	Reserved.
[3:0]	**PSCNT0**	**EPWM Trigger EADC Prescale Counter 0** User can monitor PSCNT0 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN0 is 0. **Note 2:** Write data limitation: PSCNT0 < EADCPSC0.

EPWM Trigger EADC Prescale Counter Register 1 (EPWM_EADCPSCNT1)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCPSCNT1	EPWMx_BA+0x194	R/W	EPWM Trigger EADC Prescale Counter Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**PSCNT5**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PSCNT4**			

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**PSCNT5**	**EPWM Trigger EADC Prescale Counter 5** User can monitor PSCNT5 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN5 is 0. **Note 2:** Write data limitation: PSCNT5 < EADCPSC5.
[7:4]	**Reserved**	Reserved.
[3:0]	**PSCNT4**	**EPWM Trigger EADC Prescale Counter 4** User can monitor PSCNT4 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN4 is 0. **Note 2:** Write data limitation: PSCNT4 < EADCPSC4.

EPWM Capture Input Enable Register (EPWM_CAPINEN)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPINEN	EPWMx_BA+0x200	R/W	EPWM Capture Input Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CAPINEN5**	**CAPINEN4**	**CAPINEN3**	**CAPINEN2**	**CAPINEN1**	**CAPINEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPINENn**	**Capture Input Enable Bits** 0 = EPWM Channel capture input path Disabled. The input of EPWM channel capture function is always regarded as 0. 1 = EPWM Channel capture input path Enabled. The input of EPWM channel capture function comes from correlative multifunction pin.

EPWM Capture Control Register (EPWM_CAPCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPCTL	EPWMx_BA+0x204	R/W	EPWM Capture Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		FCRLDEN5	FCRLDEN4	FCRLDEN3	FCRLDEN2	FCRLDEN1	FCRLDEN0
23	22	21	20	19	18	17	16
Reserved		RCRLDEN5	RCRLDEN4	RCRLDEN3	RCRLDEN2	RCRLDEN1	RCRLDEN0
15	14	13	12	11	10	9	8
Reserved		CAPINV5	CAPINV4	CAPINV3	CAPINV2	CAPINV1	CAPINV0
7	6	5	4	3	2	1	0
Reserved		CAPEN5	CAPEN4	CAPEN3	CAPEN2	CAPEN1	CAPEN0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**FCRLDENn**	**Falling Capture Reload Enable Bits** 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**RCRLDENn**	**Rising Capture Reload Enable Bits** 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CAPINVn**	**Capture Inverter Enable Bits** 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPENn**	**Capture Function Enable Bits** 0 = Capture function Disabled. EPWM_RCAPDATn/EPWM_FCAPDATn register will not be updated. 1 = Capture function Enabled. Capture latched the EPWM counter value when detected rising or falling edge of input signal and saved to RCAPDAT (Rising latch) and FCAPDAT (Falling latch).

EPWM Capture Status Register (EPWM_CAPSTS)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPSTS	EPWMx_BA+0x208	R	EPWM Capture Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CFLIFOV5**	**CFLIFOV4**	**CFLIFOV3**	**CFLIFOV2**	**CFLIFOV1**	**CFLIFOV0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CRLIFOV5**	**CRLIFOV4**	**CRLIFOV3**	**CRLIFOV2**	**CRLIFOV1**	**CRLIFOV0**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CFLIFOVn**	**Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIFn(EPWM_CAPIF[8+n]) is 1. **Note:** This bit will be cleared automatically when user clear corresponding CFLIFn(EPWM_CAPIF[8+n]).
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CRLIFOVn**	**Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIFn(EPWM_CAPIF[n]) is 1. **Note:** This bit will be cleared automatically when user clear corresponding CRLIFn(EPWM_CAPIF[n]).

EPWM Rising Capture Data Register 0~5 (EPWM_RCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_RCAPDAT0	EPWMx_BA+0x20C	R	EPWM Rising Capture Data Register 0	0x0000_0000
EPWM_RCAPDAT1	EPWMx_BA+0x214	R	EPWM Rising Capture Data Register 1	0x0000_0000
EPWM_RCAPDAT2	EPWMx_BA+0x21C	R	EPWM Rising Capture Data Register 2	0x0000_0000
EPWM_RCAPDAT3	EPWMx_BA+0x224	R	EPWM Rising Capture Data Register 3	0x0000_0000
EPWM_RCAPDAT4	EPWMx_BA+0x22C	R	EPWM Rising Capture Data Register 4	0x0000_0000
EPWM_RCAPDAT5	EPWMx_BA+0x234	R	EPWM Rising Capture Data Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RCAPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RCAPDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RCAPDAT**	**EPWM Rising Capture Data Register (Read Only)** When rising capture condition happened, the EPWM counter value will be saved in this register.

EPWM Falling Capture Data Register 0~5 (EPWM_FCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_FCAPDAT0	EPWMx_BA+0x210	R	EPWM Falling Capture Data Register 0	0x0000_0000
EPWM_FCAPDAT1	EPWMx_BA+0x218	R	EPWM Falling Capture Data Register 1	0x0000_0000
EPWM_FCAPDAT2	EPWMx_BA+0x220	R	EPWM Falling Capture Data Register 2	0x0000_0000
EPWM_FCAPDAT3	EPWMx_BA+0x228	R	EPWM Falling Capture Data Register 3	0x0000_0000
EPWM_FCAPDAT4	EPWMx_BA+0x230	R	EPWM Falling Capture Data Register 4	0x0000_0000
EPWM_FCAPDAT5	EPWMx_BA+0x238	R	EPWM Falling Capture Data Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FCAPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FCAPDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FCAPDAT**	**EPWM Falling Capture Data Register (Read Only)** When falling capture condition happened, the EPWM counter value will be saved in this register.

EPWM PDMA Control Register (EPWM_PDMACTL)

Register	Offset	R/W	Description	Reset Value
EPWM_PDMACTL	EPWMx_BA+0x23C	R/W	EPWM PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			CHSEL4_5	CAPORD4_5	CAPMOD4_5		CHEN4_5
15	14	13	12	11	10	9	8
Reserved			CHSEL2_3	CAPORD2_3	CAPMOD2_3		CHEN2_3
7	6	5	4	3	2	1	0
Reserved			CHSEL0_1	CAPORD0_1	CAPMOD0_1		CHEN0_1

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**CHSEL4_5**	**Select Channel 4/5 to Do PDMA Transfer** 0 = Channel4. 1 = Channel5.
[19]	**CAPORD4_5**	**Capture Channel 4/5 Rising/Falling Order** Set this bit to determine whether the EPWM_RCAPDAT4/5 or EPWM_FCAPDAT4/5 is the first captured data transferred to memory through PDMA when CAPMOD4_5 =11. 0 = EPWM_FCAPDAT4/5 is the first captured data to memory. 1 = EPWM_RCAPDAT4/5 is the first captured data to memory.
[18:17]	**CAPMOD4_5**	**Select EPWM_RCAPDAT4/5 or EPWM_FCAPDAT4/5 to Do PDMA Transfer** 00 = Reserved. 01 = EPWM_RCAPDAT4/5. 10 = EPWM_FCAPDAT4/5. 11 = Both EPWM_RCAPDAT4/5 and EPWM_FCAPDAT4/5.
[16]	**CHEN4_5**	**Channel 4/5 PDMA Enable Bit** 0 = Channel 4/5 PDMA function Disabled. 1 = Channel 4/5 PDMA function Enabled for the channel 4/5 captured data and transfer to memory.
[15:13]	**Reserved**	Reserved.
[12]	**CHSEL2_3**	**Select Channel 2/3 to Do PDMA Transfer** 0 = Channel2. 1 = Channel3.
[11]	**CAPORD2_3**	**Capture Channel 2/3 Rising/Falling Order** Set this bit to determine whether the EPWM_RCAPDAT2/3 or EPWM_FCAPDAT2/3 is the first captured data transferred to memory through PDMA when CAPMOD2_3 =11. 0 = EPWM_FCAPDAT2/3 is the first captured data to memory.

		1 = EPWM_RCAPDAT2/3 is the first captured data to memory.
[10:9]	**CAPMOD2_3**	**Select EPWM_RCAPDAT2/3 or EPWM_FCAODAT2/3 to Do PDMA Transfer** 00 = Reserved. 01 = EPWM_RCAPDAT2/3. 10 = EPWM_FCAPDAT2/3. 11 = Both EPWM_RCAPDAT2/3 and EPWM_FCAPDAT2/3.
[8]	**CHEN2_3**	**Channel 2/3 PDMA Enable Bit** 0 = Channel 2/3 PDMA function Disabled. 1 = Channel 2/3 PDMA function Enabled for the channel 2/3 captured data and transfer to memory.
[7:5]	**Reserved**	Reserved.
[4]	**CHSEL0_1**	**Select Channel 0/1 to Do PDMA Transfer** 0 = Channel0. 1 = Channel1.
[3]	**CAPORD0_1**	**Capture Channel 0/1 Rising/Falling Order** Set this bit to determine whether the EPWM_RCAPDAT0/1 or EPWM_FCAPDAT0/1 is the first captured data transferred to memory through PDMA when CAPMOD0_1 =11. 0 = EPWM_FCAPDAT0/1 is the first captured data to memory. 1 = EPWM_RCAPDAT0/1 is the first captured data to memory.
[2:1]	**CAPMOD0_1**	**Select EPWM_RCAPDAT0/1 or EPWM_FCAPDAT0/1 to Do PDMA Transfer** 00 = Reserved. 01 = EPWM_RCAPDAT0/1. 10 = EPWM_FCAPDAT0/1. 11 = Both EPWM_RCAPDAT0/1 and EPWM_FCAPDAT0/1.
[0]	**CHEN0_1**	**Channel 0/1 PDMA Enable Bit** 0 = Channel 0/1 PDMA function Disabled. 1 = Channel 0/1 PDMA function Enabled for the channel 0/1 captured data and transfer to memory.

EPWM Capture Channel 0_1, 2_3, 4_5 PDMA Register (EPWM_PDMACAP 0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_PDMACAP0_1	EPWMx_BA+0x240	R	EPWM Capture Channel 01 PDMA Register	0x0000_0000
EPWM_PDMACAP2_3	EPWMx_BA+0x244	R	EPWM Capture Channel 23 PDMA Register	0x0000_0000
EPWM_PDMACAP4_5	EPWMx_BA+0x248	R	EPWM Capture Channel 45 PDMA Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CAPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CAPBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CAPBUF**	**EPWM Capture PDMA Register (Read Only)** This register is used as a buffer to transfer EPWM capture rising or falling data to memory by PDMA.

EPWM Capture Interrupt Enable Register (EPWM_CAPIEN)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPIEN	EPWMx_BA+0x250	R/W	EPWM Capture Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CAPFIEN5	CAPFIEN4	CAPFIEN3	CAPFIEN2	CAPFIEN1	CAPFIEN0
7	6	5	4	3	2	1	0
Reserved		CAPRIEN5	CAPRIEN4	CAPRIEN3	CAPRIEN2	CAPRIEN1	CAPRIEN0

Bits	Description	
[31:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CAPFIENn**	**EPWM Capture Falling Latch Interrupt Enable Bits** 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPRIENn**	**EPWM Capture Rising Latch Interrupt Enable Bits** 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled.

EPWM Capture Interrupt Flag Register (EPWM_CAPIF)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPIF	EPWMx_BA+0x254	R/W	EPWM Capture Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CFLIF5**	**CFLIF4**	**CFLIF3**	**CFLIF2**	**CFLIF1**	**CFLIF0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CRLIF5**	**CRLIF4**	**CRLIF3**	**CRLIF2**	**CRLIF1**	**CRLIF0**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CFLIFn**	**EPWM Capture Falling Latch Interrupt Flag** 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high. **Note 1:** When Capture with PDMA operating, EPWM_CAPIF corresponding channel CFLIFn will be cleared by hardware after PDMA transfer data. **Note 2:** This bit is cleared by writing 1 to it.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CRLIFn**	**EPWM Capture Rising Latch Interrupt Flag** 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high. **Note 1:** When Capture with PDMA operating, EPWM_CAPIF corresponding channel CRLIFn will be cleared by hardware after PDMA transfer data. **Note 2:** This bit is cleared by writing 1 to it.

EPWM Capture Input Noise Filter Register (EPWM_CAPNF0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPNF0	EPWMx_BA+0x258	R/W	EPWM Capture Input Noise Filter Register 0	0x0000_0000
EPWM_CAPNF1	EPWMx_BA+0x25C	R/W	EPWM Capture Input Noise Filter Register 1	0x0000_0000
EPWM_CAPNF2	EPWMx_BA+0x260	R/W	EPWM Capture Input Noise Filter Register 2	0x0000_0000
EPWM_CAPNF3	EPWMx_BA+0x264	R/W	EPWM Capture Input Noise Filter Register 3	0x0000_0000
EPWM_CAPNF4	EPWMx_BA+0x268	R/W	EPWM Capture Input Noise Filter Register 4	0x0000_0000
EPWM_CAPNF5	EPWMx_BA+0x26C	R/W	EPWM Capture Input Noise Filter Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					CAPNFCNT		
7	6	5	4	3	2	1	0
Reserved	CAPNFSEL			Reserved			CAPNFEN

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**CAPNFCNT**	**Capture Edge Detector Noise Filter Count** The register bits control the capture filter counter to count from 0 to CAPNFCNT.
[7]	**Reserved**	Reserved.
[6:4]	**CAPNFSEL**	**Capture Edge Detector Noise Filter Clock Selection** 000 = Filter clock = PCLK. 001 = Filter clock = PCLK/2. 010 = Filter clock = PCLK/4. 011 = Filter clock = PCLK/8. 100 = Filter clock = PCLK/16. 101 = Filter clock = PCLK/32. 110 = Filter clock = PCLK/64. 111 = Filter clock = PCLK/128.
[3:1]	**Reserved**	Reserved.
[0]	**CAPNFEN**	**Capture Noise Filter Enable Bits** 0 = Capture Noise Filter function Disabled. 1 = Capture Noise Filter function Enabled.

EPWM External Event Trigger Control Register 0~5 (EPWMx_EXTETCTL0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_EXTETCTL0	EPWMx_BA+0x270	R/W	EPWM External Event Trigger Control Register 0	0x0000_0000
EPWM_EXTETCTL1	EPWMx_BA+0x274	R/W	EPWM External Event Trigger Control Register 1	0x0000_0000
EPWM_EXTETCTL2	EPWMx_BA+0x278	R/W	EPWM External Event Trigger Control Register 2	0x0000_0000
EPWM_EXTETCTL3	EPWMx_BA+0x27C	R/W	EPWM External Event Trigger Control Register 3	0x0000_0000
EPWM_EXTETCTL4	EPWMx_BA+0x280	R/W	EPWM External Event Trigger Control Register 4	0x0000_0000
EPWM_EXTETCTL5	EPWMx_BA+0x284	R/W	EPWM External Event Trigger Control Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				EXTTRGS			
7	6	5	4	3	2	1	0
Reserved		CNTACTS		Reserved			EXTETEN

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**EXTTRGS**	**External Trigger Selection** 0000 = INT0. 0001 = INT1. 0010 = INT2. 0011 = INT3. 0100 = INT4. 0101 = INT5. 0110 = INT6. 0111 = INT7. Other = Resrved.
[7:6]	**Reserved**	Reserved.
[5:4]	**CNTACTS**	**Counter Action Selection** 00 = Counter reset. 01 = Counter start. 10 = Counter reset and start. 11 = Reseved.
[3:1]	**Reserved**	Reserved.
[0]	**EXTETEN**	**External Event Trigger Enable Bit**

		0 = External Event Trigger function Disabled. 1 = External Event Trigger function Enabled.

EPWM Software Event Output Force Control Register (EPWMx_SWEOFCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_SWEOFCTL	EPWMx_BA+0x288	R/W	EPWM Software Event Output Force Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**OUTACTS5**		**OUTACTS4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OUTACTS3		**OUTACTS2**		**OUTACTS1**		**OUTACTS0**	

Bits	Description	
[31:12]	**Reserved**	Reserved.
[2n+1:2n] n=0,1..5	**OUTACTSn**	**Output Action Selection** 00 = Do nothing. 01 = EPWM output Low. 10 = EPWM output High. 11 = EPWM output Toggle.

EPWM Software Event Output Force Trigger Register (EPWMx_SWEOFTRG)

Register	Offset	R/W	Description	Reset Value
EPWM_SWEOFTRG	EPWMx_BA+0x28C	R/W	EPWM Software Event Output Force Trigger Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**SWETRG5**	**SWETRG4**	**SWETRG3**	**SWETRG2**	**SWETRG1**	**SWETRG0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**SWETRGn**	**Software Event Trigger** Write 1 to this bit will change EPWM output status according to OUTACTSn in EPWMx_SWEOFCTL setting. **Note:** This bit will auto cleared by hardware.

EPWM Clock Prescale Register 0~5 (EPWM_CLKPSC0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CLKPSC0	EPWMx_BA+0x290	R/W	EPWM Clock Prescale Register 0	0x0000_0000
EPWM_CLKPSC1	EPWMx_BA+0x294	R/W	EPWM Clock Prescale Register 1	0x0000_0000
EPWM_CLKPSC2	EPWMx_BA+0x298	R/W	EPWM Clock Prescale Register 2	0x0000_0000
EPWM_CLKPSC3	EPWMx_BA+0x29C	R/W	EPWM Clock Prescale Register 3	0x0000_0000
EPWM_CLKPSC4	EPWMx_BA+0x2A0	R/W	EPWM Clock Prescale Register 4	0x0000_0000
EPWM_CLKPSC5	EPWMx_BA+0x2A4	R/W	EPWM Clock Prescale Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CLKPSC**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**CLKPSC**	**EPWM Counter Clock Prescale** The clock of EPWM counter is decided by clock prescaler. Each EPWM pair share one EPWM counter clock prescaler. The clock of EPWM counter is divided by (CLKPSC+ 1).

EPWM Rising Dead-time Counter Register 0_1, 2_3, 4_5 (EPWM_RDTCNT0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_RDTCNT0_1	EPWMx_BA+0x2A8	R/W	EPWM Rising Dead-time Counter Register 0/1	0x0000_0000
EPWM_RDTCNT2_3	EPWMx_BA+0x2AC	R/W	EPWM Rising Dead-time Counter Register 2/3	0x0000_0000
EPWM_RDTCNT4_5	EPWMx_BA+0x2B0	R/W	EPWM Rising Dead-time Counter Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**RDTCNT**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RDTCNT							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**RDTCNT**	**Rising Dead-time Counter (Write Protect)** The Rising dead-time can be calculated from the following formula: RDTCKSEL=0: Rising Dead-time = (RDTCNT[11:0]+1) * EPWM_CLK period. RDTCKSEL=1: Rising Dead-time = (RDTCNT[11:0]+1) * EPWM_CLK period * (CLKPSC+1). **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Falling Dead-time Counter Register 0_1, 2_3, 4_5 (EPWM_FDTCNT0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_FDTCNT0_1	EPWMx_BA+0x2B4	R/W	EPWM Falling Dead-time Counter Register 0/1	0x0000_0000
EPWM_FDTCNT2_3	EPWMx_BA+0x2B8	R/W	EPWM Falling Dead-time Counter Register 2/3	0x0000_0000
EPWM_FDTCNT4_5	EPWMx_BA+0x2BC	R/W	EPWM Falling Dead-time Counter Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				FDTCNT			
7	6	5	4	3	2	1	0
FDTCNT							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**FDTCNT**	**Falling Dead-time Counter (Write Protect)** The dead-time can be calculated from the following formula: FDTCKSEL=0: Falling Dead-time = (FDTCNT[11:0]+1) * EPWM_CLK period. FDTCKSEL=1: Falling Dead-time = (FDTCNT[11:0]+1) * EPWM_CLK period * (CLKPSC+1). **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Dead-time Control Register (EPWM_DTCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_DTCTL	EPWMx_BA+0x2C0	R/W	EPWM Dead-time Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					DTCKSEL4	DTCKSEL2	DTCKSEL0
15	14	13	12	11	10	9	8
Reserved					FDTEN4	FDTEN2	FDTEN0
7	6	5	4	3	2	1	0
Reserved					RDTEN4	RDTEN2	RDTEN0

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16+n/2] n=0,2,4	**DTCKSELn**	**Dead-time Clock Select for EPWM Pair (Write Protect)** 0 = Dead-time clock source from EPWM_CLK. 1 = Dead-time clock source from prescaler output. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[15:11]	**Reserved**	Reserved.
[8+n/2] n=0,2,4	**FDTENn**	**Enable Falling Dead-time Insertion for EPWM Pair (Write Protect)** Falling Dead-time insertion is only active when this pair of complementary EPWM is enabled. If falling dead- time insertion is inactive, the outputs of pin pair are complementary without any delay. 0 = Falling Dead-time insertion Disabled on the pin pair. 1 = Falling Dead-time insertion Enabled on the pin pair. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**RDTENn**	**Enable Rising Dead-time Insertion for EPWM Pair (Write Protect)** Rising Dead-time insertion is only active when this pair of complementary EPWM is enabled. If rising dead- time insertion is inactive, the outputs of pin pair are complementary without any delay. 0 = Rising Dead-time insertion Disabled on the pin pair. 1 = Rising Dead-time insertion Enabled on the pin pair. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Period Register Buffer 0~5 (EPWM_PBUF0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_PBUF0	EPWMx_BA+0x304	R	EPWM PERIOD0 Buffer	0x0000_0000
EPWM_PBUF1	EPWMx_BA+0x308	R	EPWM PERIOD1 Buffer	0x0000_0000
EPWM_PBUF2	EPWMx_BA+0x30C	R	EPWM PERIOD2 Buffer	0x0000_0000
EPWM_PBUF3	EPWMx_BA+0x310	R	EPWM PERIOD3 Buffer	0x0000_0000
EPWM_PBUF4	EPWMx_BA+0x314	R	EPWM PERIOD4 Buffer	0x0000_0000
EPWM_PBUF5	EPWMx_BA+0x318	R	EPWM PERIOD5 Buffer	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PBUF**	**EPWM Period Register Buffer (Read Only)** Used as PERIOD active register.

EPWM Comparator Register Buffer0~5 (EPWM_CMPBUF0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CMPBUF0	EPWMx_BA+0x31C	R	EPWM CMPDAT0 Buffer	0x0000_0000
EPWM_CMPBUF1	EPWMx_BA+0x320	R	EPWM CMPDAT1 Buffer	0x0000_0000
EPWM_CMPBUF2	EPWMx_BA+0x324	R	EPWM CMPDAT2 Buffer	0x0000_0000
EPWM_CMPBUF3	EPWMx_BA+0x328	R	EPWM CMPDAT3 Buffer	0x0000_0000
EPWM_CMPBUF4	EPWMx_BA+0x32C	R	EPWM CMPDAT4 Buffer	0x0000_0000
EPWM_CMPBUF5	EPWMx_BA+0x330	R	EPWM CMPDAT5 Buffer	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPBUF**	**EPWM Comparator Register Buffer (Read Only)** Used as CMP active register.

EPWM_FTCMPDAT Buffer (EPWM_FTCBUF0_1,2_3,4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_FTCBUF0_1	EPWMx_BA+0x340	R	EPWM FTCMPDAT0_1 Buffer	0x0000_0000
EPWM_FTCBUF2_3	EPWMx_BA+0x344	R	EPWM FTCMPDAT2_3 Buffer	0x0000_0000
EPWM_FTCBUF4_5	EPWMx_BA+0x348	R	EPWM FTCMPDAT4_5 Buffer	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FTCMPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FTCMPBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FTCMPBUF**	**EPWM FTCMPDAT Buffer (Read Only)** Used as FTCMP active buffer.

EPWM FTCMPDAT Indicator Register (EPWM_FTCI)

Register	Offset	R/W	Description	Reset Value
EPWM_FTCI	EPWMx_BA+0x34C	R/W	EPWM FTCMPDAT Indicator Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**FTCMD4**	**FTCMD2**	**FTCMD0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**FTCMU4**	**FTCMU2**	**FTCMU0**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[8+n/2] n=0,2,4	**FTCMDn**	**EPWM FTCMPDAT Down Indicator** Indicator is set by hardware when EPWM counter down count and reaches EPWM_FTCMPDATn, software can clear this bit by writing 1 to it.
[7:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**FTCMUn**	**EPWM FTCMPDAT Up Indicator** Indicator is set by hardware when EPWM counter up count and reaches EPWM_FTCMPDATn, software can clear this bit by writing 1 to it.

EPWM CLKPSC Buffer 0~5 (EPWM_CPSCBUF0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CPSCBUF0	EPWMx_BA+0x350	R	EPWM CLKPSC0 Buffer	0x0000_0000
EPWM_CPSCBUF1	EPWMx_BA+0x354	R	EPWM CLKPSC1 Buffer	0x0000_0000
EPWM_CPSCBUF2	EPWMx_BA+0x358	R	EPWM CLKPSC2 Buffer	0x0000_0000
EPWM_CPSCBUF3	EPWMx_BA+0x35C	R	EPWM CLKPSC3 Buffer	0x0000_0000
EPWM_CPSCBUF4	EPWMx_BA+0x360	R	EPWM CLKPSC4 Buffer	0x0000_0000
EPWM_CPSCBUF5	EPWMx_BA+0x364	R	EPWM CLKPSC5 Buffer	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CPSCBUF**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CPSCBUF							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**CPSCBUF**	**EPWM Counter Clock Prescale Buffer** Used as EPWM counter clock pre-scare active register.

EPWM Interrupt Flag Accumulator Counter0~5 (EPWM_IFACNT0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_IFACNT0	EPWMx_BA+0x368	R	EPWM Interrupt Flag Accumulator Counter 0	0x0000_0000
EPWM_IFACNT1	EPWMx_BA+0x36C	R	EPWM Interrupt Flag Accumulator Counter 1	0x0000_0000
EPWM_IFACNT2	EPWMx_BA+0x370	R	EPWM Interrupt Flag Accumulator Counter 2	0x0000_0000
EPWM_IFACNT3	EPWMx_BA+0x374	R	EPWM Interrupt Flag Accumulator Counter 3	0x0000_0000
EPWM_IFACNT4	EPWMx_BA+0x378	R	EPWM Interrupt Flag Accumulator Counter 4	0x0000_0000
EPWM_IFACNT5	EPWMx_BA+0x37C	R	EPWM Interrupt Flag Accumulator Counter 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACUCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACUCNT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**ACUCNT**	**Accumulator Counter (Read Only)** This value indicates how many interrupt are accumulated when using interrupt flag accumulator function.

6.16 Enhanced Quadrature Encoder Interface (EQEI)

6.16.1 Overview

There are two EQEI controllers in this device. The Enhanced Quadrature Encoder Interface (EQEI) decodes speed of rotation and motion sensor information. It can be used in any application that uses a quadrature encoder for feedback.

6.16.2 Features

6.16.2.1 Enhanced Quadrature Encoder Interface (EQEI) Features

- Up to two EQEI controllers, EQEI0 and EQEI1.
- Two EQEI phase inputs, QEA and QEB; One Index input.
- A 32-bit up/down Quadrature Encoder Pulse Counter (EQEI_CNT)
- A 32-bit software-latch Quadrature Encoder Pulse Counter Hold Register (EQEI_CNTHOLD)
- A 32-bit Quadrature Encoder Pulse Counter Index Latch Register (EQEI_CNTLATCH)
- A 32-bit Quadrature Encoder Pulse Counter Compare Register (EQEI_CNTCMP) with a Pre-set Maximum Count Register (EQEI_CNTMAX)
- A 32-bit up count Unit Timer Pulse Counter (EQEI_UTCNT)
- A 32-bit Unit Timer Pulse Counter Compare Register (EQEI_UTCMP)
- Two EQEI control registers (EQEI_CTL and QEI_CTL2) and one EQEI Status Register (EQEI_STATUS)
- Four Quadrature encoder pulse counter operation modes:
  - Supports X4 free-counting mode
  - Supports X2 free-counting mode
  - Supports X4 compare-counting mode
  - Supports X2 compare-counting mode
- Two Quadrature encoder phase counter operation modes:

  - Supports X1 1-phase 2 input compare-counting mode

  - Supports X2 1-phase 2 input compare-counting mode
- Two Quadrature encoder directional counter operation modes:

  - Supports X1 1-phase 1 input compare-counting mode

  - Supports X2 1-phase 1 input compare-counting mode
- Supports swap function for input signals QEA and QEB
- Supports for detecting the occurrence of phase error from input signals QEA and QEB
- Supports one times index signal reset function for Quadrature encoder pulse counter
- Encoder Pulse Width measurement mode by ECAP
- Input frequency of QEA/QEB/IDX without noise filter must lower than PCLK/4
- Input frequency of QEA/QEB/IDX with noise filter must lower than Noise Filter Clock/8

6.16.3 Block Diagram

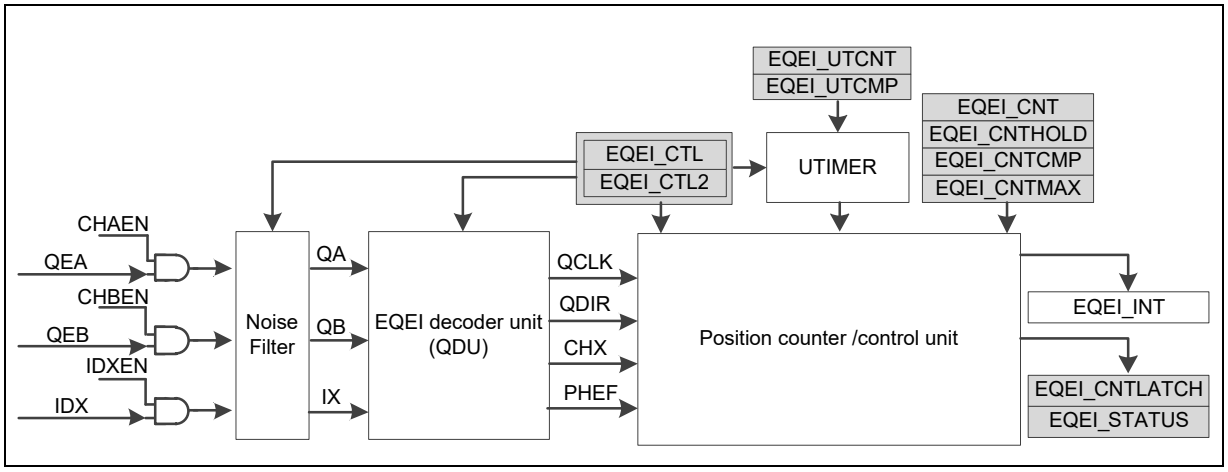


Figure 6.16-1 EQEI Block Diagram

The EQEI controller inputs, QEA and QEB, accept the outputs from a quadrature encoded source, such as incremental optical shaft encoder. Two channels, A and B, nominally 90 degrees out of phase, are required. A quadrature encoder usually provides an index signal (to pin IDX) which can be used to indicate an absolute position. There is a noise filter and polarity control for each signal before EQEI control unit.

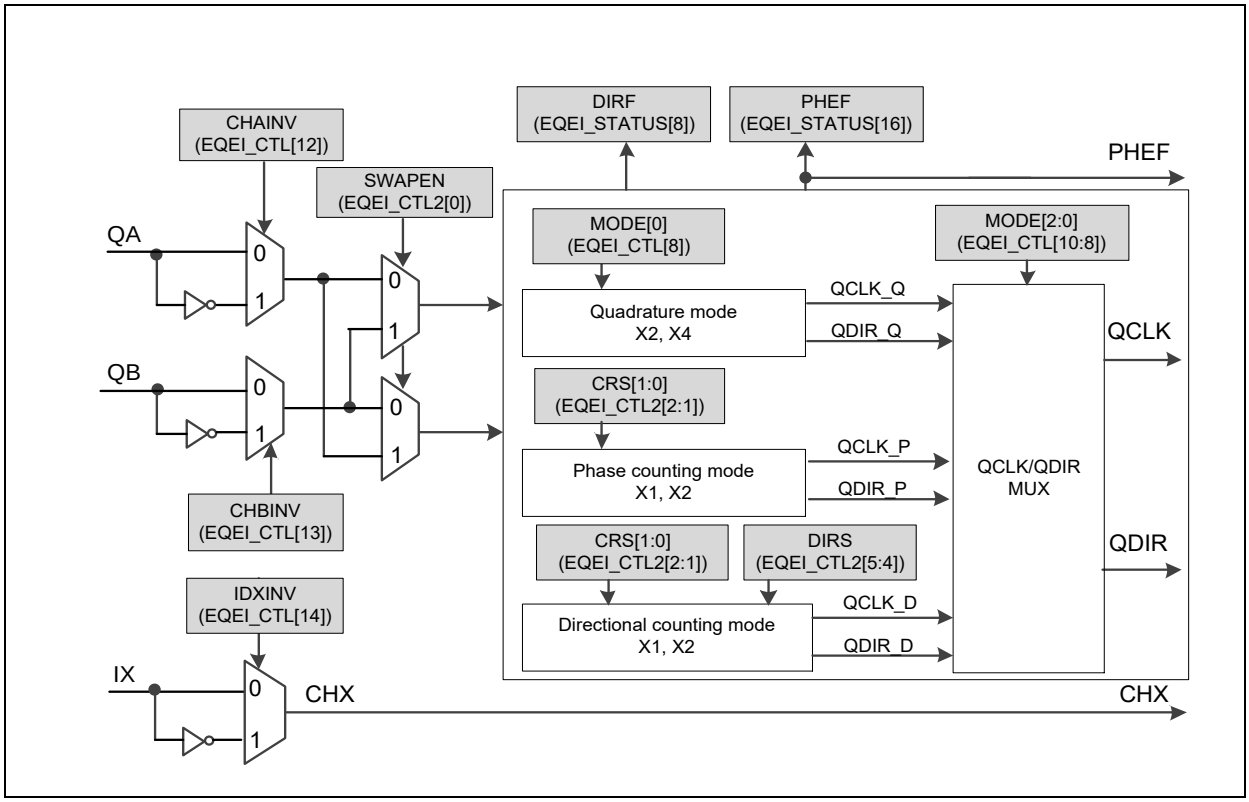


Figure 6.16-2 EQEI Decoder Unit

EQEI Decoder Unit (QDU) module contains inverse and swap function for input signals QA and QB, EQEI mode selection, EQEI phase counter clock rate selection, and the mechanism for detecting phase error. Users can use the function they want by controlling the registers.

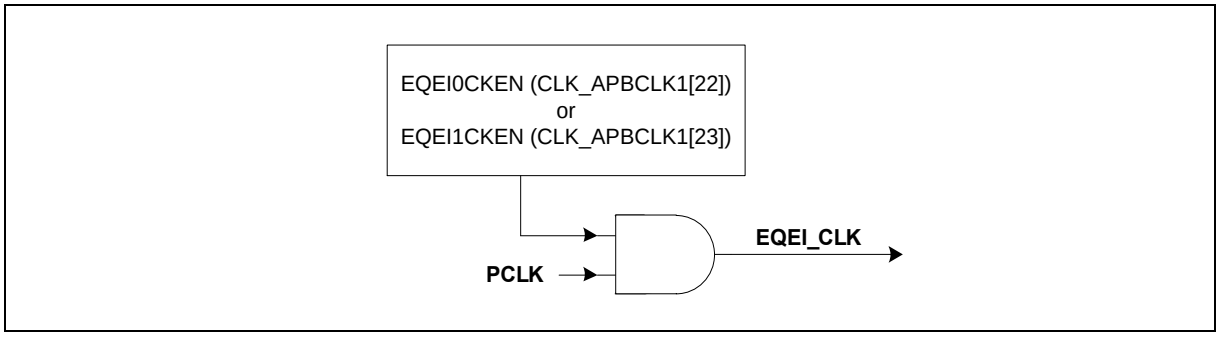


Figure 6.16-3 EQEI Clock Source Control

6.16.4 Basic Configuration

6.16.4.1 EQEI0 Basic Configuration

- Clock Source Configuration
  - Enable EQEI0 peripheral clock in EQEI0CKEN (CLK_APBCLK1[22]).
- Reset Configuration
  - Reset EQEI0 controller in EQEI0RST (SYS_IPRST2[22]).

6.16.4.2 EQEI1 Basic Configuration

- Clock Source Configuration
  - Enable EQEI1 peripheral clock in EQEI1CKEN (CLK_APBCLK1[23]).
- Reset Configuration
  - Reset EQEI1 controller in EQEI1RST (SYS_IPRST2[23]).

6.16.5 Functional Description

The EQEI control logic detects the relation of phase lead/lag between the filtered signals CHA and CHB and CHX to produce direction indication bit DIRF(EQEI_STATUS[8]) to control the pulse counter. The comparator/reload logic compares the pulse counter and maximum count and control the function of reloading pulse counter in compare-counting mode. In Free-counting mode the pulse counter CNT(EQEI_CNT[31:0]) will count until the 0xFFFF_FFFF value; while in Compare-counting mode the pulse counter will counts until the CNTMAX (EQEI_CNTMAX[31:0]) and the pulse counter will be reset to 0 to restart the next cyclic counting.

6.16.5.1 Input Noise Filter

Each pin of EQEI inputs is equipped with a noise filter which can filter the unwanted noise form. The QEA, QEB and IDX noise filters can be disabled through bits NFDIS (EQEI_CTL[3]). If enabled, the capture logic is required to sample 4 consecutive same capture input value in order to recognize an edge as a capture event. In Figure 6.16-6, it is a possible implementation of digital noise filter; the interval between pulses requirement for input capture is 4 EQEI_CLK clocks width. Any pulse width less than or equal to 3 EQEI_CLK clocks will not have any trigger. The timing requirement through Noise Filter is also shown in Figure 6.16-6. CHA, CHB and CHX are the outputs of QEA, QEB and IDX respectively after going through noise filter and polarity control. Refer to Figure 6.16-4. If the noise filter is disabled the input signals QEA, QEB and IDX are passed to the internal signals CHA, CHB and CHX respectively without any delay.

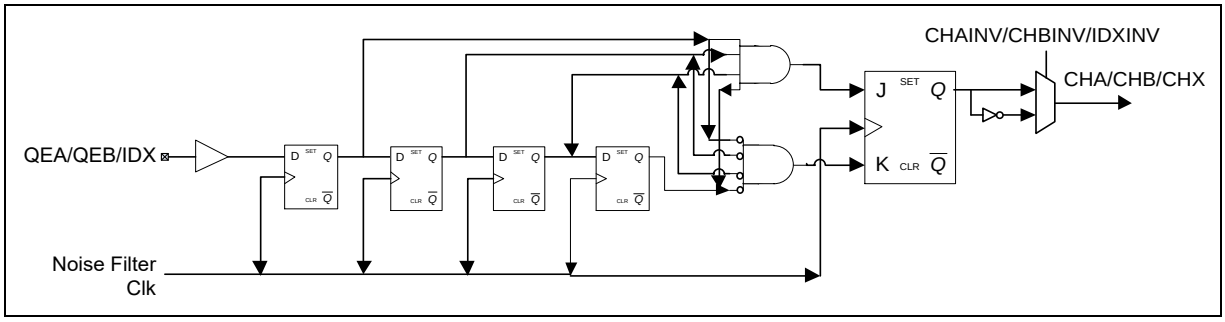


Figure 6.16-4 Noise Filter

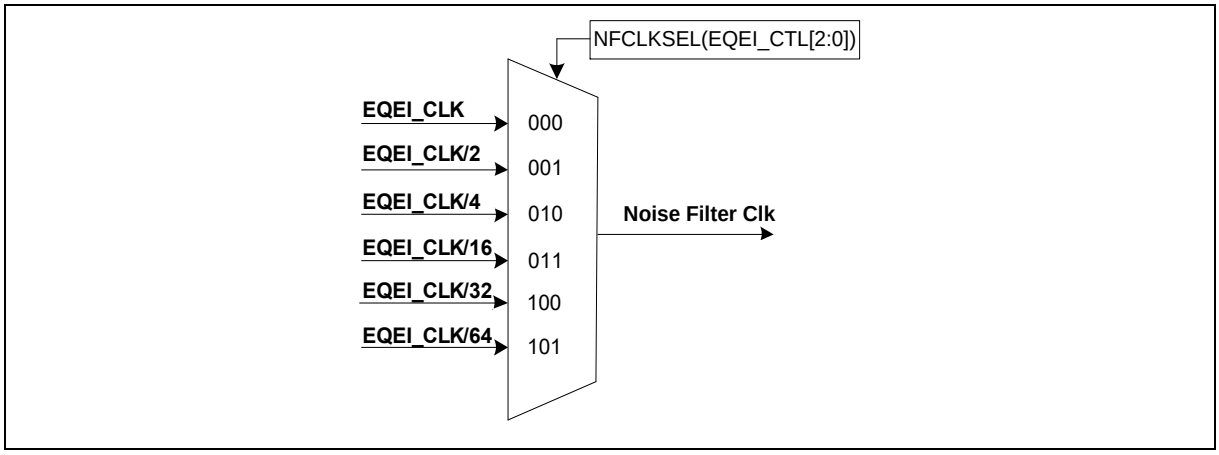


Figure 6.16-5 Noise Filter Sampling Clock Selection

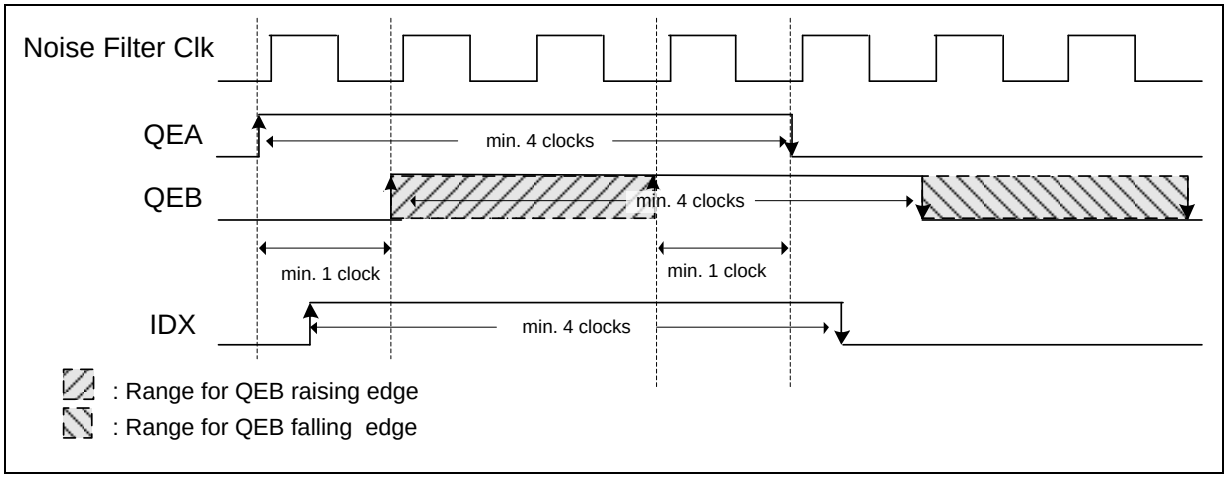


Figure 6.16-6 QEA/QEB/IDX Timing Requirement Through Noise Filter

6.16.5.2 Operation of Quadrature Encoder Interface

There are seven Quadrature encoder pulse counter operation modes:

- Mode0: X4 Quadrature free-counting mode
- Mode1: X2 Quadrature free-counting mode
- Mode2: X4 Quadrature compare-counting mode
- Mode3: X2 Quadrature compare-counting mode

- Mode4: 1-phase 2 input phase counting mode type 1
- Mode5: 1-phase 2 input phase counting mode type 2
- Mode6: directional counting mode.

6.16.5.3 Free-counting Mode

The quadrature encoder pulse counter CNT(EQEI_CNT[31:0]) up or down counts according to the direction indication bit DIRF (EQEI_STATUS[8]). When overflow or underflow occurs, it sets flag OVUNF (EQEI_STATUS[2]). Refer to Figure 6.16-7 and Figure 6.16-8 for detailed timing.

6.16.5.4 Compare-counting Mode

The pulse counter up or down counts according to the direction indication bit DIRF (EQEI_STATUS[8]). On up counting, flag OVUNF (EQEI_STATUS[2]) will be asserted when CNT(EQEI_CNT[31:0]) overflows from CNTMAX (EQEI_CNTMAX[31:0]) to 0 on the next CHA edge for X2 counting mode, and on CHA/CHB edge for X4 counting mode. On down counting, flag OVUNF (EQEI_STATUS[2]) will be asserted when CNT(EQEI_CNT[31:0]) underflows from 0 to CNTMAX (EQEI_CNTMAX[31:0]) on the next CHA edge for X2 counting mode, and on CHA/CHB edge for X4 counting mode. This mode provides the position of a rotor to user. If a quadrature encoder output 1024 pulses to CHA per round, user can write EQEI_MAXCNT and CNTCMP(EQEI_CNTCMP[31:0]) with 4095 in X4 mode or 2047 in x2 mode and reset CNT(EQEI_CNT[31:0]) at initial before compare-counting mode is active. When the CNT(EQEI_CNT[31:0]) overflows from CNTCMP(EQEI_CNTCMP[31:0]), here CNTCMP(EQEI_CNTCMP[31:0]) should be preset the same value as CNTMAX (EQEI_CNTMAX[31:0]), it means rotor runs one round on next CHA/CHB edge. Refer to Figure 6.16-7 and Figure 6.16-8 for detailed timing.

6.16.5.5 X4/X2 Counting Modes

In **X4 Counting mode**, the pulse counter increases or decreases one on every CHA and CHB edge based on the phase relationship of CHA and CHB signals.

EQEI X4 Counting mode provides a finer resolution of the rotor position, since the counter increments or decrements more frequently for each QEA/QEB input pulse pair than in EQEI x2 mode. This mode is selected by setting the EQEI Counting Mode Selection bits MODE(EQEI_CTL[10:8]) to 000 or 010. In this mode, the EQEI logic detects every edge on every QEA and QEB input edges.

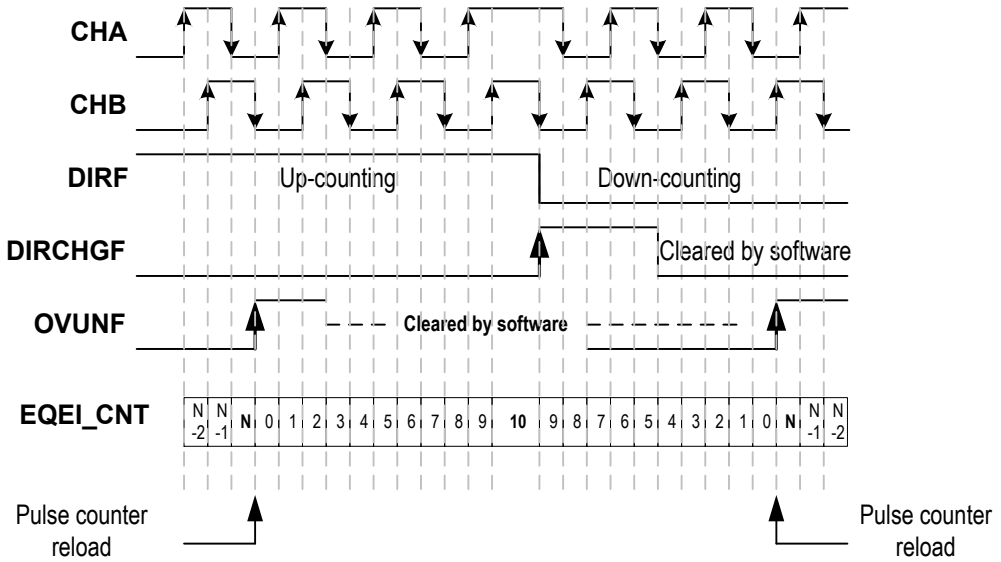
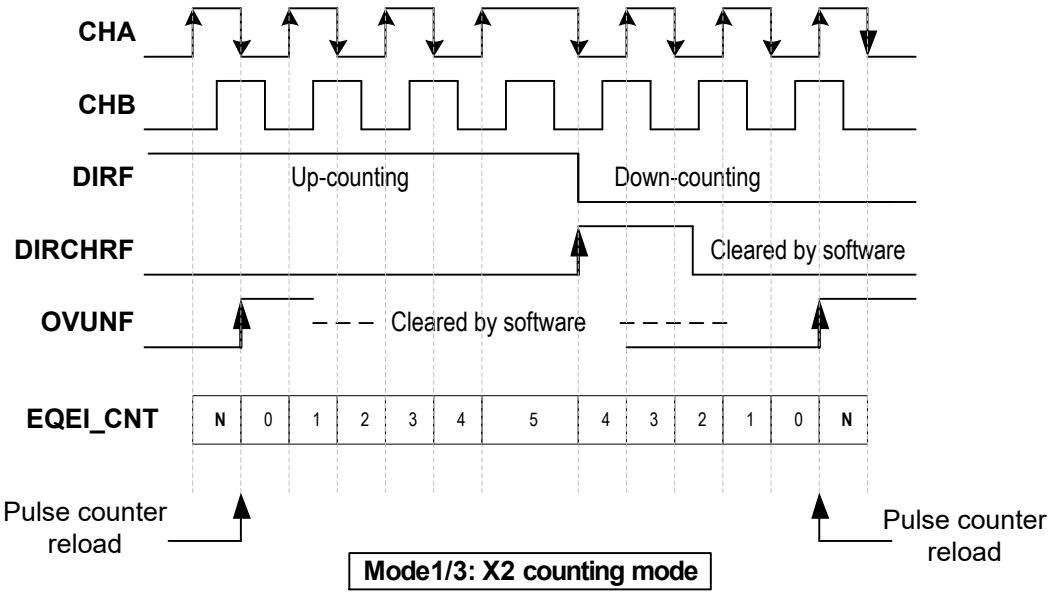


Figure 6.16-7 X4 Counting Mode

In **X2 Counting mode**, the pulse counter increases or decreases one on every CHA edge based on the phase relationship of CHA and CHB signals.

EQEI X2 Counting mode is selected by setting the EQEI Counting Mode Selection bits (EQEI_CTL[10:8]) to 001 or 011. In this mode, the EQEI logic detects every edge on the QEA input only. Every rising and falling edge on the QEA signal clocks the pulse counter.



Note:
Mode1: X2 free-counting mode, N=0xFFFF_FFFF
Mode3: X2 compare-counting mode, N=CNTMAX
CHA & CHB are output signals of digital filter

Figure 6.16-8 X2 Counting Mode

6.16.5.6 Direction of Counting

If CHA leads CHB, the pulse counter is increased by 1. If CHA lags CHB, the pulse counter is decreased by 1. The EQEI control logic generates a signal that sets the DIRF (EQEI_STATUS[8]); this in turn determines the direction of the count. When CHA leads CHB, DIRF(EQEI_STATUS[8]) is set as 1, and the position counter increments on every active edge. When CHA lags CHB, DIRF(EQEI_STATUS[8]) is cleared, and the position counter decrements on every active edge. Refer to Table 6.16-1.

Current Signal Detected	Previous Signal Detected				DIR (Counting Direction)
	Rising		Falling		
	CHA	CHB	CHA	CHB	
CHA rising				√	1 (Increment)
		√			0 (Decrement)
			√		Toggle (direction change)
CHA falling				√	0 (Decrement)
		√			1 (Increment)
	√				Toggle (direction change)
CHB rising	√				1 (Increment)
			√		0 (Decrement)
				√	Toggle (direction change)
CHB falling			√		1 (Increment)

	√				0 (Decrement)
		√			Toggle (direction change)

Table 6.16-1 Direction of Counting

6.16.5.7 Up-Counting

Under the forward direction the DIRF(EQEI_STATUS[8]) bit is 1 when up-counting. Software needs to clear the OVUNF (EQEI_STATUS[2]) flag. For the free-counting mode the CNT(EQEI_CNT[31:0]) counter will count until it matches 0xFFFF_FFFF and next edges on the forward direction will set the bit OVUNF (EQEI_STATUS[2]) high and reset CNT(EQEI_CNT[31:0]) to 0. For compare-counting mode the CNT(EQEI_CNT[31:0]) counter counts until the CNTMAX (EQEI_CNTMAX[31:0]) value and next edges on the forward direction will set the bit OVUNF (EQEI_STATUS[2]) high and reset CNT(EQEI_CNT[31:0]) to 0. Changes of direction trigger a down-count and CNT(EQEI_CNT[31:0]) decreasing in counter value. For X2 mode, only CHA edge will set OVUNF (EQEI_STATUS[2]) while for X4 mode both CHA and CHB edges will set OVUNF (EQEI_STATUS[2]).

6.16.5.8 Down-Counting

A change of direction will cause the counter to down count for X2/X4 counting mode. It is indicated with the DIRF(EQEI_STATUS[8]) bit as 0 and DIRCHGF (EQEI_STATUS[3]) flag is set to 1. At this stage the CNT(EQEI_CNT[31:0]) will start to down-count. In free-counting mode the pulse counter will reload with 0xFFFF_FFFF when it down counts to 0 and sets OVUNF (EQEI_STATUS[2]) to high in the next edge. The pulse counter will reload with CNTMAX (EQEI_CNTMAX[31:0]) when it down counts to 0 in compare-counting mode and sets OVUNF (EQEI_STATUS[2]) to high in the next edge. For X2 mode, only CHA edge will set OVUNF (EQEI_STATUS[2]) while for X4 mode both CHA and CHB edges will set OVUNF (EQEI_STATUS[2]).

6.16.5.9 Compare Function

The compare function in EQEI controller is to compare the dynamic counting CNT(EQEI_CNT[31:0]) with the compare register CNTCMP(EQEI_CNTCMP[31:0]). When CNT(EQEI_CNT[31:0]) up or down counts and reaches CNTCMP(EQEI_CNTCMP[31:0]), the flag CMPF(EQEI_STATUS[1]) will be set. Set bit CMP_EN (EQEI_CTL[28]) to one to enable the compare function otherwise disable it.

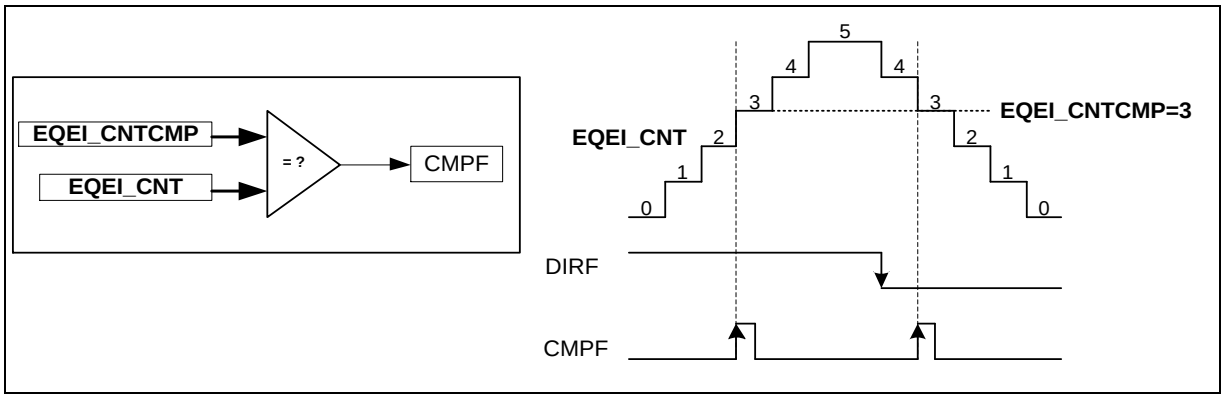


Figure 6.16-9 Compare Operation

6.16.5.10 1-phase 2 Input Phase Counting Mode

There are two different kinds of 1-phase 2 input phase counting modes in EQEI mode. The mode1 is used for counting pulse signals when the initial input signals are level high, and the other one, mode2, is used for counting level low signals. In phase counting mode, user can select clock rate X1 or X2 to trigger pulse counter by setting the EQEI Swap Function Clock Rate bits (EQEI_CTL2[2:1]).

Phase counting mode, initial high (mode1)

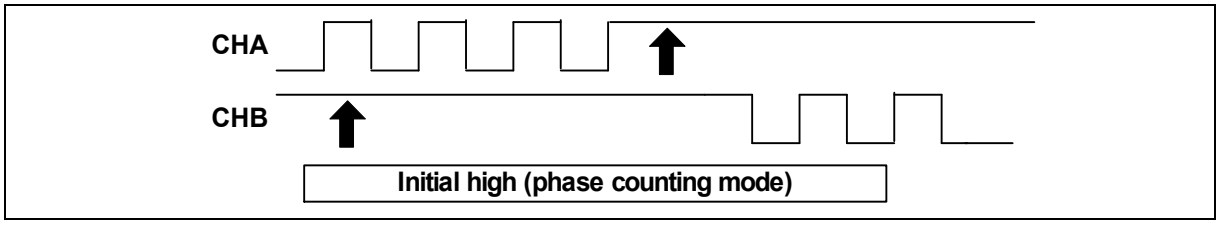


Figure 6.16-10 Phase Counting Mode, Mode1

Phase counting mode, initial low (mode2)

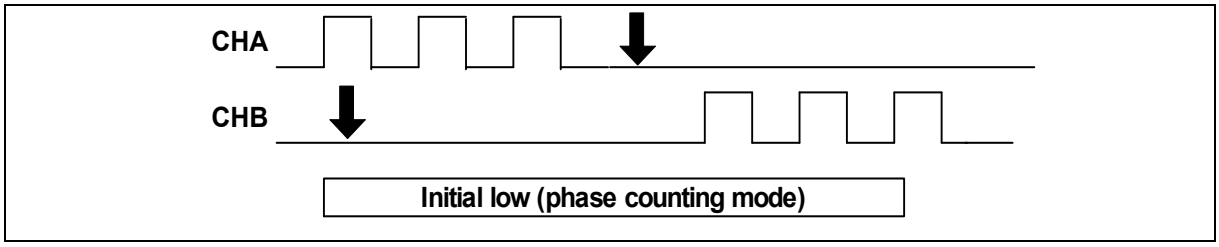


Figure 6.16-11 Phase Counting Mode, Mode2

X1/X2, 1-phase 2 input phase counting mode

In **X1 phase counting mode**, the pulse counter increases one on every CHA rising or falling edge and decrease one on every CHB rising or falling edge. User can select rising edge or falling edge to trigger pulse counter by setting the EQEI Clock Rate Setting without Quadrature Mode bits (EQEI_CTL2[2:1]) to 00 or 01. In this mode, every rising or falling edge on the QEA and QEB signal clocks the pulse counter.

In **X2 phase counting mode**, the pulse counter increases one on every CHA rising and falling edge and decreases one on every CHB rising and falling edge. User can trigger pulse counter with both rising and falling edge by setting the EQEI Clock Rate Setting without Quadrature Mode bits (EQEI_CTL2[2:1]) to 10. In this mode, every rising and falling edge on the QEA and QEB signal clocks the pulse counter.

X1 Phase counting mode, initial high (mode1), rising trigger

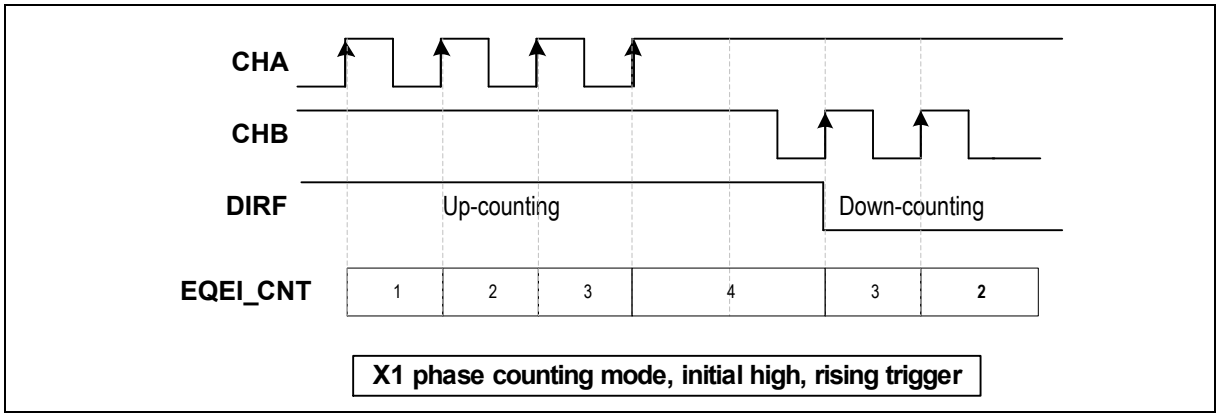


Figure 6.16-12 X1 Phase Counting Mode, Mode1, Rising Trigger

X1 Phase counting mode, initial high (mode1), falling trigger

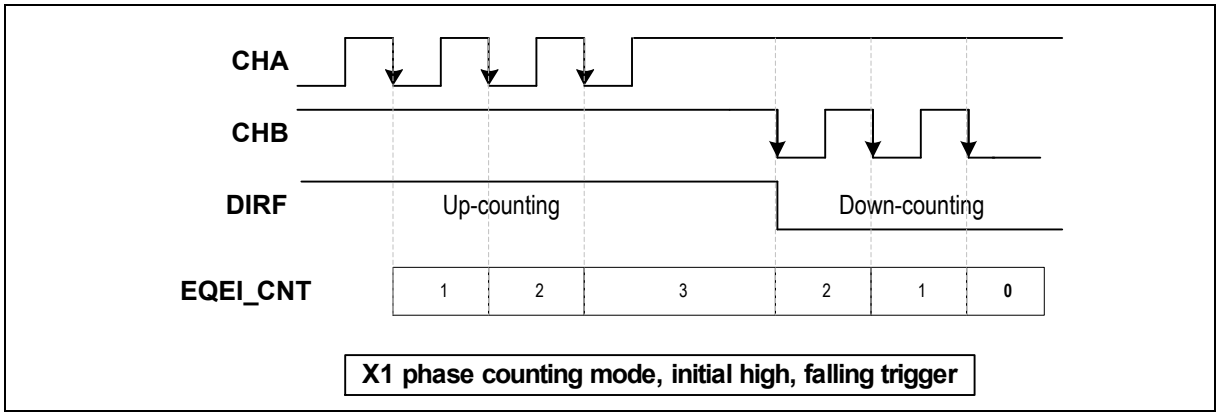


Figure 6.16-13 X1 Phase Counting Mode, Mode1, Falling Trigger

X2 Phase counting mode, initial high (mode1)

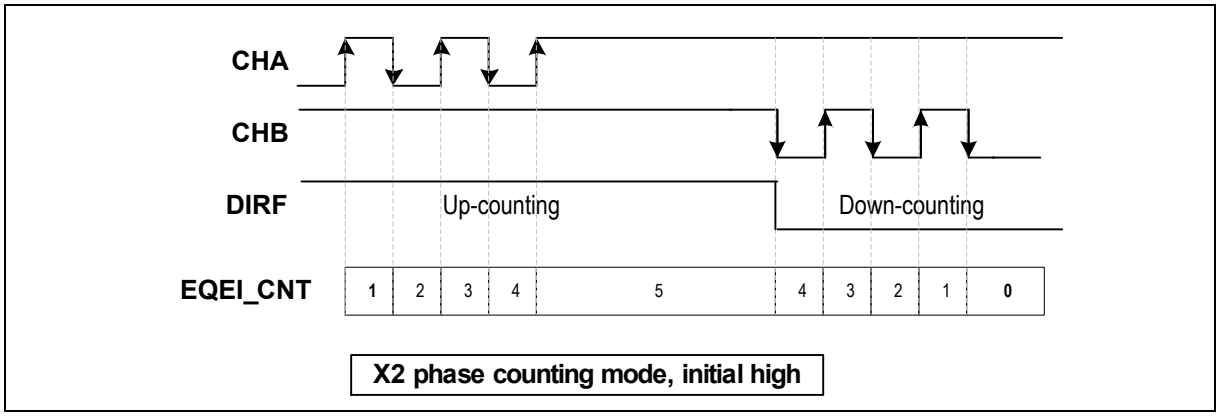


Figure 6.16-14 X2 Phase Counting Mode, Mode1

X1 Phase counting mode, initial low (mode2), rising trigger

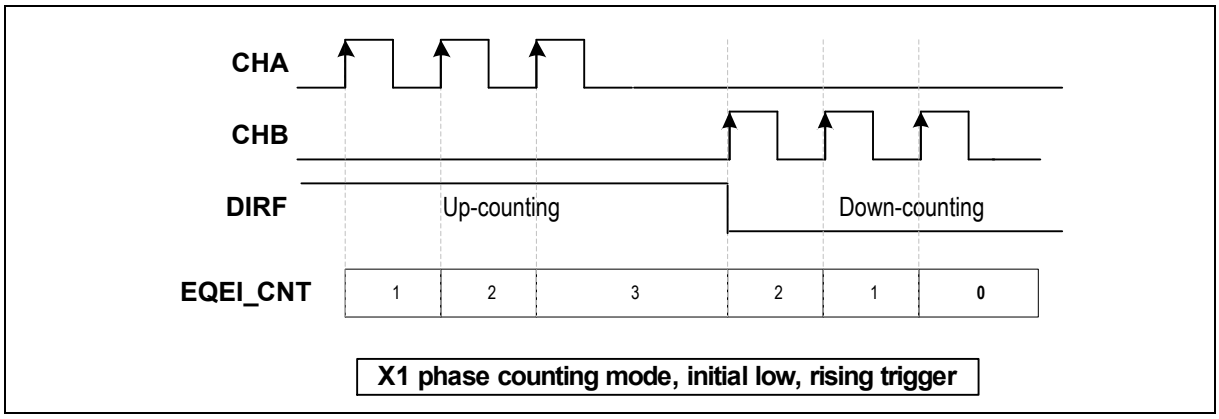


Figure 6.16-15 X1 Phase Counting Mode, Mode2, Rising Trigger

X1 Phase counting mode, initial low (mode2), falling trigger

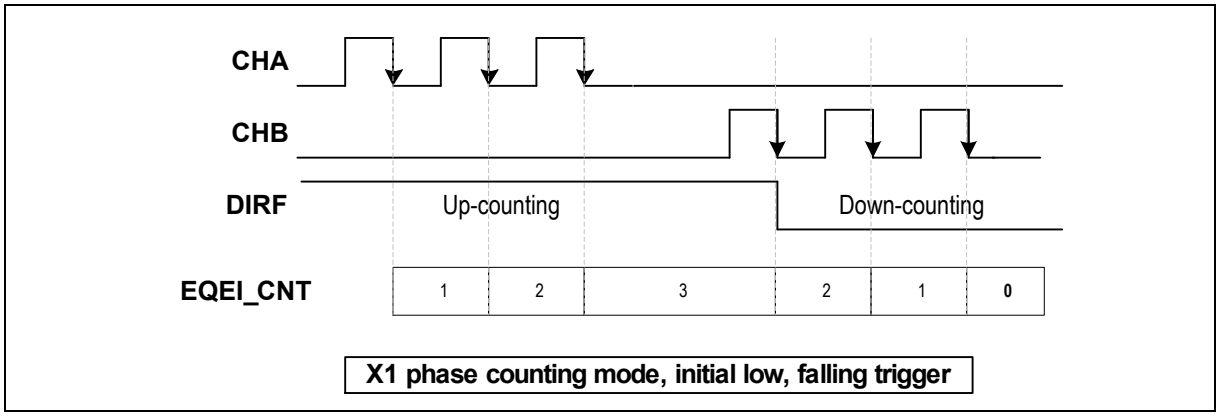


Figure 6.16-16 X1 Phase Counting Mode, Mode2, Falling Trigger

X2 Phase counting mode, initial low (mode2)

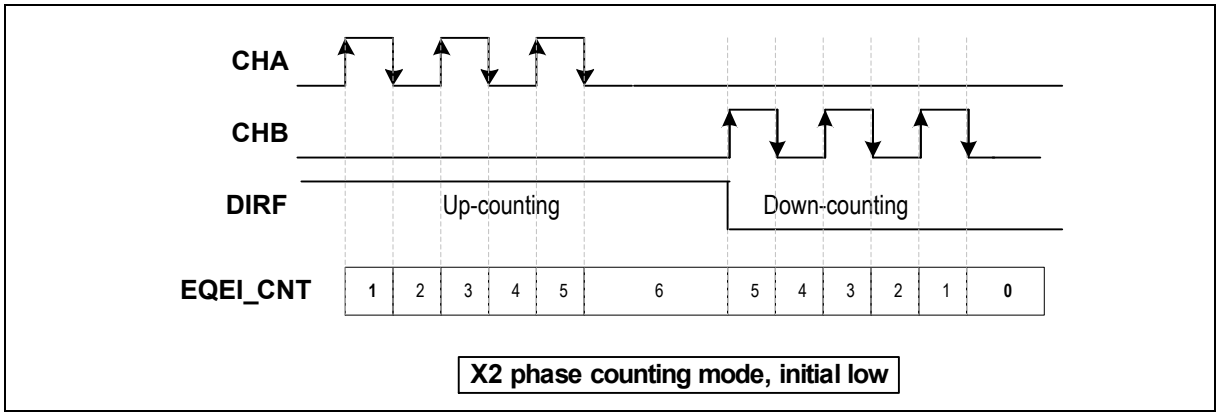


Figure 6.16-17 X2 Phase Counting Mode, Mode2

6.16.5.11 1-phase 1 input directional counting mode

There are three different kinds of 1-phase 1 input directional counting modes in EQEI mode. The first one is called “direction count mode”, which needs to input direction and clock signals from CHA and CHB. When user entered the signals into CHA and CHB, it will determine the pulse counter should add or subtract, and it also determine when the pulse counter should be driven. The second mode in 1-phase 1 input directional counting mode is called “direction up count mode”. In direction up count mode, user only needs to input clock to CHA and sets the register DIRSRC (EQEI_CTL2[5:4]) to 10. Then the pulse counter adds one automatically when the input clock triggers the counter. The last mode in EQEI mode is called “direction down count mode”. When user sets the register DIRSRC (EQEI_CTL2[5:4]) to 11 and input clock signal to CHA, then the pulse counter starts to substrate one by clock trigger automatically.

X1/X2, 1-phase 1 input directional counting mode

In **X1 directional counting mode**, the pulse counter increases or decreases one on every CHA rising or falling edge. User can select rising edge or falling edge to trigger pulse counter by setting the EQEI Clock Rate Setting without Quadrature Mode bits (EQEI_CTL2[2:1]) to 00 or 01. In this mode, every rising or falling edge on CHA signal clocks the pulse counter.

In **X2 directional counting mode**, the pulse counter increases or decreases one on every CHA rising and falling edge. User can trigger pulse counter with both rising and falling edge by setting the EQEI Clock Rate Setting without Quadrature Mode bits (EQEI_CTL2[2:1]) to 10. In this mode, every rising and falling edge on CHA signal clocks the pulse counter.

Direction count mode

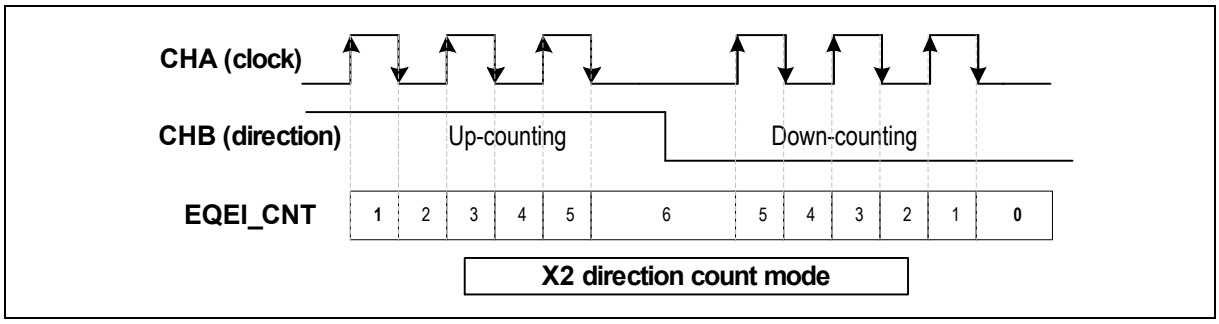


Figure 6.16-18 X2 Direction Count Mode

Direction up count mode

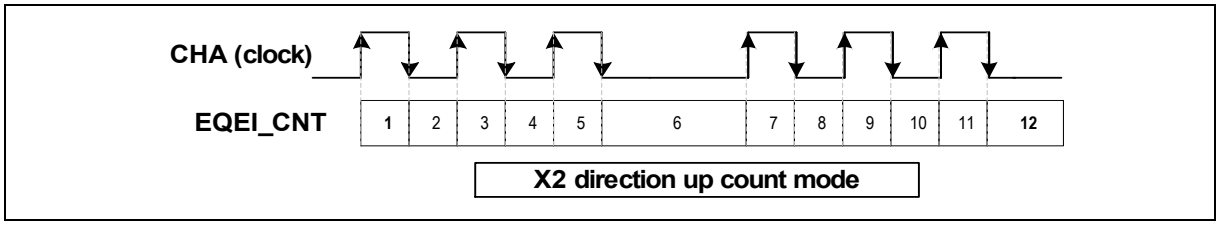


Figure 6.16-19 x2 Direction Up Count Mode

Direction down count mode

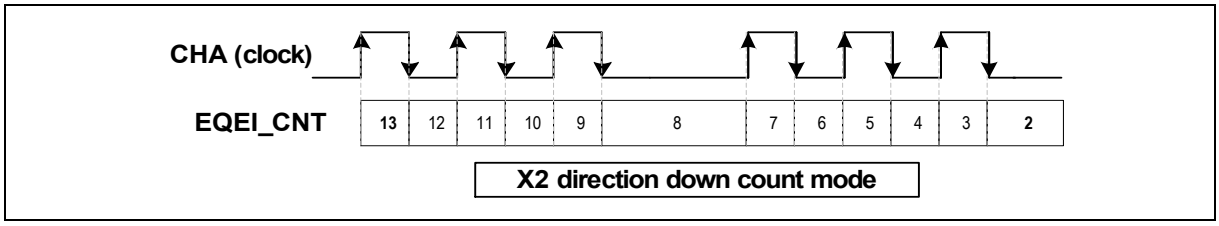


Figure 6.16-20 X2 Direction Down Count Mode

6.16.5.12 EQEI Swap Function for QA and QB

Swap function is a method to alter the destination of the input signals, which can help user to use EQEI mode more conveniently. User can shift QA and QB by controlling the register SWAPEN (EQEI_CTL2[0]) to 1. Normally the swap function is disabled if user did not set the register. Refer to Table 6.16-2.

MODE	QEA	QEB	SWAPEN (EQEI_CTL2[0])	DIRSRC (EQEI_CTL2[5:4])	QCLK	QDIR
Direction-count	CLK	DIR	0	00	CLK	DIR
Direction-count	DIR	CLK	1	00	CLK	DIR
Up count	CLK	x	0	10	CLK	1
Up count	x	CLK	1	10	CLK	1
Down count	CLK	x	0	11	CLK	0
Down count	x	CLK	1	11	CLK	0

MODE	QEA	QEB	SWAPEN (EQEI_CTL2[0])	DIRSRC (EQEI_CTL2[5:4])	QCLK	QDIR
Phase counting	A	B	0	00	CLK	DIR
Phase counting	B	A	1	00	CLK	DIR
MODE	QEA	QEB	SWAPEN (EQEI_CTL2[0])	DIRSRC (EQEI_CTL2[5:4])	QCLK	QDIR
Quadrature mode	A	B	0	00	CLK	DIR
Quadrature mode	A	B	1	00	CLK	DIR
Quadrature mode	B	A	0	00	CLK	DIR
Quadrature mode	B	A	1	00	CLK	DIR

Table 6.16-2 Swap Function Table

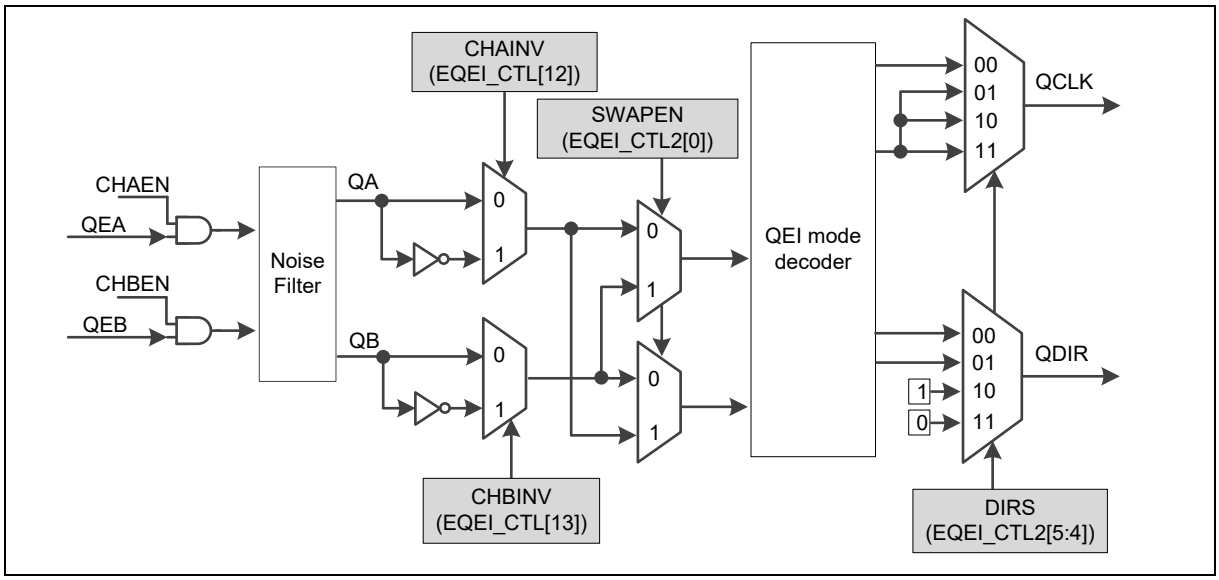


Figure 6.16-21 EQEI Swap Function for QA and QB

6.16.5.13 An unit timer counter in EQEI mode

There is a 32 bits UTCNT(EQEI_UTCNT) in EQEI mode which is triggered by clock. User can set the register UTEN(EQEI_CTL2[8]) to enable the unit timer counter to start counting. And user can also set the register UTHOLDEN(EQEI_CTL2[9]) to 1 to capture CNT(EQEI_CNT[31:0]) value and store into EQEI Counter Hold Register CNTHOLD(EQEI_CNTHOLD[31:0]) when the flag UTIEF(EQEI_STATUS[17]) is 1. If the unit timer counter value matches UTCMP(EQEI_UTCMP[31:0]), then the register UTIEF (EQEI_STATUS[17]) will be set. Unit timer can also help user to reset EQEI position counter when the register UTEVTRST(EQEI_CTL2[10]) is set to 1.There are three ways to initial unit timer value. The first one is written by software if UTEN (EQEI_CTL2[8]) = 0. The second one is the unit timer counter value matches UTCMP(EQEI_UTCMP[31:0]). And the last one is to set the register IDXRLDEN(EQEI_CTL[27]) =1 and IDXRSTUTS(EQEI_CTL2[11]=1, this would make the unit timer reset with EQEI_CNT when IDX reset event happens.

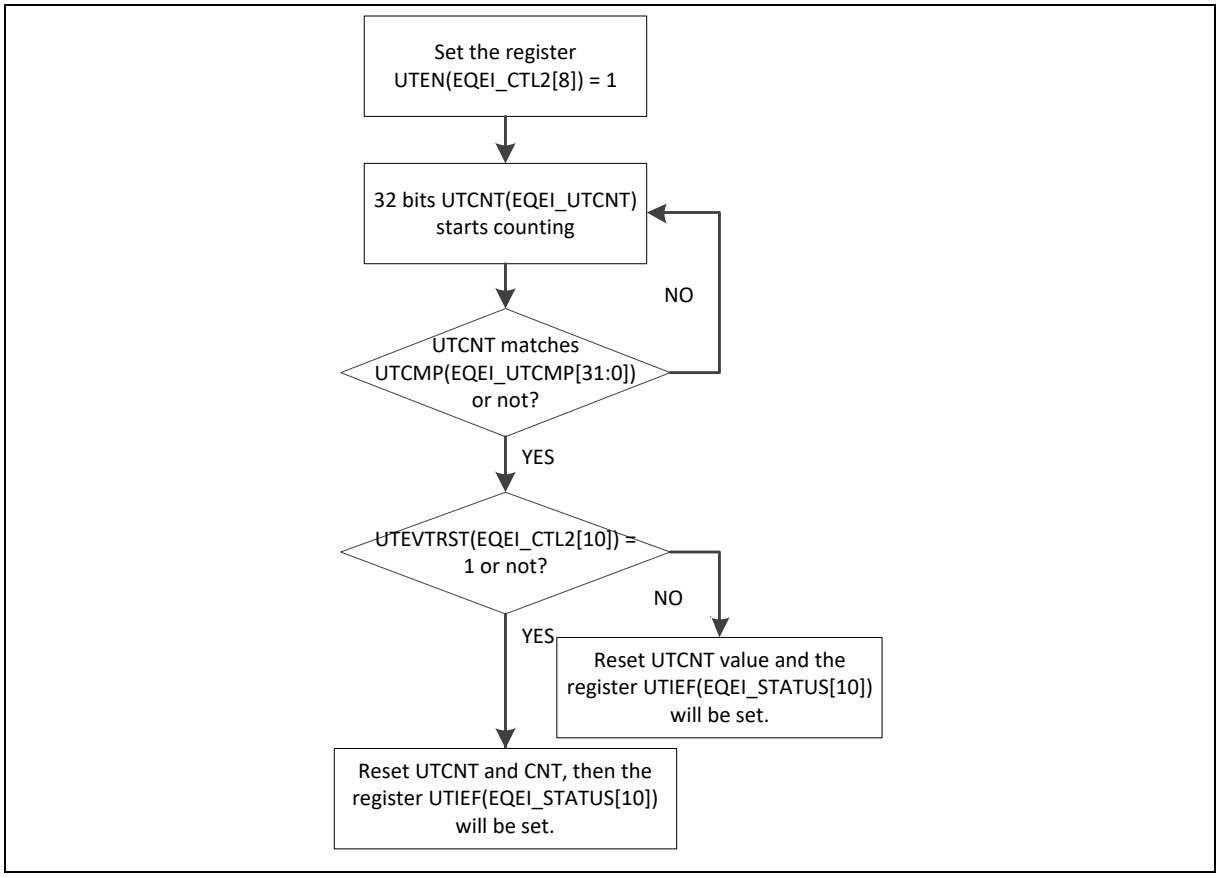


Figure 6.16-22 EQEI Unit Timer Counter Control Flow Chart

6.16.5.14 Reload Counter by Pin IDX

The CNT(EQEI_CNT[31:0]) counter can be reset to 0 or reload with the content of CNTMAX (EQEI_CNTMAX[31:0]) by the signal CHX (the filtered and polarity-set output of pin IDX) trigger. When the IDX Reload bit IDXRLD_EN(EQEI_CTL[27]) is set, a rising edge of CHX causes EQEI controller to reset the CNT(EQEI_CNT[31:0]) to 0 if the counter is in up-counting; if the counter is in down-counting the rising edge of CHX causes the EQEI controller to reload the CNT(EQEI_CNT[31:0]) with the content of CNTMAX (EQEI_CNTMAX[31:0]). Refer to Figure 6.16-23 for details.

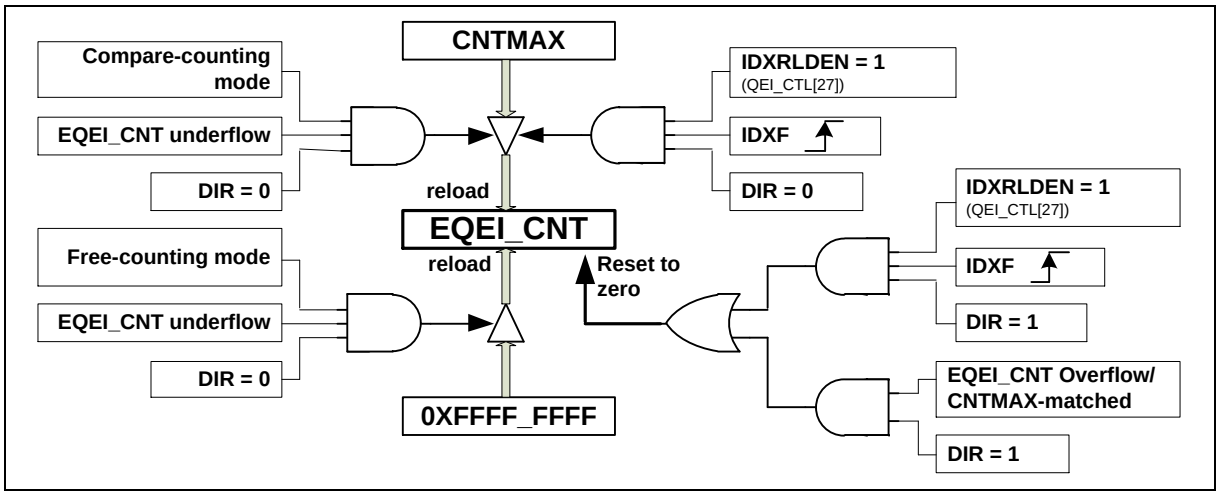


Figure 6.16-23 EQEI_CNT Reload/Reset Control

6.16.5.15 One Time Index Signal Reset Function

User can avoid CNT(EQEI_CNT[31:0]) being reset by selecting one time index signal reset function. To achieve this, user needs to set the register IDXRSTEN(EQEI_CTL[7]) and IDXRLDEN(EQEI_CTL[27]) to 1. Then CNT(EQEI_CNT[31:0]) will be reset when the first index signal coming. User should be aware that CNT(EQEI_CNT[31:0]) content will be latched into EQEI Counter Index Latch Register CNTLATCH (EQEI_CNTLATCH[31:0]) only at first rising edge of IDX signal, but UTCNT(EQEI_UTCNT[31:0]) is not subject to this restriction. User can select UTCNT(EQEI_UTCNT[31:0]) to be reset or not by setting IDXRSTUTS(EQEI_CTL[11]) register. If user wants to reset CNT(EQEI_CNT[31:0]) by next index signal when the first index signal resets has happened, user can set the register IDXRSTEV(EQEI_CTL[15]) to enable IDX to reset CNT(EQEI_CNT[31:0]) again, and monitor the flag FIDXEF(EQEI_STATUS[9]).

6.16.5.16 Capture EQEI Counter

If the bit HOLDCNT(EQEI_CTL[24]) or UTHOLDEN(EQEI_CTL2[9]) and UTIEF (EQEI_STATUS[17]) are set, the CNT(EQEI_CNT[31:0]) content will be captured into EQEI Counter Hold Register CNTHOLD(EQEI_CNTHOLD[31:0]), the data will be held until the next HOLDCNT (EQEI_CTL[24]) trigger comes. The bit HOLDCNT can be set by writing 1 to it or the rising edge of timers interrupt flags TIF (TIMERx_INTSTS[0]).

Note: The bit HOLDCNT is automatically cleared by hardware after CNTHOLD(EQEI_CNTHOLD[31:0]) captures the content of EQEI counter.

If the bit IDXLATEN (EQEI_CTL[25]) is set, the CNT(EQEI_CNT[31:0]) content will be latched into EQEI Counter Index Latch Register CNTLATCH (EQEI_CNTLATCH[31:0]) at every rising edge of CHX signal.

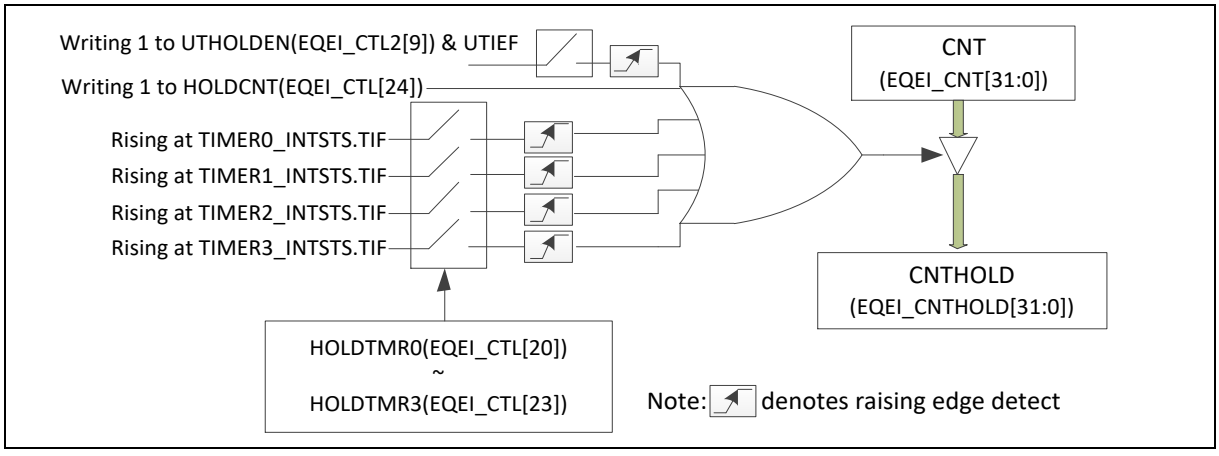


Figure 6.16-24 Trigger Control of Capturing EQEI Counter

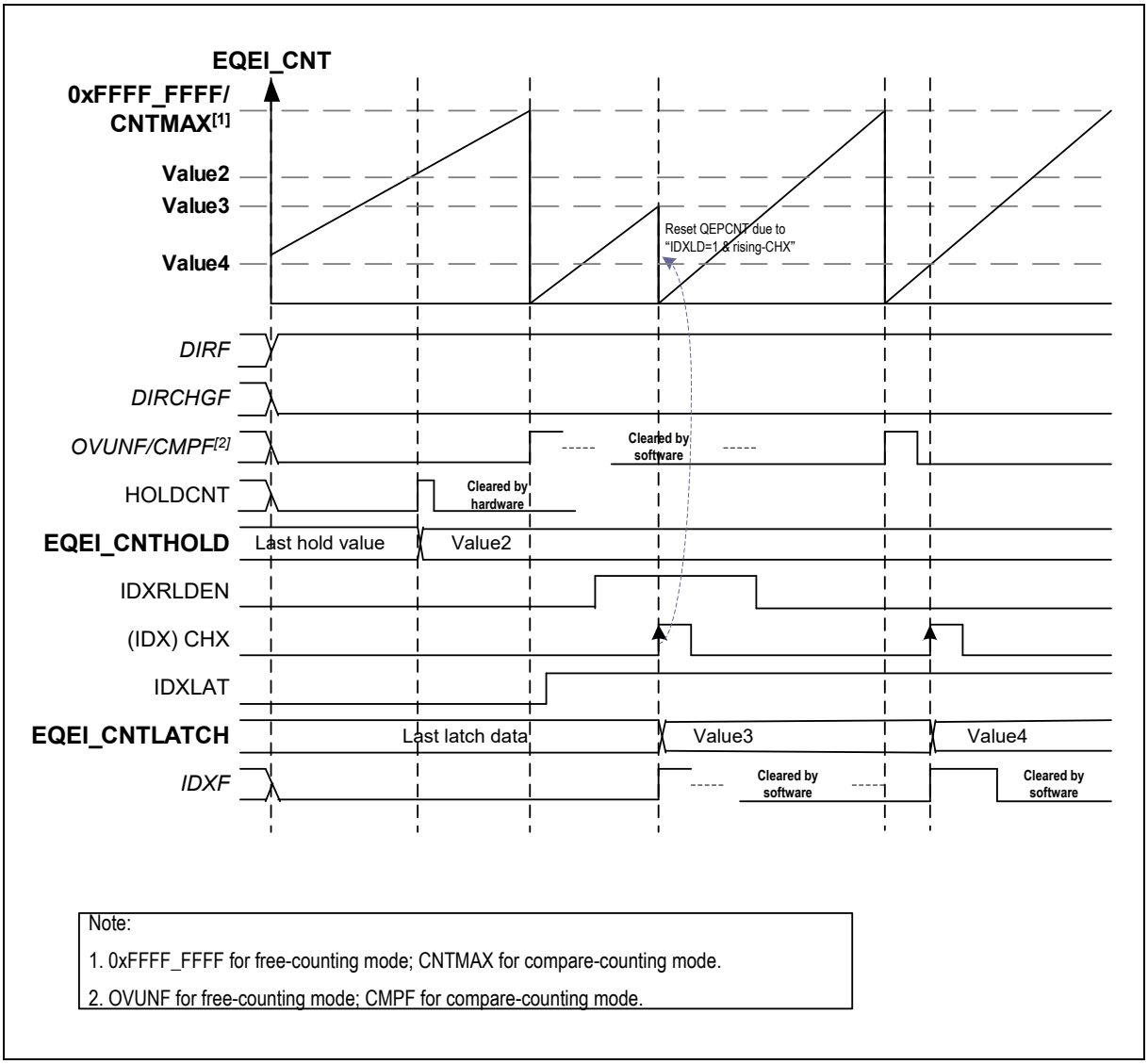


Figure 6.16-25 Capture and Latch EQEI Counter

6.16.5.17 EQEI Interrupt Architecture

There are six interrupt sources, each one of them has an interrupt flag and enable control bit, to trigger EQEI Interrupt. When the EQEI counter is up-counting and CNT(EQEI_CNT[31:0]) overflows or down-counting and underflows, the Overflow/Underflow flag OVUNF (EQEI_STATUS[2]) will be set by hardware and it will trigger EQEI Interrupt request if bit OVUNIEN (EQEI_CTL[16]) is high. When EQEI controller detects the encoder rotation change, it toggles the direction indication bit DIRF (EQEI_STATUS[8]) and the direction change flag DIRCHGF (EQEI_STATUS[3]) will be set by hardware that requests the EQEI interrupt if bit DIRIEN (EQEI_CTL[17]) is set. When the EQEI counter counting value is equal to the value of EQEI Counter Compare Register (CNTCMP(EQEI_CNTCMP[31:0])), the flag CMPF (EQEI_STATUS[1]) will be set by hardware and the EQEI Interrupt will be requested if bit CMPIEN (EQEI_CTL[18]) is high. When EQEI controller detects a rising edge at signal CHX (the filtered and polarity-set output of pin IDX), the flag IDXF(EQEI_STATUS[1]) will set by hardware and the EQEI interrupt will be requested if bit IDXIEN (EQEI_CTL[19]) is set. When the Unit Timer Counter's counting value equal to the value of Unit Timer Counter Compare Register (UTCMP(EQEI_UTCMP[31:0])), the flag UTIEF (EQEI_STATUS[10]) will be set by hardware and the EQEI Interrupt will be requested if bit TIEIEN (EQEI_CTL2[16]) is high. When the Phase error occurs in EQEI CHA and CHB, the flag PHEF (EQEI_STATUS[9]) will be set by hardware and the EQEI Interrupt will be requested if bit PHEIEN

(EQEI_CTL2[17]) is high. Note that the six flags, OVUNF(EQEI_STATUS[2]), DIRCHGF(EQEI_STATUS[3]), CMPF(EQEI_STATUS[1]), IDXF(EQEI_STATUS[0]), PHEF(EQEI_STATUS[9]), and UTIEF(EQEI_STATUS[10]) are set by hardware and must be cleared by software. Figure 6.16-26 demonstrates the architecture of Enhanced Quadrature Encoder Interface Controller interrupts.

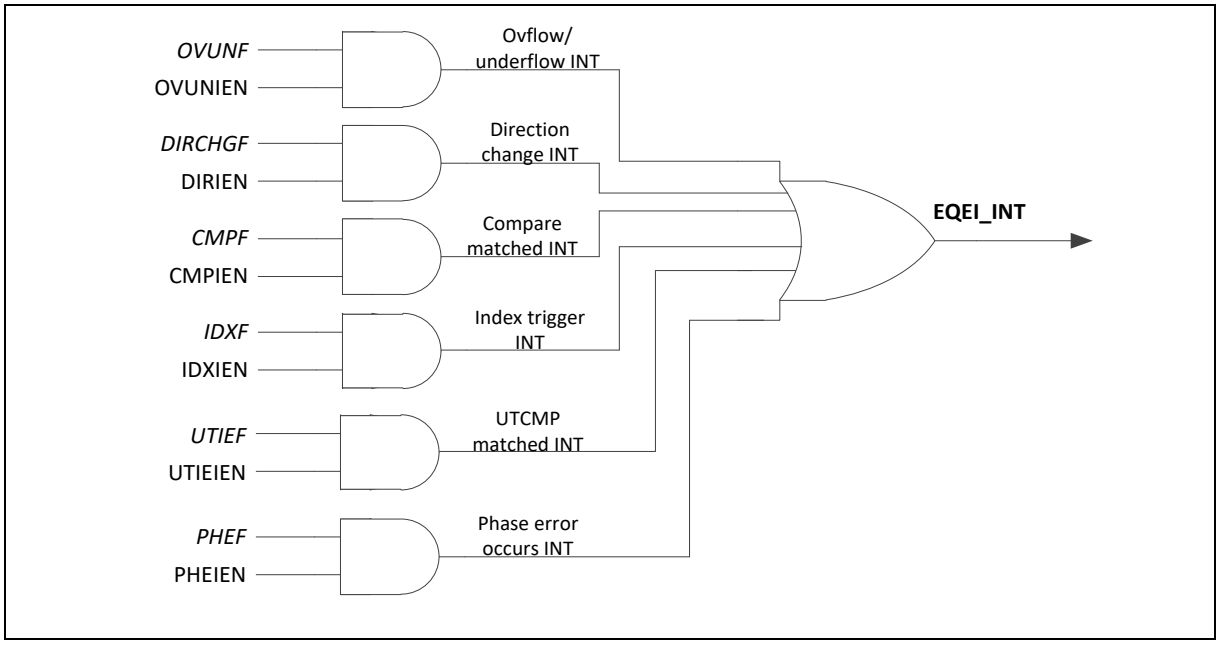


Figure 6.16-26 Enhanced Quadrature Encoder Interface Interrupt Architecture Diagram

6.16.5.18 EQEI phase error flag

Because input signals CHA and CHB have periodic changes, the mechanism of phase error monitor is used for detecting wrong phase. If the phase state is not correct, then the register PHEF (EQEI_STATUS[9]) will be set.

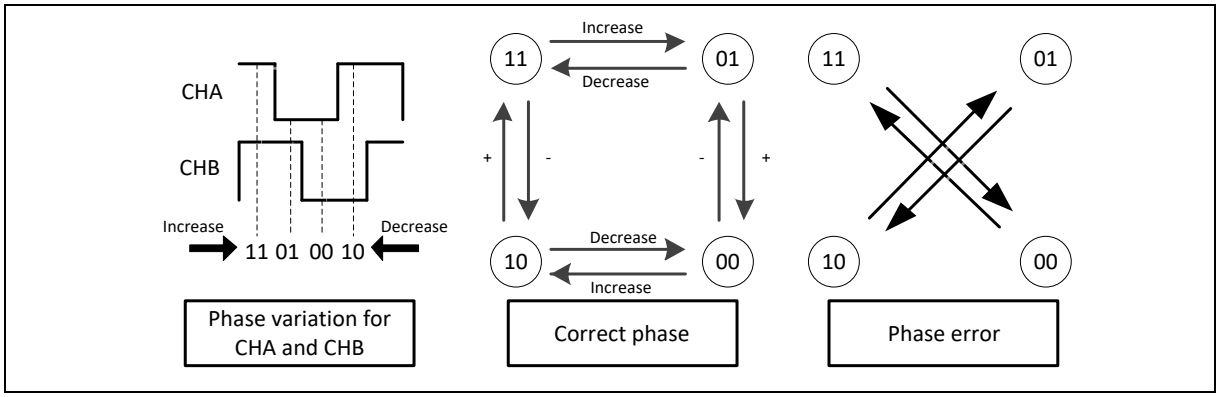


Figure 6.16-27 EQEI Phase Variation for CHA and CHB

6.16.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
EQEI Base Address: **EQEI0_BA = 0x400B_0000** **EQEI1_BA = 0x400B_1000**				
EQEI_CNT **x=0, 1**	EQEIx_BA+0x00	R/W	EQEI Counter Register	0x0000_0000
EQEI_CNTHOLD **x=0, 1**	EQEIx_BA+0x04	R/W	EQEI Counter Hold Register	0x0000_0000
EQEI_CNTLATCH **x=0, 1**	EQEIx_BA+0x08	R/W	EQEI Counter Index Latch Register	0x0000_0000
EQEI_CNTCMP **x=0, 1**	EQEIx_BA+0x0C	R/W	EQEI Counter Compare Register	0x0000_0000
EQEI_CNTMAX **x=0, 1**	EQEIx_BA+0x14	R/W	EQEI Pre-set Maximum Count Register	0x0000_0000
EQEI_CTL **x=0, 1**	EQEIx_BA+0x18	R/W	EQEI Controller Control Register	0x0000_0000
EQEI_CTL2 **x=0, 1**	EQEIx_BA+0x1C	R/W	EQEI Controller Control Register2	0x0000_0000
EQEI_UTCNT **x=0, 1**	EQEIx_BA+0x20	R/W	EQEI Unit Timer Counter Register	0x0000_0000
EQEI_UTCMP **x=0, 1**	EQEIx_BA+0x24	R/W	EQEI Unit Timer Compare Register	0xFFFF_FFFF
EQEI_STATUS **x=0, 1**	EQEIx_BA+0x2C	R/W	EQEI Controller Status Register	0x0000_0000

6.16.7 Register Description

EQEI Counter Register (EQEI_CNT)

Register	Offset	R/W	Description	Reset Value
EQEI_CNT	EQEIx_BA+0x00	R/W	EQEI Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
CNT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

Bits	Description	
[31:0]	**CNT**	**Enhanced Quadrature Encoder Interface Counter** A 32-bit up/down counter. When an effective phase pulse is detected, this counter is increased by one if the bit DIRF (EQEI_STATUS[8]) is one or decreased by one if the bit DIRF(EQEI_STATUS[8]) is 0. This register performs an integrator which count value is proportional to the encoder position. The pulse counter may be initialized to a predetermined value when one of the following events occur: ● Software is written if EQEIEN (EQEI_CTL[29]) = 0. ● Compare-match event if EQEIEN(EQEI_CTL[29])=1 and EQEI is in compare-counting mode. ● Index signal change if EQEIEN(EQEI_CTL[29])=1 and IDXRLDEN (EQEI_CTL[27])=1.

EQEI Counter Hold Register (EQEI_CNTHOLD)

Register	Offset	R/W	Description	Reset Value
EQEI_CNTHOLD	EQEIx_BA+0x04	R/W	EQEI Counter Hold Register	0x0000_0000

31	30	29	28	27	26	25	24
CNTHOLD							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CNTHOLD							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNTHOLD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNTHOLD							

Bits	Description	
[31:0]	**CNTHOLD**	**Enhanced Quadrature Encoder Interface Counter Hold** When the bit HOLDCNT (EQEI_CTL[24]) goes from low to high, the CNT(EQEI_CNT[31:0]) is copied into CNTHOLD (EQEI_CNTHOLD[31:0]) register.

EQEI Counter Index Latch Register (EQEI_CNTLATCH)

Register	Offset	R/W	Description	Reset Value
EQEI_CNTLATCH	EQEIx_BA+0x08	R/W	EQEI Counter Index Latch Register	0x0000_0000

31	30	29	28	27	26	25	24
CNTLATCH							
23	22	21	20	19	18	17	16
CNTLATCH							
15	14	13	12	11	10	9	8
CNTLATCH							
7	6	5	4	3	2	1	0
CNTLATCH							

Bits	Description	
[31:0]	CNTLATCH	**Enhanced Quadrature Encoder Interface Counter Index Latch** When the IDXF (EQEI_STATUS[0]) bit is set, the CNT(EQEI_CNT[31:0]) is copied into CNTLATCH (EQEI_CNTLATCH[31:0]) register.

EQEI Counter Compare Register (EQEI_CNTCMP)

Register	Offset	R/W	Description	Reset Value
EQEI_CNTCMP	EQEIx_BA+0x0C	R/W	EQEI Counter Compare Register	0x0000_0000

31	30	29	28	27	26	25	24
CNTCMP							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CNTCMP							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNTCMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNTCMP							

Bits	Description	
[31:0]	**CNTCMP**	**Enhanced Quadrature Encoder Interface Counter Compare** If the EQEI controller is in the compare-counting mode CMPEN (EQEI_CTL[28]) =1, when the value of CNT(EQEI_CNT[31:0]) matches CNTCMP(EQEI_CNTCMP[31:0]), CMPF will be set. This register is software writable.

EQEI Pre-set Maximum Count Register (EQEI_CNTMAX)

Register	Offset	R/W	Description	Reset Value
EQEI_CNTMAX	EQEIx_BA+0x14	R/W	EQEI Pre-set Maximum Count Register	0x0000_0000

31	30	29	28	27	26	25	24
CNTMAX							
23	22	21	20	19	18	17	16
CNTMAX							
15	14	13	12	11	10	9	8
CNTMAX							
7	6	5	4	3	2	1	0
CNTMAX							

Bits	Description	
[31:0]	**CNTMAX**	**Enhanced Quadrature Encoder Interface Preset Maximum Count** This register value determined by user stores the maximum value which may be the number of the EQEI counter for the EQEI controller compare-counting mode.

EQEI Controller Control Register (EQEI_CTL)

Register	Offset	R/W	Description	Reset Value
EQEI_CTL	EQEIx_BA+0x18	R/W	EQEI Controller Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		EQEIEN	CMPEN	IDXRLDEN	Reserved	IDXLATEN	HOLDCNT
23	22	21	20	19	18	17	16
HOLDTMR3	HOLDTMR2	HOLDTMR1	HOLDTMR0	IDXIEN	CMPIEN	DIRIEN	OVUNIEN
15	14	13	12	11	10	9	8
IDXRSTEV	IDXINV	CHBINV	CHAINV	Reserved	MODE		
7	6	5	4	3	2	1	0
IDXRSTEN	IDXEN	CHBEN	CHAEN	NFDIS	NFCLKSEL		

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**EQEIEN**	**Enhanced Quadrature Encoder Interface Controller Enable Bit** 0 = EQEI controller function Disabled. 1 = EQEI controller function Enabled.
[28]	**CMPEN**	**The Compare Function Enable Bit** The compare function in EQEI controller is to compare the dynamic counting EQEI_CNT with the compare register CNTCMP(EQEI_CNTCMP[31:0]), if CNT(EQEI_CNT[31:0]) reaches CNTCMP(EQEI_CNTCMP[31:0]), the flag CMPF will be set. 0 = Compare function Disabled. 1 = Compare function Enabled.
[27]	**IDXRLDEN**	**Index Trigger EQEI_CNT Reload Enable Bit** When this bit is high and a rising edge comes on signal CHX, the CNT(EQEI_CNT[31:0]) will be reset to 0 if the counter is in up-counting type (DIRF(EQEI_STATUS[8]) = 1); while the CNT(EQEI_CNT[31:0]) will be reloaded with CNTMAX (EQEI_CNTMAX[31:0]) content if the counter is in down-counting type (DIRF(EQEI_STATUS[8]) = 0). 0 = Reload function Disabled. 1 = EQEI_CNT re-initialized by Index signal Enabled.
[26]	**Reserved**	Reserved.
[25]	**IDXLATEN**	**Index Latch EQEI_CNT Enable Bit** If this bit is set to high, the CNT(EQEI_CNT[31:0]) content will be latched into CNTLATCH (EQEI_CNTLATCH[31:0]) at every rising on signal CHX. 0 = The index signal latch EQEI counter function Disabled. 1 = The index signal latch EQEI counter function Enabled.

[24]	**HOLDCNT**	**Hold EQEI_CNT Control** When this bit is set from low to high, the CNT(EQEI_CNT[31:0]) is copied into CNTHOLD(EQEI_CNTHOLD[31:0]). This bit may be set by writing 1 to it or Timer0~Timer3 interrupt flag TIF (TIMERx_INTSTS[0]). 0 = No operation. 1 = EQEI_CNT content is captured and stored in CNTHOLD(EQEI_CNTHOLD[31:0]). **Note:** This bit is automatically cleared after EQEI_CNTHOLD holds EQEI_CNT value.
[23]	**HOLDTMR3**	**Hold EQEI_CNT by Timer 3** 0 = TIF (TIMER3_INTSTS[0]) has no effect on HOLDCNT. 1 = A rising edge of bit TIF(TIMER3_INTSTS[0]) in timer 3 sets HOLDCNT to 1.
[22]	**HOLDTMR2**	**Hold EQEI_CNT by Timer 2** 0 = TIF(TIMER2_INTSTS[0]) has no effect on HOLDCNT. 1 = A rising edge of bit TIF(TIMER2_INTSTS[0]) in timer 2 sets HOLDCNT to 1.
[21]	**HOLDTMR1**	**Hold EQEI_CNT by Timer 1** 0 = TIF(TIMER1_INTSTS[0]) has no effect on HOLDCNT. 1 = A rising edge of bit TIF (TIMER1_INTSTS[0]) in timer 1 sets HOLDCNT to 1.
[20]	**HOLDTMR0**	**Hold EQEI_CNT by Timer 0** 0 = TIF (TIMER0_INTSTS[0]) has no effect on HOLDCNT. 1 = A rising edge of bit TIF(TIMER0_INTSTS[0]) in timer 0 sets HOLDCNT to 1.
[19]	**IDXIEN**	**IDXF Trigger EQEI Interrupt Enable Bit** 0 = The IDXF can trigger EQEI interrupt Disabled. 1 = The IDXF can trigger EQEI interrupt Enabled.
[18]	**CMPIEN**	**CMPF Trigger EQEI Interrupt Enable Bit** 0 = CMPF can trigger EQEI controller interrupt Disabled. 1 = CMPF can trigger EQEI controller interrupt Enabled.
[17]	**DIRIEN**	**DIRCHGF Trigger EQEI Interrupt Enable Bit** 0 = DIRCHGF can trigger EQEI controller interrupt Disabled. 1 = DIRCHGF can trigger EQEI controller interrupt Enabled.
[16]	**OVUNIEN**	**OVUNF Trigger EQEI Interrupt Enable Bit** 0 = OVUNF can trigger EQEI controller interrupt Disabled. 1 = OVUNF can trigger EQEI controller interrupt Enabled.
[15]	**IDXRSTEV**	**IDX Signal Resets Enable Bit in First IDX Reset Event (Write Only)** 0 = The next IDX level high signal reset function Disabled. 1 = The next IDX level high signal reset function Eenabled. **Note:** This bit is only effective when IDXRSTEN (EQEI_CTL[7])=1 and IDXRLDEN (EQEI_CTL[27]) = 1.
[14]	**IDXINV**	**Inverse IDX Input Polarity** 0 = Not inverse IDX input polarity. 1 = IDX input polarity is inversed to EQEI controller.
[13]	**CHBINV**	**Inverse QEB Input Polarity** 0 = Not inverse QEB input polarity. 1 = QEB input polarity is inversed to EQEI controller.
[12]	**CHAINV**	**Inverse QEA Input Polarity** 0 = Not inverse QEA input polarity. 1 = QEA input polarity is inversed to EQEI controller.

[11]	**Reserved**	Reserved.
[10:8]	**MODE**	**EQEI Counting Mode Selection** There are seven quadrature encoder pulse counter operation modes. 000 = X4 Free-counting Mode. 001 = X2 Free-counting Mode. 010 = X4 Compare-counting Mode. 011 = X2 Compare-counting Mode. 100 = Phase Counting Mode Type 1. (PCMT1). 101 = Phase Counting Mode Type 2. (PCMT2). 110 = Directional Counting Mode. 111 = Reserved. **Note:** User needs to set DIRSRC(EQEI_CTL2[5:4]) when MODE(EQEI_CTL[10:8]) selects to directional counting mode.
[7]	**IDXRSTEN**	**IDX Reset EQEI Position Counter Enable Bit** 0 = Reset EQEI position counter in every time IDX signal. 1 = Reset EQEI position counter in first time IDX signal. **Note:** IDXRLDEN(EQEI_CTL[27]) should be set 1.
[6]	**IDXEN**	**IDX Input to EQEI Controller Enable Bit** 0 = IDX input to EQEI Controller Disabled. 1 = IDX input to EQEI Controller Enabled.
[5]	**CHBEN**	**QEB Input to EQEI Controller Enable Bit** 0 = QEB input to EQEI Controller Disabled. 1 = QEB input to EQEI Controller Enabled.
[4]	**CHAEN**	**QEA Input to EQEI Controller Enable Bit** 0 = QEA input to EQEI Controller Disabled. 1 = QEA input to EQEI Controller Enabled.
[3]	**NFDIS**	**EQEI Controller Input Noise Filter Disable Bit** 0 = The noise filter of EQEI controller Enabled. 1 = The noise filter of EQEI controller Disabled.
[2:0]	**NFCLKSEL**	**Noise Filter Clock Pre-divide Selection** To determine the sampling frequency of the Noise Filter clock. 000 = EQEI_CLK. 001 = EQEI_CLK/2. 010 = EQEI_CLK/4. 011 = EQEI_CLK/16. 100 = EQEI_CLK/32. 101 = EQEI_CLK/64. 110 = reserved. 111 = reserved.

EQEI Controller Control Register2 (EQEI_CTL2)

Register	Offset	R/W	Description	Reset Value
EQEI_CTL2	EQEIx_BA+0x1C	R/W	EQEI Controller Control Register2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						UTIEIEN	PHEIEN
15	14	13	12	11	10	9	8
Reserved				IDXRSTUTS	UTEVTRST	UTHOLDEN	UTEN
7	6	5	4	3	2	1	0
Reserved		DIRSRC		Reserved	CRS		SWAPEN

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**UTIEIEN**	**UTIEF Trigger EQEI Interrupt Enable Bit** 0 = UTIEF can trigger EQEI controller interrupt Disabled. 1 = UTIEF can trigger EQEI controller interrupt Enabled.
[16]	**PHEIEN**	**PHEF Trigger EQEI Interrupt Enable Bit** 0 = PHEF can trigger EQEI controller interrupt Disabled. 1 = PHEF can trigger EQEI controller interrupt Enabled.
[15:12]	**Reserved**	Reserved.
[11]	**IDXRSTUTS**	**IDX Resets Unit Timer Select Bit** 0 = Unit timer will not be reset when IDX reset event happens. 1 = Resets unit timer or not will follow EQEI_CNT when IDX reset event happens.
[10]	**UTEVTRST**	**Enable Bit to Reset EQEI Position Counter by Unit Timer Event** 0 = Reset EQEI position counter feature when unit timer counter event occurs Disabled. 1 = Reset EQEI position counter feature when unit timer counter event occurs Enabled.
[9]	**UTHOLDEN**	**Unit Timer Counter Hold Enable Bit** 0 = No operation. 1 = EQEI_CNT content is captured and stored in CNTHOLD(EQEI_CNTHOLD[31:0]) when UTCNT matches UTCMP(EQEI_UTCMP[31:0]).
[8]	**UTEN**	**Unit Timer Function Enable Bit** 0 = EQEI unit timer function Disabled. 1 = EQEI unit timer function Enabled.
[7:6]	**Reserved**	Reserved.

[5:4]	**DIRSRC**	**Direction Signal Source Select** 00 = Direction signal is determined from EQEI system calculation. 01 = reserved. 10 = Direction signal is tied 1 only for direction up count mode. 11 = Direction signal is tied 0 only for down count mode.
[3]	**Reserved**	Reserved.
[2:1]	**CRS**	**Clock Rate Setting without Quadrature Mode** 00 = EQEI counter only counts the falling edge. 01 = EQEI counter only counts the rising edge. 10 = EQEI counter counts the rising and falling edge. 11 = reserved.
[0]	**SWAPEN**	**Swap Function Enable Bit** 0 = EQEI swap function Disabled. 1 = EQEI swap function Enabled.

EQEI Unit Timer Counter Register (EQEI_UTCNT)

Register	Offset	R/W	Description	Reset Value
EQEI_UTCNT	EQEIx_BA+0x20	R/W	EQEI Unit Timer Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
UTCNT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
UTCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
UTCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UTCNT							

Bits	Description	
[31:0]	**UTCNT**	**Unit Timer Counter** A 32-bit unit timer counter which may be reset to an initial value when any of the following events occur: ● Software is written if UTEN (EQEI_CTL2[8]) = 0. ● UT_EN (EQEI_CTL2[8]) =1, and the unit timer counter value matches UTCMP(EQEI_UTCMP[31:0]). ● IDXRLDEN(EQEI_CTL[27]) =1 and IDXRSTUTS(EQEI_CTL2[11]=1, determine the unit timer to be reset or not will follow EQEI_CNT when IDX reset event happens.

EQEI Unit Timer Compare Register (EQEI_UTCMP)

Register	Offset	R/W	Description	Reset Value
EQEI_UTCMP	EQEIx_BA+0x24	R/W	EQEI Unit Timer Compare Register	0xFFFF_FFFF

31	30	29	28	27	26	25	24
UTCMP							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
UTCMP							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
UTCMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UTCMP							

Bits	Description	
[31:0]	**UTCMP**	**Unit Timer Counter Compare** If the EQEI unit timer is enabled (EQEI_CTL2[8]) =1, and the unit timer counter value also matches UTCMP(EQEI_UTCMP[31:0]), then UTIEF (EQEI_STATUS[10]) will be set. This register is software writable.

EQEI Controller Status Register (EQEI_STATUS)

Register	Offset	R/W	Description	Reset Value
EQEI_STATUS	EQEIx_BA+0x2C	R/W	EQEI Controller Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						UTIEF	PHEF
15	14	13	12	11	10	9	8
Reserved						FIDXEF	DIRF
7	6	5	4	3	2	1	0
Reserved				DIRCHGF	OVUNF	CMPF	IDXF

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**UTIEF**	**EQEI Unit Timer Event Flag** 0 = No timer event occurs in EQEI unit timer counter. 1 = Unit timer event occurs in EQEI unit timer counter. **Note:** This bit is only cleared by writing 1 to it.
[16]	**PHEF**	**EQEI Phase Error Flag** 0 = No Phase error occurs in EQEI CHA and CHB. 1 = Phase error occurs in EQEI CHA and CHB. **Note:** This bit is only cleared by writing 1 to it.
[15:10]	**Reserved**	Reserved.
[9]	**FIDXEF**	**First IDX Signal Reset Event Flag (Read Only)** 0 = The first IDX reset event has not happened yet. 1 = The first IDX reset event has happened. **Note:** This bit is only effective when IDXRSTEN (EQEI_CTL[7])=1 and IDXRLDEN (EQEI_CTL[27])=1.
[8]	**DIRF**	**EQEI Counter Counting Direction Indication** 0 = EQEI Counter is in down-counting. 1 = EQEI Counter is in up-counting. **Note:** This bit is set/reset by hardware according to the phase detection between CHA and CHB.
[7:4]	**Reserved**	Reserved.
[3]	**DIRCHGF**	**Direction Change Flag** Flag is set by hardware while EQEI counter counting direction is changed. Software can clear this bit by writing 1 to it. 0 = No change in EQEI counter counting direction. 1 = EQEI counter counting direction is changed. **Note:** This bit is only cleared by writing 1 to it.

[2]	**OVUNF**	**EQEI Counter Overflow or Underflow Flag** Flag is set by hardware while CNT(EQEI_CNT[31:0]) overflows from 0xFFFF_FFFF to 0 in free-counting mode or from the CNTMAX (EQEI_CNTMAX[31:0]) to 0 in compare-counting mode. Similarly, the flag is set while EQEI counter underflows from 0 to 0xFFFF_FFFF or CNTMAX (EQEI_CNTMAX[31:0]). 0 = No overflow or underflow occurs in EQEI counter. 1 = EQEI counter occurs counting overflow or underflow. **Note:** This bit is only cleared by writing 1 to it.
[1]	**CMPF**	**Compare-match Flag** If the EQEI compare function is enabled, the flag is set by hardware while EQEI counter up or down counts and reach to the CNTCMP(EQEI_CNTCMP[31:0]). 0 = EQEI counter does not match with CNTCMP(EQEI_CNTCMP[31:0]). 1 = EQEI counter counts to the same as CNTCMP(EQEI_CNTCMP[31:0]). **Note:** This bit is only cleared by writing 1 to it.
[0]	**IDXF**	**IDX Detected Flag** When the EQEI controller detects a rising edge on signal CHX it will set flag IDXF to high. 0 = No rising edge detected on signal CHX. 1 = A rising edge occurs on signal CHX. **Note:** This bit is only cleared by writing 1 to it.

6.17 Enhanced Input Capture Timer (ECAP)

6.17.1 Overview

The chip provides up to two units of Input Capture Timer/Counter whose capture function can detect the digital edge-changed signal at channel inputs. Each unit has three input capture channels. The timer/counter is equipped with up counting, reload and compare-match capabilities.

6.17.2 Features

- Up to two Input Capture Timer/Counter units, CAP0 and CAP1.
- Each unit has 3 input channels.
- Each unit has its own interrupt vector.
- Each input channel has its own capture counter hold register.
- 24-bit Input Capture up-counting timer/counter.
- With noise filter in front end of input ports.
- Edge detector with three options:
  - Rising edge detection
  - Falling edge detection
  - Both edge detection
- Captured events reset and/or reload capture counter.
- Supports compare-match function.

6.17.3 Block Diagram

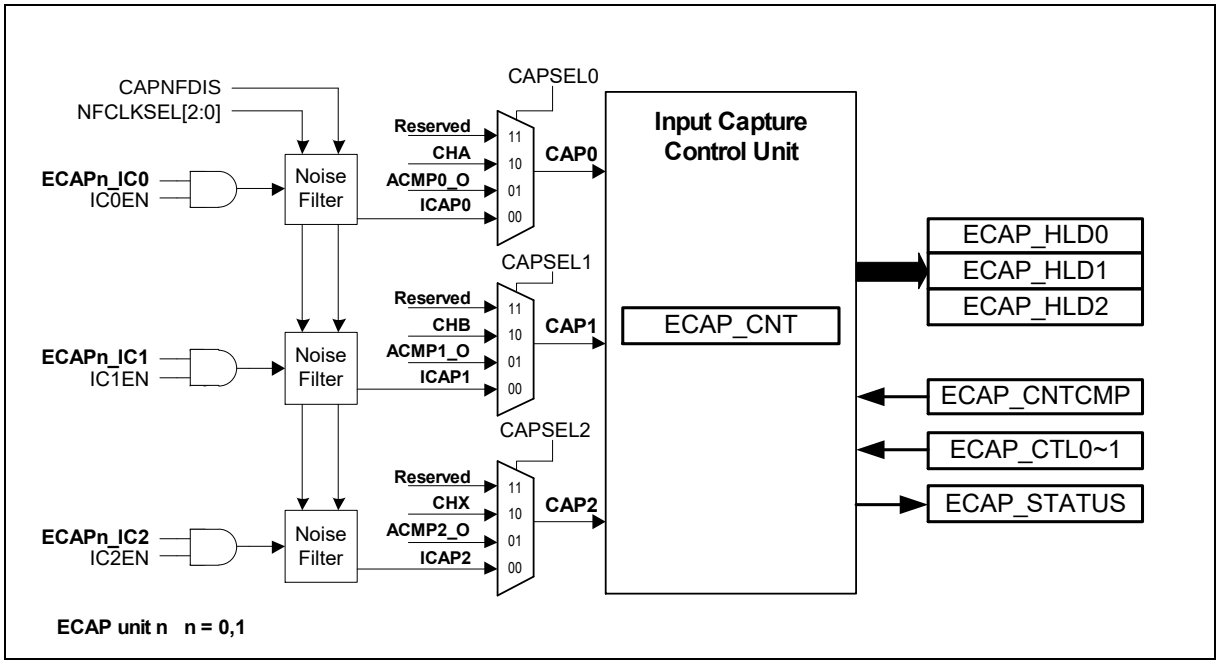


Figure 6.17-1 Input Capture Timer/Counter Architecture

6.17.4 Basic Configuration

6.17.4.1 ECAP0 Basic Configuration

- Clock Source Configuration
  - Enable ECAP0 peripheral clock in ECAP0CKEN(CLK_APBCLK1[26]).
- Reset Configuration
  - Reset ECAP0 peripheral in ECAP0RST (SYS_IPRST2[26]).

6.17.4.2 ECAP1 Basic Configuration

- Clock Source Configuration
  - Enable ECAP1 peripheral clock in ECAP1CKEN (CLK_APBCLK1[27]).
- Reset Configuration
  - Reset ECAP1 peripheral in ECAP1RST (SYS_IPRST2[27]).

6.17.5 Functional Description

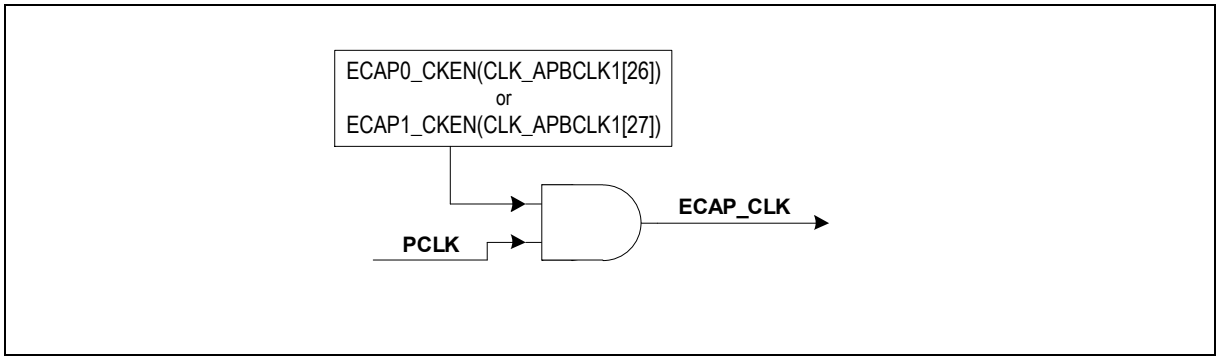


Figure 6.17-2 Input Capture Timer/Counter Clock Source Control

Figure 6.17-1 illustrates the architecture of the Input Capture. Each input capture timer/counter unit supports 3 input channels with programmable input signal sources. The port pins ECAP_IC0 to ECAP_IC2 can be fed to the inputs of capture unit through noise filter or bypass it (CAPNFDIS = 1), and the QEI controller input signals (CHA, CHB and CHX) and ACMP output signals (ACMP0_O, ACMP1_O and ACMP2_O) also can be internally routed to the capture inputs by setting the register ECAP_CTL0 (CAPSEL0~CAPSEL2). Note that the source toggle speed must be lower than PCLK/2 to successful detect capture source edge transtion.

6.17.5.1 Input Noise Filter

The architecture of input noise filter is similar to that one used for QEI (refer to Figure 6.16-4). With 6 sampling-rate options, it supports a wide range of filtering noise whose duration is from 42 ns (NFCLKSEL = 000) to 3.5 us (NFCLKSEL = 101) while PCLK is running at the frequency of 72 MHz. Table 6.17-1 lists the relation between the setting of NFCLKSEL, the duration of filtered noise and the duration of the signal that is guaranteed to be sampled.

The maximum duration of the noise that is filtered out should not be more than 3 clock cycle, while the minimum duration of the signal that is guaranteed to be sampled should be more than 4 clock cycle.

NFCLKSEL	Maximum Duration Of Noise	Minimum Duration Of Signal
000	42 ns	56 ns
001	83 ns	111 ns
010	167 ns	222 ns
011	667 ns	889 ns
100	1333 ns	1778 ns
101	2667 ns	3556 ns
		PCLK= 72 MHz

Table 6.17-1 Typical Case of Noise Filter Settings

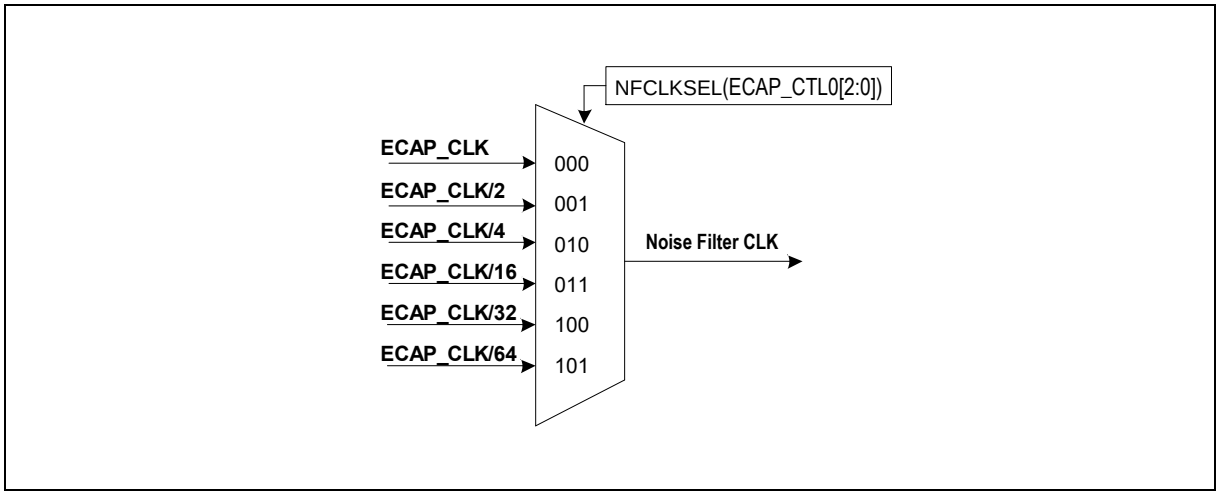


Figure 6.17-3 Noise Filter Sampling Clock Selection

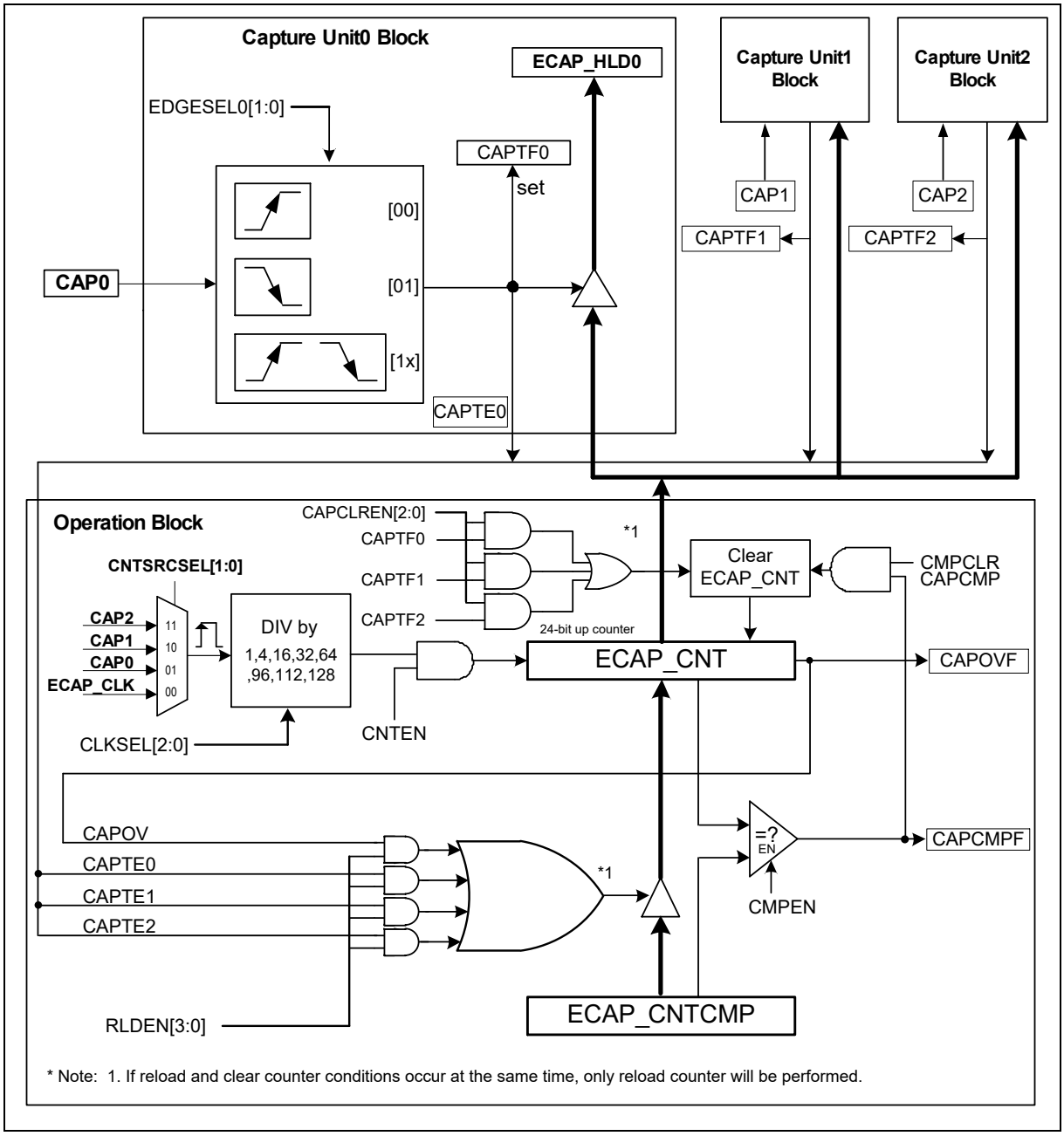


Figure 6.17-4 Input Capture Timer/Counter Function Block

6.17.5.2 Input Capture Timer/Counter Operation

An Input Capture Timer/Counter unit consists of 2 main functional blocks, Capture block and Operation block. There are 3 Input Capture units in Capture block for 3 input channel.

The capture units function as detecting and measuring the pulse width and the period of a square wave. The input channel 0 to 2 have their own edge detectors, which are in Input Capture block but share with one capture timer/counter, ECAP_CNT, which is in Operation block. The edge trigger option is programmable through EDGESEL (ECAP_CTL1[5,4], [3,2], [1,0]) register supporting positive edge, negative edge and both edge triggers. Each capture unit consists of an enable control bit, IC0EN ~ IC2EN (ECAP_CTL0[6:4]) to enable/disable each input channel and a status bit CAP0 ~ CAP2 (ECAP_STATUS[10:8]) to let software monitor the current status of each channel.

The Input Capture supports reload mode and compare mode. For both mode, the capture counter (ECAP_CNT) serves as a 24-bit up-counting counter whose clock comes from the output of the clock divider and is gated with CNTEN, and the clock source of the clock divider, which can be set by CLKSEL[2:0] to divide clock by 1,4,16,32,64,96,112 and 128, is programmable (by setting CNTSRCSEL[1:0]) to be from system clock source, ECAP_CLK or input channel CAP0 ~ CAP2. In reload mode, ECAP_CNTCMP serves as a reload register while in compare mode ECAP_CNTCMP serves as a compare register. The Input Capture Timer/Counter Enable bit (CAPEN) must be set to enable Input Capture Timer/Counter functions. More details of operation are described in the following sections.

Capture Function

Each time when the capture input detect a valid edge change, it triggers a valid capture event (CAPTE0~2) so that the content of the free running 24-bit capture counter ECAP_CNT will be captured/transferred into the capture hold registers, ECAP_HLD0~2 depending on which channel is triggered. This event also causes the corresponding flag CAPTFx (ECAP_STATUS[2:0]) to be set, which will generate an interrupt if the corresponding interrupt enable bit CAPIENx (ECAP_CTL0[18:16]) is set. Triggered Flags are set by hardware and should be cleared by software. Software can read the register ECAP_STATUS to get the status of flags and has to write 1 to the corresponding bit(s) of ECAP_STATUS to clear flag(s).

In addition, setting the CAPxCLREN(ECAP_CTL1[22:20]) will allow hardware to reset capture counter (ECAP_CNT) automatically whenever the event happens.

Compare Mode

The compare function is enabled by setting the CMPEN (ECAP_CTL0[28]) bit to 1, and ECAP_CNTCMP will serve as a compare register. As ECAP_CNT counting up, upon matching ECAP_CNTCMP value, the flag, CAPCMPF (ECAP_STATUS[4]), will be set, which will generate an interrupt, CMP_INT, if compare interrupt enable bit, CMPIEN (ECAP_CTL0[21]), is set.

Besides, setting the CMPCLR (ECAP_CTL0[25]) will allow hardware to make capture counter cleared to 0 automatically after a compare-match event occurs.

Reload Mode

The Input Capture Timer/Counter can also be configured for reload mode. The reload function is enabled by setting the RLDEN[3:0] bits (ECAP_CTL1[11:8]) – OVRLDEN, CAPxRLDEN – to 1, and each bit enables a reload source.

In this mode, ECAP_CNTCMP serves as a reload register. If OVRLDEN is set, a reload event is generated and causes the content of the ECAP_CNTCMP register to be loaded into the ECAP_CNT register when ECAP_CNT overflows. Furthermore, CAPxRLDEN, x=0~2, are enable bits of making CAPTE2 ~ CAPTE0 as reload sources.

One thing should be noted is that if CAPxCLREN as well as CAPxRLDEN are set, when a valid trigger event (CAPTEx) occurs, only RELAOD function will be executed.

6.17.5.3 Input Capture Timer/Counter Interrupt Architecture

Figure 6.17-5 demonstrates the architecture of Input Capture Timer/Counter interrupt module. There are 5 interrupt sources (OVF_INT, CMP_INT, CAPTF0_INT~CAPTF2_INT), which are logical ‘OR’ together, in a input capture unit, and each one has an interrupt flag (CAPOVF, CAPCMPF, CAPTF0~CAPTF2), which can trigger Interrupt (ECAP_INT), as well as an the enable control bit (OVIEN, CMPIEN, CAPIEN0~CAPIEN2) to enable/disable the flag.

Note that all the interrupt flags are set by hardware and must be cleared by software by writing 1 to the bit (ECAP_STATUS[5:4],[2:0]) corresponding to the flag.

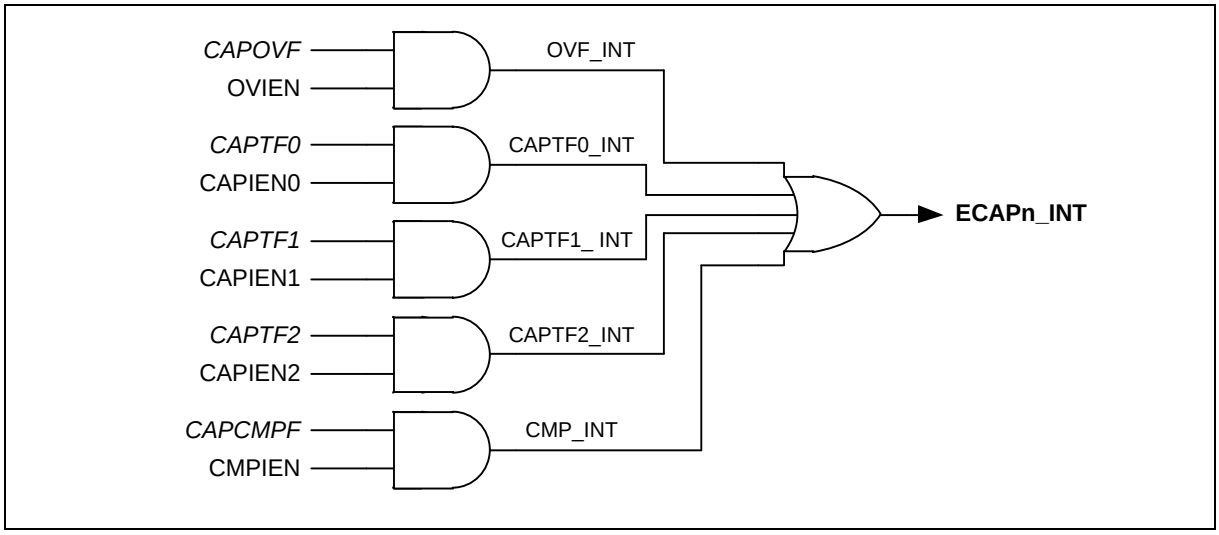


Figure 6.17-5 Input Capture Timer/Counter Interrupt Architecture Diagram

6.17.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
ECAP Base Address: **ECAPn_BA = 0x400B_4000 + (0x0000_1000 * n)** **n=0, 1**				
ECAP_CNT	ECAPn_BA+0x00	R/W	Input Capture Counter (24-bit Up Counter)	0x0000_0000
ECAP_HLD0	ECAPn_BA+0x04	R/W	Input Capture Hold Register 0	0x0000_0000
ECAP_HLD1	ECAPn_BA+0x08	R/W	Input Capture Hold Register 1	0x0000_0000
ECAP_HLD2	ECAPn_BA+0x0C	R/W	Input Capture Hold Register 2	0x0000_0000
ECAP_CNTCMP	ECAPn_BA+0x10	R/W	Input Capture Compare Register	0x0000_0000
ECAP_CTL0	ECAPn_BA+0x14	R/W	Input Capture Control Register 0	0x0000_0000
ECAP_CTL1	ECAPn_BA+0x18	R/W	Input Capture Control Register 1	0x0000_0000
ECAP_STATUS	ECAPn_BA+0x1C	R/W	Input Capture Status Register	0x0000_0000

6.17.7 Register Description

Input Capture Counter (ECAP_CNT)

Register	Offset	R/W	Description	Reset Value
ECAP_CNT	ECAPn_BA+0x00	R/W	Input Capture Counter (24-bit Up Counter)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CNT**	**Input Capture Timer/Counter** The input Capture Timer/Counter is a 24-bit up-counting counter. The clock source for the counter is from the clock divider.

Input Capture Counter Hold Register (ECAP_HLD0~2)

Register	Offset	R/W	Description	Reset Value
ECAP_HLD0	ECAPn_BA+0x04	R/W	Input Capture Hold Register 0	0x0000_0000
ECAP_HLD1	ECAPn_BA+0x08	R/W	Input Capture Hold Register 1	0x0000_0000
ECAP_HLD2	ECAPn_BA+0x0C	R/W	Input Capture Hold Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HOLD							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HOLD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HOLD							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**HOLD**	**Input Capture Counter Hold Register** When an active input capture channel detects a valid edge signal change, the ECAPCNT value is latched into the corresponding holding register. Each input channel has its own holding register named by ECAP_HLDx where x is from 0 to 2 to indicate inputs from IC0 to IC2, respectively.

Input Capture Counter Compare Register (ECAP_CNTCMP)

Register	Offset	R/W	Description	Reset Value
ECAP_CNTCMP	ECAPn_BA+0x10	R/W	Input Capture Compare Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CNTCMP							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNTCMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNTCMP							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CNTCMP**	**Input Capture Counter Compare Register** If the compare function is enabled (CMPEN = 1), this register (ECAP_CNTCMP) is used to compare with the capture counter (ECAP_CNT). If the reload control is enabled (RLDEN[n] = 1, n=0~3), an overflow event or capture events will trigger the hardware to load the value of this register (ECAP_CNTCMP) into ECAP_CNT.

Input Capture Timer/Counter Control Register (ECAP_CTL0)

Register	Offset	R/W	Description	Reset Value
ECAP_CTL0	ECAPn_BA+0x14	R/W	Input Capture Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CAPEN	CMPEN	Reserved		CMPCLREN	CNTEN
23	22	21	20	19	18	17	16
Reserved		CMPIEN	OVIEN	Reserved	CAPIEN2	CAPIEN1	CAPIEN0
15	14	13	12	11	10	9	8
Reserved		CAPSEL2		CAPSEL1		CAPSEL0	
7	6	5	4	3	2	1	0
Reserved	IC2EN	IC1EN	IC0EN	CAPNFDIS	NFCLKSEL		

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**CAPEN**	**Input Capture Timer/Counter Enable Bit** 0 = Input Capture function Disabled. 1 = Input Capture function Enabled.
[28]	**CMPEN**	**Compare Function Enable Bit** The compare function in input capture timer/counter is to compare the dynamic counting ECAP_CNT with the compare register ECAP_CNTCMP, if ECAP_CNT value reaches ECAP_CNTCMP, the flag CAPCMPF will be set. 0 = The compare function Disabled. 1 = The compare function Enabled.
[27:26]	**Reserved**	Reserved.
[25]	**CMPCLREN**	**Input Capture Counter Cleared by Compare-match Control** If this bit is set to 1, the capture counter (ECAP_CNT) will be cleared to 0 when the compare-match event (CAPCMPF = 1) occurs. 0 = Compare-match event (CAPCMPF) can clear capture counter (ECAP_CNT) Disabled. 1 = Compare-match event (CAPCMPF) can clear capture counter (ECAP_CNT) Enabled.
[24]	**CNTEN**	**Input Capture Counter Start Counting Control** Setting this bit to 1, the capture counter (ECAP_CNT) starts up-counting synchronously with the clock from the . 0 = ECAP_CNT stop counting. 1 = ECAP_CNT starts up-counting.
[23:22]	**Reserved**	Reserved.
[21]	**CMPIEN**	**CAPCMPF Trigger Input Capture Interrupt Enable Bit** 0 = The flag CAPCMPF can trigger Input Capture interrupt Disabled. 1 = The flag CAPCMPF can trigger Input Capture interrupt Enabled.

Bits	Description	
[20]	**OVIEN**	**CAPOVF Trigger Input Capture Interrupt Enable Bit** 0 = The flag CAPOVF can trigger Input Capture interrupt Disabled. 1 = The flag CAPOVF can trigger Input Capture interrupt Enabled.
[19]	**Reserved**	Reserved.
[18]	**CAPIEN2**	**Input Capture Channel 2 Interrupt Enable Bit** 0 = The flag CAPTF2 can trigger Input Capture interrupt Disabled. 1 = The flag CAPTF2 can trigger Input Capture interrupt Enabled.
[17]	**CAPIEN1**	**Input Capture Channel 1 Interrupt Enable Bit** 0 = The flag CAPTF1 can trigger Input Capture interrupt Disabled. 1 = The flag CAPTF1 can trigger Input Capture interrupt Enabled.
[16]	**CAPIEN0**	**Input Capture Channel 0 Interrupt Enable Bit** 0 = The flag CAPTF0 can trigger Input Capture interrupt Disabled. 1 = The flag CAPTF0 can trigger Input Capture interrupt Enabled.
[15:14]	**Reserved**	Reserved.
[13:12]	**CAPSEL2**	**CAP2 Input Source Selection** 00 = CAP2 input is from port pin ICAP2. 01 = CAP2 input is from signal ACMP2_O. 10 = CAP2 input is from signal CHX of QEI controller unit n. 11 = Reserved. **Note:** Input capture unit n matches QEIn, where n = 0~1.
[11:10]	**CAPSEL1**	**CAP1 Input Source Selection** 00 = CAP1 input is from port pin ICAP1. 01 = CAP1 input is from signal ACMP1_O. 10 = CAP1 input is from signal CHB of QEI controller unit n. 11 = Reserved. **Note:** Input capture unit n matches QEIn, where n = 0~1.
[9:8]	**CAPSEL0**	**CAP0 Input Source Selection** 00 = CAP0 input is from port pin ICAP0. 01 = CAP0 input is from signal ACMP0_O. 10 = CAP0 input is from signal CHA of QEI controller unit n. 11 = Reserved. **Note:** Input capture unit n matches QEIn, where n = 0~1.
[7]	**Reserved**	Reserved.
[6]	**IC2EN**	**Port Pin IC2 Input to Input Capture Unit Enable Bit** 0 = IC2 input to Input Capture Unit Disabled. 1 = IC2 input to Input Capture Unit Enabled.
[5]	**IC1EN**	**Port Pin IC1 Input to Input Capture Unit Enable Bit** 0 = IC1 input to Input Capture Unit Disabled. 1 = IC1 input to Input Capture Unit Enabled.
[4]	**IC0EN**	**Port Pin IC0 Input to Input Capture Unit Enable Bit** 0 = IC0 input to Input Capture Unit Disabled. 1 = IC0 input to Input Capture Unit Enabled.

Bits	Description	
[3]	**CAPNFDIS**	**Input Capture Noise Filter Disable Bit** 0 = Noise filter of Input Capture Enabled. 1 = Noise filter of Input Capture Disabled (Bypass).
[2:0]	**NFCLKSEL**	**Noise Filter Clock Pre-divide Selection** To determine the sampling frequency of the Noise Filter clock 000 = ECAP_CLK. 001 = ECAP_CLK/2. 010 = ECAP_CLK/4. 011 = ECAP_CLK/16. 100 = ECAP_CLK/32. 101 = ECAP_CLK/64.

Input Capture Timer/Counter Control Register (ECAP_CTL1)

Register	Offset	R/W	Description	Reset Value
ECAP_CTL1	ECAPn_BA+0x18	R/W	Input Capture Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	CAP2CLREN	CAP1CLREN	CAP0CLREN	Reserved		CNTSRCSEL	
15	14	13	12	11	10	9	8
Reserved	CLKSEL			OVRLDEN	CAP2RLDEN	CAP1RLDEN	CAP0RLDEN
7	6	5	4	3	2	1	0
Reserved		EDGESEL2		EDGESEL1		EDGESEL0	

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22]	**CAP2CLREN**	**Capture Counter Cleared by Capture Event2 Control** 0 = Event CAPTE2 can clear capture counter (ECAP_CNT) Disabled. 1 = Event CAPTE2 can clear capture counter (ECAP_CNT) Enabled.
[21]	**CAP1CLREN**	**Capture Counter Cleared by Capture Event1 Control** 0 = Event CAPTE1 can clear capture counter (ECAP_CNT) Disabled. 1 = Event CAPTE1 can clear capture counter (ECAP_CNT) Enabled.
[20]	**CAP0CLREN**	**Capture Counter Cleared by Capture Event0 Control** 0 = Event CAPTE0 can clear capture counter (ECAP_CNT) Disabled. 1 = Event CAPTE0 can clear capture counter (ECAP_CNT) Enabled.
[19:18]	**Reserved**	Reserved.
[17:16]	**CNTSRCSEL**	**Capture Timer/Counter Clock Source Selection** Select the capture timer/counter clock source. 00 = ECAP_CLK (default). 01 = CAP0. 10 = CAP1. 11 = CAP2.
[15]	**Reserved**	Reserved.

Bits	Description	
[14:12]	**CLKSEL**	**Capture Timer Clock Divide Selection** The capture timer clock has a pre-divider with eight divided options controlled by CLKSEL[2:0]. 000 = ECAP_CLK/1. 001 = ECAP_CLK/4. 010 = ECAP_CLK/16. 011 = ECAP_CLK/32. 100 = ECAP_CLK/64. 101 = ECAP_CLK/96. 110 = ECAP_CLK/112. 111 = ECAP_CLK/128.
[11]	**OVRLDEN**	**Capture Counter's Reload Function Triggered by Overflow Enable Bit** 0 = The reload triggered by CAPOV Disabled. 1 = The reload triggered by CAPOV Enabled.
[10]	**CAP2RLDEN**	**Capture Counter's Reload Function Triggered by Event CAPTE2 Enable Bit** 0 = The reload triggered by Event CAPTE2 Disabled. 1 = The reload triggered by Event CAPTE2 Enabled.
[9]	**CAP1RLDEN**	**Capture Counter's Reload Function Triggered by Event CAPTE1 Enable Bit** 0 = The reload triggered by Event CAPTE1 Disabled. 1 = The reload triggered by Event CAPTE1 Enabled.
[8]	**CAP0RLDEN**	**Capture Counter's Reload Function Triggered by Event CAPTE0 Enable Bit** 0 = The reload triggered by Event CAPTE0 Disabled. 1 = The reload triggered by Event CAPTE0 Enabled.
[7:6]	**Reserved**	Reserved.
[5:4]	**EDGESEL2**	**Channel 2 Captured Edge Selection** Input capture2 can detect falling edge change only, rising edge change only or both edge changes 00 = Detect rising edge only. 01 = Detect falling edge only. 1x = Detect both rising and falling edge.
[3:2]	**EDGESEL1**	**Channel 1 Captured Edge Selection** Input capture1 can detect falling edge change only, rising edge change only or both edge change 00 = Detect rising edge only. 01 = Detect falling edge only. 1x = Detect both rising and falling edge.
[1:0]	**EDGESEL0**	**Channel 0 Captured Edge Selection** Input capture0 can detect falling edge change only, rising edge change only or both edge change 00 = Detect rising edge only. 01 = Detect falling edge only. 1x = Detect both rising and falling edge.

Input Capture Timer/Counter Status Register (ECAP_STATUS)

Register	Offset	R/W	Description	Reset Value
ECAP_STATUS	ECAPn_BA+0x1C	R/W	Input Capture Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					CAP2	CAP1	CAP0
7	6	5	4	3	2	1	0
Reserved		CAPOVF	CAPCMPF	Reserved	CAPTF2	CAPTF1	CAPTF0

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**CAP2**	**Value of Input Channel 2, CAP2 (Read Only)** Reflecting the value of input channel 2, CAP2. **Note:** The bit is read only and write is ignored.
[9]	**CAP1**	**Value of Input Channel 1, CAP1 (Read Only)** Reflecting the value of input channel 1, CAP1 **Note:** The bit is read only and write is ignored.
[8]	**CAP0**	**Value of Input Channel 0, CAP0 (Read Only)** Reflecting the value of input channel 0, CAP0 **Note:** The bit is read only and write is ignored.
[7:6]	**Reserved**	Reserved.
[5]	**CAPOVF**	**Input Capture Counter Overflow Flag** Flag is set by hardware when counter (ECAP_CNT) overflows from 0x00FF_FFFF to 0. 0 = No overflow event has occurred since last clear. 1 = Overflow event(s) has/have occurred since last clear. **Note:** This bit is only cleared by writing 1 to it.
[4]	**CAPCMPF**	**Input Capture Compare-match Flag** If the input capture compare function is enabled, the flag is set by hardware when capture counter (ECAP_CNT) up counts and reaches the ECAP_CNTCMP value. 0 = ECAP_CNT has not matched ECAP_CNTCMP value since last clear. 1 = ECAP_CNT has matched ECAP_CNTCMP value at least once since last clear. **Note:** This bit is only cleared by writing 1 to it.
[3]	**Reserved**	Reserved.

Bits	Description	
[2]	**CAPTF2**	**Input Capture Channel 2 Triggered Flag** When the input capture channel 2 detects a valid edge change at CAP2 input, it will set flag CAPTF2 to high. 0 = No valid edge change has been detected at CAP2 input since last clear. 1 = At least a valid edge change has been detected at CAP2 input since last clear. **Note:** This bit is only cleared by writing 1 to it.
[1]	**CAPTF1**	**Input Capture Channel 1 Triggered Flag** When the input capture channel 1 detects a valid edge change at CAP1 input, it will set flag CAPTF1 to high. 0 = No valid edge change has been detected at CAP1 input since last clear. 1 = At least a valid edge change has been detected at CAP1 input since last clear. **Note:** This bit is only cleared by writing 1 to it.
[0]	**CAPTF0**	**Input Capture Channel 0 Triggered Flag** When the input capture channel 0 detects a valid edge change at CAP0 input, it will set flag CAPTF0 to high. 0 = No valid edge change has been detected at CAP0 input since last clear. 1 = At least a valid edge change has been detected at CAP0 input since last clear. **Note:** This bit is only cleared by writing 1 to it.

6.18 Watchdog Timer (WDT)

6.18.1 Overview

The Watchdog Timer (WDT) is used to perform a system reset when system runs into an unknown state. This prevents system from hanging for an infinite period of time. Besides, this Watchdog Timer supports the function to wake up system from Idle/Power-down mode.

6.18.2 Features

- 20-bit free running up counter for WDT time-out interval
- Selectable time-out interval (2^4 ~ 2^{20}) and the time-out interval is 0.5 ms ~ 32.768 s if WDT_CLK = 32 kHz
- System kept in reset state for a period of (1 / WDT_CLK) * 63
- Supports selectable WDT reset delay period, including 1026, 130, 18 or 3 WDT_CLK reset delay period
- Supports to force WDT enabled after chip powered on or reset by setting CWDTEN[2:0] in Config0 register
- Supports WDT time-out wake-up function only if WDT clock source is selected as LIRC or LXT

6.18.3 Block Diagram

The WDT clock control is shown in Figure 6.18-1.

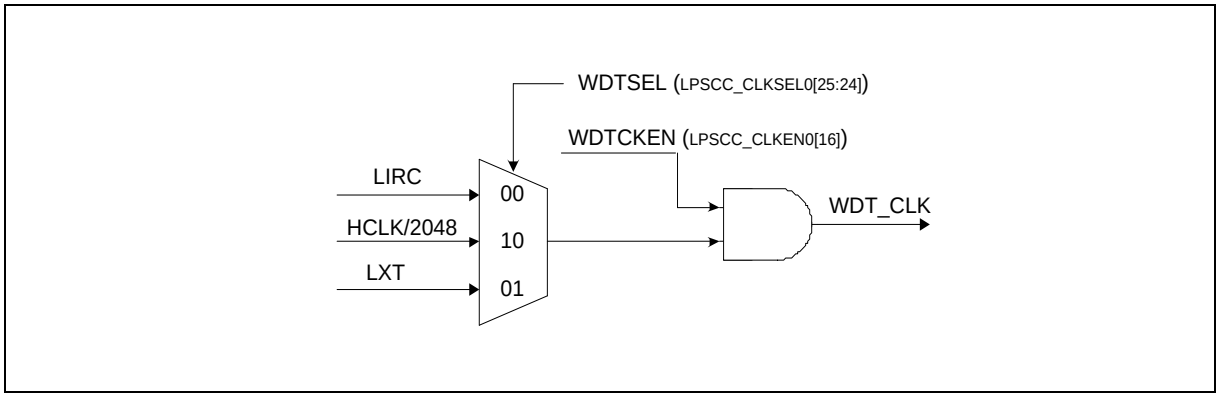
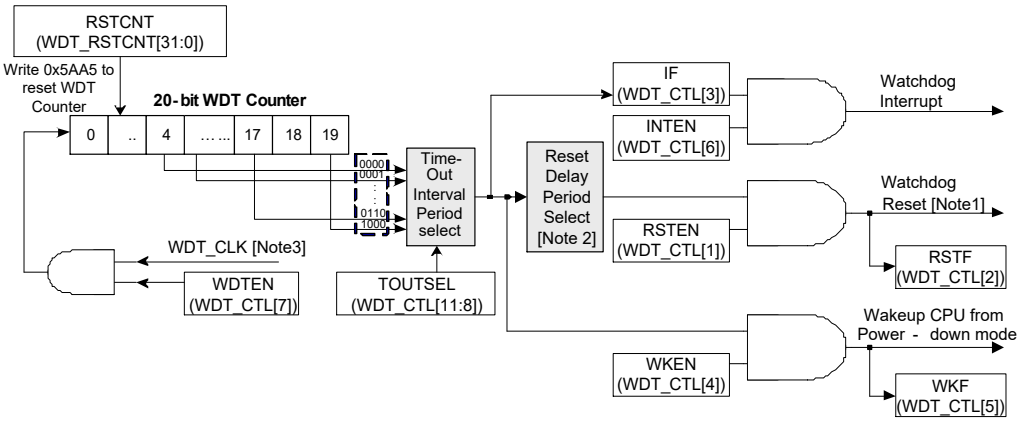


Figure 6.18-1 Watchdog Timer Clock Control



Note1: WDT resets CPU and lasts 63 WDT_CLK.

Note2: The WDT reset delay period can be selected as 3/18/130/1026 WDT_CLK.

Note3: Chip can be woken by WDT time-out interrupt signal generated only if WDT clock source selects LXT or LIRC

Figure 6.18-2 Watchdog Timer Block Diagram

6.18.4 Basic Configuration

- Clock Source Configuration
  - Select the source of WDT peripheral clock on WDTSEL (LPSCC_CLKSEL0[25:24])
  - Enable WDT peripheral clock in WDTCKEN (LPSCC_CLKEN0[16])
  - Force to enable WDT controller after chip powered on or reset in CWDTEN[2:0] (CWDTEN[2] is Config0[31], CWDTEN[1:0] is Config0[4:3]) while CWDTRN[2:0] is not configured to 111
- User Configuration

CFG_cwdten CONFIG0[31][4][3]	LPSCC_CLKEN0[16]	WDT_CLK
111	0	Gated
111	1	WDT_CLK can select LIRC or LXT or HCLK1/2048 • Normal mode: present • Power-down mode: present if LIRC or LXT is selected , else gated
Others	x	WDT_CLK forced as LIRC and is always on

Table 6.18-1 User Configuration Setting Table

6.18.5 Functional Description

The WDT includes a 20-bit free running up counter with programmable time-out intervals. Table 6.18-2 shows the WDT time-out interval period selection and Figure 6.18-3 shows the WDT time-out interval and reset period timing.

6.18.5.1 WDT Time-out Interrupt

Setting WDTEN (WDT_CTL[7]) to 1 will enable the WDT function and the WDT counter to start counting up. The SYNC (WDT_CTL[30]) can be indicated whether enable/disable WDTEN function is completed or not. There are nine time-out interval period can be selected by setting TOUTSEL (WDT_CTL[11:8]). When the WDT up counter reaches the TOUTSEL (WDT_CTL[11:8]) settings, WDT time-out interrupt will occur then WDT time-out interrupt flag IF (WDT_CTL[3]) will be set to 1 immediately. If INTEN (WDT_CTL[6]) is enabled, WDT time-out interrupt will inform CPU.

6.18.5.2 WDT Reset Delay Period and Reset System

There is a specified T_{RSTD} reset delay period follows the IF (WDT_CTL[3]) is setting to 1. User should program 0x00005AA5 to RSTCNT (WDT_RSTCNT[31:0]) to reset the 20-bit WDT up counter value to avoid generate WDT time-out reset signal before the T_{RSTD} reset delay period expires. Moreover, user should set RSTDSEL (WDT_ALTCTL[1:0]) to select reset delay period to clear WDT counter. If the WDT up counter value has not been cleared after the specific T_{RSTD} delay period expires, the WDT control will set RSTF (WDT_CTL[2]) to 1 if RSTEN (WDT_CTL[1]) bit is enabled, then chip enters to reset state immediately. T_{RST} reset period will keep last 63 WDT clocks and then chip restarts executing program from reset vector (0x0000_0000). The RSTF (WDT_CTL[2]) will keep 1 after WDT time-out resets the chip, user can check RSTF (WDT_CTL[2]) by software to recognize the system has been reset by WDT time-out reset or not.

TOUTSEL	Time-Out Interval Period T_{TIS}	Reset Delay Period T_{RSTD}
0000	$2^4 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0001	$2^6 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0010	$2^8 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0011	$2^{10} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0100	$2^{12} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0101	$2^{14} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0110	$2^{16} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0111	$2^{18} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
1000	$2^{20} * T_{WDT}$	(3/18/130/1026) * T_{WDT}

Table 6.18-2 Watchdog Timer Time-out Interval Period Selection

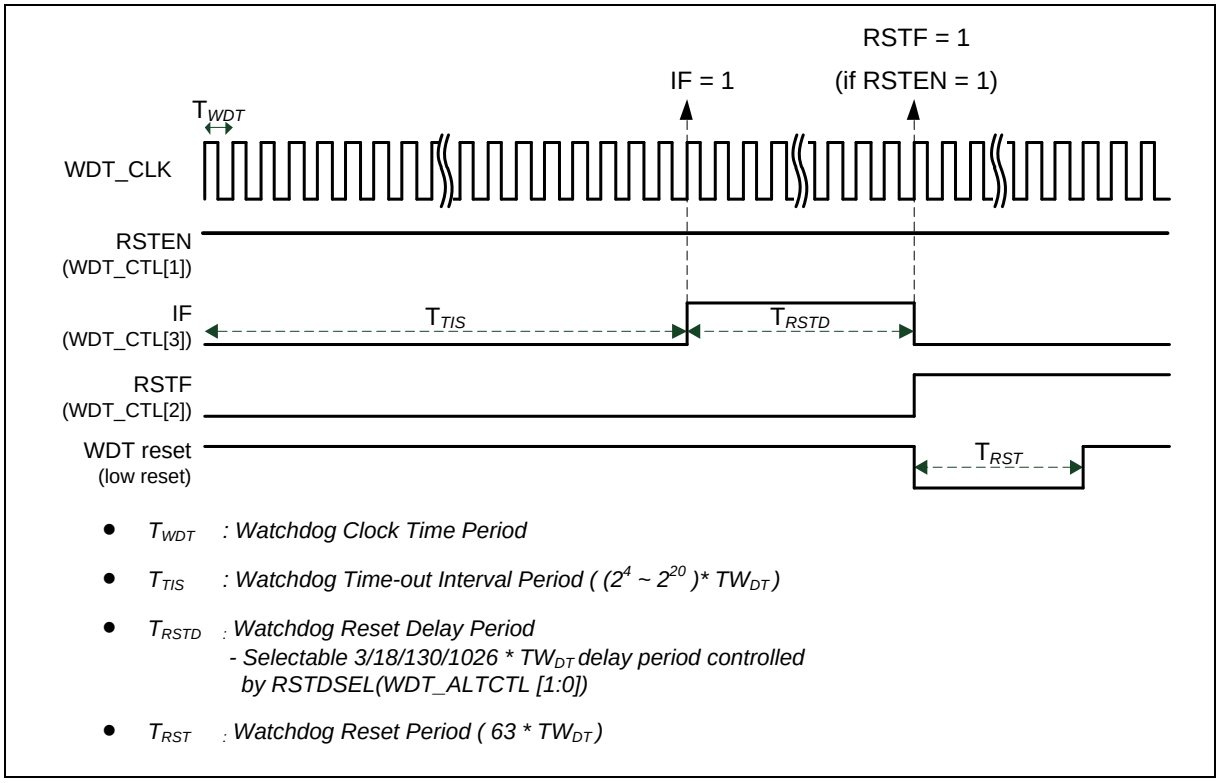


Figure 6.18-3 Watchdog Timer Time-out Interval and Reset Period Timing

6.18.5.3 WDT Wake-up

If WDT clock source is selected to LIRC or LXT, system can be woken up from Power-down mode while WDT time-out interrupt signal is generated and WKEN (WDT_CTL[4]) enabled. Note that user should set LXTEN (CLK_PWRCTL[1]) or LIRCEN (CLK_PWRCTL[3]) to select clock source before system enters Power-down mode because the system peripheral clocks are disabled when system is in Power-down mode. In the meanwhile, the WKF (WDT_CTL[5]) will be set to 1 automatically, and user can check WKF (WDT_CTL[5]) status by software to recognize the system has been woken up by WDT time-out interrupt or not.

6.18.5.4 WDT ICE Debug

When ICE is connected to MCU, WDT counter is counting or not by ICEDEBUG (WDT_CTL[31]). The default value of ICEDEBUG is 0, and WDT counter will stop counting when CPU is held by ICE. If ICEDEBUG is set to 1, WDT counter will keep counting no matter CPU is held by ICE or not.

6.18.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
WDT Base Address: **WDT_BA = 0x4004_0000**				
WDT_CTL	WDT_BA+0x00	R/W	WDT Control Register	0x0000_0800
WDT_ALTCTL	WDT_BA+0x04	R/W	WDT Alternative Control Register	0x0000_0000
WDT_RSTCNT	WDT_BA+0x08	W	WDT Reset Counter Register	0x0000_0000

6.18.7 Register Description

WDT Control Register (WDT_CTL)

Register	Offset	R/W	Description	Reset Value
WDT_CTL	WDT_BA+0x00	R/W	WDT Control Register	0x0000_0800

31	30	29	28	27	26	25	24
ICEDEBUG	SYNC	PDRSTCNT	Reserved				
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				TOUTSEL			
7	6	5	4	3	2	1	0
WDTEN	INTEN	WKF	WKEN	IF	RSTF	RSTEN	Reserved

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 0 = ICE debug mode acknowledgement affects WDT counting. WDT up counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. WDT up counter will keep going no matter CPU is held by ICE or not. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[30]	**SYNC**	**WDT Enable Control SYNC Flag Indicator (Read Only)** If user executes enable/disable WDTEN (WDT_CTL[7]), this flag can be indicated enable/disable WDTEN function is completed or not. 0 = Setting WDTEN bit is completed and WDT is ready. 1 = Setting WDTEN bit is synchronizing and not become active yet. **Note:** Performing enable or disable WDTEN bit needs 4 * WDT_CLK period to become active.
[29]	**PDRSTCNT**	**Reset Counter When Entering Power Down Enable Bit** 0 = WDT up counter will keep going no matter CPU enter power down or not. 1 = Reset WDT up counter value to 0 when entering power down.
[28:12]	**Reserved**	Reserved.
[11:8]	**TOUTSEL**	**WDT Time-out Interval Selection (Write Protect)** These four bits select the time-out interval period for the WDT. 0000 = 2^4 * WDT_CLK. 0001 = 2^6 * WDT_CLK. 0010 = 2^8 * WDT_CLK. 0011 = 2^{10} * WDT_CLK. 0100 = 2^{12} * WDT_CLK. 0101 = 2^{14} * WDT_CLK. 0110 = 2^{16} * WDT_CLK. 0111 = 2^{18} * WDT_CLK.

		1000 = 2^{20} * WDT_CLK. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7]	**WDTEN**	**WDT Enable Bit (Write Protect)** 0 = WDT Disabled (This action will reset the internal up counter value). 1 = WDT Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** If CWDTEN[2:0] (combined by Config0[31] and Config0[4:3]) bits is not configured to 111, this bit is forced as 1 and user cannot change this bit to 0.
[6]	**INTEN**	**WDT Time-out Interrupt Enable Bit (Write Protect)** If this bit is enabled, the WDT time-out interrupt signal is generated and inform to CPU. 0 = WDT time-out interrupt Disabled. 1 = WDT time-out interrupt Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5]	**WKF**	**WDT Time-out Wake-up Flag** This bit indicates the interrupt wake-up flag status of WDT 0 = WDT does not cause chip wake-up. 1 = Chip wake-up from Idle or Power-down mode if WDT time-out interrupt signal generated. **Note:** This bit is cleared by writing 1 to it.
[4]	**WKEN**	**WDT Time-out Wake-up Function Control (Write Protect)** If this bit is set to 1, while WDT time-out interrupt flag IF (WDT_CTL[3]) is generated to 1 and interrupt enable bit INTEN (WDT_CTL[6]) is enabled, the WDT time-out interrupt signal will generate a wake-up trigger event to chip. 0 = Wake-up trigger event Disabled if WDT time-out interrupt signal generated. 1 = Wake-up trigger event Enabled if WDT time-out interrupt signal generated. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** Chip can be woken up by WDT time-out interrupt signal generated only if WDT clock source is selected to internal low speed RC oscillator (LIRC) or LXT.
[3]	**IF**	**WDT Time-out Interrupt Flag** This bit will set to 1 while WDT up counter value reaches the selected WDT time-out interval. 0 = WDT time-out interrupt did not occur. 1 = WDT time-out interrupt occurred. **Note:** This bit is cleared by writing 1 to it.
[2]	**RSTF**	**WDT Time-out Reset Flag** This bit indicates the system has been reset by WDT time-out reset or not. 0 = WDT time-out reset did not occur. 1 = WDT time-out reset occurred. **Note:** This bit is cleared by writing 1 to it.
[1]	**RSTEN**	**WDT Time-out Reset Enable Bit (Write Protect)** Setting this bit will enable the WDT time-out reset function If the WDT up counter value has not been cleared after the specific WDT reset delay period expires. 0 = WDT time-out reset function Disabled. 1 = WDT time-out reset function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**Reserved**	Reserved.

WDT Alternative Control Register (WDT_ALTCTL)

Register	Offset	R/W	Description	Reset Value
WDT_ALTCTL	WDT_BA+0x04	R/W	WDT Alternative Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**RSTDSEL**	

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1:0]	**RSTDSEL**	**WDT Reset Delay Selection (Write Protect)** When WDT time-out happened, user has a time named WDT Reset Delay Period to clear WDT counter by programming 0x5AA5 to prevent WDT time-out reset happened. User can select a suitable setting of RSTDSEL for different WDT Reset Delay Period. 00 = WDT Reset Delay Period is 1026 * WDT_CLK. 01 = WDT Reset Delay Period is 130 * WDT_CLK. 10 = WDT Reset Delay Period is 18 * WDT_CLK. 11 = WDT Reset Delay Period is 3 * WDT_CLK. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** This register will be reset to 0 if WDT time-out reset happened.

WDT Reset Counter Register (WDT_RSTCNT)

Register	Offset	R/W	Description	Reset Value
WDT_RSTCNT	WDT_BA+0x08	W	WDT Reset Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
RSTCNT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RSTCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RSTCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RSTCNT							

Bits	Description	
[31:0]	**RSTCNT**	**WDT Reset Counter Register** Writing 0x00005AA5 to this field will reset the internal 20-bit WDT up counter value to 0. **Note:** Performing RSTCNT to reset counter needs 2 * WDT_CLK period to become active.

6.19 Window Watchdog Timer (WWDT)

6.19.1 Overview

The Window Watchdog Timer (WWDT) is used to perform a system reset within a specified window period to prevent software running to uncontrollable status by any unpredictable condition.

6.19.2 Features

- 6-bit down counter value (CNTDAT, WWDT_CNT[5:0]) and 6-bit compare value (CMPDAT, WWDT_CTL[21:16]) to make the WWDT time-out window period flexible
- Supports 4-bit value (PSCSEL, WWDT_CTL[11:8]) to programmable maximum 11-bit prescale counter period of WWDT counter
- WWDT counter suspends in Idle/Power-down mode

6.19.3 Block Diagram

The WWDT clock control is shown in Figure 6.19-1.

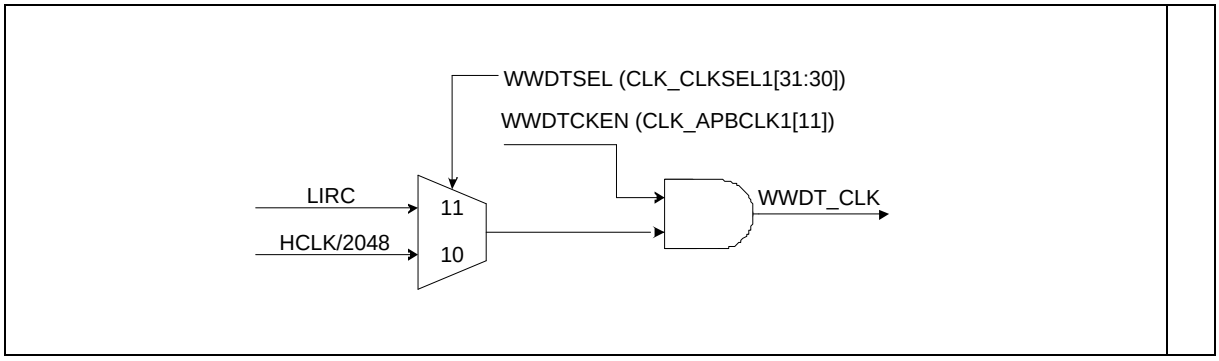


Figure 6.19-1 WWDT Clock Control

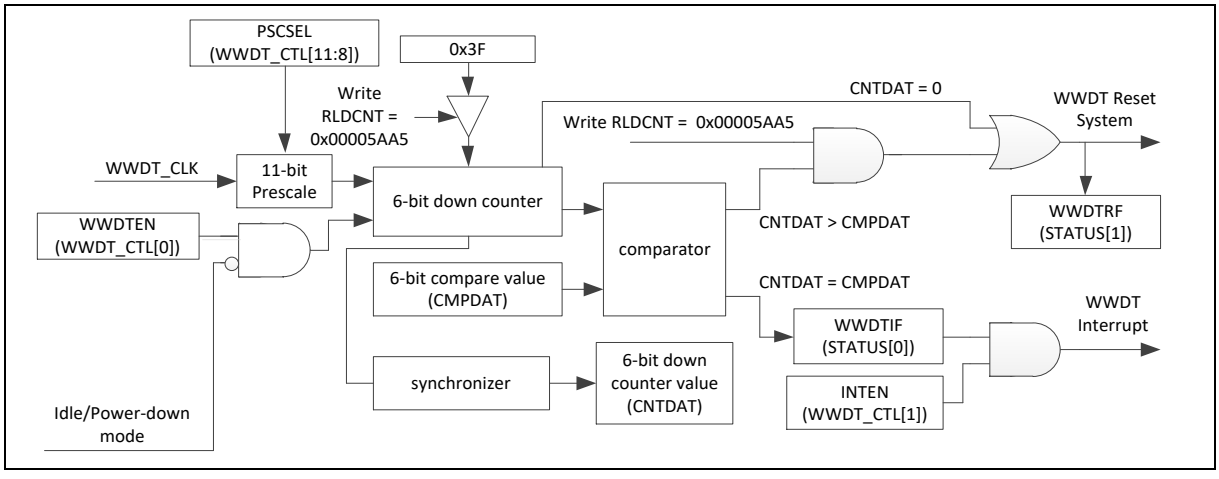


Figure 6.19-2 WWDT Block Diagram

6.19.4 Basic Configuration

- Clock Source Configuration
  - Select the source of WWDT peripheral clock on WWDTSEL (CLK_CLKSEL1[31:30])
  - Enable WWDT peripheral clock in WWDTCKEN (CLK_APBCLK1[11]).

6.19.5 Functional Description

The WWDT includes a 6-bit down counter with programmable prescale value to define different WWDT time-out intervals. The clock source of 6-bit WWDT is based on system clock divide 2048 (HCLK/2048) or internal low speed RC oscillator (LIRC) with a programmable 11-bit prescale counter value which controlled by PSCSEL (WWDT_CTL[11:8]). Also, the correlate of PSCSEL (WWDT_CTL[11:8]) and prescale value are listed in Table 6.19-1.

PSCSEL	Prescale Value	Max. Time-Out Period	Max. Time-Out Interval (WWDT_CLK= 32 kHz)
0000	1	1 * 64 * T_{WWDT}	1.95 ms
0001	2	2 * 64 * T_{WWDT}	3.90 ms
0010	4	4 * 64 * T_{WWDT}	7.8125 ms
0011	8	8 * 64 * T_{WWDT}	15.625 ms
0100	16	16 * 64 * T_{WWDT}	31.25 ms
0101	32	32 * 64 * T_{WWDT}	62.5 ms
0110	64	64 * 64 * T_{WWDT}	125 ms
0111	128	128 * 64 * T_{WWDT}	250 ms
1000	192	192 * 64 * T_{WWDT}	375 ms
1001	256	256 * 64 * T_{WWDT}	500 ms
1010	384	384 * 64 * T_{WWDT}	750 ms
1011	512	512 * 64 * T_{WWDT}	1000 ms
1100	768	768 * 64 * T_{WWDT}	1500 ms
1101	1024	1024 * 64 * T_{WWDT}	2000 ms
1110	1536	1536 * 64 * T_{WWDT}	3000 ms
1111	2048	2048 * 64 * T_{WWDT}	4000 ms

Table 6.19-1 WWDT Prescale Value Selection

6.19.5.1 WWDT Counting

When the WWDTEN (WWDT_CTL[0]) is set, WWDT down counter will start counting from 0x3F to 0. To prevent program runs to disable WWDT counter counting unexpected, the WWDT_CTL register can only be written once after chip is powered on or reset. User cannot disable WWDT counter counting (WWDTEN), change counter prescale period (PSCSEL) or change window compare value (CMPDAT) while WWDTEN (WWDT_CTL[0]) has been enabled by user unless chip is reset.

To avoid the system is reset while CPU clock is disabled, the WWDT counter will stop counting when CPU enters Idle/Power-down mode. After CPU enters normal mode, the WWDT counter will start down counting.

6.19.5.2 WWDT Compare Match Interrupt

During down counting by the WWDT counter, the WWDTIF (WWDT_STATUS[0]) is set to 1 while the WWDT counter value (CNTDAT) is equal to window compare value (CMPDAT) and WWDTIF can be cleared by user; if INTEN (WWDT_CTL[1]) is also set to 1 by user, the WWDT compare match interrupt signal is generated also while WWDTIF is set to 1 by hardware.

6.19.5.3 WWDT Reset System

Figure 6.19-3 shows three cases of WWDT reset and reload behavior.

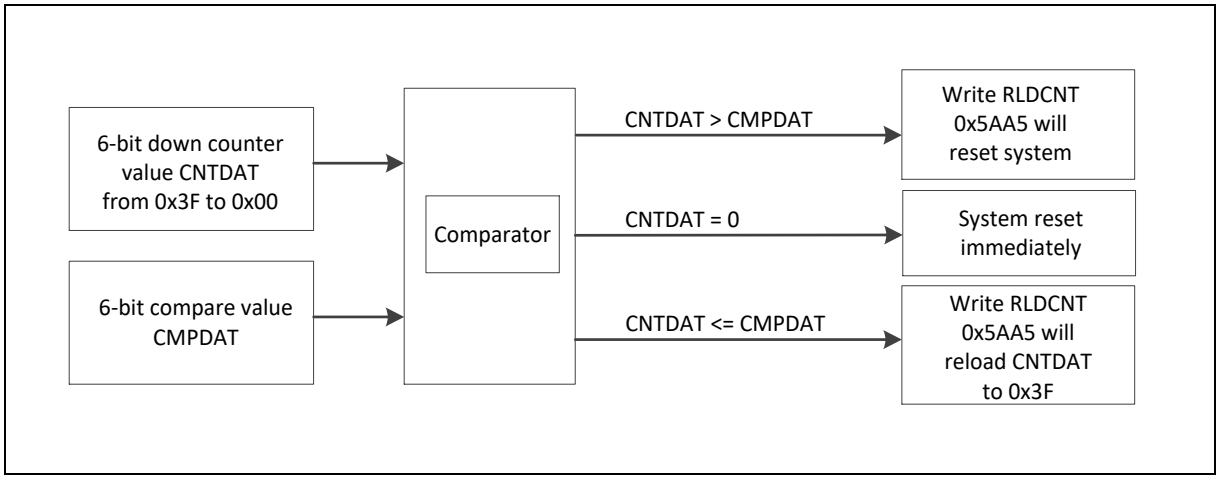


Figure 6.19-3 WWDT Reset and Reload Behavior

If the current CNTDAT (WWDT_CNT[5:0]) is larger than CMPDAT (WWDT_CTL[21:16]) and user writes 0x00005AA5 to the WWDT_RLDCNT register, the WWDT reset system signal will be generated immediately to cause chip reset also. The waveform of WWDT reload counter when CNTDAT > CMPDAT is shown in Figure 6.19-4.

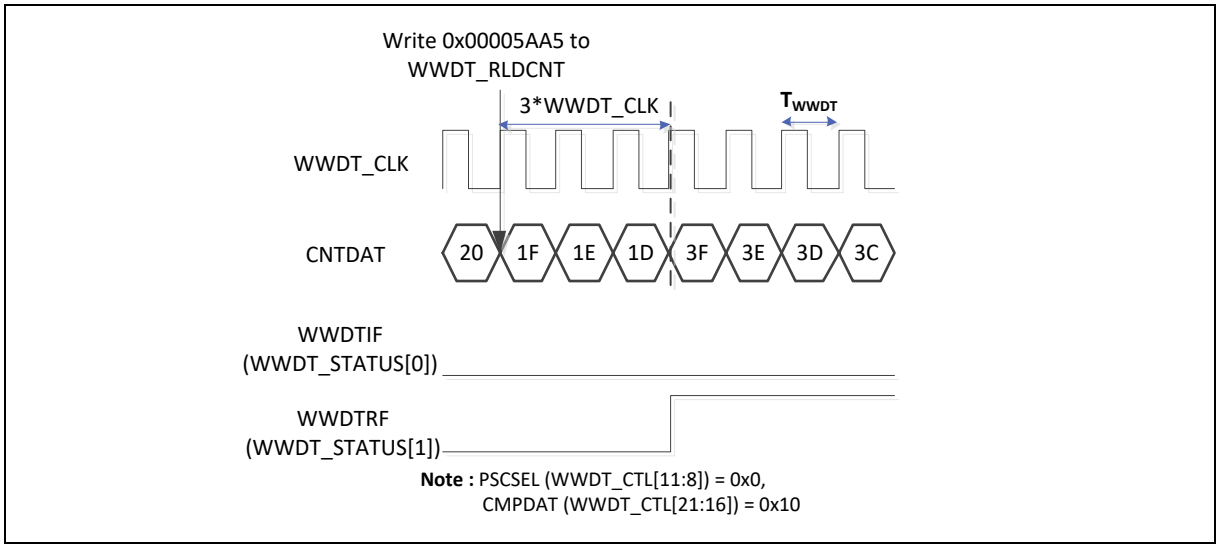


Figure 6.19-4 WWDT Reload Counter When CNTDAT > CMPDAT

When WWDTIF (WWDT_STATUS[0]) is generated, user must reload WWDT counter value to 0x3F by writing 0x00005AA5 to WWDT_RLDCNT register, and also to prevent the WWDT counter value from reaching 0 and generate WWDT reset system signal to inform system reset. Figure 6.19-5 shows the waveform of WWDT reload counter when CNTDAT is less than CMPDAT and Figure 6.19-6 shows the WWDT generates reset system signal (WWDTRF) if user does not write 0x00005AA5 to WWDT_RLD before WWDT counter value reaches 0.

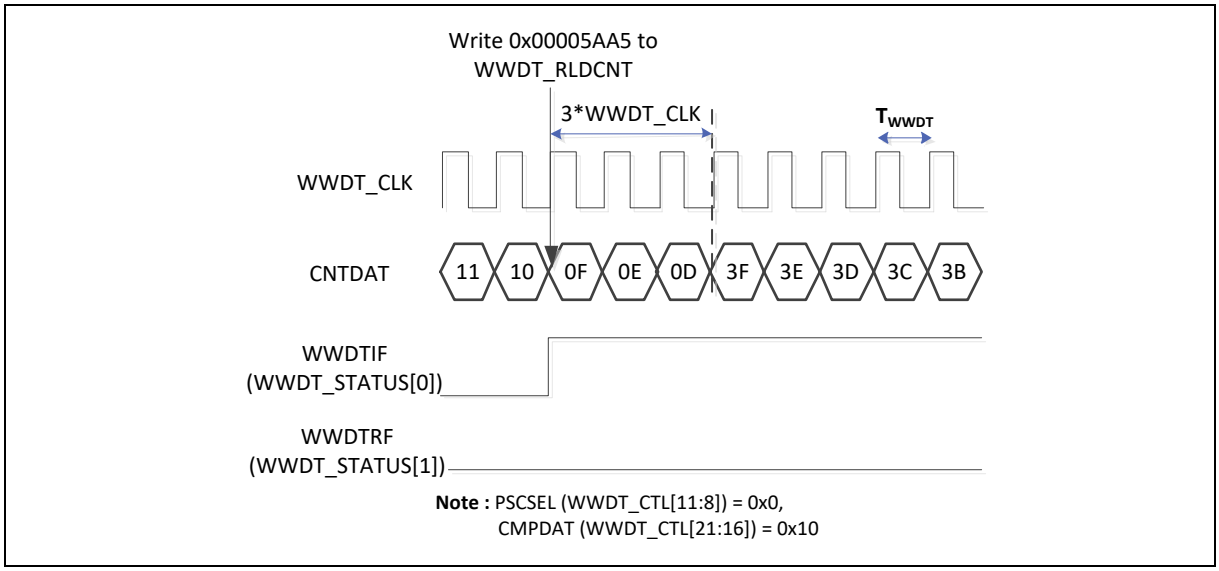


Figure 6.19-5 WWDT Reload Counter When CNTDAT <= CMPDAT

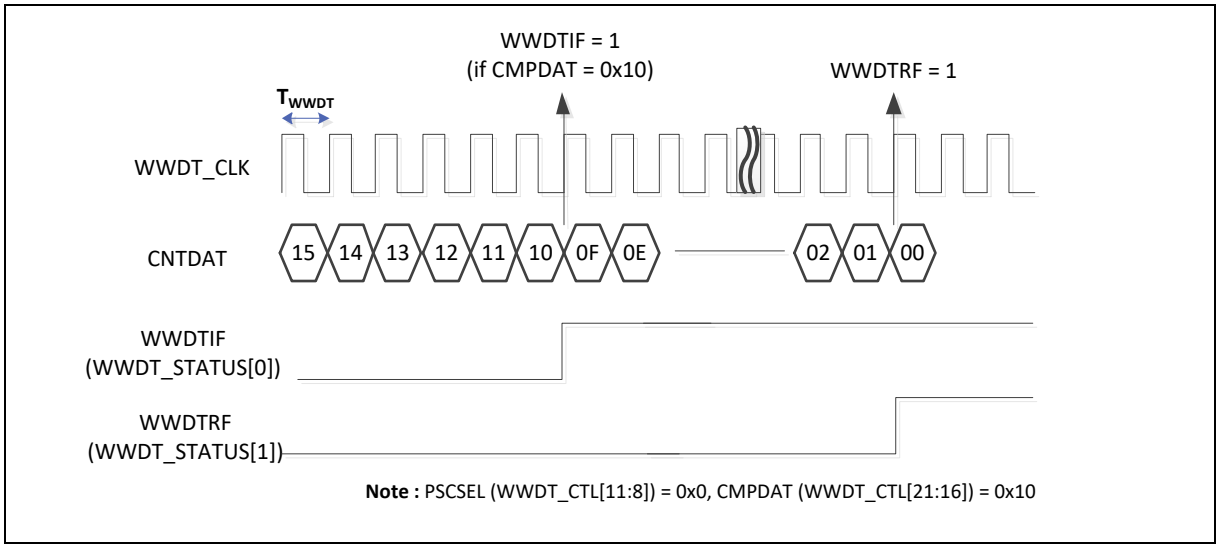


Figure 6.19-6 WWDT Interrupt and Reset Signals

6.19.5.4 WWDT Window Setting Limitation

When user writes 0x00005AA5 to WWDT_RLDCNT register to reload WWDT counter value to 0x3F, it needs 3 WWDT clocks to sync the reload command to actually perform reload action. Note that if user sets PSCSEL (WWDT_CTL[11:8]) to 0000, the counter prescale value should be as 1, and the CMPDAT (WWDT_CTL[21:16]) must be larger than 2. Otherwise, writing WWDT_RLDCNT register to reload WWDT counter value to 0x3F is unavailable, WWDTIF (WWDT_STATUS[0]) is generated, and WWDT reset system event always happened. The WWDT CMPDAT setting limitation is shown in Table 6.19-2.

If user sets CMPDAT as 0x3F and 0x0, the interrupt doesn't occur. The reset occurs when WWDT counts to 0x0, so the interrupt doesn't occur when CMPDAT is 0x0.

PSCSEL	Prescale Value	Valid CMPDAT Value
0000	1	0x3 ~ 0x3E
0001	2	0x2 ~ 0x3E
Others	Others	0x1 ~ 0x3E

Table 6.19-2 CMPDAT Setting Limitation

6.19.5.5 WWDT ICE Debug

When ICE is connected to MCU, the WWDT counter is counting (or not) by ICEDEBUG (WWDT_CTL[31]). The default value of ICEDEBUG is 0. The WWDT counter will stop counting when CPU is held by ICE. If ICEDEBUG is set to 1, WWDT counter will keep counting no matter CPU is held by ICE or not.

6.19.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
WWDT Base Address: **WWDT_BA = 0x4009_6000**				
WWDT_RLDCNT	WWDT_BA+0x00	W	WWDT Reload Counter Register	0x0000_0000
WWDT_CTL	WWDT_BA+0x04	R/W	WWDT Control Register	0x003F_0800
WWDT_STATUS	WWDT_BA+0x08	R/W	WWDT Status Register	0x0000_0000
WWDT_CNT	WWDT_BA+0x0C	R	WWDT Counter Value Register	0x0000_003F

6.19.7 Register Description

WWDT Reload Counter Register (WWDT_RLDCNT)

Register	Offset	R/W	Description	Reset Value
WWDT_RLDCNT	WWDT_BA+0x00	W	WWDT Reload Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
RLDCNT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RLDCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RLDCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RLDCNT							

Bits	Description	
[31:0]	**RLDCNT**	**WWDT Reload Counter Register** Writing 0x00005AA5 to this register will reload the WWDT counter value to 0x3F. **Note:** User can only write WWDT_RLDCNT register to reload WWDT counter value when current WWDT counter value between 0 and CMPDAT (WWDT_CTL[21:16]). If user writes WWDT_RLDCNT when current WWDT counter value is larger than CMPDAT, WWDT reset signal will be generated immediately.

WWDT Control Register (WWDT_CTL)

Register	Offset	R/W	Description	Reset Value
WWDT_CTL	WWDT_BA+0x04	R/W	WWDT Control Register	0x003F_0800

Note: This register can be written only one time after chip is powered on or reset.

31	30	29	28	27	26	25	24
ICEDEBUG	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**CMPDAT**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**PSCSEL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**INTEN**	**WWDTEN**

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Bit** 0 = ICE debug mode acknowledgement effects WWDT counting. WWDT down counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. **Note:** WWDT down counter will keep going no matter CPU is held by ICE or not.
[30:22]	**Reserved**	Reserved.
[21:16]	**CMPDAT**	**WWDT Window Compare** Set this register to adjust the valid reload window. **Note:** User can only write WWDT_RLDCNT register to reload WWDT counter value when current WWDT counter value between 0 and CMPDAT. If user writes WWDT_RLDCNT register when current WWDT counter value larger than CMPDAT, WWDT reset signal will generate immediately.
[15:12]	**Reserved**	Reserved.
[11:8]	**PSCSEL**	**WWDT Counter Prescale Period Selection** 0000 = Pre-scale is 1; Max time-out period is 1 * 64 * WWDT_CLK. 0001 = Pre-scale is 2; Max time-out period is 2 * 64 * WWDT_CLK. 0010 = Pre-scale is 4; Max time-out period is 4 * 64 * WWDT_CLK. 0011 = Pre-scale is 8; Max time-out period is 8 * 64 * WWDT_CLK. 0100 = Pre-scale is 16; Max time-out period is 16 * 64 * WWDT_CLK. 0101 = Pre-scale is 32; Max time-out period is 32 * 64 * WWDT_CLK. 0110 = Pre-scale is 64; Max time-out period is 64 * 64 * WWDT_CLK. 0111 = Pre-scale is 128; Max time-out period is 128 * 64 * WWDT_CLK. 1000 = Pre-scale is 192; Max time-out period is 192 * 64 * WWDT_CLK. 1001 = Pre-scale is 256; Max time-out period is 256 * 64 * WWDT_CLK. 1010 = Pre-scale is 384; Max time-out period is 384 * 64 * WWDT_CLK. 1011 = Pre-scale is 512; Max time-out period is 512 * 64 * WWDT_CLK. 1100 = Pre-scale is 768; Max time-out period is 768 * 64 * WWDT_CLK.

		1101 = Pre-scale is 1024; Max time-out period is 1024 * 64 * WWDT_CLK. 1110 = Pre-scale is 1536; Max time-out period is 1536 * 64 * WWDT_CLK. 1111 = Pre-scale is 2048; Max time-out period is 2048 * 64 * WWDT_CLK.
[7:2]	**Reserved**	Reserved.
[1]	**INTEN**	**WWDT Interrupt Enable Bit** If this bit is enabled, the WWDT counter compare match interrupt signal is generated and inform to CPU. 0 = WWDT counter compare match interrupt Disabled. 1 = WWDT counter compare match interrupt Enabled.
[0]	**WWDTEN**	**WWDT Enable Bit** 0 = WWDT counter is stopped. 1 = WWDT counter starts counting.

WWDT Status Register (WWDT_STATUS)

Register	Offset	R/W	Description	Reset Value
WWDT_STATUS	WWDT_BA+0x08	R/W	WWDT Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**WWDTRF**	**WWDTIF**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WWDTRF**	**WWDT Timer-out Reset Flag** This bit indicates the system has been reset by WWDT time-out reset or not. 0 = WWDT time-out reset did not occur. 1 = WWDT time-out reset occurred. **Note:** This bit is cleared by writing 1 to it.
[0]	**WWDTIF**	**WWDT Compare Match Interrupt Flag** This bit indicates the interrupt flag status of WWDT while WWDT counter value matches CMPDAT (WWDT_CTL[21:16]). 0 = No effect. 1 = WWDT counter value matches CMPDAT. **Note:** This bit is cleared by writing 1 to it.

WWDT Counter Value Register (WWDT_CNT)

Register	Offset	R/W	Description	Reset Value
WWDT_CNT	WWDT_BA+0x0C	R	WWDT Counter Value Register	0x0000_003F

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CNTDAT**					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**CNTDAT**	**WWDT Counter Value** CNTDAT will be updated continuously to monitor 6-bit WWDT down counter value.

6.20 Real Time Clock (RTC)

6.20.1 Overview

The Real Time Clock (RTC) controller provides the real time and calendar message. The RTC offers programmable time tick and alarm match interrupts. The data format of time and calendar messages are expressed in BCD format. A digital frequency compensation feature is available to compensate external crystal oscillator frequency accuracy.

6.20.2 Features

- Supports external power pin V_{BAT}.
- Supports real time counter in RTC_TIME (hour, minute, second) and calendar counter in RTC_CAL (year, month, day) for RTC time and calendar check.
- Supports alarm time (hour, minute, second) and calendar (year, month, day) settings in RTC_TALM and RTC_CALM.
- Supports alarm time (hour, minute, second) and calendar (year, month, day) mask enable in RTC_TAMSK and RTC_CAMSK.
- Selectable 12-hour or 24-hour time scale in RTC_CLKFMT register.
- Optional support 1/128 second HZCNT in RTC_TIME and RTC_TALM.
- Supports Leap Year indication in RTC_LEAPYEAR register.
- Supports Day of the Week counter in RTC_WEEKDAY register.
- Frequency of RTC clock source compensate by RTC_FREQADJ register.
- All time and calendar message expressed in BCD format.
- Supports periodic RTC Time Tick interrupt with 8 period interval options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second.
- Supports RTC Time Tick and Alarm Match interrupt.
- Supports 1 Hz clock output.
- Supports chip wake-up from Idle or Power-down mode while a RTC interrupt signal is generated.
- Supports Daylight Saving Time software control in RTC_DSTCTL.
- Supports up to 3 individual tamper pins.
- Supports 20 bytes spare registers and tamper pins detection to clear the content of these spare registers.
- Supports RRAM mass erase operate will also clear the 20 bytes spare registers content.

6.20.3 Block Diagram

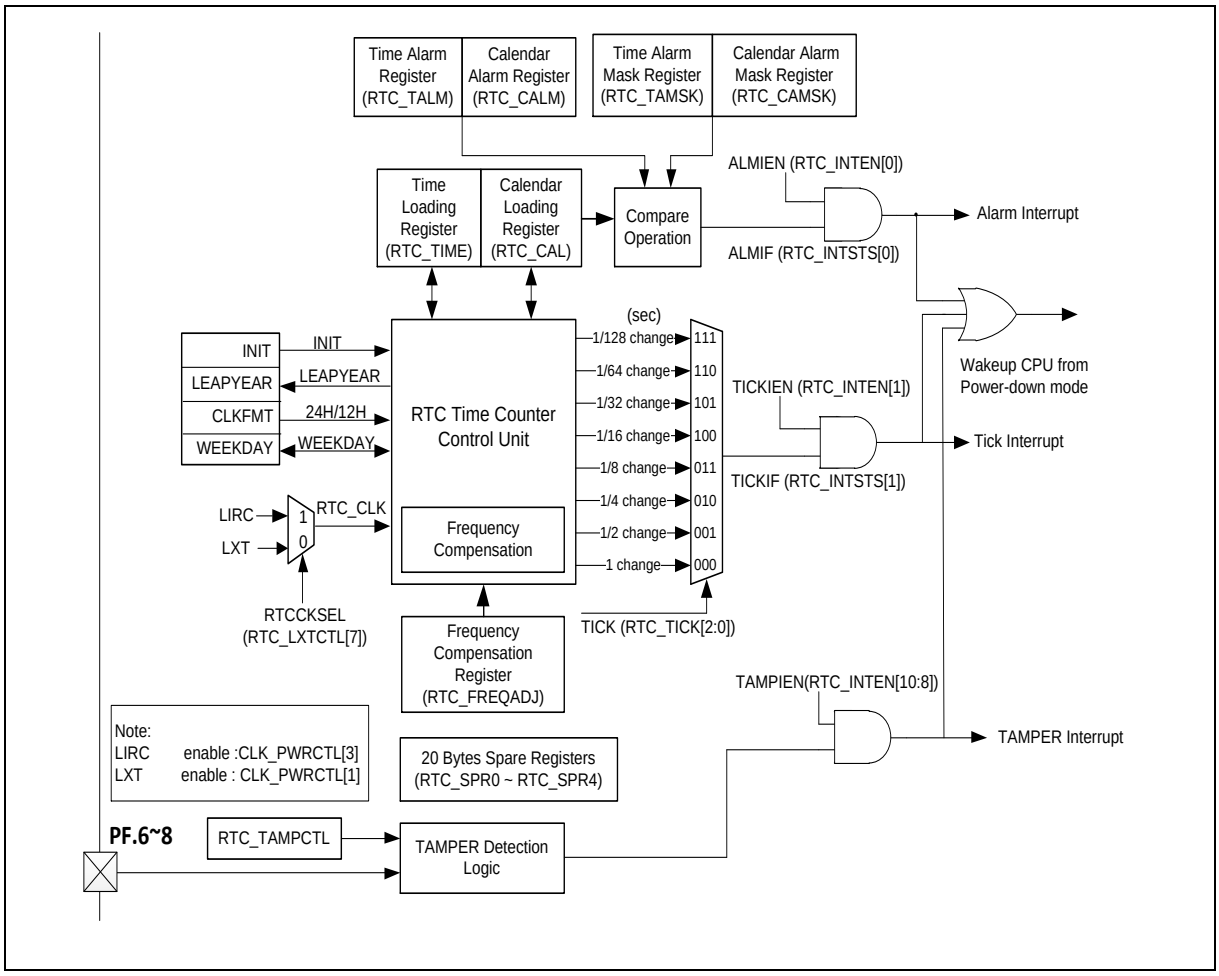


Figure 6.20-1 RTC Block Diagram

6.20.4 Basic Configuration

- Clock Source Configuration
  - The RTC controller clock source is enabled by RTCCKEN (APBCLK0[1])
  - The RTC Time Counter source can be selected as
    - Internal LIRC if RTCCKSEL (RTC_LXTCTL[7]) = 1
    - External LXT if RTCCKSEL (RTC_LXTCTL[7]) = 0

6.20.5 Functional Description

6.20.5.1 RTC Initiation

When a RTC block is powered on, RTC is at reset state. User has to write a number 0xa5eb1357 to RTC initial register INIT (RTC_INIT[31:0]) to make RTC leaving reset state. Once the INIT register is written as 0xa5eb1357, the RTC will be in normal active state permanently. User can read Active bit (RTC_INIT[0]) to check the RTC is at normal active state or reset state.

6.20.5.2 RTC Control Registers Access Attribute

The RTC control registers access attributes are shown in Table 6.20-1 .

Register	INIT = 0	INIT = 1
RTC_INIT	R/W	R/W
RTC_FREQADJ	R/W	R/W
RTC_TIME	Not available	R/W
RTC_CAL	Not available	R/W
RTC_CLKFMT	Not available	R/W
RTC_WEEKDAY	Not available	R/W
RTC_TALM	Not available	R/W
RTC_CALM	Not available	R/W
RTC_LEAPYEAR	Not available	R
RTC_INTEN	R/W	R/W
RTC_INTSTS	R/W	R/W
RTC_TICK	Not available	R/W
RTC_TAMSK	Not available	R/W
RTC_CAMSK	Not available	R/W
RTC_SPRCTL	R/W	R/W
RTC_SPRx	R/W	R/W
RTC_LXTCTL	R/W	R/W
RTC_GPIOCTL0	R/W	R/W
RTC_GPIOCTL1	R/W	R/W
RTC_DSTCTL	Not available	R/W
RTC_TAMPCTL	R/W	R/W

Table 6.20-1 RTC Read/Write Enable

6.20.5.3 Frequency Compensation

The frequency compensation circuit supports dynamic compensation to adjust compensation value without the compensation circuit reset. The enable bit for dynamic compensation is DCOMPEN (RTC_CLKFMT[16]).

If dynamic compensation is enabled, the minimal interval to continuous writing RTC_FREQADJ is 0.024 seconds, and FCRBUSY(RTC_FREQADJ[31]) flag is used to indicate that new register write operation is prohibited. And the compensate value will load into compensation circuit after 0.024~1.016 seconds when write to RTC_FREQADJ.

The RTC_FREQADJ register allows user to make digital compensation to a clock input. Please follow the example and formula below to write the actual frequency of 32k crystal to RTC_FREQADJ register. Following are the compensation examples for higher or lower than 32768 Hz.

Example 1:

Frequency counter measurement: 32773.65 Hz

Integer Part: 32773 => Setting INTEGER (RTC_FREQADJ[12:8]) = 0x15

Fraction Part: 0.65 X 64 = 41.6 (0x2A) => Setting FRACTION (RTC_FREQADJ[5:0]) = 0x2A

Example 2:

Frequency counter measurement: 32763.25 Hz

Integer part: 32763 => Setting INTEGER (RTC_FREQADJ[12:8]) = 0x0B

Fraction part: 0.25 X 64 = 16 (0x10) => Setting FRACTION (RTC_FREQADJ[5:0]) = 0x10

Note: The value of RTC_FREQADJ register will be the default value (0x0000_1000) while the compensation is not executed. User can utilize a frequency counter to measure RTC clock source via clock output function in manufacturing. Meanwhile, user can use clock output function to check the result of RTC frequency compensation.

6.20.5.4 Time and Calendar counter

RTC_TIME and RTC_CAL are used to load the real time and calendar. RTC_TALM and RTC_CALM are used for setup alarm time and calendar.

6.20.5.5 12/24 hour Time Scale Selection

The 12/24 hour time scale selection depends on 24HEN (RTC_CLKFMT[0]).

When RTC runs as 12-hour time scale mode, RTC_TIME[21] (the high bit of TENHR[1:0]) means AM/PM indication, if RTC_TIME[21] is 1, it indicates PM time message and RTC_TIME[21] is 0 indicates AM time message.

Note: The Hour Value Write Into RTC_TIME[21:16], Messages Are Expressed In BCD Format			
24-Hour Time Scale (24HEN = 1)		12-Hour Time Scale (PM Time + 0x20) (24HEN = 0)	
0x00 (AM12)	0x12 (PM12)	0x12 (AM12)	0x32 (PM12)
0x01 (AM01)	0x13 (PM01)	0x01 (AM01)	0x21 (PM01)
0x02 (AM02)	0x14 (PM02)	0x02 (AM02)	0x22 (PM02)
0x03 (AM03)	0x15 (PM03)	0x03 (AM03)	0x23 (PM03)
0x04 (AM04)	0x16 (PM04)	0x04 (AM04)	0x24 (PM04)
0x05 (AM05)	0x17 (PM05)	0x05 (AM05)	0x25 (PM05)
0x06 (AM06)	0x18 (PM06)	0x06 (AM06)	0x26 (PM06)
0x07 (AM07)	0x19 (PM07)	0x07 (AM07)	0x27 (PM07)
0x08 (AM08)	0x20 (PM08)	0x08 (AM08)	0x28 (PM08)
0x09 (AM09)	0x21 (PM09)	0x09 (AM09)	0x29 (PM09)
0x10 (AM10)	0x22 (PM10)	0x10 (AM10)	0x30 (PM10)
0x11 (AM11)	0x23 (PM11)	0x11 (AM11)	0x31 (PM11)

Table 6.20-212/24 Hour Time Scale Selection

6.20.5.6 Day of the Week Counter

The RTC controller provides day of week in WEEKDAY (RTC_WEEKDAY[2:0]) bits. The value is defined from 0 to 6 to represent Sunday to Saturday respectively.

6.20.5.7 Periodic Time Tick Interrupt

The Periodic Time Tick interrupt has 8 period interval options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second that are selected by TICK (RTC_TICK[2:0]) bits. When Periodic Time Tick interrupt is enabled by setting TICKIEN (RTC_INTEN[1]) to 1, the Periodic Time Tick interrupt is requested periodically in the period selected by RTC_TICK[2:0] settings.

6.20.5.8 Alarm Interrupt

When the real time and calendar message in RTC_TIME and RTC_CAL registers are equal to alarm time and calendar values in RTC_TALM and RTC_CALM registers, the RTC alarm interrupt flag ALMIF (RTC_INTSTS[0]) is set to 1 and the RTC alarm interrupt signal assert if the alarm interrupt enable ALMIEN (RTC_INTEN[0]) is enabled.

The RTC controller provides Time Alarm Mask Register (RTC_TAMSK) and Calendar Alarm Mask Register (RTC_CAMSK) to mask the specified digit and generate periodic interrupt without changing the alarm match condition in RTC_TALM and RTC_CALM registers in each alarm interrupt service routine.

6.20.5.9 Daylight Saving Time

The RTC controller also provides RTC_DSTCTL register to store the control settings of daylight saving time application. User can read RTC_DSTCTL value to check current RTC date/time counter runs in daylight saving time mode or normal mode.

6.20.5.101 Hz clock output

The RTC controller provides 1Hz clock output to CLKO function pin. User can set CLK1HZEN (CLK_CLKOCTL[6]) to 1 and enable RTC, 1Hz clock will output to CLKO function pin.

6.20.5.11 Application Note

1. All data in RTC_TALM, RTC_CALM, RTC_TIME and RTC_CAL registers are all expressed in BCD format.
2. User has to make sure that the loaded values are reasonable. For example, Load RTC_CAL as 201a (year), 13 (month), 00 (day), or RTC_CAL does not match with RTC_WEEKDAY, etc.
3. In RTC_CAL and RTC_CALM, only 2 BCD digits are used to express “year”. The 2 BCD digits of xy means 20xy, rather than 19xy or 21xy.
4. Example of 12-Hour Time Setting
   If current RTC time is PM12:59:30 in 12-Hour Time Scale mode, the RTC_TIME setting as:

   HOUR:
   RTC_TIME[21:16]: 0x32 (0x12+0x20) combined by TENHR (RTC_TIME[21:20]) is 0x3, HR (RTC_TIME[19:16]) is 0x2.

   MIN:
   RTC_TIME[14:8]: 0x59 combined by TENMIN (RTC_TIME[14:12]) is 0x5, MIN (RTC_TIME[11:8]) is 0x9.

   SEC:
   RTC_TIME[6:0]: 0x30 combined by TENSEC (RTC_TIME[6:4]) is 0x3, SEC (RTC_TIME[3:0]) is 0x0.

Table 6.20-3 shows registers value after both core power and battery power are first powered on.

Register	Reset State
RTC_INIT	0
RTC_CAL	15/8/8 (year/month/day)

RTC_TIME	00:00:00 (hour : minute : second)
RTC_CALM	00/00/00 (year/month/day)
RTC_TALM	00:00:00 (hour : minute : second)
RTC_CLKFMT	1 (24-hour mode)
RTC_WEEKDAY	6 (Saturday)
RTC_INTEN	0
RTC_INTSTS	0
RTC_LEAPYEAR	0
RTC_TICK	0
RTC_DSTCTL	0

Table 6.20-3 Register Value After Powered On

6.20.5.12 Spare Registers and Tamper Detector

The RTC module is equipped with 20 bytes spare registers to store user's important information. These spare registers are located in RTC domain, user needs to enable SPRRWEN (RTC_SPRCTL[2]) before writing one of 5 spare registers (RTC_SPR0 ~ RTC_SPR4).

When the transition condition defined in RTC_TAMPCTL is detected, tamper detected interrupt flag TAMPxIF (RTC_INTSTS[10:8]) will be generated. Meanwhile, the 20 bytes spare registers (RTC_SPR0 ~ RTC_SPR4) content will be cleared automatically by hardware to prevent the security data be disclosure and current RTC time and calendar will be loaded to RTC_TAMPTIME and RTC_TAMPCAL registers, these values only can be update again when all TAMPxIF are cleared to 0. And if TAMPxIF is set to 1, the interrupt is generated to NVIC if the tamper detect interrupt enable RTC_TAMPxIEN (RTC_INTEN[10:8]) is active.

After Flash mass erase operation, these 20 bytes spare registers content will be cleared automatically.

RTC support 3 individual tamper pin. The TAMPxLV (RTC_TAMPCTL[4x+9], x=0, 1,2) depend on level attribute of TAMPERx pin for individual tamper detection.

Static Tamper Programming Sequence Example:

Enable Tamper Programming Sequence:

1. Clean the TAMPxIF (RTC_INTSTS[x+8], x=0, 1, 2)
2. Set TAMPxLV (RTC_TAMPCTL[4x+9], x=0, 1, 2) and TAMPxDBEN (RTC_TAMPCTL[4x+10], x=0, 1, 2).
3. Enable TAMPxEN (RTC_TAMPCTL[4x+8], x=0, 1, 2).

Disable or Re-configure Tamper Programming Sequence:

1. Disable TAMPxEN (RTC_TAMPCTL[4x+8], x=0, 1, 2).
2. Waiting for 3 RTC CLK.
3. Re-configure TAMPxLV (RTC_TAMPCTL[4x+9], x=0, 1, 2) and TAMPxDBEN (RTC_TAMPCTL[4x+10], x=0, 1, 2).
4. Enable TAMPxEN (RTC_TAMPCTL[4x+8], x=0, 1, 2)

6.20.5.13 Backup Domain GPIO Function

When PF.4/X32_OUT and PF.5/X32_IN pins are not used as low speed 32K oscillator function, PF.4~11 can be used as GPIO pin function. The IOCTLSEL (RTC_LXTCTL[8]) is used to select the PF.4~11 pins

are controlled by RTC_GPIOCTL0/1 registers by V_{BAT} power domain or relative GPIO control registers by GPIO module. Figure 6.20-2 shows backup I/O control diagram.

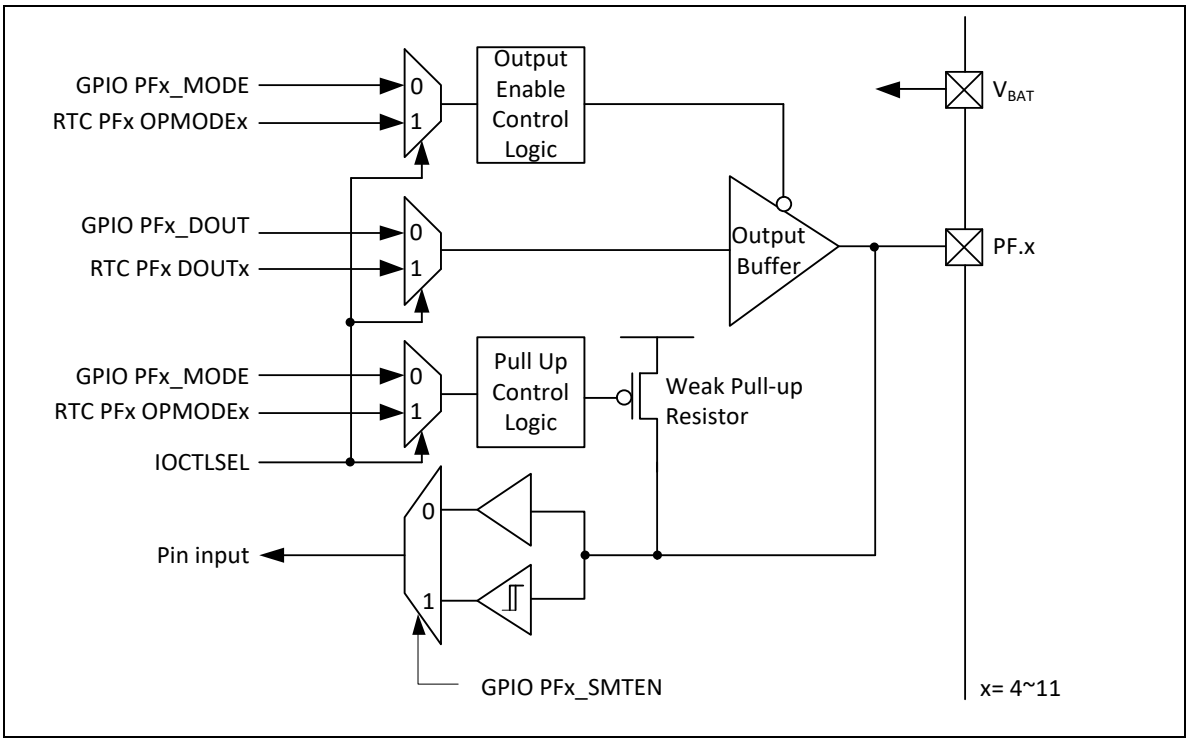


Figure 6.20-2 Backup I/O Control Diagram

6.20.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
RTC Base Address: **RTC_BA = 0x4004_1000**				
RTC_INIT	RTC_BA+0x00	R/W	RTC Initiation Register	0x0000_0000
RTC_FREQADJ	RTC_BA+0x08	R/W	RTC Frequency Compensation Register	0x0000_1000
RTC_TIME	RTC_BA+0x0C	R/W	RTC Time Loading Register	0x0000_0000
RTC_CAL	RTC_BA+0x10	R/W	RTC Calendar Loading Register	0x0015_0808
RTC_CLKFMT	RTC_BA+0x14	R/W	RTC Time Scale Selection Register	0x0000_0001
RTC_WEEKDAY	RTC_BA+0x18	R/W	RTC Day of the Week Register	0x0000_0006
RTC_TALM	RTC_BA+0x1C	R/W	RTC Time Alarm Register	0x0000_0000
RTC_CALM	RTC_BA+0x20	R/W	RTC Calendar Alarm Register	0x0000_0000
RTC_LEAPYEAR	RTC_BA+0x24	R	RTC Leap Year Indicator Register	0x0000_0000
RTC_INTEN	RTC_BA+0x28	R/W	RTC Interrupt Enable Register	0x0000_0000
RTC_INTSTS	RTC_BA+0x2C	R/W	RTC Interrupt Status Register	0x0000_0000
RTC_TICK	RTC_BA+0x30	R/W	RTC Time Tick Register	0x0000_0000
RTC_TAMSK	RTC_BA+0x34	R/W	RTC Time Alarm Mask Register	0x0000_0000
RTC_CAMSK	RTC_BA+0x38	R/W	RTC Calendar Alarm Mask Register	0x0000_0000
RTC_SPRCTL	RTC_BA+0x3C	R/W	RTC Spare Functional Control Register	0x0000_0000
RTC_SPR0	RTC_BA+0x40	R/W	RTC Spare Register 0	0x0000_0000
RTC_SPR1	RTC_BA+0x44	R/W	RTC Spare Register 1	0x0000_0000
RTC_SPR2	RTC_BA+0x48	R/W	RTC Spare Register 2	0x0000_0000
RTC_SPR3	RTC_BA+0x4C	R/W	RTC Spare Register 3	0x0000_0000
RTC_SPR4	RTC_BA+0x50	R/W	RTC Spare Register 4	0x0000_0000
RTC_LXTCTL	RTC_BA+0x100	R/W	RTC 32.768 kHz Oscillator Control Register	0xXXXX_XX06
RTC_GPIOCTL0	RTC_BA+0x104	R/W	RTC GPIO Control 0 Register	0x0000_0000
RTC_GPIOCTL1	RTC_BA+0x108	R/W	RTC GPIO Control 1 Register	0x0000_0000
RTC_DSTCTL	RTC_BA+0x110	R/W	RTC Daylight Saving Time Control Register	0x0000_0000
RTC_TAMPCTL	RTC_BA+0x120	R/W	RTC Tamper Pin Control Register	0x0000_0000
RTC_TAMPTIME	RTC_BA+0x130	R	RTC Tamper Time Register	0x0000_0000

RTC_TAMPCAL	RTC_BA+0x134	R	RTC Tamper Calendar Register	0x0000_0000

6.20.7 Register Description

RTC Initiation Register (RTC_INIT)

Register	Offset	R/W	Description	Reset Value
RTC_INIT	RTC_BA+0x00	R/W	RTC Initiation Register	0x0000_0000

31	30	29	28	27	26	25	24
INIT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
INIT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
INIT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INIT							**ACTIVE**

Bits	Description	
[31:1]	**INIT**	**RTC Initiation (Write Only)** When RTC block is powered on, RTC is at reset state. User has to write a number (0x a5eb1357) to INIT to make RTC leave reset state. Once the INIT is written as 0xa5eb1357, the RTC will be in un-reset state permanently. The INIT is a write-only field and read value will be always 0.
[0]	**ACTIVE**	**RTC Active Status (Read Only)** 0 = RTC is at reset state. 1 = RTC is at normal active state.

RTC Frequency Compensation Register (RTC_FREQADJ)

Register	Offset	R/W	Description	Reset Value
RTC_FREQADJ	RTC_BA+0x08	R/W	RTC Frequency Compensation Register	0x0000_1000

31	30	29	28	27	26	25	24
FCRBUSY	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			INTEGER				
7	6	5	4	3	2	1	0
Reserved		FRACTION					

Bits	Description	
[31]	**FCRBUSY**	**Frequency Compensation Register Write Operation Busy (Read Only)** 0 = The new register write operation is acceptable. 1 = The last write operation is in progress and new register write operation prohibited. **Note:** This bit is only used when DCOMPEN (RTC_CLKFMT[16]) enabled.
[30:13]	**Reserved**	Reserved.
[12:8]	**INTEGER**	**Integer Part** 00000 = Integer part of detected value is 32752. 00001 = Integer part of detected value is 32753. 00010 = Integer part of detected value is 32754. 00011 = Integer part of detected value is 32755. 00100 = Integer part of detected value is 32756. 00101 = Integer part of detected value is 32757. 00110 = Integer part of detected value is 32758. 00111 = Integer part of detected value is 32759. 01000 = Integer part of detected value is 32760. 01001 = Integer part of detected value is 32761. 01010 = Integer part of detected value is 32762. 01011 = Integer part of detected value is 32763. 01100 = Integer part of detected value is 32764. 01101 = Integer part of detected value is 32765. 01110 = Integer part of detected value is 32766. 01111 = Integer part of detected value is 32767. 10000 = Integer part of detected value is 32768. 10001 = Integer part of detected value is 32769. 10010 = Integer part of detected value is 32770. 10011 = Integer part of detected value is 32771. 10100 = Integer part of detected value is 32772. 10101 = Integer part of detected value is 32773.

		10110 = Integer part of detected value is 32774. 10111 = Integer part of detected value is 32775. 11000 = Integer part of detected value is 32776. 11001 = Integer part of detected value is 32777. 11010 = Integer part of detected value is 32778. 11011 = Integer part of detected value is 32779. 11100 = Integer part of detected value is 32780. 11101 = Integer part of detected value is 32781. 11110 = Integer part of detected value is 32782. 11111 = Integer part of detected value is 32783.
[7:6]	**Reserved**	Reserved.
[5:0]	**FRACTION**	**Fraction Part** Formula: FRACTION = (fraction part of detected value) X 64. **Note:** Digit in FCR must be expressed as hexadecimal number.

Note: If the dynamic compensation is not enabled, i.e. DCOMPEN (RTC_CLKFMT[16]) = 0, the FREQADJ counter will be reset for starting to compensate when writing RTC_FREQADJ. It will impact if the RTC counter will be restarted.

RTC Time Loading Register (RTC_TIME)

Register	Offset	R/W	Description	Reset Value
RTC_TIME	RTC_BA+0x0C	R/W	RTC Time Loading Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	HZCNT						
23	22	21	20	19	18	17	16
Reserved		TENHR		HR			
15	14	13	12	11	10	9	8
Reserved	TENMIN			MIN			
7	6	5	4	3	2	1	0
Reserved	TENSEC			SEC			

Bits	Description	
[31]	**Reserved**	Reserved.
[30:24]	**HZCNT**	Index of sub-second counter (0x00~0x7F)
[23:22]	**Reserved**	Reserved.
[21:20]	**TENHR**	10-Hour Time Digit (0~2) When RTC runs as 12-hour time scale mode, RTC_TIME[21] (the high bit of TENHR[1:0]) means AM/PM indication (If RTC_TIME[21] is 1, it indicates PM time message.)
[19:16]	**HR**	1-Hour Time Digit (0~9)
[15]	**Reserved**	Reserved.
[14:12]	**TENMIN**	10-Min Time Digit (0~5)
[11:8]	**MIN**	1-Min Time Digit (0~9)
[7]	**Reserved**	Reserved.
[6:4]	**TENSEC**	10-Sec Time Digit (0~5)
[3:0]	**SEC**	1-Sec Time Digit (0~9)

Note:

1. RTC_TIME is a BCD digit counter (except HZCNT) and RTC will not check loaded data.
2. The reasonable value range is listed in the parenthesis.
3. The FREQADJ counter will be reset for starting to compensate when writing RTC_TIME, RTC_CAL, RTC_WEEKDAY. It will impact if the RTC counter will be restarted.
4. HZCNT will set to “0” when writing RTC_TIME in RTC_CLKFMT[8] = 0. User can write value 0x00 ~ 0x7F in RTC_CLKFMT[8] = 1 .

RTC Calendar Loading Register (RTC_CAL)

Register	Offset	R/W	Description	Reset Value
RTC_CAL	RTC_BA+0x10	R/W	RTC Calendar Loading Register	0x0015_0808

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TENYEAR				**YEAR**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**TENMON**	**MON**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TENDAY**		**DAY**			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**TENYEAR**	10-Year Calendar Digit (0~9)
[19:16]	**YEAR**	1-Year Calendar Digit (0~9)
[15:13]	**Reserved**	Reserved.
[12]	**TENMON**	10-Month Calendar Digit (0~1)
[11:8]	**MON**	1-Month Calendar Digit (0~9)
[7:6]	**Reserved**	Reserved.
[5:4]	**TENDAY**	10-Day Calendar Digit (0~3)
[3:0]	**DAY**	1-Day Calendar Digit (0~9)

Note:

1. RTC_CAL is a BCD digit counter and RTC will not check loaded data.
2. The reasonable value range is listed in the parenthesis.
3. The FREQADJ counter will be reset for starting to compensate when writing RTC_TIME, RTC_CAL, RTC_WEEKDAY. It will impact if the RTC counter will be restarted.

RTC Time Scale Selection Register (RTC_CLKFMT)

Register	Offset	R/W	Description	Reset Value
RTC_CLKFMT	RTC_BA+0x14	R/W	RTC Time Scale Selection Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**DCOMPEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**HZCNTEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**24HEN**

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**DCOMPEN**	**Dynamic Compensation Enable Bit** 0 = Dynamic Compensation Disabled. 1 = Dynamic Compensation Enabled.
[15:9]	**Reserved**	Reserved.
[8]	**HZCNTEN**	**Sub-second Counter Enable Bit** 0 = HZCNT disabled in RTC_TIME and RTC_TALM. 1 = HZCNT enabled in RTC_TIME and RTC_TALM.
[7:1]	**Reserved**	Reserved.
[0]	**24HEN**	**24-hour / 12-hour Time Scale Selection** Indicates that RTC_TIME and RTC_TALM are in 24-hour time scale or 12-hour time scale 0 = 12-hour time scale with AM and PM indication selected. 1 = 24-hour time scale selected.

RTC Day of the Week Register (RTC_WEEKDAY)

Register	Offset	R/W	Description	Reset Value
RTC_WEEKDAY	RTC_BA+0x18	R/W	RTC Day of the Week Register	0x0000_0006

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**WEEKDAY**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**WEEKDAY**	**Day of the Week Register** 000 = Sunday. 001 = Monday. 010 = Tuesday. 011 = Wednesday. 100 = Thursday. 101 = Friday. 110 = Saturday. 111 = Reserved.

Note: The FREQADJ counter will be reset for starting to compensate when writing RTC_TIME, RTC_CAL, RTC_WEEKDAY. It will impact if the RTC counter will be restarted.

RTC Time Alarm Register (RTC_TALM)

Register	Offset	R/W	Description	Reset Value
RTC_TALM	RTC_BA+0x1C	R/W	RTC Time Alarm Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	HZCNT						
23	22	21	20	19	18	17	16
Reserved		TENHR		HR			
15	14	13	12	11	10	9	8
Reserved	TENMIN			MIN			
7	6	5	4	3	2	1	0
Reserved	TENSEC			SEC			

Bits	Description	
[31]	**Reserved**	Reserved.
[30:24]	**HZCNT**	Index of sub-second counter (0x00~0x7F)
[23:22]	**Reserved**	Reserved.
[21:20]	**TENHR**	10-Hour Time Digit of Alarm Setting (0~2) When RTC runs as 12-hour time scale mode, RTC_TIME[21] (the high bit of TENHR[1:0]) means AM/PM indication (If RTC_TIME[21] is 1, it indicates PM time message.)
[19:16]	**HR**	1-Hour Time Digit of Alarm Setting (0~9)
[15]	**Reserved**	Reserved.
[14:12]	**TENMIN**	10-Min Time Digit of Alarm Setting (0~5)
[11:8]	**MIN**	1-Min Time Digit of Alarm Setting (0~9)
[7]	**Reserved**	Reserved.
[6:4]	**TENSEC**	10-Sec Time Digit of Alarm Setting (0~5)
[3:0]	**SEC**	1-Sec Time Digit of Alarm Setting (0~9)

Note:

1. RTC_TALM is a BCD digit counter, except HZCNT.
2. The reasonable value range is listed in the parenthesis.

RTC Calendar Alarm Register (RTC_CALM)

Register	Offset	R/W	Description	Reset Value
RTC_CALM	RTC_BA+0x20	R/W	RTC Calendar Alarm Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TENYEAR				**YEAR**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**TENMON**	**MON**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TENDAY**		**DAY**			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**TENYEAR**	10-Year Calendar Digit of Alarm Setting (0~9)
[19:16]	**YEAR**	1-Year Calendar Digit of Alarm Setting (0~9)
[15:13]	**Reserved**	Reserved.
[12]	**TENMON**	10-Month Calendar Digit of Alarm Setting (0~1)
[11:8]	**MON**	1-Month Calendar Digit of Alarm Setting (0~9)
[7:6]	**Reserved**	Reserved.
[5:4]	**TENDAY**	10-Day Calendar Digit of Alarm Setting (0~3)
[3:0]	**DAY**	1-Day Calendar Digit of Alarm Setting (0~9)

Note:

1. RTC_CALM is a BCD digit counter.
2. The reasonable value range is listed in the parenthesis.

RTC Leap Year Indication Register (RTC_LEAPYEAR)

Register	Offset	R/W	Description	Reset Value
RTC_LEAPYEAR	RTC_BA+0x24	R	RTC Leap Year Indicator Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**LEAPYEAR**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**LEAPYEAR**	**Leap Year Indication (Read Only)** 0 = This year is not a leap year. 1 = This year is leap year.

RTC Interrupt Enable Register (RTC_INTEN)

Register	Offset	R/W	Description	Reset Value
RTC_INTEN	RTC_BA+0x28	R/W	RTC Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					TAMP2IEN	TAMP1IEN	TAMP0IEN
7	6	5	4	3	2	1	0
Reserved						TICKIEN	ALMIEN

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**TAMP2IEN**	**Tamper 2 Interrupt Enable Bit** Set TAMP2IEN to 1 can also enable chip wake-up function when tamper 2 interrupt event is generated. 0 = Tamper 2 interrupt Disabled. 1 = Tamper 2 interrupt Enabled.
[9]	**TAMP1IEN**	**Tamper 1 Interrupt Enable Bit** Set TAMP1IEN to 1 can also enable chip wake-up function when tamper 1 interrupt event is generated. 0 = Tamper 1 interrupt Disabled. 1 = Tamper 1 interrupt Enabled.
[8]	**TAMP0IEN**	**Tamper 0 Interrupt Enable Bit** Set TAMP0IEN to 1 can also enable chip wake-up function when tamper 0 interrupt event is generated. 0 = Tamper 0 interrupt Disabled. 1 = Tamper 0 interrupt Enabled.
[7:2]	**Reserved**	Reserved.
[1]	**TICKIEN**	**Time Tick Interrupt Enable Bit** Set TICKIEN to 1 can also enable chip wake-up function when RTC tick interrupt event is generated. 0 = RTC Time Tick interrupt Disabled. 1 = RTC Time Tick interrupt Enabled.
[0]	**ALMIEN**	**Alarm Interrupt Enable Bit** Set ALMIEN to 1 can also enable chip wake-up function when RTC alarm interrupt event is generated. 0 = RTC Alarm interrupt Disabled. 1 = RTC Alarm interrupt Enabled.

RTC Interrupt Status Register (RTC_INTSTS)

Register	Offset	R/W	Description	Reset Value
RTC_INTSTS	RTC_BA+0x2C	R/W	RTC Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					TAMP2IF	TAMP1IF	TAMP0IF
7	6	5	4	3	2	1	0
Reserved						TICKIF	ALMIF

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**TAMP2IF**	**Tamper 2 Interrupt Flag** This bit is set when TAMPER2 detected level non-equal TAMP2LV (RTC_TAMPCTL[17]). 0 = No Tamper 2 interrupt flag is generated. 1 = Tamper 2 interrupt flag is generated. **Note 1:** Write 1 to clear this bit. **Note 2:** This interrupt flag will generate again when Tamper setting condition is not restoration. **Note 3:** Need to clear all TAMPxIF, after that, new RTC_TAMPTIME and RTC_TAMPCAL values can loaded while a tamper event occurred.
[9]	**TAMP1IF**	**Tamper 1 Interrupt Flag** This bit is set when TAMPER1 detected level non-equal TAMP1LV (RTC_TAMPCTL[13]). 0 = No Tamper 1 interrupt flag is generated. 1 = Tamper 1 interrupt flag is generated. **Note 1:** Write 1 to clear this bit. **Note 2:** This interrupt flag will generate again when Tamper setting condition is not restoration. **Note 3:** Need to clear all TAMPxIF, after that, new RTC_TAMPTIME and RTC_TAMPCAL values can loaded while a tamper event occurred.
[8]	**TAMP0IF**	**Tamper 0 Interrupt Flag** This bit is set when TAMPER0 detected level non-equal TAMP0LV (RTC_TAMPCTL[9]). 0 = No Tamper 0 interrupt flag is generated. 1 = Tamper 0 interrupt flag is generated. **Note 1:** Write 1 to clear this bit. **Note 2:** This interrupt flag will generate again when Tamper setting condition is not restoration. **Note 3:** Need to clear all TAMPxIF, after that, new RTC_TAMPTIME and RTC_TAMPCAL values can loaded while a tamper event occurred.
[7:2]	**Reserved**	Reserved.

[1]	**TICKIF**	**RTC Time Tick Interrupt Flag** 0 = Tick condition did not occur. 1 = Tick condition occurred. **Note:** Write 1 to clear this bit.
[0]	**ALMIF**	**RTC Alarm Interrupt Flag** 0 = Alarm condition is not matched. 1 = Alarm condition is matched. **Note:** Write 1 to clear this bit.

RTC Time Tick Register (RTC_TICK)

Register	Offset	R/W	Description	Reset Value
RTC_TICK	RTC_BA+0x30	R/W	RTC Time Tick Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					TICK		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**TICK**	**Time Tick Register** These bits are used to select RTC time tick period for Periodic Time Tick Interrupt request. 000 = Time tick is 1 second. 001 = Time tick is 1/2 second. 010 = Time tick is 1/4 second. 011 = Time tick is 1/8 second. 100 = Time tick is 1/16 second. 101 = Time tick is 1/32 second. 110 = Time tick is 1/64 second. 111 = Time tick is 1/128 second.

RTC Time Alarm MASK Register (RTC_TAMSK)

Register	Offset	R/W	Description	Reset Value
RTC_TAMSK	RTC_BA+0x34	R/W	RTC Time Alarm Mask Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MTENHR	MHR	MTENMIN	MMIN	MTENSEC	MSEC

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**MTENHR**	Mask 10-Hour Time Digit of Alarm Setting (0~2)
[4]	**MHR**	Mask 1-Hour Time Digit of Alarm Setting (0~9)
[3]	**MTENMIN**	Mask 10-Min Time Digit of Alarm Setting (0~5)
[2]	**MMIN**	Mask 1-Min Time Digit of Alarm Setting (0~9)
[1]	**MTENSEC**	Mask 10-Sec Time Digit of Alarm Setting (0~5)
[0]	**MSEC**	Mask 1-Sec Time Digit of Alarm Setting (0~9)

Note:

1. RTC_TALM is a BCD digit counter and RTC will not check loaded data.
2. The reasonable value range is listed in the parenthesis.
3. MTENHR/MHR base on 24 hour Time Scale.

RTC Calendar Alarm MASK Register (RTC_CAMSK)

Register	Offset	R/W	Description	Reset Value
RTC_CAMSK	RTC_BA+0x38	R/W	RTC Calendar Alarm Mask Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**MTENYEAR**	**MYEAR**	**MTENMON**	**MMON**	**MTENDAY**	**MDAY**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**MTENYEAR**	Mask 10-Year Calendar Digit of Alarm Setting (0~9)
[4]	**MYEAR**	Mask 1-Year Calendar Digit of Alarm Setting (0~9)
[3]	**MTENMON**	Mask 10-Month Calendar Digit of Alarm Setting (0~1)
[2]	**MMON**	Mask 1-Month Calendar Digit of Alarm Setting (0~9)
[1]	**MTENDAY**	Mask 10-Day Calendar Digit of Alarm Setting (0~3)
[0]	**MDAY**	Mask 1-Day Calendar Digit of Alarm Setting (0~9)

Note:

1. RTC_CALM is a BCD digit counter and RTC will not check loaded data.
2. The reasonable value range is listed in the parenthesis.

RTC Spare Functional Control Register (RTC_SPRCTL)

Register	Offset	R/W	Description	Reset Value
RTC_SPRCTL	RTC_BA+0x3C	R/W	RTC Spare Functional Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		SPRCSTS	Reserved		SPRRWEN	Reserved	

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**SPRCSTS**	**SPR Clear Flag** This bit indicates if the RTC_SPR0 ~ RTC_SPR4 content is cleared when specify tamper event is detected. 0 = Spare register content is not cleared. 1 = Spare register content is cleared. **Note 1:** Write 1 to clear this bit. **Note 2:** This bit keeps 1 when RTC_INTSTS[10:8] is not equal to 0.
[4:3]	**Reserved**	Reserved.
[2]	**SPRRWEN**	**Spare Register Enable Bit** 0 = Spare register Disabled. 1 = Spare register Enabled. **Note:** When spare register is disabled, RTC_SPR0 ~ RTC_SPR4 cannot be accessed.
[1:0]	**Reserved**	Reserved.

RTC Spare Register (RTC_SPRx)

Register	Offset	R/W	Description	Reset Value
RTC_SPR0	RTC_BA+0x40	R/W	RTC Spare Register 0	0x0000_0000
RTC_SPR1	RTC_BA+0x44	R/W	RTC Spare Register 1	0x0000_0000
RTC_SPR2	RTC_BA+0x48	R/W	RTC Spare Register 2	0x0000_0000
RTC_SPR3	RTC_BA+0x4C	R/W	RTC Spare Register 3	0x0000_0000
RTC_SPR4	RTC_BA+0x50	R/W	RTC Spare Register 4	0x0000_0000

31	30	29	28	27	26	25	24
SPARE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SPARE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SPARE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPARE							

Bits	Description	
[31:0]	**SPARE**	**Spare Register** This field is used to store back-up information defined by user. This field will be cleared by hardware automatically in the following conditions, a tamper pin event is detected, or after RRAM mass operation.

RTC 32K Oscillator Control Register (RTC_LXTCTL)

Register	Offset	R/W	Description	Reset Value
RTC_LXTCTL	RTC_BA+0x100	R/W	RTC 32.768 kHz Oscillator Control Register	0xXXXX_XX06

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							IOCTLSEL
7	6	5	4	3	2	1	0
RTCCKSEL	Reserved		GAIN				Reserved

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**IOCTLSEL**	**I/O Pin Backup Control Selection** When low speed 32 kHz oscillator is disabled or TAMPxEN is disabled, PF.4 pin (X32_OUT pin), PF.5 pin (X32_IN pin) or PF.6~8 pin (TAMPERx pin) can be used as GPIO function. User can program IOCTLSEL to decide PF.4~11 I/O function is controlled by system power domain GPIO module or V_{BAT} power domain RTC_GPIOCTL0/1 control register. 0 = PF.4~11 pin I/O function is controlled by GPIO module. 1 = PF.4~11 pin I/O function is controlled by V_{BAT} power domain. **Note:** IOCTLSEL will automatically be set by hardware to 1 when system power is off and any writable RTC registers has been written at RTCCKEN(CLK_APBCLK0[1]) enabled.
[7]	**RTCCKSEL**	**RTC Clock Source Selection** 0 = Clock source from external low speed crystal oscillator (LXT) 1 = Clock source from internal low speed RC oscillator (LIRC).
[6:5]	**Reserved**	Reserved.
[4:1]	**GAIN**	**Oscillator Gain Option** User can select oscillator gain according to crystal external loading and operating temperature range. The larger gain value corresponding to stronger driving capability and higher power consumption. 0000 = L0 mode. 0001 = L1 mode. 0010 = L2 mode. 0011 = L3 mode. (Default) 0100 = L4 mode. 0101 = L5 mode. 0110 = L6 mode. 0111 = L7 mode. 1000 = L8 mode. 1001 = L9 mode.

		1010 = L10 mode. 1011 = L11 mode. 1100 = L12 mode. 1101 = L13 mode. 1110 = L14 mode. 1111 = L15 mode.
[0]	**Reserved**	Reserved.

RTC GPIO Control Register0 (RTC_GPIOCTL0)

Register	Offset	R/W	Description	Reset Value
RTC_GPIOCTL0	RTC_BA+0x104	R/W	RTC GPIO Control 0 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		PUSEL3		DINOFF3	DOUT3	OPMODE3	
23	22	21	20	19	18	17	16
Reserved		PUSEL2		DINOFF2	DOUT2	OPMODE2	
15	14	13	12	11	10	9	8
Reserved		PUSEL1		DINOFF1	DOUT1	OPMODE1	
7	6	5	4	3	2	1	0
Reserved		PUSEL0		DINOFF0	DOUT0	OPMODE0	

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:28]	**PUSEL3**	**I/O Pull-up and Pull-down Enable Bits** Determine PF.7 I/O pull-up or pull-down. 00 = PF.7 pull-up and pull-down Disabled. 01 = PF.7 pull-up Enabled. 10 = PF.7 pull-down Enabled. 11 = PF.7 pull-up and pull-down Disabled. **Note:** Basically, the pull-up control and pull-down control has following behavior limitation. The independent pull-up / pull-down control register is only valid when OPMODE3 is set as input tri-state and open-drain mode.
[27]	**DINOFF3**	**I/O Pin Digital Input Path Disable Bit** 0 = PF.7 digital input path Enabled. 1 = PF.7 digital input path Disabled (digital input tied to low).
[26]	**DOUT3**	**I/O Output Data** 0 = PF.7 output low. 1 = PF.7 output high.
[25:24]	**OPMODE3**	**I/O Operation Mode** 00 = PF.7 is input only mode. 01 = PF.7 is output push pull mode. 10 = PF.7 is open drain mode. 11 = PF.7 is quasi-bidirectional mode.
[23:22]	**Reserved**	Reserved.
[21:20]	**PUSEL2**	**I/O Pull-up and Pull-down Enable Bits** Determine PF.6 I/O Pull-up or Pull-down. 00 = PF.6 pull-up and pull-down Disabled. 01 = PF.6 pull-up Enabled. 10 = PF.6 pull-down Enabled.

		11 = PF.6 pull-up and pull-down Disabled. **Note:** Basically, the pull-up control and pull-down control has following behavior limitation. The independent pull-up / pull-down control register is only valid when OPMODE2 is set as input tri-state and open-drain mode.
[19]	**DINOFF2**	**I/O Pin Digital Input Path Disable Bit** 0 = PF.6 digital input path Enabled. 1 = PF.6 digital input path Disabled (digital input tied to low).
[18]	**DOUT2**	**I/O Output Data** 0 = PF.6 output low. 1 = PF.6 output high.
[17:16]	**OPMODE2**	**I/O Operation Mode** 00 = PF.6 is input only mode. 01 = PF.6 is output push pull mode. 10 = PF.6 is open drain mode. 11 = PF.6 is quasi-bidirectional mode.
[15:14]	**Reserved**	Reserved.
[13:12]	**PUSEL1**	**I/O Pull-up and Pull-down Enable Bits** Determine PF.5 I/O pull-up or pull-down. 00 = PF.5 pull-up and pull-down Disabled. 01 = PF.5 pull-up Enabled. 10 = PF.5 pull-down Enabled. 11 = PF.5 pull-up and pull-down Disabled. **Note:** Basically, the pull-up control and pull-down control has following behavior limitation. The independent pull-up / pull-down control register is only valid when OPMODE1 is set as input tri-state and open-drain mode.
[11]	**DINOFF1**	**I/O Pin Digital Input Path Disable Bit** 0 = PF.5 digital input path Enabled. 1 = PF.5 digital input path Disabled (digital input tied to low).
[10]	**DOUT1**	**I/O Output Data** 0 = PF.5 output low. 1 = PF.5 output high.
[9:8]	**OPMODE1**	**I/O Operation Mode** 00 = PF.5 is input only mode. 01 = PF.5 is output push pull mode. 10 = PF.5 is open drain mode. 11 = PF.5 is quasi-bidirectional mode.
[7:6]	**Reserved**	Reserved.
[5:4]	**PUSEL0**	**I/O Pull-up and Pull-down Enable Bits** Determine PF.4 I/O pull-up or pull-down. 00 = PF.4 pull-up and pull-down Disabled. 01 = PF.4 pull-up Enabled. 10 = PF.4 pull-down Enabled. 11 = PF.4 pull-up and pull-down Disabled. **Note:** Basically, the pull-up control and pull-down control has following behavior limitation. The independent pull-up / pull-down control register is only valid when OPMODE0 is set as input tri-state and open-drain mode.

[3]	**DINOFF0**	**I/O Pin Digital Input Path Disable Bit** 0 = PF.4 digital input path Enabled. 1 = PF.4 digital input path Disabled (digital input tied to low).
[2]	**DOUT0**	**I/O Output Data** 0 = PF.4 output low. 1 = PF.4 output high.
[1:0]	**OPMODE0**	**I/O Operation Mode** 00 = PF.4 is input only mode. 01 = PF.4 is output push pull mode. 10 = PF.4 is open drain mode. 11 = PF.4 is quasi-bidirectional mode.

Note: RTC GPIO will be input mode when not programmed, and user needs to avoid the pin floating problem.

RTC GPIO Control Register1 (RTC_GPIOCTL1)

Register	Offset	R/W	Description	Reset Value
RTC_GPIOCTL1	RTC_BA+0x108	R/W	RTC GPIO Control 1 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		PUSEL7		DINOFF7	DOUT7	OPMODE7	
23	22	21	20	19	18	17	16
Reserved		PUSEL6		DINOFF6	DOUT6	OPMODE6	
15	14	13	12	11	10	9	8
Reserved		PUSEL5		DINOFF5	DOUT5	OPMODE5	
7	6	5	4	3	2	1	0
Reserved		PUSEL4		DINOFF4	DOUT4	OPMODE4	

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:28]	**PUSEL7**	**I/O Pull-up and Pull-down Enable Bits** Determine PF.11 I/O pull-up or pull-down. 00 = PF.11 pull-up and pull-down Disabled. 01 = PF.11 pull-up Enabled. 10 = PF.11 pull-down Enabled. 11 = PF.11 pull-up and pull-down Disabled. **Note:** Basically, the pull-up control and pull-down control has following behavior limitation. The independent pull-up / pull-down control register is only valid when OPMODE7 is set as input tri-state and open-drain mode.
[27]	**DINOFF7**	**I/O Pin Digital Input Path Disable Bit** 0 = PF.11 digital input path Enabled. 1 = PF.11 digital input path Disabled (digital input tied to low).
[26]	**DOUT7**	**I/O Output Data** 0 = PF.11 output low. 1 = PF.11 output high.
[25:24]	**OPMODE7**	**I/O Operation Mode** 00 = PF.11 is input only mode. 01 = PF.11 is output push pull mode. 10 = PF.11 is open drain mode. 11 = PF.11 is quasi-bidirectional mode.
[23:22]	**Reserved**	Reserved.
[21:20]	**PUSEL6**	**I/O Pull-up and Pull-down Enable Bits** Determine PF.10 I/O pull-up or pull-down. 00 = PF.10 pull-up and pull-down Disabled. 01 = PF.10 pull-up Enabled. 10 = PF.10 pull-down Enabled.

		11 = PF.10 pull-up and pull-down Disabled. **Note:** Basically, the pull-up control and pull-down control has following behavior limitation. The independent pull-up / pull-down control register is only valid when OPMODE6 is set as input tri-state and open-drain mode.
[19]	**DINOFF6**	**I/O Pin Digital Input Path Disable Bit** 0 = PF.10 digital input path Enabled. 1 = PF.10 digital input path Disabled (digital input tied to low).
[18]	**DOUT6**	**I/O Output Data** 0 = PF.10 output low. 1 = PF.10 output high.
[17:16]	**OPMODE6**	**I/O Operation Mode** 00 = PF.10 is input only mode. 01 = PF.10 is output push pull mode. 10 = PF.10 is open drain mode. 11 = PF.10 is quasi-bidirectional mode .
[15:14]	**Reserved**	Reserved.
[13:12]	**PUSEL5**	**I/O Pull-up and Pull-down Enable Bits** Determine PF.9 I/O pull-up or pull-down. 00 = PF.9 pull-up and pull-down Disabled. 01 = PF.9 pull-up Enabled. 10 = PF.9 pull-down Enabled. 11 = PF.9 pull-up and pull-down Disabled. **Note:** Basically, the pull-up control and pull-down control has following behavior limitation. The independent pull-up / pull-down control register is only valid when OPMODE5 is set as input tri-state and open-drain mode.
[11]	**DINOFF5**	**I/O Pin Digital Input Path Disable Bit** 0 = PF.9 digital input path Enabled. 1 = PF.9 digital input path Disabled (digital input tied to low).
[10]	**DOUT5**	**I/O Output Data** 0 = PF.9 output low. 1 = PF.9 output high.
[9:8]	**OPMODE5**	**I/O Operation Mode** 00 = PF.9 is input only mode. 01 = PF.9 is output push pull mode. 10 = PF.9 is open drain mode. 11 = PF.9 is quasi-bidirectional mode .
[7:6]	**Reserved**	Reserved.
[5:4]	**PUSEL4**	**I/O Pull-up and Pull-down Enable Bits** Determine PF.8 I/O pull-up or pull-down. 00 = PF.8 pull-up and pull-down Disabled. 01 = PF.8 pull-up Enabled. 10 = PF.8 pull-down Enabled. 11 = PF.8 pull-up and pull-down Disabled. **Note:** Basically, the pull-up control and pull-down control has following behavior limitation. The independent pull-up / pull-down control register is only valid when OPMODE4 is set as input tri-state and open-drain mode.

[3]	**DINOFF4**	**I/O Pin Digital Input Path Disable Bit** 0 = PF.8 digital input path Enabled. 1 = PF.8 digital input path Disabled (digital input tied to low).
[2]	**DOUT4**	**I/O Output Data** 0 = PF.8 output low. 1 = PF.8 output high.
[1:0]	**OPMODE4**	**I/O Operation Mode** 00 = PF.8 is input only mode. 01 = PF.8 is output push pull mode. 10 = PF.8 is open drain mode. 11 = PF.8 is quasi-bidirectional mode.

Note: RTC GPIO will be input mode when not programmed, and user needs to avoid the pin floating problem.

RTC Daylight Saving Time Control Register (RTC_DSTCTL)

Register	Offset	R/W	Description	Reset Value
RTC_DSTCTL	RTC_BA+0x110	R/W	RTC Daylight Saving Time Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					DSBAK	SUBHR	ADDHR

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**DSBAK**	**Daylight Saving Back** 0 = Daylight Saving Change is not performed. 1 = Daylight Saving Change is performed.
[1]	**SUBHR**	**Subtract 1 Hour** 0 = No effect. 1 = Indicates RTC hour digit has been subtracted one hour for winter time change.
[0]	**ADDHR**	**Add 1 Hour** 0 = No effect. 1 = Indicates RTC hour digit has been added one hour for summer time change.

RTC Tamper Pin Control Register (RTC_TAMPCTL)

Register	Offset	R/W	Description	Reset Value
RTC_TAMPCTL	RTC_BA+0x120	R/W	RTC Tamper Pin Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					TAMP2DEN	TAMP2LV	TAMP2EN
15	14	13	12	11	10	9	8
Reserved	TAMP1DEN	TAMP1LV	TAMP1EN	Reserved	TAMP0DEN	TAMP0LV	TAMP0EN
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**TAMP2DEN**	**Tamper 2 De-bounce Enable Bit** 0 = Tamper 2 de-bounce Disabled. 1 = Tamper 2 de-bounce Enabled, tamper detection pin will sync 1 RTC counter clock.
[17]	**TAMP2LV**	**Tamper 2 Level** This bit depends on level attribute of tamper pin for static tamper detection. 0 = Detect voltage level is low. 1 = Detect voltage level is high.
[16]	**TAMP2EN**	**Tamper 2 Detect Enable Bit** 0 = Tamper 2 detect Disabled. 1 = Tamper 2 detect Enabled. **Note:** The detection reference clock is RTC counter clock. Tamper detector need to sync 2 ~ 3 RTC counter clock.
[15]	**Reserved**	Reserved.
[14]	**TAMP1DEN**	**Tamper 1 De-bounce Enable Bit** 0 = Tamper 1 de-bounce Disabled. 1 = Tamper 1 de-bounce Enabled, tamper detection pin will sync 1 RTC counter clock.
[13]	**TAMP1LV**	**Tamper 1 Level** This bit depends on level attribute of tamper pin for static tamper detection. 0 = Detect voltage level is low. 1 = Detect voltage level is high.
[12]	**TAMP1EN**	**Tamper 1 Detect Enable Bit** 0 = Tamper 1 detect Disabled. 1 = Tamper 1 detect Enabled. **Note:** The detection reference clock is RTC counter clock. Tamper detector needs to sync 2 ~ 3 RTC counter clock.

[11]	**Reserved**	Reserved.
[10]	**TAMP0DEN**	**Tamper 0 De-bounce Enable Bit** 0 = Tamper 0 de-bounce Disabled. 1 = Tamper 0 de-bounce Enabled, tamper detection pin will sync 1 RTC counter clock.
[9]	**TAMP0LV**	**Tamper 0 Level** This bit depends on level attribute of tamper pin for static tamper detection. 0 = Detect voltage level is low. 1 = Detect voltage level is high.
[8]	**TAMP0EN**	**Tamper0 Detect Enable Bit** 0 = Tamper 0 detect Disabled. 1 = Tamper 0 detect Enabled. **Note:** The detection reference clock is RTC counter clock. Tamper detector needs to sync 2 ~ 3 RTC counter clock.
[7:0]	**Reserved**	Reserved.

RTC Tamper Time Register (RTC_TAMPTIME)

Register	Offset	R/W	Description	Reset Value
RTC_TAMPTIME	RTC_BA+0x130	R	RTC Tamper Time Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**HZCNT**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**TENHR**		**HR**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**TENMIN**			**MIN**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**TENSEC**			**SEC**			

Bits	Description	
[31]	**Reserved**	Reserved.
[30:24]	**HZCNT**	Index of sub-second counter (0x00~0x7F)
[23:22]	**Reserved**	Reserved.
[21:20]	**TENHR**	10-Hour Time Digit of TAMPER Time (0~2) **Note:** 24-hour time scale only .
[19:16]	**HR**	1-Hour Time Digit of TAMPER Time (0~9)
[15]	**Reserved**	Reserved.
[14:12]	**TENMIN**	10-Min Time Digit of TAMPER Time (0~5)
[11:8]	**MIN**	1-Min Time Digit of TAMPER Time (0~9)
[7]	**Reserved**	Reserved.
[6:4]	**TENSEC**	10-Sec Time Digit of TAMPER Time (0~5)
[3:0]	**SEC**	1-Sec Time Digit of TAMPER Time (0~9)

Note:

1. RTC_TAMPTIME is a BCD digit counter, except HZCNT.
2. The reasonable value range is listed in the parenthesis.
3. This fields cannot be updated until all TAMPxIF are cleared.

RTC Tamper Calendar Register (RTC_TAMPCAL)

Register	Offset	R/W	Description	Reset Value
RTC_TAMPCAL	RTC_BA+0x134	R	RTC Tamper Calendar Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TENYEAR				**YEAR**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**TENMON**	**MON**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TENDAY**		**DAY**			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**TENYEAR**	10-Year Calendar Digit of TAMPER Calendar (0~9)
[19:16]	**YEAR**	1-Year Calendar Digit of TAMPER Calendar (0~9)
[15:13]	**Reserved**	Reserved.
[12]	**TENMON**	10-Month Calendar Digit of TAMPER Calendar (0~1)
[11:8]	**MON**	1-Month Calendar Digit of TAMPER Calendar (0~9)
[7:6]	**Reserved**	Reserved.
[5:4]	**TENDAY**	10-Day Calendar Digit of TAMPER Calendar (0~3)
[3:0]	**DAY**	1-Day Calendar Digit of TAMPER Calendar (0~9)

Note:

1. RTC_TAMPCAL is a BCD digit counter.
2. The reasonable value range is listed in the parenthesis.
3. This fields cannot be updated until all TAMPxIF are cleared.

6.21 I²C Serial Interface Controller (I²C)

6.21.1 Overview

I²C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. The I²C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously.

There are four sets of I²C controllers which support Power-down Wake-up function.

6.21.2 Features

The I²C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main features of the I²C bus include:

- Supports up to four I²C ports
- Master/Slave mode
- Bidirectional data transfer between masters and slaves
- Multi-master bus (no central master)
- Supports Standard mode (100 kbps), Fast mode (400 kbps) and Fast mode plus (1 Mbps)
- Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
- Serial clock synchronization allows devices with different bit rates to communicate via one serial bus
- Serial clock synchronization used as a handshake mechanism to suspend and resume serial transfer
- Built-in 14-bit time-out counter requesting the I²C interrupt if the I²C bus hangs up and timer-out counter overflow
- Programmable clocks allow for versatile rate control
- Supports 7-bit addressing and 10-bit addressing mode
- Supports multiple address recognition (four slave address with mask option)
- Supports Power-down wake-up function
- Supports PDMA with one buffer capability
- Supports two-level buffer function
- Supports setup/hold time programmable
- Add pin swap function
- Supports Bus Management (SM/PM compatible) function Compatible with the SMBUS Specification Rev 2.0 (http://smbus.org/specs/) and PMBUS Specification Rev 1.2 (http://pmbus.org/)

6.21.3 Block Diagram

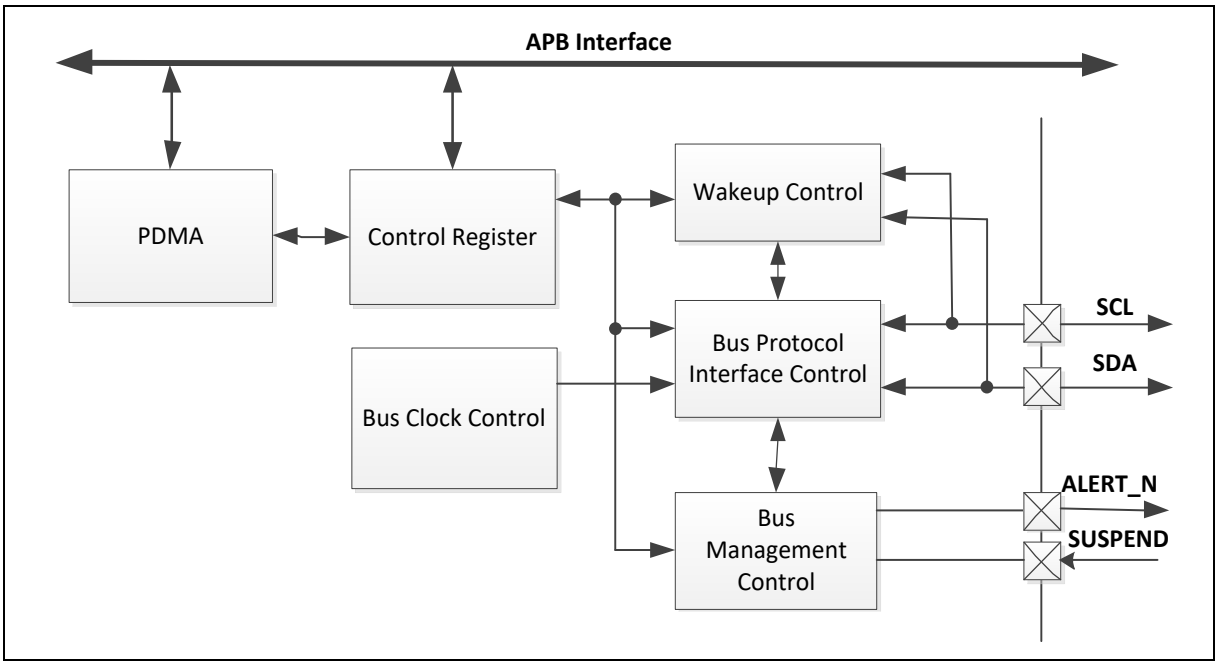


Figure 6.21-1 I²C Controller Block Diagram

6.21.4 Basic Configuration

6.21.4.1 I2C0 Basic Configuration

- Clock Source Configuration
  - Enable I2C0 peripheral clock in I2C0CKEN (CLK_APBCLK0[8]).
- Reset Configuration
  - Reset I2C0 controller in I2C0RST (SYS_IPRST1[8]).
- SCL/SDA Pin Switch
  - Set SWITCHEN (I2C0_CTL1[10] = 1) to switch I2C0 SCL/SDA pin configuration.

6.21.4.2 I2C1 Basic Configuration

- Clock Source Configuration
  - Enable I2C1 peripheral clock in I2C1CKEN (CLK_APBCLK0[9]).
- Reset Configuration
  - Reset I2C1 controller in I2C1RST (SYS_IPRST1[9]).
- SCL/SDA Pin Switch
  - Set SWITCHEN (I2C1_CTL1[10] = 1) to switch I2C1 SCL/SDA pin configuration.

6.21.4.3 I2C2 Basic Configuration

- Clock Source Configuration
  - Enable I2C2 peripheral clock in I2C2CKEN (CLK_APBCLK0[10]).
- Reset Configuration
  - Reset I2C2 controller in I2C2RST (SYS_IPRST1[10]).
- SCL/SDA Pin Switch

- Set SWITCHEN (I2C2_CTL1[10] = 1) to switch I2C2 SCL/SDA pin configuration.

6.21.4.4 I2C3 Basic Configuration

- Clock Source Configuration
  - Enable I2C3 peripheral clock in I2C3CKEN (CLK_APBCLK0[11]).
- Reset Configuration
  - Reset I2C3 controller in I2C3RST (SYS_IPRST1[11]).
- SCL/SDA Pin Switch
  - Set SWITCHEN (I2C3_CTL1[10] = 1) to switch I2C3 SCL/SDA pin configuration.

6.21.5 Functional Description

On I²C bus, data is transferred between a Master and a Slave. Data bits transfer on the SCL and SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit long. There is one SCL clock pulse for each data bit with the MSB being transmitted first, and an acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (START or STOP). Please refer to Figure 6.21-2 for more detailed I²C BUS Timing.

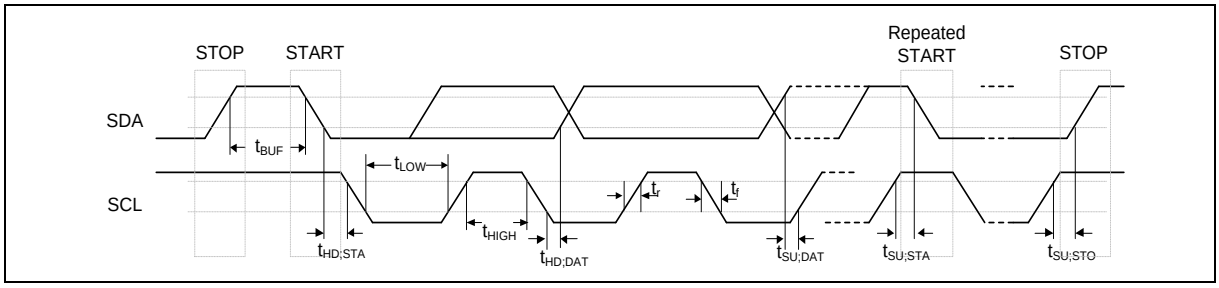


Figure 6.21-2 I²C Bus Timing

The device's on-chip I²C provides the serial interface that meets the I²C bus standard mode specification. The I²C port handles byte transfers autonomously. To enable this port, the bit I2CEN(I2C_CTL0[6]) should be set to '1'. The I²C hardware interfaces to the I²C bus via two pins: SDA and SCL. When I/O pins are used as I²C ports, user must set the pins function to I²C in advance.

There is two-level buffer to improve the performance of I²C bus. In Two-level Buffer mode, the next transmitted or the last received data can be active even if the current data is transmitted or the last received isn't read back yet.

There are under run or over run interrupt when the Two-level Buffer mode is enabled and the interrupt event enable is set. When two-level buffer under run, user will capture the last data be read. When two-level buffer over run, user will capture the last data be written.

Note: Pull-up resistor is needed for I²C operation as the SDA and SCL are open-drain pins.

6.21.5.1 I²C Protocol

Figure 6.21-3 shows the typical I²C protocol. Normally, a standard communication consists of four parts:

- START or Repeated START signal generation
- Slave address and R/W bit transfer
- Data transfer
- STOP signal generation

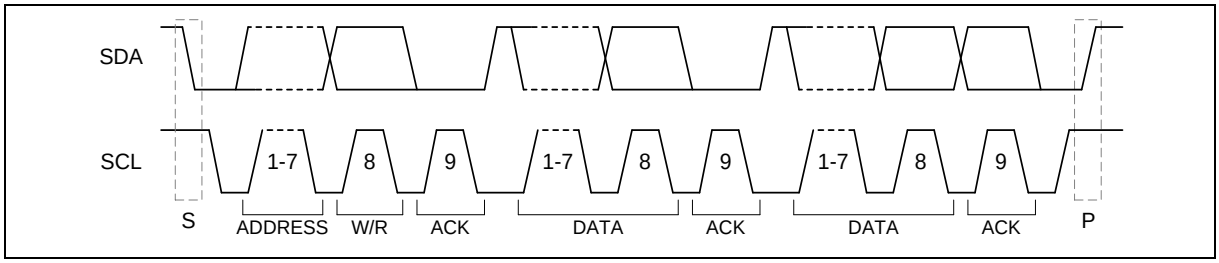


Figure 6.21-3 I²C Protocol

- START or Repeated START signal

  When the bus is free/idle, which means no master device is engaging the bus (both SCL and SDA lines are high), a master can initiate a transfer by sending a START signal. A START signal, usually referred to as the “S” bit, is defined as a HIGH to LOW transition on the SDA line while SCL is HIGH. The START signal denotes the beginning of a new data transmission.

  After having sent the address byte (address and read/write bit), the master may send any number of bytes followed by a stop condition. Instead of sending the stop condition it is also allowed to send another start condition again followed by an address (and of course including a read/write bit) and more data. The start condition is called as Repeat START (Sr). This is defined recursively allowing any number of start conditions to be sent. The purpose of this is to allow combined write/read operations to one or more devices without releasing the bus and thus with the guarantee that the operation is not interrupted. The controller uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

- STOP signal

  The master can terminate the communication by generating a STOP signal. A STOP signal, usually referred to as the “P” bit, is defined as a LOW to HIGH transition on the SDA line while SCL is HIGH.

  Figure 6.21-4 shows the waveform of START, Repeat START and STOP.

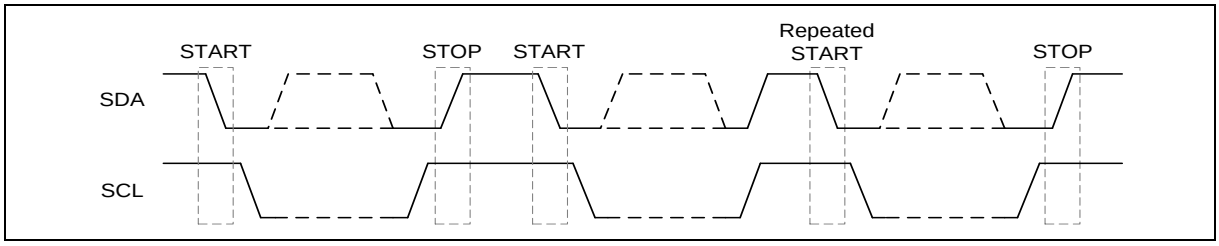


Figure 6.21-4 START and STOP Conditions

- Slave Address Transfer

  After a (Repeated) START condition, the master sends a slave address to identify the target device of the communication. The start address can comprise one or two address bytes (for 7-bit or for 10-bit addressing schemes). After an address byte, a slave sensitive to the transmitted address has to acknowledge the reception.

  Therefore, the slave’s address can be programmed in the device, where it is compared to the received address. In case of a match, the slave answers with an acknowledge (SDA = 0). Slaves that are not targeted answer with a non-acknowledge (SDA = 1). In addition to the match of the programmed address, another address byte value has to be answered with an acknowledge if the slave is capable to handle the corresponding requests.

- Data Transfer

  When a slave receives a correct address with an R/W bit, the data will follow R/W bit specified to transfer. Each transferred byte is followed by an acknowledge bit on the 9th SCL clock cycle. If the slave signals a Not Acknowledge (NACK), the master can generate a STOP signal to abort the data transfer or generate a Repeated START signal and start a new transfer cycle.

  If the master, as a receiving device, does Not Acknowledge (NACK) the slave, the slave releases the SDA line for the master to generate a STOP or Repeated START signal. The Figure 6.21-5 and Figure 6.21-6 shows the waveform of bit transfer and acknowledge.

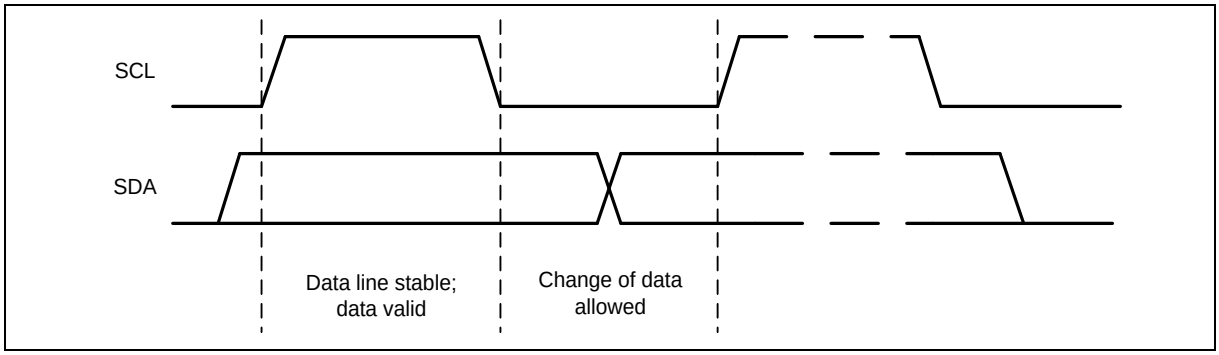


Figure 6.21-5 Bit Transfer on the I^{2}C Bus

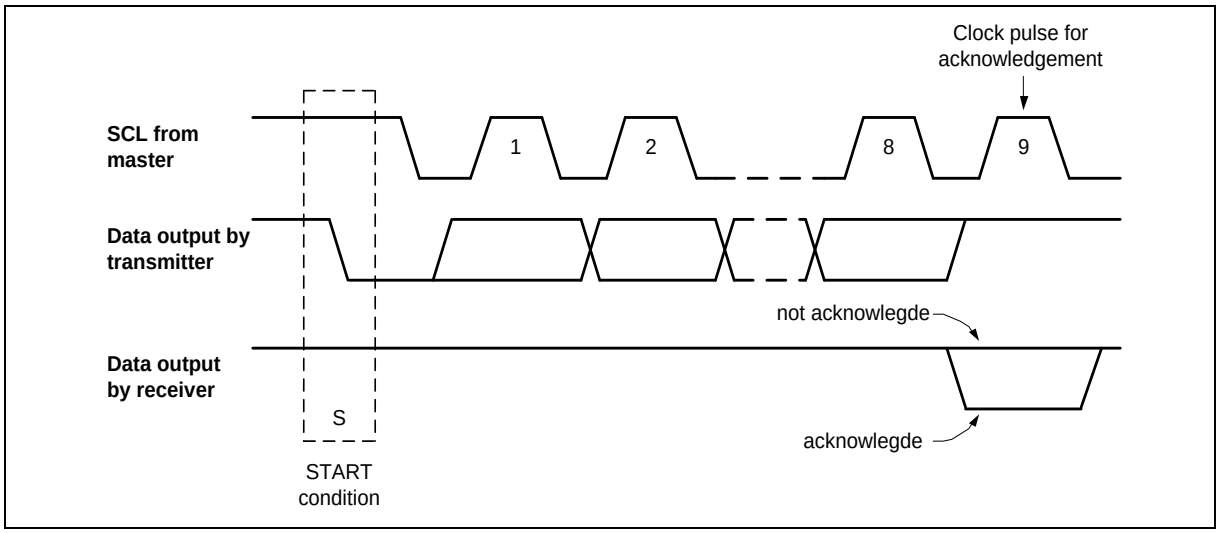


Figure 6.21-6 Acknowledge on the I^{2}C Bus

- Data transfer on I^{2}C bus

  Figure 6.21-7 shows a master transmits data to slave by 7-bit. A master addresses a slave with a 7-bit address and 1-bit write index to denote that the master wants to transmit data to the slave. The master keeps transmitting data after the slave returns acknowledge to the master.

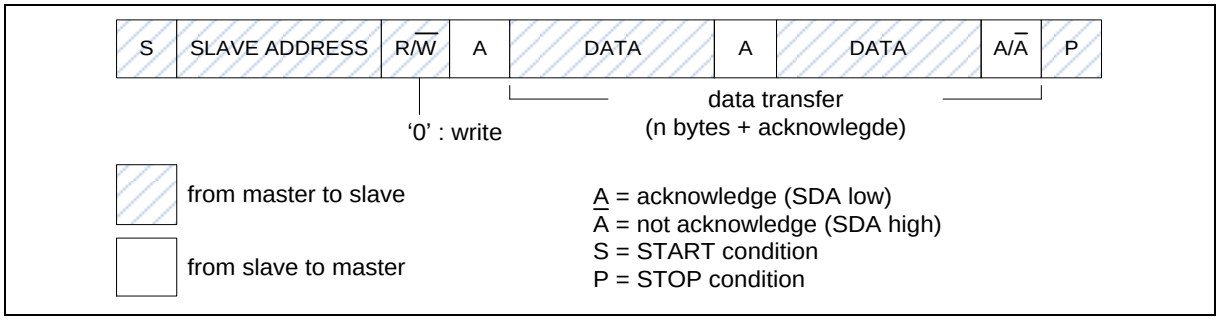


Figure 6.21-7 Master Transmits Data to Slave by 7-bit

Figure 6.21-8 shows a master read data from slave by 7-bit. A master addresses a slave with a 7-bit address and 1-bit read index to denote that the master wants to read data from the slave. The slave will start transmitting data after the slave returns acknowledge to the master.

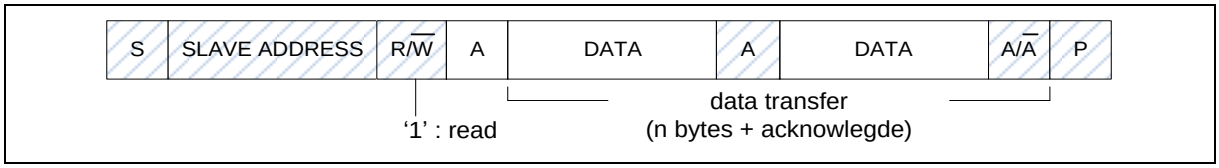


Figure 6.21-8 Master Reads Data from Slave by 7-bit

Figure 6.21-9 shows a master transmits data to slave by 10-bit. A master addresses a slave with a 10-bit address. First byte contains 10-bit address indicator (5'b11110) and 2-bit address with write index, second byte contains 8-bit address. The master keeps transmitting data after the second byte end. Note that 7-bit and 10-bit address device can work on the same bus.

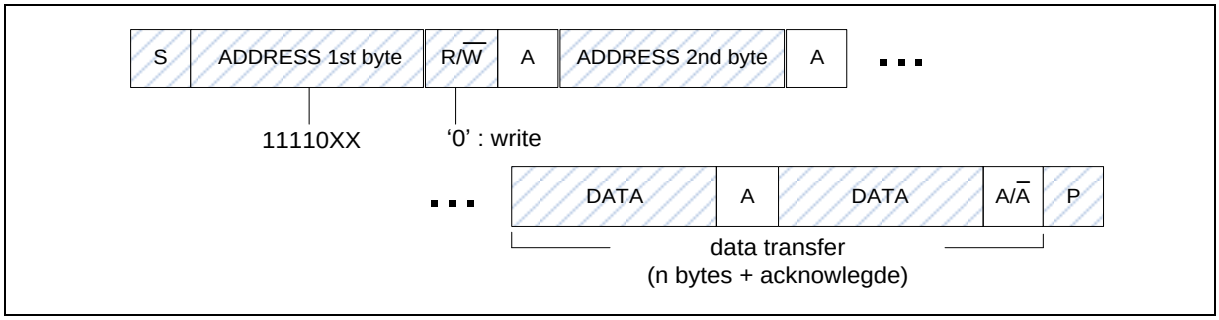


Figure 6.21-9 Master Transmits Data to Slave by 10-bit

Figure 6.21-10 shows a master read data from slave by 10-bit. A master addresses a slave with a 10-bit address. First mater transmits 10-bit address to slave, after that master transmits first byte with read index. The slave will start transmitting data after the first byte with read index.

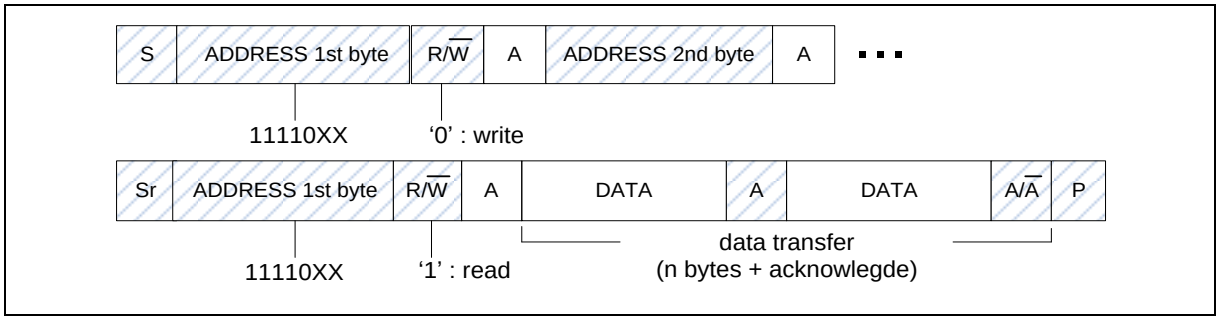


Figure 6.21-10 Master Reads Data from Slave by 10-bit

6.21.5.2 Operation Modes

The on-chip I²C ports support three operation modes, Master, Slave, and General Call Mode.

In a given application, I²C port may operate as a master or as a slave. In Slave mode, the I²C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master(by setting the AA bit), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, hardware waits until the bus is free before entering Master mode so that a possible slave action is not be interrupted. If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer.

To control the I²C bus transfer in each mode, user needs to set I2C_CTL0, I2C_DAT registers according to current status code of I2C_STATUS0 register. In other words, for each I²C bus action, user needs to check current status by I2C_STATUS0 register, and then set I2C_CTL0, I2C_DAT registers to take bus action. Finally, check the response status by I2C_STATUS0.

The bits, STA, STO and AA in I2C_CTL0 register are used to control the next state of the I²C hardware after SI flag of I2C_CTL0 [3] register is cleared. Upon completion of the new action, a new status code will be updated in I2C_STATUS0 register and the SI flag of I2C_CTL0 register will be set. But the SI flag will not be set when I²C STOP. If the I²C interrupt control bit INTEN (I2C_CTL0 [7]) is set, appropriate action or software branch of the new status code can be performed in the Interrupt service routine.

Figure 6.21-11 shows the current I²C status code is 0x08, and then set I2C_DATA=SLA+W and (STA,STO,SI,AA) = (0,0,1,x) to send the address to I²C bus. If a slave on the bus matches the address and response ACK, the I2C_STATUS0 will be updated by status code 0x18.

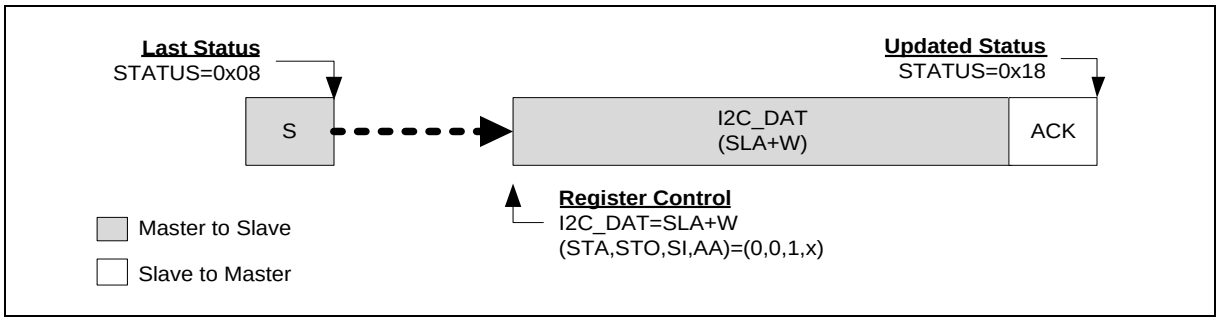


Figure 6.21-11 Control I²C Bus according to the Current I²C Status

Master Mode

In Figure 6.21-12 and Figure 6.21-13, all possible protocols for I²C master are shown. User needs to follow proper path of the flow to implement required I²C protocol.

In other words, user can send a START signal to bus and I²C will be in Master Transmitter (MT) mode (Figure 6.21-12) or Master receiver (MR) mode (Figure 6.21-13) after START signal has been sent

successfully and new status code would be 0x08. Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I²C protocol.

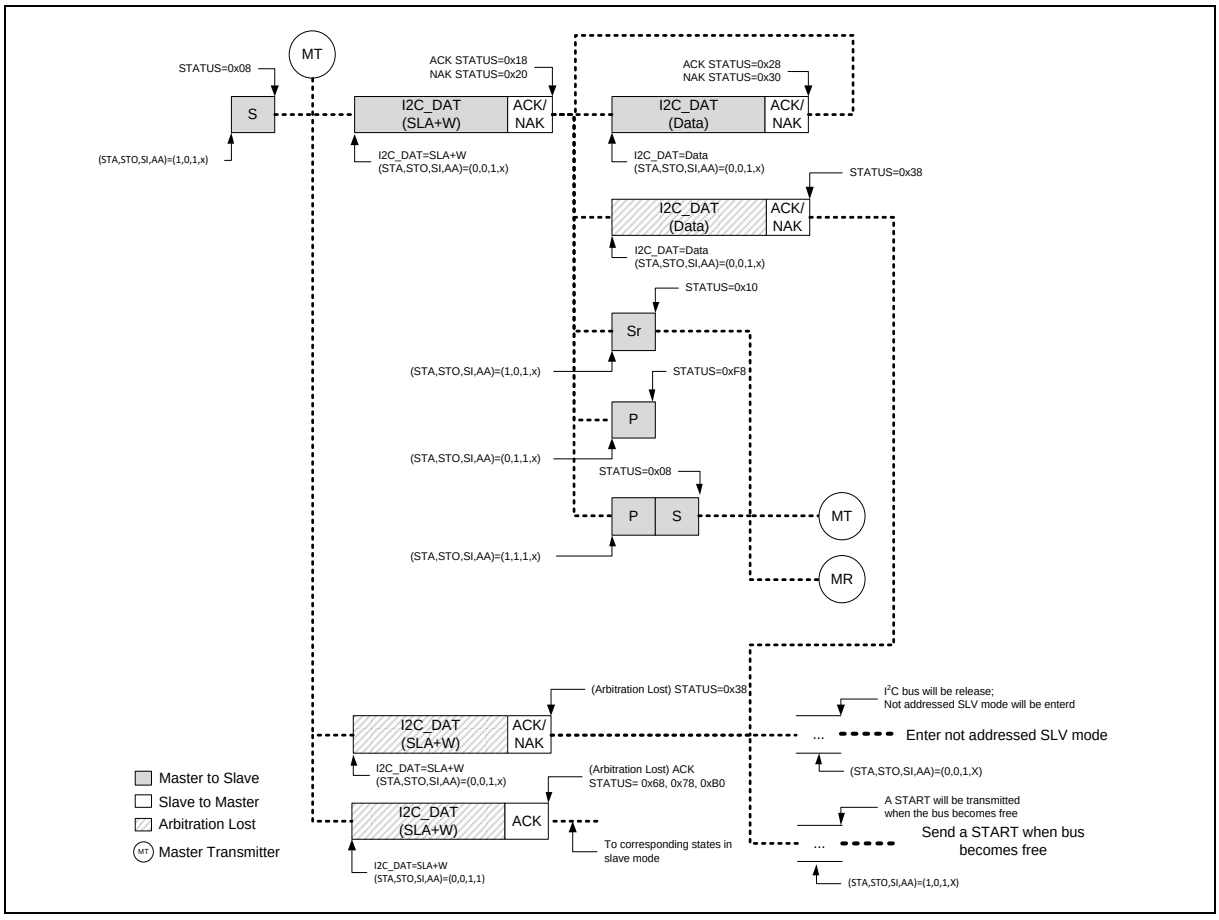


Figure 6.21-12 Master Transmitter Mode Control Flow

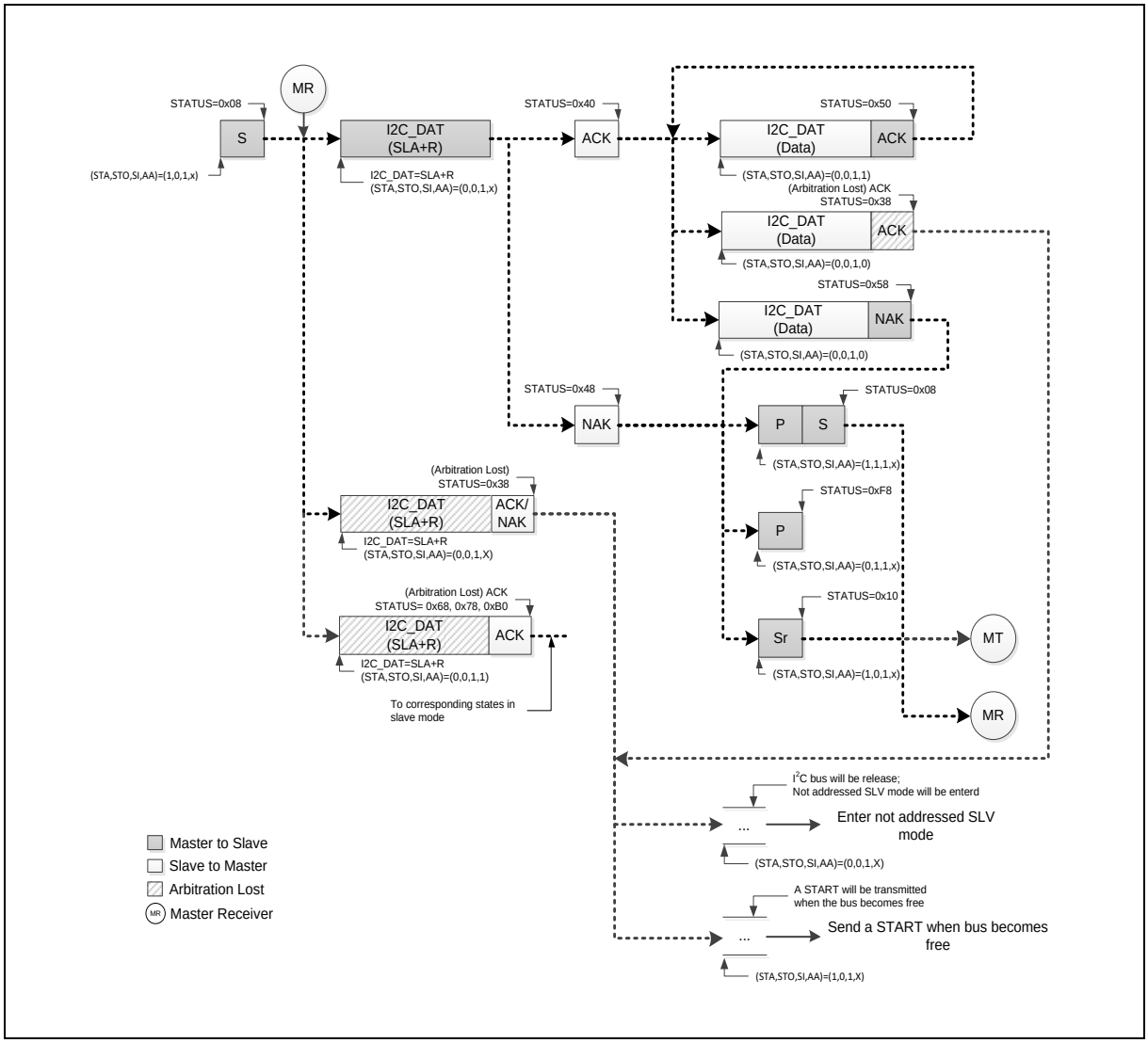


Figure 6.21-13 Master Receiver Mode Control Flow

If the I²C is in Master mode and gets arbitration lost, the status code will be 0x38. In status 0x38, user may set (STA, STO, SI, AA) = (1, 0, 1, X) to send START to re-start Master operation when bus become free. Otherwise, user may set (STA, STO, SI, AA) = (0, 0, 1, X) to release I²C bus and enter not addressed Slave mode.

Slave Mode

When reset default, I²C is not addressed and will not recognize the address on I²C bus. User can set slave address by I2C_ADDRn (n=0~3) and set (STA, STO, SI, AA) = (0, 0, 1, 1) to let I²C recognize the address sent by master. Figure 6.21-14 shows all the possible flow for I²C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.21-14 to implement their own I²C protocol.

If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) after arbitration lost, the status code is 0x68. If the detected address is SLA+R (Master want to read data from Slave) after arbitration lost, the status code is 0xB0.

Note: During I²C communication, the SCL clock will be released when writing '1' to clear SI flag in Slave mode.

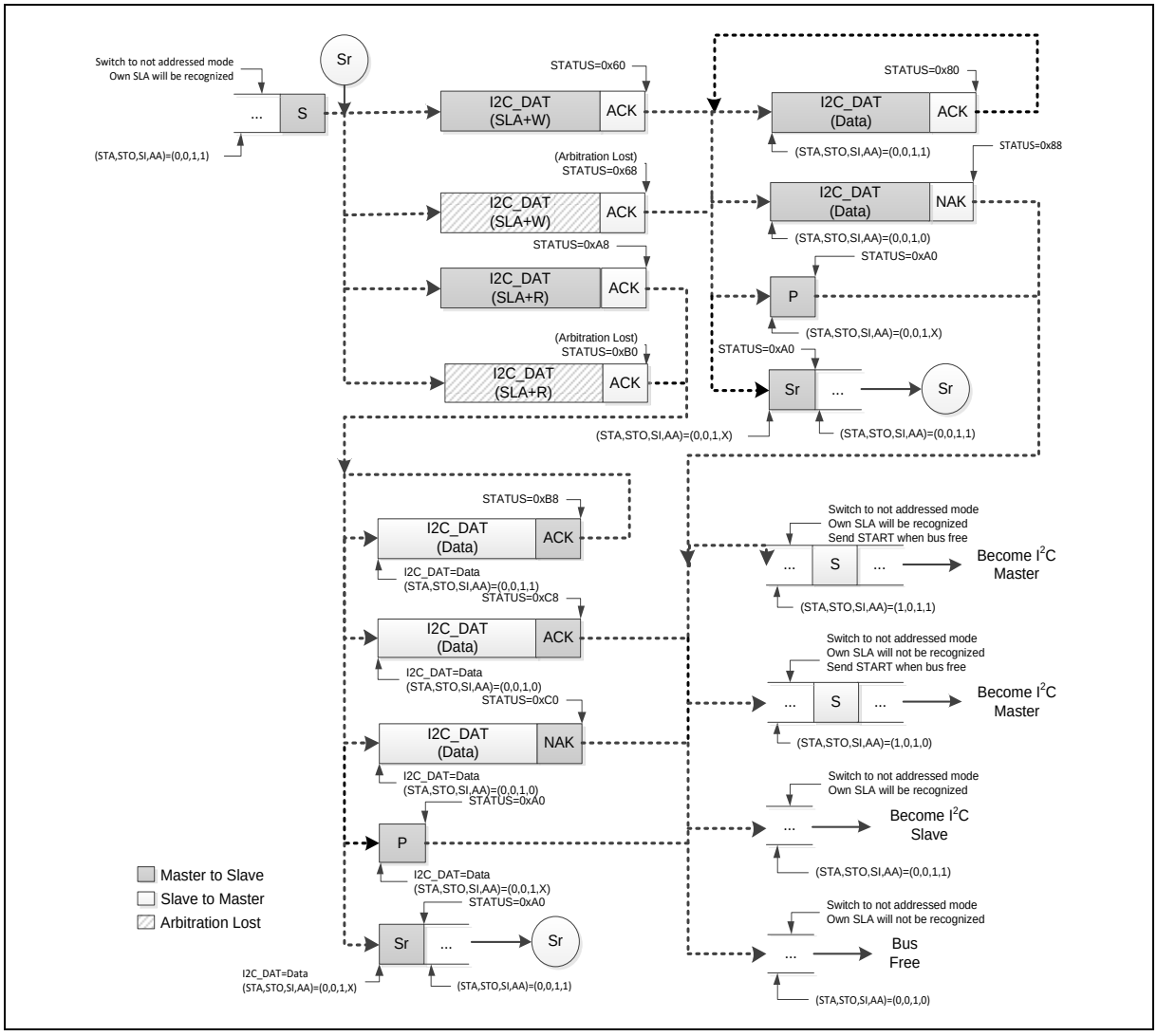


Figure 6.21-14 Slave Mode Control Flow

If I²C is still receiving data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x88 as shown in the above figure when getting 0xA0 status.

If I²C is still transmitting data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0xC8 as shown in the above figure when getting 0xA0 status.

Note: After slave gets status of 0x88, 0xC8, 0xC0 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I²C signal or address from master. At this status, I²C should enter idle mode.

General Call (GC) Mode

If the GC bit (I2C_ADDRn [0]) is set, the I²C port hardware will respond to General Call address (00H). User can clear GC bit to disable general call function. When the GC bit is set and the I²C in Slave mode, it can receive the general call address by 0x00 after master send general call address to I²C bus, then it will follow status of GC mode.

The GC mode can wake up when address matched. Note that the default address is 0x00, but user must set an address except for 0x00.

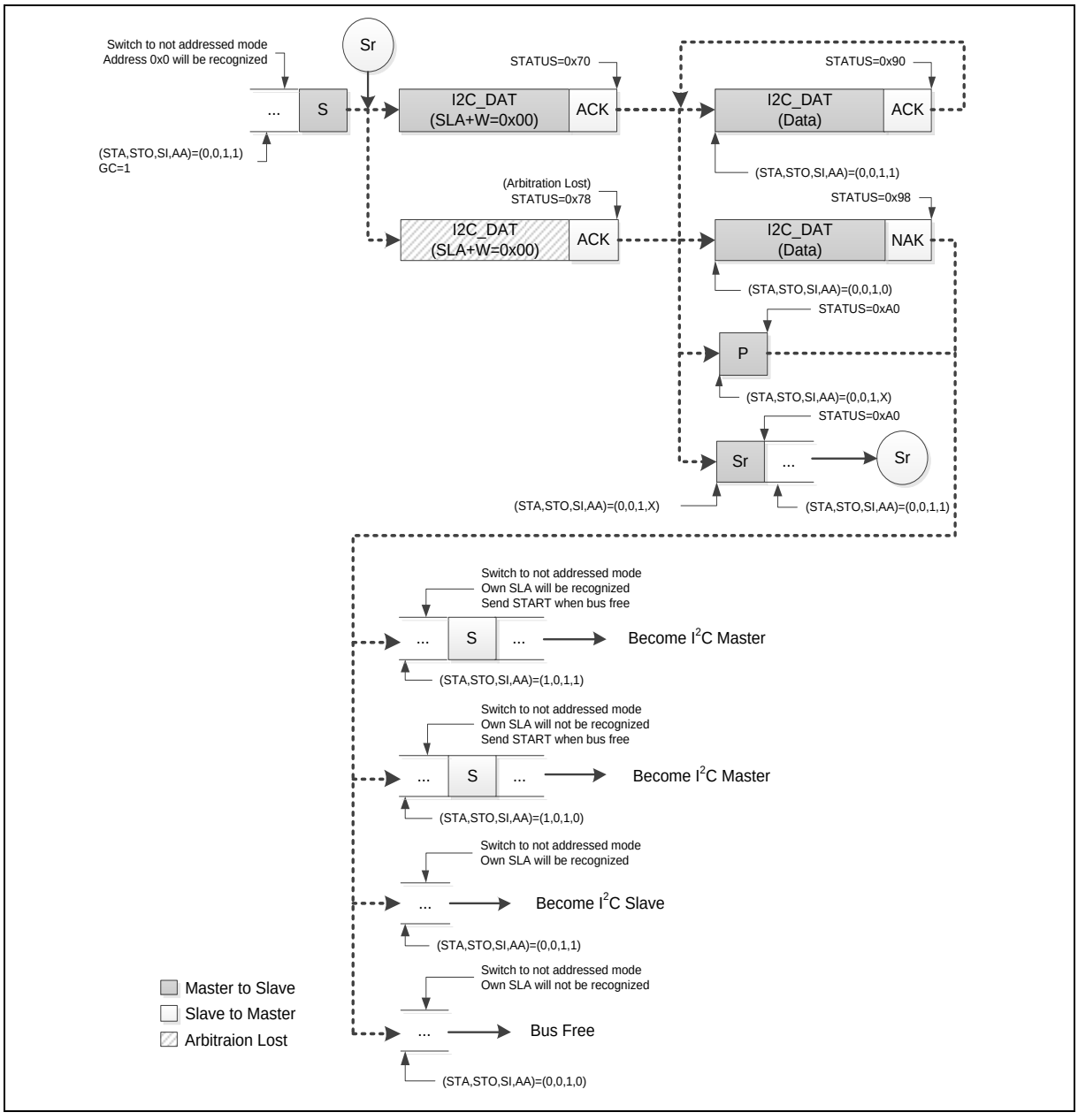


Figure 6.21-15 GC Mode

If I²C is still receiving data in GC mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x98 in above figure when getting 0xA0 status.

Note: After slave gets status of 0x98 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I²C signal or address from master. At this time, the I²C controller should enter idle mode.

Multi-Master

In some applications, there are two or more masters on the same I²C bus to access slaves, and the masters may transmit data simultaneously. The I²C supports multi-master by including collision detection and arbitration to prevent data corruption.

If for some reason two masters initiate command at the same time, the arbitration procedure determines which master wins and can continue with the command. Arbitration is performed on the SDA signal while the SCL signal is high. Each master checks if the SDA signal on the bus corresponds to the generated SDA signal. If the SDA signal on the bus is low but it should be high, then this master has lost arbitration. The device that has lost arbitration can generate SCL pulses until the byte ends and must then release the bus and go into slave mode. The arbitration procedure can continue until all the data is transferred. This means that in multi-master system each master must monitor the bus for collisions and act accordingly.

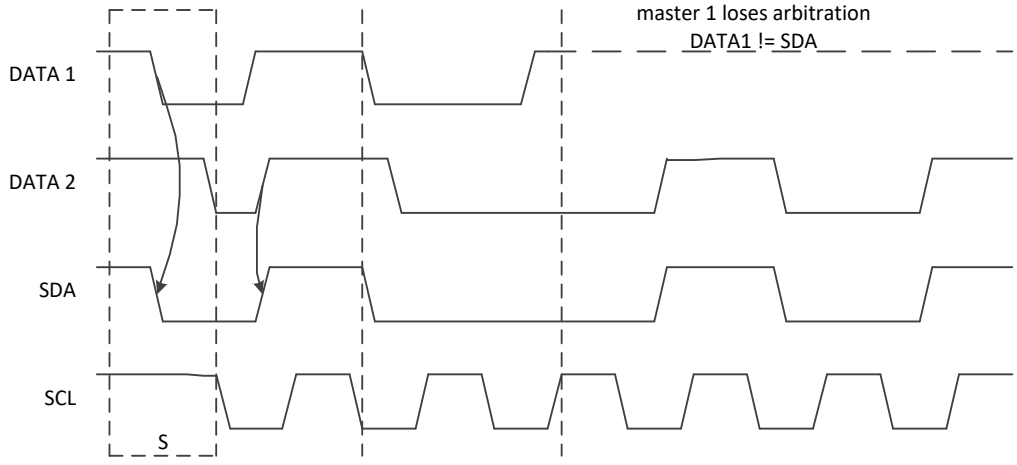


Figure 6.21-16 Arbitration Lost

- When I2C_STATUS0 = 0x38, an “Arbitration Lost” is received. Arbitration lost event maybe occur during the send START bit, data bits or STOP bit. User could set (STA, STO, SI, AA) = (1, 0, 1, X) to send START again when bus free, or set (STA, STO, SI, AA) = (0, 0, 1, X) to not addressed Slave mode. User can detect bus free by ONBUSY (I2C_STATUS1 [8]).
- When I2C_STATUS0 = 0x00, a “Bus Error” is received. To recover I^2C bus from a bus error, STO should be set and SI should be cleared, and then STO is cleared to release bus.
  - Set (STA, STO, SI, AA) = (0, 1, 1, X) to stop current transfer
  - Set (STA, STO, SI, AA) = (0, 0, 1, X) to release bus

Bus Management (SMBus/PMBus Compatible)

This section is relevant only when Bus Management feature is supported.

Introduction

The Bus Management is an I^2C interface through which various devices can communicate with each other and with the rest of the system. It is based on I^2C principles of operation. The Bus Management provides a control bus for system and power management related tasks.

This peripheral is compatible with the SMBUS Specification Rev 2.0 (http://smbus.org/specs/) and PMBUS Specification Rev 1.2 (http://pmbus.org/).

The System Management Bus Specification refers to three types of devices.

- A slave is a device that receives or responds to a command.

- A master is a device that issues commands, generates the clocks and terminates the transfer.
- A host is a specialized master that provides the main interface to the system's CPU. A host must be a master-slave and must support the SMBus host notify protocol. Only one host is allowed in a system.

This Bus Management peripheral is based on I^2C Specification Rev 2.1.

Device Identification – Slave Address

Any device that exists on the Bus Management as a slave has a unique address called the Slave Address. For reference, the following addresses are reserved and must not be used by or assign to any Bus Management device. (Refer to SMBus Specification for detailed information)

Slave Address Bits 7-1	R/W Bit Bit 0	Comment
0000 000	0	General Call Address
0000 000	1	START byte
0000 001	X	CBUS address
0000 010	X	Address reserved for different bus format
0000 011	X	Reserved for future use
0000 1XX	X	Reserved for future use
0101 000	X	Reserved for ACCESS.bus host
0110 111	X	Reserved for ACCESS.bus default address
1111 0XX	X	10-bit slave addressing
1111 1XX	X	Reserved for future use
0001 000	X	SMBus Host
0001 100	X	SMBus Alert Response Address
1100 001	X	SMBus Device Default Address

Table 6.21-1 Reserved SMBus Address

Bus Protocols

There are eleven possible command protocols for any given device. A device may use any or all of the eleven protocols to communicate. The protocols are Quick Command, Send Byte, Receive Byte, Write Byte, Write Word, Read Byte, Read Word, Process Call, Block Read, Block Write and Block Write-Block Read Process Call. These protocols should be implemented by the user software. (For more details of these protocols, refer to SMBus Specification Ver. 2.0)

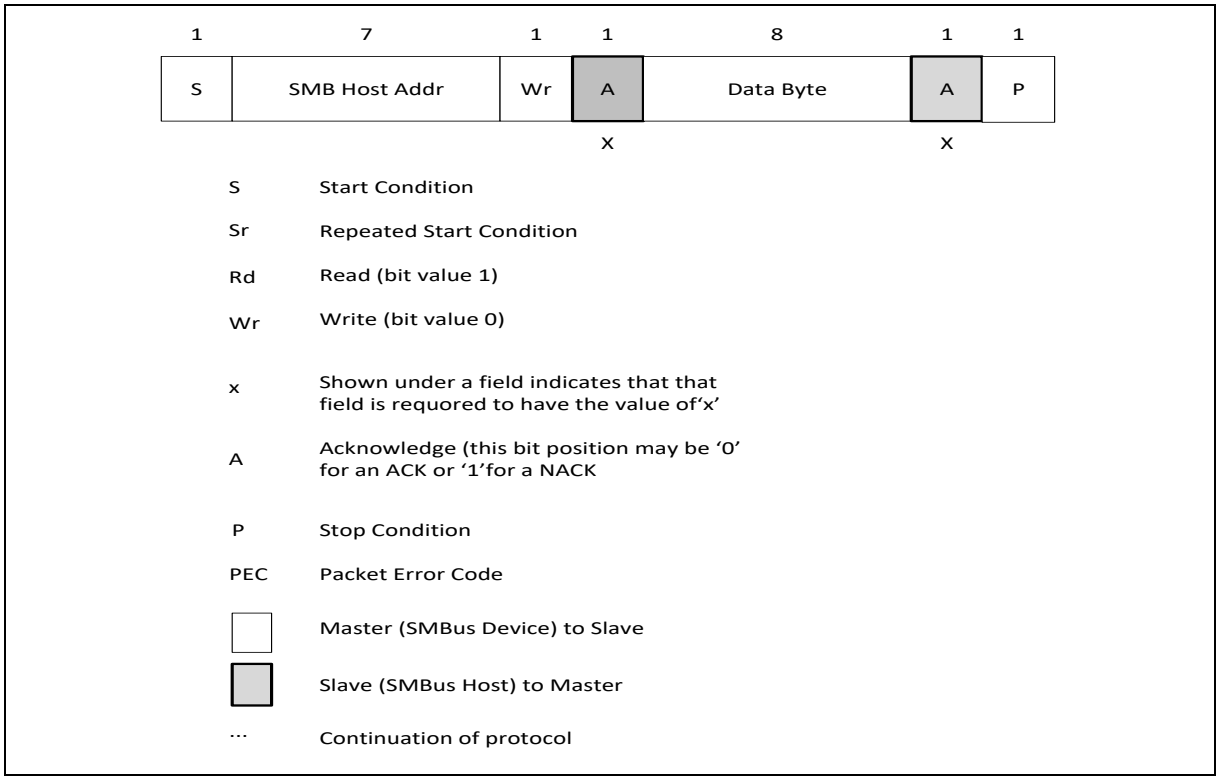


Figure 6.21-17 Bus Management Packet Protocol Diagram Element Key

Address Resolution Protocol (ARP)

Bus Management slave address conflicts can be resolved by dynamically assigning a new unique address to each slave device. In order to provide a mechanism to isolate each device for the purpose of address assignment each device must implement a unique device identifier (UDID). This 128-bit number is implemented by software.

This peripheral supports the Address Resolution Protocol (ARP). The Bus Management Device Default Address (0b1100 001) is enabled by setting BUSEN (I2C_BUSCTL[7]), BMDEN (I2C_BUSCTL[2]) and ALERTEN (I2C_BUSCTL[4]) bits. The ARP commands should be implemented by the user software. Arbitration is also performed in slave mode for ARP support.

Received Command and Data acknowledge control

A Bus Management receiver must be able to NACK each received command or data. In order to allow the ACK control in slave mode, the Slave Byte Control mode must be enabled by setting ACKMEN bit (I2C_BUSCTL[0]).

Host Notify Protocol

To prevent message coming to the Bus Management host controller from unknown devices in unknown formats only one method of communication is allowed, a modified form of the Write Word protocol. The standard Write Word protocol is modified by replacing the command code with the alerting device's address.

This peripheral supports the Host Notify protocol by setting the BUSEN (I2C_BUSCTL[7]), BMHEN (I2C_BUSCTL[3]) and ALERTEN (I2C_BUSCTL[4]). In this case the host will acknowledge the Bus Management Host address (0b0001000). This protocol is used when the device acts as a master and the host as a slave.

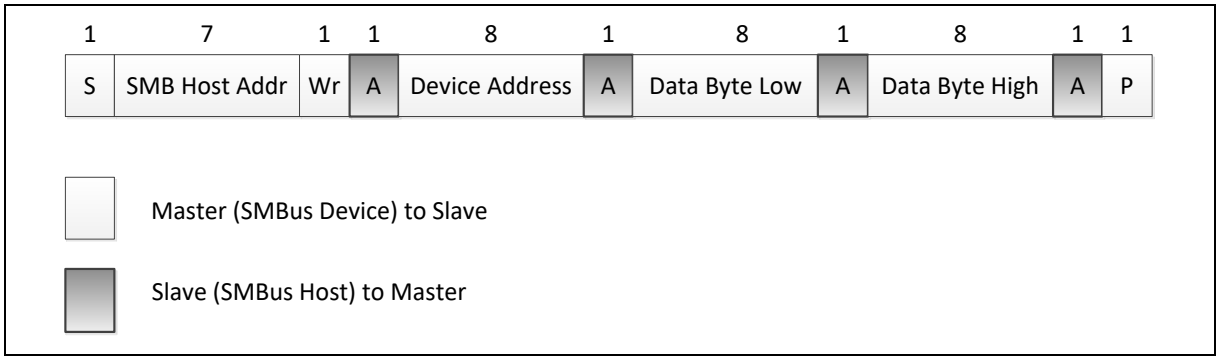


Figure 6.21-187-bit Addressable Device to Host Communication

Bus Management Alert

The Bus Management ALERT optional signal is supported. A slave-only device can signal the host through the Bus Management ALERT pin that it wants to talk. The host processes the interrupt and simultaneously accesses all Bus Management ALERT pin's devices through the Alert Response Address (0b0001 100). Only the device(s) which pulled Bus Management ALERT pin low will acknowledge the Alert Response Address.

When configured as a slave device(BMHEN=0), the Bus Management ALERT pin is pulled low by setting the ALERTEN bit (I2C_BUSCTL[4]). The Alert Response Address (ARA) is enabled at the same time.

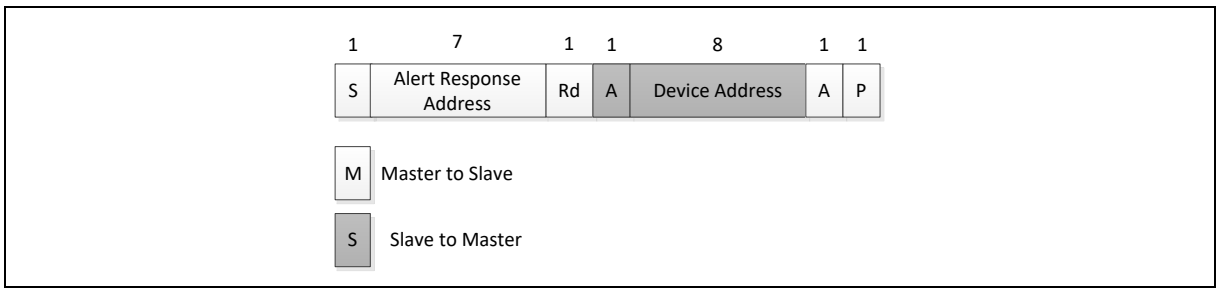


Figure 6.21-197-bit Addressable Device Responds to an ARA

When configured as a host (BMHEN=1), the ALERT flag (I2C_BUSSTS[3]) is set when a falling edge is detected on the Bus Management ALERT pin and ALERTEN=1. When ALERTEN=0, the ALERT line is considered high even if the external Bus Management ALERT pin is low. If the Bus Management ALERT pin is not needed, the Bus Management ALERT pin can be used as a standard GPIO if ALERTEN = 0;

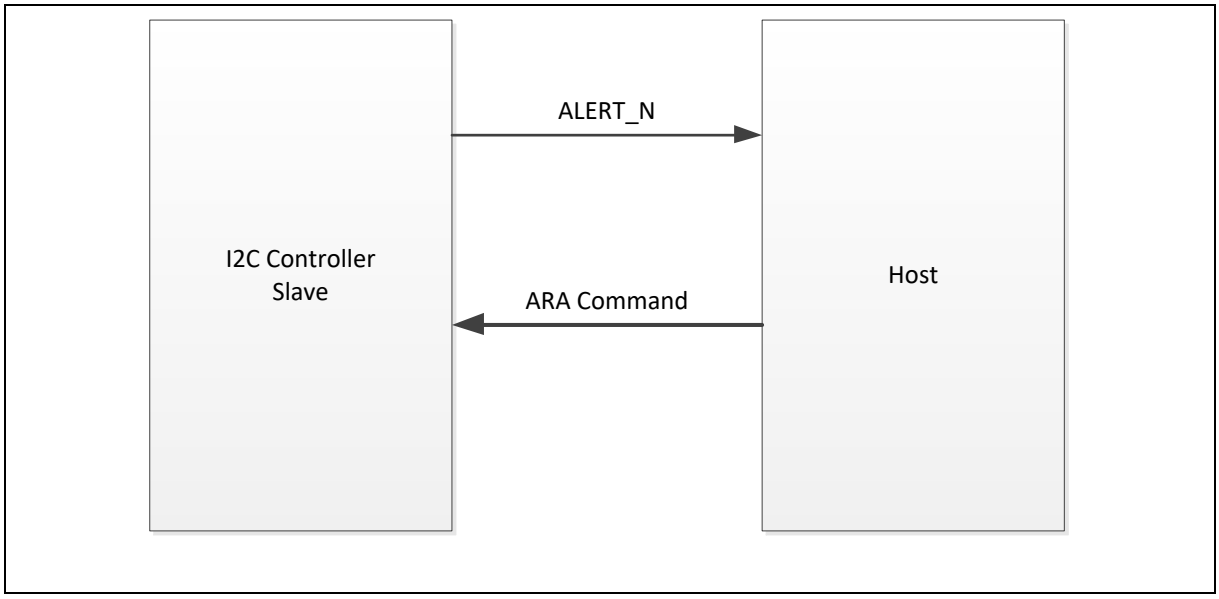


Figure 6.21-20 Bus Management ALERT Function

Packet Error Checking

A packet error checking mechanism has been introduced in the SMBus Specification to improve reliability and communication robustness. Packet Error Checking is implemented by appending a Packet Error Code (PEC) at the end of each message transfer. The PEC is calculated by using the C(x) = x8 + x2 + x + 1 CRC-8 polynomial on all the message bytes (including addresses and read/write bits).

The peripheral embeds a hardware PEC calculator when the PECEN bit (I2C_BUSCTL[1]) is set and allows to send a Not Acknowledge automatically when the received byte does not match with the hardware calculated PEC. The calculated value of PEC also can be read back on I2C_PKTCRC.

Time-out

This peripheral embeds hardware timers in order to be compliant with the 3 time-outs defined in SMBus Specification Ver. 2.0.

Bus Management Time-out:

The SCL low time-out condition when bus no IDLE

$T_{Time\text{-}out}$ = (BUSTO(I2C_BUSTOUT[7:0]) +1) x 16x1024 (14-bit) x T_{PCLK} (if TOCDIV4 = 0).

= (BUSTO(I2C_BUSTOUT[7:0])+1) x 16x1024 (14-bit) x 4 x T_{PCLK} (if TOCDIV4 = 1)

The bus idle condition (both SCL and SDA high) when bus IDLE

$T_{Time\text{-}out}$ = (BUSTO(I2C_BUSTOUT[7:0])+ 1) x 4 x T_{PCLK}.

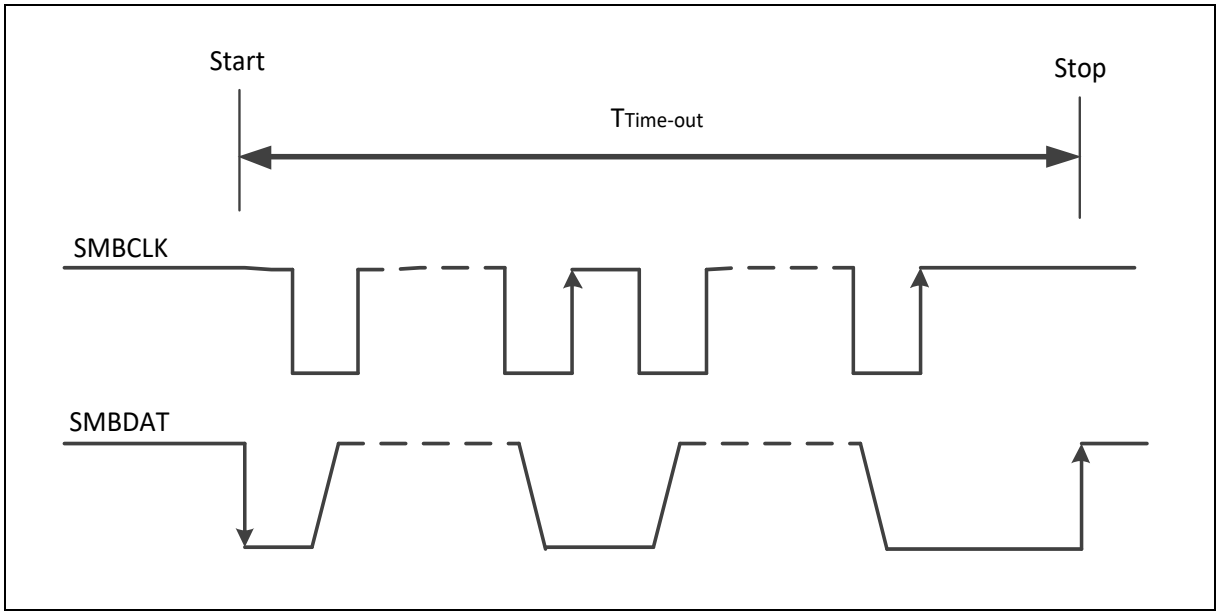


Figure 6.21-21 Bus Management Time Out Timing

Bus Clock Low Time-out:

In Master mode, the Master cumulative clock low extend time ($T_{LOW:MEXT}$) is detected

In Slave mode, the slave cumulative clock low extend time ($T_{LOW:SEXT}$) is detected

$T_{TLOW:EXT}$ = (CLKTO (I2C_CLKTOUT[7:0])+1) x 16x1024 (14-bit) x T_{PCLK} (if TOCDIV4= 0).

= (CLKTO (I2C_CLKTOUT[7:0])+1) x 16x1024 (14-bit) x 4 x T_{PCLK} (if TOCDIV4= 1)

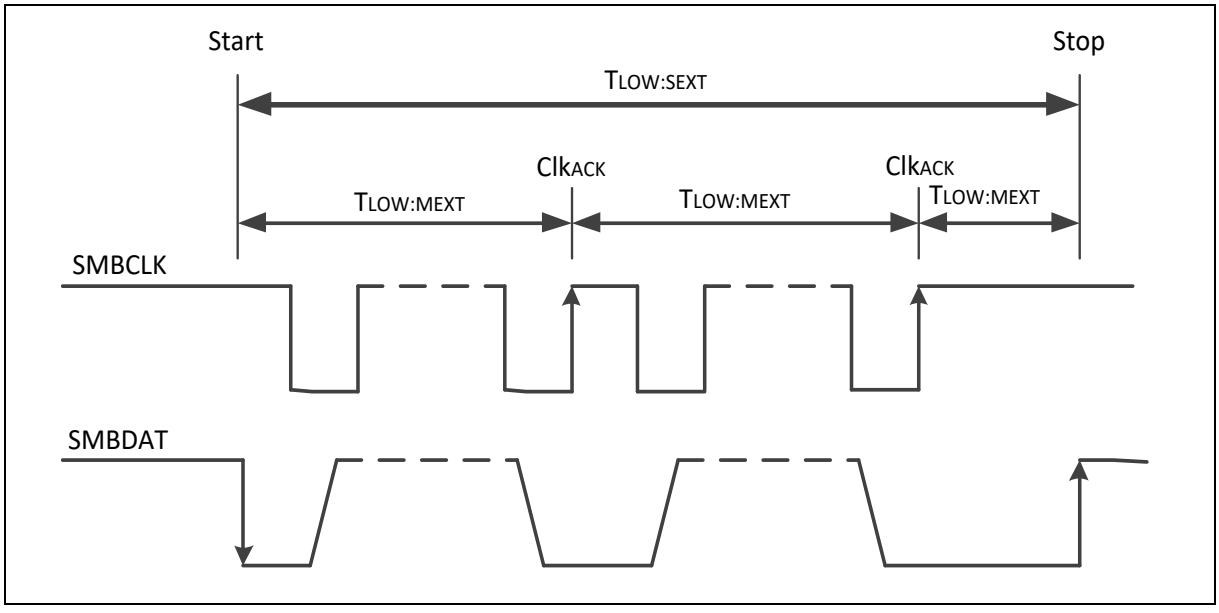


Figure 6.21-22 Bus Clock Low Time Out Timing

Bus Idle Detection

A master can assume that the bus is free if it detects that the clock and data signals have been high for T_{IDLE} greater than $T_{HIGH,MAX}$.

This timing parameter covers the condition where a master has been dynamically added to the bus and

may not have detected a state transition on the SMBCLK or SMBDAT lines. In this case, the master must wait long enough to ensure that a transfer is not currently in progress. The peripheral supports a hardware bus idle detection.

6.21.5.3 PDMA Transfer Function

The I²C controller supports PDMA transfer function. When TXPDMAEN (I2C_CTL1 [0]) is set to 1, the I²C controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (I2C_CTL1 [1]) is set to 1, the I²C controller will start the receive PDMA process. The I²C controller will issue the request to PDMA controller automatically when there is data written into the received BUFFER.

When I²C enters PDMA mode, the mostly status interrupt will be masked. Let the interrupt not occur besides the bus error or NACK or STOP interrupt (0x20, 0x30, 0x38, 0x48, 0x58, 0x00, 0xA0, 0xC0, 0x88 and 0x98).

Set the PDMASTR (I2C_CTL1 [8]) only the I²C controller in master TX mode. If PDMASTR is cleared to 0, I²C will send STOP automatically after PDMA transfer done and buffer empty. If PDMASTR is set to 1, SI will be set to 1 and I²C bus will be stretched by hardware after PDMA transfer done and buffer empty.

6.21.5.4 Two-level Buffer Mode on I²C bus

Set TWOBUFEN = 1 (I2C_CTL1[5]) to enable the two-level Buffer for I²C transmitted or received buffer. It only support 7-bit slave mode and not support wakeup function. The function is used to improve transfer/receive performance.

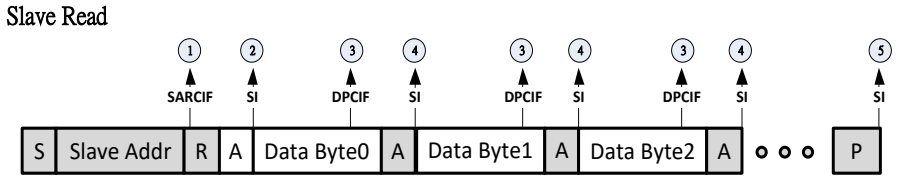
Two-level Buffer mode I²C will not stretch SCL, so user must enable SRCINTEN (I2C_CTL0[13]), DPCINTEN (I2C_CTL0[12]) and set DPBITSEL (I2C_CTL0[9:8]) for avoid data transfer errors.

SARCIF (I2C_CTL0[15]) will be set 1 by hardware when I²C slave detect address read command and SRCINTEN is enable. SARCIF (I2C_CTL0[15]) is used to send interrupt in advance to let software has enough time to prepare data.

After software enable DPCINTEN(I2C_CTL0[12]) and set DPBITSEL(I2C_CTL0[9:8]), DPCIF (I2C_CTL0[14]) will be set by hardware when slave transfer/receive data bit counts equal to the set up value. User can modify DPBITSEL(I2C_CTL0[9:8]) to adjust interrupt timing (DPCIF). The method is to use early entry handler to avoid missing interrupt events.

For example, one of circumstance is master send repeat start after write data ack, the two event will send interrupt very closely. Software may miss the write data ack status then slave will lose the last data transferred by master.

To ensure transfer/receive data correctly in Two-level Buffer mode (non-stretch mode), user must follow bellow control flow in two buffer mode.



IRQ handler control :

If SARCIF = 1

Slave address read frame: ①

Step 1 : write data0 to I2C_DAT

Step 2 : Wait until SI is set to 1

Step 3 : write data1 to I2C_DAT(read status before clear SI, SARCIF) ②

if SARCIF = 0

if DPCIF = 1

data frame: ③

Step 4 : Wait for data0 transfer until SI is set to 1

Step 5 : write data2 to I2C_DAT (read status before clear SI & DPCIF)

repeat step4 and step5 (write next transfer data to I2C_DAT after SI is high) ④

if DPCIF = 0

STOP frame: ⑤

Check I2C status before clear SI flag

Master to Slave

Slave to Master

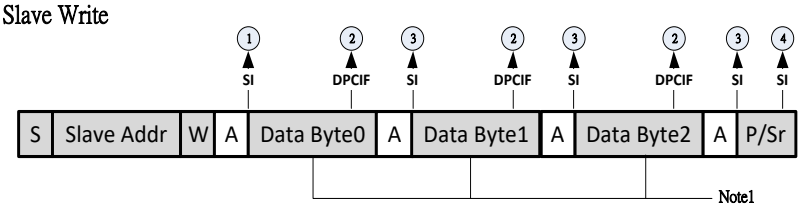
A = acknowledge (SDA low)

S = START condition

P = STOP condition

Figure 6.21-23 Two-level Buffer Mode Slave Read Control

Slave Write



IRQ handler control :
If DPCIF = 0
slave write command interrupt, STOP or Repeat START frame : ① & ④
Step 1 : check status I2C status before clear SI flag
If DPCIF = 1 ②
Data frame:
Step 2 : Wait until SI is set to 1
Step 3 : If status is 0x80 or 0x88, need to read the data(I2C_DAT) transmit from master ③
(read status before clear SI and DPCIF)
Repeat step2 and step3

Note 1 : DPCIF is used to prevent missing interrupt event. In data frame DPCIF will cause hardware assert interrupt in advance.(before SI)

Master to Slave
Slave to Master
S = START condition
Sr = Repeat START condition
P = STOP condition
A = acknowledge (SDA low)

Figure 6.21-24 Two-level Buffer Mode Slave Write Control

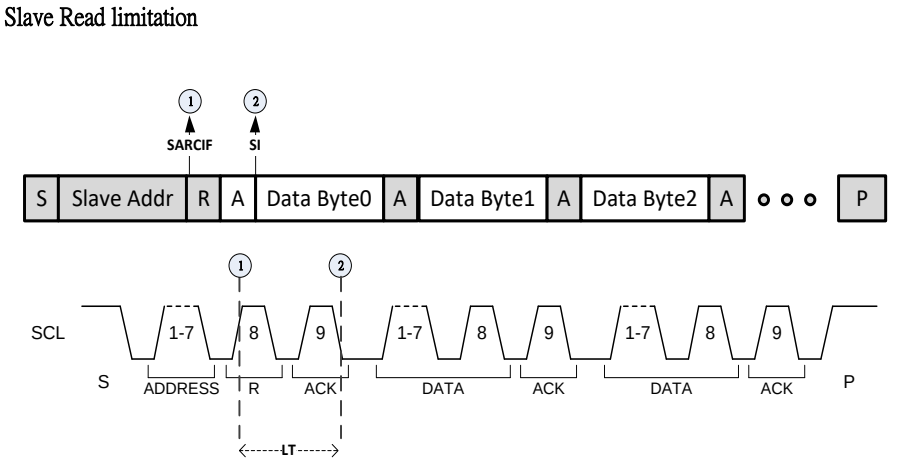


Figure 6.21-25 Two-level Buffer Mode Slave Read Limitation

Note: When two-level buffer is started up, PDMA cannot operate at the same time. PDMA and two-level buffer are only used independently.

6.21.5.5 Programmable Setup and Hold Times

To guarantee a correct data setup and hold time, the timing must be configured. By programming HTCTL (I2C_TMCTL[24:16]) to configure hold time and STCTL (I2C_TMCTL[8:0]) to configure setup time.

The delay timing refer peripheral clock (PCLK). When device stretch master clock, the setup and hold time configuration value will not affected by stretched.

User should focus the limitation of setup and hold time configuration, the timing setting must follow I^2C protocol. Once setup time configuration greater than design limitation, that means if setup time setting make SCL output less than three PCLKs, the I^2C controller can't work normally due to SCL must sample three times. And once hold time configuration greater than I^2C clock limitation, I^2C will occur bus error. It is suggested that user calculate suitable timing with baud rate and protocol before setting timing. Table 6.21-2 shows the relationship between I^2C baud rate and PCLK, the number of table represent one clock duty contain how many PCLKs. Setup and hold time configuration even can program some extreme values in the design, but user should follow I^2C protocol standard.

I²C Baud Rate PCLK	100k	200k	400k	800k	1200k
12 MHz	120	60	30	15	10
24 MHz	240	120	60	30	20
48 MHz	480	240	120	60	40
72 MHz	720	360	180	90	60

Table 6.21-2 Relationship between I²C Baud Rate and PCLK

For setup time wrong adjustment example, assuming one SCL cycle contains 5 PCLKs and set STCTL (I2C_TMCTL[8:0]) to 3 that stretch three PCLKs for setup time setting. The setup time maximum setting value: ST_{limit} = (I2C_CLKDIV[9:0]+1) X 2 - 6.

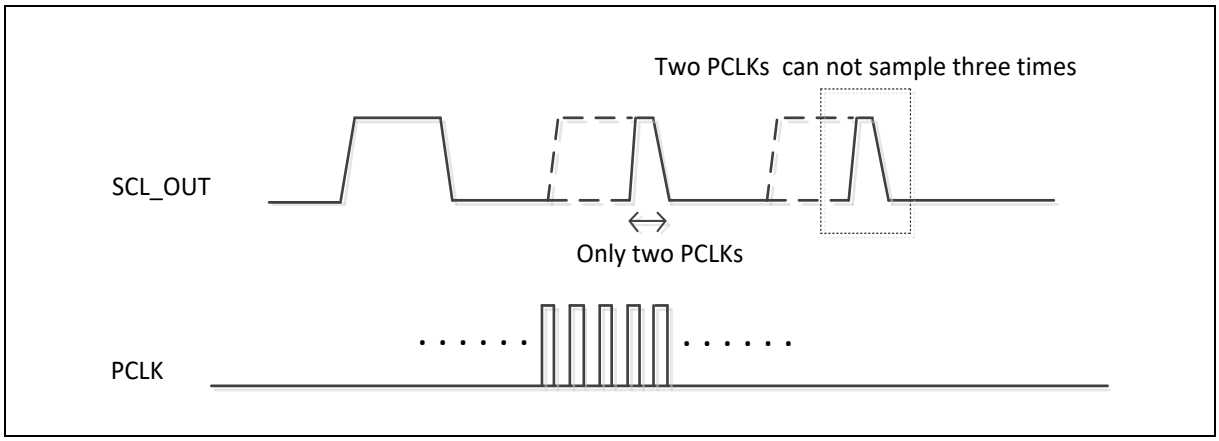


Figure 6.21-26 Setup Time Wrong Adjustment

For hold time wrong adjustment example, use I²C Baud Rate = 1200k and PCLK = 72 MHz, the SCL high/low duty = 60 PCLK. When HTCTL (I2C_TMCTL[24:16]) is set to 61 and STCTL (I2C_TMCTL[8:0]) is set to 0, then SDA output delay will over SCL high duty and cause bus error. The hold time maximum setting value: HT_{limit} = (I2C_CLKDIV[9:0]+1) X 2 - 9.

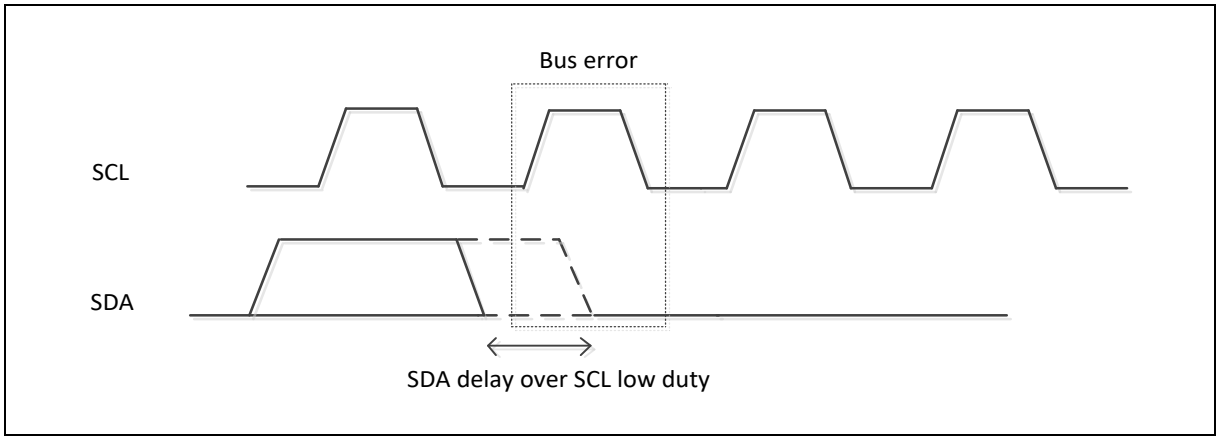


Figure 6.21-27 Hold Time Wrong Adjustment

6.21.5.6 I²C Protocol Registers

To control I²C port through the following fifteen special function registers: I2C_CTL0 (control register), I2C_STATUS0 (status register), I2C_DAT (data register), I2C_ADDRn (address registers, n=0~3),

I2C_ADDRMSKn (address mask registers, n=0~3), I2C_CLKDIV (clock rate register), I2C_TOCTL (Time-out control register), I2C_WKCTL(wake up control register) and I2C_WKSTS(wake up status register).

Address Registers (I2C_ADDR)

The I²C port is equipped with four slave address registers, I2C_ADDRn (n=0~3). The contents of the register are irrelevant when I²C is in Master mode. In Slave mode, the bit field ADDR(I2C_ADDRn[7:1]) must be loaded with the chip's own slave address. The I²C hardware will react if the contents of I2C_ADDRn are matched with the received slave address.

The I²C ports support the "General Call" function. If the GC bit (I2C_ADDRn [0]) is set the I²C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.

When the GC bit is set and the I²C is in Slave mode, it can receive the general call address by 00H after Master send general call address to I²C bus, then it will follow status of GC mode.

Slave Address Mask Registers (I2C_ADDRMSK)

The I²C bus controller supports multiple address recognition with four address mask registers I2C_ADDRMSKn (n=0~3). When the bit in the address mask register is set to 1, it means the received corresponding address bit is "Don't care". If the bit is set to 0, it means the received corresponding register bit should be exactly the same as address register.

Data Register (I2C_DAT)

This register contains a byte of serial data to be transmitted or a byte which just has been received. The CPU can be read from or written to the 8-bit (I2C_DAT [7:0]) directly while it is not in the process of shifting a byte. When I²C is in a defined state and the serial interrupt flag (SI) is set, data in I2C_DAT [7:0] remains stable. While data is being shifted out, data on the bus is simultaneously being shifted in; I2C_DAT [7:0] always contains the last data byte presented on the bus.

The acknowledge bit is controlled by the I²C hardware and cannot be accessed by the CPU. Serial data is shifted into I2C_DAT [7:0] on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into I2C_DAT [7:0], the serial data is available in I2C_DAT [7:0], and the acknowledge bit (ACK or NACK) is returned by the control logic during the ninth clock pulse. In order to monitor bus status while sending data, the bus data will be shifted to I2C_DAT[7:0] when sending I2C_DAT[7:0] to bus. In the case of sending data, serial data bits are shifted out from I2C_DAT [7:0] on the falling edge of SCL clocks, and is shifted to I2C_DAT [7:0] on the rising edge of SCL clocks. Figure 6.21-28 shows I²C Data Shifting Direction.

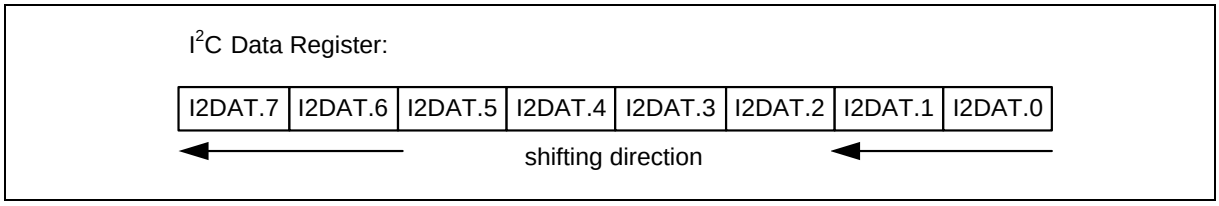


Figure 6.21-28 I²C Data Shifting Direction

Control Register (I2C_CTL0)

The CPU can be read from and written to I2C_CTL0 [7:0] directly. When the I²C port is enabled by setting I2CEN (I2C_CTL0 [6]) to high, the internal states will be controlled by I2C_CTL0 and I²C logic hardware.

There are two bits are affected by hardware: the SI bit is set when the I²C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when I2CEN = 0.

Once a new status code is generated and stored in I2C_STATUS0, the I²C Interrupt Flag bit SI (I2C_CTL0 [3]) will be set automatically. If the Enable Interrupt bit INTEN (I2C_CTL0 [7]) is set at this

time, the I²C interrupt will be generated. The bit field I2C_STATUS0[7:0] stores the internal state code, the content keeps stable until SI is cleared by software.

Status Register (I2C_STATUS0)

I2C_STATUS0 [7:0] is an 8-bit read-only register. The bit field I2C_STATUS0 [7:0] contains the status code and there are 26 possible status codes. All states are listed in Table 6.21-3. When I2C_STATUS0 [7:0] is F8H, no serial interrupt is requested. All other I2C_STATUS0 [7:0] values correspond to the defined I²C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2C_STATUS0[7:0] one cycle PCLK after SI set by hardware and is still present one cycle PCLK after SI reset by software.

In addition, the state 00H stands for a Bus Error, which occurs when a START or STOP condition is present at an incorrect position in the I²C format frame. A Bus Error may occur during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I²C from bus error, STO should be set and SI should be cleared to enter Not Addressed Slave mode. Then STO is cleared to release bus and to wait for a new communication. The I²C bus cannot recognize stop condition during this action when a bus error occurs.

Master Mode		Slave Mode	
STATUS	Description	STATUS	Description
0x08[1]	Start	0xA0	Slave Transmit Repeat Start or Stop
0x10[1]	Master Repeat Start	0xA8[1]	Slave Transmit Address ACK
0x18[1]	Master Transmit Address ACK	0xB8[1]	Slave Transmit Data ACK
0x20	Master Transmit Address NACK	0xC0	Slave Transmit Data NACK
0x28[1]	Master Transmit Data ACK	0xC8[1]	Slave Transmit Last Data ACK
0x30	Master Transmit Data NACK	0x60[1]	Slave Receive Address ACK
0x38	Master Arbitration Lost	0x68[1]	Slave Receive Arbitration Lost
0x40[1]	Master Receive Address ACK	0x80[1]	Slave Receive Data ACK
0x48	Master Receive Address NACK	0x88	Slave Receive Data NACK
0x50[1]	Master Receive Data ACK	0x70[1]	GC mode Address ACK
0x58	Master Receive Data NACK	0x78[1]	GC mode Arbitration Lost
0x00	Bus error	0x90[1]	GC mode Data ACK
		0x98	GC mode Data NACK
		0xB0[1]	Address Transmit Arbitration Lost
0xF0	If the BMDEN =1 and the ACKMEN bit is enabled, the information of I2C_STATUS0 will be fixed as 0xF0 in slave receive condition.		
0xF8	Bus Released **Note:** Status “0xF8” exists in both master/slave modes, and it won’t raise interrupt.		
Note [1]: No interrupt in PDMA mode			

Table 6.21-3 I²C Status Code Description

Clock Baud Rate Bits (I2C_CLKDIV)

The data baud rate of I²C is determines by DIVIDER(I2C_CLKDIV [9:0]) register when I²C is in Master mode, and it is not necessary in a Slave mode. In the Slave mode, I²C will automatically synchronize it with any clock frequency from master I²C device. In the slave mode, system clock frequency should

greater than I²C bus maximum clock 20 times.

The data baud rate of I²C setting is Data Baud Rate of I²C = (system clock) / (4x (I2C_CLKDIV [9:0] +1)). If system clock = 16 MHz, the I2C_CLKDIV [9:0] = 40 (28H), the data baud rate of I²C = 16 MHz/ (4x (40 +1)) = 97.5 kbit/s.

Time-out Control Register (I2C_TOCTL)

There is a 14-bit time-out counter which can be used to deal with the I²C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows (TOIF=1) and generates I²C interrupt to CPU or stops counting by clearing TOCEN to 0. When time-out counter is enabled, writing 1 to the SI flag will reset counter and re-start up counting after SI is cleared. If I²C bus hangs up, it causes the I2C_STATUS0 and flag SI are not updated for a period, the 14-bit time-out counter may overflow and acknowledge CPU the I²C interrupt. Refer to Figure 6.21-29 for the 14-bit time-out counter. User may write 1 to clear TOIF to 0.

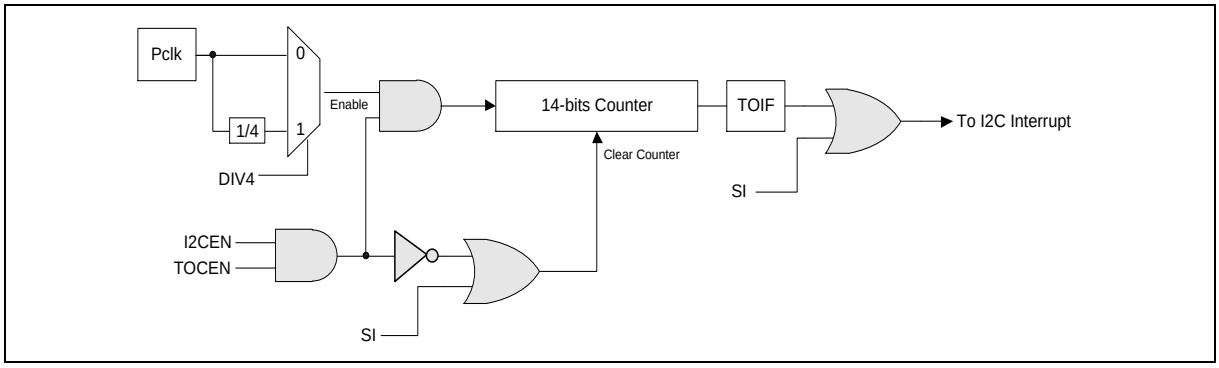


Figure 6.21-29 I²C Time-out Count Block Diagram

Wake-up Control Register (I2C_WKCTL)

When chip enters Power-down mode and set WKEN (I2C_WKCTL [0]) to 1, other I²C master can wake up the chip by addressing the I²C device, user must configure the related setting before entering sleep mode. The ACK bit cycle of address match frame is done in power-down. The controller will stretch the SCL to low when the address is matched the device's address and the ACK cycle done, then the I²C controller will go ahead. If NHDBUSEN (I2C_WKCTL [7]) is set, the controller will don't stretch the SCL to low. Note that when the controller don't stretch the SCL to low, transmit or receive data will perform immediately. If data transmitted or received when SI event is not clear, user must reset the I²C controller and execute the original operation again.

Wake-up Status Register (I2C_WKSTS)

When system is woken up by other I²C master device, WKIF is set to indicate this event. User needs write "1" to clear this bit.

When the chip is woken-up by address match with one of the device address register (I2C_ADDRn), the user shall check the WKAKDONE (I2C_WKSTS [1]) bit is set to 1 to confirm the address byte has done. The WKAKDONE bit indicates that the ACK bit cycle of address byte is done in power-down. The controller will stretch the SCL to low when the address is matched the device's slave address and the ACK cycle done. The SCL is stretched until WKAKDONE is clear by user. If the frequency of SCL is low speed and the system has wakeup from address match frame, the user shall check WKAKDONE to confirm this frame has transaction done and then to do the wakeup procedure. Note that user can't release WKIF through clearing the WKAKDONE bit to 0.

The WRSTSWK (I2C_WKSTS [2]) bit records the Read/Write command before the I²C controller sends address. The user can read this bit's status to prepare the next transmitted data (WRSTSWK = 1) or to wait the incoming data (WRSTSWK = 0) can be stored in time after the system is wake-up by the address match frame. Note that the WRSTSWK (I2C_WKSTS [2]) bit is cleared when writing one to the WKAKDONE (I2C_WKSTS [1]) bit.

When system is woken up by other I²C master device, WKIF is set to indicate this event. User needs to write “1” to clear this bit.

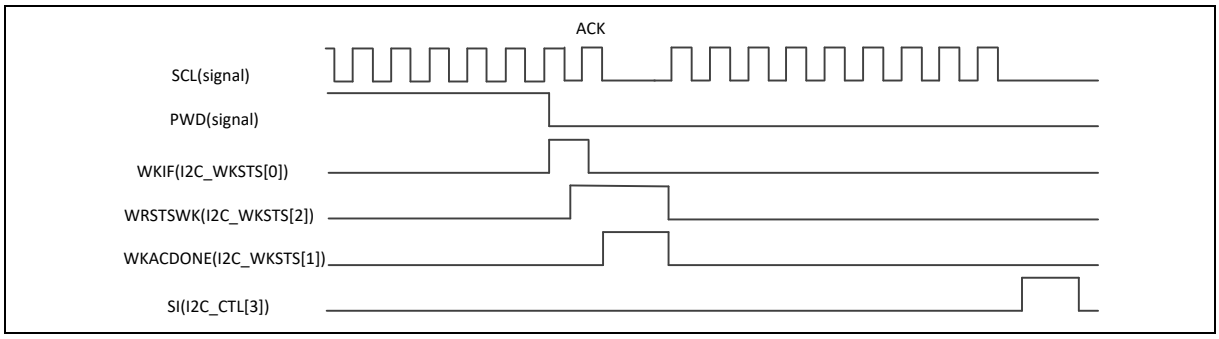


Figure 6.21-30 I²C Wake-Up Related Signals Waveform

I²C Control Register 1 (I2C_CTL1)

For the TWOBUFEN (I2C_CTL1[5]) bit, it is used to enable the two-level buffer for I²C transmitted or received buffer. It is used to improve the performance of the I²C bus.

Set UDRIEN (I2C_CTL1[4]) or OVRIEN (I2C_CTL1[3]) that interrupt will occur when two-level buffer generate the under run or over run event. It is worth noting when buffer empty, it will pull the bus. When I²C transmit or receive STOP, the data in buffer will be cleared. If user want to reset two-level buffer, then set the TWOBUFRST (I2C_CTL1 [6]) to clear buffer. Set NSTRETCH (I2C_CTL1[7]) that I²C SCL bus is not stretched by hardware if the SI is not cleared in master mode.

If enable 10-bit addressing mode ADDR10EN (I2C_CTL1 [9]) is set, the I²C will run in 10-bit mode.

For PDMA function, set TXPDMAEN (I2C_CTL1 [0]) and RXPDMAEN (I2C_CTL1 [1]) can be set to operate. And set PDMARST (I2C_CTL1 [2]) to reset the PDMA control logic.

I²C Status Register 1 (I2C_STATUS1)

The two-level buffer status, busy free information, EMPTY(I2C_STATUS1 [5]), FULL(I2C_STATUS1 [4]) and UDR(I2C_STATUS1 [7]) or OVR(I2C_STATUS1 [6]) are also list in the this register.

The I²C controller supports four slave address flag registers, ADMAT0, ADMAT1, ADMAT2 and ADMAT3 (I2C_STATUS1[3:0]). Every control register represent which address is used and set 1 to inform software.

I²C Timing Configure Control Register (I2C_TMCTL)

In order to configure setup/hold time, the HTCTL (I2C_TMCTL[24:16]) and STCTL (I2C_TMCTL[8:0]) are set based on actual demand.

Bus Management Control Register (I2C_BUSCTL)

The SM bus management control events are defined in this register. It includes the Acknowledge Control by Manual (ACKMEN (I2C_BUSCTL[0])), Packet Error Checking Enable (PECEN (I2C_BUSCTL[1])), device (BMDEN(I2C_BUSCTL[2])) or host (BMHEN (I2C_BUSCTL[3])) enable in this peripheral device. Both the alert and the suspend function can be set in ALERTEN (I2C_BUSCTL[4]), SCTLOSTS (I2C_BUSCTL[5])) and SCTLOEN (I2C_BUSCTL[6]).

The system bus management enables control by BUSEN(I2C_BUSCTL[7]) bit. The BUSTOUT (I2C_BUSCTL[9]) is used to calculate the time-out of clock low in bus active and the idle period in bus Idle.

The calculated PEC (when the PECEN is set) value is transmitted or received can be controlled by PECTXEN bit (I2C_BUSCTL[8]).

There is a special bit of ACKM9SI (I2C_BUSCTL[11]). When the ACKMEN is set, there is SI interrupt in the 8th clock input and the user can read the data and status register. If the 8th clock bus is released

when the SI interrupt is cleared, there is another SI interrupt event in the 9th clock cycle when this bit is set to 1 to know the bus status in this transaction frame done.

Set the PECDIEN (I2C_BUSCTL [13]), BCDIEN (I2C_BUSCTL[12]) or PECCLR (I2C_BUSCTL[10]) for PEC control flow.

I²C Bus Management Timer Control Register (I2C_BUSTCTL)

Set TORSTEN (I2C_BUSTCTL [4]), CLKTOIEN (I2C_BUSTCTL [3]), BUSTOIEN (I2C_BUSTCTL[2]), CLKTOEN (I2C_BUSTCTL [1]) and BUSTOEN (I2C_BUSTCTL [0]) for bus time-out or clock low time-out control flow.

I²C Bus Management Status Register (I2C_BUSSTS)

Monitor the PECDONE (I2C_BUSSTS [7]), BCDONE (I2C_BUSSTS[1]) or PECERR (I2C_BUSSTS[2]) for PEC control flow.

Monitor the SCTLDIN (I2C_BUSSTS[4]) for SUSCON input status.

I²C Byte Number Register (I2C_PKTSIZE)

When the PECEN bit (I2C_BUSCTL [1]) is set. The I²C controller will calculate the PEC value of the data on the bus. The PLDSIZE (I2C_PKTSIZE[8:0]) is used to define the data number in the bus. When the counter reach the value of PLDSIZE, the final PEC value will be transmitted or received automatically when the PECTXEN bit (I2C_BUSCTL[8]) is set.

I²C PEC VALUR Register (I2C_PKTCRC)

The register indicates the calculated PECCRC (I2C_PKTCRC[7:0]) value of data on the I²C bus. The detail of information is defined in the Bus Management (SMBus/PMBus Compatible).

I²C Bus Management Timer and I²C Clock Low Timer Register (I2C_BUSTOUT/ I2C_CLKTOUT)

Both of the definitions of these registers are described in the Bus Management (SMBus/PMBus Compatible).

6.21.5.7 Example for Random Read on EEPROM

The following steps are used to configure the I2C0 related registers when using I²C to read data from EEPROM.

1. Set I2C0 the multi-function pin as SCL and SDA pins. Refer to the Basic Configuration section for the multi-function configuration.
2. Enable I2C0 APB clock. Refer to the Basic Configuration section for the clock configuration.
3. Set I2C0RST=1 to reset I2C0 controller then set I2C0 controller to normal operation. Refer to the Basic Configuration section for the reset controller configuration.
4. Set I2CEN=1 to enable I2C0 controller in the “I2C_CTL0” register.
5. Give I2C0 clock a divided register value for I²C clock rate in the “I2C_CLKDIV”.
6. Enable system I2C0 IRQ in system “NVIC” control register.
7. Set INTEN=1 to enable I2C0 Interrupt in the “I2C_CTL0” register.
8. Set I2C0 address registers “I2C_ADDR0 ~ I2C_ADDR3”.

Random read operation is one of the methods of access EEPROM. The method allows the master to access any address of EEPROM space. Figure 6.21-31 shows the EEPROM random read operation.

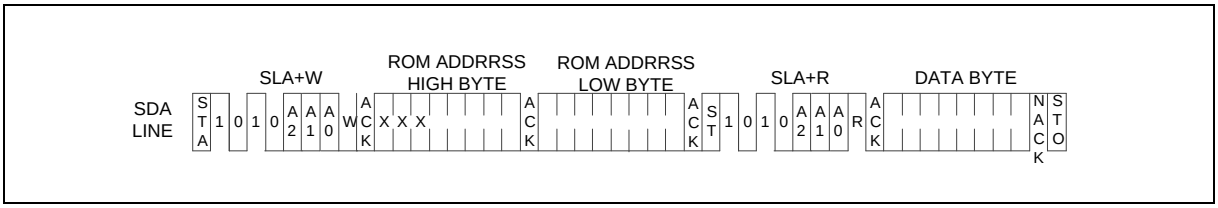


Figure 6.21-31 EEPROM Random Read

Figure 6.21-32 shows how to use the I²C controller to implement the protocol of EEPROM random read.

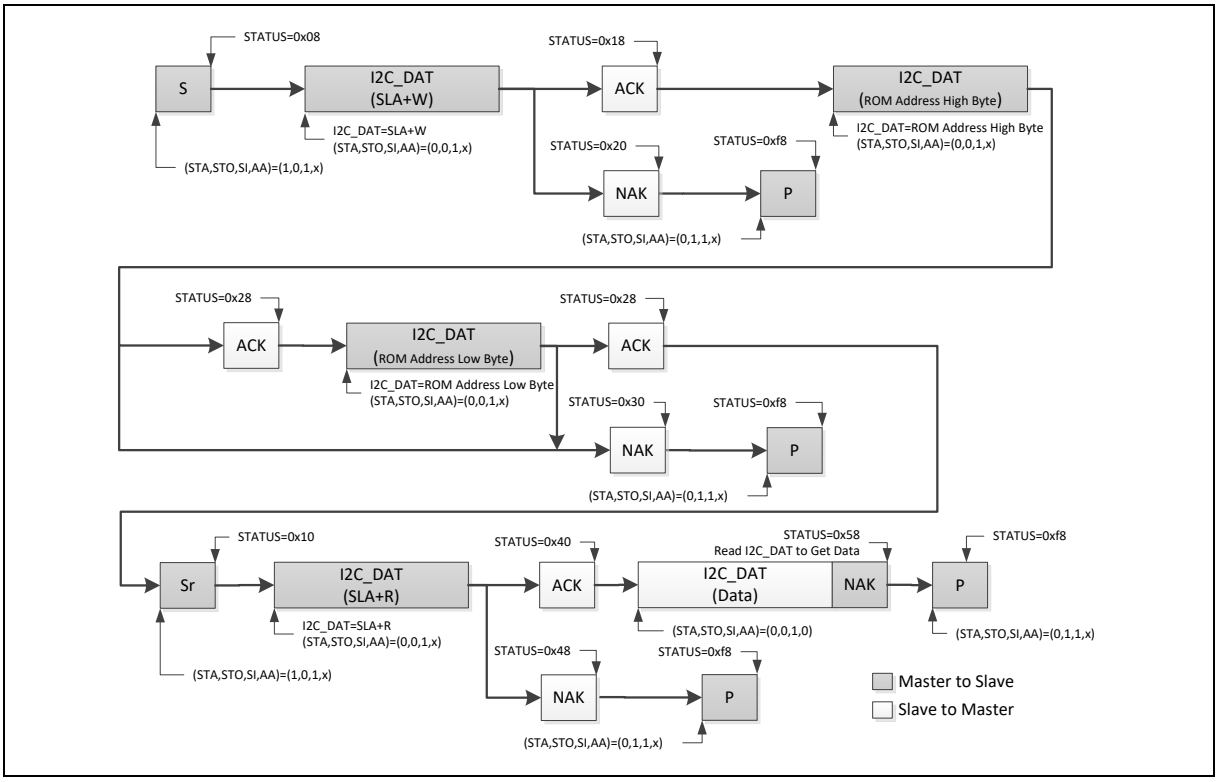


Figure 6.21-32 Protocol of EEPROM Random Read

The I²C controller, which is a master, sends START to bus. Then, it sends a SLA+W (Slave address + Write bit) to EERPOM followed by two bytes data address to set the EEPROM address to read. Finally, a Repeat START followed by SLA+R is sent to read the data from EEPROM.

6.21.6 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
I²C Base Address: **I2Cn_BA = 0x4008_0000 + (0x1000 *n)** **n= 0,1,2,3**				
I2Cn_CTL0	I2Cn_BA+0x00	R/W	I²C Control Register 0	0x0000_0000
I2Cn_ADDR0	I2Cn_BA+0x04	R/W	I²C Slave Address Register0	0x0000_0000
I2Cn_DAT	I2Cn_BA+0x08	R/W	I²C Data Register	0x0000_0000
I2Cn_STATUS0	I2Cn_BA+0x0C	R	I²C Status Register 0	0x0000_00F8
I2Cn_CLKDIV	I2Cn_BA+0x10	R/W	I²C Clock Divided Register	0x0000_0000
I2Cn_TOCTL	I2Cn_BA+0x14	R/W	I²C Time-out Control Register	0x0000_0000
I2Cn_ADDR1	I2Cn_BA+0x18	R/W	I²C Slave Address Register1	0x0000_0000
I2Cn_ADDR2	I2Cn_BA+0x1C	R/W	I²C Slave Address Register2	0x0000_0000
I2Cn_ADDR3	I2Cn_BA+0x20	R/W	I²C Slave Address Register3	0x0000_0000
I2Cn_ADDRMSK0	I2Cn_BA+0x24	R/W	I²C Slave Address Mask Register0	0x0000_0000
I2Cn_ADDRMSK1	I2Cn_BA+0x28	R/W	I²C Slave Address Mask Register1	0x0000_0000
I2Cn_ADDRMSK2	I2Cn_BA+0x2C	R/W	I²C Slave Address Mask Register2	0x0000_0000
I2Cn_ADDRMSK3	I2Cn_BA+0x30	R/W	I²C Slave Address Mask Register3	0x0000_0000
I2Cn_WKCTL	I2Cn_BA+0x3C	R/W	I²C Wake-up Control Register	0x0000_0000
I2Cn_WKSTS	I2Cn_BA+0x40	R/W	I²C Wake-up Status Register	0x0000_0000
I2Cn_CTL1	I2Cn_BA+0x44	R/W	I²C Control Register 1	0x0000_0000
I2Cn_STATUS1	I2Cn_BA+0x48	R/W	I²C Status Register 1	0x0000_0000
I2Cn_TMCTL	I2Cn_BA+0x4C	R/W	I²C Timing Configure Control Register	0x0002_0000
I2Cn_BUSCTL	I2Cn_BA+0x50	R/W	I²C Bus Management Control Register	0x0000_0000
I2Cn_BUSTCTL	I2Cn_BA+0x54	R/W	I²C Bus Management Timer Control Register	0x0000_0000
I2Cn_BUSSTS	I2Cn_BA+0x58	R/W	I²C Bus Management Status Register	0x0000_0000
I2Cn_PKTSIZE	I2Cn_BA+0x5C	R/W	I²C Packet Error Checking Byte Number Register	0x0000_0000
I2Cn_PKTCRC	I2Cn_BA+0x60	R	I²C Packet Error Checking Byte Value Register	0x0000_0000
I2Cn_BUSTOUT	I2Cn_BA+0x64	R/W	I²C Bus Management Timer Register	0x0000_0005
I2Cn_CLKTOUT	I2Cn_BA+0x68	R/W	I²C Bus Management Clock Low Timer Register	0x0000_0005

6.21.7 Register Description

I²C Control Register (I2C_CTL0)

Register	Offset	R/W	Description	Reset Value
I2Cn_CTL0	I2Cn_BA+0x00	R/W	I²C Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SARCIF	**DPCIF**	**SRCINTEN**	**DPCINTEN**	**Reserved**		**DPBITSEL**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTEN	**I2CEN**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**SARCIF**	**Slave Address Read Command Interrupt Flag** This bit is set by hardware when I²C receive address match read command. This bit is cleared by write 1 to it.
[14]	**DPCIF**	**Data Phase Count Interrupt Flag** This bit is set by hardware when I²C transfer bit count equal to DPBITSEL setting This bit is cleared by write 1 to it.
[13]	**SRCINTEN**	**Slave Read Command Interrupt Enable Bit** 0 = Slave Read Command Interrupt Disabled. 1 = Slave Read Command Interrupt Enabled.
[12]	**DPCINTEN**	**Data Phase Count Interrupt Enable Bit** 0 = Data Phase Count Interrupt Disabled. 1 = Data Phase Count Interrupt Enabled.
[11:10]	**Reserved**	Reserved.
[9:8]	**DPBITSEL**	**Data Phase Bit Count Select** 00 = DPCIF never set by hardware. 01 = When I²C is transfer data and bit count equal to 6, DPCIF will be set by hardware. 10 = When I²C is transfer data and bit count equal to 7, DPCIF will be set by hardware. 11 = When I²C is transfer data and bit count equal to 8, DPCIF will be set by hardware.
[7]	**INTEN**	**Enable Interrupt** 0 = I²C interrupt Disabled. 1 = I²C interrupt Enabled.
[6]	**I2CEN**	**I²C Controller Enable Bit** Set to enable I²C serial function controller. When I2CEN=1 the I²C serial function enable. The multi-function pin function must set to SDA, and SCL of I²C function first. 0 = I²C controller Disabled.

		1 = I²C controller Enabled.
[5]	**STA**	**I²C START Control** Setting STA to logic 1 to enter Master mode, the I²C hardware sends a START or repeat START condition to bus when the bus is free. This bit will be cleared by hardware automatically.
[4]	**STO**	**I²C STOP Control** In Master mode, setting STO to transmit a STOP condition to bus then I²C controller will check the bus condition if a STOP condition is detected. This bit will be cleared by hardware automatically.
[3]	**SI**	**I²C Interrupt Flag** When a new I²C state is present in the I2C_STATUS0 register, the SI flag is set by hardware. If bit INTEN (I2C_CTL0 [7]) is set, the I²C interrupt is requested. SI must be cleared by software. Clear SI by writing 1 to this bit. For ACKMEN is set in slave read mode, the SI flag is set in 8th clock period for user to confirm the acknowledge bit and 9th clock period for user to read the data in the data buffer.
[2]	**AA**	**Assert Acknowledge Control** When AA =1 prior to address or data is received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.). A slave is acknowledging the address sent from master, 2.) The receiver devices are acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line.
[1:0]	**Reserved**	Reserved.

I²C Data Register (I2C_DAT)

Register	Offset	R/W	Description	Reset Value
I2Cn_DAT	I2Cn_BA+0x08	R/W	I²C Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAT							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DAT**	**I²C Data** Bit [7:0] is located with the 8-bit transferred/received data of I²C serial port.

I²C Status Register (I2C_STATUS0)

Register	Offset	R/W	Description	Reset Value
I2Cn_STATUS0	I2Cn_BA+0x0C	R	I²C Status Register 0	0x0000_00F8

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STATUS							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**STATUS**	**I²C Status** The three least significant bits are always 0. The five most significant bits contain the status code. There are 28 possible status codes. When the content of I2C_STATUS0 is F8H, no serial interrupt is requested. Others I2C_STATUS0 values correspond to defined I²C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2C_STATUS0 one cycle after SI is set by hardware and is still present one cycle after SI has been reset by software. In addition, states 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the formation frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit.

I²C Clock Divided Register (I2C_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2Cn_CLKDIV	I2Cn_BA+0x10	R/W	I²C Clock Divided Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NFCNT				**Reserved**		**DIVIDER**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:12]	**NFCNT**	**Noise Filter Count** The register bits control the input filter width. If the pulse width is narrower than the setting((3+N)*PCLK), it will be ignored. 0 : filter width 3*PCLK 1 : filter width 4*PCLK *N* : filter width (3+*N*)*PCKL **Note:** Filter width Min :3*PCLK, Max : 18*PCLK
[11:10]	**Reserved**	Reserved.
[9:0]	**DIVIDER**	**I²C Clock Divided** Indicates the I²C clock rate: Data Baud Rate of I²C = (system clock) / (4x (DIVIDER+1)). **Note:** The minimum value of DIVIDER is 4.

I²C Time-out Control Register (I2C_TOCTL)

Register	Offset	R/W	Description	Reset Value
I2Cn_TOCTL	I2Cn_BA+0x14	R/W	I²C Time-out Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**TOCEN**	**TOCDIV4**	**TOIF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**TOCEN**	**Time-out Counter Enable Bit** When enabled, the 14-bit time-out counter will start counting when SI is cleared. Setting flag SI to '1' will reset counter and re-start up counting after SI is cleared. 0 = Time-out counter Disabled. 1 = Time-out counter Enabled.
[1]	**TOCDIV4**	**Time-out Counter Input Clock Divided by 4** When enabled, the time-out period is extended 4 times. 0 = Time-out period is extend 4 times Disabled. 1 = Time-out period is extend 4 times Enabled.
[0]	**TOIF**	**Time-out Flag** This bit is set by hardware when I²C time-out happened and it can interrupt CPU if I²C interrupt enable bit (INTEN) is set to 1. **Note:** Software can write 1 to clear this bit.

I²C Slave Address Register (ADDRx)

Register	Offset	R/W	Description	Reset Value
I2Cn_ADDR0	I2Cn_BA+0x04	R/W	I²C Slave Address Register0	0x0000_0000
I2Cn_ADDR1	I2Cn_BA+0x18	R/W	I²C Slave Address Register1	0x0000_0000
I2Cn_ADDR2	I2Cn_BA+0x1C	R/W	I²C Slave Address Register2	0x0000_0000
I2Cn_ADDR3	I2Cn_BA+0x20	R/W	I²C Slave Address Register3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**ADDR**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDR							**GC**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:1]	**ADDR**	**I²C Address** The content of this register is irrelevant when I²C is in Master mode. In the slave mode, the seven most significant bits must be loaded with the chip's own address. The I²C hardware will react if either of the address is matched. **Note:** When software set 10'h000, the address cannot be used.
[0]	**GC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled.

I²C Slave Address Mask Register (ADDRMSKx)

Register	Offset	R/W	Description	Reset Value
I2Cn_ADDRMSK0	I2Cn_BA+0x24	R/W	I²C Slave Address Mask Register0	0x0000_0000
I2Cn_ADDRMSK1	I2Cn_BA+0x28	R/W	I²C Slave Address Mask Register1	0x0000_0000
I2Cn_ADDRMSK2	I2Cn_BA+0x2C	R/W	I²C Slave Address Mask Register2	0x0000_0000
I2Cn_ADDRMSK3	I2Cn_BA+0x30	R/W	I²C Slave Address Mask Register3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					ADDRMSK		
7	6	5	4	3	2	1	0
ADDRMSK							Reserved

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:1]	**ADDRMSK**	**I²C Address Mask** 0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don't care.). I²C bus controllers support multiple address recognition with four address mask register. When the bit in the address mask register is set to one, it means the received corresponding address bit is don't-care. If the bit is set to zero, that means the received corresponding register bit should be exact the same as address register. **Note:** The wake-up function cannot use address mask in 10-bit mode.
[0]	**Reserved**	Reserved.

I²C Wake-up Control Register (I2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
I2Cn_WKCTL	I2Cn_BA+0x3C	R/W	I²C Wake-up Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NHDBUSEN	**Reserved**						**WKEN**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**NHDBUSEN**	**I²C No Hold BUS Enable Bit** 0 = I²C hold bus after wake-up. 1= I²C don't hold bus after wake-up. **Note:** The I²C controller could respond when WKIF event is not clear, it may cause error data transmitted or received. If data transmitted or received when WKIF event is not clear, user must reset I²C controller and execute the original operation again.
[6:1]	**Reserved**	Reserved.
[0]	**WKEN**	**I²C Wake-up Enable Bit** 0 = I²C wake-up function Disabled. 1= I²C wake-up function Enabled.

I²C Wake-up Status Register (I2C_WKSTS)

Register	Offset	R/W	Description	Reset Value
I2Cn_WKSTS	I2Cn_BA+0x40	R/W	I²C Wake-up Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**WRSTSWK**	**WKAKDONE**	**WKIF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**WRSTSWK**	**Read/Write Status Bit in Address Wakeup Frame** 0 = Write command be record on the address match wakeup frame. 1 = Read command be record on the address match wakeup frame. **Note:** This bit will be cleared when software can write 1 to WKAKDONE bit.
[1]	**WKAKDONE**	**Wakeup Address Frame Acknowledge Bit Done** 0 = The ACK bit cycle of address match frame isn't done. 1 = The ACK bit cycle of address match frame is done in power-down. **Note:** This bit can't release WKIF. Software can write 1 to clear this bit.
[0]	**WKIF**	**I²C Wake-up Flag** When chip is woken up from Power-down mode by I²C, this bit is set to 1. Software can write 1 to clear this bit.

I²C Control Register 1 (I2C_CTL1)

Register	Offset	R/W	Description	Reset Value
I2Cn_CTL1	I2Cn_BA+0x44	R/W	I²C Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**SWITCHEN**	**ADDR10EN**	**PDMASTR**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TWOBUFEN**	**UDRIEN**	**OVRIEN**	**PDMARST**	**RXPDMAEN**	**TXPDMAEN**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**SWITCHEN**	**SCL And SDA Pin Switch Enable Bit** 0 = I²C use original pin configuration. 1 = I²C switch SCL and SDA pin configuration. **Note: Original pin configuration table is shown in Basic Configuration chapter.**
[9]	**ADDR10EN**	**Address 10-bit Function Enable Bit** 0 = Address match 10-bit function Disabled. 1 = Address match 10-bit function Enabled.
[8]	**PDMASTR**	**PDMA Stretch Bit** 0 = I²C send STOP automatically after PDMA transfer done. (only master TX) 1 = I²C SCL bus is stretched by hardware after PDMA transfer done if the SI is not cleared. (only master TX)
[7:6]	**Reserved**	Reserved.
[5]	**TWOBUFEN**	**Two-level Buffer Enable Bit** 0 = Two-level buffer Disabled. 1 = Two-level buffer Enabled. Set to enable the two-level buffer for I²C transmitted or received buffer. It is used to improve the performance of the I²C bus.
[4]	**UDRIEN**	**I²C Under Run Interrupt Control Bit** Setting UDRIEN to logic 1 will send a interrupt to system when the TWOBUFEN bit is enabled and there is under run event happened in transmitted buffer.
[3]	**OVRIEN**	**I²C over Run Interrupt Control Bit** Setting OVRIEN to logic 1 will send a interrupt to system when the TWOBUFEN bit is enabled and there is over run event in received buffer.
[2]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the I²C request to PDMA.

[1]	**RXPDMAEN**	**PDMA Receive Channel Available** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[0]	**TXPDMAEN**	**PDMA Transmit Channel Available** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled.

I²C Status Register 1 (I2C_STATUS1)

Register	Offset	R/W	Description	Reset Value
I2Cn_STATUS1	I2Cn_BA+0x48	R/W	I²C Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							ONBUSY
7	6	5	4	3	2	1	0
UDR	OVR	EMPTY	FULL	ADMAT3	ADMAT2	ADMAT1	ADMAT0

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**ONBUSY**	**On Bus Busy (Read Only)** Indicates that a communication is in progress on the bus. It is set by hardware when a START condition is detected. It is cleared by hardware when a STOP condition is detected. 0 = The bus is IDLE (both SCL and SDA High). 1 = The bus is busy.
[7]	**UDR**	**I²C Under Run Status Bit** This bit indicates the transmitted two-level buffer TX or RX is under run when the TWOBUFEN = 1. **Note:** This bit is read only.
[6]	**OVR**	**I²C over Run Status Bit** This bit indicates the received two-level buffer TX or RX is over run when the TWOBUFEN = 1. **Note:** This bit is read only.
[5]	**EMPTY**	**TWO-LEVEL BUFFER EMPTY** This bit indicates two-level buffer TX or RX empty or not when the TWOBUFEN = 1. This bit is set when POINTER is equal to 0. **Note:** This bit is read only.
[4]	**FULL**	**TWO-LEVEL BUFFER FULL** This bit indicates two-level buffer TX or RX full or not when the TWOBUFEN = 1. This bit is set when POINTER is equal to 2 **Note:** This bit is read only.
[3]	**ADMAT3**	**I²C Address 3 Match Status** When address 3 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.

[2]	**ADMAT2**	**I²C Address 2 Match Status** When address 2 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[1]	**ADMAT1**	**I²C Address 1 Match Status** When address 1 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[0]	**ADMAT0**	**I²C Address 0 Match Status** When address 0 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.

I²C Timing Configure Control Register (I2C_TMCTL)

Register	Offset	R/W	Description	Reset Value
I2Cn_TMCTL	I2Cn_BA+0x4C	R/W	I²C Timing Configure Control Register	0x0002_0000

31	30	29	28	27	26	25	24
Reserved							**HTCTL**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HTCTL							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**STCTL**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STCTL							

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24:16]	**HTCTL**	**Hold Time Configure Control** This field is used to generate the delay timing between SCL falling edge and SDA rising edge in transmission mode. The delay hold time is numbers of peripheral clock = HTCTL x PCLK.
[15:9]	**Reserved**	Reserved.
[8:0]	**STCTL**	**Setup Time Configure Control** This field is used to generate a delay timing between SDA falling edge and SCL rising edge in transmission mode. The delay setup time is numbers of peripheral clock = STCTL x PCLK. **Note:** Setup time setting should not make SCL output less than three PCLKs.

I²C Bus Manage Control Register (I2C_BUSCTL)

Register	Offset	R/W	Description	Reset Value
I2Cn_BUSCTL	I2Cn_BA+0x50	R/W	I²C Bus Management Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		PECDIEN	BCDIEN	ACKM9SI	PECCLR	TIDLE	PECTXEN
7	6	5	4	3	2	1	0
BUSEN	SCTLOEN	SCTLOSTS	ALERTEN	BMHEN	BMDEN	PECEN	ACKMEN

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**PECDIEN**	**Packet Error Checking Byte Transfer Done Interrupt Enable Bit** 0 = PEC transfer done interrupt Disabled. 1 = PEC transfer done interrupt Enabled. **Note:** This bit is used in PECEN =1.
[12]	**BCDIEN**	**Packet Error Checking Byte Count Done Interrupt Enable Bit** 0 = Byte count done interrupt Disabled. 1 = Byte count done interrupt Enabled. **Note:** This bit is used in PECEN =1.
[11]	**ACKM9SI**	**Acknowledge Manual Enable Extra SI Interrupt** 0 = There is no SI interrupt in the 9th clock cycle when the BUSEN =1 and ACKMEN =1. 1 = There is SI interrupt in the 9th clock cycle when the BUSEN =1 and ACKMEN =1.
[10]	**PECCLR**	**PEC Clear at Repeat Start** The calculation of PEC starts when PECEN is set to 1 and it is cleared when the STA or STO bit is detected. This PECCLR bit is used to enable the condition of REPEAT START can clear the PEC calculation. 0 = PEC calculation is cleared by “Repeat Start” function Disabled. 1 = PEC calculation is cleared by “Repeat Start” function Enabled.
[9]	**TIDLE**	**Timer Check in Idle State** The BUSTOUT is used to calculate the time-out of clock low in bus active and the idle period in bus Idle. This bit is used to define which condition is enabled. 0 = BUSTOUT is used to calculate the clock low period in bus active. 1 = BUSTOUT is used to calculate the IDLE period in bus Idle. **Note:** The BUSY (I2C_BUSSTS[0]) indicate the current bus state.
[8]	**PECTXEN**	**Packet Error Checking Byte Transmission/Reception** 0 = No PEC transfer. 1 = PEC transmission is requested.

		Note: 1.This bit has no effect in slave mode when ACKMEN =0.
[7]	**BUSEN**	**BUS Enable Bit** 0 = The system management function Disabled. 1 = The system management function Enabled. **Note:** When the bit is enabled, the internal 14-bit counter is used to calculate the time out event of clock low condition.
[6]	**SCTLOEN**	**Suspend or Control Pin Output Enable Bit** 0 = The SUSCON pin in input. 1 = The output enable is active on the SUSCON pin.
[5]	**SCTLOSTS**	**Suspend/Control Data Output Status** 0 = The output of SUSCON pin is low. 1 = The output of SUSCON pin is high.
[4]	**ALERTEN**	**Bus Management Alert Enable Bit** Device Mode (BMHEN =0). 0 = Release the BM_ALERT pin high and Alert Response Header disabled: 0001100x followed by NACK if both of BMDEN and ACKMEN are enabled. 1 = Drive BM_ALERT pin low and Alert Response Address Header enables: 0001100x followed by ACK if both of BMDEN and ACKMEN are enabled. Host Mode (BMHEN =1). 0 = BM_ALERT pin not supported. 1 = BM_ALERT pin supported.
[3]	**BMHEN**	**Bus Management Host Enable Bit** 0 = Host function Disabled. 1 = Host function Enabled.
[2]	**BMDEN**	**Bus Management Device Default Address Enable Bit** 0 = Device default address Disable. When the address 0'b1100001x comes and either BMDEN or ACKMEN is disabled, the device responses NACK. 1 = Device default address Enabled. When the address 0'b1100001x comes and both BMDEN and ACKMEN are enabled, the device responses ACK.
[1]	**PECEN**	**Packet Error Checking Calculation Enable Bit** 0 = Packet Error Checking Calculation Disabled. 1 = Packet Error Checking Calculation Enabled. **Note:** When I^2C enter power-down mode, the bit should be enabled after wake-up if needed PEC calculation.
[0]	**ACKMEN**	**Acknowledge Control by Manual** In order to allow ACK control in slave reception including the command and data, slave byte control mode must be enabled by setting the ACKMEN bit. 0 = Slave byte control Disabled. 1 = Slave byte control Enabled. The 9th bit can response the ACK or NACK according the received data by user. When the byte is received, stretching the SCL signal low between the 8th and 9th SCL pulse. **Note:** If the BMDEN =1 and this bit is enabled, the information of I2C_STATUS0 will be fixed as 0xF0 in slave receive condition.

I²C Bus Management Timer Control Register (I2C_BUSTCTL)

Register	Offset	R/W	Description	Reset Value
I2Cn_BUSTCTL	I2Cn_BA+0x54	R/W	I²C Bus Management Timer Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			TORSTEN	CLKTOIEN	BUSTOIEN	CLKTOEN	BUSTOEN

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**TORSTEN**	**Time Out Reset Enable Bit** 0 = I²C state machine reset Disabled. 1 = I²C state machine reset Enabled. (The clock and data bus will be released to high)
[3]	**CLKTOIEN**	**Extended Clock Time Out Interrupt Enable Bit** 0 = Clock time out interrupt Disabled. 1 = Clock time out interrupt Enabled.
[2]	**BUSTOIEN**	**Time-out Interrupt Enable Bit** BUSY =1. 0 = SCL low time-out interrupt Disabled. 1 = SCL low time-out interrupt Enabled. BUSY =0. 0 = Bus IDLE time-out interrupt Disabled. 1 = Bus IDLE time-out interrupt Enabled.
[1]	**CLKTOEN**	**Cumulative Clock Low Time Out Enable Bit** 0 = Cumulative clock low time-out detection Disabled. 1 = Cumulative clock low time-out detection Enabled. For Master, it calculates the period from START to ACK For Slave, it calculates the period from START to STOP
[0]	**BUSTOEN**	**Bus Time Out Enable Bit** 0 = Bus clock low time-out detection Disabled. 1 = Bus clock low time-out detection Enabled (bus clock is low for more than BUSTO (I2C_BUSTOUT[7:0]) (in TIDLE=0) or high more than BUSTO (in TIDLE =1).

I²C Bus Management Status Register (I2C_BUSSTS)

Register	Offset	R/W	Description	Reset Value
I2Cn_BUSSTS	I2Cn_BA+0x58	R/W	I²C Bus Management Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PECDONE	CLKTO	BUSTO	SCTLDIN	ALERT	PECERR	BCDONE	BUSY

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**PECDONE**	**PEC Byte Transmission/Receive Done** 0 = PEC transmission/ receive is not finished when the PECEN is set. 1 = PEC transmission/ receive is finished when the PECEN is set. **Note:** Software can write 1 to clear this bit.
[6]	**CLKTO**	**Clock Low Cumulate Time-out Status** 0 = Cumulative clock low is no any time-out. 1 = Cumulative clock low time-out occurred. **Note:** Software can write 1 to clear this bit.
[5]	**BUSTO**	**Bus Time-out Status** 0 = There is no any time-out or external clock time-out. 1 = A time-out or external clock time-out occurred. In bus busy, the bit indicates the total clock low time-out event occurred; otherwise, it indicates the bus idle time-out event occurred. **Note:** Software can write 1 to clear this bit.
[4]	**SCTLDIN**	**Bus Suspend or Control Signal Input Status (Read Only)** 0 = The input status of SUSCON pin is 0. 1 = The input status of SUSCON pin is 1.
[3]	**ALERT**	**SMBus Alert Status** Device Mode (BMHEN =0). 0 = SMBALERT pin state is low. 1 = SMBALERT pin state is high. Host Mode (BMHEN =1). 0 = No SMBALERT event. 1 = There is SMBALERT event (falling edge) is detected in SMALERT pin when the BMHEN = 1 (SMBus host configuration) and the ALERTEN = 1. **Note 1:** The SMBALERT pin is an open-drain pin, and the pull-high resistor is must in the system.

		Note 2: Software can write 1 to clear this bit.
[2]	**PECERR**	**PEC Error in Reception** 0 = PEC value equal the received PEC data packet. 1 = PEC value doesn't match the receive PEC data packet. **Note:** Software can write 1 to clear this bit.
[1]	**BCDONE**	**Byte Count Transmission/Receive Done** 0 = Byte count transmission/ receive is not finished when the PECEN is set. 1 = Byte count transmission/ receive is finished when the PECEN is set. **Note:** Software can write 1 to clear this bit.
[0]	**BUSY**	**Bus Busy (Read Only)** Indicates that a communication is in progress on the bus. It is set by hardware when a START condition is detected. It is cleared by hardware when a STOP condition is detected 0 = Bus is IDLE (both SCL and SDA High). 1 = Bus is busy.

I²C Byte Number Register (I2C_PKTSIZE)

Register	Offset	R/W	Description	Reset Value
I2Cn_PKTSIZE	I2Cn_BA+0x5C	R/W	I²C Packet Error Checking Byte Number Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**PLDSIZE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PLDSIZE							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**PLDSIZE**	**Transfer Byte Number** The transmission or receive byte number in one transaction when the PECEN is set. The maximum transaction or receive byte is 256 Bytes. **Note:** The byte number counting includes address, command code, and data frame.

I²C PEC Value Register (I2C_PKTCRC)

Register	Offset	R/W	Description	Reset Value
I2Cn_PKTCRC	I2Cn_BA+0x60	R	I²C Packet Error Checking Byte Value Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PECCRC							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**PECCRC**	**Packet Error Checking Byte Value** This byte indicates the packet error checking content after transmission or receive byte count by using the C(x) = X8 + X2 + X + 1. It is read only.

I²C Bus Management Timer Register (I2C_BUSTOUT)

Register	Offset	R/W	Description	Reset Value
I2Cn_BUSTOUT	I2Cn_BA+0x64	R/W	I²C Bus Management Timer Register	0x0000_0005

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BUSTO							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**BUSTO**	**Bus Management Time-out Value** Indicates the bus time-out value in bus is IDLE or SCL low. **Note:** If the user wants to revise the value of BUSTOUT, the TORSTEN (I2C_BUSTCTL[4]) bit shall be set to 1 and cleared to 0 first when the BUSEN(I2C_BUSCTL[7]) is set.

I²C Clock Low Timer Register (I2C_CLKTOUT)

Register	Offset	R/W	Description	Reset Value
I2Cn_CLKTOUT	I2Cn_BA+0x68	R/W	I²C Bus Management Clock Low Timer Register	0x0000_0005

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKTO							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**CLKTO**	**Bus Clock Low Timer** The field is used to configure the cumulative clock extension time-out. **Note:** If the user wants to revise the value of CLKLTOUT, the TORSTEN bit shall be set to 1 and cleared to 0 first when the BUSEN is set.

6.22 Low Power I²C Serial Interface Controller (LPI2C)

6.22.1 Overview

LPI2C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices.

There is one set of LPI2C controller which supports Power-down Wake-up function.

6.22.2 Features

The LPI2C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main features of LPI2C bus include:

- Supports one LPI2C port
- Master/Slave mode
- Bidirectional data transfer between masters and slaves
- Supports Standard mode (100 kbps), Fast mode (400 kbps) and Fast mode plus (1 Mbps) when clock source is greater than 20 MHz
- Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
- Serial clock synchronization allows devices with different bit rates to communicate via one serial bus
- Serial clock synchronization used as a handshake mechanism to suspend and resume serial transfer
- Built-in 14-bit time-out counter requesting the LPI2C interrupt if the I²C bus hangs up and timer-out counter overflow
- Programmable clocks allow for versatile rate control
- Supports 7-bit addressing
- Supports multiple address recognition (four slave address with mask option)
- Supports Power-down wake-up function
- Supports LPPDMA with one buffer capability
- Supports setup/hold time programmable
- Add pin swap function
- Supports auto-operation mode
- Supports trigger source from LPTMR0/1, TTMR0/1 and WKIOA/B/C/D rising/falling edge event

6.22.3 Block Diagram

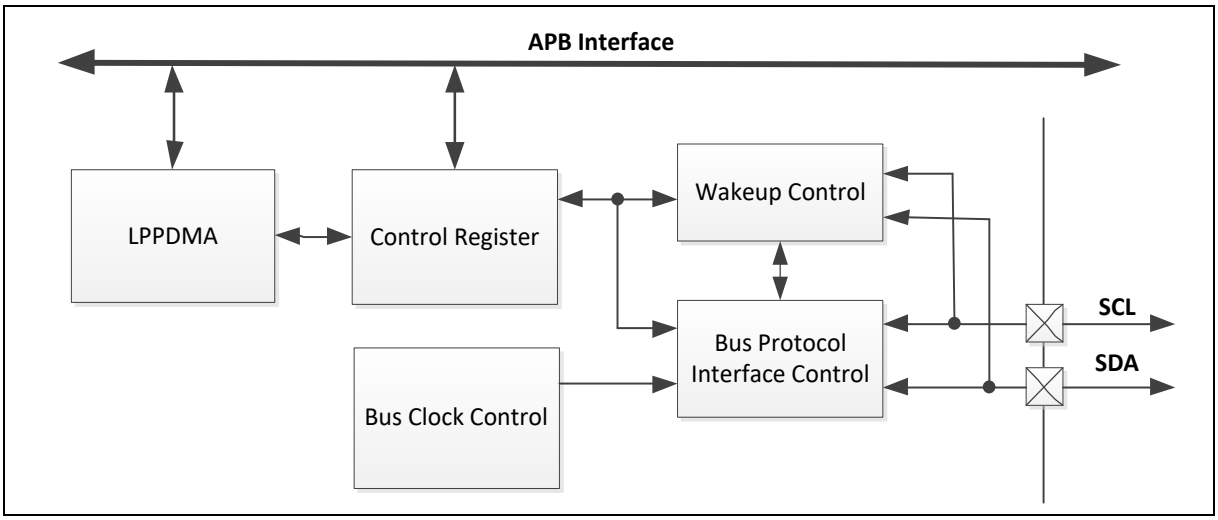


Figure 6.22-1 LPI2C Controller Block Diagram

6.22.4 Basic Configuration

6.22.4.1 LPI2C0 Basic Configuration

- Clock Source Configuration
  - Enable LPI2C0 peripheral clock in LPI2C0CKEN (LPSCC_CLKEN0[18]).
- Reset Configuration
  - Reset LPI2C0 controller in LPI2C0RST (LPSCC_IPRST0[18]).
- SCL/SDA Pin Switch
  - Set SWITCHEN (LPI2C0_CTL1[10] = 1) to switch LPI2C0 SCL/SDA pin configuration.

6.22.5 Functional Description

On I²C bus, data is transferred between a Master and a Slave. Data bits transfer on the SCL and SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit long. There is one SCL clock pulse for each data bit with the MSB being transmitted first, and an acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (START or STOP). Please refer to Figure 6.21-2 for more detailed I²C Bus Timing.

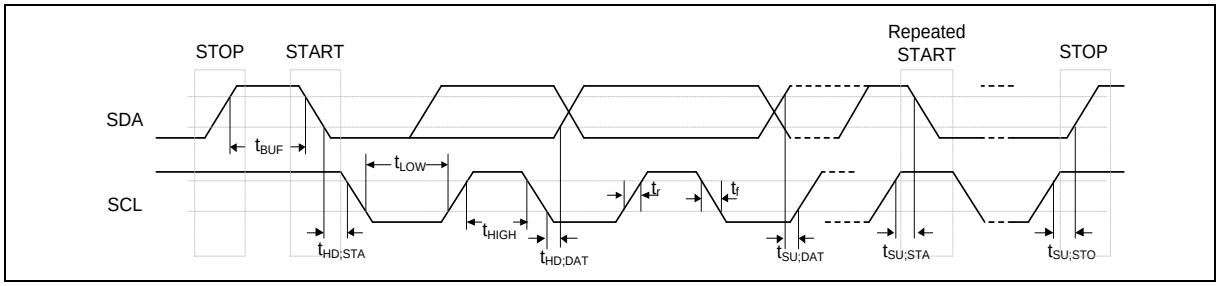


Figure 6.22-2 I²C Bus Timing

The device's on-chip LPI2C provides the serial interface that meets the I²C bus standard mode specification. The LPI2C port handles byte transfers autonomously. To enable this port, the bit LPI2CEN(LPI2C_CTL0[6]) should be set to '1'. The LPI2C hardware interfaces to the I²C bus via two pins: SDA and SCL. When I/O pins are used as LPI2C ports, user must set the pins function to LPI2C in advance.

Note: Pull-up resistor is needed for LPI2C operation as the SDA and SCL are open-drain pins.

6.22.5.1 I²C Protocol

Figure 6.21-3 shows the typical I²C protocol. Normally, a standard communication consists of four parts:

- START or Repeated START signal generation
- Slave address and R/W bit transfer
- Data transfer
- STOP signal generation

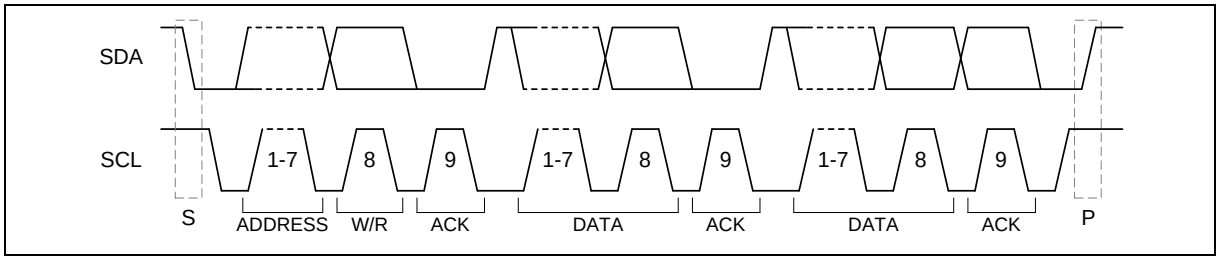


Figure 6.22-3 I²C Protocol

- START or Repeated START signal

  When the bus is free/idle, which means no master device is engaging the bus (both SCL and SDA lines are high), a master can initiate a transfer by sending a START signal. A START signal, usually referred to as the "S" bit, is defined as a HIGH to LOW transition on the SDA line while SCL is HIGH. The START signal denotes the beginning of a new data transmission.

  After having sent the address byte (address and read/write bit), the master may send any number of bytes followed by a stop condition. Instead of sending the stop condition it is

also allowed to send another start condition again followed by an address (and of course including a read/write bit) and more data. The start condition is called as Repeat START (Sr). This is defined recursively allowing any number of start conditions to be sent. The purpose of this is to allow combined write/read operations to one or more devices without releasing the bus and thus with the guarantee that the operation is not interrupted. The controller uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

- STOP signal

  The master can terminate the communication by generating a STOP signal. A STOP signal, usually referred to as the “P” bit, is defined as a LOW to HIGH transition on the SDA line while SCL is HIGH.

  Figure 6.21-4 shows the waveform of START, Repeat START and STOP.

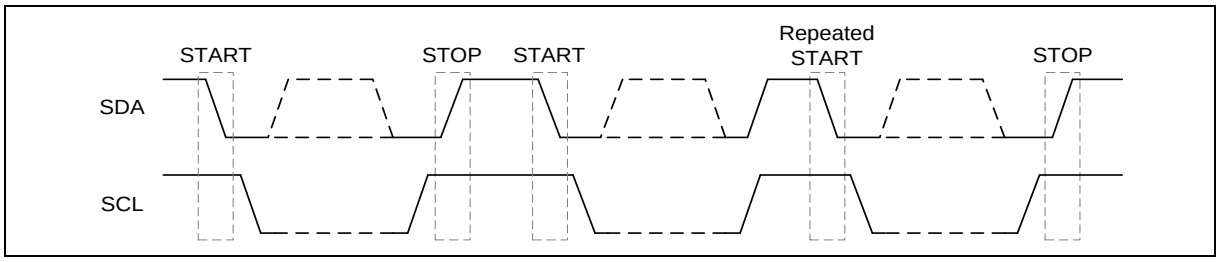


Figure 6.22-4 START and STOP Conditions

- Slave Address Transfer

  After a (Repeated) START condition, the master sends a slave address to identify the target device of the communication. The start address can comprise one or two address bytes (for 7-bit or for 10-bit addressing schemes). After an address byte, a slave sensitive to the transmitted address has to acknowledge the reception.

  Therefore, the slave’s address can be programmed in the device, where it is compared to the received address. In case of a match, the slave answers with an acknowledge (SDA = 0). Slaves that are not targeted answer with a non-acknowledge (SDA = 1). In addition to the match of the programmed address, another address byte value has to be answered with an acknowledge if the slave is capable to handle the corresponding requests.

- Data Transfer

  When a slave receives a correct address with an R/W bit, the data will follow R/W bit specified to transfer. Each transferred byte is followed by an acknowledge bit on the 9th SCL clock cycle. If the slave signals a Not Acknowledge (NACK), the master can generate a STOP signal to abort the data transfer or generate a Repeated START signal and start a new transfer cycle.

  If the master, as a receiving device, does Not Acknowledge (NACK) the slave, the slave releases the SDA line for the master to generate a STOP or Repeated START signal. The Figure 6.21-5 and Figure 6.21-6 shows the waveform of bit transfer and acknowledge.

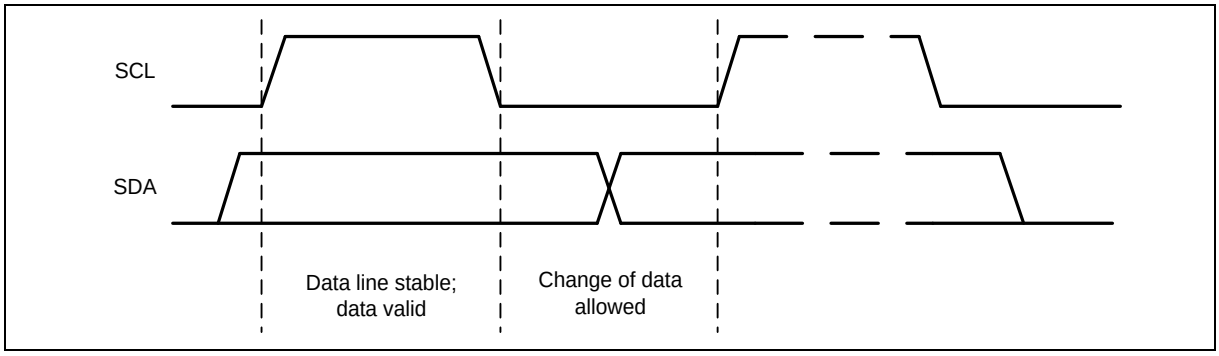


Figure 6.22-5 Bit Transfer on the I²C Bus

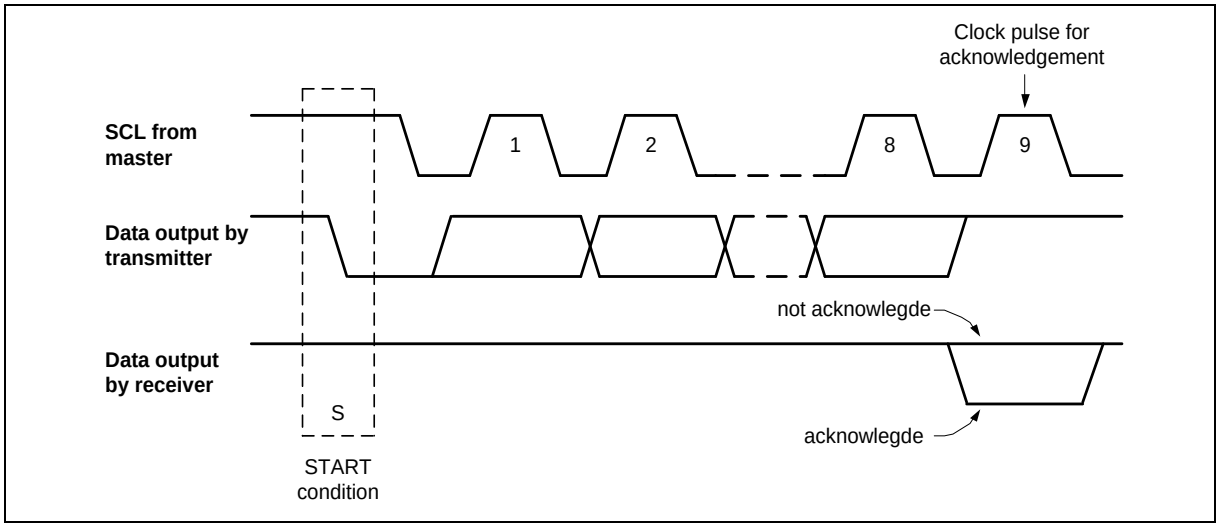


Figure 6.22-6 Acknowledge on the I²C Bus

- Data transfer on I²C bus

  Figure 6.21-7 shows a master transmits data to slave by 7-bit. A master addresses a slave with a 7-bit address and 1-bit write index to denote that the master wants to transmit data to the slave. The master keeps transmitting data after the slave returns acknowledge to the master.

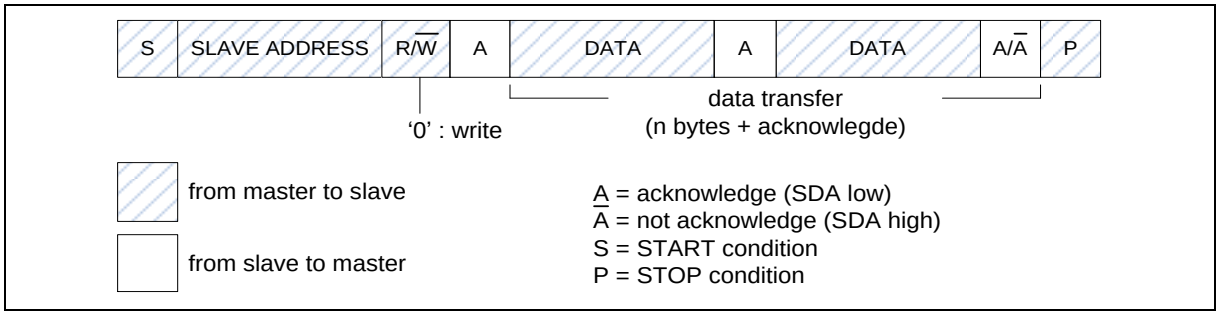


Figure 6.22-7 Master Transmits Data to Slave by 7-bit

Figure 6.21-8 shows a master read data from slave by 7-bit. A master addresses a slave with a 7-bit address and 1-bit read index to denote that the master wants to read data from the slave. The slave will start transmitting data after the slave returns acknowledge to the master.

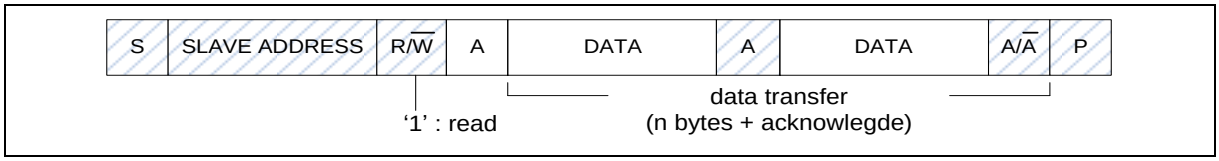


Figure 6.22-8 Master Reads Data from Slave by 7-bit

6.22.5.2 Operation Modes

The on-chip LPI2C ports support three operation modes, Master, Slave, and General Call Mode.

In a given application, LPI2C port may operate as a master or as a slave. In Slave mode, the LPI2C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master(by setting the AA bit), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, hardware waits until the bus is free before entering Master mode so that a possible slave action is not be interrupted. If bus arbitration is lost in Master mode, LPI2C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer.

To control the I^2C bus transfer in each mode, user needs to set LPI2C_CTL0, LPI2C_DAT registers according to current status code of LPI2C_STATUS0 register. In other words, for each I^2C bus action, user needs to check current status by LPI2C_STATUS0 register, and then set LPI2C_CTL0, LPI2C_DAT registers to take bus action. Finally, check the response status by LPI2C_STATUS0.

The bits, STA, STO and AA in LPI2C_CTL0 register are used to control the next state of the I^2C hardware after SI flag of LPI2C_CTL0[3] register is cleared. Upon completion of the new action, a new status code will be updated in LPI2C_STATUS0 register and the SI flag of LPI2C_CTL0 register will be set. But the SI flag will not be set when LPI2C STOP. If the LPI2C interrupt control bit INTEN (LPI2C_CTL0 [7]) is set, appropriate action or software branch of the new status code can be performed in the Interrupt service routine.

Figure 6.21-11 shows the current LPI2C status code is 0x08, and then set LPI2C_DAT=SLA+W and (STA,STO,SI,AA) = (0,0,1,x) to send the address to I^2C bus. If a slave on the bus matches the address and response ACK, the LPI2C_STATUS0 will be updated by status code 0x18.

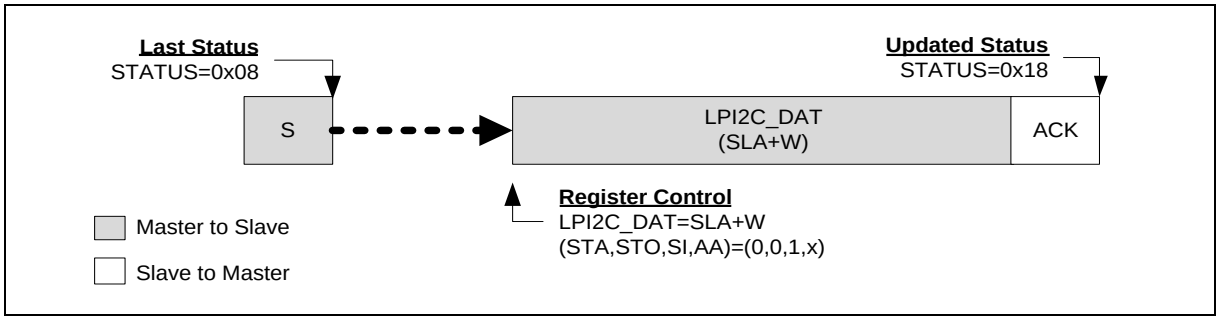


Figure 6.22-9 Control I^2C Bus according to the Current LPI2C Status

Master Mode

In Figure 6.21-12 and Figure 6.21-13, all possible protocols for LPI2C master are shown. User needs to follow proper path of the flow to implement required LPI2C protocol.

In other words, user can send a START signal to bus and LPI2C will be in Master Transmitter (MT) mode (Figure 6.21-12) or Master receiver (MR) mode (Figure 6.21-13) after START signal has been sent successfully and new status code would be 0x08. Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform LPI2C protocol.

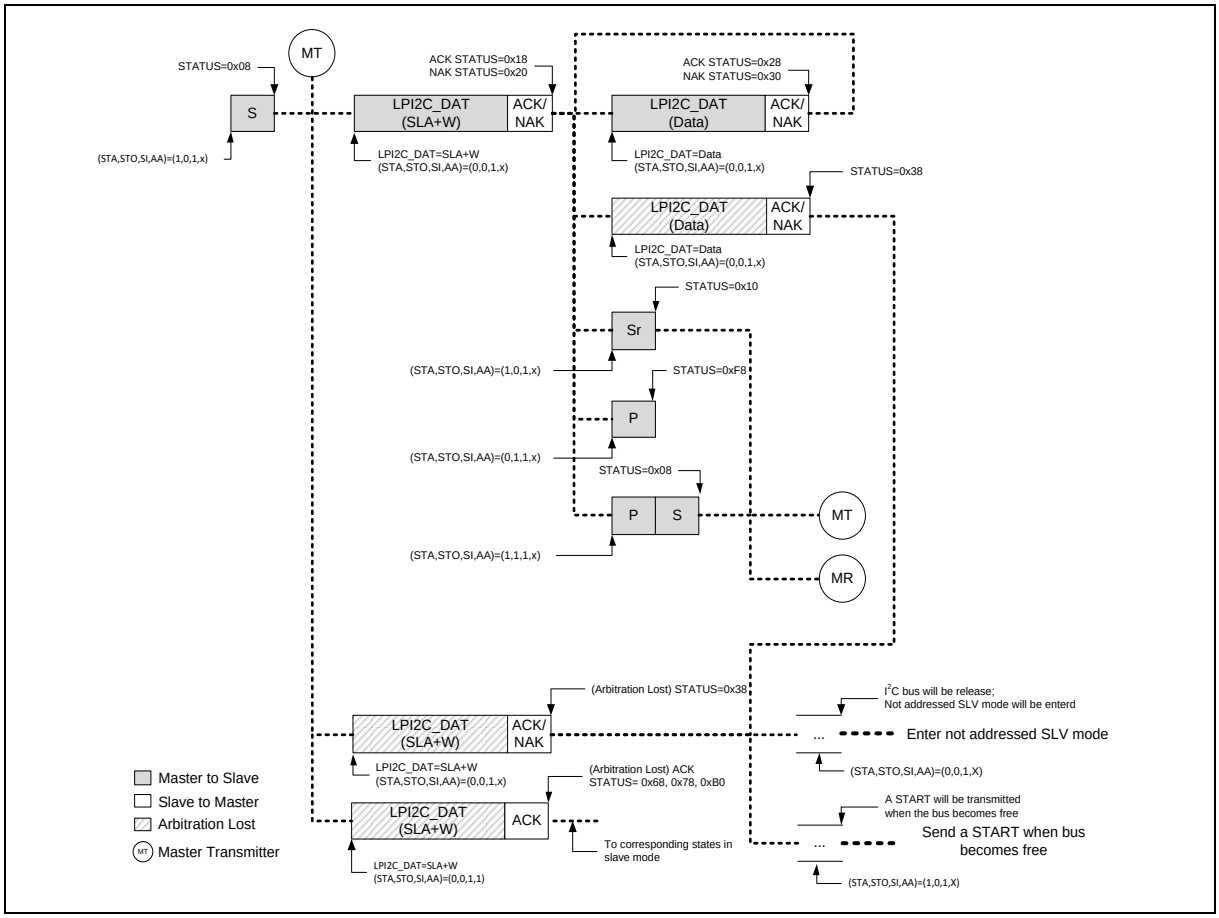


Figure 6.22-10 Master Transmitter Mode Control Flow

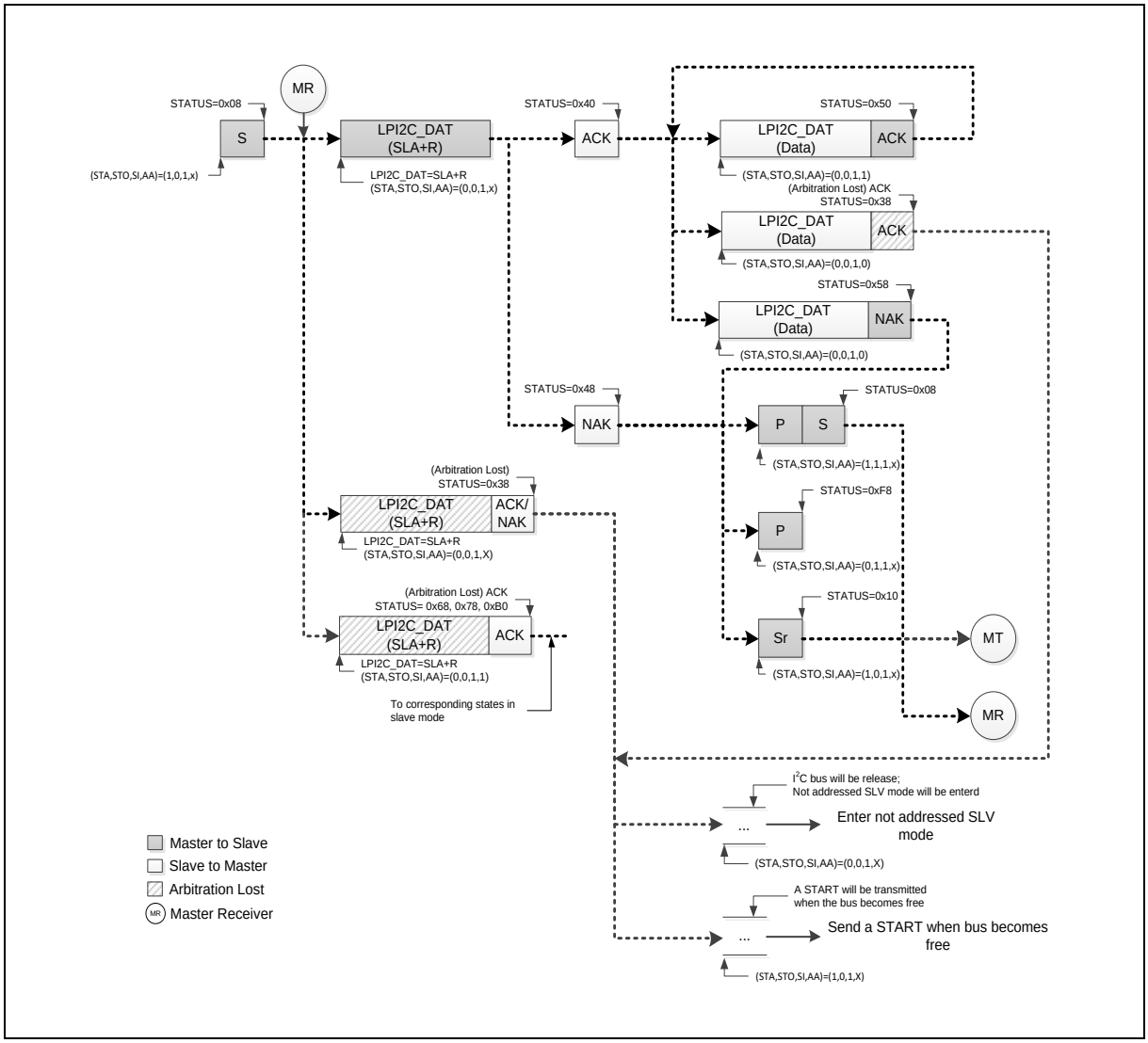


Figure 6.22-11 Master Receiver Mode Control Flow

If LPI2C is in Master mode and gets arbitration lost, the status code will be 0x38. In status 0x38, user may set (STA, STO, SI, AA) = (1, 0, 1, X) to send START to re-start Master operation when bus become free. Otherwise, user may set (STA, STO, SI, AA) = (0, 0, 1, X) to release LPI2C bus and enter not addressed Slave mode.

Slave Mode

When reset default, LPI2C is not addressed and will not recognize the address on I^{2}C bus. User can set slave address by LPI2C_ADDRn (n=0~3) and set (STA, STO, SI, AA) = (0, 0, 1, 1) to let LPI2C recognize the address sent by master. Figure 6.21-14 shows all the possible flow for LPI2C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.21-14 to implement their own LPI2C protocol.

If bus arbitration is lost in Master mode, LPI2C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) after arbitration lost, the status code is 0x68. If the detected address is SLA+R (Master want to read data from Slave) after arbitration lost, the status code is 0xB0.

Note: During LPI2C communication, the SCL clock will be released when writing '1' to clear SI flag in Slave mode.

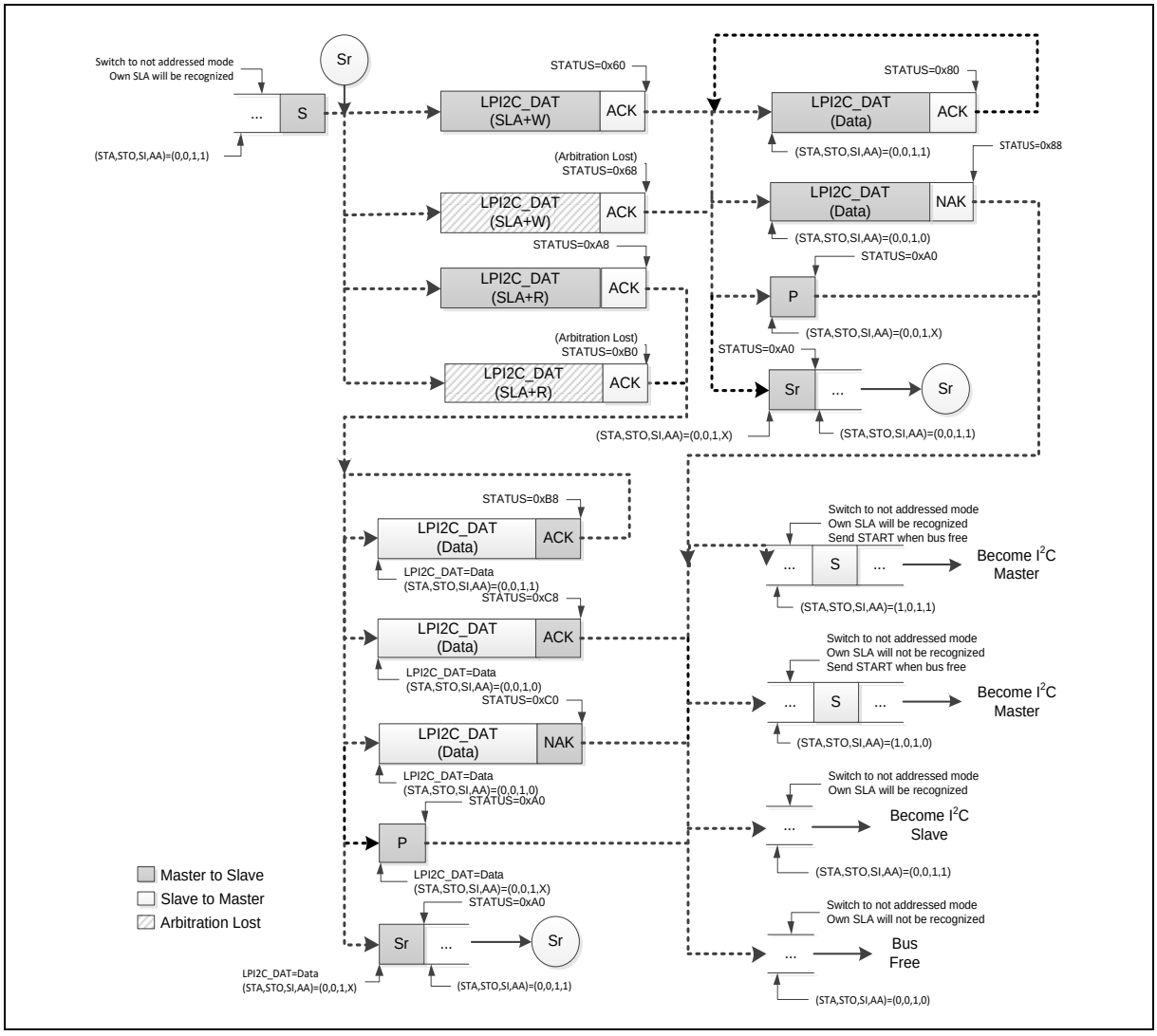


Figure 6.22-12 Slave Mode Control Flow

If LPI2C is still receiving data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x88 as shown in the above figure when getting 0xA0 status.

If LPI2C is still transmitting data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0xC8 as shown in the above figure when getting 0xA0 status.

Note: After slave gets status of 0x88, 0xC8, 0xC0 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any LPI2C signal or address from master. At this status, LPI2C should enter idle mode.

General Call (GC) Mode

If the GC bit (LPI2C_ADDRn [0]) is set, the LPI2C port hardware will respond to General Call address (00H). User can clear GC bit to disable general call function. When the GC bit is set and LPI2C in Slave mode, it can receive the general call address by 0x00 after master send general call address to LPI2C bus, then it will follow status of GC mode.

The GC mode can wake up when address matched. Note that the default address is 0x00, but user must set an address except for 0x00.

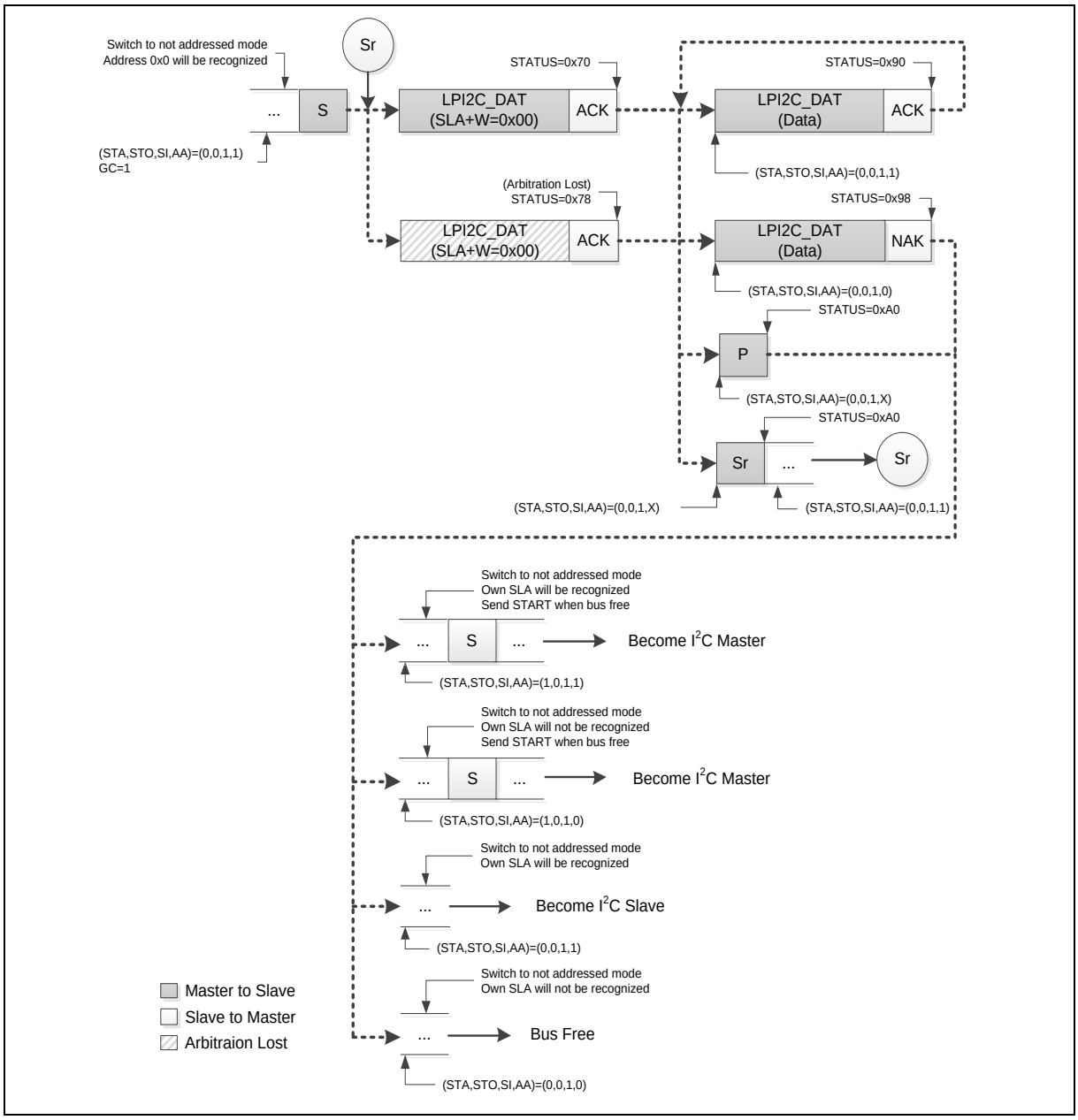


Figure 6.22-13 GC Mode

If LPI2C is still receiving data in GC mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x98 in above figure when getting 0xA0 status.

Note: After slave gets status of 0x98 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any LPI2C signal or address from master. At this time, LPI2C should enter idle mode.

6.22.5.3 LPPDMA Transfer Function

LPI2C supports LPPDMA transfer function. When TXPDMAEN (LPI2C_CTL1 [0]) is set to 1, LPI2C will issue request to LPPDMA controller to start the LPPDMA transmission process automatically.

When RXPDMAEN (LPI2C_CTL1 [1]) is set to 1, LPI2C will start the receive LPPDMA process. LPI2C will issue the request to LPPDMA controller automatically when there is data written into the received BUFFER.

When LPI2C enters LPPDMA mode, the mostly status interrupt will be masked. Let the interrupt not occur besides the bus error or NACK or STOP interrupt (0x20, 0x30, 0x38, 0x48, 0x58, 0x00, 0xA0, 0xC0, 0x88 and 0x98).

Set the PDMASTR (LPI2C_CTL1 [8]) only LPI2C is in master TX mode. If PDMASTR is cleared to 0, LPI2C will send STOP automatically after LPPDMA transfer done and buffer empty. If PDMASTR is set to 1, SI will be set to 1 and LPI2C bus will be stretched by hardware after LPPDMA transfer done and buffer empty.

6.22.5.4 Programmable setup and hold times

To guarantee a correct data setup and hold time, the timing must be configured. By programming HTCTL (LPI2C_TMCTL[24:16]) to configure hold time and STCTL (LPI2C_TMCTL[8:0]) to configure setup time.

The delay timing refer peripheral clock (PCLK). When device stretch master clock, the setup and hold time configuration value will not affected by stretched.

User should focus the limitation of setup and hold time configuration, the timing setting must follow LPI2C protocol. Once setup time configuration greater than design limitation, that means if setup time setting make SCL output less than three PCLKs, LPI2C cannot work normally due to SCL must sample three times. And once hold time configuration greater than LPI2C clock limitation, LPI2C will occur bus error. It is suggested that user calculate suitable timing with baud rate and protocol before setting timing. Table 6.21-2 shows the relationship between LPI2C baud rate and PCLK, the number of table represent one clock duty contain how many PCLKs. Setup and hold time configuration even can program some extreme values in the design, but user should follow LPI2C protocol standard.

LPI2C Baud Rate PCLK	100k	200k	400k	800k	1200k
12 MHz	120	60	30	15	10
24 MHz	240	120	60	30	20
48 MHz	480	240	120	60	40
72 MHz	720	360	180	90	60

Table 6.22-1 Relationship between LPI2C Baud Rate and PCLK

For setup time wrong adjustment example, assuming one SCL cycle contains 5 PCLKs and set STCTL (LPI2C_TMCTL[8:0]) to 3 that stretch three PCLKs for setup time setting. The setup time maximum setting value: ST_{limit} = (LPI2C_CLKDIV[9:0]+1) X 2 - 6.

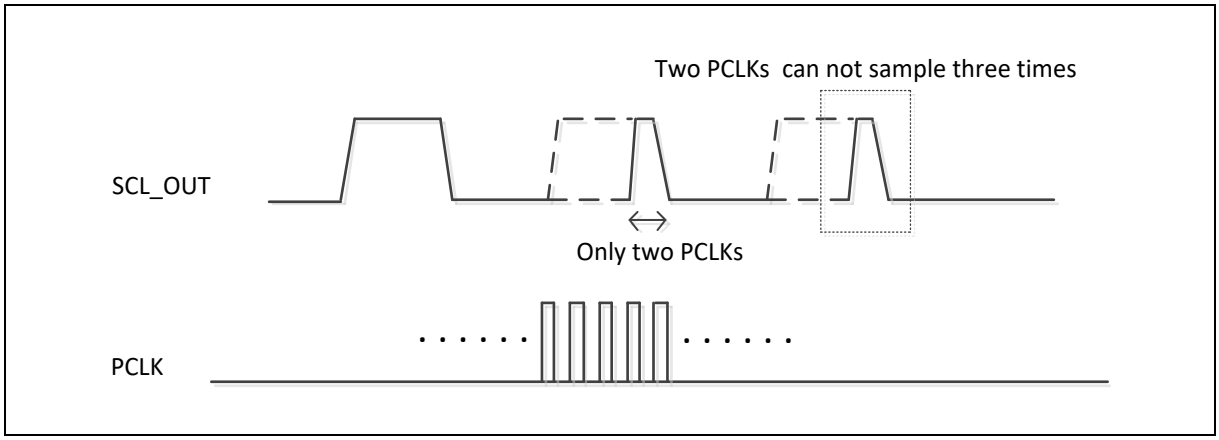


Figure 6.22-14 Setup Time Wrong Adjustment

For hold time wrong adjustment example, use LPI2C Baud Rate = 1200k and PCLK = 72 MHz, the SCL high/low duty = 60 PCLK. When HTCTL (LPI2C_TMCTL[24:16]) is set to 61 and STCTL (LPI2C_TMCTL[8:0]) is set to 0, then SDA output delay will over SCL high duty and cause bus error. The hold time maximum setting value: HT_{limit} = (LPI2C_CLKDIV[9:0]+1) X 2 - 9.

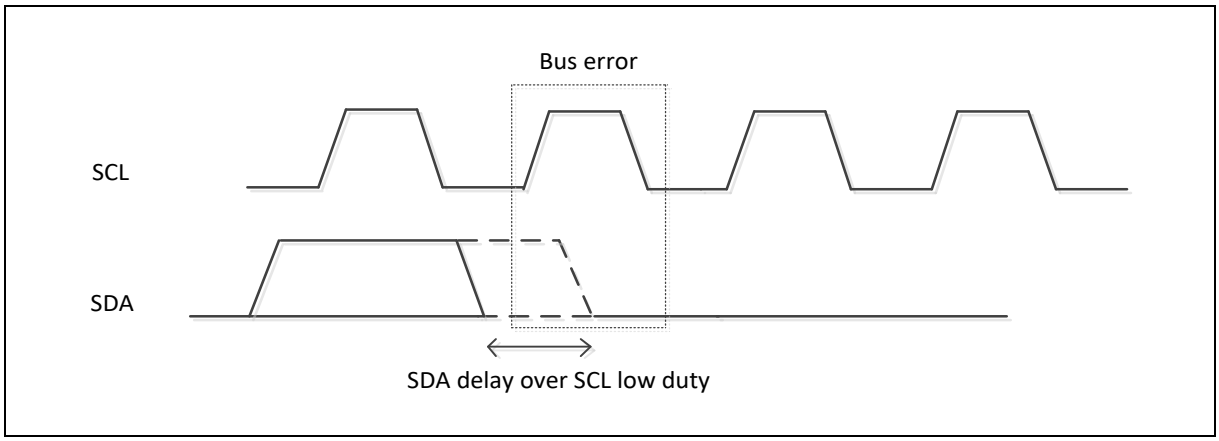


Figure 6.22-15 Hold Time Wrong Adjustment

6.22.5.5 LPI2C Protocol Registers

To control LPI2C port through the following fifteen special function registers: LPI2C_CTL0 (control register), LPI2C_STATUS0 (status register), LPI2C_DAT (data register), LPI2C_ADDRn (address registers, n=0~3), LPI2C_ADDRMSKn (address mask registers, n=0~3), LPI2C_CLKDIV (clock rate register), LPI2C_TOCTL (Time-out control register), LPI2C_WKCTL(wake up control register) and LPI2C_WKSTS(wake up status register).

Address Registers (LPI2C_ADDR)

The LPI2C port is equipped with four slave address registers, LPI2C_ADDRn (n=0~3). The contents of the register are irrelevant when LPI2C is in Master mode. In Slave mode, the bit field ADDR(LPI2C_ADDRn[7:1]) must be loaded with the chip's own slave address. The LPI2C hardware will react if the contents of LPI2C_ADDRn are matched with the received slave address.

The LPI2C ports support the "General Call" function. If the GC bit (LPI2C_ADDRn [0]) is set the LPI2C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.

When the GC bit is set and LPI2C is in Slave mode, it can receive the general call address by 00H after Master send general call address to LPI2C bus, then it will follow status of GC mode.

Slave Address Mask Registers (LPI2C_ADDRMSK)

The LPI2C bus controller supports multiple address recognition with four address mask registers LPI2C_ADDRMSKn (n=0~3). When the bit in the address mask register is set to 1, it means the received corresponding address bit is "Don't care". If the bit is set to 0, it means the received corresponding register bit should be exactly the same as address register.

Data Register (LPI2C_DAT)

This register contains a byte of serial data to be transmitted or a byte which just has been received. The CPU can be read from or written to the 8-bit (LPI2C_DAT [7:0]) directly while it is not in the process of shifting a byte. When LPI2C is in a defined state and the serial interrupt flag (SI) is set, data in LPI2C_DAT [7:0] remains stable. While data is being shifted out, data on the bus is simultaneously being shifted in; LPI2C_DAT [7:0] always contains the last data byte presented on the bus.

The acknowledge bit is controlled by the LPI2C hardware and cannot be accessed by the CPU. Serial data is shifted into LPI2C_DAT [7:0] on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into LPI2C_DAT [7:0], the serial data is available in LPI2C_DAT [7:0], and the acknowledge bit (ACK or NACK) is returned by the control logic during the ninth clock pulse. In order to monitor bus status while sending data, the bus data will be shifted to LPI2C_DAT[7:0] when sending LPI2C_DAT[7:0] to bus. In the case of sending data, serial data bits are shifted out from LPI2C_DAT [7:0] on the falling edge of SCL clocks, and is shifted to LPI2C_DAT [7:0] on the rising edge of SCL clocks. Figure 6.21-28 shows LPI2C Data Shifting Direction.

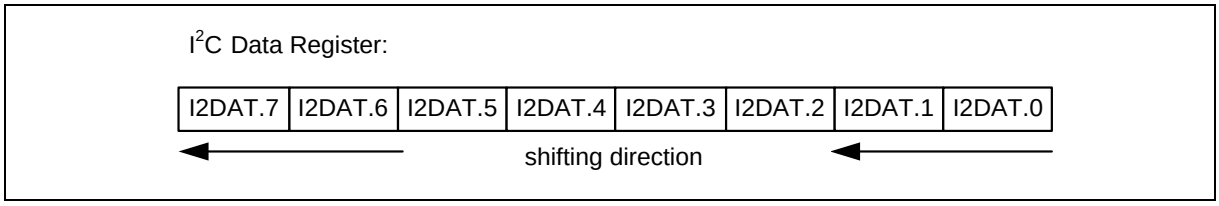


Figure 6.22-16 LPI2C Data Shifting Direction

Control Register (LPI2C_CTL0)

The CPU can be read from and written to LPI2C_CTL0 [7:0] directly. When the LPI2C port is enabled by setting LPI2CEN (LPI2C_CTL0 [6]) to high, the internal states will be controlled by LPI2C_CTL0 and LPI2C logic hardware.

There are two bits are affected by hardware: the SI bit is set when the LPI2C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when LPI2CEN = 0.

Once a new status code is generated and stored in LPI2C_STATUS0, the LPI2C Interrupt Flag bit SI (LPI2C_CTL0 [3]) will be set automatically. If the Enable Interrupt bit INTEN (LPI2C_CTL0 [7]) is set at this time, the LPI2C interrupt will be generated. The bit field LPI2C_STATUS0[7:0] stores the internal state code, the content keeps stable until SI is cleared by software.

Status Register (LPI2C_STATUS0)

LPI2C_STATUS0 [7:0] is an 8-bit read-only register. The bit field LPI2C_STATUS0 [7:0] contains the status code and there are 26 possible status codes. All states are listed in Table 6.21-3. When LPI2C_STATUS0 [7:0] is F8H, no serial interrupt is requested. All other LPI2C_STATUS0 [7:0] values correspond to the defined LPI2C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in LPI2C_STATUS0[7:0] one cycle PCLK after SI set by hardware and is still present one cycle PCLK after SI reset by software.

In addition, the state 00H stands for a Bus Error, which occurs when a START or STOP condition is present at an incorrect position in the LPI2C format frame. A Bus Error may occur during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover LPI2C from bus error, STO should be set and SI should be cleared to enter Not Addressed Slave mode. Then STO is cleared to release bus and to wait for a new communication. The LPI2C bus cannot recognize stop condition during

this action when a bus error occurs.

Master Mode		Slave Mode	
STATUS	Description	STATUS	Description
0x08[1]	Start	0xA0	Slave Transmit Repeat Start or Stop
0x10[1]	Master Repeat Start	0xA8[1]	Slave Transmit Address ACK
0x18[1]	Master Transmit Address ACK	0xB8[1]	Slave Transmit Data ACK
0x20	Master Transmit Address NACK	0xC0	Slave Transmit Data NACK
0x28[1]	Master Transmit Data ACK	0xC8[1]	Slave Transmit Last Data ACK
0x30	Master Transmit Data NACK	0x60[1]	Slave Receive Address ACK
0x38	Master Arbitration Lost	0x68[1]	Slave Receive Arbitration Lost
0x40[1]	Master Receive Address ACK	0x80[1]	Slave Receive Data ACK
0x48	Master Receive Address NACK	0x88	Slave Receive Data NACK
0x50[1]	Master Receive Data ACK	0x70[1]	GC mode Address ACK
0x58	Master Receive Data NACK	0x78[1]	GC mode Arbitration Lost
0x00	Bus error	0x90[1]	GC mode Data ACK
		0x98	GC mode Data NACK
		0xB0[1]	Address Transmit Arbitration Lost
0xF8	Bus Released **Note:** Status “0xF8” exists in both master/slave modes, and it won’t raise interrupt.		
Note [1]: No interrupt in LPPDMA mode			

Table 6.22-2 LPI2C Status Code Description

Clock Baud Rate Bits (LPI2C_CLKDIV)

The data baud rate of LPI2C is determines by DIVIDER(LPI2C_CLKDIV [9:0]) register when LPI2C is in Master mode, and it is not necessary in a Slave mode. In the Slave mode, LPI2C will automatically synchronize it with any clock frequency from master LPI2C device. In the slave mode, system clock frequency should greater than LPI2C bus maximum clock 20 times.

The data baud rate of LPI2C setting is Data Baud Rate of LPI2C = (system clock) / (4x (LPI2C_CLKDIV [9:0] +1)). If system clock = 16 MHz, the LPI2C_CLKDIV [9:0] = 40 (28H), the data baud rate of LPI2C = 16 MHz/ (4x (40 +1)) = 97.5 kbit/s.

Time-out Control Register (LPI2C_TOCTL)

There is a 14-bit time-out counter which can be used to deal with the LPI2C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows (TOIF=1) and generates LPI2C interrupt to CPU or stops counting by clearing TOCEN to 0. When time-out counter is enabled, writing 1 to the SI flag will reset counter and re-start up counting after SI is cleared. If LPI2C bus hangs up, it causes the LPI2C_STATUS0 and flag SI are not updated for a period, the 14-bit time-out counter may overflow and acknowledge CPU the LPI2C interrupt. Refer to Figure 6.21-29 for the 14-bit time-out counter. User may write 1 to clear TOIF to 0.

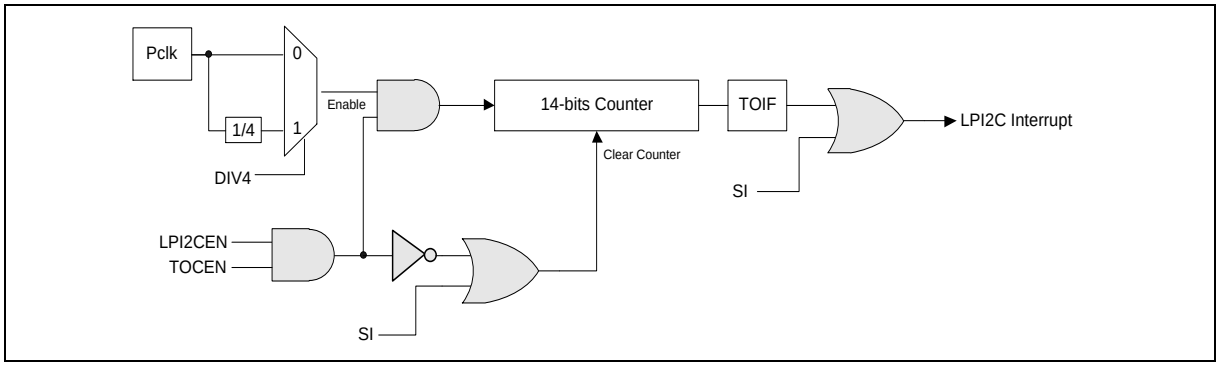


Figure 6.22-17 LPI2C Time-out Count Block Diagram

Wake-up Control Register (LPI2C_WKCTL)

When chip enters Power-down mode and set WKEN (LPI2C_WKCTL [0]) to 1, other LPI2C master can wake up the chip by addressing the LPI2C device, user must configure the related setting before entering sleep mode. The ACK bit cycle of address match frame is done in power-down. The controller will stretch the SCL to low when the address is matched the device's address and the ACK cycle done, then the LPI2C will go ahead. If NHDBUSEN (LPI2C_WKCTL [7]) is set, the controller will don't stretch the SCL to low. Note that when the controller don't stretch the SCL to low, transmit or receive data will perform immediately. If data transmitted or received when SI event is not clear, user must reset LPI2C and execute the original operation again.

Wake-up Status Register (LPI2C_WKSTS)

When system is woken up by other LPI2C master device, WKIF is set to indicate this event. User needs write "1" to clear this bit.

When the chip is woken-up by address match with one of the device address register (LPI2C_ADDRn), the user shall check the WKAKDONE (LPI2C_WKSTS [1]) bit is set to 1 to confirm the address byte has done. The WKAKDONE bit indicates that the ACK bit cycle of address byte is done in power-down. The controller will stretch the SCL to low when the address is matched the device's slave address and the ACK cycle done. The SCL is stretched until WKAKDONE is clear by user. If the frequency of SCL is low speed and the system has wakeup from address match frame, the user shall check WKAKDONE to confirm this frame has transaction done and then to do the wakeup procedure. Note that user cannot release WKIF through clearing the WKAKDONE bit to 0.

The WRSTSWK (LPI2C_WKSTS [2]) bit records the Read/Write command before the LPI2C sends address. The user can read this bit's status to prepare the next transmitted data (WRSTSWK = 1) or to wait the incoming data (WRSTSWK = 0) can be stored in time after the system is wake-up by the address match frame. Note that the WRSTSWK (LPI2C_WKSTS [2]) bit is cleared when writing one to the WKAKDONE (LPI2C_WKSTS [1]) bit.

When system is woken up by other LPI2C master device, WKIF is set to indicate this event. User needs to write "1" to clear this bit.

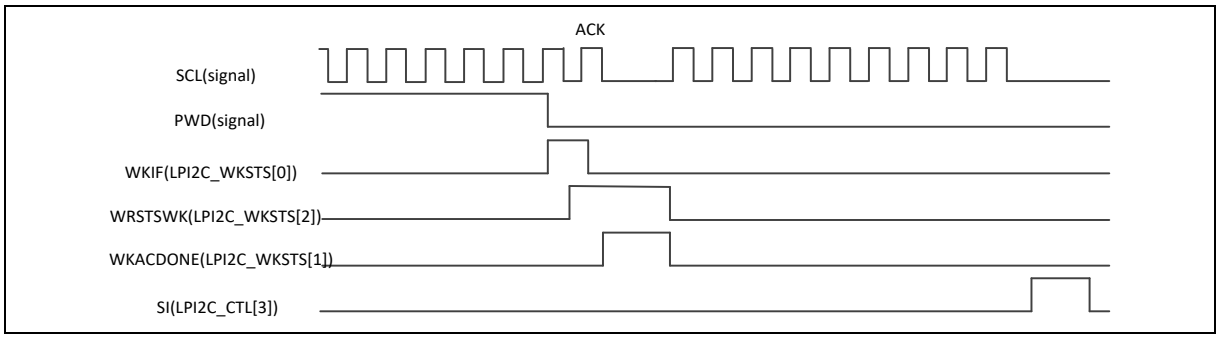


Figure 6.22-18 LPI2C Wake-Up Related Signals Waveform

LPI2C Control Register 1 (LPI2C_CTL1)

For LPPDMA function, set TXPDMAEN (LPI2C_CTL1 [0]) and RXPDMAEN (LPI2C_CTL1 [1]) can be set to operate. And set PDMARST (LPI2C_CTL1 [2]) to reset the LPPDMA control logic.

LPI2C Status Register 1 (LPI2C_STATUS1)

LPI2C supports four slave address flag registers, ADMAT0, ADMAT1, ADMAT2 and ADMAT3 (LPI2C_STATUS1[3:0]). Every control register represent which address is used and set 1 to inform software.

LPI2C Timing Configure Control Register (LPI2C_TMCTL)

In order to configure setup/hold time, the HTCTL (LPI2C_TMCTL[24:16]) and STCTL (LPI2C_TMCTL[8:0]) are set based on actual demand.

.

6.22.5.6 Example for Random Read on EEPROM

The following steps are used to configure the LPI2C0 related registers when using LPI2C to read data from EEPROM.

1. Set LPI2C0 the multi-function pin as SCL and SDA pins. Refer to the Basic Configuration section for the multi-function configuration.
2. Enable LPI2C0 APB clock. Refer to the Basic Configuration section for the clock configuration.
3. Set LPI2C0RST=1 to reset LPI2C0 controller then set LPI2C0 controller to normal operation. Refer to the Basic Configuration section for the reset controller configuration.
4. Set LPI2CEN=1 to enable LPI2C0 controller in the “LPI2C_CTL0” register.
5. Give LPI2C0 clock a divided register value for LPI2C clock rate in the “LPI2C_CLKDIV”.
6. Enable system LPI2C0 IRQ in system “NVIC” control register.
7. Set INTEN=1 to enable LPI2C0 Interrupt in the “LPI2C_CTL0” register.
8. Set LPI2C0 address registers “LPI2C_ADDR0 ~ LPI2C_ADDR3”.

Random read operation is one of the methods of access EEPROM. The method allows the master to access any address of EEPROM space. Figure 6.21-31 shows the EEPROM random read operation.

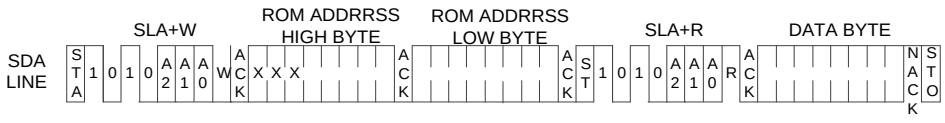


Figure 6.22-19 EEPROM Random Read

Figure 6.21-32 shows how to use LPI2C to implement the protocol of EEPROM random read.

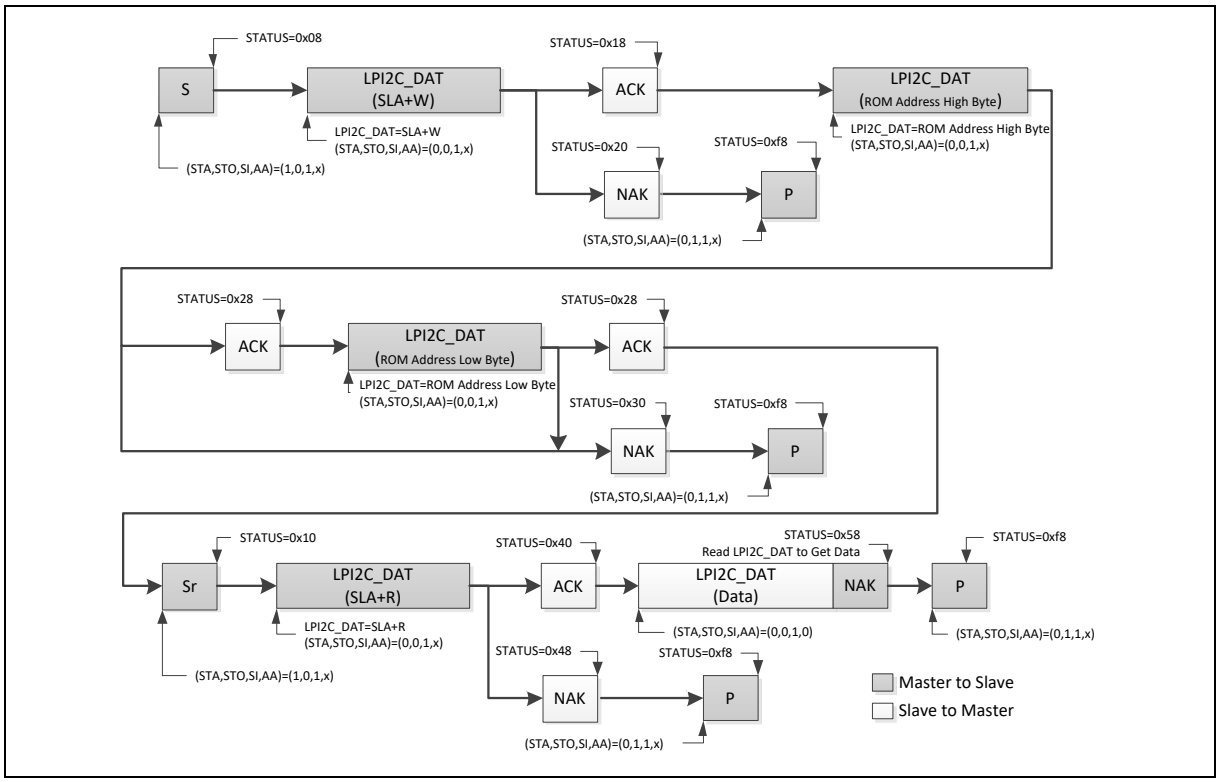


Figure 6.22-20 Protocol of EEPROM Random Read

LPI2C, which is a master, sends START to bus. Then, it sends a SLA+W (Slave address + Write bit) to EERPOM followed by two bytes data address to set the EEPROM address to read. Finally, a Repeat START followed by SLA+R is sent to read the data from EEPROM.

6.22.5.7 Automatic Operation Modes (auto-operation)

LPI2C supports three auto-operations in master mode while chip is in Power-down mode(NPD0, NPD1, NPD2, NPD3, NPD4). Three auto-operations are *Write Command*, *Read Command* and *Random Read Command*. Configure AUTOMODE (LPI2C_AUTOCTL [10:8]) and TRGEN (LPI2C_AUTOCTL[4]) first to do auto-operation. There are 4 kinds of sources to trigger auto-operation. Auto-operation can be triggered by LPTMR, TTMR, WKIO and SOFTWARE trigger.

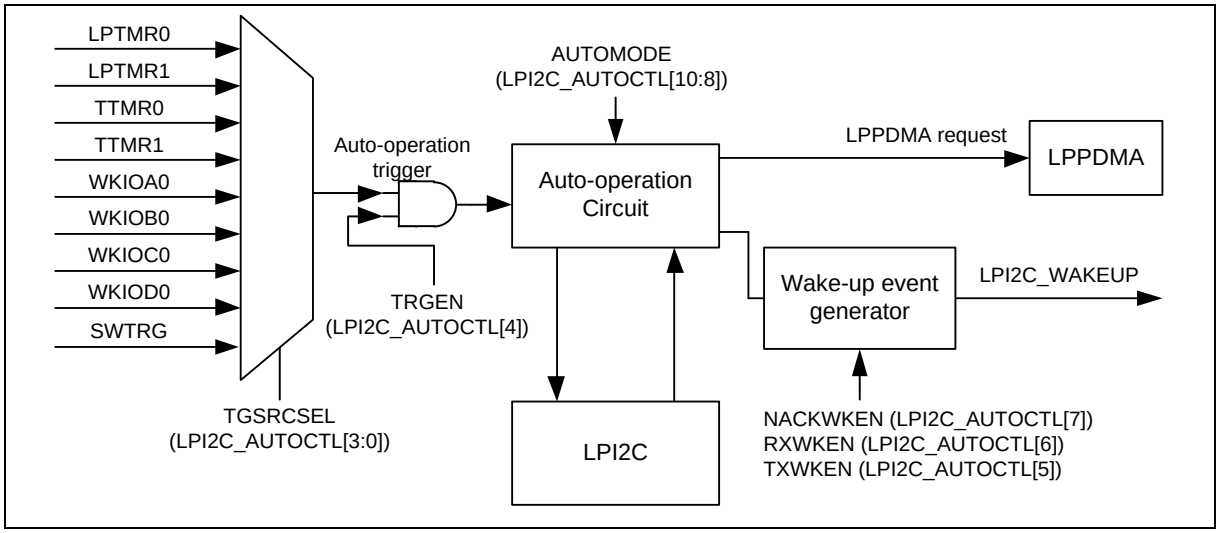


Figure 6.22-21 LPI2C Automatic Operation Block Diagram

6.22.5.8 Control Flow

Write command (Master TX):

1. Configure Trigger Source (LPTMR, TTMR or WKIO) and LPPDMA.
2. Enable LPI2C peripheral clock.
3. Select Trigger Source (LPTMR/TTMR/WKIO/SOFTWARE) (LPI2C_AUTOCTL[3:0])
4. Set AUTOMODE(LPI2C_AUTOCTL [10:8]) = 001, TRGEN(LPI2C_AUTOCTL[4] = 1). If CPU need to wakeup when receiving NACK in address phase(ADDR + W + NACK), set LPI2C NACKWKEN (LPI2C_AUTOCTL[7] = 1). If CPU need to wakeup after finishing Write Command, set TXWKEN(LPI2C_AUTOCTL[5] = 1).
5. Set TXCNT(LPI2C_AUTOCNT[7:0]) (TXCNT + 1).

Read Command (Master RX):

1. Configure Trigger Source (LPTMR, TTMR or WKIO) and LPPDMA.
2. Enable LPI2C peripheral clock.
3. Select Trigger Source (LPTMR/TTMR/WKIO/SOFTWARE) (LPI2C_AUTOCTL[3:0])
4. Set AUTOMODE(LPI2C_AUTOCTL [10:8]) = 010, TRGEN(LPI2C_AUTOCTL[4] = 1). If CPU need to wakeup when receiving NACK in address phase(ADDR + R + NACK), set LPI2C NACKWKEN (LPI2C_AUTOCTL[7] = 1). If CPU need to wakeup after finishing Read Command, set RXWKEN(LPI2C_AUTOCTL[6] = 1).
5. Set RXCNT(LPI2C_AUTOCNT[23:16]) (RXCNT + 1).

Random Read Command (Master TX then RX):

1. Configure Trigger Source (LPTMR, TTMR or WKIO) and LPPDMA. TX and RX channel need to be different LPPDMA channel.
2. Enable LPI2C peripheral clock.
3. Select Trigger Source (LPTMR/TTMR/WKIO/SOFTWARE) (LPI2C_AUTOCTL[3:0])
4. Set AUTOMODE(LPI2C_AUTOCTL [10:8]) = 011(repeat start between TX and RX transfer) or 100(stop first then start between TX and RX transfer), TRGEN (LPI2C_AUTOCTL[4] = 1). If CPU need to wakeup when receiving NACK in address phase(ADDR + W + NACK), set

LPI2C NACKWKEN (LPI2C_AUTOCTL[7] = 1). If CPU need to wakeup after finishing Read Command, set RXWKEN(LPI2C_AUTOCTL[6] = 1). If CPU need to wakeup after finishing Write Command, set TXWKEN(LPI2C_AUTOCTL[5] = 1).

5. Set TXCNT(LPI2C_AUTOCNT[7:0]) (TXCNT + 1) and RXCNT(LPI2C_AUTOCNT[23:16]) (RXCNT + 1). RXCNT must be set at 1 or above when AUTOMODE (LPI2C_AUTOCTL[10:8]) is 3'b011 or 3'b100.

6.22.6 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
LPI2C Base Address: **LPI2Cn_BA = 0x400E_2000 + (0x1000 *n)** **n= 0**				
LPI2C_CTL0	LPI2Cn_BA+0x00	R/W	LPI2C Control Register 0	0x0000_0000
LPI2C_ADDR0	LPI2Cn_BA+0x04	R/W	LPI2C Slave Address Register0	0x0000_0000
LPI2C_DAT	LPI2Cn_BA+0x08	R/W	LPI2C Data Register	0x0000_0000
LPI2C_STATUS0	LPI2Cn_BA+0x0C	R	LPI2C Status Register 0	0x0000_00F8
LPI2C_CLKDIV	LPI2Cn_BA+0x10	R/W	LPI2C Clock Divided Register	0x0000_0000
LPI2C_TOCTL	LPI2Cn_BA+0x14	R/W	LPI2C Time-out Control Register	0x0000_0000
LPI2C_ADDR1	LPI2Cn_BA+0x18	R/W	LPI2C Slave Address Register1	0x0000_0000
LPI2C_ADDR2	LPI2Cn_BA+0x1C	R/W	LPI2C Slave Address Register2	0x0000_0000
LPI2C_ADDR3	LPI2Cn_BA+0x20	R/W	LPI2C Slave Address Register3	0x0000_0000
LPI2C_ADDRMSK0	LPI2Cn_BA+0x24	R/W	LPI2C Slave Address Mask Register0	0x0000_0000
LPI2C_ADDRMSK1	LPI2Cn_BA+0x28	R/W	LPI2C Slave Address Mask Register1	0x0000_0000
LPI2C_ADDRMSK2	LPI2Cn_BA+0x2C	R/W	LPI2C Slave Address Mask Register2	0x0000_0000
LPI2C_ADDRMSK3	LPI2Cn_BA+0x30	R/W	LPI2C Slave Address Mask Register3	0x0000_0000
LPI2C_WKCTL	LPI2Cn_BA+0x3C	R/W	LPI2C Wake-up Control Register	0x0000_0000
LPI2C_WKSTS	LPI2Cn_BA+0x40	R/W	LPI2C Wake-up Status Register	0x0000_0000
LPI2C_CTL1	LPI2Cn_BA+0x44	R/W	LPI2C Control Register 1	0x0000_0000
LPI2C_STATUS1	LPI2Cn_BA+0x48	R/W	LPI2C Status Register 1	0x0000_0000
LPI2C_TMCTL	LPI2Cn_BA+0x4C	R/W	LPI2C Timing Configure Control Register	0x0002_0000
LPI2C_AUTOCTL	LPI2Cn_BA+0x70	R/W	LPI2C Automatic Operation Control Register	0x0000_0000
LPI2C_AUTOSTS	LPI2Cn_BA+0x74	R/W	LPI2C Automatic Operation Status Register	0x0000_0000
LPI2C_AUTOCNT	LPI2Cn_BA+0x78	R/W	LPI2C Automatic Operation Transfer Count Register	0x0000_0000

6.22.7 Register Description

LPI2C Control Register (LPI2C_CTL0)

Register	Offset	R/W	Description	Reset Value
LPI2C_CTL0	LPI2Cn_BA+0x00	R/W	LPI2C Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTEN	**LPI2CEN**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**INTEN**	**Enable Interrupt** 0 = LPI2C interrupt Disabled. 1 = LPI2C interrupt Enabled.
[6]	**LPI2CEN**	**LPI2C Controller Enable Bit** Set to enable LPI2C serial function controller. When LPI2CEN=1 the LPI2C serial function enable. The multi-function pin function must set to SDA, and SCL of LPI2C function first. 0 = LPI2C controller Disabled. 1 = LPI2C controller Enabled.
[5]	**STA**	**LPI2C START Control** Setting STA to logic 1 to enter Master mode, the LPI2C hardware sends a START or repeat START condition to bus when the bus is free. This bit will be cleared by hardware automatically.
[4]	**STO**	**LPI2C STOP Control** In Master mode, setting STO to transmit a STOP condition to bus then LPI2C will check the bus condition if a STOP condition is detected. This bit will be cleared by hardware automatically.
[3]	**SI**	**LPI2C Interrupt Flag** When a new LPI2C state is present in the LPI2C_STATUS0 register, the SI flag is set by hardware. If bit INTEN (LPI2C_CTL0 [7]) is set, the LPI2C interrupt is requested. SI must be cleared by software. Clear SI by writing 1 to this bit.
[2]	**AA**	**Assert Acknowledge Control** When AA =1 prior to address or data is received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.) A slave is acknowledging the address sent from master, 2.) The receiver devices are acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line.
[1:0]	**Reserved**	Reserved.

LPI2C Data Register (LPI2C_DAT)

Register	Offset	R/W	Description	Reset Value
LPI2C_DAT	LPI2Cn_BA+0x08	R/W	LPI2C Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAT							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DAT**	**LPI2C Data** Bit [7:0] is located with the 8-bit transferred/received data of LPI2C serial port.

LPI2C Status Register (LPI2C_STATUS0)

Register	Offset	R/W	Description	Reset Value
LPI2C_STATUS0	LPI2Cn_BA+0x0C	R	LPI2C Status Register 0	0x0000_00F8

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STATUS							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**STATUS**	**LPI2C Status** The three least significant bits are always 0. The five most significant bits contain the status code. There are 28 possible status codes. When the content of LPI2C_STATUS0 is F8H, no serial interrupt is requested. Others LPI2C_STATUS0 values correspond to defined LPI2C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in LPI2C_STATUS0 one cycle after SI is set by hardware and is still present one cycle after SI has been reset by software. In addition, states 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the formation frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit.

LPI2C Clock Divided Register (LPI2C_CLKDIV)

Register	Offset	R/W	Description	Reset Value
LPI2C_CLKDIV	LPI2Cn_BA+0x10	R/W	LPI2C Clock Divided Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NFCNT				**Reserved**		**DIVIDER**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:12]	**NFCNT**	**Noise Filter Count** The register bits control the input filter width. If the pulse width is narrower than the setting((3+N)*PCLK), it will be ignored. 0 = Filter width 3*PCLK 1 = Filter width 4*PCLK *N* = Filter width (3+*N*)*PCKL **Note:** Filter width Min :3*PCLK, Max : 18*PCLK
[11:10]	**Reserved**	Reserved.
[9:0]	**DIVIDER**	**LPI2C Clock Divider** Indicates the LPI2C clock rate: Data Baud Rate of LPI2C = (system clock) / (4x (DIVDER+1)). **Note:** The minimum value of DIVIDER is 4.

LPI2C Time-out Control Register (LPI2C_TOCTL)

Register	Offset	R/W	Description	Reset Value
LPI2C_TOCTL	LPI2Cn_BA+0x14	R/W	LPI2C Time-out Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					TOCEN	TOCDIV4	TOIF

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**TOCEN**	**Time-out Counter Enable Bit** When enabled, the 14-bit time-out counter will start counting when SI is cleared. Setting flag SI to '1' will reset counter and re-start up counting after SI is cleared. 0 = Time-out counter Disabled. 1 = Time-out counter Enabled.
[1]	**TOCDIV4**	**Time-out Counter Input Clock Divided by 4** When enabled, the time-out period is extended 4 times. 0 = Time-out period is extend 4 times Disabled. 1 = Time-out period is extend 4 times Enabled.
[0]	**TOIF**	**Time-out Flag** This bit is set by hardware when LPI2C time-out happened and it can interrupt CPU if LPI2C interrupt enable bit (INTEN) is set to 1. **Note:** Software can write 1 to clear this bit.

LPI2C Slave Address Register (ADDRx)

Register	Offset	R/W	Description	Reset Value
LPI2C_ADDR0	LPI2Cn_BA+0x04	R/W	LPI2C Slave Address Register0	0x0000_0000
LPI2C_ADDR1	LPI2Cn_BA+0x18	R/W	LPI2C Slave Address Register1	0x0000_0000
LPI2C_ADDR2	LPI2Cn_BA+0x1C	R/W	LPI2C Slave Address Register2	0x0000_0000
LPI2C_ADDR3	LPI2Cn_BA+0x20	R/W	LPI2C Slave Address Register3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDR							**GC**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**ADDR**	**LPI2C Address** The content of this register is irrelevant when LPI2C is in Master mode. In the slave mode, the seven most significant bits must be loaded with the chip's own address. The LPI2C hardware will react if either of the address is matched. **Note:** When software sets 7'h000, the address cannot be used.
[0]	**GC**	**General Call Function** 0 = General Call function Disabled. 1 = General Call function Enabled.

LPI2C Slave Address Mask Register (ADDRMSKx)

Register	Offset	R/W	Description	Reset Value
LPI2C_ADDRMSK0	LPI2Cn_BA+0x24	R/W	LPI2C Slave Address Mask Register0	0x0000_0000
LPI2C_ADDRMSK1	LPI2Cn_BA+0x28	R/W	LPI2C Slave Address Mask Register1	0x0000_0000
LPI2C_ADDRMSK2	LPI2Cn_BA+0x2C	R/W	LPI2C Slave Address Mask Register2	0x0000_0000
LPI2C_ADDRMSK3	LPI2Cn_BA+0x30	R/W	LPI2C Slave Address Mask Register3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDRMSK							**Reserved**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**ADDRMSK**	**LPI2C Address Mask** 0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don't care.). LPI2C bus controllers support multiple address recognition with four address mask register. When the bit in the address mask register is set to one, it means the received corresponding address bit is don't-care. If the bit is set to zero, that means the received corresponding register bit should be exact the same as address register.
[0]	**Reserved**	Reserved.

LPI2C Wake-up Control Register (LPI2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
LPI2C_WKCTL	LPI2Cn_BA+0x3C	R/W	LPI2C Wake-up Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NHDBUSEN	**Reserved**						**WKEN**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**NHDBUSEN**	**LPI2C No Hold BUS Enable Bit** 0 = LPI2C hold bus after wake-up. 1= LPI2C didn't hold bus after wake-up. **Note:** LPI2C could respond when WKIF event is not clear, it may cause error data transmitted or received. If data transmitted or received when WKIF event is not clear, user must reset LPI2C and execute the original operation again.
[6:1]	**Reserved**	Reserved.
[0]	**WKEN**	**LPI2C Wake-up Enable Bit** 0 = LPI2C wake-up function Disabled. 1 = LPI2C wake-up function Enabled. **Note 1:** This is for slave address match wakeup. **Note 2:** Not support in NPD5 Mode. Disable this bit before entering NPD5.

LPI2C Wake-up Status Register (LPI2C_WKSTS)

Register	Offset	R/W	Description	Reset Value
LPI2C_WKSTS	LPI2Cn_BA+0x40	R/W	LPI2C Wake-up Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**WRSTSWK**	**WKAKDONE**	**WKIF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**WRSTSWK**	**Read/Write Status Bit in Address Wakeup Frame** 0 = Write command be record on the address match wakeup frame. 1 = Read command be record on the address match wakeup frame. **Note:** This bit will be cleared when software can write 1 to WKAKDONE bit.
[1]	**WKAKDONE**	**Wakeup Address Frame Acknowledge Bit Done** 0 = The ACK bit cycle of address match frame isn't done. 1 = The ACK bit cycle of address match frame is done in power-down. **Note:** This bit cannot release WKIF. Software can write 1 to clear this bit.
[0]	**WKIF**	**LPI2C Wake-up Flag** When chip is woken up from Power-down mode by LPI2C, this bit is set to 1. Software can write 1 to clear this bit.

LPI2C Control Register 1 (LPI2C_CTL1)

Register	Offset	R/W	Description	Reset Value
LPI2C_CTL1	LPI2Cn_BA+0x44	R/W	LPI2C Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					SWITCHEN	Reserved	PDMASTR
7	6	5	4	3	2	1	0
Reserved					PDMARST	RXPDMAEN	TXPDMAEN

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**SWITCHEN**	**SCL And SDA Pin Switch Enable Bit** 0 = LPI2C use original pin configuration. 1 = LPI2C switch SCL and SDA pin configuration. **Note:** Original pin configuration table is shown in Basic Configuration chapter.
[9]	**Reserved**	Reserved.
[8]	**PDMASTR**	**LPPDMA Stretch Bit** 0 = LPI2C send STOP automatically after LPPDMA transfer done. (only master TX) 1 = LPI2C SCL bus is stretched by hardware after LPPDMA transfer done if the SI is not cleared. (only master TX)
[7:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**LPPDMA Reset** 0 = No effect. 1 = Reset the LPI2C request to LPPDMA.
[1]	**RXPDMAEN**	**LPPDMA Receive Channel Available** 0 = Receive LPPDMA function Disabled. 1 = Receive LPPDMA function Enabled.
[0]	**TXPDMAEN**	**LPPDMA Transmit Channel Available** 0 = Transmit LPPDMA function Disabled. 1 = Transmit LPPDMA function Enabled.

LPI2C Status Register 1 (LPI2C_STATUS1)

Register	Offset	R/W	Description	Reset Value
LPI2C_STATUS1	LPI2Cn_BA+0x48	R/W	LPI2C Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							ONBUSY
7	6	5	4	3	2	1	0
Reserved				ADMAT3	ADMAT2	ADMAT1	ADMAT0

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**ONBUSY**	**On Bus Busy (Read Only)** Indicates that a communication is in progress on the bus. It is set by hardware when a START condition is detected. It is cleared by hardware when a STOP condition is detected. 0 = The bus is IDLE (both SCL and SDA High). 1 = The bus is busy.
[7:4]	**Reserved**	Reserved.
[3]	**ADMAT3**	**LPI2C Address 3 Match Status** When address 3 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[2]	**ADMAT2**	**LPI2C Address 2 Match Status** When address 2 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[1]	**ADMAT1**	**LPI2C Address 1 Match Status** When address 1 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[0]	**ADMAT0**	**LPI2C Address 0 Match Status** When address 0 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.

LPI2C Timing Configure Control Register (LPI2C_TMCTL)

Register	Offset	R/W	Description	Reset Value
LPI2C_TMCTL	LPI2Cn_BA+0x4C	R/W	LPI2C Timing Configure Control Register	0x0002_0000

31	30	29	28	27	26	25	24
Reserved							**HTCTL**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HTCTL							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**STCTL**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STCTL							

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24:16]	**HTCTL**	**Hold Time Configure Control** This field is used to generate the delay timing between SCL falling edge and SDA rising edge in transmission mode. The delay hold time is numbers of peripheral clock = HTCTL x PCLK.
[15:9]	**Reserved**	Reserved.
[8:0]	**STCTL**	**Setup Time Configure Control** This field is used to generate a delay timing between SDA falling edge and SCL rising edge in transmission mode. The delay setup time is numbers of peripheral clock = STCTL x PCLK. **Note:** Setup time setting should not make SCL output less than three PCLKs.

LPI2C Automatic Operation Control Register (LPI2C_AUTOCTL)

Register	Offset	R/W	Description	Reset Value
LPI2C_AUTOCTL	LPI2Cn_BA+0x70	R/W	LPI2C Automatic Operation Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SWTRG	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					AUTOMODE		
7	6	5	4	3	2	1	0
NACKWKEN	RXWKEN	TXWKEN	TRGEN	TGSRCSEL			

Bits	Description	
[31]	**SWTRG**	**Auto-operation Mode Software Trigger Bit** 0 = No operation. 1 = Software trigger auto-operattion. **Note:** TGSRCSEL(LPI2C_AUTOCTL[3:0]) need to be set to 0x8 first. This bit will be cleared by hardware automatically.
[30:11]	**Reserved**	Reserved.
[10:8]	**AUTOMODE**	**Auto-operation Mode Select** 000 = No auto-operation. 001 = Auto TXPDMA transfer mode. 010 = Auto RXPDMA transfer mode. 011 = Random Read mode, repeat start between TX/RX. 100 = Random Read mode, stop and start between TX/RX. Others = Reserved. **Note:** Use TXWKEN**(**LPI2C_AUTOCTL[5]**)** to wake up system when using auto TXPDMA transfer mode AUTOMODE(LPI2C_AUTOCTL[10:8]) in Power-down mode.
[7]	**NACKWKEN**	**Receive Slave NACK Wakeup Enable bit** 0 = Stop after receiving NACK. 1 = Wakeup after receiving NACK.
[6]	**RXWKEN**	**RX Transfer Count Match Wakeup Enable bit** 0 = Power off after RX transfer count matches RXCNT. 1 = Wakeup after RX transfer count matches RXCNT.
[5]	**TXWKEN**	**TX Transfer Count Match Wakeup Enable bit** 0 = Power off after TX transfer count matches TXCNT. 1 = Wakeup after TX transfer count matches TXCNT.
[4]	**TRGEN**	**Trigger Source Enable bit** 0 = LPI²C trigger source disable. 1 = LPI²C trigger source enable.

[3:0]	**TGSRCSEL**	**Low Power Auto-operation Trigger Source Select** 0000 = Low Power Auto-operation Trigger Source from LPTMR0. 0001 = Low Power Auto-operation Trigger Source from LPTMR1. 0010 = Low Power Auto-operation Trigger Source from TTMR0. 0011 = Low Power Auto-operation Trigger Source from TTMR1. 0100 = Low Power Auto-operation Trigger Source from WKIOA0(GPA0). 0101 = Low Power Auto-operation Trigger Source from WKIOB0(GPB0). 0110 = Low Power Auto-operation Trigger Source from WKIOC0(GPC0). 0111 = Low Power Auto-operation Trigger Source from WKIOD0(GPD0). 1000 = Low Power Auto-operation Trigger Source from Software setting SWTRG(LPI2C_AUTOCTL[31]) to 1. 1001~1111 = Reserved.

LPI2C Automatic Operation Status Register (LPI2C_AUTOSTS)

Register	Offset	R/W	Description	Reset Value
LPI2C_AUTOSTS	LPI2Cn_BA+0x74	R/W	LPI2C Automatic Operation Status Register	0x0000_0000

31	30	29	28	27	26	25	24
BUSY	Reserved						AOFINISH
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					ERRORIF	RXFINISH	TXFINISH
7	6	5	4	3	2	1	0
Reserved					ERRORWKF	RXWKF	TXWKF

Bits	Description	
[31]	**BUSY**	**Busy Flag** When chip is woken up and this bit is set to 1, it means that there are one or more requests from trigger source during transfer. Software can write 1 to clear this bit.
[30:25]	**Reserved**	Reserved.
[24]	**AOFINISH**	**Automatic Operation Finish Flag** AOFINISH is set when finishing one round of auto operation, but it will not issue interrupt. Software can write 1 to clear this bit.
[23:11]	**Reserved**	Reserved.
[10]	**ERRORIF**	**ERROR Interrupt Flag** When LPI2C receiving NACK from slave, running into bus error or arbitration lost, this bit is set to 1. Software can write 1 to clear this bit.
[9]	**RXFINISH**	**Automatic Operation RX Finish Flag** RXFINISH is set when finishing one round of auto operation(Automode = 2/3/4), but it will not issue interrupt. Software can write 1 to clear this bit.
[8]	**TXFINISH**	**Automatic Operation TX Finish Flag** AOFINISH is set when finishing one round of auto operation(Automode = 1), but it will not issue interrupt. Software can write 1 to clear this bit.
[7:3]	**Reserved**	Reserved.
[2]	**ERRORWKF**	**Error Condition Wakeup Flag** When chip is woken up from Power-down mode by LPI2C receiving NACK from slave, running into bus error or arbitration lost, this bit is set to 1. Software can write 1 to clear this bit.
[1]	**RXWKF**	**RX Transfer Count Match Wakeup Flag** When chip is woken up from Power-down mode by LPI2C RX transfer count match, this bit is set to 1. Software can write 1 to clear this bit.
[0]	**TXWKF**	**TX Transfer Count Match Wakeup Flag** When chip is woken up from Power-down mode by LPI2C TX transfer count match, this bit is set to 1. Software can write 1 to clear this bit.

LPI2C Automatic Operation Transfer Count Register (LPI2C_AUTOCNT)

Register	Offset	R/W	Description	Reset Value
LPI2C_AUTOCNT	LPI2Cn_BA+0x78	R/W	LPI2C Automatic Operation Transfer Count Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXCNT							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**RXCNT**	**RX Transfer Count** Bit [23:16] is set for RXPDMA and Random Read transfer mode. RXCNT represents the required number of the transfer, the real transfer count is RXCNT + 1. The maximum transfer count is 255 + 1. **Note:** RXCNT must be set at 1 or above when AUTOMODE (LPI2C_AUTOCTL[10:8]) is 3'b011 or 3'b100
[15:8]	**Reserved**	Reserved.
[7:0]	**TXCNT**	**TX Transfer Count** Bit [7:0] is set for TXPDMA and Random Read transfer mode. TXCNT represents the required number of the transfer, the real transfer count is TXCNT + 1. The maximum transfer count is 255 + 1.

6.23 USCI - Universal Serial Control Interface Controller (USCI)

6.23.1 Overview

The Universal Serial Control Interface (USCI) is a flexible interface module covering several serial communication protocols. The user can configure this controller as UART, SPI, or I²C functional protocol.

6.23.2 Features

The controller can be individually configured to match the application needs. The following protocols are supported:

- UART
- SPI
- I²C

6.23.3 Block Diagram

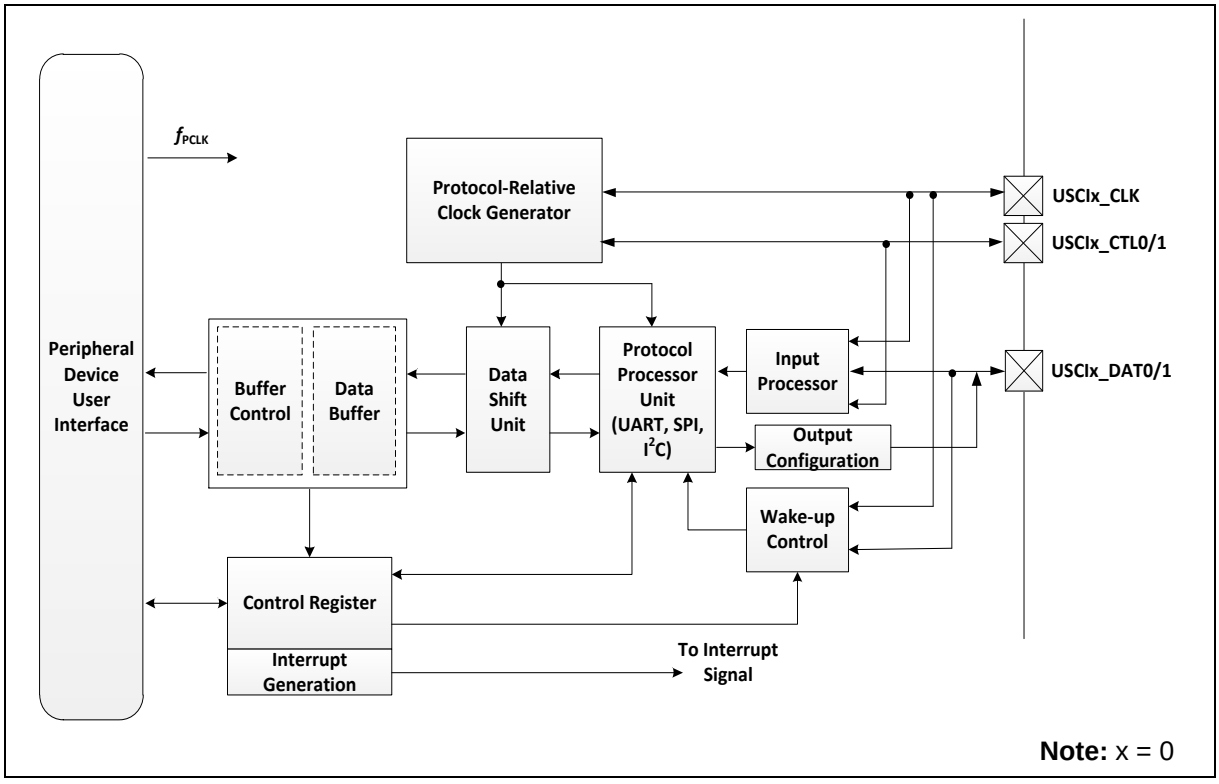


Figure 6.23-1 USCI Block Diagram

6.23.4 Functional Description

The structure of the Universal Serial Control Interface (USCI) controller is shown in Figure 6.23-1 USCI Block Diagram. The input signal is implemented in input processor. The data buffers and the data shift unit support the data transfers. Each protocol-specific function is handled by the protocol processor unit. The timing and time event control signals of the specific protocol are handled by the protocol-relative clock generator. All the protocol-specific events are processed in the interrupt generation unit. The wake-up function of the specific protocol is implemented in the wake-up control unit.

The USCI is equipped with three protocols including UART, SPI, and I²C. They can be selected by FUNMODE (USCI_CTL [2:0]). Note that the FUNMODE must be set to 0 before changing protocol.

6.23.4.1 I/O Processer

Input Signal

All input stages offer the similar feature set. They are used for all protocols.

Table 6.23-1lists the relative input signals for each selected protocol. Each input signal is handled by an input processor for signal conditioning, such as signal inverse selection control, or a digital input filter.

Selected Protocol		UART	SPI	I²C
Serial Bus Clock Input	USCIx_CLK	-	SPI_CLK	SCL
Control Input	USCIx_CTL0	nCTS	SPI_SS	-
	USCIx_CTL1	-	-	-
Data Input	USCIx_DAT0	RX	SPI_MOSI_0	SDA
	USCIx_DAT1	-	SPI_MISO_0	-

Table 6.23-1 Input Signals for Different Protocols

The description of protocol-specific items are given in the related protocol chapters.

General Input Structure

The input structures of data and control signals include inverter, digital filter and edge detection (data signal only).

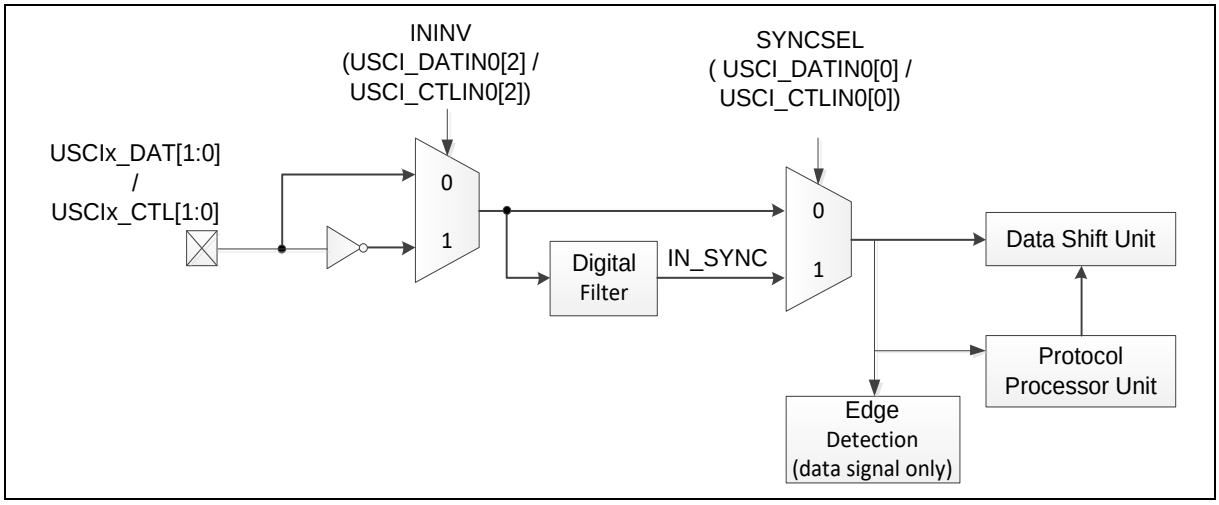


Figure 6.23-2 Input Conditioning for USCIx_DAT[1:0] and USCIx_CTL[1:0]

The input structure of USCIx_CLK is similar to USCIx_CTL[1:0] input structure, except it does not support inverse function.

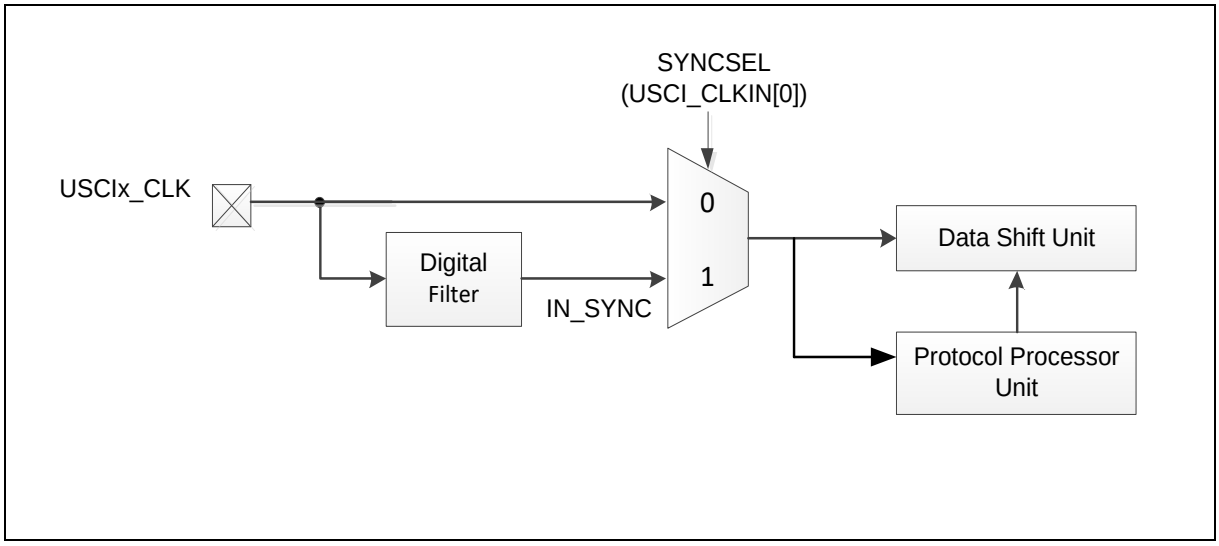


Figure 6.23-3 Input Conditioning for USCIx_CLK

All configurations of control, clock and data input structures are in USCI_CTLIN0, USCI_CLKIN and USCI_DATIN0 registers respectively. EDGEDET (USCI_DATIN0[4:3]) is used to select the edge detection condition. Note that the EDGEDET for USCI_DATIN0 must be set 2'b10 in UART mode. The programmable edge detection indicates that the desired event has occurred by activating the trigger signal.

ININV (USCI_DATIN0[2] / USCI_CTLIN0[2]) allows a polarity inversion of the selected input signal to adapt the input signal polarity to the internal polarity of the data shift unit and the protocol state machine.

If the SYNCSEL (USCI_DATIN0[0] / USCI_CTLIN0[0] / USCI_CLKIN[0]) is set to 0, the paths of input signals do not contain any delay due to synchronization or filtering. If there is noise on the input signals, there is the possibility to synchronize the input signal (signal IN_SYNC is synchronized to f_{PCLK}). The synchronized input signal is taken into account by SYNCSEL = 1. The synchronization leads to a delay in the signal path of 2-3 times the period of f_{PCLK}.

Output Signals

Table 6.23-2 shows the relative output signals for each protocol. The number of actually used outputs depends on the selected protocol and they can be classified according to their meaning for the protocols.

Selected Protocol		UART	SPI	I²C
Serial Bus Clock Output	USCIx_CLK	-	SPI_CLK	SCL
Control Output	USCIx_CTL0	-	SPI_SS	-
	USCIx_CTL1	nRTS	-	-
Data Output	USCIx_DAT0	-	SPI_MOSI_0	SDA
	USCIx_DAT1	TX	SPI_MISO_0	-

Table 6.23-2 Output Signals for Different Protocols

The description of protocol-specific items are given in the related protocol chapters.

6.23.4.2 Data Buffering

The data handling of the USCI controller is based on a Data Shift Unit (DSU) and a buffer structure. Both of the data shift and buffer registers are 16-bit wide. The inputs of Data Shift Unit include the shift data, the serial bus clock, and the shift control. The output pin of transmission can be USCIx_DAT0 pin or USCIx_DAT1 pin depends on what protocol is selected.

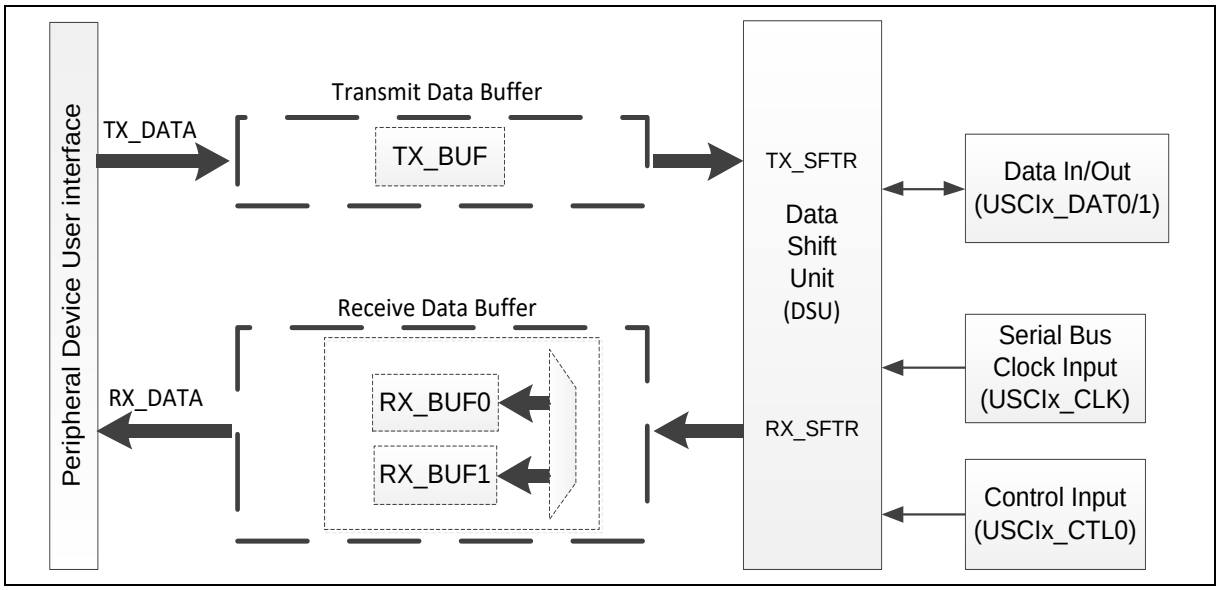


Figure 6.23-4 Block Diagram of Data Buffering

The operation of data handling includes:

- The peripheral device user interface (APB) is used to handle data, interrupts, status and control information.
- A transmitter includes transmit shift register (TX_SFTR) and a transmit data buffer (TX_BUF). The TXFULL / TXEMPTY (USCI_BUFSTS[9:8]) and TXENDIF (USCI_PROTSTS[2]) can indicate the status of transmitter.
- A receiver includes receive shift register (RX_SFTR) and a double receive buffer structure (RX_BUF0, RX_BUF1). In double buffer structure, user need not care about the reception sequence and two received data can be hold if user does not read the data of USCI_RXDAT register in time.

Data Access Structure

The Data Access Structure includes read access to received data and write access of data to be transmitted. The received data is stored in the receiver buffers including RX_BUF0 and RX_BUF1. User need not care about the reception sequence. The receive buffer can be accessed by reading USCI_RXDAT register. The first received data is read out first and the next received data becomes visible in USCI_RXDAT and can be read out next.

Transmit data can be loaded to TX_BUF by writing to the transmit register USCI_TXDAT.

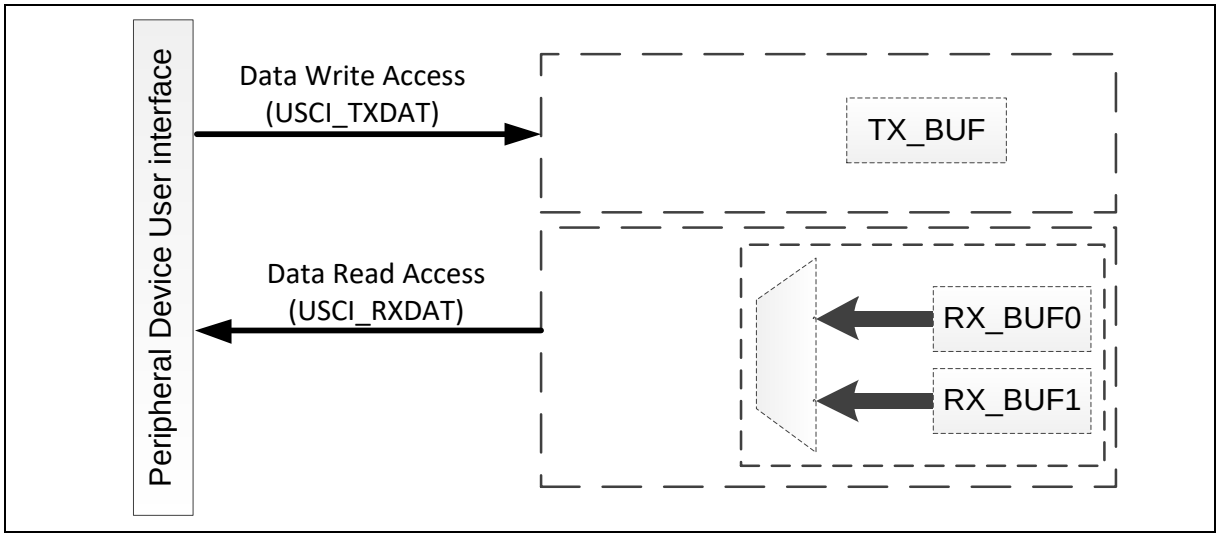


Figure 6.23-5 Data Access Structure

Transmit Data Path

The transmit data path is based on 16-bit wide transmit shift register (TX_SFTR) and transmit buffer TX_BUF. The data transfer parameters like data word length is controlled commonly for transmission and reception by the line control register USCI_LINECTL.

Transmit Buffering

The transmit shift register cannot be directly accessed by user. It is updated automatically with the value stored in the transmit buffer (TX_BUF) if a currently transmitted data is finished and new data is valid for transmission.

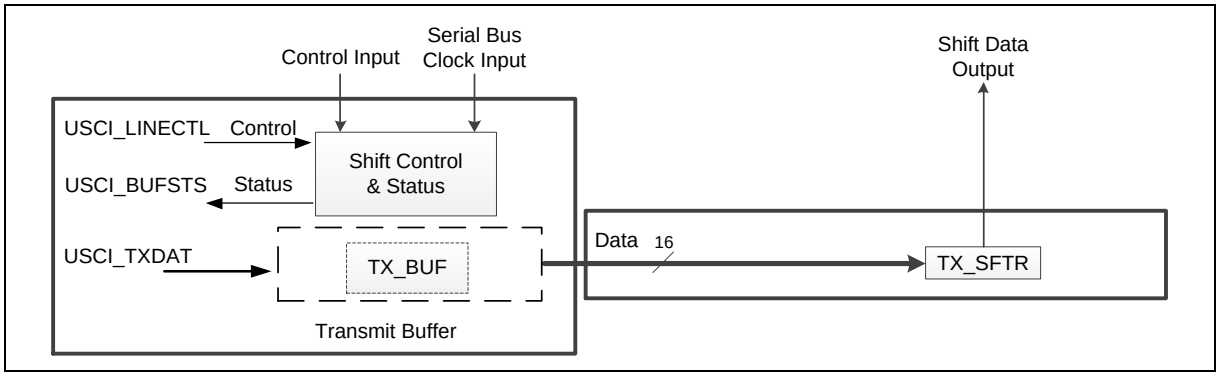


Figure 6.23-6 Transmit Data Path

Transmit Data Validation

The status of TXEMPTY (USCI_BUFSTS[8]) indicates the transmission data is valid or not in the transmit buffer (TX_BUF) and the TXSTIF (USCI_PROTSTS[1]) labels the start conditions for each data.

- If the USCI controller is a Master, the data transfer can only be started with valid data in the transmit buffer (TX_BUF). In this case, the transmit shift register is loaded with the content of transmit buffer.

  Note: Master defines the start of data transfer.

- If the USCI controller is a Slave, a data transfer requested by Master and it has to be started independently of the status in transmit buffer (TX_BUF). If a data transfer is requested and started by the Master, the transmit shift register is loaded from specific

protocol control signal if it is valid for transmission.

Note: Slave can not define the start itself, but has to react.

- The timing of loading data from transmit buffer to data shift unit depends on protocol configurations.
- **UART:** A transmission of the data word in transmit buffer can be started if TXEMPTY = 0 in normal operation. In auto flow control, A transmission of the data word in transmit buffer can be started while TXEMPTY = 0 and USCIx_CTL0 in active stage.
- **SPI:** In Master mode, data transmission will be started when TXEMPTY (USCI_BUFSTS[8]) is 0. In Slave mode, the data transmission can be started only when slave selection signal is at active state and clock is presented on USCIx_CLK pin.
- **I²C:** A transmission of the data byte in transmit buffer can be started if TXEMPTY = 0.
- A transmission data which is located in transmit buffer can be started if the TXEMPTY (USCI_BUFSTS [8]) = 0. The content of the transmit buffer (in TX_BUF condition) should not be overwritten with new data while it is valid for transmission and a new transmission can start. If the content of TX_BUF has to be changed, user can set TXRST (USCI_BUFCTL [16]) to 1 to clear the content of TX_BUF before updating the data. Moreover, TXEMPTY (USCI_BUFSTS [8]) will be cleared automatically when transmit buffer (TX_BUF) is updated with new data. While a transmission is in progress, TX_BUF can be loaded with new data. User has to update the TX_BUF before a new transmission.

Receive Data Path

The receive data path is based on 16-bit wide receive shift register RX_SFTR and receive buffers RX_BUF0 and RX_BUF1. The data transfer parameters like data word length, or the shift direction are controlled commonly for transmission and reception by the line control register USCI_LINECTL. Register USCI_BUFSTS monitors the data validation of USCI_RXDAT.

Receive Buffering

The receive shift register cannot be directly accessed by user, but its content is automatically loaded into the receive buffer if a complete data word has been received or the frame is finished. The received data words in Receive Buffer can be read out automatically from register USCI_RXDAT.

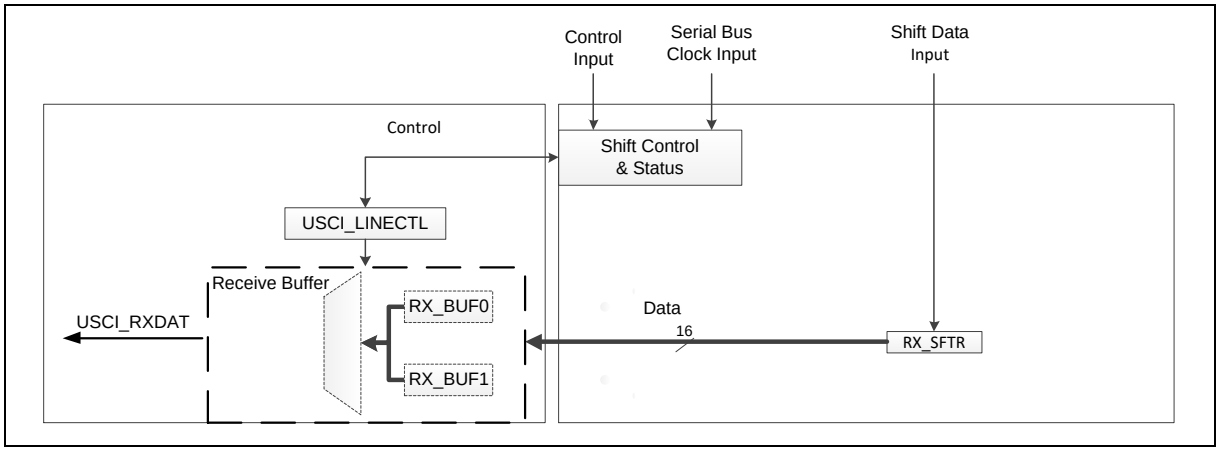


Figure 6.23-7 Receive Data Path

6.23.4.3 Port Direction Control

In SPI protocol with half-duplex configurations, the data port is bidirectional. Port direction control is intended to control the pin direction through a dedicated hardware interface.

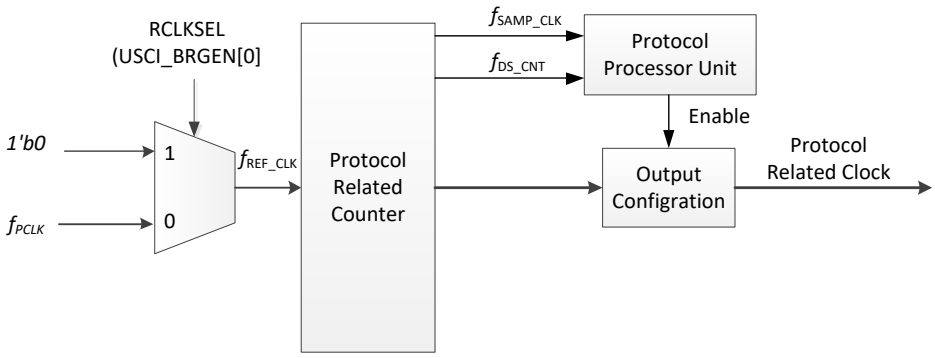
The direction of selected pin is controlled by PORTDIR (USCI_TXDAT[16]). When user writes USCI_TXDAT register, the transmit data and its port direction are settled simultaneously.

6.23.4.4 Protocol Control and Status

The protocol-related control and status information are located in the protocol control register USCI_PROTCTL and in the protocol status register USCI_PROTSTS. These registers are shared between the available protocols. As a consequence, the meaning of the bit positions in these registers is different within the protocols. Refer to each protocol's relative register for detail information.

6.23.4.5 Protocol-Relative Clock Generator

The USCI controller contains a protocol-relative clock generator and it is controlled by register USCI_BRGEN. It is reset when the USCI_BRGEN register is written. The structured of protocol-relative clock generator is shown below.



Note: Refer the **Basic Clock Divider Counter** section to get the f_{SAMP_CLK}

Figure 6.23-8 Protocol-Relative Clock Generator

The protocol related counter contains basic clock divider counter and timing measurement counter. It is based on a divider stages, providing the frequencies needed for the different protocols. It contains:

- The basic clock divider counter provides the protocol relative clock signal and other protocol-related signals (f_{SAMP_CLK} and f_{DS_CLK}).
- The timing measurement counter for time interval measurement, e.g. baud rate detection on UART protocol.
- The output signals of protocol relative clock generator can be made available on pins (e.g USCIx_CLK for SPI).

Basic Clock Divider Counter

The basic clock divider counter is used for an integer division delivering f_{REF_CLK2}, f_{REF_CLK}, f_{DIV_CLK}, f_{SCLK} ,and f_{SAMP_CLK} . The frequencies of this divider are controlled by PTCLKSEL (USCI_BRGEN [1]), CLKDIV (USCI_BRGEN [25:16]), SPCLKSEL (USCI_BRGEN [3:2]).

The basic clock divider counter is used to generate the relative protocol timing signals.

$$f_{\mathrm{DIV_CLK}} = f_{REF_CLK} \times \frac{1}{\mathrm{CLKDIV} + 1} \text{ if } \mathrm{PTCLKSEL} = 0$$

$$f_{\mathrm{DIV_CLK}} = f_{REF_CLK} \times \frac{1}{(\mathrm{CLKDIV} + 1) \times 2} \text{ if } \mathrm{PTCLKSEL} = 1$$

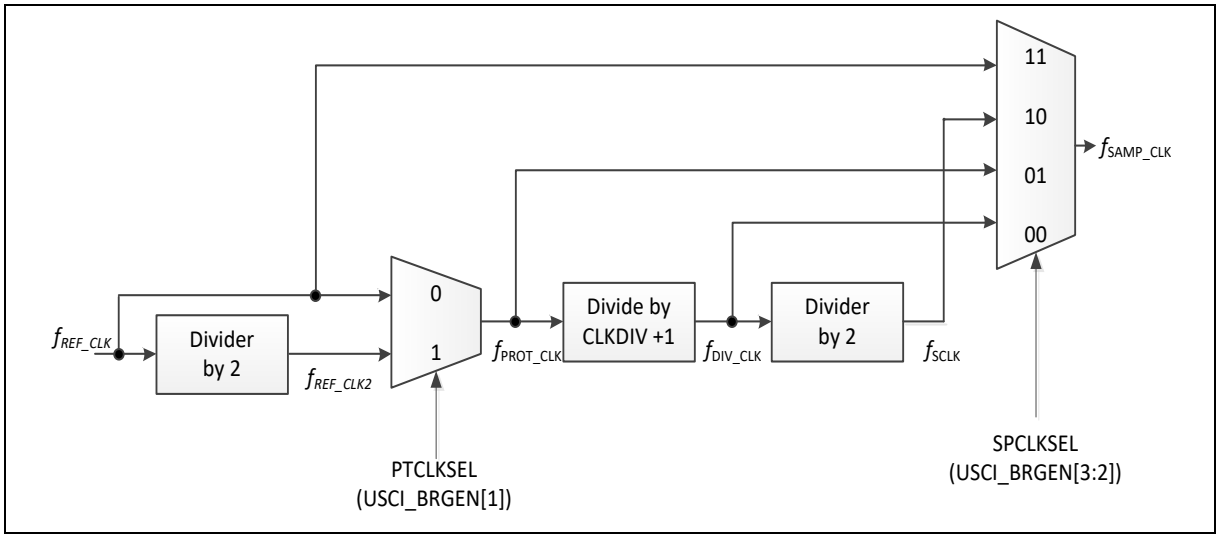


Figure 6.23-9 Basic Clock Divider Counter

Timing Measurement Counter

The timing measurement counter is used for time interval measurement and is enabled by TMCNTEN (USCI_BRGEN [4]) = 1. When TMCNTSRC (USCI_BRGEN [5]) is set to 1, the timer works on f_{DIV_CLK}, otherwise, the timer works independently from f_{PROT_CLK}. Therefore, any serial data reception or transmission can continue while the timer is performing timing measurements. The timer counts the length of protocol-related signals with f_{PROT_CLK} or f_{DIV_CLK}. It stops counting when it reaches the user-specified value.

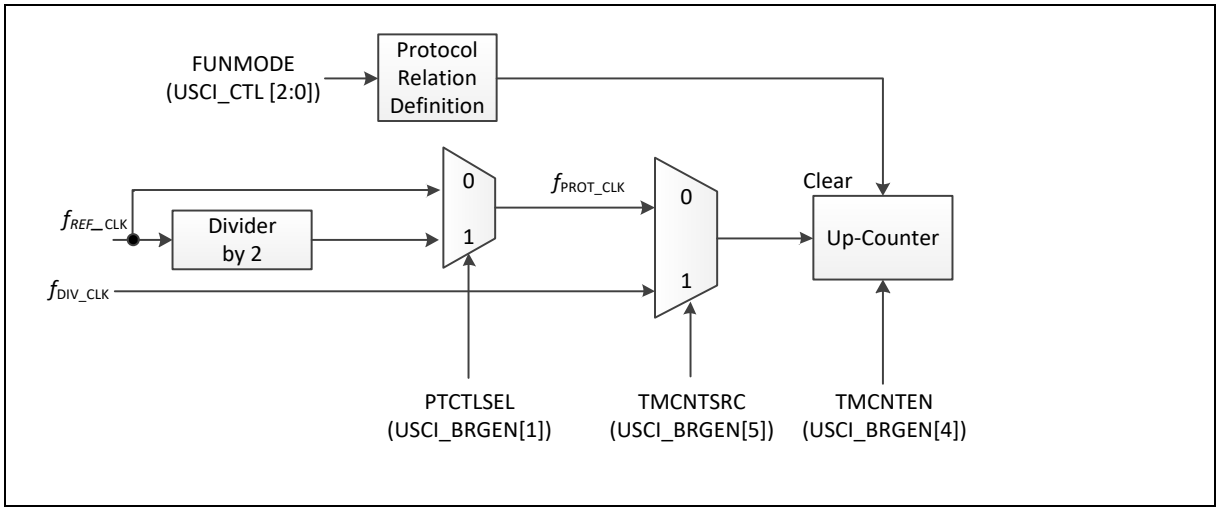


Figure 6.23-10 Block of Timing Measurement Counter

The timing measurement counter is used to perform time-out function or auto-baud rate mechanism. Its functionality depends on the selected protocol as shown below.

- UART: The timing measurement counter is used in auto baud rate detection.
- SPI: The timing measurement counter is used for counting the slave time-out period.
- I^2C: The timing measurement counter indicates time-out clock cycle.

Sample Time Counter

A sample time counter associated to the protocol related counter defining protocol specific timings, such

shift control signals or bit timings, based on the input frequency f_{SAMP_CLK}. The sample time counter allows generating time intervals for protocol-specific purposes. The period of a sample frequency f_{PDS_CNT} is given by the selected input frequency f_{SAMP_CLK} and the programmed pre-divider value (PDSCNT (USCI_BRGEN [9:8])). The meaning of the sample time depends on the selected protocol. Please refer to the corresponding chapters for more protocol-specific information.

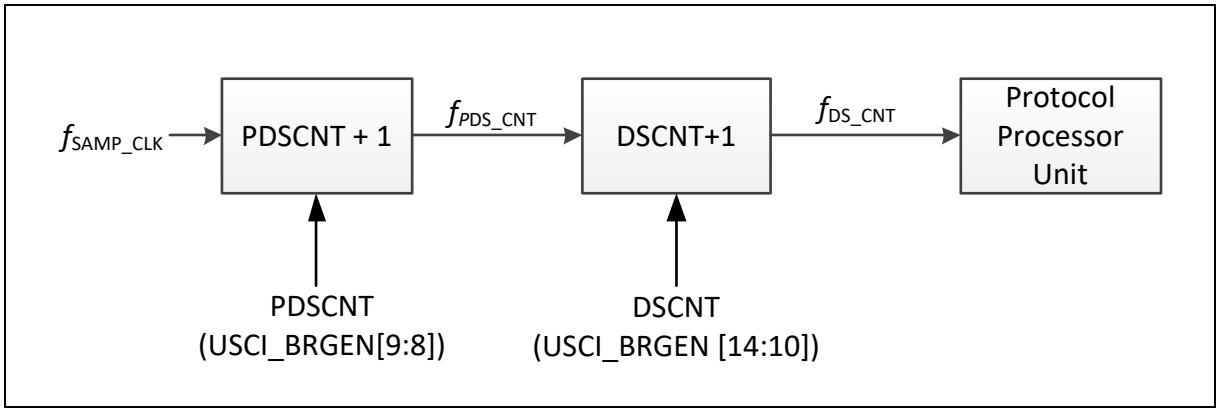


Figure 6.23-11 Sample Time Counter

6.23.4.6 Data Transfer Events and Interrupts

The data transfer events are based on the transmission or reception of a data word. The related indication flags are located in register USCI_PROTSTS. All events can be individually enabled for interrupt generation. If the FUNMODE (USCI_CTL [2:0]) is set to 0, the USCI is disabled. When FUNMODE (USCI_CTL [2:0]) is setting for a protocol port, the internal states will be controlled by logic hardware of the selected protocol.

- Transmit start interrupt event to indicate that a data word has been started:

  A transmit start interrupt event occurs when the data is loaded into transmitted shift register. It is indicated by flag TXSTIF (USCI_PROTSTS [1]) and, if enabled, leads to transmit start interrupt.

- Transmit end interrupt event to indicate that a data word transmission has been done:

  A transmit end interrupt event occurs when the current transmit data in shift register had been finished. It is indicated by flag TXENDIF (USCI_PROTSTS [2]) and, if enabled, leads to transmit end interrupt. This event also indicates when the shift control settings (word length, shift direction, etc.) are internally “frozen” for the current data word transmission. In UART and I^2C mode, the transmit data valid is according to TXEMPTY (USCI_BUFSTS [8]) and protocol relative internal signal with the transmit end interrupt event.

- Receiver start event to indicate that a data word reception has started:

  When the receive clock edge that shifts in the first bit of a new data word is detected and reception is enabled, a receiver start event occurs. It is indicated by flag RXSTIF (USCI_PROTSTS [3]) and, if enabled, leads to receiver start interrupt.

- Receive event to indicate that a data word has been received:

  If a new received word becomes available in the receive buffer, a receive event occurs. It is indicated by flag RXENDIF (USCI_PROTSTS [4]) and, if enabled, leads to receive interrupt.

- Data lost event to indicate a loss of the newest received data word:

  If the data word available in register USCI_RXDAT (oldest data word from RX_BUF0 or RX_BUF1) has not been read out and the receive buffer is FULL, the new incoming data will lose and this event occurs. It is indicated by flag RXOVIF (USCI_BUFSTS[3]) and, if enabled, leads to a protocol interrupt.

The general event and interrupt structure is shown in Figure 6.23-12.

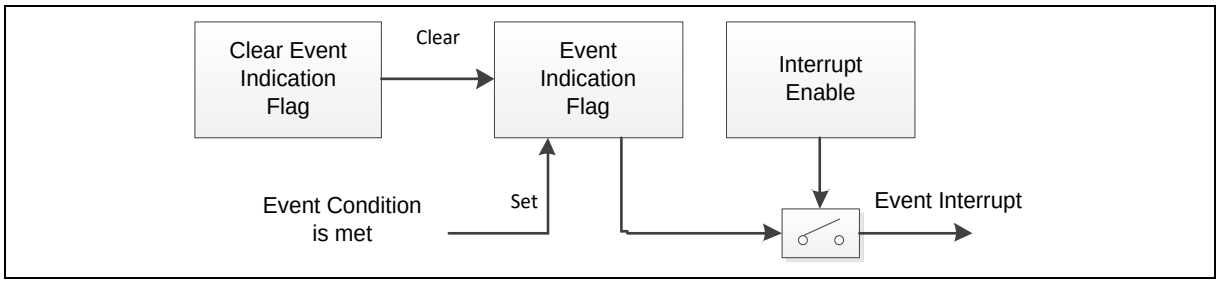


Figure 6.23-12 Event and Interrupt Structure

Each general interrupt enable can set by RXENDIEN, RXSTIEN, TXENDIEN, and TXSTIEN of USCI_INTEN [4:1]. The events are including receive end interrupt event, receive start interrupt event, transmit end interrupt event, and transmit start interrupt event. For protocol-specific interrupt, it is specified in each protocol interrupt enable register.

If a defined condition is met, an event is detected and an event indication flag becomes automatically set. The flag stays set until it is cleared by software. If enabled, an interrupt can be generated if an event is detected.

The registers, bits and bit fields indicate the data transfer events and control the general interrupts of a USCI are shown in Table 6.23-3.

Event	Indication Flag	Indication Cleared By	Interrupt Enabled By
Transmit start interrupt event	TXSTIF (USCI_PROTSTS [1])	It is cleared by software writes 1 to corresponding interrupt bit of USCI_PROTSTS.	TXSTIEN (USCI_INTEN [1])
Transmit end interrupt event	TXENDIF (USCI_PROTSTS [2])		TXENDIEN (USCI_INTEN [2])
Receive start interrupt event	RXSTIF (USCI_PROTSTS [3])		RXSTIEN (USCI_INTEN [3])
Receive end interrupt event	RXENDIF (USCI_PROTSTS [4])		RXENDIEN (USCI_INTEN [4])

Table 6.23-3 Data Transfer Events and Interrupt Handling

6.23.4.7 Protocol-specific Events and Interrupts

These events are related to protocol-specific actions that are described in the corresponding protocol chapters. The related indication flags are located in register USCI_PROTSTS. All events can be individually enabled for the generation of the common protocol interrupt.

Event	Indication Flag	Indication Cleared By	Interrupt Enabled By
Protocol-specific events in UART mode	USCI_PROTSTS [17:16] and USCI_PROTSTS [11:5]	It is cleared by software writes 1 to corresponding interrupt bit of USCI_PROTSTS.	USCI_PROTIEN[2:1]
Protocol-specific events in SPI mode	USCI_PROTSTS [9:8], USCI_PROTSTS [6:5]		USCI_PROTIEN [3:0]
Protocol-specific events in I^2C mode	USCI_PROTSTS [13:8], USCI_PROTSTS [5]		USCI_PROTIEN [6:0]

Table 6.23-4 Protocol-specific Events and Interrupt Handling

6.23.4.8 Wake-up

The protocol-related wake-up functional information is located in the Wake-up Control Register (USCI_WKCTL) and in the Wake-up Status Register (USCI_WKSTS). These registers are shared between the available protocols. As a consequence, the meaning of the bit positions in these registers is different within the protocols.

6.23.4.9 PDMA

The USCI supports PDMA transfer function. When PDMAEN (USCI_PDMACTL [3]) is set to 1, the PDMA function is enabled.

When TXPDMAEN (USCI_PDMACTL [1]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (USCI_PDMACTL [2]) is set to 1, the controller will start the PDMA reception process. USCI will issue request to PDMA controller automatically when there is data in the receive FIFO buffer.

In UART function, the requirement of RXPDMAEN will be cleared and hold if there is any error condition events including frame error, parity error or break detection. The user shall read out the current data and then the requirement of RXPDMAEN will send to the PDMA module in the next data.

6.24 USCI – UART Mode

6.24.1 Overview

The asynchronous serial channel UART covers the reception and the transmission of asynchronous data frames. It performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the controller. The receiver and transmitter are independent, and the transmission and reception can be started separately.

The UART controller also provides auto flow control. There are two conditions to wake up the system.

6.24.2 Features

- Supports one transmit buffer and two receive buffer for data payload
- Supports hardware auto flow control function
- Supports programmable baud-rate generator
- Support 9-bit Data Transfer (Support 9-bit RS-485)
- Baud rate detection possible by built-in capture event of baud rate generator
- Supports PDMA transfer
- Supports Wake-up function (Incoming Data and nCTS Wakeup Only)

6.24.3 Block Diagram

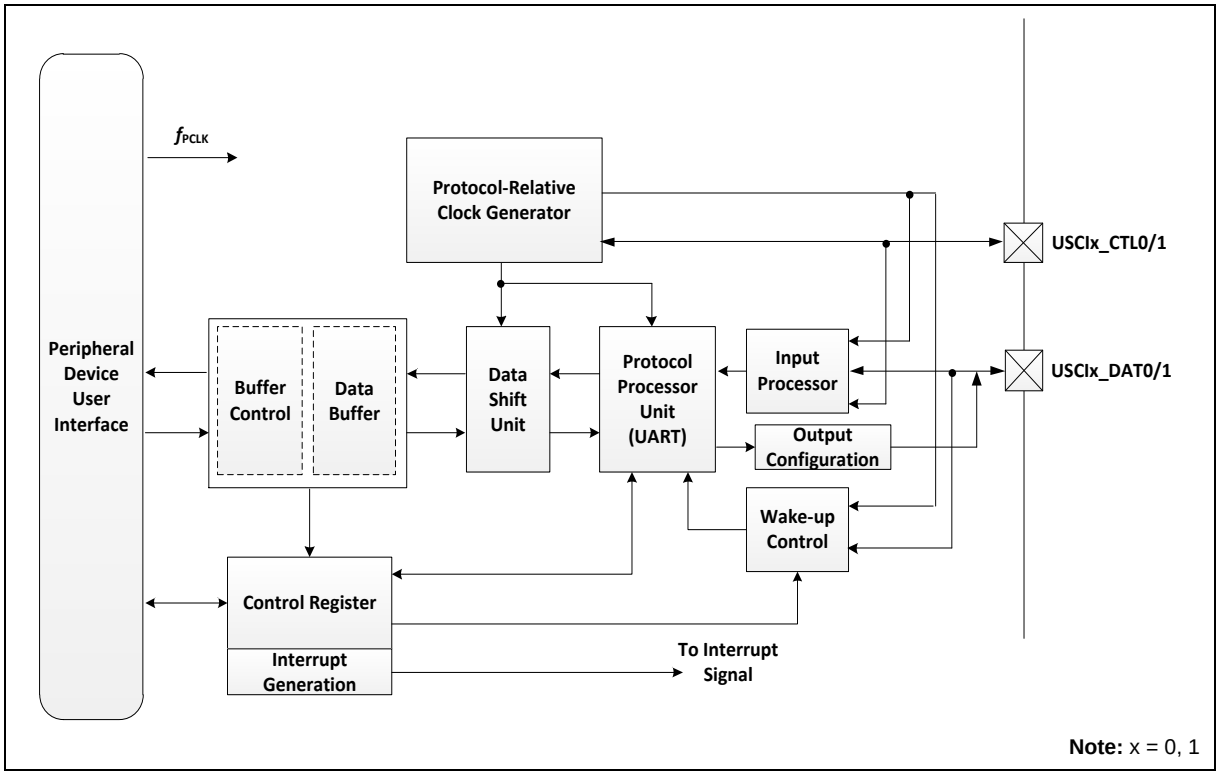


Figure 6.24-1 USCI-UART Mode Block Diagram

6.24.4 Basic Configuration

The basic configurations of USCI0_UART are as follows:

- Clock Source Configuration

- Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK1[8]).
- Enable USCI0_UART function in FUNMODE (UUART_CTL[2:0]=0x2).

- Reset Configuration
  - Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).
-

The basic configurations of USCI1_UART are as follows:

- Clock Source Configuration
  - Enable USCI1 peripheral clock in USCI1CKEN (CLK_APBCLK1[9]).
  - Enable USCI1_UART function in FUNMODE (UUART_CTL[2:0]=0x2).
- Reset Configuration
  - Reset USCI1 controller in USCI1RST (SYS_IPRST2[9]).
-

6.24.5 Functional Description

6.24.5.1 UUART Common Functional Description

Please refer to USCI chapter for detailed information.

6.24.5.2 Signal Description

An UART connection is characterized by the use of a single connection line between a transmitter and a receiver. The receiver input signal (RXD) is handled by the input stage USCIx_DAT0 and the transmit output (TXD) signal is handled by the output stage of USCIx_DAT1.

For full-duplex communication, an independent communication line is needed for each transfer direction. Figure 6.24-2 shows an example with a point-to-point full-duplex connection between two communication partners UART module A and UART module B.

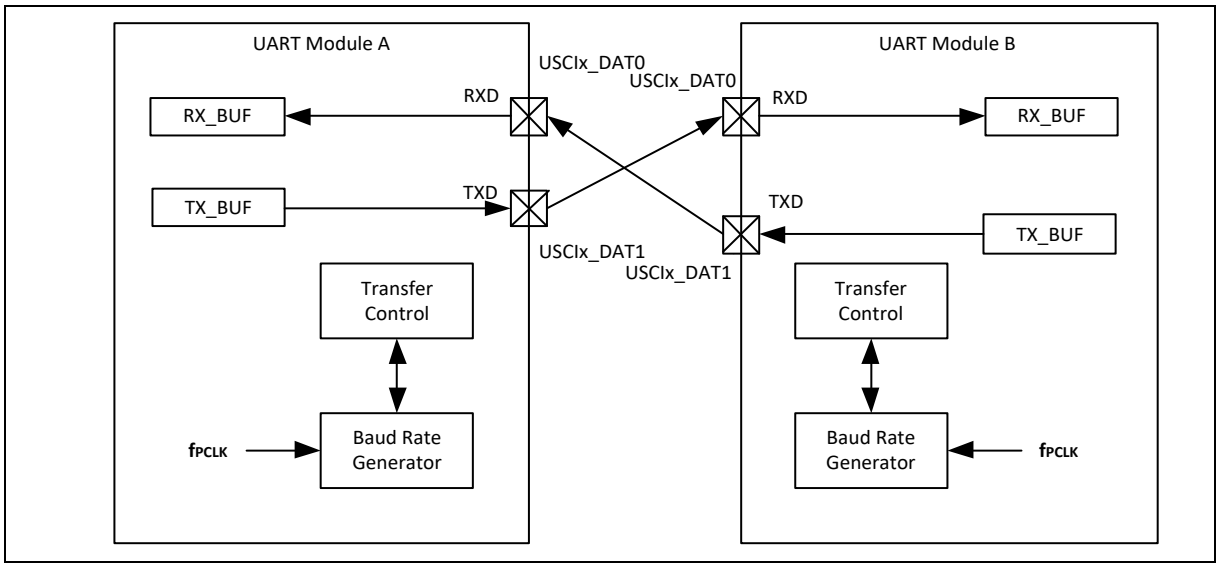


Figure 6.24-2 UART Signal Connection for Full-Duplex Communication

Input Signal

For UART protocol, the number of input signals is shown in Table 6.24-1. Each input signal is handled by an input processor for signal conditioning, such as signal inverse selection control, or a digital input filter. They can be classified according to their meaning for the protocols (see Table 6.24-1).

Selected Protocol		UART
Control Input	USCIx_CTL0	nCTS
	USCIx_CTL1	-
Data Input	USCIx_DAT0	RX
	USCIx_DAT1	-

Table 6.24-1 Input Signals for UART Protocol

Output Signals

For UART protocol, up to each protocol-related output signals are available. The number of actually used outputs depends on the selected protocol. They can be classified according to their meaning for the protocols.

Selected Protocol		UART
Control Output	USCIx_CTL0	-
	USCIx_CTL1	nRTS
Data Output	USCIx_DAT0	-
	USCIx_DAT1	TX

Table 6.24-2 Output Signals for Different Protocol

6.24.5.3 Frame Format

A standard UART frame is shown in Figure 6.24-3. It consists of:

- An idle time with the signal level 1.
- One start of frame bit (SOF) with the signal level 0.
- 6~13 bit data
- A PARITY bit (P), programmable for either even or odd parity. It is optionally possible to handle frames without PARITY bit.
- One or two STOP bits with the signal level 1.

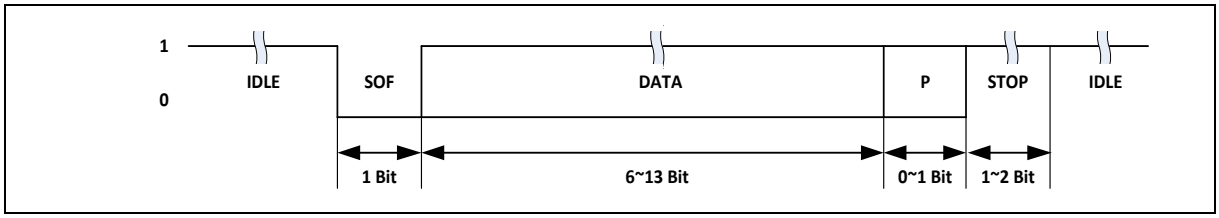


Figure 6.24-3 UART Standard Frame Format

The protocol specific bits (SOF, P, STOP) are automatically handled by the UART protocol state machine and do not appear in the data flow via the receive and transmit buffers.

START bit

The receiver input signal USCIx_DAT0 is checked for a falling edge. An SOF bit is detected when a falling edge occurs while the receiver is idle or after the sampling point of the last STOP bit. To increase noise immunity, the SOF bit timing starts with the first falling edge that is detected. If the sampled bit value of the SOF is 1, the previous falling edge is considered to be due to noise and the receiver is considered to be idle again.

Data Field

The length of the data field (number of data bits) can be programmed by the bit field of DWIDTH (UUART_LINECTL[11:8]). It can vary between 6 to 13 data bits.

Note: In UART protocol, the data transmission order is LSB first by setting LSB (UUART_LINECTL[0]) to 1.

PARITY bit

The UART allows parity generation for transmission and parity check for reception on frame base. The type of parity can be selected by bit field PARITYEN (UUART_PROTCTL[1]) and EVENPARITY (UUART_PROTCTL[2]), common for transmission and reception (no parity, even or odd parity). If the parity handling is disabled, the UART frame does not contain any PARITY bit. For consistency reasons, all communication partners have to be programmed to the same parity mode.

After the last data bit of the data field, the transmitter automatically sends out its calculated PARITY bit if parity generation has been enabled. The receiver interprets this bit as received parity and compares it to its internally calculated one. The result of the parity check and frame check (STOP bit) are monitored in the protocol status registers (UUART_PROTSTS). The register contains bits to monitor a protocol-related status and protocol-related error indication (FRMERR, PARITYERR).

STOP bit

Each UART frame is completed by 1 or 2 of STOP bits with the signal level 1 (same level as the idle level). The number of STOP bits is programmable by bit STOPB (UUART_PROTCTL[0]). A new START bit can be transferred directly after the last STOP bit.

Transfer Status Indication

RXBUSY (UUART_PROTSTS[10]) indicates the receiver status.

The receiver status can be monitored by RXBUSY bit. In this case, bit RXBUSY is set during a complete frame reception from the beginning of the start of frame bit to the end of the last STOP bit.

6.24.5.4 Operating Mode

To operate the UART protocol, the following issues have to be considered:

Select UART Mode

The UART protocol can be selected by setting FUNMODE (UUART_CTL[2:0]) to 0x2 and the UART protocol can be enabled by setting PROTEN (UUART_PROTCTL[31]) to 1. Note that the FUNMODE must be set 0 before protocol changing and it is recommended to configure all parameters of the UART before UART protocol is enabled.

Pin Connections

The USCIx_DAT0 pin is used for UART receive data input signal (RX) in UART protocol. The property of input data signal can be configured in UUART_DATIN0. It is suggested to set EDGEDET (UUART_DATIN0[4:3]) as 10B for START bit detection.

The USCIx_DAT1 pin is used for UART transmit data output signal (TX) in UART protocol. The property of output data signal can be configured in UUART_LINECTL.

The USCIx_CTL0 pin is used for UART clear to send signal (nCTS) in UART protocol. The property of input control signal can be configured in UUART_CTLIN0.

The USCIx_CTL1 pin is used for UART request to send signal (nRTS) in UART protocol. The property of output control signal can be configured in UUART_LINECTL.

Bit Timing Configuration

The desired baud rate setting has to be selected, comprising the baud rate generator and the bit timing.

Frame Format Configuration

The word length, the STOP bit number, and the parity mode has to be set up according to the application requirements by programming UUART_LINECTL and the UUART_PROTCTL register. If required by the application, the data input and output signals can be inverted. The data transmission order is LSB first by setting LSB (UUART_LINECTL[0]) to 1.

6.24.5.5 Bit Timing

In UART mode, each frame bit is divided into data sample time in order to provide granularity in the sub-bit range to adjust the sample point to the application requirements. The number of data sample time per bit is defined by bit fields DSCNT (UUART_BRGEN[14:10]) and the length of a data sample time is given by PDSCNT (UUART_BRGEN[9:8]).

In the example given in Figure 6.24-4, one bit time is composed of 16 data sample time DSCNT(UUART_BRGEN[14:10]) = 15. It is not recommended to program less and equal than 4 data sample time per bit time.

The position of the sampling point for the bit value is fixed in 1/2 samples time. It is possible to sample the bit value to take the average of samples.

The bit timing setup (number of data sample time) is common for the transmitter and the receiver because they use the same hardware circuit.

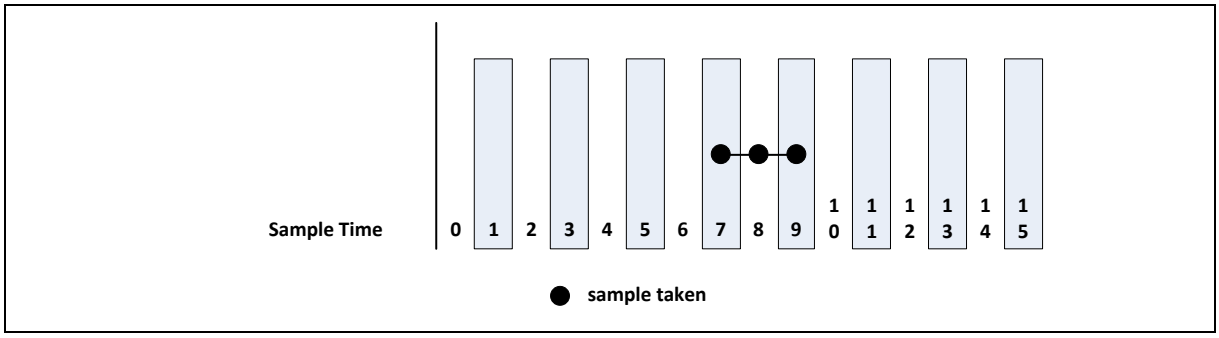


Figure 6.24-4 UART Bit Timing (Data Sample Time)

6.24.5.6 Baud Rate Generation

The baud rate fUART in UART mode depends on the number of data sample time per bit time and their timing. The baud rate setting should only be changed while the transmitter and the receiver are idle. The bits RCLKSEL, SPCLKSEL, PDSCNT, and DSCNT define the baud rate setting:

RCLKSEL (UUART_BRGEN[0])

to define the input frequency f_{REF_CLK}

SPCLKSEL (UUART_BRGEN[3:2])

to define the multiple source of the sample clock f_{SAMP_CLK}

PDSCNT (UUART_BRGEN[9:8])

to define the length of a data sample time (division of f_{REF_CLK} by 1, 2, 3, or 4)

DSCNT (UUART_BRGEN[14:10])

to define the number of data sample time per bit time

The standard setting is given by RCLKSEL = 0 ($f_{REF_CLK} = f_{PCLK}$), PTCLKSEL = 0 ($f_{PROT_CLK} = f_{REF_CLK}$) and SPCLKSEL = 0 ($f_{SAMP_CLK} = f_{DIV_CLK}$). Under these conditions, the baud rate is given by:

$$f_{\mathrm{UART}} = f_{REF_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

In order to generate slower frequencies, additional divide-by-2 stages can be selected by PTCLKSEL =

1 (f_{PROT_CLK} = f_{REF_CLK2}), leading to:

$$f_{\mathrm{UART}} = \frac{f_{REF_CLK}}{2} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

If SPCLKSEL = 2 (f_{SAMP_CLK} = f_{SCLK}), and RCLKSEL = 0 (f_{REF_CLK} = f_{PCLK}), PTCLKSEL = 0 (f_{PROT_CLK} = f_{REF_CLK}). The baud rate is given by:

$$f_{\mathrm{UART}} = f_{REF_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{2} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

There is error tolerance for the UART baud rate after setting the baud rate parameter. Table 6.24-3 lists the relative error percentage examples for users to calculate their relative baud rate setting. The clock source is standard setting (SPCLKSEL=0, PTCLKSEL=0 and RCLKSEL=0).

HCLK Source	PCLK Source	Expect Baud Rate	CLKDIV (UUART_BRGEN[25:16])	DSCNT (UUART_BRGEN[14:10])	PDSCNT	Active Baud Rate	Error Percentage
12 MHz	HCLK	115200	0xC	0x7	0x0	115384	0.16%
12 MHz	HCLK	9600	0x7C	0x9	0x0	9600	0%
12 MHz	HCLK	2400	0x1F3	0x9	0x0	2400	0%

Table 6.24-3 Baud Rate Relationship

6.24.5.7 Auto Baud Rate Detection

The UART controller supports auto baud rate detection function. It is used to identify the input baud rate from the receiver signal (USCIx_DAT0) and then revised the baud rate clock divider CLKDIV (UUART_BRGEN[25:16]) after the baud rate function done to meet the detected baud rate information. According the section of Timing Measurement Counter, the timing measurement counter is used for time interval measurement of the input signal (USCIx_DAT0) and the actual timer value is captured into bit field BRDETITV (UUART_PROTCTL[24:16]) in each falling edge of the detected signal.

When the ABREN (UUART_PROTOCOL[6]) bit is enabled, the 0x55 data patterns is necessary for auto baud rate detection. The falling edge of input signal starts the baud rate counter and it loads the timing measurement counter value into the BRDETITV (UUART_PROTCTL[24:16]) in the next falling edge. It is suggested to use the f_{DIV_CLK} (TMCNTSRC (UUART_BRGENC[5]) =1) as the counter source.

The CLKDIV (UUART_BRGEN[25:16]) will be revised by BRDETITV (UUART_PROTCTL[25:16]) after the auto baud rate function done (the time of 4th falling edge of input signal). If the user want to receive the next successive frame correctly, it is better to set the value of CLKDIV (UUART_BRGEN[25:16]) and DSCNT (UUART_BRGEN[14:10]) as the same value (the value shall be among the rang of 0xF and 0x5 because the DSCNT is used to define the sample counter of each bit and the PDSCNT (UUART_BRGEN[9:8]) is 0x0.

During the auto baud rate detection, the ABRDETIF (UUART_PROTSTS[9]) and the BRDETITV (UUART_PROTCTL[24:16]) will be updated after each falling edge of input signal and the auto baud rate pattern, 0x55, won't be received into the receiver buffer after the frame done. The bit of ABREN will be cleared by hardware after the 4th falling edge of input signal is detected thus the user can read the status of ABREN to know the auto baud rate function is done or not.

If the CLKDIV and DSCNT are not set as the same value in calculation the auto baud rate function, the user shall calculate the proper average baud rate by the value of BRDETITV and CLKDIV after the auto baud rate function done.

If the baud rate of input signal is very slower and the bit time of timing measurement counter can't calculate the correct period of the input bit time, there is a ABERRSTS bit (UUART_PROTSTS[11]) to indicate the error information of the auto baud rate detection. At this time, the user shall revise the value of CLKDIV and require the Host device to send the 0x55 pattern again.

According the limitation of timing measurement counter, the maximum auto baud rate detection is 0x1FE

for BRDETITV. The UART Auto Baud Rate Control is shown in Figure 6.24-5.

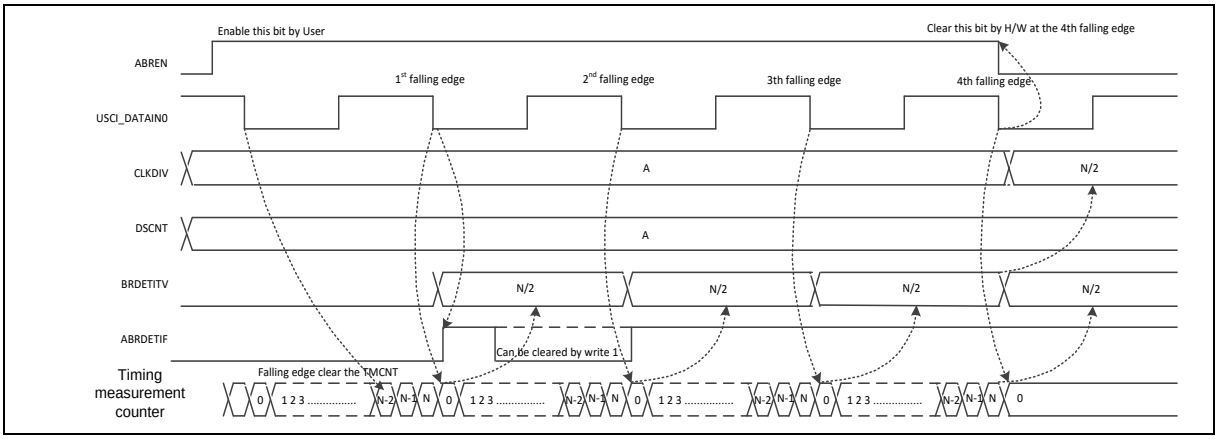


Figure 6.24-5 UART Auto Baud Rate Control

6.24.5.8 Auto Flow Control

The UART supports hardware auto-flow control that provides nRTS flow control by indicator RXFULL (UUART_BUFSTS[1]) on receiver buffer. When the buffer is full (RXFULL = 1), the nRTS is de-asserted.

The UART also provides nCTS flow control on transmitter. The nCTS is used to control the transmitted data is sent out when the nCTS is asserted.

6.24.5.9 RS-485 Support

The UART controller can play the role of the RS-485 master transmitter will identify an address character by setting the parity (9-th bit) to 1. For data characters, the parity is set to 0. Software can use the bit15 of each data to control the PARITY bit (PARITYEN (UUART_PROTCTL[1]) be set) when the STICKEN (UUART_PROTCTL[26]) is set. For example, if the STICKEN is set to 1 and data sequence are 0x8015, 0x8033, 0x0055, 0x0033 and 0x80AA the transmitted parity of data 0x15, 0x33, 0x55 0x33 and 0xAA will be 1, 1, 0, 0 and 1.

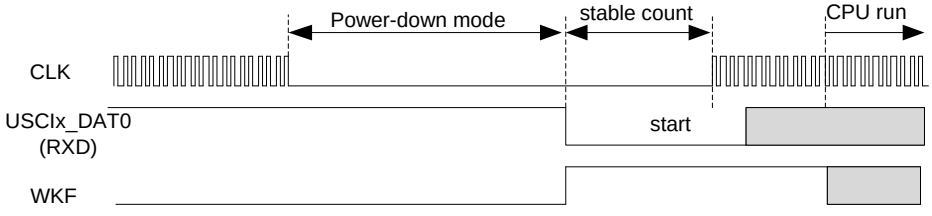
The UART controller can also play as an RS-485 addressable slave, the protocol-related error of PARITYERR (UUART_PROTSTS[5]) can be acted as the address bit detection when the PARITYEN (UUART_PROTCTL[1]), EVENPARITY (UUART_PROTCTL[2]) and STICKEN (UUART_PROTCTL[26]) are set. If the PARITYERR has been set, it means that the address bit in the received bus is detected; otherwise, the data is received into Buffer.

6.24.5.10 Wake-up Function

The USCI Controller in UART mode supports wake-up system function. The wake-up source includes incoming data and nCTS pin. Each wake-up source description is as follows:

(a) Incoming data wake-up

When system is in power-down and both of the WKEN (UUART_WKCTL[0]) and DATWKEN (UUART_PROTCTL[9]) are set, the toggle of incoming data pin can wake-up the system. In order to receive the incoming data after the system wake-up, the WAKECNT (UUART_PROTCTL[14:11]) shall be set. These bits field of WAKECNT (UUART_PROCTL[14:11]) indicate how many clock cycle selected by f_{PDS_CNT} do the controller can get the 1st bit (START bit) when the device is wakeup from Power-down mode. The incoming data wake-up is shown in Figure 6.24-6.



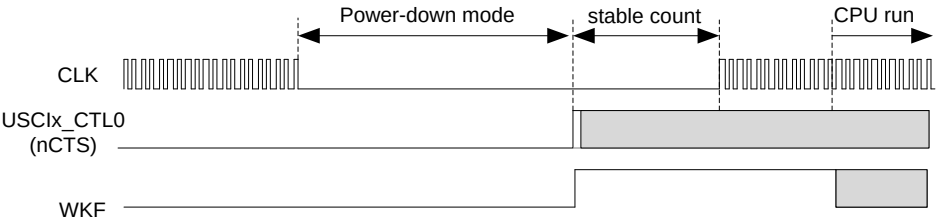
Note: Stable count means HCLK source recovery stable count.

Figure 6.24-6 Incoming Data Wake-Up

(b) nCTS pin wake-up

When system is in power-down and both of the WKEN (UUART_WKCTL[0]) and CTSWKEN (UUART_PROTCTL[10]) are set, the toggle of nCTS pin can wake-up the system. The nCTS wake-up is shown in Figure 6.24-7 and Figure 6.24-8.

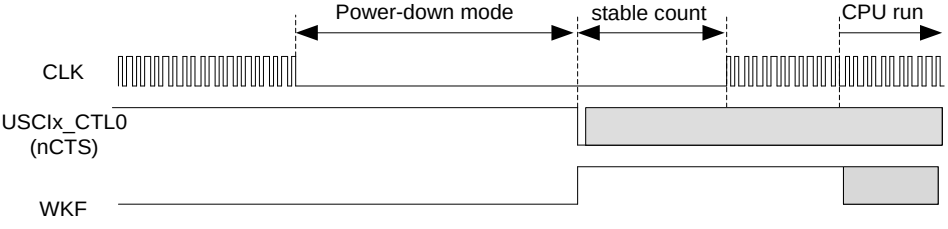
Case 1(nCTS transition from low to high):



Note: Stable count means HCLK source recovery stable count.

Figure 6.24-7 nCTS Wake-Up Case 1

Case 2 (nCTS transition from high to low):



Note 1: Stable count means HCLK source recovery stable count.

Figure 6.24-8 nCTS Wake-Up Case 2

6.24.5.11 Interrupt Events

The UART provided interrupt for protocol event and data transfer event. The description show below:

Protocol Interrupt Events

The following protocol-related events are generated in UART mode and can lead to a protocol interrupt.

Please note that the bits in register UUART_PROTSTS are not automatically cleared by hardware and have to be cleared by software in order to monitor new incoming events.

Receiver Line Status

The protocol-related error FRMERR (UUART_PROTSTS[6]) or PARITYERR (UUART_PROTSTS[5]) are two flags that are assigned to each received data word in the corresponding receiver buffer status registers.

In UART mode, the result of the parity check by the protocol-related error indication (0 = received PARITY bit equal to calculated parity value), and the result of frame check by the protocol-related error indication (0 = received STOP bit equal to the format value '1'). This information is elaborated for each data frame.

The break error flag BREAK (UUART_PROTSTS[7]) is assigned when the receive data is 0, the received parity and the STOP bit are also 0.

The interrupt indicates that there are parity error, frame error or the break data detection in the BREAK, FRMERR, PARITYERR (UUART_PROTSTS[7:5]) bits.

Auto Baud Rate Detection

The auto baud rate interrupt, ABRDETIF (UUART_PROTSTS[9]), indicates that the timing measurement counter has getting 2-bit duration for auto baud rate capture function.

The auto baud rate detection function will be enabled in the first falling edge of receiver signal. The auto baud rate detection function is measurement after the next following falling is detected and it is finished when the frame transfer done. After the transfer done, the timing measurement counter value divided by twice is equal to the number of sample time per bit. The user can read the value of BRDETITV (UUART_PROTCTL[24:16]) and write into the baud rate generator register CLKDIV (UUART_BRGEN[25:16]).

Data Transfer Interrupt Handling

The data transfer interrupts indicate events related to UART frame handling.

Transmit Start Interrupt

Bit TXSTIF (UUART_PROTSTS[1]) is set after the START bit of a data word. In buffer mode, this is the earliest point in time when a new data word can be written to UUART_TXDAT.

Transmitter Finished

This interrupt indicates that the transmitter has completely finished all data in the buffer. Bit TXENDIF (UUART_PROTSTS[2]) becomes set at the end of the last STOP bit.

Receiver Starts Interrupt

Bit RXSTIF (UUART_PROTSTS[3]) is set after the sample point of the START bit.

Receiver Frame Finished

This interrupt indicates that the receiver has completely finished a frame. Bit RXENDIF (UUART_PROTSTS[4]) becomes set at the end of the last receive bit.

6.24.5.12 Programming Example

The following steps are used to configure the UART protocol setting and the data transmission.

1. Set FUNMODE (UUART_CTL[2:0]) to 0x2 to select UART protocol.
2. Write baud rate generator register UUART_BRGEN to select desired baud rate.
   - Set SPCLKSEL (UUART_BRGEN[3:2]), PTCLKSEL (UUART_BRGEN[1]) and RCLKSEL (UUART_BRGEN[0]) to select the clock source.
   - Configure CLKDIV (UUART_BRGEN[25:16]), DSCNT (UUART_BRGEN[14:10]) and PDSCNT (UUART_BRGEN[9:8]) to determine the baud rate divider.
3. Write line control register UUART_LINECTL and protocol control register UUART_PROTCTL

to configure the transmission data format and UART protocol setting.

- Program data field length in DWIDTH (UUART_LINECTL[11:8]).
- Enable PARITY bit and determine the PARITY bit type by setting EVENPARITY (UUART_PROTCTL[2]) and PARITYEN (UUART_PROTCTL[1]).
- Configure STOP bit length by setting STOPB (UUART_PROTCTL[0]).
- Enable LSB (UUART_LINECTL[0]) to select LSB first transmission for UART protocol.
- Set EDGEDET (UUART_DATIN0[4:3]) to “10” to select the detected edge as falling edge for receiver START bit detection.

4. Set PROTEN (UUART_PROTCTL[31]) to 1 to enable UART protocol.
5. Transmit and receive data.
   - Write transmit data register UUART_TXDAT to transmit data.
   - Wait until TXSTIF(UUART_PROTSTS[1]) is set and then user can write the next data in UUART_TXDAT.
   - When TXENDIF(UUART_PROTSTS[2]) is set, the transmit buffer is empty and the STOP bit of the last data has been transmitted.
   - If RXENDIF(UUART_PROTSTS[4]) is set, the receiver has finished a data frame completely. User can get the data by reading receive data register UUART_RXDAT.

6.24.6 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
USCI_UART Base Address: **UUARTn_BA = 0x400D_0000 + (0x1000 * n)** **n= 0, 1**				
UUARTn_CTL	UUARTn_BA+0x00	R/W	USCI Control Register	0x0000_0000
UUARTn_INTEN	UUARTn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000
UUARTn_BRGEN	UUARTn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
UUARTn_DATIN0	UUARTn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000
UUARTn_CTLIN0	UUARTn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000
UUARTn_CLKIN	UUARTn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000
UUARTn_LINECTL	UUARTn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
UUARTn_TXDAT	UUARTn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
UUARTn_RXDAT	UUARTn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
UUARTn_BUFCTL	UUARTn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000
UUARTn_BUFSTS	UUARTn_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101
UUARTn_PDMACTL	UUARTn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000
UUARTn_WKCTL	UUARTn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
UUARTn_WKSTS	UUARTn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
UUARTn_PROTCTL	UUARTn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000
UUARTn_PROTIEN	UUARTn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
UUARTn_PROTSTS	UUARTn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

6.24.7 Register Description

USCI Control Register (UUART_CTL)

Register	Offset	R/W	Description	Reset Value
UUARTn_CTL	UUARTn_BA+0x00	R/W	USCI Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**FUNMODE**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 000 to FUNMODE. 000 = The USCI is disabled. All protocol related state machines are set to idle state. 001 = The SPI protocol is selected. 010 = The UART protocol is selected. 100 = The I^2C protocol is selected. Others = Reserved.

USCI Interrupt Enable Register (UUART_INTEN)

Register	Offset	R/W	Description	Reset Value
UUARTn_INTEN	UUARTn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**RXENDIEN**	**RXSTIEN**	**TXENDIEN**	**TXSTIEN**	**Reserved**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**RXENDIEN**	**Receive End Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive finish event. 0 = The receive end interrupt Disabled. 1 = The receive end interrupt Enabled.
[3]	**RXSTIEN**	**Receive Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive start event. 0 = The receive start interrupt Disabled. 1 = The receive start interrupt Enabled.
[2]	**TXENDIEN**	**Transmit End Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit finish event. 0 = The transmit finish interrupt Disabled. 1 = The transmit finish interrupt Enabled.
[1]	**TXSTIEN**	**Transmit Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit start event. 0 = The transmit start interrupt Disabled. 1 = The transmit start interrupt Enabled.
[0]	**Reserved**	Reserved.

USCI Baud Rate Generator Register (UUART_BRGEN)

Register	Offset	R/W	Description	Reset Value
UUARTn_BRGEN	UUARTn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						CLKDIV	
23	22	21	20	19	18	17	16
CLKDIV							
15	14	13	12	11	10	9	8
Reserved	DSCNT					PDSCNT	
7	6	5	4	3	2	1	0
Reserved		TMCNTSRC	TMCNTEN	SPCLKSEL		PTCLKSEL	RCLKSEL

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} ($f_{DIV_CLK} = f_{PROT_CLK} / (CLKDIV+1)$). **Note:** In UART function, it can be updated by hardware in the 4th falling edge of the input data 0x55 when the auto baud rate function (ABREN(UUART_PROTCTL[6])) is enabled. The revised value is the average bit time between bit 5 and bit 6. The user can use revised CLKDIV and new BRDETITV (UUART_PROTCTL[24:16]) to calculate the precise baud rate.
[15]	**Reserved**	Reserved.
[14:10]	**DSCNT**	**Denominator for Sample Counter** This bit field defines the divide ratio of the sample clock f_{SAMP_CLK}. The divided frequency $f_{DS_CNT} = f_{PDS_CNT} / (DSCNT+1)$. **Note:** The maximum value of DSCNT is 0xF on UART mode and suggest to set over 4 to confirm the receiver data is sampled in right value.
[9:8]	**PDSCNT**	**Pre-divider for Sample Counter** This bit field defines the divide ratio of the clock division from sample clock f_{SAMP_CLK}. The divided frequency $f_{PDS_CNT} = f_{SAMP_CLK} / (PDSCNT+1)$.
[7:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Timing Measurement Counter Clock Source Selection** 0 = Timing measurement counter with f_{PROT_CLK}. 1 = Timing measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Timing Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Timing measurement counter is Disabled. 1 = Timing measurement counter is Enabled.
[3:2]	**SPCLKSEL**	**Sample Clock Source Selection** This bit field used for the clock source selection of a sample clock (f_{SAMP_CLK}) for the

		protocol processor. 00 = f_{SAMP_CLK} is selected to f_{DIV_CLK}. 01 = f_{SAMP_CLK} is selected to f_{PROT_CLK}. 10 = f_{SAMP_CLK} is selected to f_{SCLK}. 11 = f_{SAMP_CLK} is selected to f_{REF_CLK}.
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source signal of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source signal of reference clock (f_{REF_CLK}). 0 = Peripheral device clock f_{PCLK}. 1 = Reserved.

USCI Input Data Signal Configuration (UUART_DATIN0)

Register	Offset	R/W	Description	Reset Value
UUARTn_DATIN0	UUARTn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**EDGEDET**		**ININV**	**Reserved**	**SYNCSEL**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:3]	**EDGEDET**	**Input Signal Edge Detection Mode** This bit field selects which edge actives the trigger event of input data signal. 00 = The trigger event activation is disabled. 01 = A rising edge activates the trigger event of input data signal. 10 = A falling edge activates the trigger event of input data signal. 11 = Both edges activate the trigger event of input data signal. **Note:** In UART function mode, it is suggested to set this bit field as 0x2.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Signal Synchronization Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Input Control Signal Configuration (UUART_CTLIN0)

Register	Offset	R/W	Description	Reset Value
UUARTn_CTLIN0	UUARTn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ININV**	**Reserved**	**SYNCSEL**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Input Clock Signal Configuration (UUART_CLKIN)

Register	Offset	R/W	Description	Reset Value
UUARTn_CLKIN	UUARTn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SYNCSEL**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Line Control Register (UUART_LINECTL)

Register	Offset	R/W	Description	Reset Value
UUARTn_LINECTL	UUARTn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**DWIDTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTLOINV	**Reserved**	**DATOINV**	**Reserved**				**LSB**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0000 = The data word contains 16 bits located at bit positions [15:0]. 0001 = Reserved. 0010 = Reserved. 0011 = Reserved. 0100 = The data word contains 4 bits located at bit positions [3:0]. 0101 = The data word contains 5 bits located at bit positions [4:0]. 0110 = The data word contains 6 bits located at bit positions [5:0]. 0111 = The data word contains 7 bits located at bit positions [6:0]. 1000 = The data word contains 8 bits located at bit positions [7:0]. 1001 = The data word contains 9 bits located at bit positions [8:0]. 1010 = The data word contains 10 bits located at bit positions [9:0]. 1011 = The data word contains 11 bits located at bit positions [10:0]. 1100 = The data word contains 12 bits located at bit positions [11:0]. 1101 = The data word contains 13 bits located at bit positions [12:0]. 1110 = The data word contains 14 bits located at bit positions [13:0]. 1111 = The data word contains 15 bits located at bit positions [14:0]. **Note:** In UART protocol, the length can be configured as 6~13 bits.
[7]	**CTLOINV**	**Control Signal Output Inverse Selection** This bit defines the relation between the internal control signal and the output control signal. 0 = No effect. 1 = The control signal will be inverted before its output. **Note:** In UART protocol, the control signal means nRTS signal.
[6]	**Reserved**	Reserved.

[5]	**DATOINV**	**Data Output Inverse Selection** This bit defines the relation between the internal shift data value and the output data signal of USCIx_DAT1 pin. 0 = The value of USCIx_DAT1 is equal to the data shift register. 1 = The value of USCIx_DAT1 is the inversion of data shift register.
[4:1]	**Reserved**	Reserved.
[0]	**LSB**	**LSB First Transmission Selection** 0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (UUART_TXDAT)

Register	Offset	R/W	Description	Reset Value
UUARTn_TXDAT	UUARTn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission.

USCI Receive Data Register (UUART_RXDAT)

Register	Offset	R/W	Description	Reset Value
UUARTn_RXDAT	UUARTn_BA+0x34	R	USCI Receive Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer. **Note:** RXDAT[15:13] indicate the same frame status of BREAK, FRMERR and PARITYERR (UUART_PROTSTS[7:5]).

USCI Transmitter/Receive Buffer Control Register (UUART_BUFCTL)

Register	Offset	R/W	Description	Reset Value
UUARTn_BUFCTL	UUARTn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**RXRST**	**TXRST**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXCLR	**RXOVIEN**	**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXCLR	**Reserved**						

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset the receive-related counters, state machine, and the content of receive shift register and data buffer. **Note 1:** It is cleared automatically after one PCLK cycle. **Note 2:** It is suggested to check the RXBUSY (UUART_PROTSTS[10]) before this bit will be set to 1.
[16]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset the transmit-related counters, state machine, and the content of transmit shift register and data buffer**.** **Note:** It is cleared automatically after one PCLK cycle.
[15]	**RXCLR**	**Clear Receive Buffer** 0 = No effect. 1 = The receive buffer is cleared (filling level is cleared and output pointer is set to input pointer value). Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[14]	**RXOVIEN**	**Receive Buffer Overrun Error Interrupt Enable Bit** 0 = Receive overrun interrupt Disabled. 1 = Receive overrun interrupt Enabled.
[13:8]	**Reserved**	Reserved.
[7]	**TXCLR**	**Clear Transmit Buffer** 0 = No effect. 1 = The transmit buffer is cleared (filling level is cleared and output pointer is set to input pointer value). Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[6:0]	**Reserved**	Reserved.

USCI Transmit/Receive Buffer Status Register (UUART_BUFSTS)

Register	Offset	R/W	Description	Reset Value
UUARTn_BUFSTS	UUARTn_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**TXFULL**	**TXEMPTY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**RXOVIF**	**Reserved**	**RXFULL**	**RXEMPTY**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**TXFULL**	**Transmit Buffer Full Indicator** 0 = Transmit buffer is not full. 1 = Transmit buffer is full.
[8]	**TXEMPTY**	**Transmit Buffer Empty Indicator** 0 = Transmit buffer is not empty. 1 = Transmit buffer is empty.
[7:4]	**Reserved**	Reserved.
[3]	**RXOVIF**	**Receive Buffer Over-run Error Interrupt Status** This bit indicates that a receive buffer overrun error event has been detected. If RXOVIEN (UUART_BUFCTL[14]) is enabled, the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit. 0 = A receive buffer overrun error event has not been detected. 1 = A receive buffer overrun error event has been detected.
[2]	**Reserved**	Reserved.
[1]	**RXFULL**	**Receive Buffer Full Indicator** 0 = Receive buffer is not full. 1 = Receive buffer is full.
[0]	**RXEMPTY**	**Receive Buffer Empty Indicator** 0 = Receive buffer is not empty. 1 = Receive buffer is empty.

USCI PDMA Control Register (UUART_PDMACTL)

Register	Offset	R/W	Description	Reset Value
UUARTn_PDMACTL	UUARTn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PDMAEN**	**RXPDMAEN**	**TXPDMAEN**	**PDMARST**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PDMAEN**	**PDMA Mode Enable Bit** 0 = PDMA function Disabled. 1 = PDMA function Enabled.
[2]	**RXPDMAEN**	**PDMA Receive Channel Available** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[1]	**TXPDMAEN**	**PDMA Transmit Channel Available** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled.
[0]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the USCI's PDMA control logic. This bit will be cleared to 0 automatically.

USCI Wake-up Control Register (UUART_WKCTL)

Register	Offset	R/W	Description	Reset Value
UUARTn_WKCTL	UUARTn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**PDBOPT**	**Reserved**	**WKEN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDBOPT**	**Power Down Blocking Option** 0 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, MCU will stop the transfer and enter Power-down mode immediately. 1 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, the on-going transfer will not be stopped and MCU will enter idle mode immediately.
[1]	**Reserved**	Reserved.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (UUART_WKSTS)

Register	Offset	R/W	Description	Reset Value
UUARTn_WKSTS	UUARTn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**WKF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – UART (UUART_PROTCTL)

Register	Offset	R/W	Description	Reset Value
UUARTn_PROTCTL	UUARTn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	DGE	BCEN	Reserved		STICKEN	Reserved	BRDETITV
23	22	21	20	19	18	17	16
BRDETITV							
15	14	13	12	11	10	9	8
Reserved	WAKECNT				CTSWKEN	DATWKEN	Reserved
7	6	5	4	3	2	1	0
Reserved	ABREN	RTSAUDIREN	CTSAUTOEN	RTSAUTOEN	EVENPARITY	PARITYEN	STOPB

Bits	Description	
[31]	**PROTEN**	**UART Protocol Enable Bit** 0 = UART Protocol Disabled. 1 = UART Protocol Enabled.
[30]	**DGE**	**Deglitch Enable Bit** 0 = Deglitch Disabled. 1 = Deglitch Enabled. **Note 1:** When this bit is set to logic 1, any pulse width less than about 150 ns will be considered a glitch and will be removed in the serial data input (RX). This bit acts only on RX line and has no effect on the transmitter logic. **Note 2:** It is recommended to set this bit only when operating at baud rate under 2.5 Mbps.
[29]	**BCEN**	**Transmit Break Control Enable Bit** 0 = Transmit Break Control Disabled. 1 = Transmit Break Control Enabled. **Note:** When this bit is set to logic 1, the serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX line and has no effect on the transmitter logic.
[28:27]	**Reserved**	Reserved.
[26]	**STICKEN**	**Stick Parity Enable Bit** 0 = Stick parity Disabled. 1 = Stick parity Enabled. **Note:** Refer to RS-485 Support section for detailed information.
[25]	**Reserved**	Reserved.
[24:16]	**BRDETITV**	**Baud Rate Detection Interval** This bit fields indicate how many clock cycle selected by TMCNTSRC (UUART_BRGEN[5]) does the slave calculates the baud rate in one bits. The order of the bus shall be 1 and 0 step by step (e.g. the input data pattern shall be 0x55). The user can read the value to know the current input baud rate of the bus whenever the ABRDETIF (UUART_PROTCTL[9]) is set. **Note:** This bit can be cleared to 0 by software writing '0' to the **BRDETITV**.

[15]	**Reserved**	Reserved.
[14:11]	**WAKECNT**	**Wake-up Counter** These bits field indicate how many clock cycle selected by f_{PDS_CNT} do the slave can get the 1st bit (START bit) when the device is wake-up from Power-down mode.
[10]	**CTSWKEN**	**nCTS Wake-up Mode Enable Bit** 0 = nCTS wake-up mode Disabled. 1 = nCTS wake-up mode Enabled.
[9]	**DATWKEN**	**Data Wake-up Mode Enable Bit** 0 = Data wake-up mode Disabled. 1 = Data wake-up mode Enabled.
[8:7]	**Reserved**	Reserved.
[6]	**ABREN**	**Auto-baud Rate Detect Enable Bit** 0 = Auto-baud rate detect function Disabled. 1 = Auto-baud rate detect function Enabled. **Note:** When the auto - baud rate detect operation finishes, hardware will clear this bit. The associated interrupt ABRDETIF (UUART_PROTST[9]) will be generated (If ARBIEN (UUART_PROTIEN[1]) is enabled).
[5]	**RTSAUDIREN**	**nRTS Auto Direction Enable Bit** When nRTS auto direction is enabled, if the transmitted bytes in the TX buffer is empty, the UART asserted nRTS signal automatically. 0 = nRTS auto direction control Disabled. 1 = nRTS auto direction control Enabled. **Note 1:** This bit is used for nRTS auto direction control for RS485. **Note 2:** This bit has effect only when the RTSAUTOEN is not set.
[4]	**CTSAUTOEN**	**nCTS Auto-flow Control Enable Bit** When nCTS auto-flow is enabled, the UART will send data to external device when nCTS input assert (UART will not send data to device if nCTS input is dis-asserted). 0 = nCTS auto-flow control Disabled. 1 = nCTS auto-flow control Enabled.
[3]	**RTSAUTOEN**	**nRTS Auto-flow Control Enable Bit** When nRTS auto-flow is enabled, if the receiver buffer is full (RXFULL (UUART_BUFSTS[1] =1), the UART will de-assert nRTS signal. 0 = nRTS auto-flow control Disabled. 1 = nRTS auto-flow control Enabled. **Note:** This bit has effect only when the RTSAUDIREN is not set.
[2]	**EVENPARITY**	**Even Parity Enable Bit** 0 = Odd number of logic 1's is transmitted and checked in each word. 1 = Even number of logic 1's is transmitted and checked in each word. **Note:** This bit has effect only when PARITYEN is set.
[1]	**PARITYEN**	**Parity Enable Bit** This bit defines the PARITY bit is enabled in an UART frame. 0 = The PARITY bit Disabled. 1 = The PARITY bit Enabled.
[0]	**STOPB**	**STOP bits** This bit defines the number of STOP bits in an UART frame. 0 = The number of STOP bits is 1.

		1 = The number of STOP bits is 2.

USCI Protocol Interrupt Enable Register – UART (UUART_PROTIEN)

Register	Offset	R/W	Description	Reset Value
UUARTn_PROTIEN	UUARTn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**RLSIEN**	**ABRIEN**	**Reserved**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable Bit** 0 = Receive line status interrupt Disabled. 1 = Receive line status interrupt Enabled. **Note:** UUART_PROTSTS[7:5] indicates the current interrupt event for receive line status interrupt.
[1]	**ABRIEN**	**Auto-baud Rate Interrupt Enable Bit** 0 = Auto-baud rate interrupt Disabled. 1 = Auto-baud rate interrupt Enabled.
[0]	**Reserved**	Reserved.

USCI Protocol Status Register – UART (UUART_PROTSTS)

Register	Offset	R/W	Description	Reset Value
UUARTn_PROTSTS	UUARTn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**CTSLV**	**CTSSYNCLV**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ABERRSTS**	**RXBUSY**	**ABRDETIF**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BREAK	**FRMERR**	**PARITYERR**	**RXENDIF**	**RXSTIF**	**TXENDIF**	**TXSTIF**	**Reserved**

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**CTSLV**	**nCTS Pin Status (Read Only)** This bit used to monitor the current status of nCTS pin input. 0 = nCTS pin input is low level voltage logic state. 1 = nCTS pin input is high level voltage logic state.
[16]	**CTSSYNCLV**	**nCTS Synchronized Level Status (Read Only)** This bit used to indicate the current status of the internal synchronized nCTS signal. 0 = The internal synchronized nCTS is low. 1 = The internal synchronized nCTS is high.
[15:12]	**Reserved**	Reserved.
[11]	**ABERRSTS**	**Auto-baud Rate Error Status** This bit is set when auto-baud rate detection counter overrun. When the auto-baud rate counter overrun, the user shall revise the CLKDIV (UUART_BRGEN[25:16]) value and enable ABREN (UUART_PROTCTL[6]) to detect the correct baud rate again. 0 = Auto-baud rate detect counter is not overrun. 1 = Auto-baud rate detect counter is overrun. **Note 1:** This bit is set at the same time of ABRDETIF. **Note 2:** This bit can be cleared by writing “1” to ABRDETIF or ABERRSTS.
[10]	**RXBUSY**	**RX Bus Status Flag (Read Only)** This bit indicates the busy status of the receiver. 0 = The receiver is Idle. 1 = The receiver is BUSY.
[9]	**ABRDETIF**	**Auto-baud Rate Interrupt Flag** This bit is set when auto-baud rate detection is done among the falling edge of the input data. If the ABRIEN (UUART_PROTCTL[6]) is set, the auto-baud rate interrupt will be

		generated. This bit can be set 4 times when the input data pattern is 0x55 and it is cleared before the next falling edge of the input bus. 0 = Auto-baud rate detect function is not done. 1 = One Bit auto-baud rate detect function is done. **Note:** This bit can be cleared by writing “1” to it.
[8]	**Reserved**	Reserved.
[7]	**BREAK**	**Break Flag** This bit is set to logic 1 whenever the received data input (RX) is held in the “spacing state” (logic 0) for longer than a full word transmission time (that is, the total time of “START bit” + data bits + parity + STOP bits). 0 = No Break is generated. 1 = Break is generated in the receiver bus. **Note:** This bit can be cleared by writing “1” among the BREAK, FRMERR and PARITYERR bits.
[6]	**FRMERR**	**Framing Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “STOP bit” (that is, the STOP bit following the last data bit or PARITY bit is detected as logic 0). 0 = No framing error is generated. 1 = Framing error is generated. **Note:** This bit can be cleared by writing “1” among the BREAK, FRMERR and PARITYERR bits.
[5]	**PARITYERR**	**Parity Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “PARITY bit”. 0 = No parity error is generated. 1 = Parity error is generated. **Note:** This bit can be cleared by writing “1” among the BREAK, FRMERR and PARITYERR bits.
[4]	**RXENDIF**	**Receive End Interrupt Flag** 0 = A receive finish interrupt status has not occurred. 1 = A receive finish interrupt status has occurred. **Note:** It is cleared by software writing 1 into this bit.
[3]	**RXSTIF**	**Receive Start Interrupt Flag** 0 = A receive start interrupt status has not occurred. 1 = A receive start interrupt status has occurred. **Note:** It is cleared by software writing 1 into this bit.
[2]	**TXENDIF**	**Transmit End Interrupt Flag** 0 = A transmit end interrupt status has not occurred. 1 = A transmit end interrupt status has occurred. **Note:** It is cleared by software writing 1 into this bit.
[1]	**TXSTIF**	**Transmit Start Interrupt Flag** 0 = A transmit start interrupt status has not occurred. 1 = A transmit start interrupt status has occurred. **Note 1:** It is cleared by software writing one into this bit. **Note 2:** Used for user to load next transmit data when there is no data in transmit buffer.
[0]	**Reserved**	Reserved.

6.25 USCI - SPI Mode

6.25.1 Overview

The SPI protocol of USCI controller applies to synchronous serial data communication and allows full duplex transfer. It supports both master and Slave operation mode with the 4-wire bi-direction interface. SPI mode of USCI controller performs a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. The SPI mode is selected by FUNMODE (USPI_CTL[2:0]) = 0x1

This SPI protocol can operate as master or Slave mode by setting the SLAVE (USPI_PROTCTL[0]) to communicate with the off-chip SPI Slave or master device. The application block diagrams in master and Slave mode are shown below.

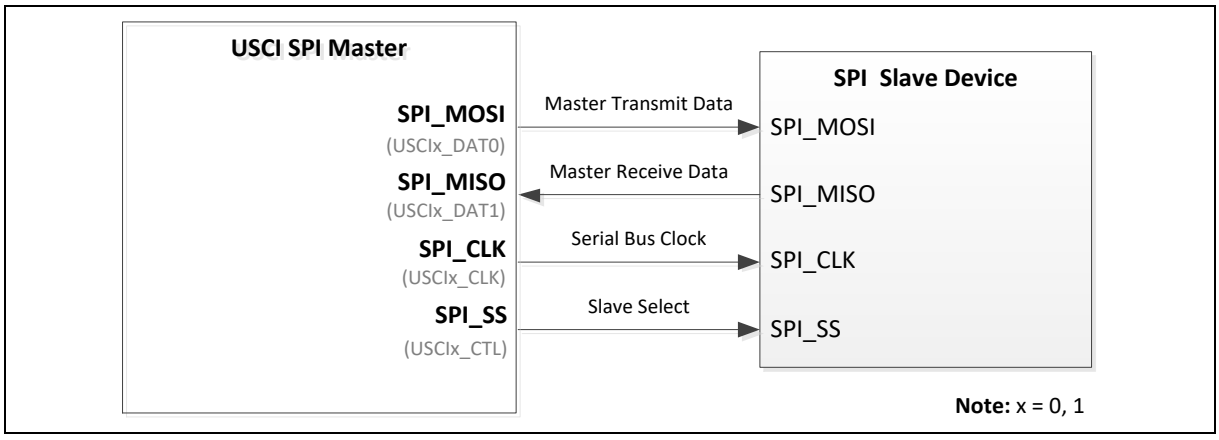


Figure 6.25-1 SPI Master Mode Application Block Diagram

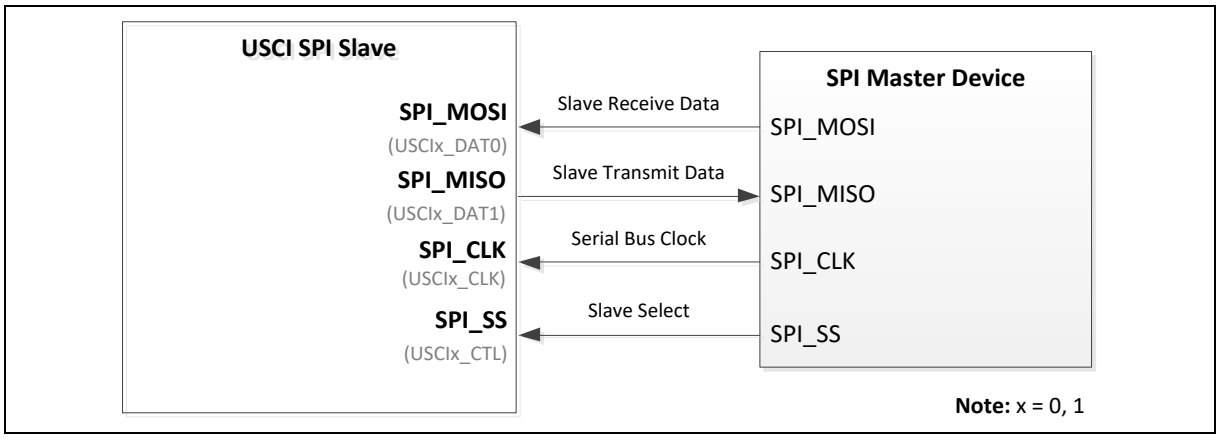


Figure 6.25-2 SPI Slave Mode Application Block Diagram

6.25.2 Features

- Supports Master or Slave mode operation (the maximum frequency -- Master < f_{PCLK} / 2, Slave < f_{PCLK} / 5)
- Configurable bit length of a transfer word from 4 to 16-bit
- Supports one transmit buffer and two receive buffers for data payload
- Supports MSB first or LSB first transfer sequence
- Supports Word Suspend function

- Supports PDMA transfer
- Supports 3-wire, no slave select signal, bi-direction interface
- Supports wake-up function by slave select signal in Slave mode
- Supports one data channel half-duplex transfer

6.25.3 Block Diagram

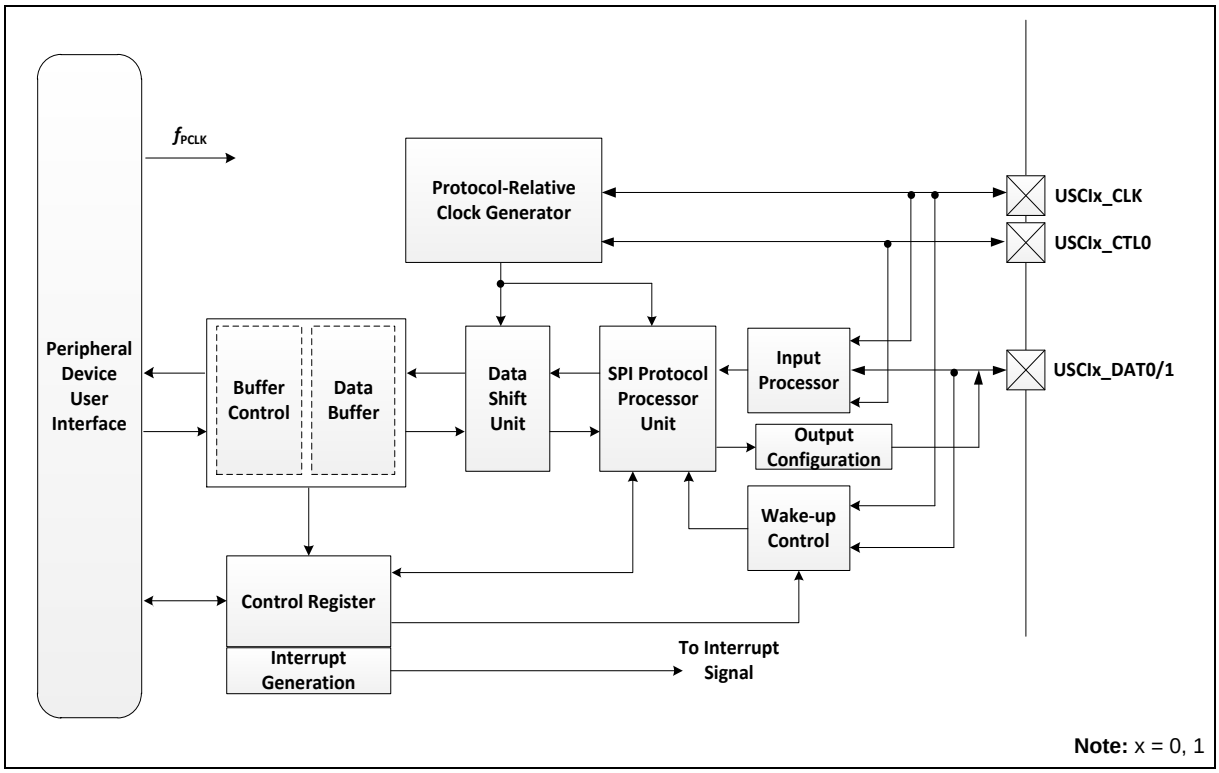


Figure 6.25-3 USCI SPI Mode Block Diagram

6.25.4 Basic Configuration

6.25.4.1 USCI0 SPI Basic Configurations

- Clock Source Configuration
  - Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK1[8]).
  - Enable USCI0_SPI function USPI_CTL[2:0]) register, USPI_CTL[2:0]=3'b001
- Reset Configuration
  - Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).

6.25.4.2 USCI1 SPI Basic Configurations

- Clock Source Configuration
  - Enable USCI1 peripheral clock in USCI1CKEN (CLK_APBCLK1[9]).
  - Enable USCI1_SPI function USPI_CTL[2:0]) register, USPI_CTL[2:0]=3'b001
- Reset Configuration
  - Reset USCI1 controller in USCI1RST (SYS_IPRST2[9]).

6.25.5 Functional Description

6.25.5.1 USCI Common Functional Description

Please refer to USCI chapter for detailed information.

6.25.5.2 Signal Description

A device operating in Master mode controls the start and end of a data transfer, as well as the generation of the SPI bus clock and slave select signal. The slave select signal indicates the start and the end of a data transfer, and the master device can use it to enable the transmitting or receiving operations of Slave device. Slave device receives the SPI bus clock and optionally a slave select signal for data transaction. The signals for SPI communication are shown below.

SPI Mode	Receive Data	Transmit Data	Serial Bus Clock	Slave Select
Full-duplex SPI Master	SPI_MISO (USCIx_DAT1)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
Full-duplex SPI Slave	SPI_MOSI (USCIx_DAT0)	SPI_MISO (USCIx_DAT1)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
Half-duplex SPI Master/Slave	SPI_MOSI (USCIx_DAT0)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)

SPI Communication Signals

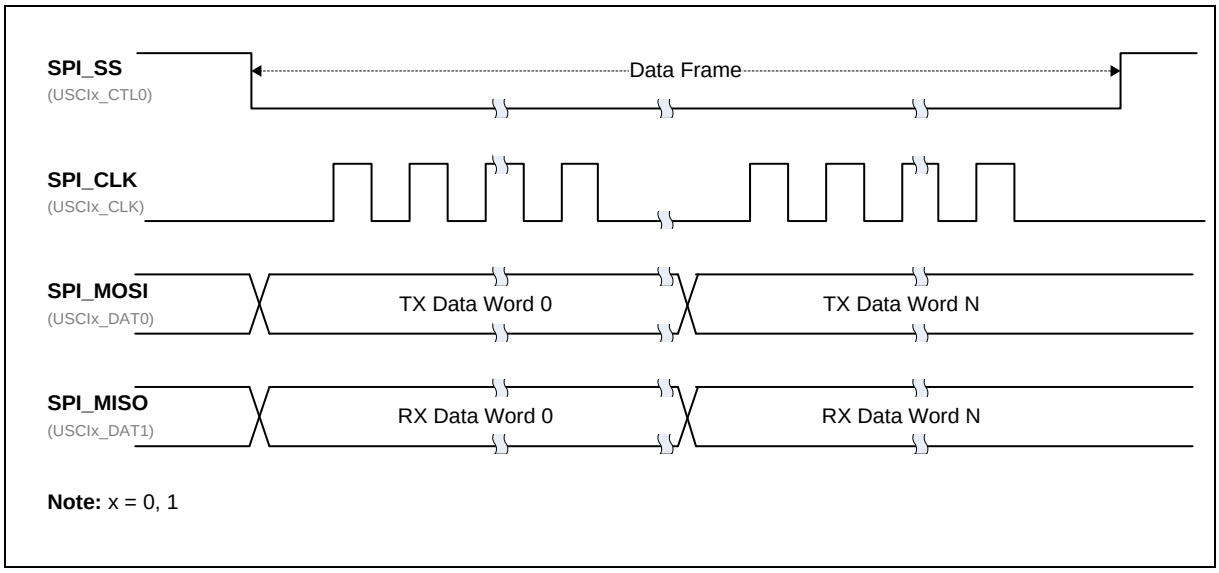


Figure 6.25-4 4-Wire Full-Duplex SPI Communication Signals (Master Mode)

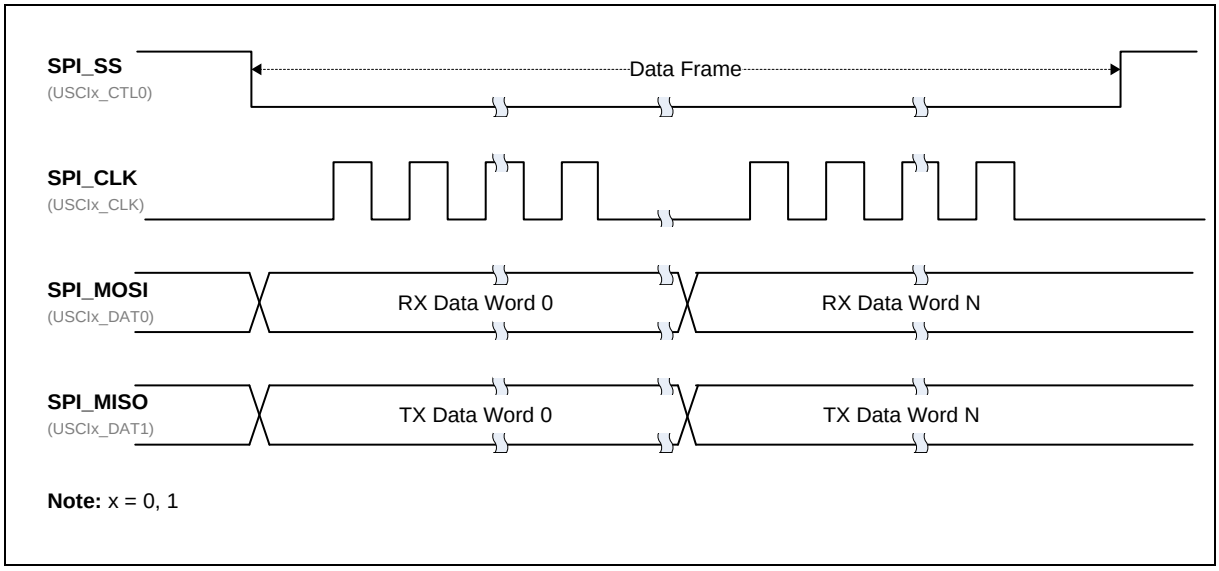


Figure 6.25-5 4-Wire Full-Duplex SPI Communication Signals (Slave Mode)

6.25.5.3 Serial Bus Clock Configuration

The USCI controller needs the peripheral clock to drive the USCI logic unit to perform the data transfer. The peripheral clock frequency is equal to PCLK frequency.

In Master mode, the frequency of the SPI bus clock is determined by protocol-relative clock generator. In general, the SPI bus clock is denoted as SPI clock. The frequency of SPI clock is half of f_{SAMP_CLK}, which can be selected by SPCLKSEL (USPI_BRGEN[3:2]). Refer to USCI chapter for details of protocol-relative clock generator.

In Slave mode, the SPI bus clock is provided by an off-chip Master device. The peripheral clock frequency, f_{PCLK}, of SPI Slave device must be 5-times faster than the serial bus clock rate of the SPI Master device connected together (i.e. the clock rate of serial bus clock < 1/5 peripheral clock f_{PCLK} in Slave mode).

In SPI protocol, SCLKMODE (USPI_PROTCTL[7:6]) defines not only the idle state of serial bus clock but also the serial clock edge used for transmit and receive data. Both Master and Slave devices on the same communication bus should have the same SCLKMODE configuration. The four kinds of serial bus clock configuration are shown below.

SCLKMODE [1:0]	SPI Clock Idle State	Transmit Timing	Receive Timing
0x0	Low	Falling edge	Rising edge
0x1	Low	Rising edge	Falling edge
0x2	High	Rising edge	Falling edge
0x3	High	Falling edge	Rising edge

Table 6.25-1 Serial Bus Clock Configuration

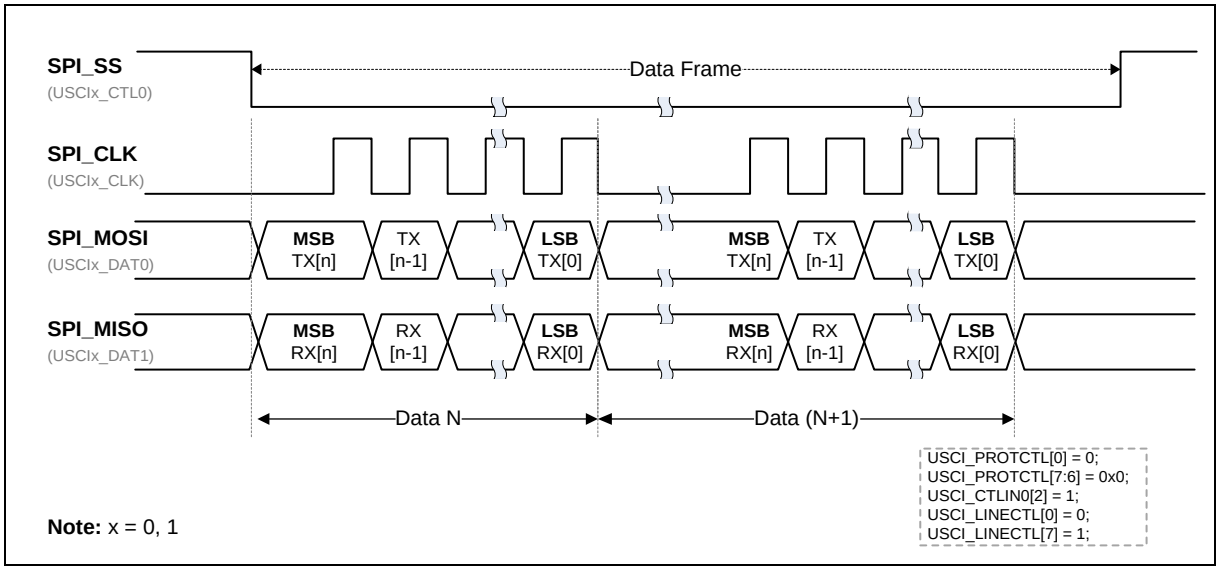


Figure 6.25-6 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x0)

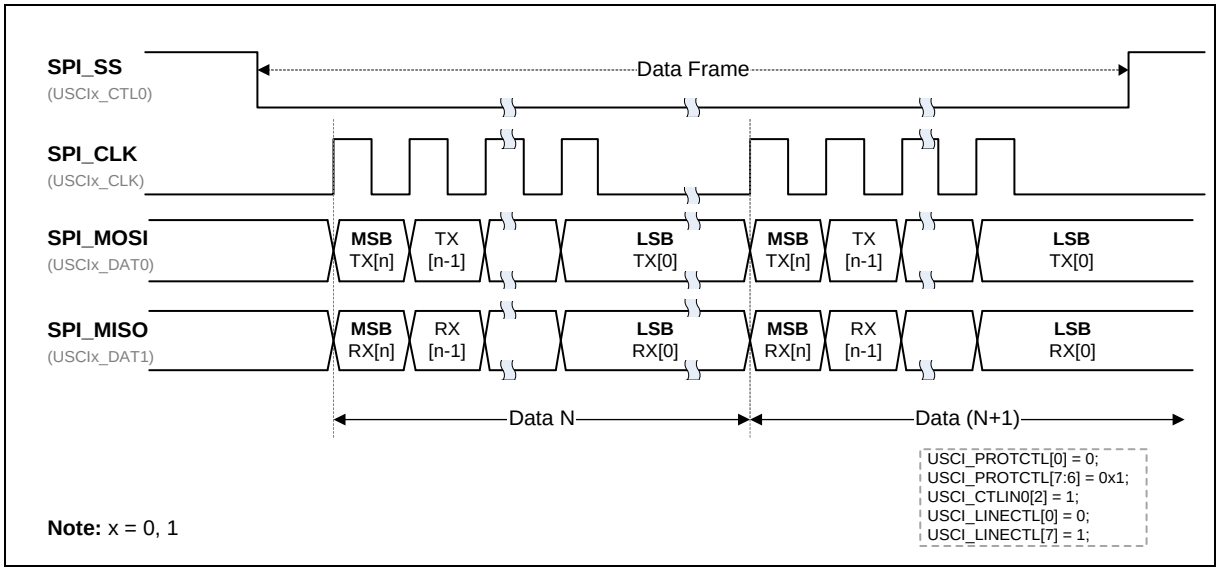


Figure 6.25-7 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x1)

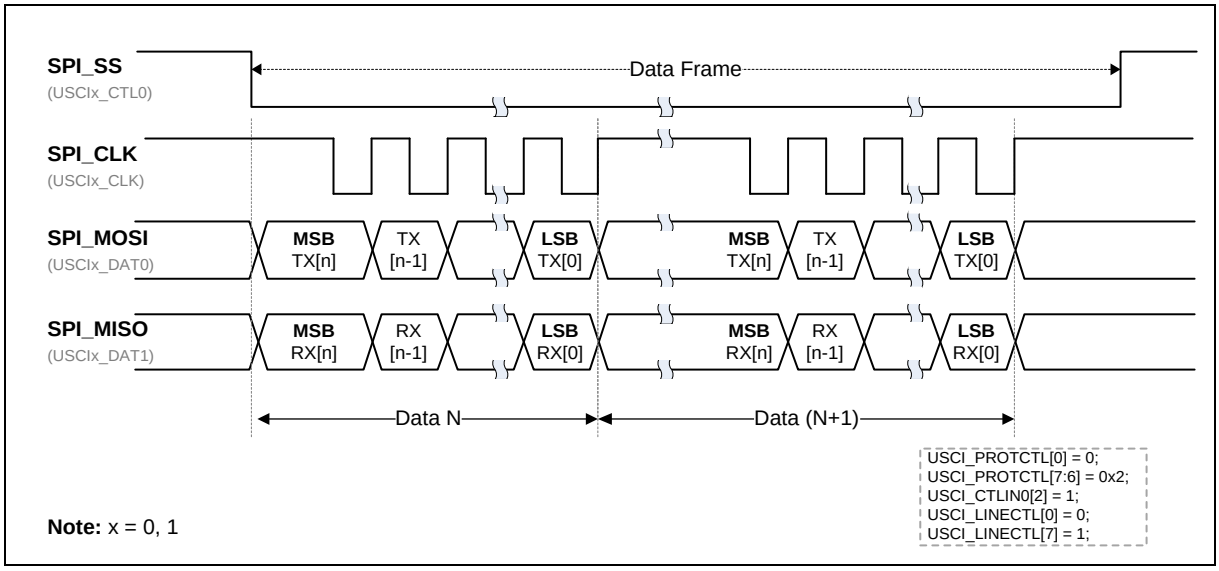


Figure 6.25-8 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x2)

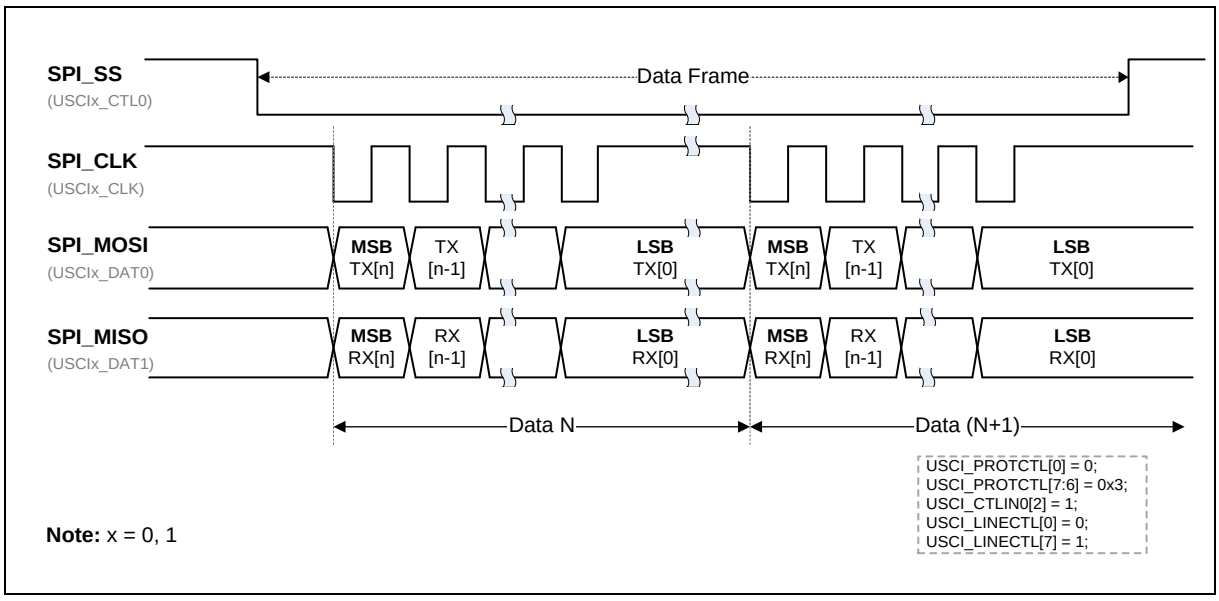


Figure 6.25-9 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x3)

6.25.5.4 Slave Select Signal

The slave selection signal of SPI protocol is active high by default. In SPI Master mode, the USCI controller can drive the control signal to off-chip SPI Slave device through slave select pin SPI_SS (USCIx_CTL0). In SPI Slave mode, the received slave select signal can be inverted by ININV (USPI_CTLIN0[2]).

If the slave select signal of external SPI Master device is low active, the ININV (USPI_CTLIN0[2]) setting of slave device should be set to 1 for the inversion of input control signal. If USCI operates as SPI Master mode, the output slave select inversion CTLOINV (USPI_LINECTL[7]) is also needed to set as 1 for the external SPI Slave device whose slave select signal is active low.

The duration between the slave select active edge and the first SPI clock input edge shall over 2 USCI peripheral clock cycles.

The input slave select signal of SPI Slave has to be kept inactive for at least 2 USCI peripheral clock

cycles between two consecutive frames in order to correctly detect the end of a frame. In Slave mode, the SPI_MISO signal type is controlled by the SPI_SS pin. If the SPI_SS pin is inactive state, the SPI_MISO is set to tri-state to avoid interference with other Slave devices. If the SPI_SS pin is active state, the SPI_MISO is set to output mode for data transfer.

6.25.5.5 Transmit and Receive Data

The bit length of a transmit/receive data word in SPI protocol of USCI controller is defined in DWIDTH (USPI_LINECTL[11:8]), and it can be configured up to 16-bit length for transmitting and receiving data in SPI communication.

The LSB bit (USPI_LINECTL[0]) defines the order of transfer data bit. If the LSB bit is set to 1, the transmission data sequence is LSB first. If the LSB bit is cleared to 0, the transmission data sequence is MSB first.

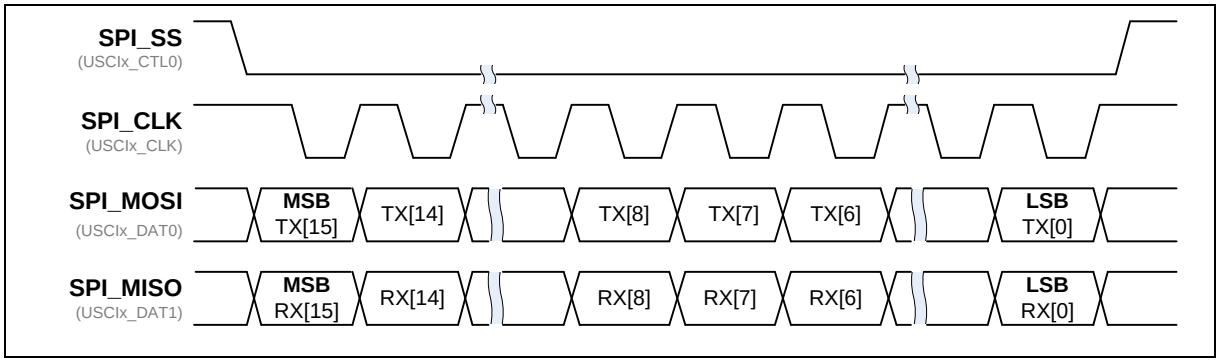


Figure 6.25-10 16-bit Data Length in One Word Transaction with MSB First Format

6.25.5.6 Word Suspend

SUSPITV (USPI_PROTCTL[11:8]) provides a configurable suspend interval, 0.5 ~ 15.5 SPI clock periods, between two successive transaction words in Master mode. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SUSPITV (USPI_PROTCTL[11:8]) is 0x3 (3.5 SPI clock cycles).

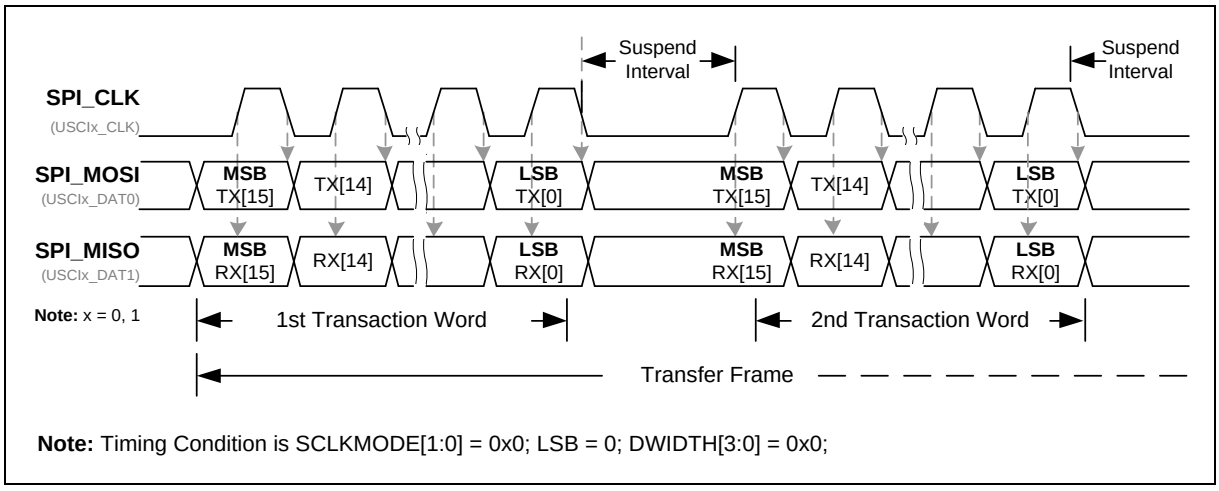


Figure 6.25-11 Word Suspend Interval between Two Transaction Words

6.25.5.7 Automatic Slave Select Function

AUTOSS (USPI_PROTCTL[3]) is used for SPI Master mode to enable the automatic slave select function. If the bit AUTOSS (USPI_PROTCTL[3]) is set, the slave select signal will be generated automatically and the setting value of SS (USPI_PROTCTL[2]) will not affect the output slave select

(through USCIx_CTL0 line). This means that the slave select signal will be asserted by the USCI controller when the SPI data transfer is started by writing to the transmit buffer. And, it will be de-asserted after either all transaction is finished or one word transaction done if the value of SUSPITV (USPI_PROTCTL[11:8]) is equal to or greater than 3.

If the AUTOSS bit (USPI_PROTCTL[3]) is cleared, the slave select on USCIx_CTL0 pin will be asserted/de-asserted by setting/clearing the SS (USPI_PROTCTL[2]). The internal slave select signal is active high and the CTLOINV (USPI_LINECTL[7]) can be used for the inversion of the slave select signal.

In SPI Master mode, if the value of SUSPITV (USPI_PROTCTL[11:8]) is less than 3 and the AUTOSS (USPI_PROTCTL[3]) is set as 1, the slave select signal will be kept at active state between two successive word transactions.

In SPI Slave mode, to recognize the inactive state of the slave select signal, the inactive period of the received slave select signal must be larger than 2 peripheral clock cycles between two successive transactions.

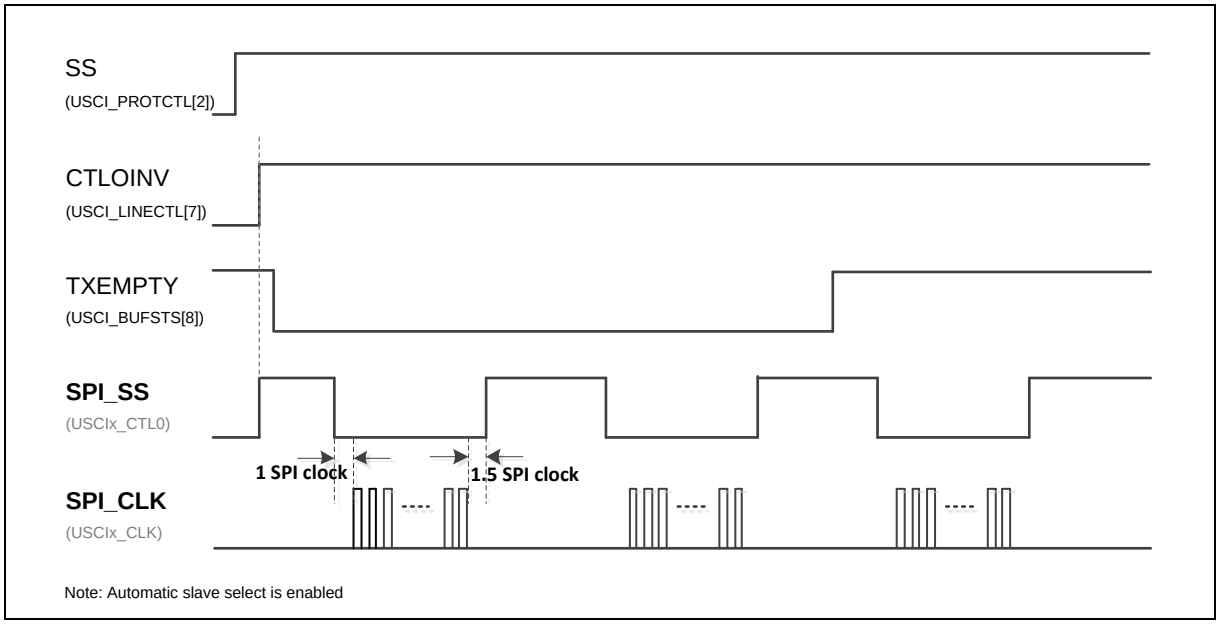


Figure 6.25-12 Auto Slave Select (SUSPITV ≧ 0x3)

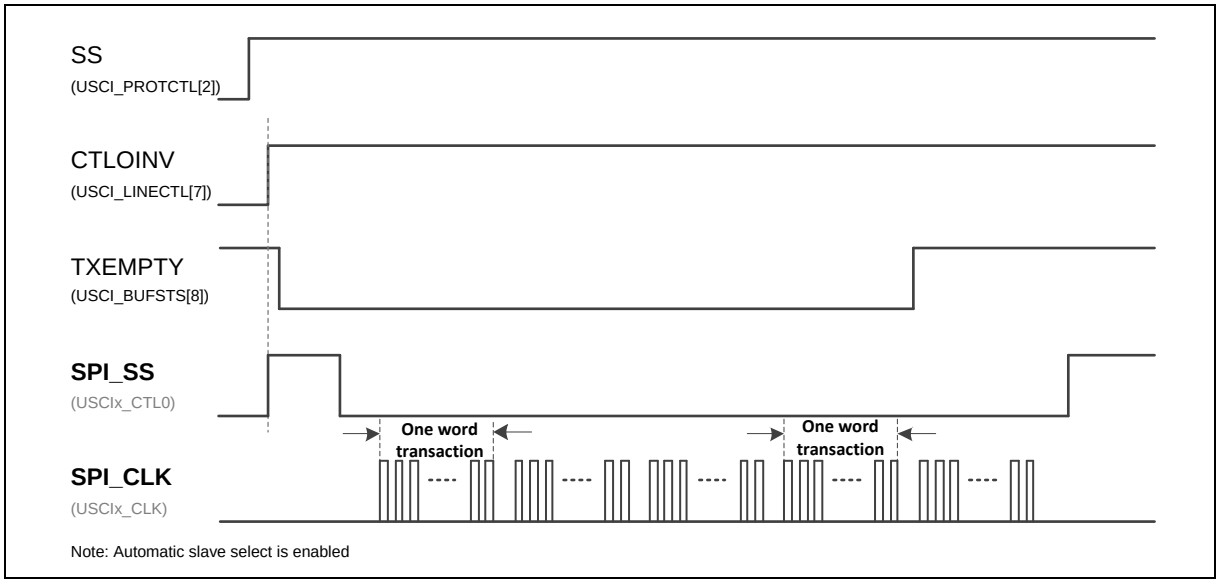


Figure 6.25-13 Auto Slave Select (SUSPITV < 0x3)

6.25.5.8 Slave 3-wire Mode

When the SLV3WIRE (USPI_PROTCTL[1]) is set by software to enable the Slave 3-wire mode, the USCI SPI communication can work with no slave select signal in Slave mode. The SLV3WIRE (USPI_PROTCTL[1]) only takes effect in SPI Slave mode. Only three pins, SPI_CLK (through USCIx_CLK line), SPI_MOSI (through USCIx_DAT0 line), and SPI_MISO (through USCIx_DAT1 line), are required to communicate with a SPI Master. When the SLV3WIRE (USPI_PROTCTL[1]) is set to 1, the SPI Slave will be ready to transmit/receive data after the SPI protocol is enabled by setting FUNMODE(USPI_CTL[2:0]) to 0x1.

6.25.5.9 Data Transfer Mode

The USCI controller supports full-duplex SPI transfer and one data channel half-duplex SPI transfer.

- Full-duplex SPI transfer

  In full-duplex SPI transfer, there are two data pins. One is used for transmitting data and the other is used for receiving data. Thus, data transmission and data reception can be performed simultaneously.

  SCLKMODE (USPI_PROTCTL[7:6]) defines the transition timing of the data shift output signal on USCIx_DAT0 pin. The transition may happen at the corresponding edge of SPI bus clock or active edge of slave select signal. The level of the last data bit of a data word is held on USCIx_DAT0 pin until the next data word begins with the next corresponding edge of the serial bus clock.

- One data channel half-duplex SPI transfer

  In one data channel half-duplex SPI transfer, there is only one data pin for data transfer. Thus, the data transmission and data reception are at different time interval. The data shift direction is determined by PORTDIR (USPI_TXDAT[16]). Refer to the register description for more detailed information.

  The function of one data channel half-duplex SPI transfer is similar to the full-duplex SPI protocol. All the transfer data timing is the same as the full-duplex SPI transfer.

  Figure 6.25-14 shows the one output data channel and one input data channel half-duplex transfer diagrams with the external device.

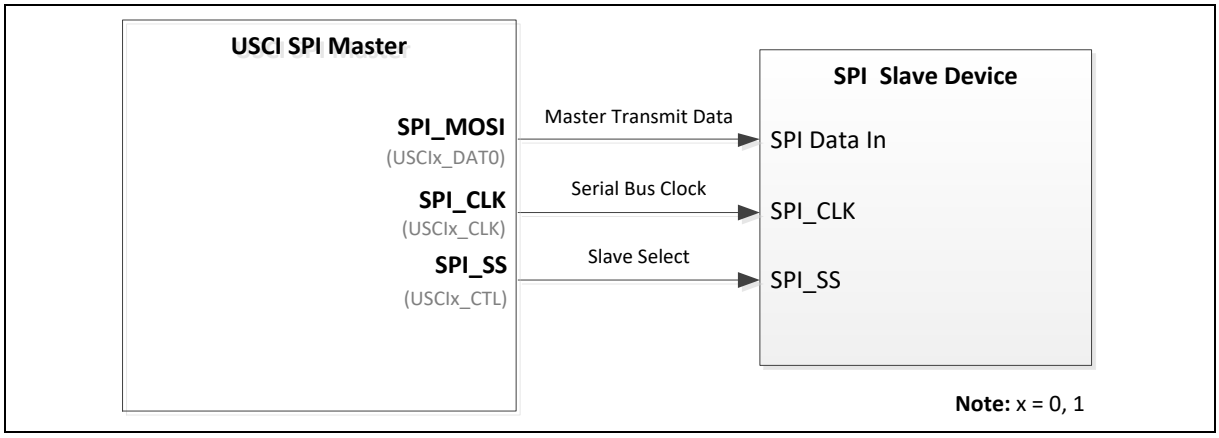


Figure 6.25-14 One Output Data Channel Half-duplex (SPI Master Mode)

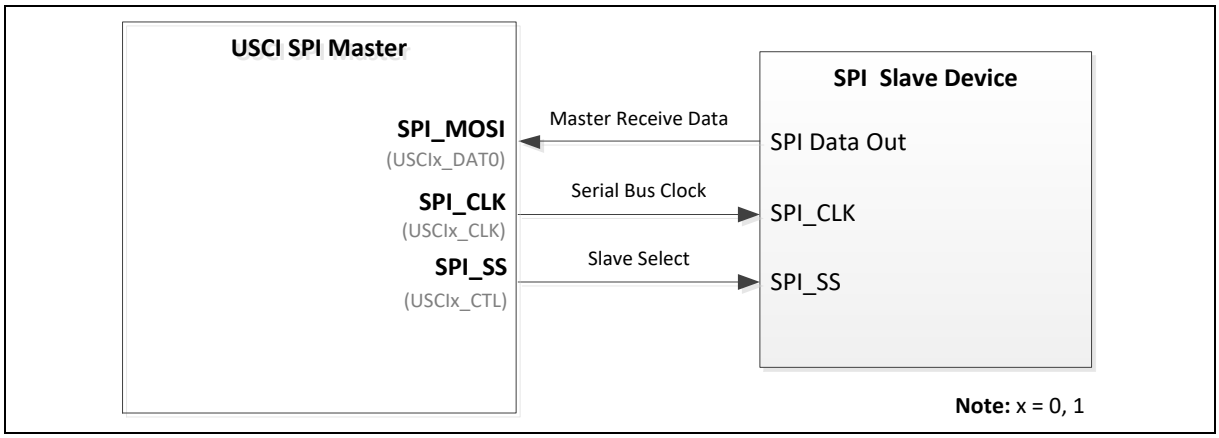


Figure 6.25-15 One Input Data Channel Half-duplex (SPI Master Mode)

The one data channel half-duplex transfer mode can be configured by TSMSEL[2:0] (USPI_PROTCTL[14:12]) and PORTDIR (USPI_TXDAT[16]) settings. When TSMSEL (USPI_PROTCTL[14:12]) is set to 0x4, one data channel half-duplex transfer mode is selected. The PORTDIR (USPI_TXDAT[16]) is used to define the direction of the corresponding transmit data. When the PORTDIR bit is set to 0, the USCI controller will send the corresponding data to external SPI device. When the PORTDIR bit is set to 1, the controller will read the corresponding data from the external SPI device.

For example, in one data channel half-duplex transfer mode with PORTDIR=0, USCI SPI transmits data through USCIx_DAT0 pin; if PORTDIR=1, USCI SPI receives data through USCIx_DAT0 pin.

6.25.5.10 Interrupt

Data Transfer Interrupts

- Transmit start interrupt

  The interrupt event TXSTIF (USPI_PROTSTS[1]) is set after the start of the first data bit of a transmit data word. It can be cleared only by writing 1 to it.

- Transmit end interrupt

  The interrupt event TXENDIF (USPI_PROTSTS[2]) is set after the start of the last data bit of the last transmit data which has been stored in transmit buffer. It can be cleared only by writing 1 to it.

- Receive start interrupt

The interrupt event RXSTIF (USPI_PROTSTS[3]) is set after the start of the first data bit of a receive data word. It can be cleared only by writing 1 to it.

- Receive end interrupt

  The interrupt event RXENDIF (USPI_PROTSTS[4]) is set after the start of the last data bit of a receive data word. It can be cleared only by writing 1 to it.

Protocol-Related Interrupts

- SPI slave select interrupt

  In SPI Slave mode, there are slave select active and in-active interrupt flags, SSACTIF (USPI_PROTSTS[9]) and SSINAIF (USPI_PROTSTS[8]), will be set to 1 when SLAVE (USPI_PROTCTL[0]) is set to 1 and Slave senses the slave select signal active or inactive. The SPI controller will issue an interrupt if SSINAIEN (USPI_PROTIEN[0]) or SSACTIEN (USPI_PROTIEN[1]) is set to 1. Because the internal slave select signal in SPI function is active high, the ININV (USPI_CTLIN0[2]) can be used for inverting the slave select signal comes from an active low device.

- Slave time-out interrupt

  In SPI Slave mode, there is Slave time-out function for user to know that there is no serial clock input during the period of one word transaction. The Slave time-out function uses the timing measurement counter for the calculation of Slave time-out period which is defined by SLVTOCNT (USPI_PROTCTL[25:16]). TMCNTSRC (USPI_BRGEN[5]) can be used for clock frequency selection of timing measurement counter to calculate the Slave time-out period.

  When the timing measurement counter is enabled by TMCNTEN (USPI_BRGEN[4]) and the setting value of SLVTOCNT (USPI_PROTCTL[25:16]) is not 0 in SPI Slave mode, the timing measurement counter will start counting after the first input serial clock of each received word data. This counter will be reset while receiving the following input serial clock and then keep counting. Finally, the timing measurement counter will be cleared and stopped after the finish of the current word transaction. If the value of the time-out counter is equal to or greater than the value of SLVTOCNT (USPI_PROTCTL[25:16]) before one word transaction is done, the Slave time-out interrupt event occurs and the SLVTOIF (USPI_PROTSTS[5]) will be set to 1.

Buffer-Related Interrupts

The buffer-related interrupts are available if there is transmit/receive buffer in USCI controller.

- Receive buffer overrun interrupt

  If there is receive buffer overrun event, RXOVIF (USPI_BUFSTS[3]) will be set as 1. It can be cleared by write 1 into it.

- Transmit buffer under-run interrupt

  If there is transmit buffer under-run event, TXUDRIF (USPI_BUFSTS[11]) will be set as 1. It can be cleared by write 1 into it.

6.25.5.11 Timing Diagram

The slave select signal of USCI SPI protocol is active high by default, and it can be inverted by CTLOINV (USPI_LINECTL[7]) setting.

The idle state of serial bus clock and the serial bus clock edge used for transmit/receive data can be configured by setting SCLKMODE (USPI_PROTCTL[7:6]). The bit length of a transaction word data is determined by DWIDTH (USPI_LINECTL[11:8]), and data bit transfer sequence is determined by LSB (USPI_LINECTL[0]). Four SPI timing diagrams for Master/Slave operations and the related settings are shown below.

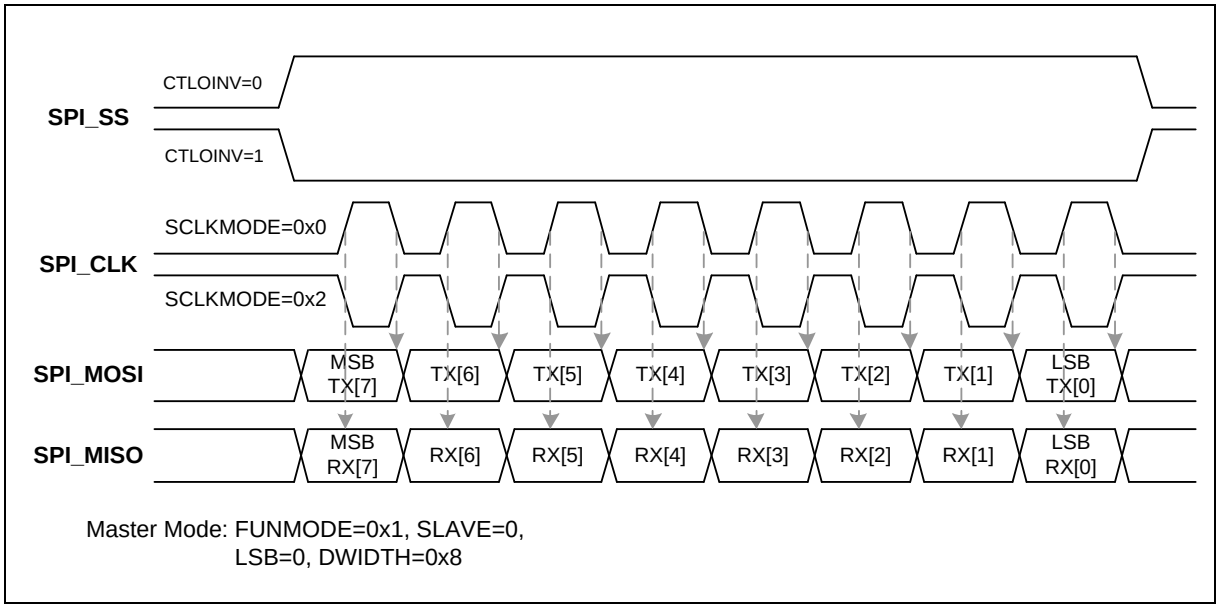


Figure 6.25-16 SPI Timing in Master Mode

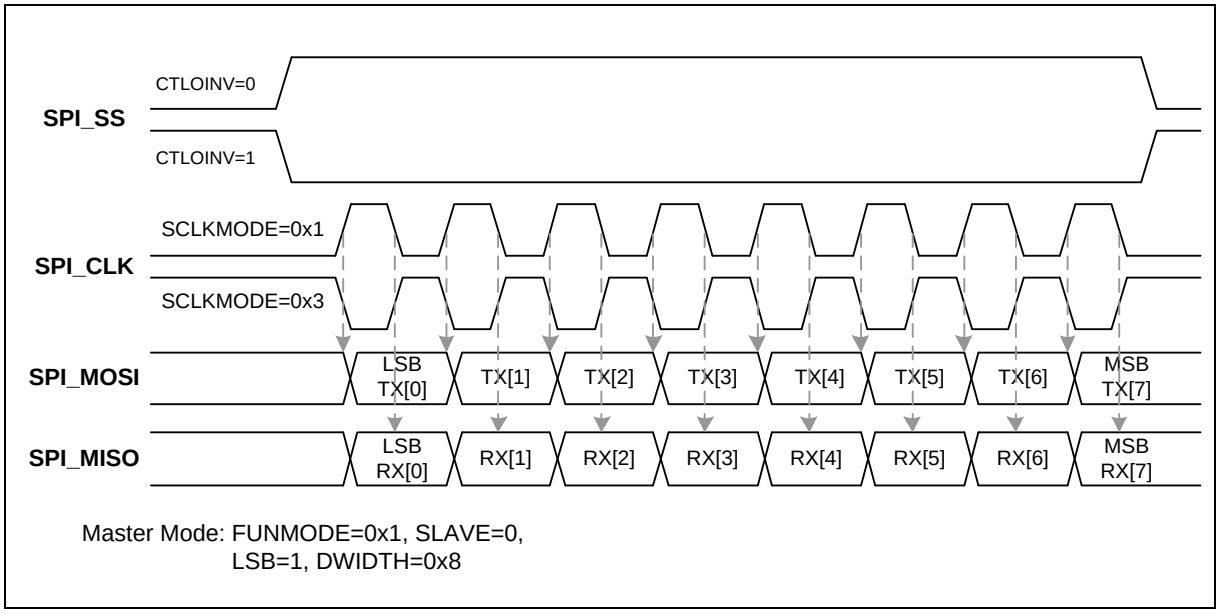


Figure 6.25-17 SPI Timing in Master Mode (Alternate Phase of Serial Bus Clock)

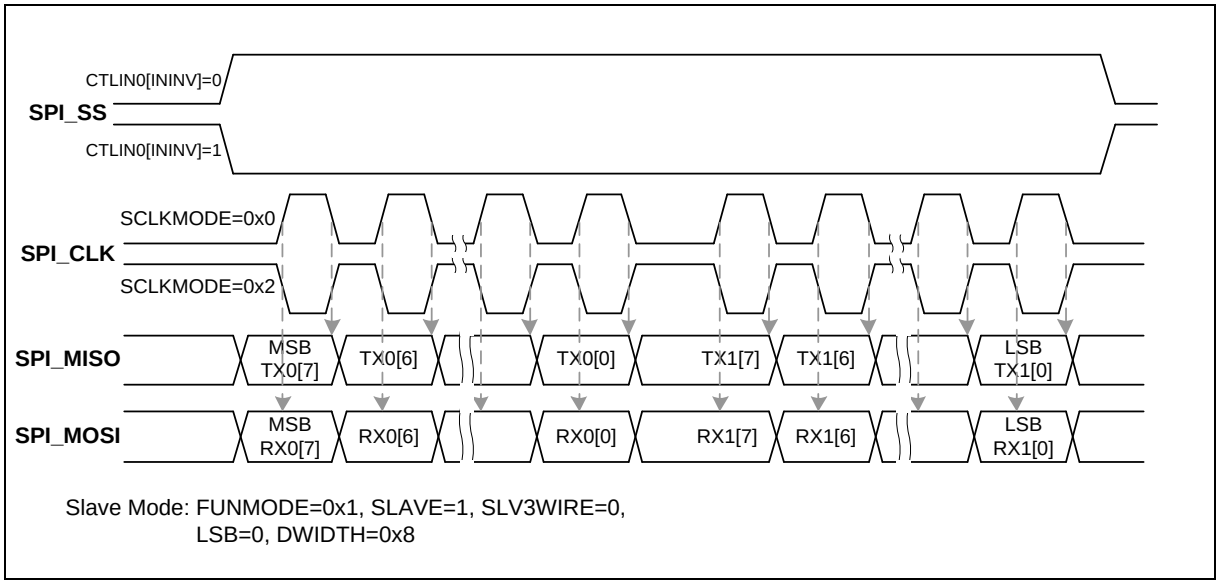


Figure 6.25-18 SPI Timing in Slave Mode

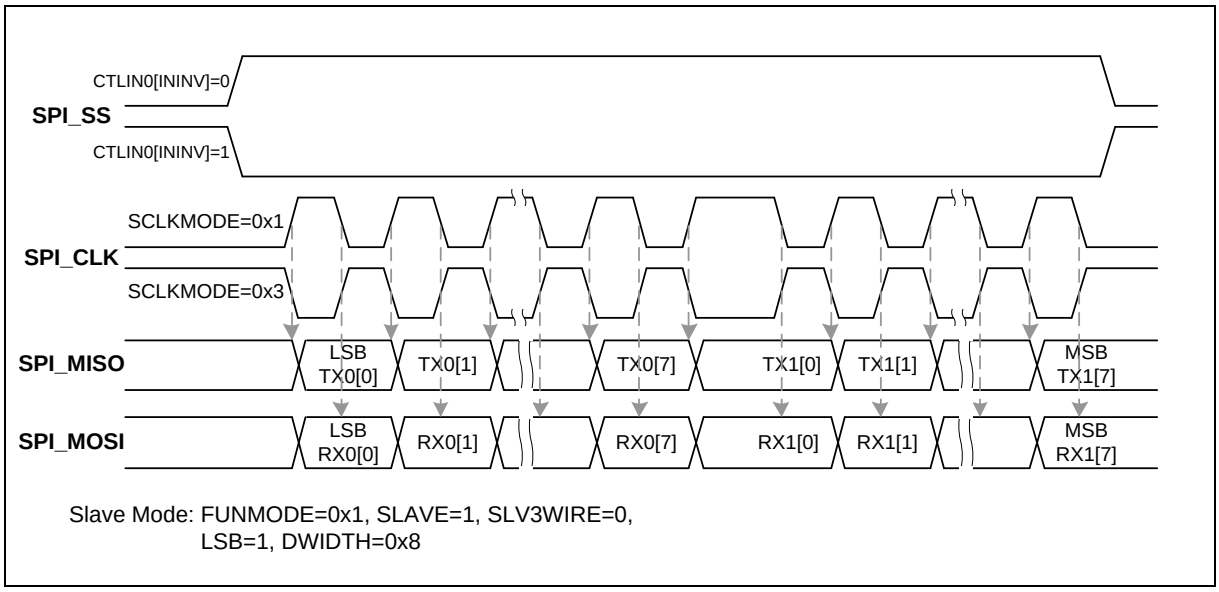


Figure 6.25-19 SPI Timing in Slave Mode (Alternate Phase of Serial Bus Clock)

6.25.5.12 Programming Flow

This section describes the programming flow for USCI SPI data transfer.

For Master mode:

1. Enable USCI peripheral clock by setting CLK_APBCLK1 register.
2. Configure user-specified pins as USCI function pins by setting corresponding multiple function control registers.
3. Set FUNMODE (USPI_CTL[2:0]) to 1 to select SPI mode.
4. Set USPI_BRGEN register to determine the SPI bus clock frequency.
5. According to the requirements of user's application, configure the settings as follows.

- CTLOINV (USPI_LINECTL[7]): If the slave selection signal is active low, set this bit to 1; otherwise, set it to 0.
- DWIDTH (USPI_LINECTL[11:8]): Data width setting.
- LSB (USPI_LINECTL[0]): LSB first or MSB first.
- TSMSEL (USPI_PROTCTL[14:12]): Full-duplex SPI transfer or one channel half-duplex SPI transfer.
- SCLKMODE (USPI_PROTCTL[7:6]): Determine the clock timing.
- AUTOSS (USPI_PROTCTL[3]): Enable automatic slave select function or not.
- SLAVE (USPI_PROTCTL[0]): Set to 0 for Master mode.
- Set PROTEN (USPI_PROTCTL[31]) to 1 to enable SPI protocol.

6. If automatic slave select function is disabled (AUTOSS=0), set SS (USPI_PROTCTL[2]) to 1 before data transfer; set SS to 0 to inactivate the slave selection signal by user's application.
7. Write USPI_TXDAT register to trigger SPI transfer. In half-duplex SPI transfer, the data pin direction is determined by PORTDIR (USPI_TXDAT[16]) setting.
8. User can get the received data by reading USPI_RXDAT register as long as RXEMPTY (USPI_BUFSTS[0]) is 0. The SPI data transfer can be triggered by writing USPI_TXDAT register as long as TXFULL (USPI_BUFSTS[9]) is 0.

For Slave mode:

1. Enable USCI peripheral clock by setting CLK_APBCLK1 register.
2. Configure user-specified pins as USCI function pins by setting corresponding multiple function control registers.
3. Set FUNMODE (USPI_CTL[2:0]) to 1 to select SPI mode.
4. According to the requirements of user's application, configure the settings as follows.
   - ININV (USPI_CTLIN0[2]): If the slave selection signal is active low, set this bit to 1; otherwise, set it to 0.
   - DWIDTH (USPI_LINECTL[11:8]): Data width setting.
   - LSB (USPI_LINECTL[0]): LSB first or MSB first.
   - TSMSEL (USPI_PROTCTL[14:12]): Full-duplex SPI transfer or one channel half-duplex SPI transfer.
   - SCLKMODE (USPI_PROTCTL[7:6]): Determine the clock timing.
   - SLAVE (USPI_PROTCTL[0]): Set to 1 for Slave mode.
5. Set PROTEN (USPI_PROTCTL[31]) to 1 to enable SPI protocol.
6. Write USPI_TXDAT register for transmission. In half-duplex SPI transfer, the data pin direction is determined by PORTDIR (USPI_TXDAT[16]) setting.
7. User can get the received data by reading USPI_RXDAT register as long as RXEMPTY (USPI_BUFSTS[0]) is 0. The next datum for transmission can be written to USPI_TXDAT register as long as TXFULL (USPI_BUFSTS[9]) is 0.

6.25.5.13 Wake-up Function

The USCI Controller in SPI mode supports wake-up system function. The wake-up source in SPI protocol is the transition of input slave select signal.

6.25.6 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
USCI_SPI Base Address: **USPIn_BA = 0x400D_0000 + (0x1000 * n)** **n= 0, 1**				
USPIn_CTL	USPIn_BA+0x00	R/W	USCI Control Register	0x0000_0000
USPIn_INTEN	USPIn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000
USPIn_BRGEN	USPIn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
USPIn_DATIN0	USPIn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000
USPIn_CTLIN0	USPIn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000
USPIn_CLKIN	USPIn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000
USPIn_LINECTL	USPIn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
USPIn_TXDAT	USPIn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
USPIn_RXDAT	USPIn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
USPIn_BUFCTL	USPIn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000
USPIn_BUFSTS	USPIn_BA+0x3C	R/W	USCI Transmit/Receive Buffer Status Register	0x0000_0101
USPIn_PDMACTL	USPIn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000
USPIn_WKCTL	USPIn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
USPIn_WKSTS	USPIn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
USPIn_PROTCTL	USPIn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0300
USPIn_PROTIEN	USPIn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
USPIn_PROTSTS	USPIn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

6.25.7 Register Description

USCI Control Register (USPI_CTL)

Register	Offset	R/W	Description	Reset Value
USPIn_CTL	USPIn_BA+0x00	R/W	USCI Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**FUNMODE**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 000 to FUNMODE. 000 = The USCI is disabled. All protocol related state machines are set to idle state. 001 = The SPI protocol is selected. 010 = The UART protocol is selected. 100 = The I^2C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Interrupt Enable Register (USPI_INTEN)

Register	Offset	R/W	Description	Reset Value
USPIn_INTEN	USPIn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**RXENDIEN**	**RXSTIEN**	**TXENDIEN**	**TXSTIEN**	**Reserved**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**RXENDIEN**	**Receive End Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive finish event. 0 = The receive end interrupt Disabled. 1 = The receive end interrupt Enabled.
[3]	**RXSTIEN**	**Receive Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive start event. 0 = The receive start interrupt Disabled. 1 = The receive start interrupt Enabled.
[2]	**TXENDIEN**	**Transmit End Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit finish event. 0 = The transmit finish interrupt Disabled. 1 = The transmit finish interrupt Enabled.
[1]	**TXSTIEN**	**Transmit Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit start event. 0 = The transmit start interrupt Disabled. 1 = The transmit start interrupt Enabled.
[0]	**Reserved**	Reserved.

USCI Baud Rate Generator Register (USPI_BRGEN)

Register	Offset	R/W	Description	Reset Value
USPIn_BRGEN	USPIn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						**CLKDIV**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CLKDIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TMCNTSRC**	**TMCNTEN**	**SPCLKSEL**		**PTCLKSEL**	**RCLKSEL**

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} (fDIV_CLK = fPROT_CLK / (CLKDIV+1)).
[15:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Time Measurement Counter Clock Source Selection** 0 = Time measurement counter with f_{PROT_CLK}. 1 = Time measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Time Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Time measurement counter Disabled. 1 = Time measurement counter Enabled.
[3:2]	**SPCLKSEL**	**Sample Clock Source Selection** This bit field used for the clock source selection of sample clock (f_{SAMP_CLK}) for the protocol processor. 00 = f_{DIV_CLK}. 01 = f_{PROT_CLK}. 10 = f_{SCLK}. 11 = f_{REF_CLK}.
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source of reference clock (f_{REF_CLK}). 0 = Peripheral device clock f_{PCLK}. 1 = Reserved.

USCI Input Data Signal Configuration (USPI_DATIN0)

Register	Offset	R/W	Description	Reset Value
USPIn_DATIN0	USPIn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ININV**	**Reserved**	**SYNCSEL**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted. **Note:** In SPI protocol, it is suggested this bit should be set as 0.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Signal Synchronization Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, it is suggested this bit should be set as 0.

USCI Input Control Signal Configuration (USPI_CTLIN0)

Register	Offset	R/W	Description	Reset Value
USPIn_CTLIN0	USPIn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ININV**	**Reserved**	**SYNCSEL**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, it is suggested this bit should be set as 0.

USCI Input Clock Signal Configuration (USPI_CLKIN)

Register	Offset	R/W	Description	Reset Value
USPIn_CLKIN	USPIn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SYNCSEL**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, it is suggested this bit should be set as 0.

USCI Line Control Register (USPI_LINECTL)

Register	Offset	R/W	Description	Reset Value
USPIn_LINECTL	USPIn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**DWIDTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTLOINV	**Reserved**	**DATOINV**	**Reserved**				**LSB**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0x0: The data word contains 16 bits located at bit positions [15:0]. 0x1: Reserved. 0x2: Reserved. 0x3: Reserved. 0x4: The data word contains 4 bits located at bit positions [3:0]. 0x5: The data word contains 5 bits located at bit positions [4:0]. ... 0xF: The data word contains 15 bits located at bit positions [14:0].
[7]	**CTLOINV**	**Control Signal Output Inverse Selection** This bit defines the relation between the internal control signal and the output control signal. 0 = No effect. 1 = The control signal will be inverted before its output. **Note:** The control signal has different definitions in different protocol. In SPI protocol, the control signal means slave select signal.
[6]	**Reserved**	Reserved.
[5]	**DATOINV**	**Data Output Inverse Selection** This bit defines the relation between the internal shift data value and the output data signal of USCIx_DAT0/1 pin. 0 = Data output level is not inverted. 1 = Data output level is inverted.
[4:1]	**Reserved**	Reserved.
[0]	**LSB**	**LSB First Transmission Selection**

		0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (USPI_TXDAT)

Register	Offset	R/W	Description	Reset Value
USPIn_TXDAT	USPIn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**PORTDIR**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**PORTDIR**	**Port Direction Control** This bit field is only available while USCI operates in SPI protocol (FUNMODE = 0x1) with half-duplex transfer. It is used to define the direction of the data port pin. When software writes USPI_TXDAT register, the transmit data and its port direction are settled simultaneously. 0 = The data pin is configured as output mode. 1 = The data pin is configured as input mode.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission. In order to avoid overwriting the transmit data, user have to check TXEMPTY (USPI_BUFSTS[8]) status before writing transmit data into this bit field.

USCI Receive Data Register (USPI_RXDAT)

Register	Offset	R/W	Description	Reset Value
USPIn_RXDAT	USPIn_BA+0x34	R	USCI Receive Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer.

USCI Transmit/Receive Buffer Control Register (USPI_BUFCTL)

Register	Offset	R/W	Description	Reset Value
USPIn_BUFCTL	USPIn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						RXRST	TXRST
15	14	13	12	11	10	9	8
RXCLR	RXOVIEN	Reserved					
7	6	5	4	3	2	1	0
TXCLR	TXUDRIEN	Reserved					

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset the receive-related counters, state machine, and the content of receive shift register and data buffer. **Note:** It is cleared automatically after one PCLK cycle.
[16]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset the transmit-related counters, state machine, and the content of transmit shift register and data buffer**.** **Note 1:** It is cleared automatically after one PCLK cycle. **Note 2:** Write 1 to this bit will set the output data pin to zero if USPI_PROTCTL[28]=0.
[15]	**RXCLR**	**Clear Receive Buffer** 0 = No effect. 1 = The receive buffer is cleared. Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[14]	**RXOVIEN**	**Receive Buffer Overrun Interrupt Enable Bit** 0 = Receive overrun interrupt Disabled. 1 = Receive overrun interrupt Enabled.
[13:8]	**Reserved**	Reserved.
[7]	**TXCLR**	**Clear Transmit Buffer** 0 = No effect. 1 = The transmit buffer is cleared. Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[6]	**TXUDRIEN**	**Slave Transmit Under-run Interrupt Enable Bit**

		0 = Transmit under-run interrupt Disabled. 1 = Transmit under-run interrupt Enabled.
[5:0]	**Reserved**	Reserved.

USCI Transmit/Receive Buffer Status Register (USPI_BUFSTS)

Register	Offset	R/W	Description	Reset Value
USPIn_BUFSTS	USPIn_BA+0x3C	R/W	USCI Transmit/Receive Buffer Status Register	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**TXUDRIF**	**Reserved**	**TXFULL**	**TXEMPTY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**RXOVIF**	**Reserved**	**RXFULL**	**RXEMPTY**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**TXUDRIF**	**Transmit Buffer Under-run Interrupt Status** This bit indicates that a transmit buffer under-run event has been detected. If enabled by TXUDRIEN (USPI_BUFCTL[6]), the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit 0 = A transmit buffer under-run event has not been detected. 1 = A transmit buffer under-run event has been detected.
[10]	**Reserved**	Reserved.
[9]	**TXFULL**	**Transmit Buffer Full Indicator** 0 = Transmit buffer is not full. 1 = Transmit buffer is full.
[8]	**TXEMPTY**	**Transmit Buffer Empty Indicator** 0 = Transmit buffer is not empty. 1 = Transmit buffer is empty and available for the next transmission datum.
[7:4]	**Reserved**	Reserved.
[3]	**RXOVIF**	**Receive Buffer Over-run Interrupt Status** This bit indicates that a receive buffer overrun event has been detected. If RXOVIEN (USPI_BUFCTL[14]) is enabled, the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit. 0 = A receive buffer overrun event has not been detected. 1 = A receive buffer overrun event has been detected.
[2]	**Reserved**	Reserved.
[1]	**RXFULL**	**Receive Buffer Full Indicator** 0 = Receive buffer is not full. 1 = Receive buffer is full.
[0]	**RXEMPTY**	**Receive Buffer Empty Indicator** 0 = Receive buffer is not empty.

		1 = Receive buffer is empty.

USCI PDMA Control Register (USPI_PDMACTL)

Register	Offset	R/W	Description	Reset Value
USPIn_PDMACTL	USPIn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PDMAEN**	**RXPDMAEN**	**TXPDMAEN**	**PDMARST**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PDMAEN**	**PDMA Mode Enable Bit** 0 = PDMA function Disabled. 1 = PDMA function Enabled.
[2]	**RXPDMAEN**	**PDMA Receive Channel Available** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[1]	**TXPDMAEN**	**PDMA Transmit Channel Available** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled. **Note 1:** In SPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, RX PDMA function cannot be enabled prior to TX PDMA function. User can enable TX PDMA function firstly or enable both functions simultaneously. **Note 2:** In SPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, TX PDMA function cannot be disabled prior to RX PDMA function. User can disable RX PDMA function firstly or disable both functions simultaneously.
[0]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the USCI's PDMA control logic. This bit will be cleared to 0 automatically.

USCI Wake-up Control Register (USPI_WKCTL)

Register	Offset	R/W	Description	Reset Value
USPIn_WKCTL	USPIn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**PDBOPT**	**Reserved**	**WKEN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDBOPT**	**Power Down Blocking Option** 0 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, MCU will stop the transfer and enter Power-down mode immediately. 1 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, the on-going transfer will not be stopped and MCU will enter idle mode immediately.
[1]	**Reserved**	Reserved.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (USPI_WKSTS)

Register	Offset	R/W	Description	Reset Value
USPIn_WKSTS	USPIn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**WKF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – USPI_PROTCTL (SPI)

Register	Offset	R/W	Description	Reset Value
USPIn_PROTCTL	USPIn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0300

31	30	29	28	27	26	25	24
PROTEN	Reserved		TXUDRPOL	Reserved		SLVTOCNT	
23	22	21	20	19	18	17	16
SLVTOCNT							
15	14	13	12	11	10	9	8
Reserved	TSMSEL			SUSPITV			
7	6	5	4	3	2	1	0
SCLKMODE		Reserved		AUTOSS	SS	SLV3WIRE	SLAVE

Bits	Description	
[31]	**PROTEN**	**SPI Protocol Enable Bit** 0 = SPI Protocol Disabled. 1 = SPI Protocol Enabled.
[30:29]	**Reserved**	Reserved.
[28]	**TXUDRPOL**	**Transmit Under-run Data Polarity for Slave Only** This bit defines the transmitting data level when no data is available for transferring. 0 = The output data level is 0 if TX under run event occurs. 1 = The output data level is 1 if TX under run event occurs.
[27:26]	**Reserved**	Reserved.
[25:16]	**SLVTOCNT**	**Slave Mode Time-out Period for Slave Only** In Slave mode, this bit field is used for Slave time-out period. This bit field indicates how many clock periods (selected by TMCNTSRC, USPI_BRGEN[5]) between the two edges of input SCLK will assert the Slave time-out event. Writing 0x0 into this bit field will disable the Slave time-out function. Example: Assume SLVTOCNT is 0x0A and TMCNTSRC (USPI_BRGEN[5]) is 1, it means the time-out event will occur if the state of SPI bus clock pin is not changed more than (10+1) periods of f_{DIV_CLK}.
[15]	**Reserved**	Reserved.
[14:12]	**TSMSEL**	**Transmit Data Mode Selection** This bit field describes how receive and transmit data is shifted in and out. TSMSEL = 000b: Full-duplex SPI. TSMSEL = 100b: Half-duplex SPI. Other = Reserved. **Note:** Changing the value of this bit field will produce the TXRST and RXRST to clear the TX/RX data buffer automatically.
[11:8]	**SUSPITV**	**Suspend Interval for Master Only** This bit field provides the configurable suspend interval between two successive transmit/receive transaction in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock

		edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. (SUSPITV[3:0] + 0.5) * period of SPI_CLK clock cycle Example: SUSPITV = 0x0 … 0.5 SPI_CLK clock cycle. SUSPITV = 0x1 … 1.5 SPI_CLK clock cycle. …… SUSPITV = 0xE … 14.5 SPI_CLK clock cycle. SUSPITV = 0xF … 15.5 SPI_CLK clock cycle.
[7:6]	**SCLKMODE**	**Serial Bus Clock Mode** This bit field defines the SCLK idle status, data transmit, and data receive edge. 00 = MODE0. The idle state of SPI clock is low level. Data is transmitted with falling edge and received with rising edge. 01 = MODE1. The idle state of SPI clock is low level. Data is transmitted with rising edge and received with falling edge. 10 = MODE2. The idle state of SPI clock is high level. Data is transmitted with rising edge and received with falling edge. 11 = MODE3. The idle state of SPI clock is high level. Data is transmitted with falling edge and received with rising edge.
[5:4]	**Reserved**	Reserved.
[3]	**AUTOSS**	**Automatic Slave Select Function Enable for Master Only** 0 = Slave select signal will be controlled by the setting value of SS (USPI_PROTCTL[2]) bit. 1 = Slave select signal will be generated automatically. The slave select signal will be asserted by the SPI controller when transmit/receive is started, and will be de-asserted after each transmit/receive is finished.
[2]	**SS**	**Slave Select Control for Master Only** If AUTOSS bit is cleared, setting this bit to 1 will set the slave select signal to active state, and setting this bit to 0 will set the slave select signal back to inactive state. If the AUTOSS function is enabled (AUTOSS = 1), the setting value of this bit will not affect the current state of slave select signal. **Note:** In SPI protocol, the internal slave select signal is active high.
[1]	**SLV3WIRE**	**Slave 3-wire Mode Selection for Slave Only** The SPI protocol can work with 3-wire interface (without slave select signal) in Slave mode. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface.
[0]	**SLAVE**	**Slave Mode Selection** 0 = Master mode. 1 = Slave mode.

USCI Protocol Interrupt Enable Register – USPI_PROTIEN (SPI)

Register	Offset	R/W	Description	Reset Value
USPIn_PROTIEN	USPIn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**SLVBEIEN**	**SLVTOIEN**	**SSACTIEN**	**SSINAIEN**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**SLVBEIEN**	**Slave Mode Bit Count Error Interrupt Enable Bit** If data transfer is terminated by slave time-out or slave select inactive event in Slave mode, so that the transmit/receive data bit count does not match the setting of DWIDTH (USPI_LINECTL[11:8]), bit count error event occurs. 0 = Slave mode bit count error interrupt Disabled. 1 = Slave mode bit count error interrupt Enabled.
[2]	**SLVTOIEN**	**Slave Time-out Interrupt Enable Bit** In SPI protocol, this bit enables the interrupt generation in case of a Slave time-out event. 0 = Slave time-out interrupt Disabled. 1 = Slave time-out interrupt Enabled.
[1]	**SSACTIEN**	**Slave Select Active Interrupt Enable Bit** This bit enables/disables the generation of a slave select interrupt if the slave select changes to active. 0 = Slave select active interrupt generation Disabled. 1 = Slave select active interrupt generation Enabled.
[0]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable Bit** This bit enables/disables the generation of a slave select interrupt if the slave select changes to inactive. 0 = Slave select inactive interrupt generation Disabled. 1 = Slave select inactive interrupt generation Enabled.

USCI Protocol Status Register – USPI_PROTSTS (SPI)

Register	Offset	R/W	Description	Reset Value
USPIn_PROTSTS	USPIn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**SLVUDR**	**BUSY**	**SSLINE**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SSACTIF**	**SSINAIF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**SLVBEIF**	**SLVTOIF**	**RXENDIF**	**RXSTIF**	**TXENDIF**	**TXSTIF**	**Reserved**

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**SLVUDR**	**Slave Mode Transmit Under-run Status (Read Only)** In Slave mode, if there is no available transmit data in buffer while transmit data shift out caused by input serial bus clock, this status flag will be set to 1. This bit indicates whether the current shift-out data of word transmission is switched to TXUDRPOL (USPI_PROTCTL[28]) or not. 0 = Slave transmit under-run event does not occur. 1 = Slave transmit under-run event occurs.
[17]	**BUSY**	**Busy Status (Read Only)** 0 = SPI is in idle state. 1 = SPI is in busy state. The following lists the bus busy conditions: ● USPI_PROTCTL[31] = 1 and the TXEMPTY = 0. ● For SPI Master mode, the TXEMPTY = 1 but the current transaction is not finished yet. ● For SPI Slave mode, the USPI_PROTCTL[31] = 1 and there is serial clock input into the SPI core logic when slave select is active. ● For SPI Slave mode, the USPI_PROTCTL[31] = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive.
[16]	**SSLINE**	**Slave Select Line Bus Status (Read Only)** This bit is only available in Slave mode. It used to monitor the current status of the input slave select signal on the bus. 0 = The slave select line status is 0. 1 = The slave select line status is 1.
[15:10]	**Reserved**	Reserved.
[9]	**SSACTIF**	**Slave Select Active Interrupt Flag for Slave Only** This bit indicates that the internal slave select signal has changed to active. It is cleared by software writes one to this bit 0 = The slave select signal has not changed to active.

		1 = The slave select signal has changed to active. **Note:** The internal slave select signal is active high.
[8]	**SSINAIF**	**Slave Select Inactive Interrupt Flag for Slave Only** This bit indicates that the internal slave select signal has changed to inactive. It is cleared by software writes 1 to this bit 0 = The slave select signal has not changed to inactive. 1 = The slave select signal has changed to inactive. **Note:** The internal slave select signal is active high.
[7]	**Reserved**	Reserved.
[6]	**SLVBEIF**	**Slave Bit Count Error Interrupt Flag for Slave Only** 0 = Slave bit count error event did not occur. 1 = Slave bit count error event occurred. **Note:** It is cleared by software write 1 to this bit.
[5]	**SLVTOIF**	**Slave Time-out Interrupt Flag for Slave Only** 0 = Slave time-out event did not occur. 1 = Slave time-out event occurred. **Note:** It is cleared by software write 1 to this bit.
[4]	**RXENDIF**	**Receive End Interrupt Flag** 0 = Receive end event did not occur. 1 = Receive end event occurred. **Note:** It is cleared by software write 1 to this bit.
[3]	**RXSTIF**	**Receive Start Interrupt Flag** 0 = Receive start event did not occur. 1 = Receive start event occurred. **Note:** It is cleared by software write 1 to this bit.
[2]	**TXENDIF**	**Transmit End Interrupt Flag** 0 = Transmit end event did not occur. 1 = Transmit end event occurred. **Note:** It is cleared by software write 1 to this bit.
[1]	**TXSTIF**	**Transmit Start Interrupt Flag** 0 = Transmit start event did not occur. 1 = Transmit start event occurred. **Note:** It is cleared by software write 1 to this bit.
[0]	**Reserved**	Reserved.

6.26 USCI - I²C Mode

6.26.1 Overview

On I²C bus, data is transferred between a Master and a Slave. Data bits transfer on the SCL and SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit. There is one SCL clock pulse for each data bit with the MSB being transmitted first, and an acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (START or STOP). Please refer to Figure 6.26-1 for more detailed I²C BUS Timing.

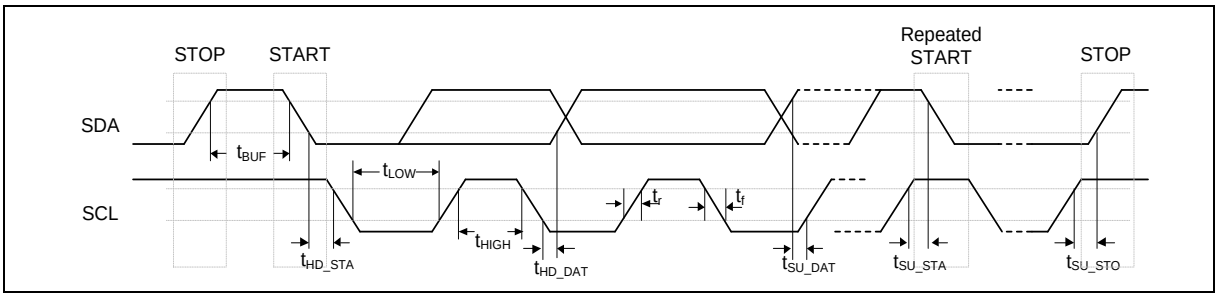


Figure 6.26-1 I²C Bus Timing

The device's on-chip I²C provides the serial interface that meets the I²C bus standard mode specification. The I²C port handles byte transfers autonomously. The I²C mode is selected by FUNMODE (UI2C_CTL [2:0]) = 100B. When enable this port, the USCI interfaces to the I²C bus via two pins: SDA and SCL. When I/O pins are used as I²C ports, user must set the pins function to I²C in advance.

Note: Pull-up resistor is needed for I²C operation because the SDA and SCL are set to open-drain pins when USCI is selected to I²C operation mode.

6.26.2 Features

- Full master and slave device capability
- Supports of 7-bit addressing, as well as 10-bit addressing
- Communication in standard mode (100 kbit/s) or in fast mode (up to 400 kbit/s)
- Supports multi-master bus
- Supports one transmit buffer and two receive buffer for data payload
- Supports 10-bit bus time-out capability
- Supports bus monitor mode.
- Supports Power down wake-up by START signal or address match
- Supports setup/hold time programmable
- Supports multiple address recognition (two slave address with mask option)

6.26.3 Block Diagram

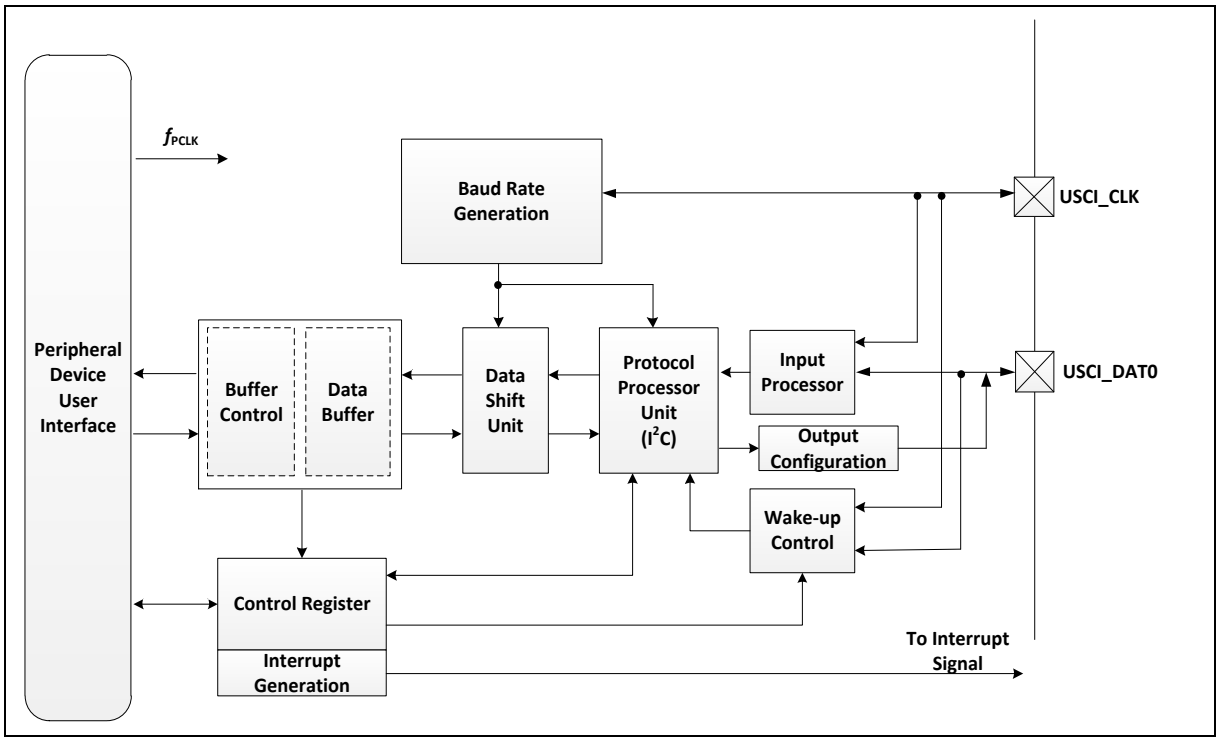


Figure 6.26-2 USCI I²C Mode Block Diagram

6.26.4 Basic Configuration

6.26.4.1 USCI0 I²C Basic Configuration

- Clock Source Configuration
  - Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK1[8]).
  - Enable USCI0_I2C function UI2C0_CTL[2:0]) register, UI2C0_CTL[2:0]=3'b100
- Reset Configuration
  - Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).

6.26.4.2 USCI1 I²C Basic Configuration

- Clock Source Configuration
  - Enable USCI1 peripheral clock in USCI1CKEN (CLK_APBCLK1[9]).
  - Enable USCI1_I2C function UI2C1_CTL[2:0]) register, UI2C1_CTL[2:0]=3'b100
- Reset Configuration
  - Reset USCI1 controller in USCI1RST (SYS_IPRST2[9]).

6.26.5 Functional Description

6.26.5.1 START or Repeated START Signal

Figure 6.26-3 shows the typical I²C protocol. Normally, a standard communication consists of four parts:

- START or Repeated START signal generation
- Slave address and R/W bit transfer
- Data transfer
- STOP signal generation

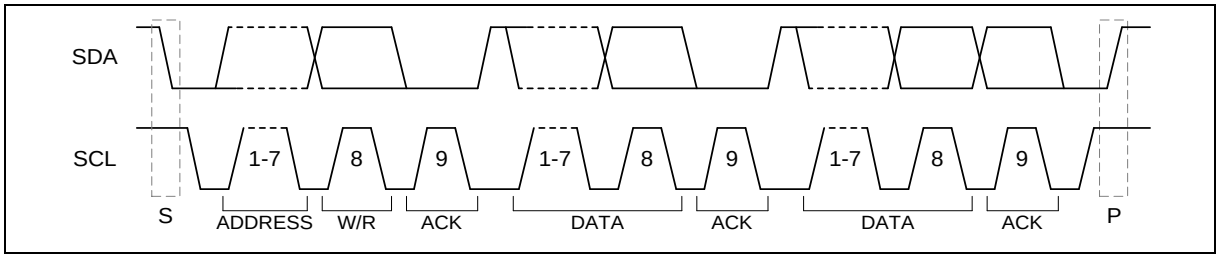


Figure 6.26-3 I²C Protocol

When the bus is free/idle, meaning no master device is engaging the bus (both SCL and SDA lines are high), a master can initiate a transfer by sending a START signal. A START signal, usually referred to as the “S” bit, is defined as a HIGH to LOW transition on the SDA line while SCL is HIGH. The START signal denotes the beginning of a new data transmission.

A Repeated START is not a STOP signal between two START signals and usually referred to as the “Sr” bit. The master uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus idle flag.

6.26.5.2 STOP Signal

The master can terminate the communication by generating a STOP signal. A STOP signal, usually referred to as the “P” bit, is defined as a LOW to HIGH transition on the SDA line while SCL is HIGH. The section between STOP and START is called bus free.

Figure 6.26-4 shows the waveform of START, Repeat START and STOP.

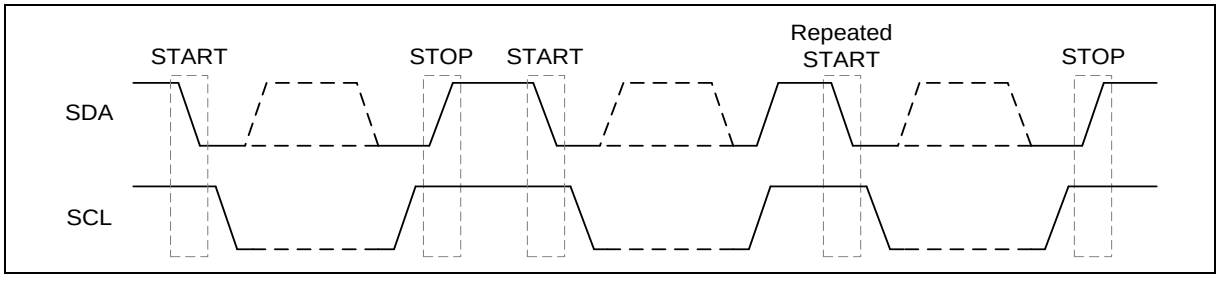


Figure 6.26-4 START and STOP Conditions

6.26.5.3 Slave Address Transfer

After a (Repeated) START condition, the master sends a slave address to identify the target device of the communication. The start address can comprise one or two address bytes (for 7-bit or for 10-bit addressing schemes). After an address byte, a slave sensitive to the transmitted address has to acknowledge the reception.

Therefore, the slave’s address can be programmed in the device, where it is compared to the received address. In case of a match, the slave answers with an acknowledge (SDA = 0). Slaves that are not

targeted answer with a non-acknowledge (SDA = 1). In addition to the match of the programmed address, another address byte value has to be answered with an acknowledge if the slave is capable to handle the corresponding requests. The address byte 00H indicates a general call address that can be acknowledged.

In order to allow selective acknowledges for the different values of the address byte(s), the following control mechanism is implemented:

- If the GCFUNC bit (UI2C_PROTCTL [0]) is set the I²C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.
- The I²C port is equipped with one device address registers, UI2C_DEVADDRn (n = 0~1). In 7-bit address mode, the first 7 bits of a received first address byte are compared to the programmed slave address (UI2C_DEVADDRn [6:0]). If these bits match, the slave sends an acknowledge.
- For 10 bit addressing mode, if the slave address is programmed to 1111 0XXB, the XX bits are compared to the bits UI2C_DEVADDR [9:8] to check for address match and also sends an acknowledge when ADDR10EN (UI2C_PROTCTL [4]) is set. The slave waits for a second address byte compares it with UI2C_DEVADDR [7:0] and sends an acknowledge accordingly to cover the 10 bit addressing mode. The user has to take care about reserved addresses (refer to I²C specification for more detailed description). Only the address 1111 0XXB is supported. Under each of these conditions, bit SLASEL (UI2C_PROTSTS [14]) will be set when the addressing delivered a match. This SLASEL (UI2C_PROTSTS [14]) bit is cleared automatically by a (Repeated) START or STOP condition.
- The I²C port is equipped multiple address recognition with one address mask registers I2C_ADDRMSKn (n = 0~1). When the bit in the address mask register is set to 1, it means the received corresponding address bit is "Don't care". If the bit is set to 0, it means the received corresponding register bit should be exactly the same as address register.

6.26.5.4 Data Transfer

When a slave receives a correct address with an R/W bit, the data will follow R/W bit specified to transfer. Each transferred byte is followed by an acknowledge bit on the 9th SCL clock cycle. If the slave signals a Not Acknowledge (NACK), the master can generate a STOP signal to abort the data transfer or generate a Repeated START signal and start a new transfer cycle.

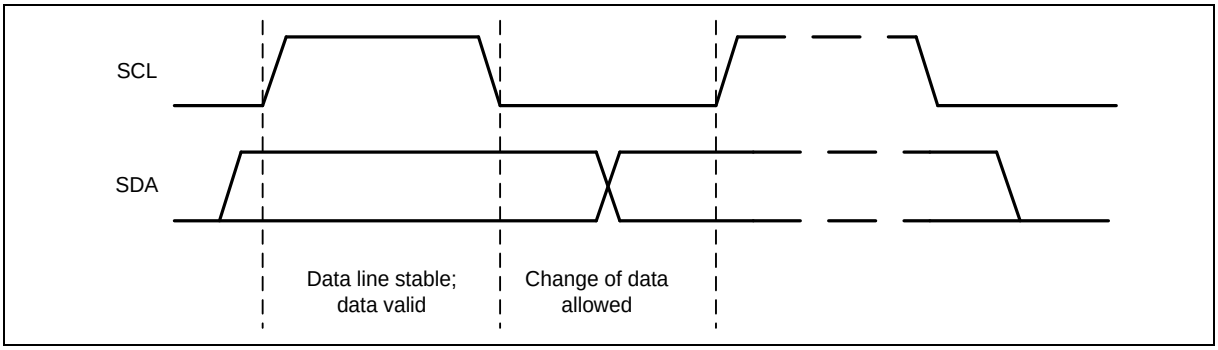


Figure 6.26-5 Bit Transfer on the I²C Bus

If the master received data, does Not Acknowledge (NACK) the slave, the slave releases the SDA line for the master to generate a STOP or Repeated START signal.

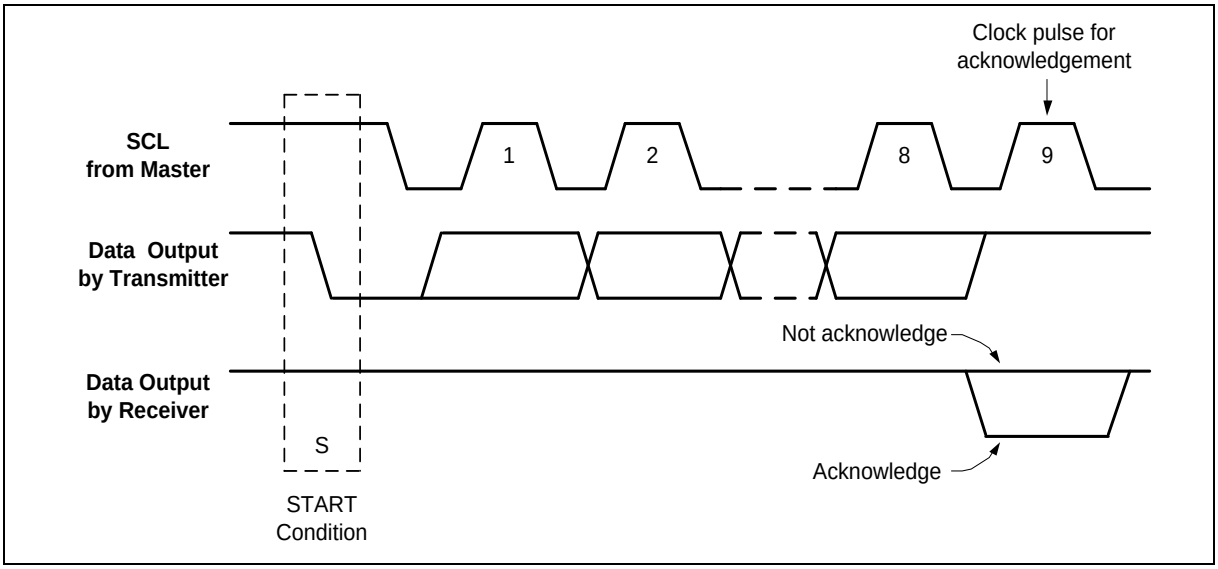


Figure 6.26-6 Acknowledge on the I²C Bus

6.26.5.5 Clock Baud Rate Bits

The data baud rate of I²C is determined by UI2C_BRGEN register when I²C is in Master Mode, and it is not necessary in a Slave mode. In the Slave mode, I²C will automatically synchronize it with any clock frequency from master I²C device. The bits RCLKSEL, SPCLKSEL, PDSCNT, and DSCNT define the baud rate setting:

- RCLKSEL (UI2C_BRGEN [0])

  to define the input frequency f_{REF_CLK}
- SPCLKSEL (UI2C_BRGEN[3:2])

  to define the multiple source of the sample clock f_{SAMP_CLK}
- CLKDIV(UI2C_BRGEN[25:16])

  I²C baud rate: $f_{I2C} = f_{REF_CLK} \times \frac{1}{\text{CLKDIV}+1} \times \frac{1}{2}$

6.26.5.6 Byte Stretching

If a device is selected as master/slave transmit mode and should transmit a data byte but the transmit buffer TXDAT does not contain valid data to be transmitted, the device ties down SCL = 0 at the end of the previous acknowledge bit. The waiting period is finished if software writes 1 to PTRG (UI2C_PROTCTL [5]).

6.26.5.7 Multi-master Arbitration

In some applications, there are two or more masters on the same I²C bus to access slaves, and the masters may transmit data simultaneously. The I²C supports multi-master by including collision detection and arbitration to prevent data corruption.

If two masters sometimes initiate I²C command at the same time, the arbitration procedure determines which master wins and can continue with the command. Arbitration is performed on the SDA signal while the SCL signal is high. Each master checks if the SDA signal on the bus corresponds to the generated SDA signal. If the SDA signal on the bus is low but it should be high, then this master has lost arbitration. Master I²C device that has lost arbitration can generate SCL pulses until the byte ends and must then release the bus and go into slave mode. The arbitration procedure can continue until all the data is transferred. This means that in multi-master system each I²C master must monitor the I²C bus for collisions and act accordingly. Figure 6.26-7 describes master1 data and master2 data are compete arbitration.

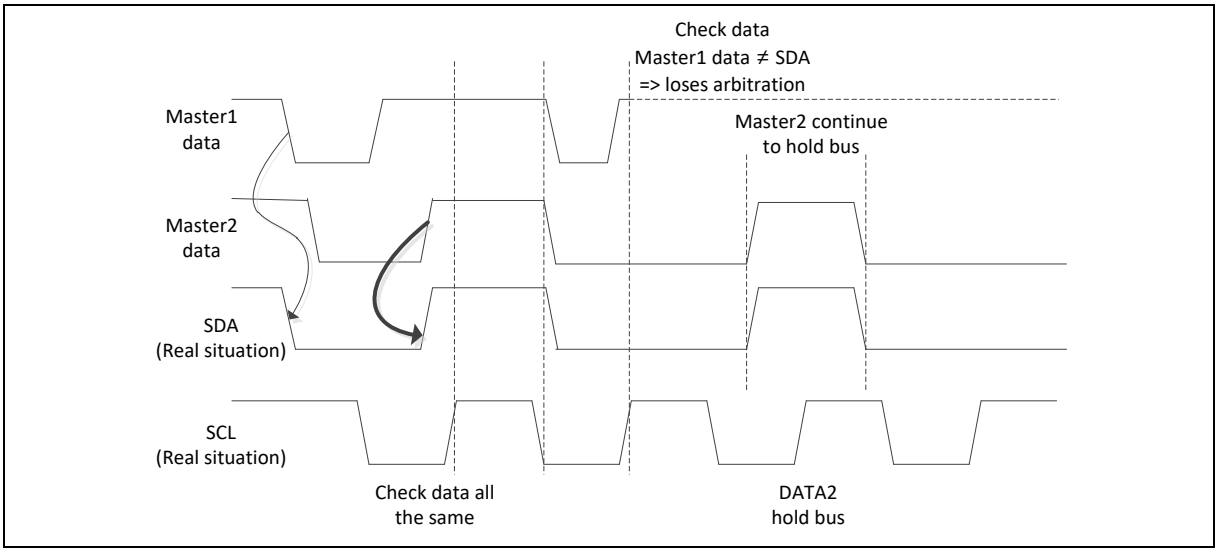


Figure 6.26-7 Arbitration Lost

In this case, during the address and data transmission, the master transmitter checks at the rising edge of SCL for each data bit if the value it is sending is equal to the value read on the SDA line. If yes, master can hold bus continuously. If this is not the case (transmitted value = 1, value read = 0), the master has lost the transmit arbitration. This is indicated by interrupt flag ARBLOIF (UI2C_PROTSTS [11]) and can generate a protocol interrupt if enabled by ARBLOIEN (UI2C_PROTIEN [4]).

When the transmit arbitration has been lost, the software has to initialize the complete frame again, starting with the first address byte together with the START condition for a new master transmit attempt. Arbitration also takes place for the ACK bit. If master arbitration lost and match the device address, then master will turn to slave.

6.26.5.8 Transmission Chain

The I^{2}C bus protocol requiring a kind of in-bit-response during the arbitration phase and while a slave is transmitting, the resulting loop delay of the transmission chain can limit the reachable maximal baud rate, strongly depending on the bus characteristics (bus load, module frequency, etc.).

The shift clock SCL is generated by the master device, output on the wire, then it passes through the input stage and the input filter. Now, the edges can be detected and the SDA data signal can be generated accordingly. The SDA signal passes through the output stage and the wire to the master receiver part. There, it passes through the input stage and the input filter before it is sampled.

This complete loop has to be finished (including all settling times to obtain stable signal levels) before the SCL signal changes again. The delays in this path have to be taken into account for the calculation of the baud rate as a function of f_{PCLK} and f_{PROT_CLK}. We suggest user adopt f_{PCLK}.

6.26.5.9 Non-Acknowledge and Error Conditions

In case of a non-acknowledge (NACKIF (UI2C_PROTSTS [10])) or an error (ERRIF(UI2C_PROTSTS [12])), no further transmission will take place. User software doesn't invalidate the transmit buffer and disable transmissions, before configuring the transmission (by writing TXDAT) again with appropriate values to react on the previous event.

6.26.5.10 I^{2}C Protocol Interrupt Events

The following protocol-related events are generated in I^{2}C mode and can lead to a protocol interrupt.

Please note that the bits in register UI2C_PROTSTS are not all automatically cleared by hardware and have to be cleared by software in order to monitor new incoming events.

- START condition received at a correct position in a frame (STARIF (UI2C_PROTSTS

[8]))

- STOP condition transferred at a correct position in a frame (STORIF (UI2C_PROTSTS [9]))
- Master arbitration lost (ARBLOIF (UI2C_PROTSTS [11]))
- Slave read requested (SLAREAD (UI2C_PROTSTS [15]))
- Acknowledge received (ACKIF (UI2C_PROTSTS [13]))
- Non-acknowledge received (NACKIF (UI2C_PROTSTS [10]))
- START condition not at the expected position in a frame (ERRIF (UI2C_PROTSTS [12]))
- STOP condition not at the expected position in a frame (ERRIF (UI2C_PROTSTS [12]))

6.26.5.11 Operating the I2C

To operate the I^{2}C protocol, the following issues have to be considered:

Select I^{2}C Mode

It is recommended to configure all parameters of the I^{2}C that do not change during run time while FUNMODE (UI2C_CTL [2:0]) = 000B. The I^{2}C control flow has to be done while FUNMODE (UI2C_CTL [2:0]) = 000B to avoid unintended edges of the input signals and the I^{2}C mode can be enabled by FUNMODE (UI2C_CTL [2:0]) = 100B afterwards.

Step 1. Set FUNMODE (UI2C_CTL [2:0]) = 000B

Step 2. Set FUNMODE (UI2C_CTL [2:0]) = 100B

Pin Connections

The pins used for SDA and SCL have to be set to open-drain mode by USCI controller to support the wired-AND structure of the I^{2}C bus lines.

Note: The step to enable the alternate output port functions should only be done after the I^{2}C mode is enabled, to avoided unintended spikes on the output.

Bit Timing Configuration

In standard mode (100 kbit/s) a minimum module frequency of 2 MHz is necessary, whereas in fast mode (400 kbit/s) a minimum of 10 MHz is required. Additionally, if the digital filter stage should be used to eliminate spikes up to 50 ns, a filter frequency of 20 MHz is necessary. There could be an uncertainty in the SCL high phase timing of maximum $1/f_{PROT_CLK}$ if another I^{2}C participant lengthens the SCL low phase on the bus. Note that the SCL maximum frequency is $f_{SAMP_CLK}/2$ and the SPCLKSEL (UI2C_BRGEN [3:2]) must be set to 0 for selecting $f_{SAMP_CLK} = f_{DIV_CLK}$.

Data Format Configuration

The data format has to be configured for 8 data bits (DWIDTH (UI2C_LINECTL [11:8]) = 8), and MSB shifted first (LSB (UI2C_LINECTL [0]) = 0). As a result, UI2C_LINECTL has to be set to 0x800.

Control Flow

The on-chip I^{2}C ports support three operation modes, Master, Slave, and General Call Mode.

In a given application, I^{2}C port may operate as a master or as a slave. In Slave mode, the I^{2}C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master (by setting the AA bit), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, hardware waits until the bus is free before entering Master mode so that a possible slave action is not be interrupted. If address arbitration is lost in Master mode, I^{2}C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer.

To control the I²C bus transfer in each mode, user needs to set UI2C_PROTCTL, UI2C_PROTIEN, TXDAT registers according to current status of UI2C_PROTSTS register. In other words, for each I²C bus action, user needs to check current status by UI2C_PROTSTS register, and then set UI2C_PROTCTL, UI2C_PROTIEN, TXDAT registers to take bus action. Finally, check the response status by UI2C_PROTSTS.

The bits, STA, STO and AA in UI2C_PROTCTL register are used to control the next state of the I²C hardware after interrupt signal is cleared. Upon completion of the new action, a new status will be updated in UI2C_PROTSTS register will be set. If the I²C interrupt control bit of UI2C_PROTIEN is set, appropriate action or software branch of the new status can be performed in the Interrupt service routine.

Figure 6.26-8 shows the current I²C STARIF (UI2C_PROTSTS [8]) is set to 1 by hardware, and then set TXDAT = SLA+W (Slave address + Write bit), (PTRG, STA, STO, AA) = (1, 0, 0, x) to send the address to I²C bus, and write 1 to STARIF (UI2C_PROTSTS [8]) to clear flag. If a slave on the bus matches the address and response ACK, the UI2C_PROTSTS will be updated by ACKIF (UI2C_PROTSTS [13]) setting.

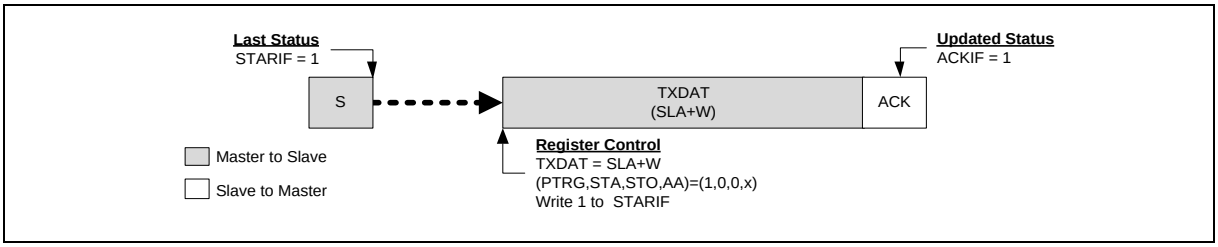


Figure 6.26-8 Control I²C Bus according to Current I²C Status

Data Transfer on the I²C Bus

Figure 6.26-9 shows a master transmits data to slave. A master addresses a slave with a 7-bit address and 1-bit write index to denote that the master wants to transmit data to the slave. The master keeps transmitting data after the slave returns acknowledge to the master.

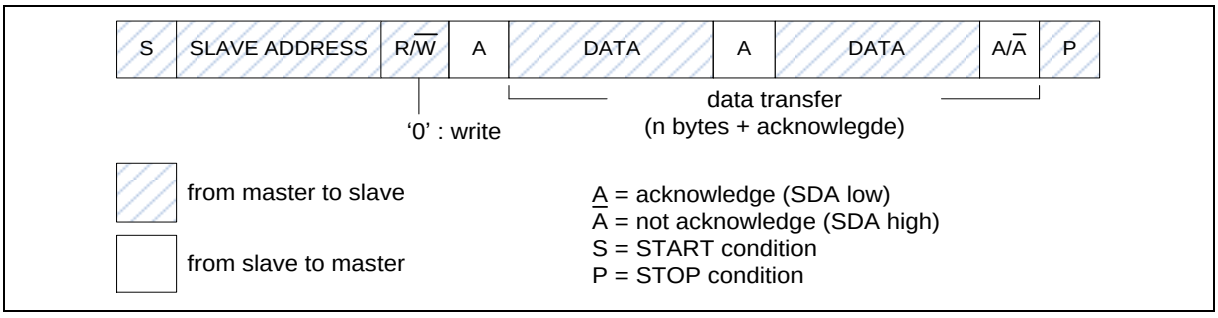


Figure 6.26-9 Master Transmits Data to Slave with a 7-bit Address

Figure 6.26-10 shows a master read data from slave. A master addresses a slave with a 7-bit address and 1-bit read index to denote that the master wants to read data from the slave. The slave will start transmitting data after the slave returns acknowledge to the master.

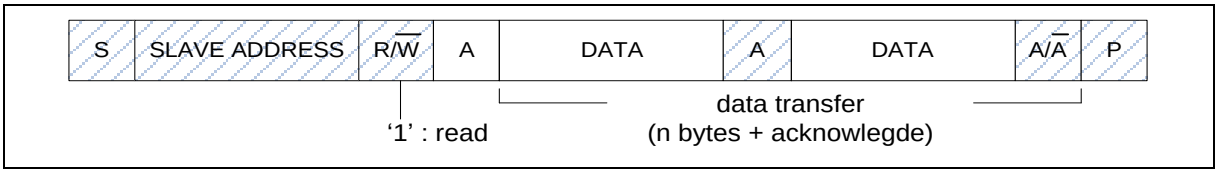


Figure 6.26-10 Master Reads Data from Slave with a 7-bit Address

Figure 6.26-11 shows a master transmits data to slave by 10-bit address. A master addresses a slave with a 10-bit address. First byte contains 10-bit address indicator (5'b11110) and 2-bit address with write

index, second byte contains 8-bit address. The master keeps transmitting data after the second byte end. Note that 7-bit and 10-bit address device can work on the same bus.

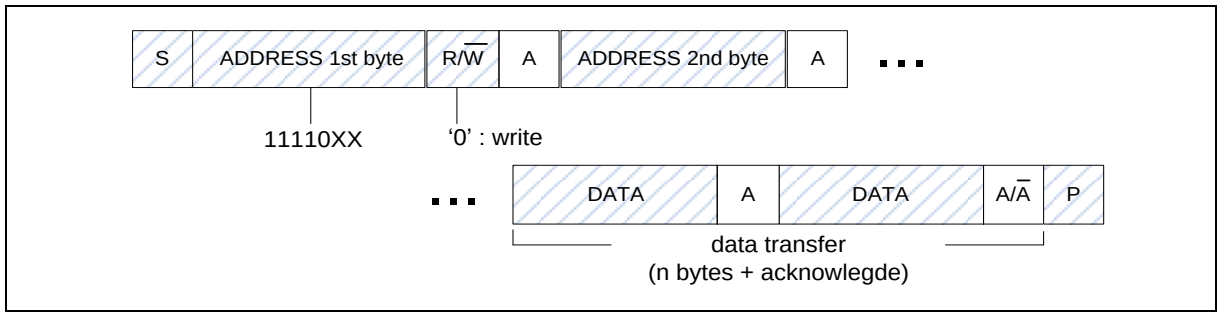


Figure 6.26-11 Master Transmits Data to Slave by 10-bit Address

Figure 6.26-12 shows a master read data from slave by 10-bit address. A master addresses a slave with a 10-bit address. First mater transmits 10-bit address to slave, after that master transmits first byte with read index. The slave will start transmitting data after the first byte with read index.

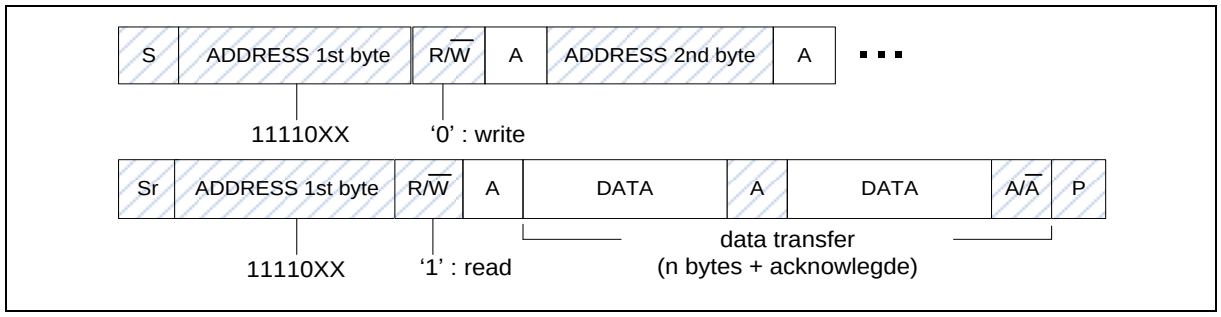


Figure 6.26-12 Master Reads Data from Slave by 10-bit Address

Master Mode

In Figure 6.26-13 and Figure 6.26-14, all possible protocols for I^2C master are shown. User needs to follow proper path of the flow to implement required I^2C protocol.

In other words, user can send a START signal to bus and I^2C will be in Master Transmitter mode (Figure 6.26-13) or Master receiver mode (Figure 6.26-14) after START signal has been sent successfully and new status register would be set STARIF (UI2C_PROTSTS [8]). Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^2C protocol.

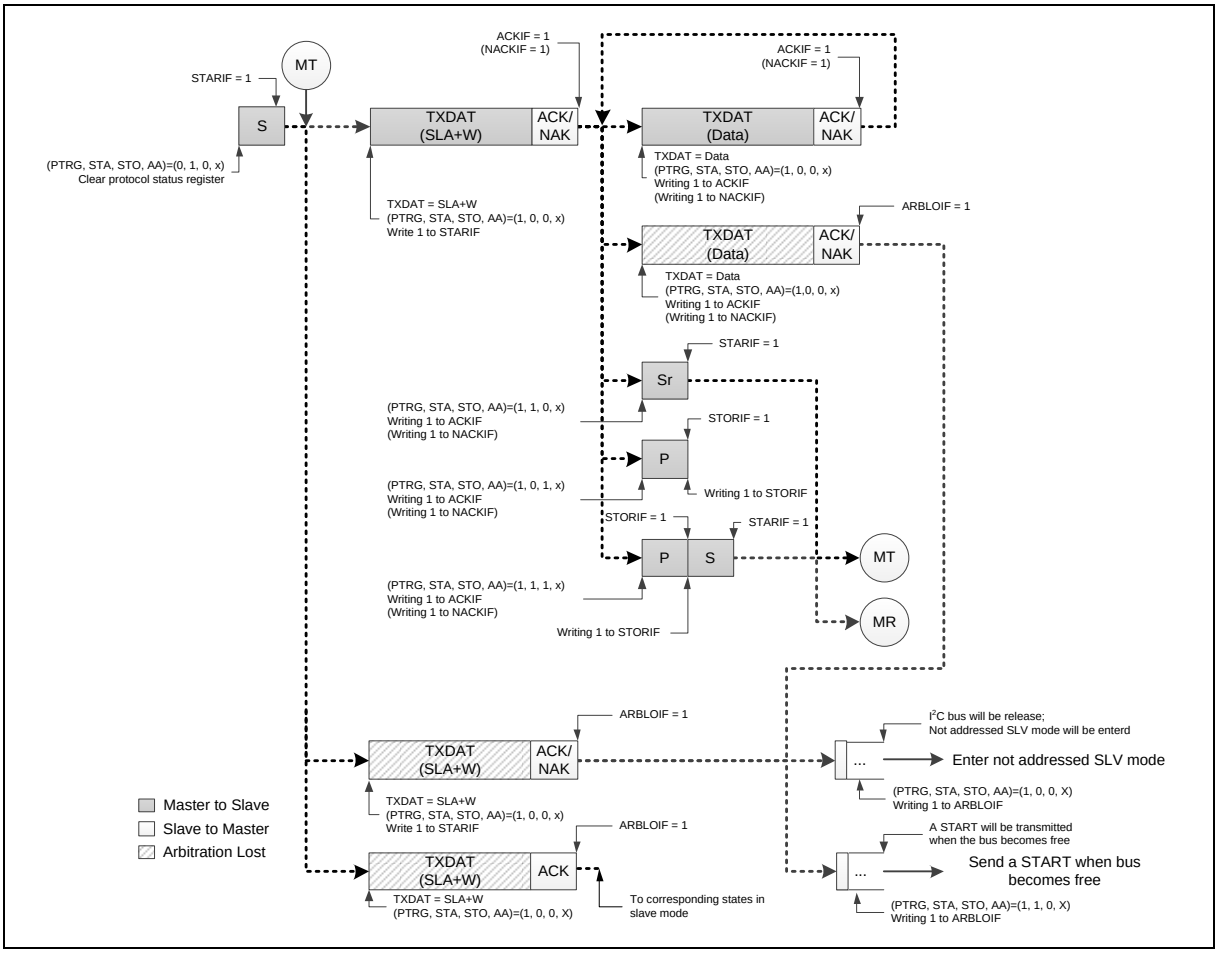


Figure 6.26-13 Master Transmitter Mode Control Flow with 7-bit Address

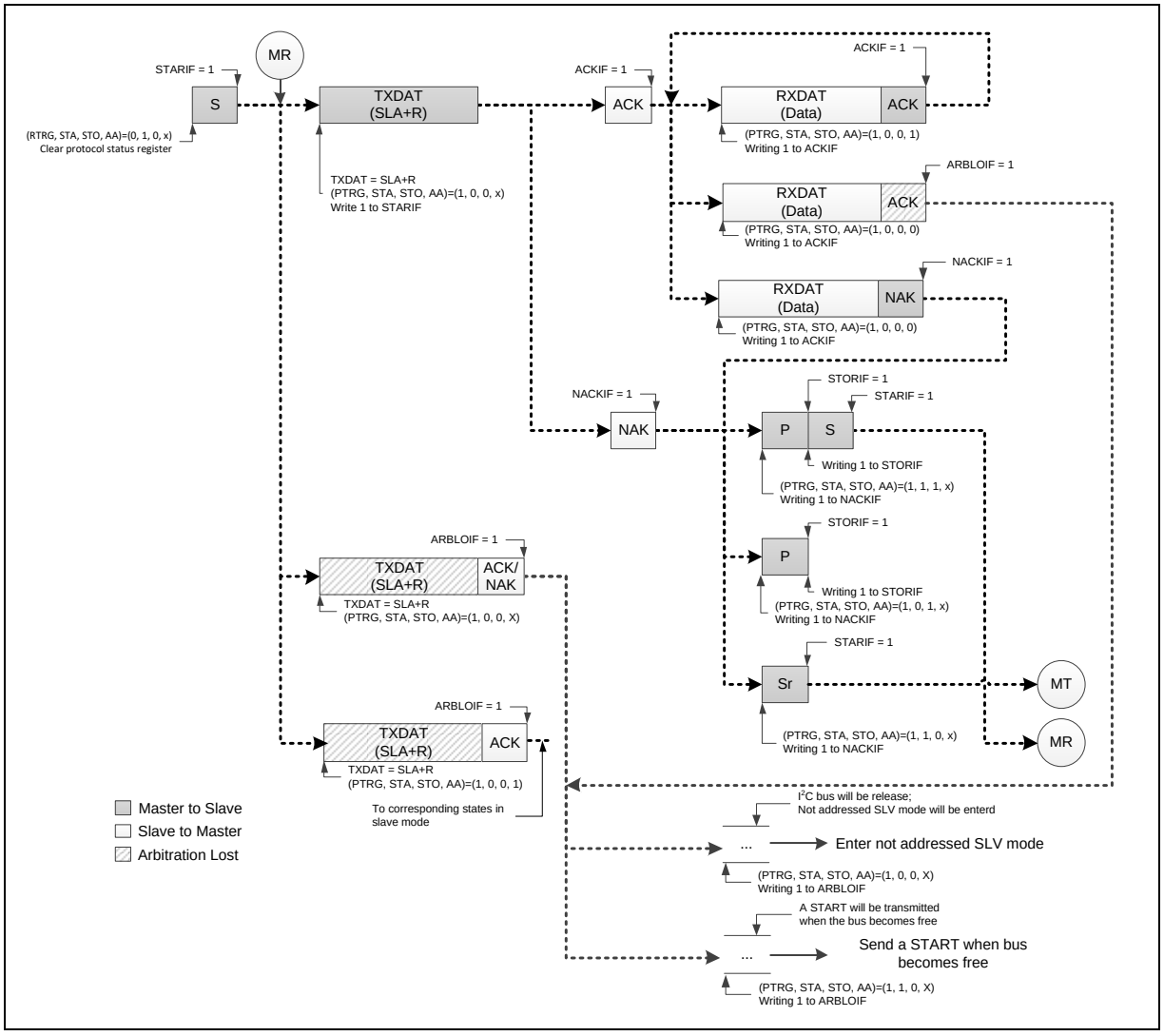


Figure 6.26-14 Master Receiver Mode Control Flow with 7-bit Address

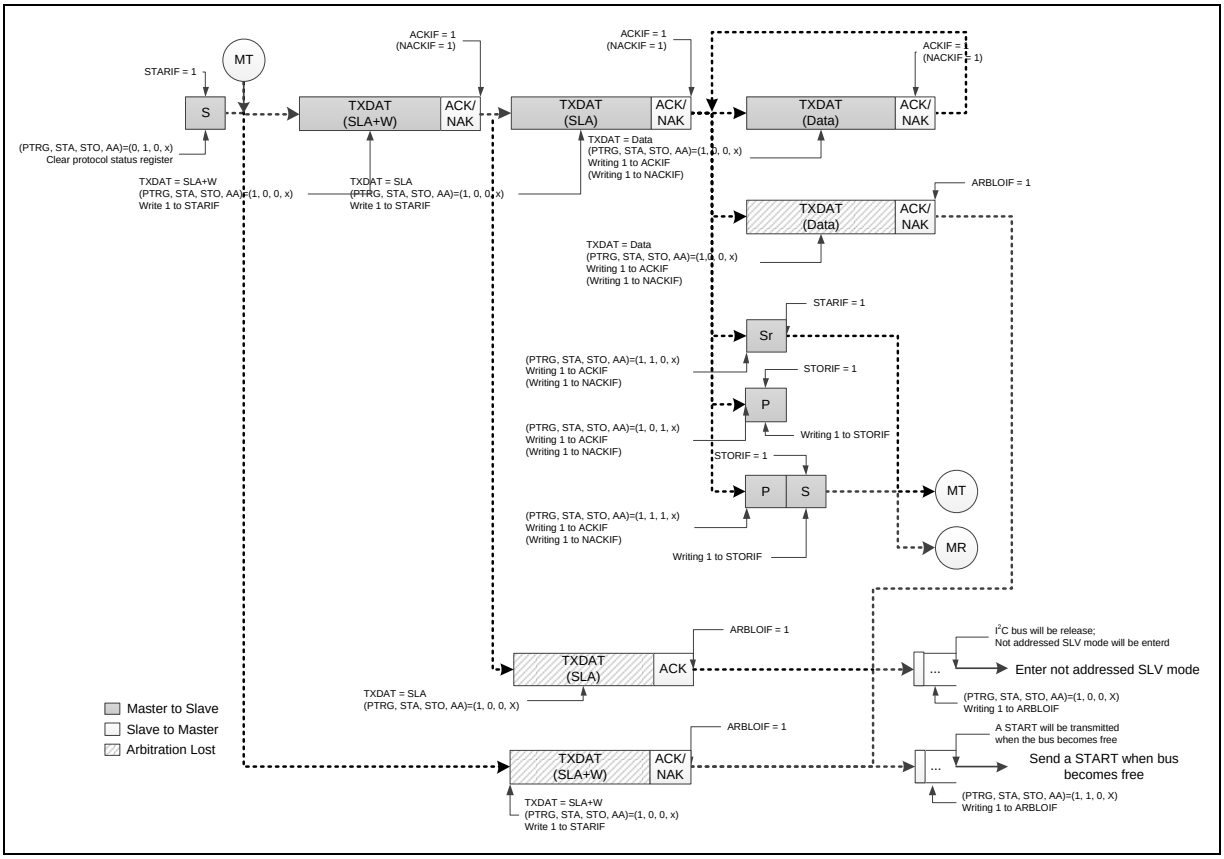


Figure 6.26-15 Master Transmitter Mode Control Flow with 10-bit Address

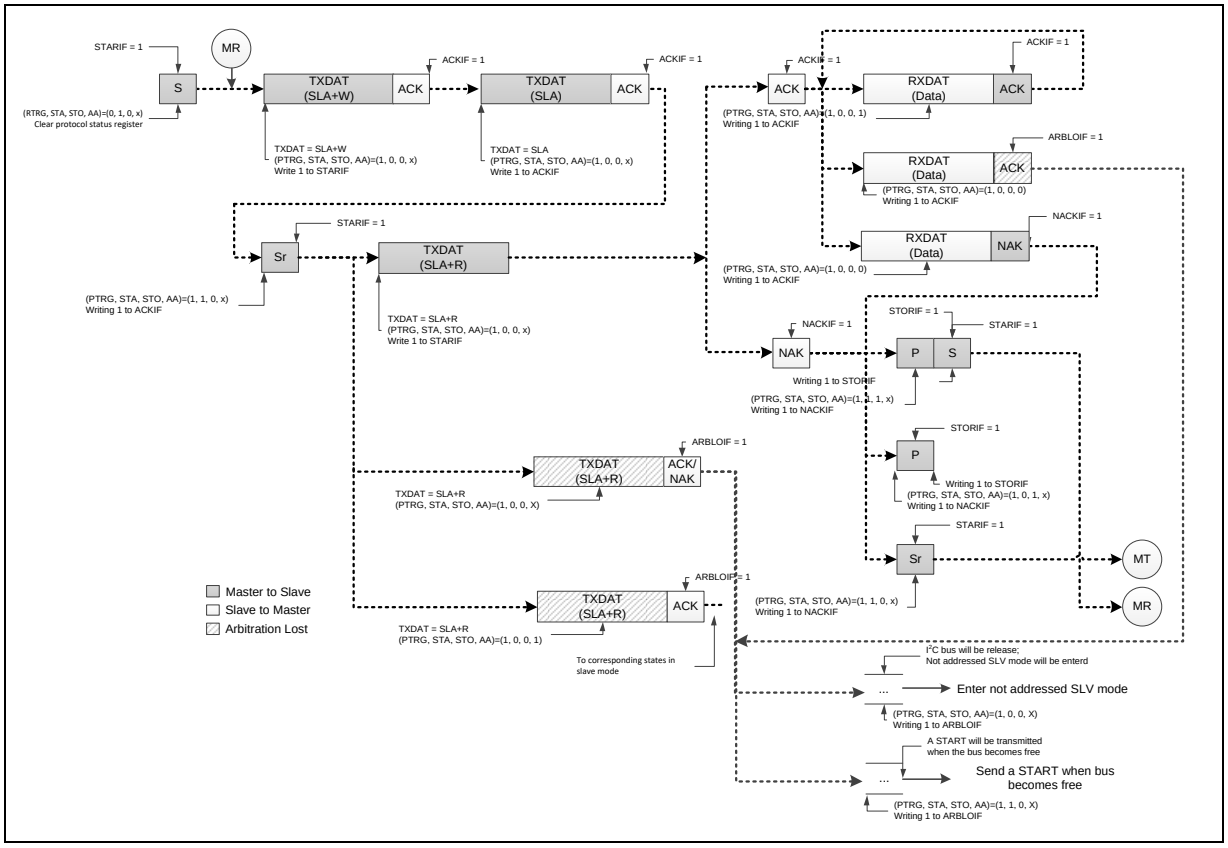


Figure 6.26-16 Master Receiver Mode Control Flow with 10-bit Address

If the I²C is in Master mode and gets arbitration lost, the bit of ARBLOIF (UI2C_PROTSTS [11]) will be set. User may write 1 to ARBLOIF (UI2C_PROTSTS [11]) and set (PTRG, STA, STO, AA) = (1, 1, 0, X) to send START to re-start Master operation when bus become free. Otherwise, user may write 1 to ARBLOIF (UI2C_PROTSTS [11]) and set (PTRG, STA, STO, AA) = (1, 0, 0, X) to release I²C bus and enter not addressed Slave mode.

Slave Mode

When reset, I²C is not addressed and will not recognize the address on I²C bus. User can set device address by UI2C_DEVADDRn and set (PTRG, STA, STO, AA) = (1, 0, 0, 1) to let I²C recognize the address sent by master. Figure 6.26-17 shows all the possible flow for I²C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.26-17 to implement their own I²C protocol.

If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) or SLA+R (Master want to read data from Slave) after arbitration lost, the ARBLOIF will be set to 1.

Note: During I²C communication, the SCL clock will be released when writing '1' to PTRG (UI2C_PROTCTL [5]) in Slave mode.

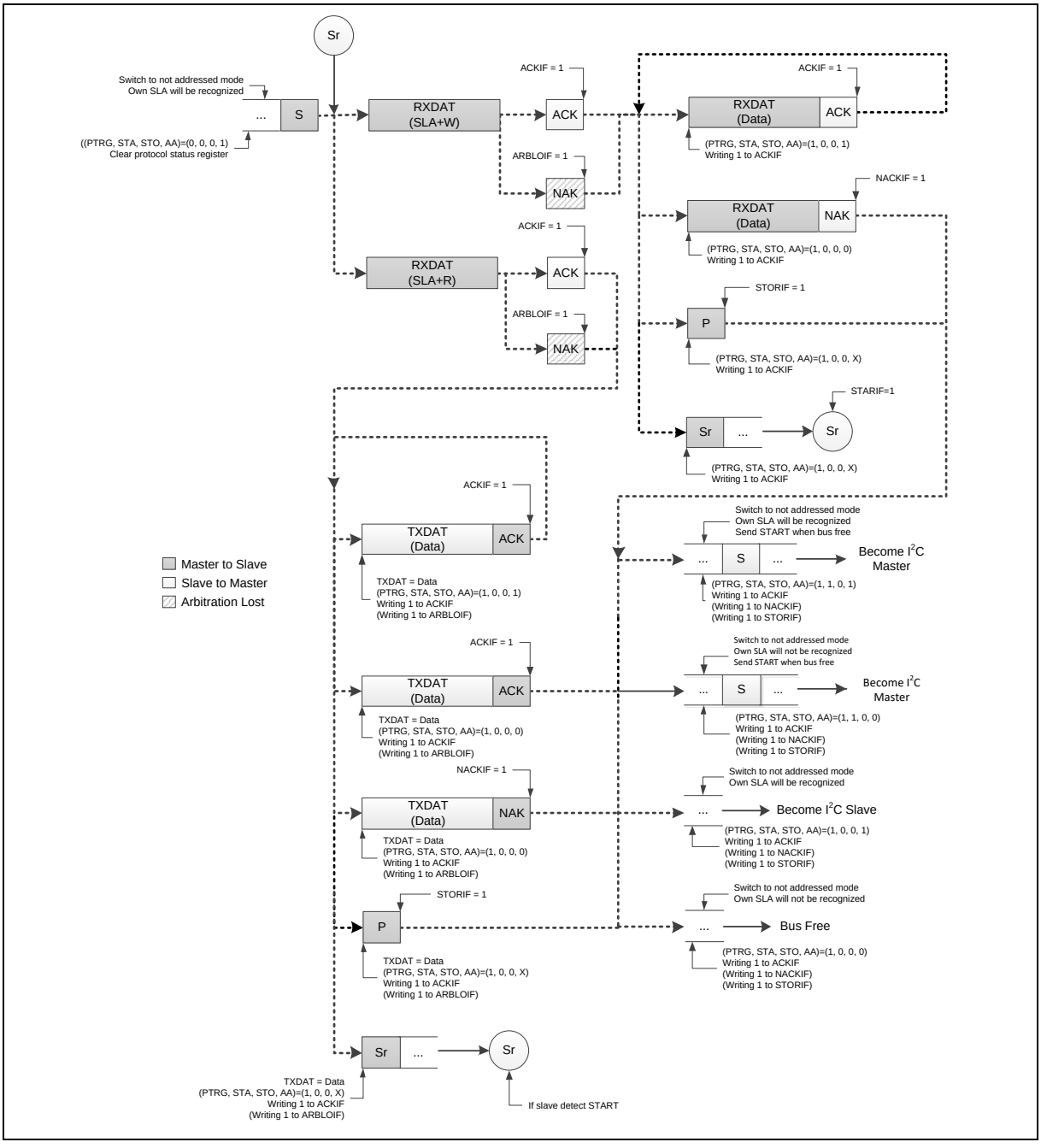


Figure 6.26-17 Save Mode Control Flow with 7-bit Address

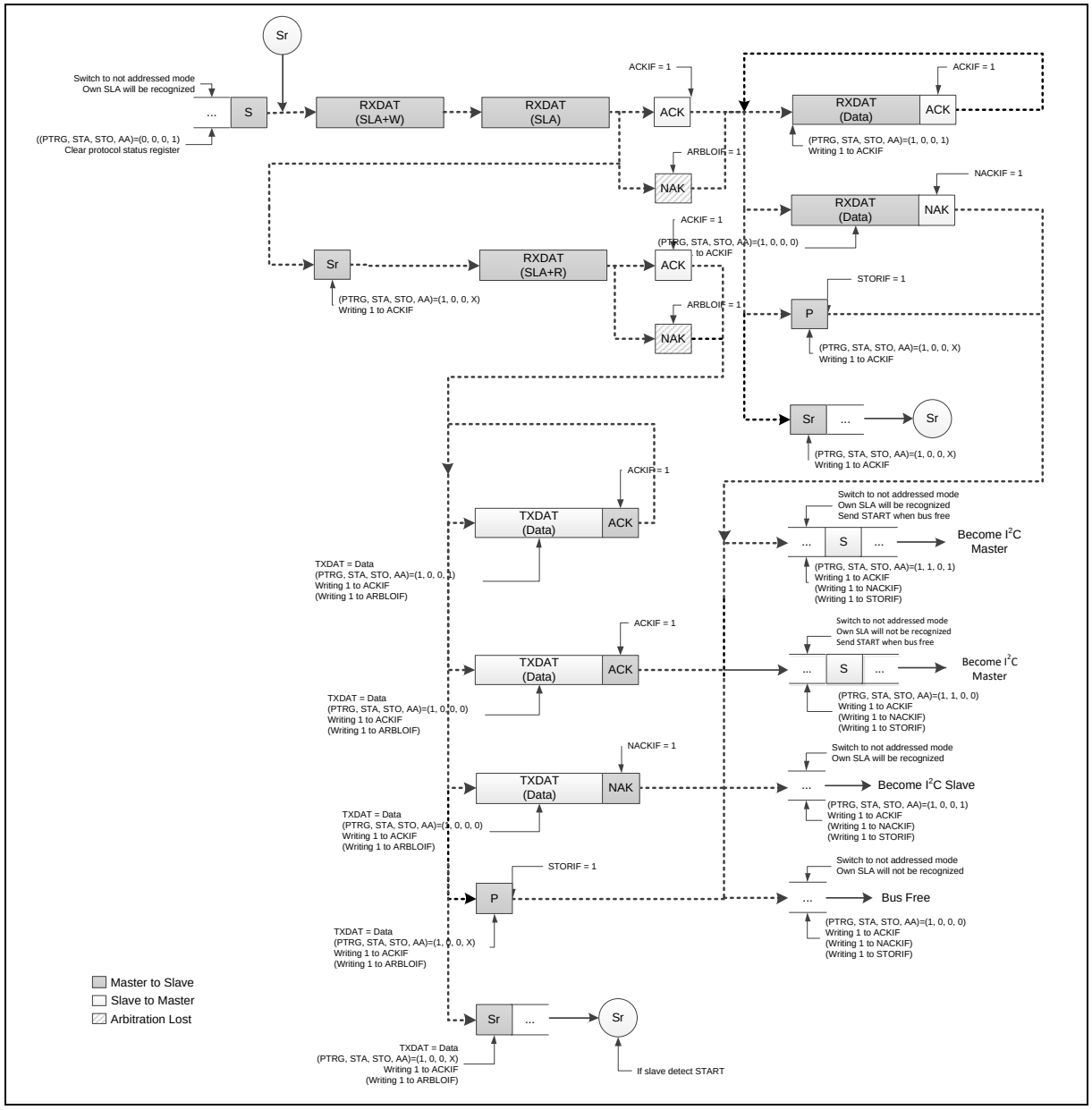


Figure 6.26-18 Save Mode Control Flow with 10-bit Address

If I²C is still transmitting and receiving data in addressed Slave mode but got a STOP or Repeat START, the register STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) will be set. User could follow the action for NACKIF (UI2C_PROTSTS [10]) as shown in the above figure when got STARIF (UI2C_PROTSTS [8]) is set.

Note: After slave gets interrupt flag of NACKIF (UI2C_PROTSTS [10]) and start/stop symbol including STARIF (UI2C_PROTSTS [8]) and STORIF (UI2C_PROTSTS [9]), slave can switch to not address mode and own SLA will not be recognized. If setting this interrupt flag, slave will not receive any I²C signal or address from master. At this status, I²C should be reset by setting FUNMODE (UI2C_CTL [2:0]) = 000B to leave this status.

General Call (GC) Mode

If the GCFUNC bit (UI2C_PROTCTL [0]) is set, the I²C port hardware will respond to General Call address (00H). User can clear GC bit to disable general call function. When the GC bit is set and the

I²C in slave mode, it can receive the general call address by 0x00 after master send general call address to I²C bus, and then it also will follow protocol status register.

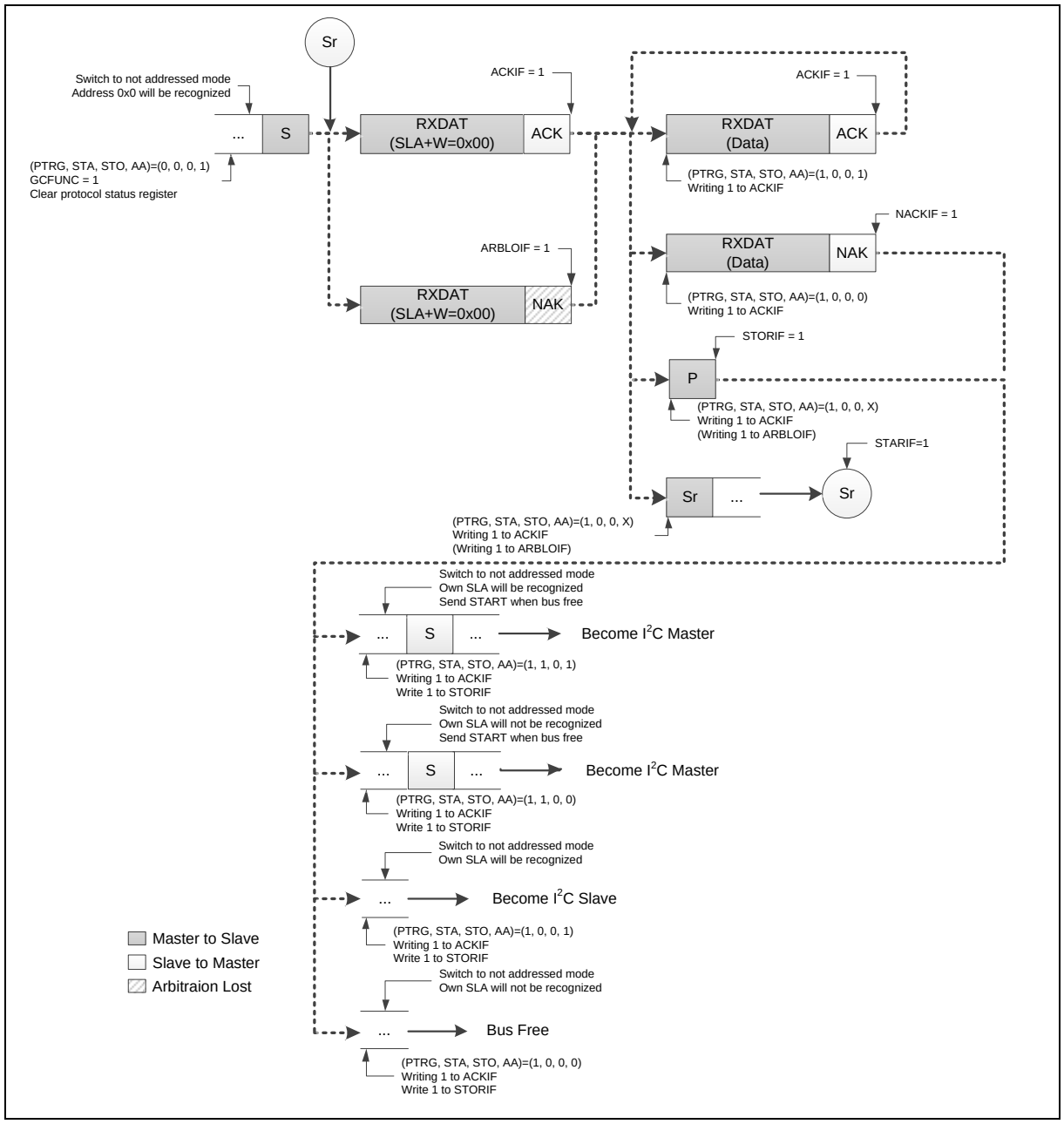


Figure 6.26-19 GC Mode with 7-bit Address

If I²C is still receiving data in GC mode but got a STOP or Repeat START, the STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) will be set. User could follow the action for NACKIF (UI2C_PROTSTS [10]) in above figure when got STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) is set.

Note: After slave gets interrupt flag of NACKIF (UI2C_PROTSTS [10]) and start/stop symbol including STARIF (UI2C_PROTSTS [8]) and STORIF (UI2C_PROTSTS [9]), slave can switch to not address mode and own SLA will not be recognized. If setting this interrupt flag, slave will not receive any I²C signal or address from master. At this time, the I²C controller should be reset by setting FUNMODE (UI2C_CTL [2:0]) = 000B to leave this status.

Protocol Functional Description

Monitor Mode

When I²C enters monitor mode, this device always returns NACK to master after each frame reception even address matching. Moreover, this device will store any receive data including address, command code, and data.

Interrupt in Monitor Mode

All interrupts will occur as normal process when the MONEN (UI2C_PROTCTL [9]) is set. Note that the first interrupt will occur when initial START, it not the same as I²C slave, but the other interrupts are the same.

Subsequent to the address-match detection, interrupts will be generated after each data byte is received as slave mode control flow, or after each byte that the module believes it has transmitted for a slave-read transfer. In this second case, the data register will actually contain data transmitted by some other slave on the bus which was actually addressed by the master. If user wants to watch other device, user can set address mask and monitor.

If the monitor has not had time to respond to interrupt, the SCL signal will be pulled to low when SCLOUTEN (UI2C_PROTCTL [8]) is set to 1. User must set PTRG (UI2C_PROTCTL [5]) to release bus when SCLOUTEN (UI2C_PROTCTL [8]) is set to 1. If SCLOUTEN (UI2C_PROTCTL [8]) is not set to 1, user doesn't need to set PTRG (UI2C_PROTCTL [5]) to 1.

When device address match, but the device response NACK, this address will be received into buffer and NACK interrupt will be generated.

Following all of these interrupts, the processor may read the data register to see what was actually transmitted on the bus.

Loss of Arbitration in Monitor Mode

In monitor mode, the I²C module will not be able to respond to a request for information by the bus master or issue an ACK. Some other slave on the bus will respond instead. Software should be aware of the fact that the module is in monitor mode and should not respond to any loss of arbitration state that is detected.

Programmable Setup and Hold Time

In order to guarantee a correct data setup and hold time, the timing must be configured. By programming HTCTL (UI2C_TMCTL [24:16]) to configure hold time and STCTL (UI2C_TMCTL[8:0]) to configure setup time.

The delay timing refers to peripheral clock (PCLK). When device stretches master clock, the setup and hold time configuration value will not be affected by stretched.

User should focus the limitation of setup and hold time configuration, the timing setting must follow I²C protocol. Once setup time configuration greater than design limitation, that means if setup time setting make SCL output less than three PCLKs, I²C controller can't work normally due to SCL must sample three times. Once hold time configuration is greater than I²C clock limitation, I²C will occur bus error. Suggest that user calculate suitable timing with baud rate and protocol before setting timing. Table 6.26-1 shows the relationship between I²C baud rate and PCLK, the number of table represent one clock duty contain how many PCLKs. Setup and hold time configuration even can program some extreme values in the design, but user should follow I²C protocol standard.

I²C Baud Rate PCLK	100k	200k	400k	800k	1200k
12 MHz	120	60	30	15	10
24 MHz	240	120	60	30	20

48 MHz	480	240	120	60	40
72 MHz	720	360	180	90	60

Table 6.26-1 Relationship between I²C Baud Rate and PCLK

For setup time wrong adjustment example, assuming one SCL cycle contains ten PCLKs and set STCTL (UI2C_TMCTL [8:0]) to 3 that stretch three PCLKs for setup time setting. The setup time setting limitation: ST_{limit} = (UI2C_BRGEN[25:16]+1) - 6.

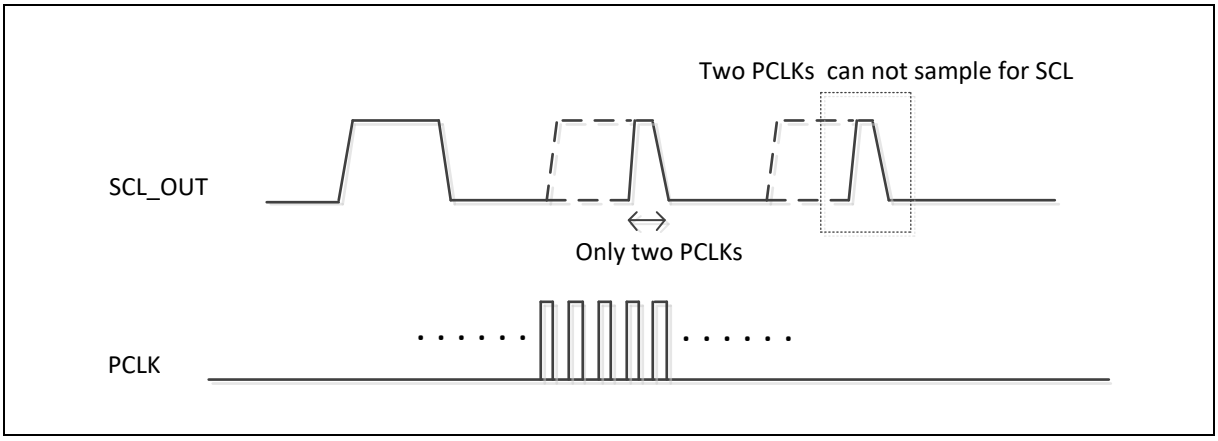


Figure 6.26-20 Setup Time Wrong Adjustment

For hold time wrong adjustment example, use I²C Baud Rate = 1200k and PCLK = 72 MHz, the SCL high/low duty = 60 PCLK. When HTCTL (UI2C_TMCTL [24:16]) is set to 63 and STCTL (UI2C_TMCTL[8:0]) is set to 0, then SDA output delay will over SCL high duty and cause bus error. The hold time setting limitation: HT_{limit} = (UI2C_BRGEN [25:16]+1) - 9.

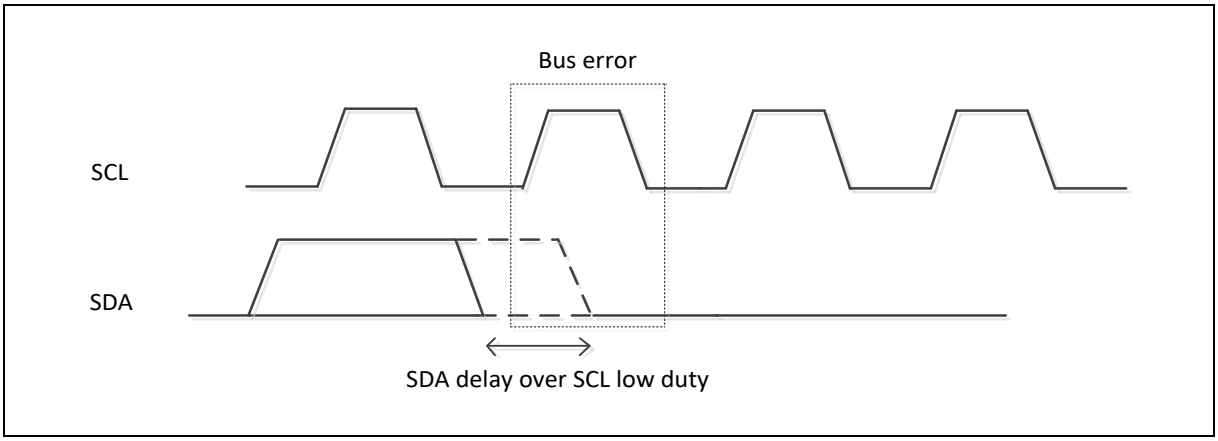


Figure 6.26-21 Hold Time Wrong Adjustment

I²C Time-out Function

There is a 10-bit time-out counter TOCNT (UI2C_PROTCTL [25:16]) which can be used to deal with the I²C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it equals TOCNT (UI2C_PROTCTL [25:16]) and generates I²C interrupt to CPU or stops counting by clearing TOIEN (UI2C_PROTIEN [0]) to 0 or setting all I²C interrupt signal (ACKIF, ERRIF, ARBLOIF, NACKIF, STORIF, STARIF). User may write 1 to clear TOIF(UI2C_PROTSTS [5]) to 0. When time-out counter is enabled, writing 1 to the TOIF will reset counter and re-start up counting after TOIF is cleared. Refer to Figure 6.26-22 for the time-out counter TOCNT (UI2C_PROTCTL [25:16]). T_{TOCNT} = (TOCNT (UI2C_PROTCTL [25:16]) +1) x32 (5-bit) x T_{PCLK}. Note that the time counter clock source TMCNTSRC (UI2C_ BRGEN

[5]) must be set as 0.

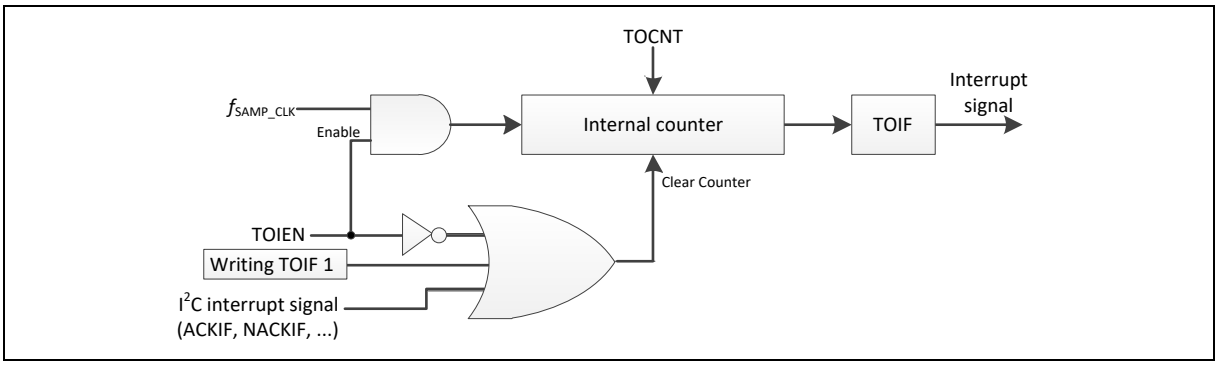


Figure 6.26-22 I²C Time-out Count Block Diagram

Wake-up Function

When chip enters Power-down mode and sets WKEN (WKCTL [0]) to 1, other I²C master can wake up the chip by addressing the I²C device. User must configure the related setting before entering sleep mode. The ACK bit cycle of address match frame is done in power-down. The controller will stretch the SCL to low when the address is matched the device's address and the ACK cycle done. The SCL is stretched until the bit is clear by user. If the frequency of SCL is low speed and the system has wakeup from address match frame, the user shall check this bit to confirm this frame has transaction done and then to do the wake-up procedure. Therefore, when the chip is woken up by address match with one of the device address register (UI2C_DEVADDRn), the user shall check the WKAKDONE (UI2C_PROTSTS [16]) bit is set to 1 to confirm if the address wakeup frame has been done. The WKAKDONE bit indicates that the ACK bit cycle of address match frame is done in power-down. The controller will stretch the SCL to low when the address is matched the device's slave address and the ACK cycle done. The SCL is stretched until the WKAKDONE bit is clear by user. If the frequency of SCL is low speed and the system has wakeup from address match frame, the user shall check this bit to confirm this frame has transaction done and then to do the wake-up procedure. Note that user must clear WKF after clearing the WKAKDONE bit to 0.

The WRSTSWK (UI2C_PROTSTS [17]) bit records the Read/Write command on the address match wake-up frame. The user can use read this bit's status to prepare the next transmitted data (WRSTSWK = 1) or to wait the incoming data (WRSTSWK = 0) can be stored in time after the system is wake-up by the address match frame.

When system is woken up by other I²C master device, WKF (UI2C_WKSTS [0]) is set to indicate this event. User needs write "1" to clear this bit.

Example for Random Read on EEPROM

The following steps are used to configure the USCI0_I2C related registers when using I²C protocol to read data from EEPROM.

1. Set USCI0_I2C the multi-function pin as SCL and SDA pins. Refer to the Basic Configuration section for the multi-function configuration.
2. Enable USCI0 APB clock. Refer to the Basic Configuration section for the multi-function configuration.
3. Set USCI0RST=1 to reset USCI controller then set USCI0RST=0 let USCI controller to normal operation. Refer to the Basic Configuration section for the reset controller configuration.
4. Set FUNMODE =100 to enable USCI0_I2C controller in the "UI2C_CTL" register.
5. Give USCI0_I2C clock a divided register value for USCI0_I2C clock rate in the "UI2C_BRGEN".

6. Enable system I2C0 IRQ in system “NVIC” control register.
7. Set ACKIEN, ERRIEN, ARBLOIEN, NACKIEN, STORIEN, STARIEN, and TOIEN to enable I²C Interrupt in the “UI2C_PROTIEN” register.
8. Set USCI address registers “UI2C_ADDR0 ~ UI2C_ADDR1”.

Random read operation is one of the methods of access EEPROM. The method allows the master to access any address of EEPROM space. Figure 6.26-23 shows the EEPROM random read operation.

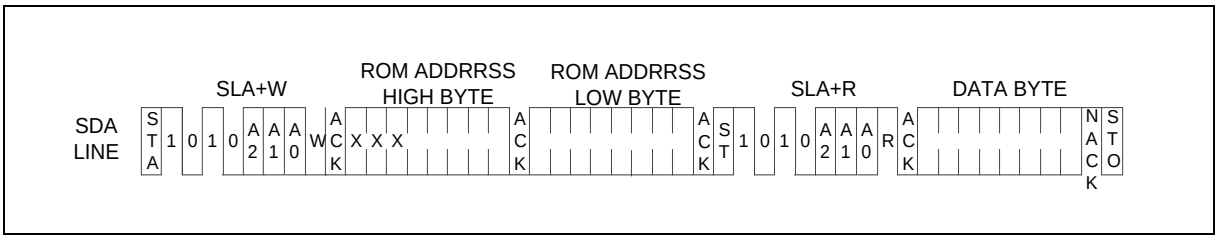


Figure 6.26-23 EEPROM Random Read

Figure 6.26-24 shows how to use I²C controller to implement the protocol of EEPROM random read.

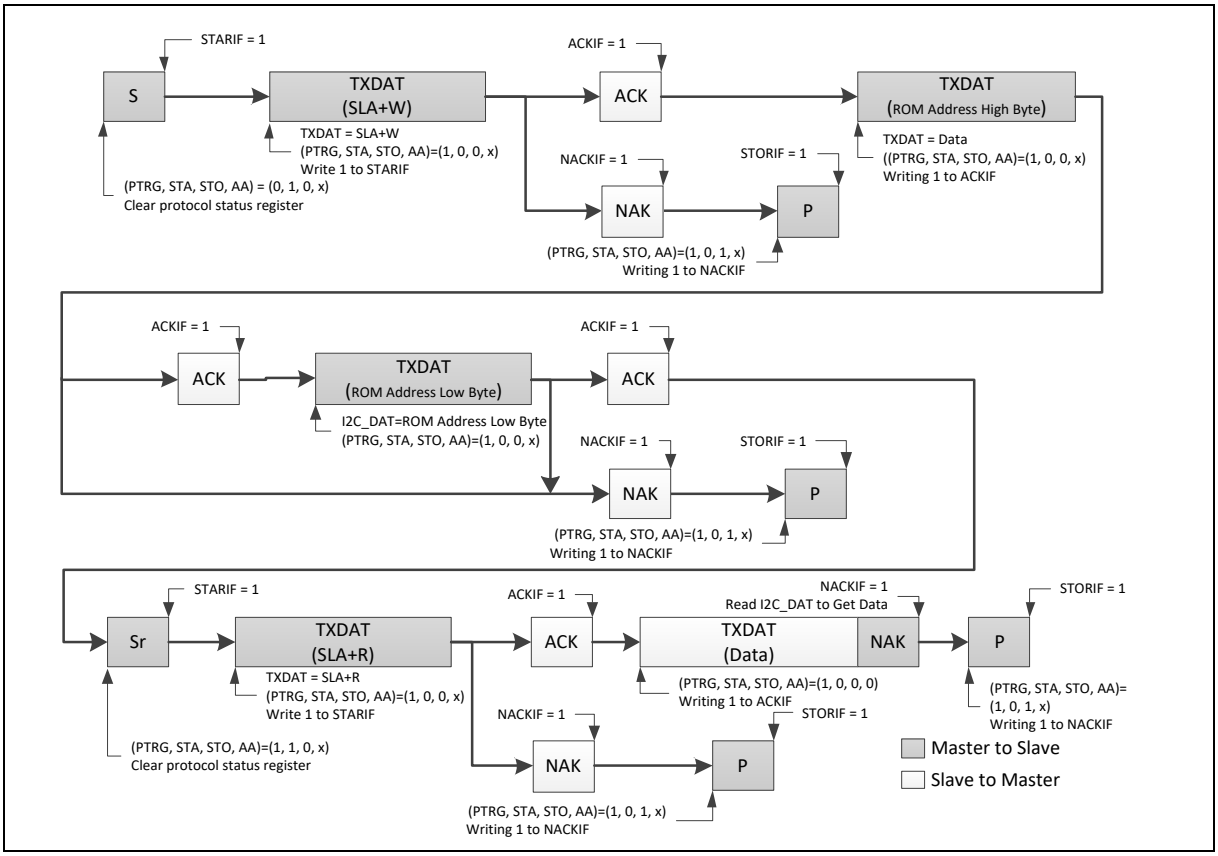


Figure 6.26-24 Protocol of EEPROM Random Read

The I²C controller, which is a master, sends START to bus. Then, it sends a SLA+W (Slave address + Write bit) to EERPOM followed by two bytes data address to set the EEPROM address to read. Finally, a Repeat START followed by SLA+R is sent to read the data from EEPROM.

6.26.6 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
USCI_I2C Base Address: **UI2Cn_BA = 0x400D_0000 + (0x1000 * n)** **n= 0,1**				
UI2Cn_CTL	UI2Cn_BA+0x00	R/W	USCI Control Register	0x0000_0000
UI2Cn_BRGEN	UI2Cn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
UI2Cn_LINECTL	UI2Cn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
UI2Cn_TXDAT	UI2Cn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
UI2Cn_RXDAT	UI2Cn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
UI2Cn_DEVADDR0	UI2Cn_BA+0x44	R/W	USCI Device Address Register 0	0x0000_0000
UI2Cn_DEVADDR1	UI2Cn_BA+0x48	R/W	USCI Device Address Register 1	0x0000_0000
UI2Cn_ADDRMSK0	UI2Cn_BA+0x4C	R/W	USCI Device Address Mask Register 0	0x0000_0000
UI2Cn_ADDRMSK1	UI2Cn_BA+0x50	R/W	USCI Device Address Mask Register 1	0x0000_0000
UI2Cn_WKCTL	UI2Cn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
UI2Cn_WKSTS	UI2Cn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
UI2Cn_PROTCTL	UI2Cn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000
UI2Cn_PROTIEN	UI2Cn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
UI2Cn_PROTSTS	UI2Cn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000
UI2Cn_TMCTL	UI2Cn_BA+0x8C	R/W	I^2C Timing Configure Control Register	0x0002_0000

6.26.7 Register Description

USCI Control Register (UI2C_CTL)

Register	Offset	R/W	Description	Reset Value
UI2Cn_CTL	UI2Cn_BA+0x00	R/W	USCI Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**FUNMODE**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 000 to FUNMODE. 000 = The USCI is disabled. All protocol related state machines are set to idle state. 001 = The SPI protocol is selected. 010 = The UART protocol is selected. 100 = The I^2C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Baud Rate Generator Register (UI2C_BRGEN)

Register	Offset	R/W	Description	Reset Value
UI2Cn_BRGEN	UI2Cn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

31	30	29	28	27	26	25	24
NFCNT				Reserved		CLKDIV	
23	22	21	20	19	18	17	16
CLKDIV							
15	14	13	12	11	10	9	8
Reserved	DSCNT					PDSCNT	
7	6	5	4	3	2	1	0
Reserved		TMCNTSRC	TMCNTEN	SPCLKSEL		PTCLKSEL	RCLKSEL

Bits	Description	
[31:28]	**NFCNT**	**Noise Filter Count** The register bits control the input filter width. 0 = Filter width 3*PCLK. 1 = Filter width 4*PCLK. *N* = filter width (3+*N*)*PCKL. **Note:** Filter width Min = 3*PCLK, Max = 18*PCLK.
[27:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} ($f_{DIV_CLK} = f_{PROT_CLK} / (CLKDIV+1)$). **Note:** The minimum value of DIVIDER is 9 for USCI_I2C.
[15]	**Reserved**	Reserved.
[14:10]	**DSCNT**	**Denominator for Sample Counter** This bit field defines the divide ratio of the sample clock f_{SAMP_CLK}. The divided frequency $f_{DS_CNT} = f_{PDS_CNT} / (DSCNT+1)$. **Note:** The maximum value of DSCNT is 0xF on UART mode and suggest to set over 4 to confirm the receiver data is sampled in right value.
[9:8]	**PDSCNT**	**Pre-divider for Sample Counter** This bit field defines the divide ratio of the clock division from sample clock f_{SAMP_CLK}. The divided frequency $f_{PDS_CNT} = f_{SAMP_CLK} / (PDSCNT+1)$.
[7:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Time Measurement Counter Clock Source Selection** 0 = Time measurement counter with f_{PROT_CLK}. 1 = Time measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Time Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Time measurement counter is Disabled.

		1 = Time measurement counter is Enabled.
[3:2]	**SPCLKSEL**	**Sample Clock Source Selection** This bit field used for the clock source selection of a sample clock (f_{SAMP_CLK}) for the protocol processor. 00 = $f_{SAMP_CLK} = f_{DIV_CLK}$. 01 = $f_{SAMP_CLK} = f_{PROT_CLK}$. 10 = $f_{SAMP_CLK} = f_{SCLK}$. 11 = $f_{SAMP_CLK} = f_{REF_CLK}$.
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source signal of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source signal of reference clock (f_{REF_CLK}). 0 = Peripheral device clock f_{PCLK}. 1 = Reserved.

USCI Line Control Register (UI2C_LINECTL)

Register	Offset	R/W	Description	Reset Value
UI2Cn_LINECTL	UI2Cn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**DWIDTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**LSB**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI supports word length from 4 to 16 bits. 0x0 = The data word contains 16 bits located at bit positions [15:0]. 0x1 = Reserved. 0x2 = Reserved. 0x3 = Reserved. 0x4 = The data word contains 4 bits located at bit positions [3:0]. 0x5 = The data word contains 5 bits located at bit positions [4:0]. ... 0Xf = The data word contains 15 bits located at bit positions [14:0].
[7:1]	**Reserved**	Reserved.
[0]	**LSB**	**LSB First Transmission Selection** 0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (UI2C_TXDAT)

Register	Offset	R/W	Description	Reset Value
UI2Cn_TXDAT	UI2Cn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission.

USCI Receive Data Register (UI2C_RXDAT)

Register	Offset	R/W	Description	Reset Value
UI2Cn_RXDAT	UI2Cn_BA+0x34	R	USCI Receive Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer. **Note:** In I^2C protocol, RXDAT[12:8] indicate the different transmission conditions which defined in I^2C.

USCI Device Address Register (UI2C_DEVADDR)

Register	Offset	R/W	Description	Reset Value
UI2Cn_DEVADDR0	UI2Cn_BA+0x44	R/W	USCI Device Address Register 0	0x0000_0000
UI2Cn_DEVADDR1	UI2Cn_BA+0x48	R/W	USCI Device Address Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						DEVADDR	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DEVADDR							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**DEVADDR**	**Device Address** In I^2C protocol, this bit field contains the programmed slave address. If the first received address byte are 1111 $0AAX_B$, the AA bits are compared to the bits DEVADDR [9:8] to check for address match, where the X is R/W bit. Then the second address byte is also compared to DEVADDR [7:0]. **Note 1:** The DEVADDR [9:7] must be set 3'b000 when I^2C is operating in 7-bit address mode. **Note 2:** When software sets 10'h000, the address cannot be used.

USCI Device Address Mask Register (UI2C_ADDRMSK) – for I²C Only

Register	Offset	R/W	Description	Reset Value
UI2Cn_ADDRMSK0	UI2Cn_BA+0x4C	R/W	USCI Device Address Mask Register 0	0x0000_0000
UI2Cn_ADDRMSK1	UI2Cn_BA+0x50	R/W	USCI Device Address Mask Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**ADDRMSK**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDRMSK							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**ADDRMSK**	**USCI Device Address Mask** 0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don't care.). USCI supports multiple address recognition with two address mask register. When the bit in the address mask register is set to 1, it means the received corresponding address bit is don't-care. If the bit is set to 0, that means the received corresponding register bit should be exactly the same as address register. **Note:** The wake-up function cannot use address mask in 10-bit mode

USCI Wake-up Control Register (UI2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
UI2Cn_WKCTL	UI2Cn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**WKADDREN**	**WKEN**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WKADDREN**	**Wake-up Address Match Enable Bit** 0 = The chip is woken up according to START signal. 1 = The chip is woken up according to address match.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (UI2C_WKSTS)

Register	Offset	R/W	Description	Reset Value
UI2Cn_WKSTS	UI2Cn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**WKF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – I²C (UI2C_PROTCTL)

Register	Offset	R/W	Description	Reset Value
UI2Cn_PROTCTL	UI2Cn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	Reserved					TOCNT	
23	22	21	20	19	18	17	16
TOCNT							
15	14	13	12	11	10	9	8
Reserved						MONEN	SCLOUTEN
7	6	5	4	3	2	1	0
Reserved		PTRG	ADDR10EN	STA	STO	AA	GCFUNC

Bits	Description	
[31]	**PROTEN**	**I²C Protocol Enable Bit** 0 = I²C Protocol Disabled. 1 = I²C Protocol Enabled.
[30:26]	**Reserved**	Reserved.
[25:16]	**TOCNT**	**Time-out Clock Cycle** This bit field indicates how many clock cycle selected by TMCNTSRC (UI2C_BRGEN [5]) when each interrupt flags are clear. The time-out is enable when TOCNT bigger than 0. **Note:** The TMCNTSRC (UI2C_BRGEN [5]) must be set zero on I²C mode.
[15:10]	**Reserved**	Reserved.
[9]	**MONEN**	**Monitor Mode Enable Bit** This bit enables monitor mode. In monitor mode the SDA output will be put in high impedance mode. This prevents the I²C module from outputting data of any kind (including ACK) onto the I²C data bus. 0 = The monitor mode Disabled. 1 = The monitor mode Enabled. **Note:** Depending on the state of the SCLOUTEN bit, the SCL output may be also forced high, preventing the module from having control over the I²C clock line.
[8]	**SCLOUTEN**	**SCL Output Enable Bit** This bit enables monitor pulling SCL to low. This monitor will pull SCL to low until it has had time to respond to an I²C interrupt. 0 = SCL output will be forced high due to open drain mechanism. 1 = I²C module may act as a slave peripheral just like in normal operation, the I²C holds the clock line low until it has had time to clear I²C interrupt.
[7:6]	**Reserved**	Reserved.

[5]	**PTRG**	**I²C Protocol Trigger (Write Only)** When a new state is present in the UI2C_PROTSTS register, if the related interrupt enable bits are set, the I²C interrupt is requested. It must write one by software to this bit after the related interrupt flags are set to 1 and the I²C protocol function will go ahead until the STOP is active or the PROTEN is disabled. 0 = I2C's stretch disabled and the I²C protocol function will go ahead. 1 = I2C's stretch is active.
[4]	**ADDR10EN**	**Address 10-bit Function Enable Bit** 0 = Address match 10 bit function Disabled. 1 = Address match 10 bit function Enabled.
[3]	**STA**	**I²C START Control** Setting STA to logic 1 to enter Master mode, the I²C hardware sends a START or repeat START condition to bus when the bus is free. This bit will be cleared by hardware automatically.
[2]	**STO**	**I²C STOP Control** In Master mode, setting STO to transmit a STOP condition to bus then I²C hardware will check the bus condition if a STOP condition is detected this bit will be cleared by hardware automatically. In a slave mode, setting STO resets I²C hardware to the defined "not addressed" slave mode when bus error (UI2C_PROTSTS.ERRIF = 1).
[1]	**AA**	**Assert Acknowledge Control** When AA =1 prior to address or data received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.). A slave is acknowledging the address sent from master, 2.) The receiver devices are acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line.
[0]	**GCFUNC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled. **Note:** When ADDR10EN (UI2C_PROTCTL [4]) is set, don't set this bit.

USCI Protocol Interrupt Enable Register – I²C (UI2C_PROTIEN)

Register	Offset	R/W	Description	Reset Value
UI2Cn_PROTIEN	UI2Cn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	ACKIEN	ERRIEN	ARBLOIEN	NACKIEN	STORIEN	STARIEN	TOIEN

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**ACKIEN**	**Acknowledge Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if an acknowledge is detected by a master. 0 = The acknowledge interrupt Disabled. 1 = The acknowledge interrupt Enabled.
[5]	**ERRIEN**	**Error Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if an I²C error condition is detected (indicated by ERRIF (UI2C_PROTSTS [12])). 0 = The error interrupt Disabled. 1 = The error interrupt Enabled.
[4]	**ARBLOIEN**	**Arbitration Lost Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if an arbitration lost event is detected. 0 = The arbitration lost interrupt Disabled. 1 = The arbitration lost interrupt Enabled.
[3]	**NACKIEN**	**Non - Acknowledge Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if a Non - acknowledge is detected by a master. 0 = The non-acknowledge interrupt Disabled. 1 = The non-acknowledge interrupt Enabled.
[2]	**STORIEN**	**STOP Condition Received Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if a STOP condition is detected. 0 = The stop condition interrupt Disabled. 1 = The stop condition interrupt Enabled.
[1]	**STARIEN**	**START Condition Received Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if a START condition is detected. 0 = The start condition interrupt Disabled.

		1 = The start condition interrupt Enabled.
[0]	**TOIEN**	**Time-out Interrupt Enable Bit** In I^2C protocol, this bit enables the interrupt generation in case of a time-out event. 0 = The time-out interrupt Disabled. 1 = The time-out interrupt Enabled.

USCI Protocol Status Register – I²C (UI2C_PROTSTS)

Register	Offset	R/W	Description	Reset Value
UI2Cn_PROTSTS	UI2Cn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				ERRARBLO	BUSHANG	WRSTSWK	WKAKDONE
15	14	13	12	11	10	9	8
SLAREAD	SLASEL	ACKIF	ERRIF	ARBLOIF	NACKIF	STORIF	STARIF
7	6	5	4	3	2	1	0
Reserved	ONBUSY	TOIF	Reserved				

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19]	**ERRARBLO**	**Error Arbitration Lost (Read Only)** This bit indicates bus arbitration lost due to bigger noise which is can't be filtered by input processor. The I²C can send start condition when ERRARBLO is set. Thus this bit doesn't be cared on slave mode. 0 = The bus is normal status for transmission. 1 = The bus is error arbitration lost status for transmission. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware when a START condition is present.
[18]	**BUSHANG**	**Bus Hang-up (Read Only)** This bit indicates bus hang-up status. There is 4-bit counter count when SCL hold high and refer f_{SAMP_CLK}. The hang-up counter will count to overflow and set this bit when SDA is low. The counter will be reset by falling edge of SCL signal. 0 = The bus is normal status for transmission. 1 = The bus is hang-up status for transmission. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware when a START condition is present.
[17]	**WRSTSWK**	**Read/Write Status Bit in Address Wake-up Frame (Read Only)** 0 = Write command be record on the address match wake-up frame. 1 = Read command be record on the address match wake-up frame.
[16]	**WKAKDONE**	**Wake-up Address Frame Acknowledge Bit Done (Read Only)** 0 = The ACK bit cycle of address match frame isn't done. 1 = The ACK bit cycle of address match frame is done in power-down. **Note:** This bit cannot be released when WKF is set.
[15]	**SLAREAD**	**Slave Read Request Status (Read Only)** This bit indicates that a slave read request has been detected. 0 = A slave R/W bit is 1 has not been detected. 1 = A slave R/W bit is 1 has been detected. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware.

[14]	**SLASEL**	**Slave Select Status (Read Only)** This bit indicates that this device has been selected as slave. 0 = The device is not selected as slave. 1 = The device is selected as slave. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware.
[13]	**ACKIF**	**Acknowledge Received Interrupt Flag** This bit indicates that an acknowledge has been received in master mode. A protocol interrupt can be generated if UI2C_PROTCTL.ACKIEN = 1. 0 = An acknowledge has not been received. 1 = An acknowledge has been received. **Note:** This bit is cleared by software writing 1 into this bit
[12]	**ERRIF**	**Error Interrupt Flag** This bit indicates that a Bus Error occurs when a START or STOP condition is present at an illegal position in the formation frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit. A protocol interrupt can be generated if UI2C_PROTCTL.ERRIEN = 1. 0 = An I^2C error has not been detected. 1 = An I^2C error has been detected. **Note 1:** This bit is cleared by software writing 1 into this bit. **Note 2:** This bit is set for slave mode, and user must write 1 into STO register to the defined “not addressed” slave mode.
[11]	**ARBLOIF**	**Arbitration Lost Interrupt Flag** This bit indicates that an arbitration has been lost. A protocol interrupt can be generated if UI2C_PROTCTL.ARBLOIEN = 1. 0 = An arbitration has not been lost. 1 = An arbitration has been lost. **Note:** This bit is cleared by software writing 1 into this bit.
[10]	**NACKIF**	**Non - Acknowledge Received Interrupt Flag** This bit indicates that a non - acknowledge has been received in master mode. A protocol interrupt can be generated if UI2C_PROTCTL.NACKIEN = 1. 0 = A non-acknowledge has not been received. 1 = A non-acknowledge has been received. **Note:** This bit is cleared by software writing 1 into this bit.
[9]	**STORIF**	**Stop Condition Received Interrupt Flag** This bit indicates that a stop condition has been detected on the I^2C bus lines. A protocol interrupt can be generated if UI2C_PROTCTL.STORIEN = 1. 0 = A stop condition has not yet been detected. 1 = A stop condition has been detected. **Note:** This bit is cleared by software writing 1 into this bit.
[8]	**STARIF**	**Start Condition Received Interrupt Flag** This bit indicates that a start condition or repeated start condition has been detected on master mode. However, this bit also indicates that a repeated start condition has been detected on slave mode. A protocol interrupt can be generated if UI2C_PROTCTL.STARIEN = 1. 0 = A start condition has not yet been detected. 1 = A start condition has been detected. **Note:** This bit is cleared by software writing 1 into this bit.
[7]	**Reserved**	Reserved.
[6]	**ONBUSY**	**On Bus Busy (Read Only)**

		Indicates that a communication is in progress on the bus. It is set by hardware when a START condition is detected. It is cleared by hardware when a STOP condition is detected 0 = The bus is IDLE (both SCLK and SDA High). 1 = The bus is busy. **Note:** This bit is read only.
[5]	**TOIF**	**Time-out Interrupt Flag** 0 = A time-out interrupt status has not occurred. 1 = A time-out interrupt status has occurred. **Note:** This bit is cleared by software writing 1 into this bit.
[4:0]	**Reserved**	Reserved.

USCI Timing Configure Control Register (UI2C_TMCTL)

Register	Offset	R/W	Description	Reset Value
UI2Cn_TMCTL	UI2Cn_BA+0x8C	R/W	I^2C Timing Configure Control Register	0x0002_0000

31	30	29	28	27	26	25	24
Reserved							**HTCTL**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HTCTL							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**STCTL**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STCTL							

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24:16]	**HTCTL**	**Hold Time Configure Control** This field is used to generate the delay timing between SCL falling edge SDA edge in transmission mode. The delay hold time is numbers of peripheral clock = HTCTL x f_{PCLK}.
[15:9]	**Reserved**	Reserved.
[8:0]	**STCTL**	**Setup Time Configure Control** This field is used to generate a delay timing between SDA edge and SCL rising edge in transmission mode.. The delay setup time is numbers of peripheral clock = STCTL x f_{PCLK}.

6.27 UART Interface Controller (UART)

6.27.1 Overview

The chip provides eight channels of Universal Asynchronous Receiver/Transmitters (UART). The UART controller performs serial-to-parallel conversion on data received from the peripheral and parallel-to-serial conversion on data transmitted from the CPU. Each UART controller channel supports eleven types of interrupts. The UART controller supports flow control function. The UART controller also supports IrDA SIR, LIN, RS-485 and Single-wire function modes and auto-baud rate measuring function.

6.27.2 Features

- Full-duplex asynchronous communications
- Separates receive and transmit 16/16 bytes entry FIFO for data payloads
- Supports hardware auto-flow control
- Programmable receiver buffer trigger level
- Supports programmable baud rate generator for each channel individually
- Supports nCTS, incoming data, Received Data FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function
- Supports 8-bit receiver buffer time-out detection function
- Programmable transmitting data delay time between the last stop and the next START bit by setting DLY (UART_TOUT[15:8])
- Supports Auto-Baud Rate measurement and baud rate compensation function
  - 9600 bps for UART_CLK is selected LXT.
- Supports break error, frame error, parity error and receive/transmit buffer overflow detection function
- Fully programmable serial-interface characteristics
  - Programmable number of data bit, 5-, 6-, 7-, 8- bit character
  - Programmable PARITY bit, even, odd, no parity or stick PARITY bit generation and detection
  - Programmable STOP bit, 1, 1.5, or 2 STOP bit generation
- Supports IrDA SIR function mode
  - 3/16 bit duration for normal mode
- Supports LIN function mode (Only UART0/UART1 with LIN function)
  - LIN master/slave mode
  - Programmable break generation function for transmitter
  - Break detection function for receiver
- Supports RS-485 function mode
  - RS-485 9-bit mode
  - Hardware or software enables to program nRTS pin to control RS-485 transmission direction
- Supports PDMA transfer function
- Supports Single-wire function mode.

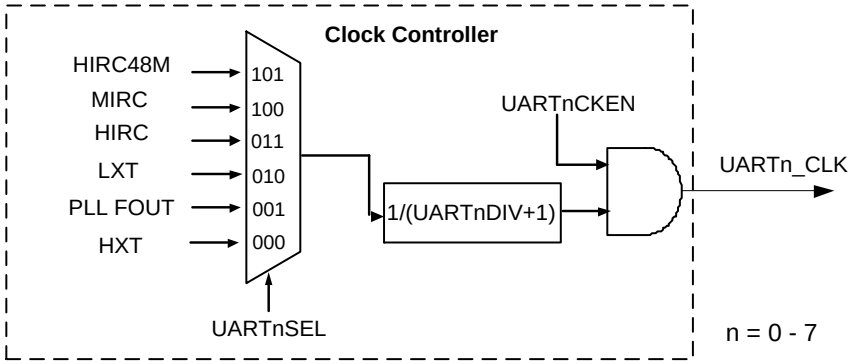
UART Feature	UART0/ UART1	UART2 ~ UART7	USCI-UART
FIFO	16 Bytes	16 Bytes	TX: 1byte RX: 2byte
Auto Flow Control (CTS/RTS)	√	√	√
IrDA	√	√	-
LIN	√	-	-
RS-485 Function Mode	√	√	√
nCTS Wake-up	√	√	√
Incoming Data Wake-up	√	√	√
Received Data FIFO reached threshold Wake-up	√	√	-
RS-485 Address Match (AAD mode) Wake-up	√	√	-
Received Data FIFO reached threshold Time-out Wake-up	√	√	-
Baud Rate Compensation	√	√	-
Auto-Baud Rate Measurement	√	√	√
STOP bit Length	1, 1.5, 2 bit	1, 1.5, 2 bit	1, 2 bit
Word Length	5, 6, 7, 8 bits	5, 6, 7, 8 bits	6~13 bits
Even / Odd Parity	√	√	√
Stick Bit	√	√	-

Table 6.27-1 M2L31 Series UART Features

6.27.3 Block Diagram

The UART clock control and block diagram are shown in Figure 6.28-1 and Figure 6.28-2 respectively.

Note: The frequency of UARTx_CLK should not be greater than 30 times HCLK.



UART0SEL	UART0DIV	UART0CKEN
CLK_CLKSEL4[2:0]	CLK_CLKDIV0[11:8]	CLK_APBCLK0[16]

UART1SEL	UART1DIV	UART1CKEN
CLK_CLKSEL4[6:4]	CLK_CLKDIV0[15:12]	CLK_APBCLK0[17]

UART2SEL	UART2DIV	UART2CKEN
CLK_CLKSEL4[10:8]	CLK_CLKDIV4[3:0]	CLK_APBCLK0[18]

UART3SEL	UART3DIV	UART3CKEN
CLK_CLKSEL4[14:12]	CLK_CLKDIV4[7:4]	CLK_APBCLK0[19]

UART4SEL	UART4DIV	UART4CKEN
CLK_CLKSEL4[18:16]	CLK_CLKDIV4[11:8]	CLK_APBCLK0[20]

UART5SEL	UART5DIV	UART5CKEN
CLK_CLKSEL4[22:20]	CLK_CLKDIV4[15:12]	CLK_APBCLK0[21]

UART6SEL	UART6DIV	UART6CKEN
CLK_CLKSEL4[26:24]	CLK_CLKDIV4[19:16]	CLK_APBCLK0[22]

UART7SEL	UART7DIV	UART7CKEN
CLK_CLKSEL4[30:28]	CLK_CLKDIV4[23:20]	CLK_APBCLK0[23]

Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 6.27-1 UART Clock Control Diagram

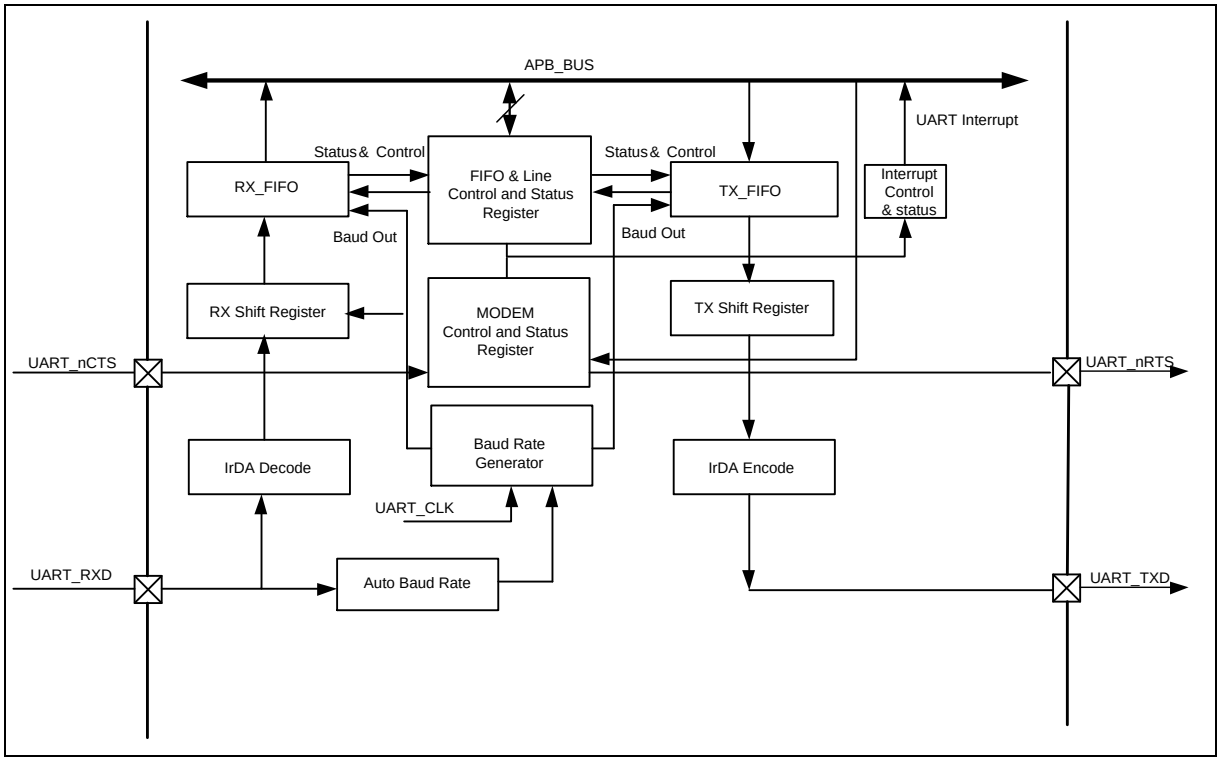


Figure 6.27-2 UART Block Diagram

Each block is described in detail as follows:

TX_FIFO

The transmitter is buffered with a 16 bytes FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO

The receiver is buffered with a 16 bytes FIFO (plus three error bits, BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]), PEF (UART_FIFOSTS[4])) to reduce the number of interrupts presented to the CPU.

TX Shift Register

This block is responsible for shifting out the transmitting data serially.

RX Shift Register

This block is responsible for shifting in the receiving data serially.

Modem Control and Status Register

This register controls the interface to the MODEM or data set (or a peripheral device emulating a MODEM).

Baud Rate Generator

Divide the external clock by the divisor to get the desired baud rate clock. Refer to baud rate equation.

IrDA Encode

This block is IrDA encoding control block.

IrDA Decode

This block is IrDA decoding control block.

FIFO & Line Control and Status Register

This field is register set that including the FIFO control register (UART_FIFO), FIFO status register (UART_FIFOSTS), and line control register (UART_LINE) for transmitter and receiver. The time-out register (UART_TOUT) identifies the condition of time-out interrupt.

Auto-Baud Rate Measurement

This block is responsible for auto-baud rate measurement.

Interrupt Control and Status Register

There are eleven types of interrupts. Interrupt enable register (UART_INTEN) enable or disable the responding interrupt and interrupt status register (UART_INTSTS) identifying the occurrence of the responding interrupt.

Interrupt	Description
RDAINT	Receive Data Available Interrupt.
THREINT	Transmit Holding Register Empty Interrupt.
TXENDINT	Transmitter Empty Interrupt.
RLSINT	Receive Line Status Interrupt (parity error or frame error or break error).
MODEMINT	MODEM Status Interrupt.
RXTOINT	Receiver Buffer Time-out Interrupt.
BUFERRINT	Buffer Error Interrupt.
LININT	LIN Bus Interrupt.
WKINT	Wake-up Interrupt.
ABRINT	Auto-Baud Rate Interrupt.
SWBEINT	Single-wire Bit Error Detect Interrupt.

Table 6.27-2 UART Interrupt

6.27.4 Basic Configuration

6.27.4.1 UART0 Basic Configuration

- Clock Source Configuration
  - Select the source of UART0 peripheral clock on UART0SEL (CLK_CLKSEL4[2:0]).
  - Select the clock divider number of UART0 peripheral clock on UART0DIV (CLK_CLKDIV0[11:8]).
  - Enable UART0 peripheral clock in UART0CKEN (CLK_APBCLK0[16]).
- Reset UART0 controller in UART0RST (SYS_IPRST1[16]).

6.27.4.2 UART1 Basic Configuration

- Clock Source Configuration
  - Select the source of UART1 peripheral clock on UART1SEL (CLK_CLKSEL4[6:4]).
  - Select the clock divider number of UART1 peripheral clock on UART1DIV (CLK_CLKDIV0[15:12]).
  - Enable UART1 peripheral clock in UART1CKEN (CLK_APBCLK0[17]).

- Reset UART1 controller in UART1RST (SYS_IPRST1[17]).

6.27.4.3 UART2 Basic Configuration

- Clock Source Configuration
  - Select the source of UART2 peripheral clock on UART2SEL (CLK_CLKSEL4[10:8]).
  - Select clock divider number of UART2 peripheral clock on UART2DIV (CLK_CLKDIV4[3:0]).
  - Enable UART2 peripheral clock in UART2CKEN (CLK_APBCLK0[18]).
- Reset UART2 controller in UART2RST (SYS_IPRST1[18]).

6.27.4.4 UART3 Basic Configuration

- Clock Source Configuration
  - Select the source of UART3 peripheral clock on UART3SEL (CLK_CLKSEL4[14:12]).
  - Select the clock divider number of UART3 peripheral clock on UART3DIV (CLK_CLKDIV4[7:4]).
  - Enable UART3 peripheral clock in UART3CKEN (CLK_APBCLK0[19]).
- Reset UART3 controller in UART3RST (SYS_IPRST1[19]).

6.27.4.5 UART4 Basic Configuration

- Clock Source Configuration
  - Select the source of UART4 peripheral clock on UART4SEL (CLK_CLKSEL4[18:16]).
  - Select the clock divider number of UART4 peripheral clock on UART4DIV (CLK_CLKDIV4[11:8]).
  - Enable UART4 peripheral clock in UART4CKEN (CLK_APBCLK0[20]).
- Reset UART4 controller in UART4RST (SYS_IPRST1[20]).

6.27.4.6 UART5 Basic Configuration

- Clock Source Configuration
  - Select the source of UART5 peripheral clock on UART5SEL (CLK_CLKSEL4[22:20]).
  - Select the clock divider number of UART5 peripheral clock on UART5DIV (CLK_CLKDIV4[15:12]).
  - Enable UART5 peripheral clock in UART5CKEN (CLK_APBCLK0[21]).
- Reset UART5 controller in UART5RST (SYS_IPRST1[21]).

6.27.4.7 UART6 Basic Configuration

- Clock Source Configuration
  - Select the source of UART6 peripheral clock on UART6SEL (CLK_CLKSEL4[26:24]).
  - Select the clock divider number of UART6 peripheral clock on UART6DIV (CLK_CLKDIV4[19:16]).
  - Enable UART6 peripheral clock in UART6CKEN (CLK_APBCLK0[22]).
- Reset UART6 controller in UART6RST (SYS_IPRST1[22]).

6.27.4.8 UART7 Basic Configuration

- Clock Source Configuration
  - Select the source of UART7 peripheral clock on UART7SEL (CLK_CLKSEL4[30:28]).

- Select the clock divider number of UART7 peripheral clock on UART7DIV (CLK_CLKDIV4[23:20]).
- Enable UART7 peripheral clock in UART7CKEN (CLK_APBCLK0[23]).

- Reset UART7 controller in UART7RST (SYS_IPRST1[23]).

UART Interface Controller Pin description is shown in Table 6.28-3:

Pin	Type	Description
UARTx_TXD	Output	UARTx transmit
UARTx_RXD	Input	UARTx receive
UARTx_nCTS	Input	UARTx modem clear to send
UARTx_nRTS	Output	UARTx modem request to send

Table 6.27-3 UART Interface Controller Pin

6.27.5 Functional Description

The UART controller supports five function modes including UART, IrDA, LIN, RS-485 and Single-wire mode. User can select a function by setting the UART_FUNCSEL register. The five function modes will be described in following section.

6.27.5.1 UART Controller Baud Rate Generator

The UART controller includes a programmable baud rate generator capable of dividing clock input by divisors to produce the serial clock that transmitter and receiver need. Table 6.28-4 list the UART baud rate equations in the various conditions. Table 6.28-5 and Table 6.28-6 list the UART baud rate parameter and register setting example. In IrDA function mode, the baud rate generator must be set in mode 0. More detailed register description is shown in UART_BAUD register. There are three setting modes. Mode 0 is set by UART_BAUD[29:28] with 00. Mode 1 is set by UART_BAUD[29:28] with 10. Mode 2 is set by UART_BAUD[29:28] with 11.

Mode	BAUDM1	BAUDM0	Baud Rate Equation
Mode 0	0	0	UART_CLK / [16 * (BRD+2)].
Mode 1	1	0	UART_CLK / [(EDIVM1+1) * (BRD+2)], EDIVM1 must >= 8.
Mode 2	1	1	UART_CLK / (BRD+2) If UART_CLK <= 3*HCLK, BRD must >= 8. If UART_CLK > 3*HCLK, BRD must >= 3*N – 1. N is the smallest integer larger than or equal to the ratio of UART_CLK /HCLK. For example, if 2*HCLK < UART_CLK =< 3*HCLK, BRD must >=11. if 4*HCLK < UART_CLK =< 5*HCLK, BRD must >=14. (If the UART_CLK is selected from LXT, BRD can be greater than or equal to 1)

Table 6.27-4 UART controller Baud Rate Equation Table

UART Peripheral Clock = 12 MHz			
Baud Rate	Mode 0	Mode 1	Mode 2
921600	Not support	Not recommended	BRD=11
460800	Not recommended	BRD=0, EDIVM1 =12	BRD=24

230400	Not recommended	BRD =2, EDIVM1 =12	BRD =50
115200	Not recommended	BRD =6, EDIVM1 =12	BRD =102
57600	BRD =11	BRD =14, EDIVM1 =12	BRD =206
38400	BRD =18	BRD =22, EDIVM1 =12	BRD =311
19200	BRD =37	BRD =61, EDIVM1 =9	BRD =623
9600	BRD =76	BRD =123, EDIVM1 =9	BRD =1248
4800	BRD =154	BRD =248, EDIVM1 =9	BRD =2498

Table 6.27-5 UART controller Baud Rate Parameter Setting Example Table

UART Peripheral Clock = 12 MHz			
Baud Rate	UART_BAUD Value		
	Mode 0	Mode 1	Mode 2
921600	Not support	Not recommended	0x3000_000B
460800	Not recommended	0x2C00_0000	0x3000_0018
230400	Not recommended	0x2C00_0002	0x3000_0032
115200	Not recommended	0x2C00_0006	0x3000_0066
57600	0x0000_000B	0x2C00_000E	0x3000_00CE
38400	0x0000_0012	0x2C00_0016	0x3000_0137
19200	0x0000_0025	0x2900_003D	0x3000_026F
9600	0x0000_004C	0x2900_007B	0x3000_04E0
4800	0x0000_009A	0x2900_00F8	0x3000_09C2

Table 6.27-6 UART controller Baud Rate Register Setting Example Table

6.27.5.2 UART Controller Baud Rate Compensation

The UART controller supports baud rate compensation function. It is used to optimize the precision in each bit. The precision of the compensation is half of UART module clock because there is BRCOMDEC (UART_BRCOMP[31]) to define the positive or negative compensation in each bit. If the BRCOMDEC (UART_BRCOMP[31]) = 0, it is positive compensation for each bit, one more module clock will be append in the compensated bit. If the BRCOMPDEC (UART_BRCOMP[31]) = 1, it is negative compensation for each bit, decrease one module clock in the compensated bit.

There is 9-bits location, BRCOMP[8:0] (UART_BRCOMP[8:0]), can be configured by user to define the relative bit is compensated or not. BRCOMP[7:0] is used to define the compensation of DAT (UART_DAT[7:0]) and BRCOMP[8] is used to define PARITY (UART_DAT[8]).

Example:

1. UART's peripheral clock = 32.768 kHz and baud rate is 9600 bps

Baud rate is 9600 bps, UART peripheral clock is 32.768 kHz → 3.413 peripheral clock/bit

If the baud divider is set 1 (3 peripheral clock/bit), the inaccuracy of each bit is -0.413 peripheral clock and BRCOMPDEC (UART_BRCOMP[31]) = 0, so that the BRCOMP (UART_BRCOMP[8:0]) can be set as 9'b010100101 = 0xa5.

Bit	Name	Total INACCURACY	BRCOMP Compensated	Final Inaccuracy
0	START	-0.413	x	-0.413
1	UART_DAT[0]	-0.826(-0.413-0.413)	1	0.174
2	UART_DAT[1]	-0.239(0.174-0.413)	0	-0.239
3	UART_DAT[2]	-0.652(-0.239-0.413)	1	0.348
4	UART_DAT[3]	-0.065(0.348-0.413)	0	-0.065
5	UART_DAT[4]	-0.478(-0.065-0.413)	0	-0.478
6	UART_DAT[5]	-0.891(-0.478-0.413)	1	0.109
7	UART_DAT[6]	-0.304(0.109-0.413)	0	-0.304
8	UART_DAT[7]	-0.717(-0.304-0.413)	1	0.283
9	PARITY	-0.130(0.283-0.413)	0	-0.13

Table 6.27-7 Baud Rate Compensation Example Table 1

2. UART's peripheral clock = 32.768 kHz and baud rate is 4800

Baud rate is 4800, UART peripheral clock is 32.768 kHz → 6.827 peripheral clock/bit

If the baud divider is set 5 (7 peripheral clock/bit), the inaccuracy of each bit is 0.173 peripheral clock and BRCOMPDEC (UART_BRCOMP[31]) = 1, so that the BRCOMP (UART_BRCOMP[8:0]) can be set as 9'b010000010 = 0x82.

Bit	Name	Total INACCURACY	BRCOMP Compensated	Final Inaccuracy
0	START	0.173	x	0.173
1	UART_DAT[0]	0.346(0.173+0.173)	0	0.346
2	UART_DAT[1]	0.519(0.346+0.173)	1	-0.481
3	UART_DAT[2]	-0.308(-0.481+0.173)	0	-0.308
4	UART_DAT[3]	-0.135(-0.308+0.173)	0	-0.135
5	UART_DAT[4]	-0.038(-0.135+0.173)	0	0.038
6	UART_DAT[5]	0.211(0.038+0.173)	0	0.211
7	UART_DAT[6]	0.384(0.211+0.173)	0	0.384
8	UART_DAT[7]	0.557(0.384+0.173)	1	-0.443
9	PARITY	-0.270(-0.443+0.173)	0	-0.270

Table 6.27-8 Baud Rate Compensation Example Table 2

6.27.5.3 UART Controller Auto-Baud Rate Function Mode:

Auto-Baud Rate function can measure baud rate of receiving data from UART RX pin automatically. When the Auto-Baud Rate measurement is finished, the measuring baud rate is loaded to BRD (UART_BAUD[15:0]). Both of the BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) are set to 1 automatically. UART RX data from START bit to 1st rising edge time is set by $2^{ABRDBITS}$ bit time in Auto-Baud Rate function detection frame.

$2^{ABRDBITS}$ bit time from START bit to the 1st rising edge is calculated by setting ABRDBITS

(UART_ALTCTL[20:19]). Set ABRDEN (UART_ALTCTL[18]) to enable auto-baud rate function. In the beginning when no data transfer, the UART RX is held high. Once the falling edge is detected, START bit is received. The auto-baud rate counter will be reset and then start counting. The auto-baud rate counter will be stop when the 1st rising edge is detected. Then, the auto-baud rate counter value divided by ABRDBITS (UART_ALTCTL[20:19]) is loaded to BRD (UART_BAUD[15:0]) automatically. ABRDEN (UART_ALTCTL[18]) is cleared. The Auto-Baud measurement is shown in Figure 6.28-3. Once the auto-baud rate measurement is finished, the ABRDIF (UART_FIFOSTS[1]) is set. When auto-baud rate counter is overflow, ABRDTOIF (UART_FIFOSTS[2]) is set. If ABRDIF (UART_FIFOSTS[1]) or ABRDTOIF (UART_FIFOSTS[2]) is set, the auto-baud rate flag ABRIF (UART_ALTCTL[17]) generated. If ABRIEN (UART_INTEN[18]) is enabled, the auto-baud rate interrupt ABRINT (UART_INTSTS[31]) is generated when ABRIF (UART_ALTCTL[17]) is set.

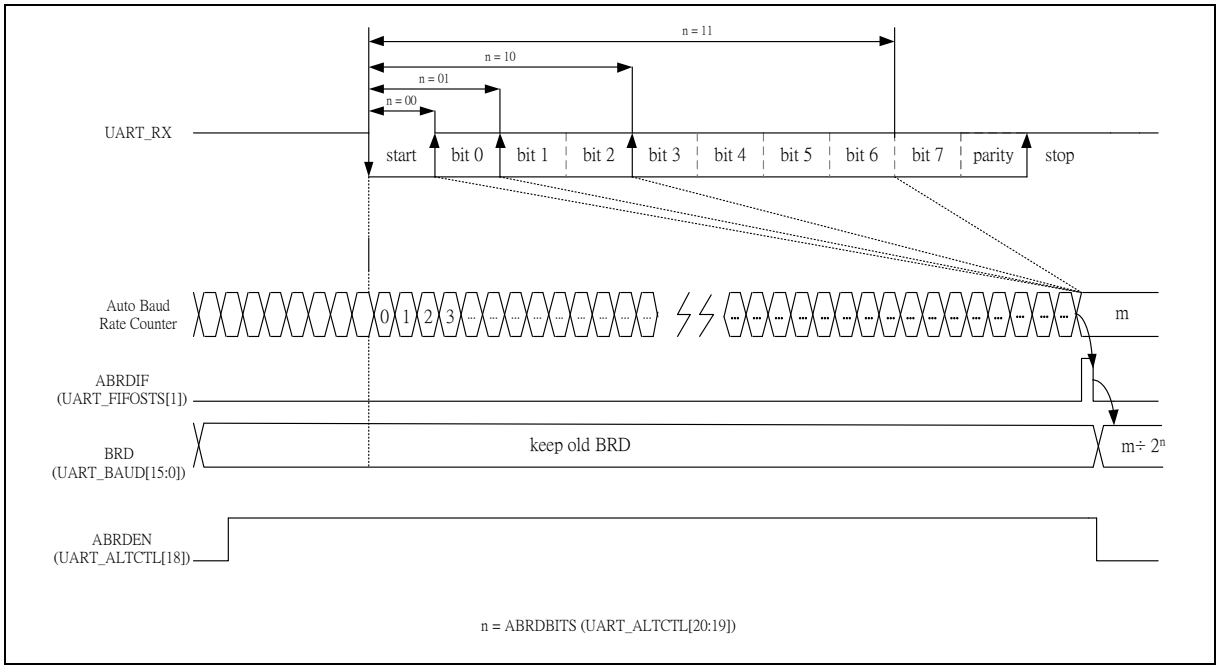


Figure 6.27-3 Auto-Baud Rate Measurement

Programming Sequence Example:

1. Program ABRDBITS (UART_ALTCTL[20:19]) to determine UART RX data 1st rising edge time from Start by $2^{ABRDBITS}$ bit time.
2. Set ABRIEN (UART_INTEN[18]) to enable auto-baud rate function interrupt.
3. Set ABRDEN (UART_ALTCTL[18]) to enable auto-baud rate function.
4. ABRDIF (UART_FIFOSTS[1]) is set, the auto-baud rate measurement is finished.
5. Operate UART transmit and receive action.
6. ABRDTOIF (UART_FIFOSTS[2]) is set, if auto-baud rate counter is overflow.
7. Go to Step 3.

6.27.5.4 UART Controller Transmit Delay Time Value

The UART controller programs DLY (UART_TOUT[15:8]) to control the transfer delay time between the last STOP bit and next START bit in transmission. The unit is baud. The operation is shown in Figure 6.28-4.

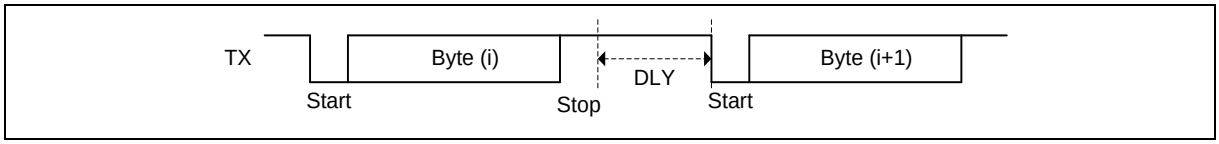


Figure 6.27-4 Transmit Delay Time Operation

6.27.5.5 UART Controller FIFO Control and Status

The UART controller is built-in with a 16 bytes transmitter FIFO (TX_FIFO) and a 16 bytes receiver FIFO (RX_FIFO) that reduces the number of interrupts presented to the CPU. The CPU can read the status of the UART at any time during operation. The reported status information includes condition of the transfer operations being performed by the UART, as well as 3 error conditions (parity error, framing error, break interrupt) occur if receiving data has parity, frame or break error. UART, IrDA, LIN and RS-485 mode support FIFO control and status function.

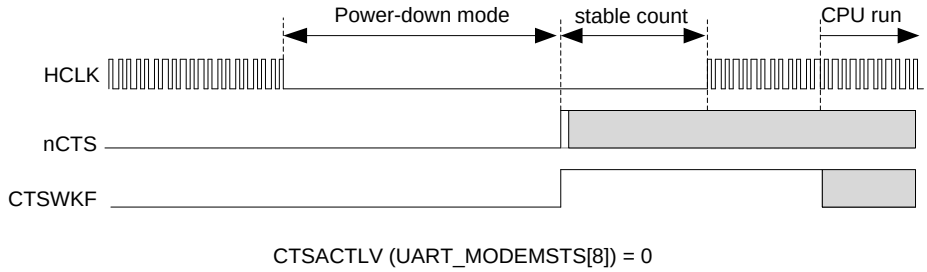
6.27.5.6 UART Controller Wake-up Function

The UART controller supports wake-up system function. The wake-up function includes nCTS pin, incoming data wake-up, Received Data FIFO reached threshold wake-up, RS-485 Address Match (AAD mode) wake-up and Received Data FIFO threshold time-out wake-up function. CTSWKF (UART_WKSTS[0]), DATWKF (UART_WKSTS[1]), RFRTWKF (UART_WKSTS[2]), RS485WKF (UART_WKSTS[3]) or TOUTWKF (UART_WKSTS[4]) cause the wake-up interrupt flag WKIF(UART_INTSTS[6]) generated. If the WKIEN (UART_INTEN[6]) is enabled, the wake-up interrupt flag WKIF(UART_INTSTS[6]) cause the wake-up interrupt WKINT (UART_INTSTS[14]) generated.

nCTS pin wake-up :

When the system is in Power-down mode and WKCTSEN (UART_WKCTL[0]) is set, the toggle of nCTS pin can wake-up system. If the WKCTSEN (UART_WKCTL[0]) is enabled, the toggle of nCTS pin cause the nCTS wake-up flag CTSWKF (UART_WKSTS[0]) generated. The nCTS wake-up is shown in Figure 6.28-5 and Figure 6.28-6.

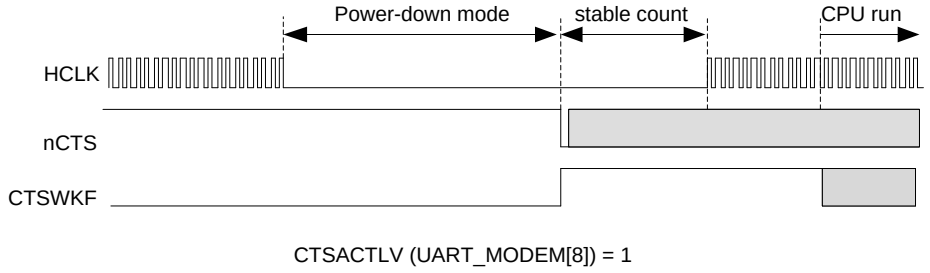
nCTS Wake-up Case 1 (nCTS transition from low to high)



Note: Stable count means HCLK source recovery stable count.

Figure 6.27-5 UART nCTS Wake-up Case1

nCTS Wake-up Case 2 (nCTS transition from high to low)



Note: Stable count means HCLK source recovery stable count.

Figure 6.27-6 UART nCTS Wake-up Case2

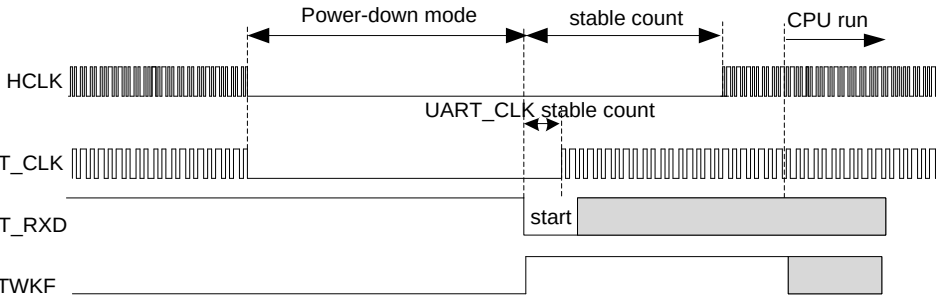
Incoming Data Wake-up

When system is in Power-down mode and the WKDATEN (UART_WKCTL[1]) is set, the toggle of incoming data (UART_RXD) pin can wake-up the system. In order to receive the incoming data after the system wake-up, the STCOMP (UART_DWKCOMP[15:0]) shall be set. These bits field of STCOMP indicate how many clock cycle selected by UART_CLK do the UART controller can get the 1st bit (START bit) when the system is wakeup from Power-down mode.

When incoming data wakes system up, the incoming data will be received and stored in FIFO. If the WKDATEN (UART_WKCTL[1]) is enabled, the toggle of incoming data (UART_RXD) pin cause the incoming data wake-up flag DATWKF (UART_WKSTS[1]) generated. The incoming data wake-up is shown in Figure 6.28-7.

Note 1: The UART controller clock source should be selected as HIRC, and the compensation time for START bit that refer to the M2L31 Datasheet for detailed information about wakeup time electrical characteristics. The STCOMP (UART_DWKCOMP[15:0]) = (wake-up stable time) * (HIRC frequency).

Note 2: The value of BRD(UART_BAUD[15:0]) should be greater than STCOMP (UART_DWKCOMP[15:0]).



Note 1: Stable count means HCLK source recovery stable count.

Note 2: UART_CLK stable count means UART clock source recovery stable count.

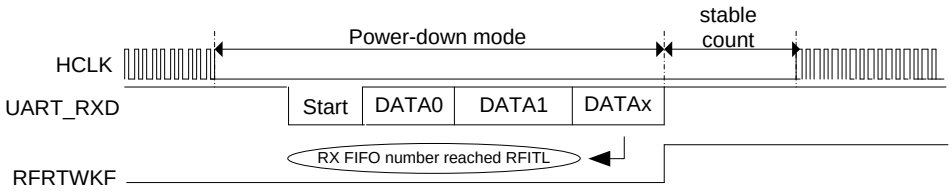
Figure 6.27-7 UART Data Wake-up

Received Data FIFO Reached Threshold Wake-up

The received data FIFO threshold reached wake-up function is enabled by setting WKRFRTEN (UART_WKCTL[2]). In Power-down mode, when the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]), it can wake-up the system. If the WKRFRTEN (UART_WKCTL[2]) is enabled, the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]) cause the received data FIFO reached threshold wake-up flag RFRTWKF (UART_WKSTS[2]) generated. The Received Data FIFO reached threshold wake-up is shown in Figure 6.28-8.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive

data.



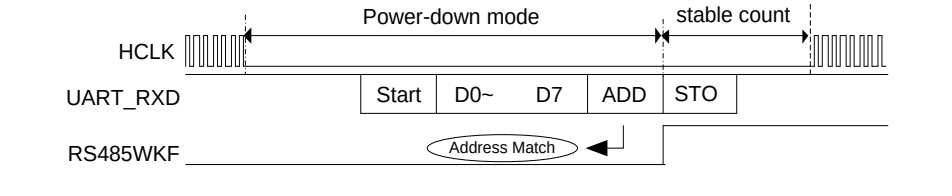
Note: Stable count means HCLK source recovery stable count.

Figure 6.27-8 UART Received Data FIFO reached threshold wake-up

RS-485 Address Match (AAD Mode) Wake-up

The RS-485 address match wake-up function is enabled by setting WKRFRTEN (UART_WKCTL[2]) and WKRS485EN (UART_WKCTL[3]). This function is used for RS-485 Auto Address Detection (AAD) mode in RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1. In Power-down mode, when an address byte is detected and matches the ADDRMV (UART_ALTCTL[31:24]) or the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]), it can wake-up the system. If the WKRS485EN (UART_WKCTL[3]) is enabled, when an address byte is detected and matches the ADDRMV (UART_ALTCTL[31:24]), the RS485 address match (AAD mode) wake-up flag RS485WKF (UART_WKSTS[3]) generated. The RS-485 Address Match (AAD mode) wake-up is shown in Figure 6.28-9.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



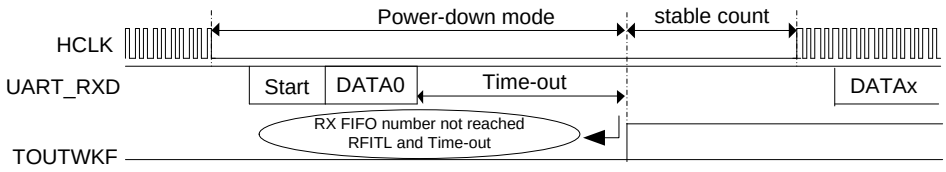
Note: Stable count means HCLK source recovery stable count.

Figure 6.27-9 UART RS-485 AAD Mode Address Match Wake-up

Received Data FIFO Threshold Time-out Wake-up

The received data FIFO threshold time-out wake-up function is enabled by setting WKRFRTEN (UART_WKCTL[2]) and WKTOUTEN (UART_WKCTL[4]). Setting TOCNTEN (UART_INTEN[11]) to enable receiver buffer time-out counter. In Power-down mode, when the number of received data in RX FIFO does not reach the threshold value RFITL (UART_FIFO[7:4]) and the time-out counter equals to the time-out value TOIC (UART_TOUT[7:0]), it can wake-up the system. If the WKTOUTEN (UART_WKCTL[4]) is enabled, when the time-out counter equals to the time-out value TOIC (UART_TOUT[7:0]), the Received Data FIFO threshold time-out wake-up flag TOUTWKF (UART_WKSTS[4]) generated. The Received Data FIFO threshold time-out wake-up is shown in Figure 6.28-10.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



Note: Stable count means HCLK source recovery stable count.

Figure 6.27-10 UART Received Data FIFO threshold time-out wake-up

6.27.5.7 UART Controller Interrupt and Status

Each UART controller supports eleven types of interrupts including:

- Receive Data Available Interrupt (RDAINT)
- Transmit Holding Register Empty Interrupt (THREINT)
- Transmitter Empty Interrupt (TXENDIF)
- Receive Line Status Interrupt (RLSINT)
  - Break Interrupt Flag (BIF)
  - Framing Error Flag (FEF)
  - Parity Error Flag (PEF)
  - RS-485 Address Byte Detect Flag (ADDRDETF)
- MODEM Status Interrupt (MODEMINT)
  - Detect nCTS State Change Flag (CTSDETF)
- Receiver Buffer Time-out Interrupt (RXTOINT)
- Buffer Error Interrupt (BUFERRINT)
  - TX Overflow Error Interrupt Flag (TXOVIF)
  - RX Overflow Error Interrupt Flag (RXOVIF)
- LIN Bus Interrupt (LININT)
  - LIN Break Detection Flag (BRKDETF)
  - Bit Error Detect Status Flag (BITEF)
  - LIN Slave ID Parity Error Flag (SLVIDPEF)
  - LIN Slave Header Error Flag (SLVHEF)
  - LIN Slave Header Detection Flag (SLVHDETF)
- Wake-up Interrupt (WKINT)
  - nCTS Wake-up Flag (CTSWKF)
  - Incoming Data Wake-up Flag (DATWKF)
  - Received Data FIFO Reached Threshold Wake-up Flag (RFRTWKF)
  - RS-485 Address Match (AAD mode) Wake-up Flag (RS485WKF)
  - Received Data FIFO Threshold Time-out Wake-up Flag (TOUTWKF)
- Auto-Baud Rate Interrupt (ABRINT)
  - Auto-baud Rate Detect Interrupt Flag (ABRDIF)

- Auto-baud Rate Detect Time-out Interrupt Flag (ABRDTOIF)

- Single-wire Bit Error Detect Interrupt (SWBEINT)

Table 6.28-9 describes the interrupt sources and flags. The interrupt is generated when the interrupt flag is generated and the interrupt enable bit is set. User must clear the interrupt flag after the interrupt is generated.

Interrupt Source	Interrupt Indicator	Interrupt Enable Bit	Interrupt Flag	Flag Caused By	Flag Cleared By
Receive Data Available Interrupt	RDAINT	RDAIEN	RDAIF	N/A	Read UART_DAT
Transmit Holding Register Empty Interrupt	THREINT	THREIEN	THREIF	N/A	Write UART_DAT
Transmitter Empty Interrupt	TXENDINT	TXENDIEN	TXENDIF	N/A	Write UART_DAT
Receive Line Status Interrupt	RLSINT	RLSIEN	RLSIF	RLSIF = BIF	Write '1' to BIF
				RLSIF = FEF	Write '1' to FEF
				RLSIF = PEF	Write '1' to PEF
				RLSIF = ADDRDETF	Write '1' to ADDRDETF
Modem Status Interrupt	MODEMINT	MODEMIEN	MODEMIF	MODEMIF = CTSDETF	Write '1' to CTSDETF
Receiver Buffer Time-out Interrupt	RXTOINT	RXTOIEN	RXTOIF	N/A	Read UART_DAT
Buffer Error Interrupt	BUFERRINT	BUFERRIEN	BUFERRIF	BUFERRIF = TXOVIF	Write '1' to TXOVIF
				BUFERRIF = RXOVIF	Write '1' to RXOVIF
LIN Bus Interrupt	LININT	LINIEN	LINIF	LINIF = BRKDETF	Write '1' to LINIF and Write '1' to BRKDETF
				LINIF = BITEF	Write '1' to LINIF and Write '1' to BITEF
				LINIF = SLVIDPEF	Write '1' to LINIF and Write '1' to SLVIDPEF
				LINIF = SLVHEF	Write '1' to LINIF and Write '1' to SLVHEF
				LINIF = SLVHDETF	Write '1' to LINIF and Write '1' to SLVHDETF
Wake-up Interrupt	WKINT	WKIEN	WKIF	WKIF = CTSWKF	Write '1' to CTSWKF
				WKIF = DATWKF	Write '1' to DATWKF

				WKIF = RFRTWKF	Write ‘1’ to RFRTWKF
				WKIF = RS485WKF	Write ‘1’ to RS485WKF
				WKIF = TOUTWKF	Write ‘1’ to TOUTWKF
Auto-Baud Rate Interrupt	ABRINT	ABRIEN	ABRIF	ABRIF = ABRDIF	Write ‘1’ to ABRDIF
				ABRIF = ABRDTOIF	Write ‘1’ to ABRDTOIF
Single-wire Bit Error Detect Interrupt	SWBEINT	SWBEIEN	SWBEIF	N/A	Writing ‘1’ to SWBEIF

Table 6.27-9 UART controller Interrupt Source and Flag List

6.27.5.8 UART Function Mode

The UART controller provides UART function (Setting FUNCSEL (UART_FUNCSEL[2:0]) to ‘000’ to enable UART function mode). The UART baud rate is up to 10 Mbps.

The UART provides full-duplex and asynchronous communications. The transmitter and receiver contain 16 bytes FIFO for payloads. User can program receiver buffer trigger level and receiver buffer time-out detection for receiver. The transmitting data delay time between the last stop and the next START bit can be programed by setting DLY (UART_TOUT[15:8]) register. The UART supports hardware auto-flow control that provides programmable nRTS flow control trigger level. The number of data bytes in RX FIFO is equal to or greater than RTSTRGLV (UART_FIFO[19:16]), the nRTS is de-asserted.

UART Line Control Function

The UART controller supports fully programmable serial-interface characteristics by setting the UART_LINE register. User can program UART_LINE register for the word length, STOP bit and PARITY bit setting. Table 6.28-10 and Table 6.28-11 list the UART word, STOP bit length and the PARITY bit settings.

NSB (UART_LINE[2])	WLS (UART_LINE[1:0])	Word Length (Bit)	Stop Length (Bit)
0	00	5	1
0	01	6	1
0	10	7	1
0	11	8	1
1	00	5	1.5
1	01	6	2
1	10	7	2
1	11	8	2

Table 6.27-10 UART Line Control of Word and Stop Length Setting

Parity Type	SPE (UART_LINE[5])	EPE (UART_LINE[4])	PSS (UART_LINE[7])	PBE (UART_LINE[3])	Description
No Parity	x	x	x	0	No PARITY bit output.

Parity source from UART_DAT	x	x	1	1	PARITY bit is generated and checked by software.
Odd Parity	0	0	0	1	Odd Parity is calculated by adding all the “1’s” in a data stream and adding a PARITY bit to the total bits, to make the total count an odd number.
Even Parity	0	1	0	1	Even Parity is calculated by adding all the “1’s” in a data stream and adding a PARITY bit to the total bits, to make the count an even number.
Forced Mask Parity	1	0	0	1	PARITY bit always logic 1. PARITY bit on the serial byte is set to “1” regardless of total number of “1’s” (even or odd counts).
Forced Space Parity	1	1	0	1	PARITY bit always logic 0. PARITY bit on the serial byte is set to “0” regardless of total number of “1’s” (even or odd counts).

Table 6.27-11 UART Line Control of PARITY bit Setting

UART Auto-Flow Control Function

The UART supports auto-flow control function that uses two signals, nCTS (clear-to-send) and nRTS (request-to-send), to control the flow of data transfer between the UART and external devices (e.g. Modem). When auto-flow is enabled, the UART is not allowed to receive data until the UART asserts nRTS to external device. When the number of bytes stored in the RX FIFO equals the value of RTSTRGLV (UART_FIFO[19:16]), the nRTS is de-asserted. The UART sends data out when UART detects nCTS is asserted from external device. If the valid asserted nCTS is not detected, the UART will not send data out. The auto flow control block diagram is shown in Figure 6.28-11.

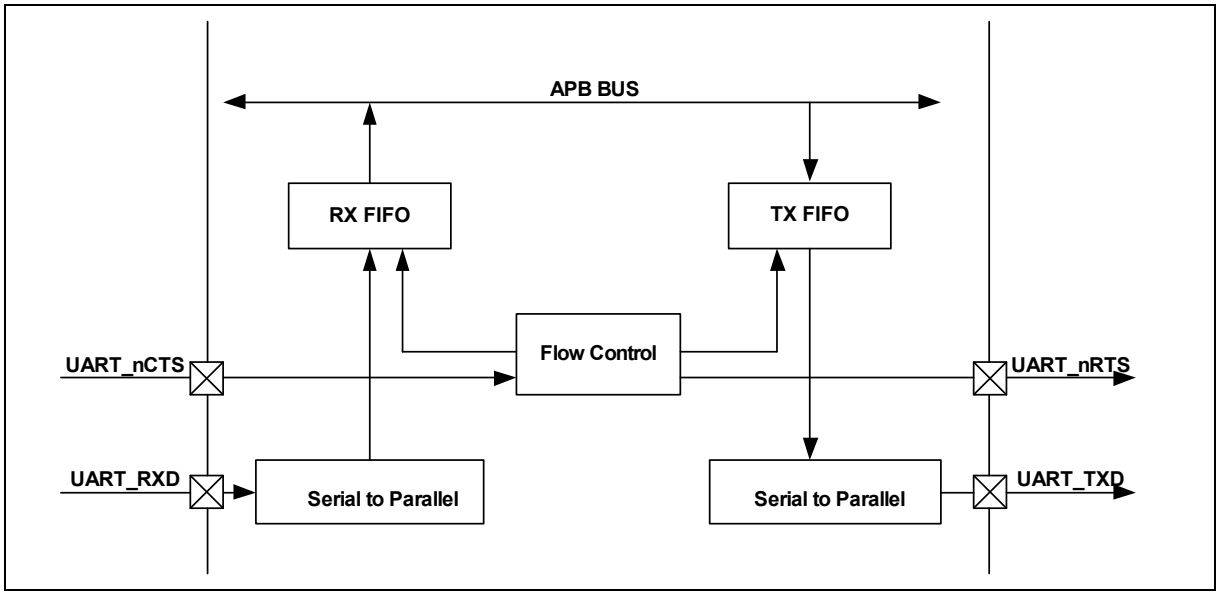


Figure 6.27-11 Auto-Flow Control Block Diagram

Figure 6.28-12 demonstrates the nCTS auto-flow control of UART function mode. User must set ATOCTSEN (UART_INTEN[13]) to enable nCTS auto-flow control function. The CTSACTLV (UART_MODEMSTS[8]) can set nCTS pin input active state. The CTSDETF (UART_MODEMSTS[0]) is set when any state change of nCTS pin input has occurred, and then TX data will be automatically transmitted from TX FIFO.

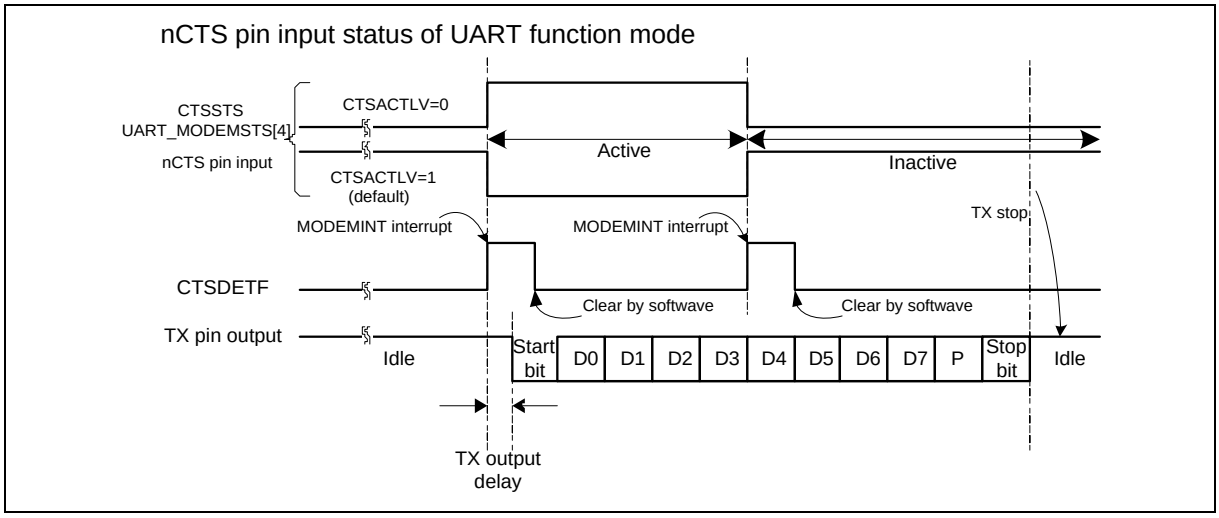


Figure 6.27-12 UART nCTS Auto-Flow Control Enabled

As shown in Figure 6.28-13, in UART nRTS auto-flow control mode (ATORTSEN(UART_INTEN[12])=1), the nRTS internal signal is controlled by UART FIFO controller with RTSTRGLV(UART_FIFO[19:16]) trigger level.

Setting RTSACTLV(UART_MODEM[9]) can control the nRTS pin output is inverse or non-inverse from nRTS signal. User can read the RTSSTS (UART_MODEM[13]) bit to get real nRTS pin output voltage logic status.

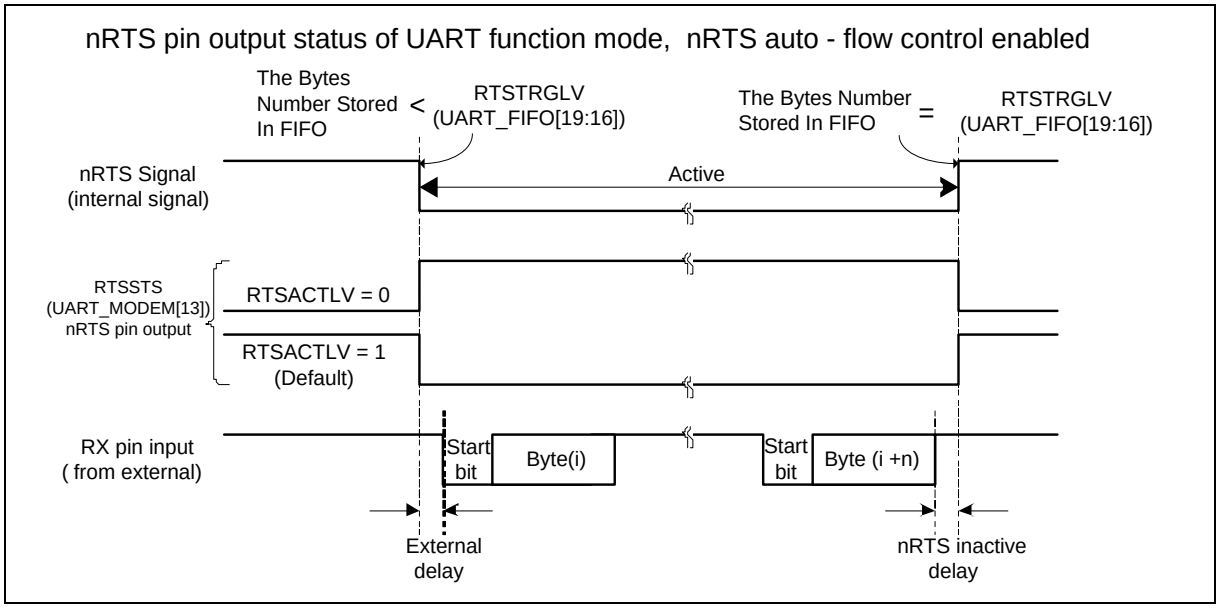


Figure 6.27-13 UART nRTS Auto-Flow Control Enabled

As shown in Figure 6.28-14, in software mode (ATORTSEN(UART_INTEN[12])=0), the nRTS flow is directly controlled by software programming of RTS(UART_MODEM[1]) control bit.

Setting RTSACTLV(UART_MODEM[9]) can control the nRTS pin output is inverse or non-inverse from RTS(UART_MODEM[1]) control bit. User can read the RTSSTS(UART_MODEM[13]) bit to get real nRTS pin output voltage logic status.

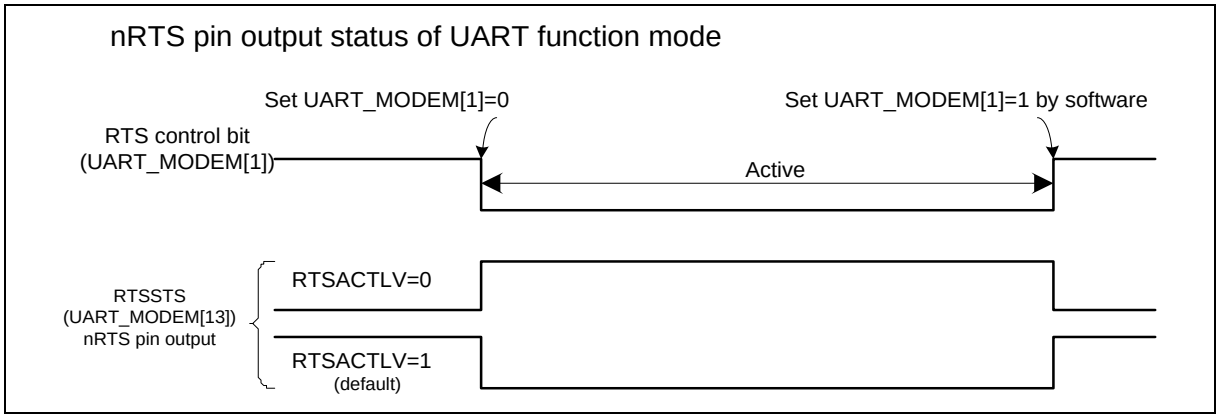


Figure 6.27-14 UART nRTS Auto-Flow with Software Control

6.27.5.9 IrDA Function Mode

The UART controller also provides Serial IrDA (SIR, Serial Infrared) function (Setting FUNCSEL (UART_FUNCSEL[2:0]) to '010' to enable the IrDA function). The SIR specification defines a short-range infrared asynchronous serial transmission mode with one START bit, 8 data bits, and 1 STOP bit. The maximum data rate is 115.2 kbps. The IrDA SIR block contains an IrDA SIR protocol encoder/decoder. The IrDA SIR protocol is half-duplex only. So, it cannot transmit and receive data at the same time. The IrDA SIR physical layer specifies a minimum 10 ms transfer delay between transmission and reception, and this delay feature must be implemented by software.

In IrDA mode, the BAUDM1 (UART_BAUD[29]) must be cleared.

Baud Rate = Clock / (16 * (BRD +2)), where BRD (UART_BAUD[15:0]) is Baud Rate Divider in UART_BAUD register.

Note: The tolerance of baud-rate is ±5% between IrDA master and IrDA slave.

The IrDA control block diagram is shown in Figure 6.27-15.

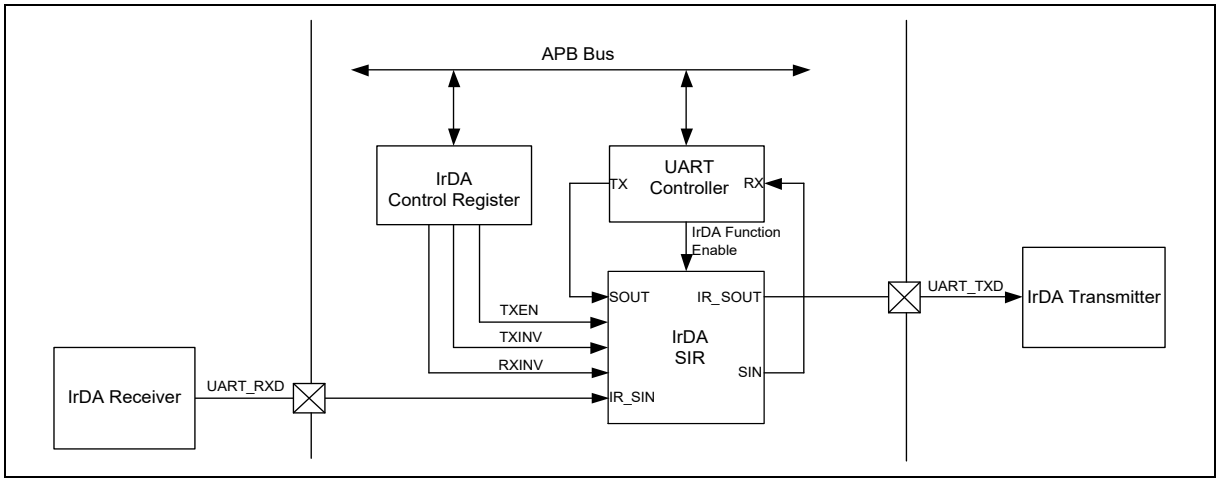


Figure 6.27-15 IrDA Control Block Diagram

IrDA SIR Transmit Encoder

The IrDA SIR Transmit Encoder modulates Non-Return-to-Zero (NRZ) transmit bit stream output from UART. The IrDA SIR physical layer specifies the use of Return-to-Zero, Inverted (RZI) modulation scheme which represents logic 0 as an infra light pulse. The modulated output pulse stream is transmitted to an external output driver and infrared light emitting diode.

The transmitted pulse width is specified as 3/16 period of baud rate.

IrDA SIR Receive Decoder

The IrDA SIR Receive Decoder demodulates the Return-to-Zero bit stream from the input detector and outputs the NRZ serial bits stream to the UART received data input.

In idle state, the decoder input is high. A START bit is detected when the decoder input is LOW. In normal operation, the RXINV (UART_IRDA[6]) is set to '1' and TXINV (UART_IRDA[5]) is set to '0'.

IrDA SIR Operation

The IrDA SIR encoder/decoder provides functionality which converts between UART data stream and half-duplex serial SIR interface. Figure 6.27-16 is IrDA encoder/decoder waveform.

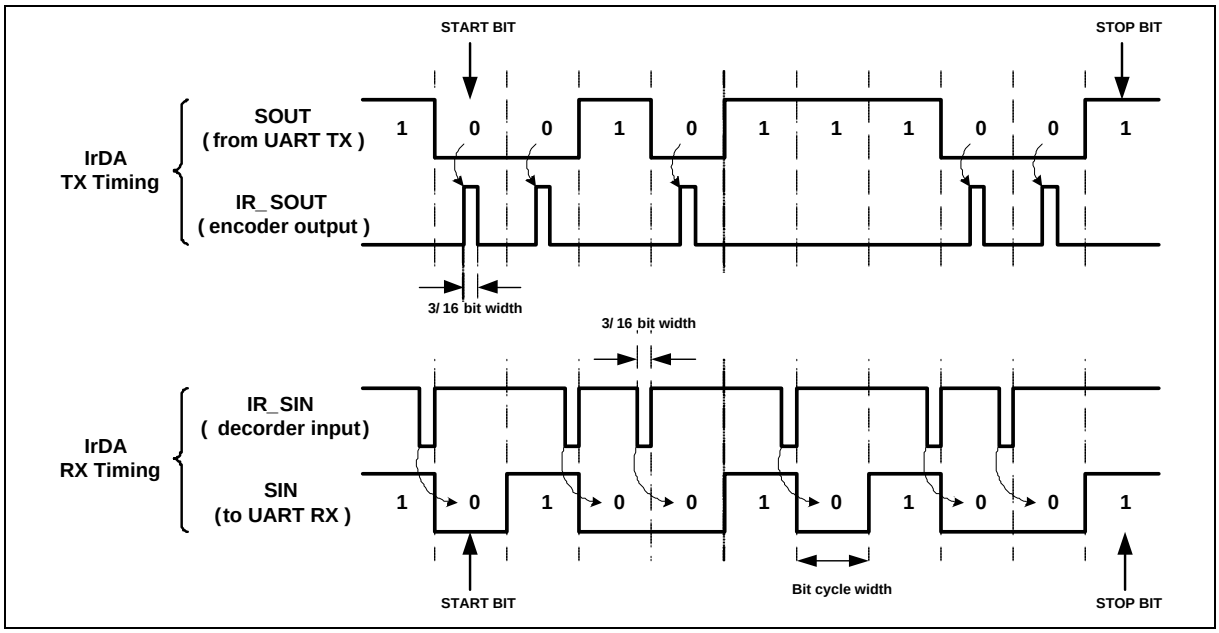


Figure 6.27-16 IrDA TX/RX Timing Diagram

6.27.5.10 LIN Function Mode (Local Interconnection Network)

The UART Controller supports LIN function. Setting FUNCSEL (UART_FUNCSEL[2:0]) to '001' to select LIN mode operation. The UART Controller supports LIN break/delimiter generation and break/delimiter detection in LIN master mode, and supports header detection and automatic resynchronization in LIN Slave mode.

Structure of LIN Frame

According to the LIN protocol, all information transmitted is packed as frames; a frame consists of a header (provided by the master task) and a response (provided by a slave task). The header (provided by the master task) consists of a break field and a sync field followed by a frame identifier (frame ID). The frame identifier uniquely defines the purpose of the frame. The slave task is appointed for providing the response associated with the frame ID. The response consists of a data field and a checksum field. Figure 6.27-17 is the structure of LIN Frame.

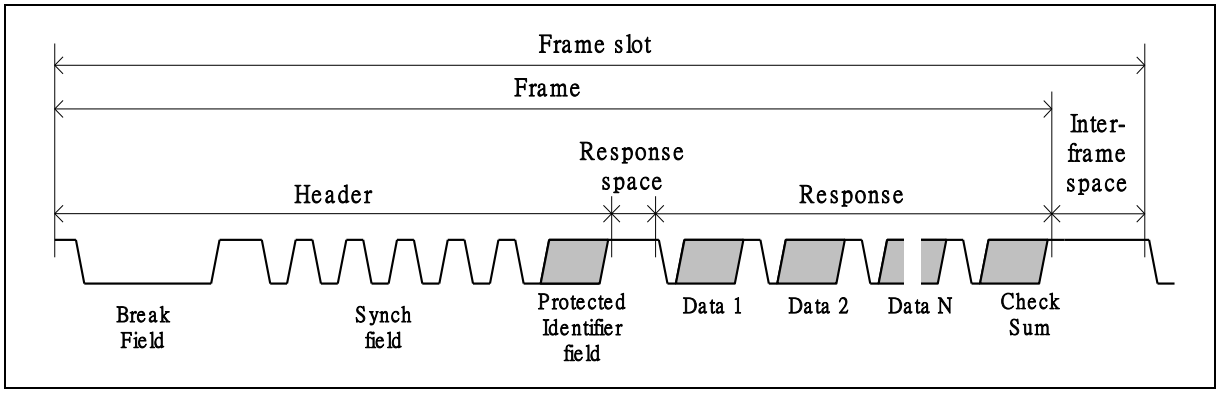


Figure 6.27-17 Structure of LIN Frame

Structure of LIN Byte

In LIN mode, each byte field is initiated by a START bit with value 0 (dominant), followed by 8 data bits and no PARITY bit, LSB is first and ended by 1 STOP bit with value 1 (recessive) in accordance with the LIN standard. The structure of Byte is shown in Figure 6.27-18.

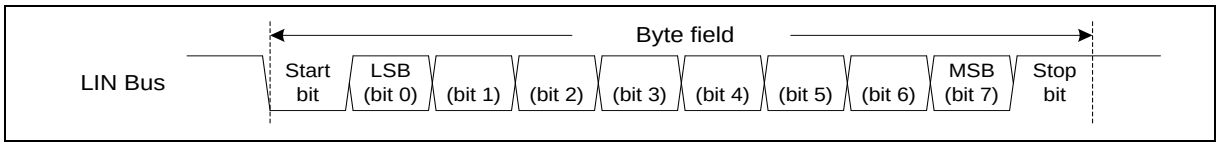


Figure 6.27-18 Structure of LIN Byte

LIN Master Mode

The UART Controller supports LIN Master mode. To enable and initialize the LIN Master mode, the following steps are necessary:

1. Set the UART_BAUD register to select the desired baud rate.
2. Set WLS (UART_LINE[1:0]) to '11' to configure the word length with 8 bits, clearing PBE (UART_LINE[3]) bit to disable parity check and clearing NSB (UART_LINE[2]) bit to configure with one STOP bit.
3. Set FUNCSEL (UART_FUNCSEL[2:0]) to '001' to select LIN function mode operation.

A complete header consists of a break field and sync field followed by a frame identifier (frame ID). The UART controller can be selected header sending by three header selected modes. The header selected mode can be "break field" or "break field and sync field" or "break field, sync field and frame ID field" by setting HSEL (UART_LINCTL[23:22]). If the selected header is "break field", software must handle the following sequence to send a complete header to bus by filling sync data (0x55) and frame ID data to the UART_DAT register. If the selected header is "break field and sync field", software must handle the sequence to send a complete header to bus by filling the frame ID data to UART_DAT register, and if the selected header is "break field, sync field and frame ID field", hardware will control the header sending sequence automatically but software must filled frame ID data to PID (UART_LINCTL[31:24]). When operating in header selected mode in which the selected header is "break field, sync field and frame ID field", the frame ID PARITY bit can be calculated by software or hardware depending whether the IDPEN (UART_LINCTL[9]) bit is set or not.

HSEL	Break Field	Sync Field	ID Field
0	Generated by Hardware	Handled by Software	Handled by Software
1	Generated by Hardware	Generated by Hardware	Handled by Software

2	Generated by Hardware	Generated by Hardware	Generated by Hardware (But Software needs to fill ID to PID (UART_LINCTL[31:24]) first)

Table 6.27-12 LIN Header Selection in Master Mode

When UART is operated in LIN data transmission, LIN bus transfer state can be monitored by hardware or software. User can enable hardware monitoring by setting BITERREN (UART_LINCTL[12]) to “1”, if the input pin (UART_RX) state is not equal to the output pin (UART_TX) state in LIN transmitter state that hardware will generate an interrupt to CPU. Software can also monitor the LIN bus transfer state by checking the read back data in UART_DAT register. The following sequence is a program sequence example.

The procedure without software error monitoring in Master mode:

1. Fill Protected Identifier to PID (UART_LINCTL[31:24]).
2. Select the hardware transmission header field including “break field + sync field + protected identifier field” by setting HSEL (UART_LINCTL[23:22]) to “10”.
3. Set SENDH (UART_LINCTL[8]) bit to 1 for requesting header transmission.
4. Wait until SENDH (UART_LINCTL[8]) bit cleared by hardware.
5. Wait until TXEMPTYF (UART_FIFOSTS[28]) set to 1 by hardware.

Note 1: The default setting of break field is 12 dominant bits (break field) and 1 recessive bit break/sync delimiter. Setting BRKFL (UART_LINCTL[19:16]) and BSL (UART_LINCTL[21:20]) to change the LIN break field length and break/sync delimiter length.

Note 2: The default setting of break/sync delimiter length is 1-bit time and the inter-byte spaces default setting is also 1-bit time. Setting BSL (UART_LINCTL[21:20]) and DLY(UART_TOUT[15:8]) can change break/sync delimiter length and inter-byte spaces.

Note 3: If the header includes the “break field, sync field and frame ID field”, software must fill frame ID to PID (UART_LINCTL[31:24]) before trigger header transmission (setting the SENDH (UART_LINCTL[8])). The frame ID parity can be generated by software or hardware depending on IDPEN (UART_LINCTL[9]) setting. If the parity generated by software with IDPEN (UART_LINCTL[9]) is set to ‘0’, software must fill 8 bit data (include 2 bit parity) in this field. If the parity generated by hardware with IDPEN (UART_LINCTL[9]) is set to ‘1’, software fills ID0~ID5 and hardware calculates P0 and P1.

Procedure with software error monitoring in Master mode:

1. Choose the hardware transmission header field to only include “break field” by setting HSEL (UART_LINCTL[23:22])] to ‘00’.
2. Enable break detection function by setting BRKDETEN (UART_LINCTL[10]).
3. Request break + break/sync delimiter transmission by setting the SENDH (UART_LINCTL[8]).
4. Wait until the BRKDETF (UART_LINSTS[8]) flag is set to “1” by hardware.
5. Request sync field transmission by writing 0x55 into UART_DAT register.
6. Wait until the RDAIF (UART_INTSTS[0]) is set to “1” by hardware and then read back the UART_DAT register.
7. Request header frame ID transmission by writing the protected identifier value to UART_DAT register.
8. Wait until the RDAIF (UART_INTSTS[0]) is set to “1” by hardware and then read back the UART_DAT register.

LIN Break and Delimiter Detection

When software enables the break detection function by setting BRKDETEN (UART_LINCTL[10]), the break detection circuit is activated. The break detection circuit is totally independent from the UART receiver.

When the break detection function is enabled, the circuit looks at the input UART_RX pin for a start signal. If UART LIN controller detects consecutive dominant is greater than 11 bits dominant followed by a recessive bit (delimiter), the BRKDETF (UART_LINSTS[8]) flag is set at the end of break field. If the LINIEN (UART_INTEN[8]) bit is set to 1, an interrupt LININT (UART_INTSTS[15]) will be generated. The behavior of the break detection and break flag are shown in Figure 6.27-19.

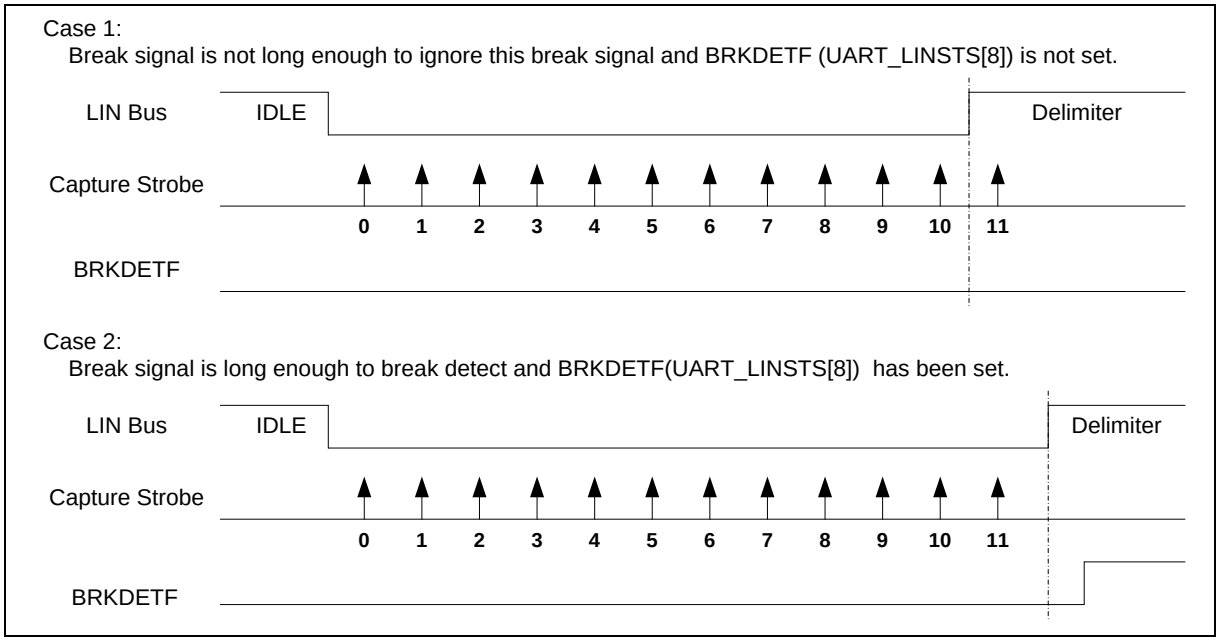


Figure 6.27-19 Break Detection in LIN Mode

LIN Frame ID and Parity Format

The LIN frame ID value in LIN function mode is shown, the frame ID parity can be generated by software or hardware depends on IDPEN (UART_LINCTL[9]).

If the parity generated by hardware (IDPEN (UART_LINCTL[9])=1), user fill ID0~ID5 (UART_LINCTL[29:24]) hardware will calculate P0 (UART_LINCTL[30]) and P1 (UART_LINCTL[31]) otherwise user must filled frame ID and parity in this field.

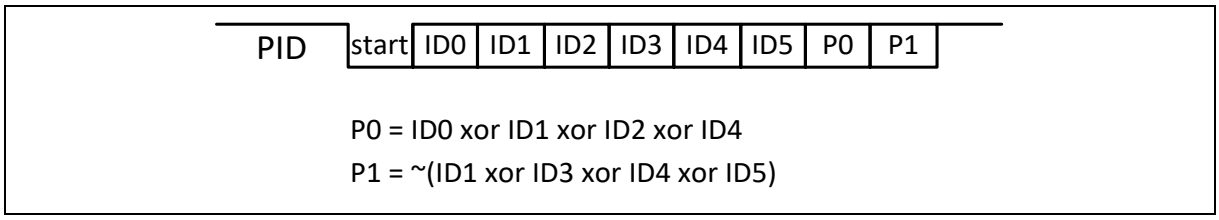


Figure 6.27-20 LIN Frame ID and Parity Format

LIN Slave Mode

The UART Controller supports LIN Slave mode. To enable and initialize the LIN Slave mode, the following steps are necessary:

1. Set the UART_BAUD register to select the desired baud rate.
2. Configure the data length to 8 bits by setting WLS (UART_LINE[1:0]) to '11' and disable parity check by clearing PBE (UART_LINE[3]) bit and configure with one STOP bit by clearing NSB

(UART_LINE[2]) bit.

3. Select LIN function mode by setting FUNCSEL (UART_FUNCSEL[2:0]) to '001'.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) to 1.

LIN Header Reception

According to the LIN protocol, a slave node must wait for a valid header which comes from the master node. Next the slave task will take one of following actions (depend on the master header frame ID value).

- Receive the response.
- Transmit the response.
- Ignore the response and wait for next header.

In LIN Slave mode, user can enable the slave header detection function by setting the SLVHDEN (UART_LINCTL[1]) to detect complete frame header (receive "break field", "sync field" and "frame ID field"). When a LIN header is received, the SLVHDETF (UART_LINSTS[0]) flag will be set. If the LINIEN (UART_INTEN[8]) bit is set to 1, an interrupt will be generated. User can enable the frame ID parity check function by setting IDPEN (UART_LINCTL[9]). If only received frame ID parity is not correct (break and sync filed are correct), the SLVIDPEF (UART_LINSTS[2]) flag is set to '1'. If the LINIEN(UART_INTEN[8]) is set to 1, an interrupt will be generated and SLVHDETF (UART_LINSTS[0]) is set to '1'. User can also put LIN in mute mode by setting MUTE (UART_LINCTL[4]) to '1'. This mode allows detection of headers only (break + sync + frame ID) and prevents the reception of any other characters. In order to avoid bit rate tolerance, the controller supports automatic resynchronization function to avoid clock deviation error, user can enable this feature by setting SLVAREN (UART_LINCTL[2]).

LIN Response Transmission

The LIN slave node can transmit response and receive response. When slave node is the publisher of the response, the slave node sends response by filling data to the UART_DAT register. If the slave node is the subscriber of the response, the slave node receives data from LIN bus.

LIN Header Time-out Error

The LIN slave controller contains a header time-out counter. If the entire header is not received within the maximum time limit of 57 bit times, the header error flag SLVHEF (UART_LINSTS[1]) will be set. The time-out counter is enabled at each break detect edge and stopped in the following conditions.

- A LIN frame ID field has been received.
- The header error flag asserts.
- Writing 1 to the SLVSYNCF (UART_LINSTS[3]) to re-search a new frame header.

Mute Mode and LIN Exit from Mute Mode Condition

In Mute mode, a LIN slave node will not receive any data until specified condition occurred. It allows header detection only and prevents the reception of any other characters. User can enable Mute mode by setting the MUTE (UART_LINCTL[4]) and exiting from Mute mode condition can be selected by HSEL (UART_LINCTL[23:22]).

Note: It is recommended to set LIN slave node to Mute mode after checksum transmission.

The LIN slave controller exiting from Mute mode is described as follows: If HSEL (UART_LINCTL[23:22]) is set to "break field", when LIN slave controller detects a valid LIN break and delimiter, the controller will enable the receiver (exit from Mute mode) and subsequent data (sync data, frame ID data, response data) are received in RX FIFO.

If HSEL (UART_LINCTL[23:22]) is set to "break field and sync field", when the LIN slave controller detects a valid LIN break and delimiter followed by a valid sync field without frame error, the controller

will enable the receiver (exit from mute mode) and subsequent data(ID data, response data) are received in RX FIFO. If HSEL (UART_LINCTL[23:22]) is set to “break field, sync field and ID field”, when the LIN slave controller detects a valid LIN break and delimiter and valid sync field without frame error followed by ID data without frame error and received ID data matched PID (UART_LINCTL[31:24]) value. The controller will enable the receiver (exit from mute mode) and subsequent data (response data) are received in RX FIFO.

Slave Mode Non-automatic Resynchronization (NAR)

User can disable the automatic resynchronization function to fix the communication baud rate. When operating in Non-Automatic Resynchronization mode, software needs some initial process, and the initialization process flow of Non-Automatic Resynchronization mode is shown as follows:

1. Select the desired baud rate by setting the UART_BAUD register.
2. Select LIN function mode by setting FUNCSEL (UART_FUNCSEL[2:0]) to ‘001’.
3. Disable automatic resynchronization function by setting SLVAREN (UART_LINCTL[2]) is set to 0.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) is set to 1.

Slave Mode with Automatic Resynchronization (AR)

In Automatic Resynchronization (AR) mode, the controller will adjust the baud rate generator after each sync field reception. The initialization process flow of Automatic Resynchronization mode is shown as follows:

1. Select the desired baud rate by setting the UART_BAUD register.
2. Select LIN function mode by setting FUNCSEL (UART_FUNCSEL[2:0]) to ‘001’.
3. Enable automatic resynchronization function by setting SLVAREN (UART_LINCTL[2]) to ‘1’.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) is set to ‘1’.

When the automatic resynchronization function is enabled, after each LIN break field, the time duration between five falling edges is sampled on peripheral clock and the result of this measurement is stored in an internal 13-bit register and the UART_BAUD register value will be automatically updated at the end of the fifth falling edge. If the measure timer (13-bit) overflows before five falling edges, then the header error flag SLVHEF (UART_LINSTS[1]) will be set.

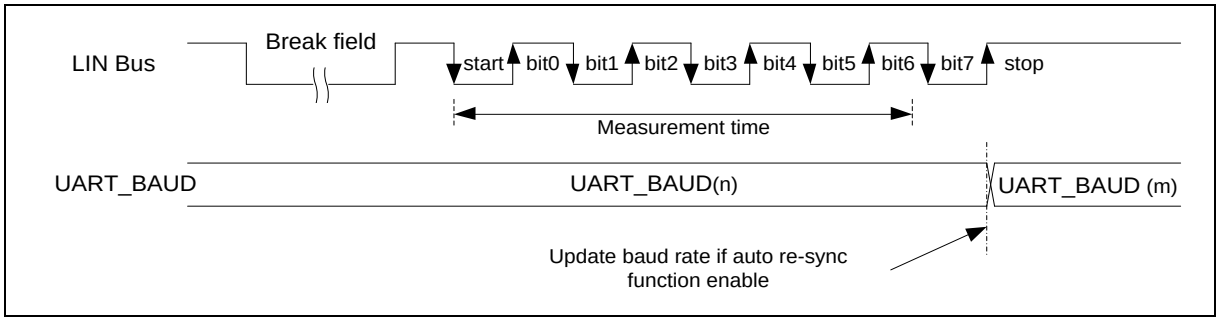


Figure 6.27-21 LIN Sync Field Measurement

When operating in Automatic Resynchronization (AR) mode, software must select the desired baud rate by setting the UART_BAUD register and hardware will store it at internal TEMP_REG register, after each LIN break field, the time duration between five falling edges is sampled on peripheral clock and the result of this measurement is stored in an internal 13-bit register BAUD_LIN and the result will be updated to UART_BAUD register automatically.

To guarantee the transmission baud rate, the baud rate generator must reload the initial value before each new break reception. The initial value is programmed by the application during initialization (TEMP_REG). User can set SLVDUEN (UART_LINCTL[3]) to enable auto reload initial baud rate value

function. If the SLVDUEN (UART_LINCTL[3]) is set, when received the next character, hardware will auto reload the initial value to UART_BAUD, and when the UART_BAUD be updated, the SLVDUEN (UART_LINCTL[3]) will be cleared automatically. The behavior of LIN updated method as shown inFigure 6.27-22.

Note 1: It is recommended to set the SLVDUEN bit before every checksum reception.

Note 2: When a header error is detected, user must write 1 to SLVSYNCF (UART_LINSTS[3]) to re-search new frame header. When writing 1 to it, hardware will reload the initial baud rate TEMP_REG and re-search new frame header.

Note 3: When operating in Automatic Resynchronization mode, the baud rate setting must be operated at mode 2 (BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) must be 1).

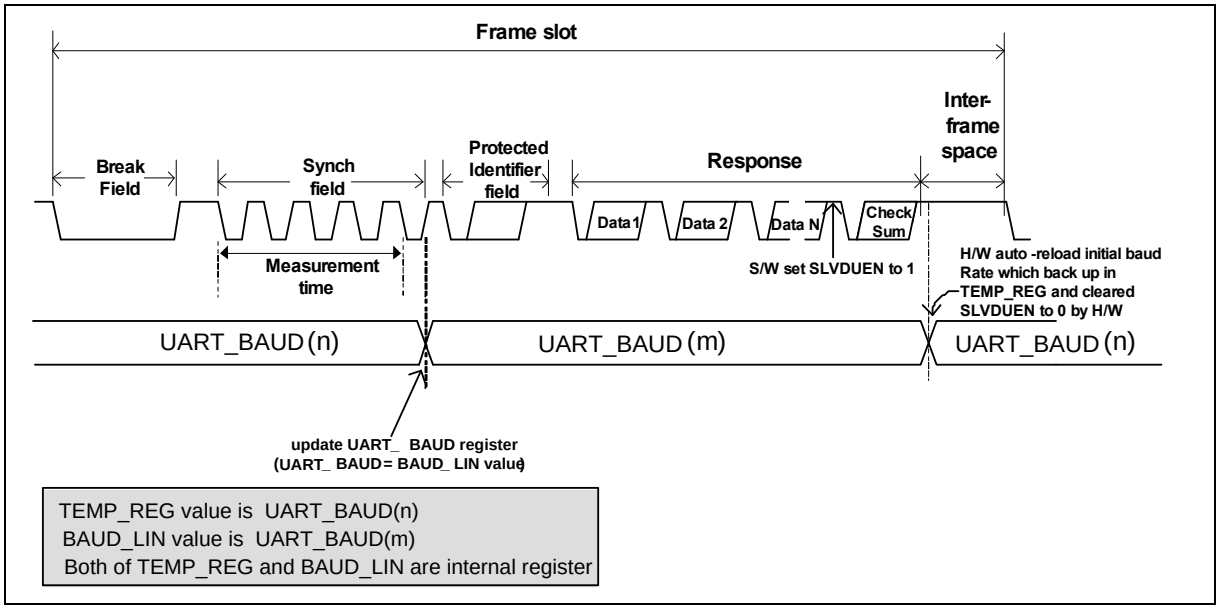


Figure 6.27-22 UART_BAUD Update Sequence in AR mode if SLVDUEN is 1

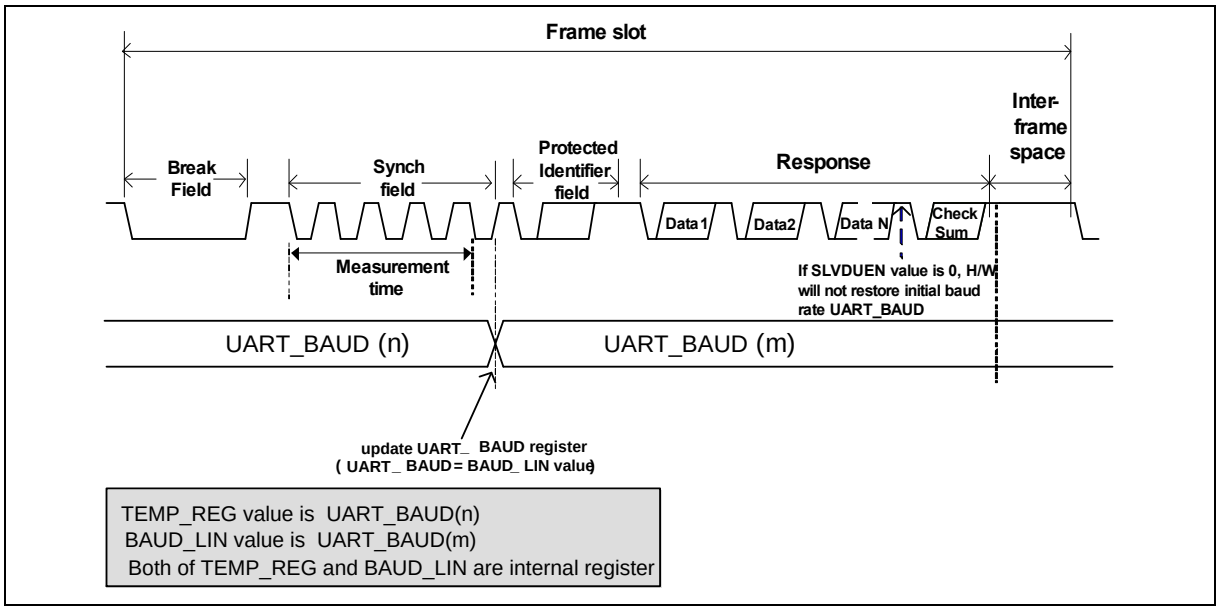


Figure 6.27-23 UART_BAUD Update Sequence in AR mode if SLVDUEN is 0

Deviation Error on the Sync Field

When operating in Automatic Resynchronization mode, the controller will check the deviation error on the sync field. The deviation error is checked by comparing the current baud rate with the received sync field. Two checks are performed in parallel.

Check1: Based on measurement between the first falling edge and the last falling edge of the sync field.

- If the difference is more than 14.84%, the header error flag SLVHEF (UART_LINSTS[1]) will be set.
- If the difference is less than 14.06%, the header error flag SLVHEF (UART_LINSTS[1]) will not be set.
- If the difference is between 14.84% and 14.06%, the header error flag SLVHEF (UART_LINSTS[1]) may either set or not.

Check2: Based on measurement of time between each falling edge of the sync field.

- If the difference is more than 18.75%, the header error flag SLVHEF (UART_LINSTS[1]) will be set.
- If the difference is less than 15.62%, the header error flag SLVHEF (UART_LINSTS[1]) will not be set.
- If the difference is between 18.75% and 15.62%, the header error flag SLVHEF (UART_LINSTS[1]) may either set or not.

Note: The deviation check is based on the current baud rate clock. Therefore, in order to guarantee correct deviation checking, the baud rate must reload the nominal value before each new break reception by setting SLVDUEN (UART_LINCTL[3]) register (It is recommend setting the SLVDUEN (UART_LINCTL[3]) bit before every checksum reception).

LIN Header Error Detection

In LIN Slave function mode, when user enables the header detection function by setting the SLVHDEN (UART_LINCTL[1]), hardware will handle the header detect flow. If the header has an error, the LIN header error flag SLVHEF (UART_LINSTS[1]) will be set and an interrupt is generated if the LINIEN (UART_INTEN[8]) bit is set. When header error is detected, user must reset the detect circuit to re-search a new frame header by writing 1 to SLVSYNCF (UART_LINSTS[3]) to re-search a new frame header.

The LIN header error flag SLVHEF (UART_LINSTS[1]) is set if one of the following conditions occurs:

- Break Delimiter is too short (less than 0.5-bit time).
- Frame error in sync field or Identifier field.
- The sync field data is not 0x55 (Non-Automatic Resynchronization mode).
- The sync field deviation error (With Automatic Resynchronization mode).
- The sync field measure time-out (With Automatic Resynchronization mode).
- LIN header reception time-out.

6.27.5.11 RS-485 Function Mode

Another alternate function of UART controller is RS-485 function (user must set FUNCSEL (UART_FUNCSEL[2:0]) to '011' to enable RS-485 function), and direction control provided by nRTS pin from an asynchronous serial port. The RS-485 transceiver control is implemented by using the nRTS control signal to enable the RS-485 driver. Many characteristics of the RX and TX are the same as UART in RS-485 mode.

The UART controller can be configured as a RS-485 addressable slave or a RS-485 master. RS-485 master transmitter will identify an address character by setting the parity (9^{th} bit) to 1. For data characters, the parity is set to 0. Software can use UART_LINE register to control the 9^{th} bit (When the

PBE, EPE and SPE are set, the 9th bit is transmitted 0 and when PBE and SPE are set and EPE is cleared, the 9th bit is transmitted 1).

The controller supports three operation modes: RS-485 Normal Multidrop Operation Mode (NMM), RS-485 Auto Address Detection Operation Mode (AAD) and RS-485 Auto Direction Control Operation Mode (AUD). Software can choose any operation mode by programming the UART_ALTCTL register, and drive the transfer delay time between the last STOP bit leaving the TX FIFO and the de-assertion of by setting DLY (UART_TOUT[15:8]) register.

RS-485 Normal Multidrop Operation Mode (NMM)

In RS-485 Normal Multidrop Operation Mode (RS485NMM (UART_ALTCTL[8]) = 1), software must first decide the data which before the address byte be detected will be stored in RX FIFO or not. To ignore any data before address byte is detected, set RXOFF (UART_FIFO[8]) to 1 and enable RS485NMM (UART_ALTCTL[8]). The receiver will ignore any data until an address byte is detected (bit 9 = 1) and store the detected address byte data in the RX FIFO. To receive any data before address byte is detected, set RXOFF (UART_FIFO[8]) to 0 and enable RS485NMM (UART_ALTCTL[8]).

If an address byte is detected (bit 9 = 1), it will generate an interrupt to CPU and RXOFF (UART_FIFO[8]) can decide whether accepting the following data bytes are stored in the RX FIFO. If software disables receiver by setting RXOFF (UART_FIFO[8]) register, when a next address byte is detected, the controller will clear the RXOFF (UART_FIFO[8]) bit and the address byte data will be stored in the RX FIFO.

RS-485 Auto Address Detection Operation Mode (AAD)

In RS-485 Auto Address Detection Operation Mode (RS485AAD (UART_ALTCTL[9]) = 1), the receiver will ignore any data until an address byte is detected (bit 9 = 1) and the address byte data matches the ADDRMV (UART_ALTCTL[31:24]) value. The address byte data will be stored in the RX FIFO. The all received byte data will be accepted and stored in the RX FIFO until an address byte data not match the ADDRMV (UART_ALTCTL[31:24]) value.

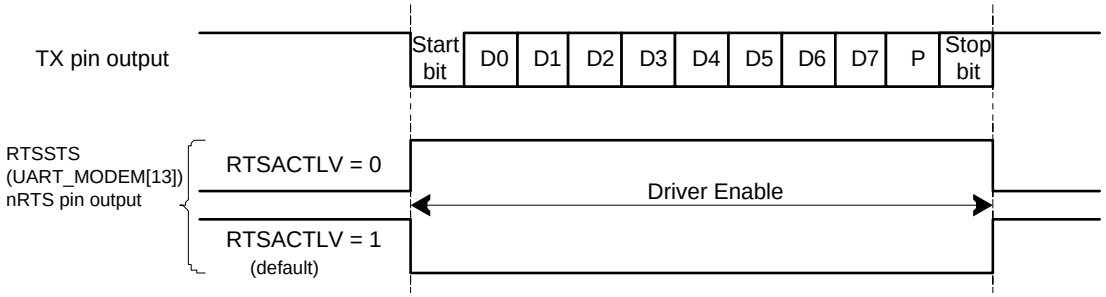
RS-485 Auto Direction Function (AUD)

RS-485 controllers supports RS-485 auto direction control function (RS485AUD (UART_ALTCTL[10]) = 1). The RS-485 transceiver control is implemented by using the nRTS control signal from an asynchronous serial port. The nRTS line is connected to the RS-485 transceiver enable pin such that setting the nRTS line to high (logic 1) enables the RS-485 transceiver. Setting the nRTS line to low (logic 0) puts the transceiver into the tri-state condition. User can set RTSACTLV in UART_MODEM register to change the nRTS driving level.

Figure 6.28-15 demonstrates the RS-485 nRTS driving level in AUD mode. The nRTS pin will be automatically driven during TX data transmission.

Setting RTSACTLV(UART_MODEM[9]) can control nRTS pin output driving level. User can read the RTSSTS(UART_MODEM[13]) bit to get real nRTS pin output voltage logic status.

nRTS pin output status of RS-485 function mode (RS-485 AUD mode enabled)

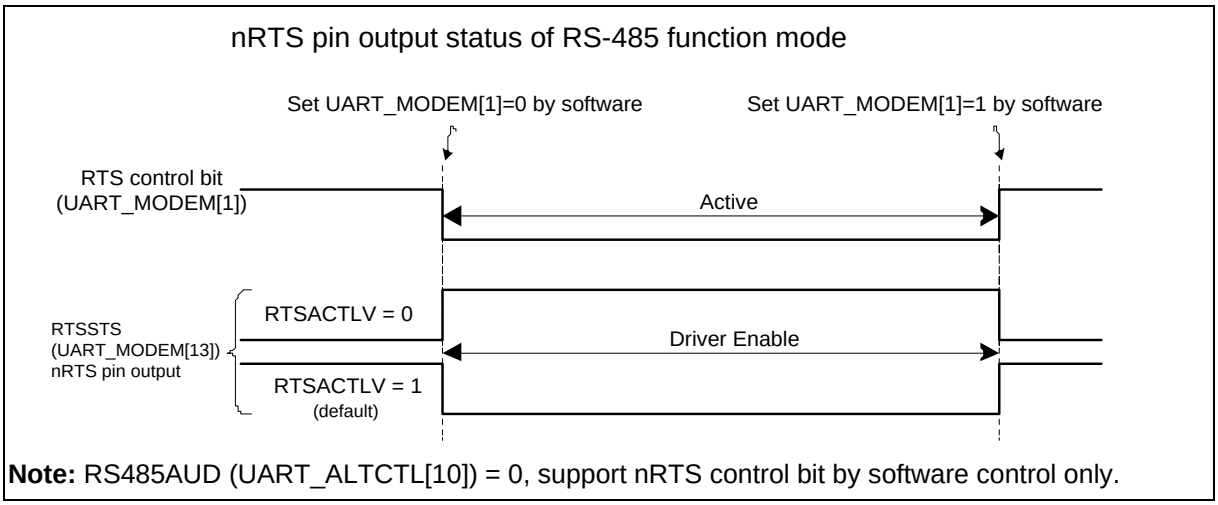


Note: RS485AUD(UART_ALTCTL[10]) = 1, the nRTS pin output by hardware control only.

Figure 6.27-24 RS-485 nRTS Driving Level in Auto Direction Mode

Figure 6.28-16 demonstrates the RS-485 nRTS driving level in software control (RS485AUD (UART_ALTCTL[10])=0). The nRTS driving level is controlled by programing the RTS(UART_MODEM[1]) control bit.

Setting RTSACTLV (UART_MODEM[9]) can control the nRTS pin output is inverse or non-inverse from RTS(UART_MODEM[1]) control bit. User can read the RTSSTS (UART_MODEM[13]) bit to get real nRTS pin output voltage logic status. The structure of RS-485 frame is shown in Figure 6.28-17.

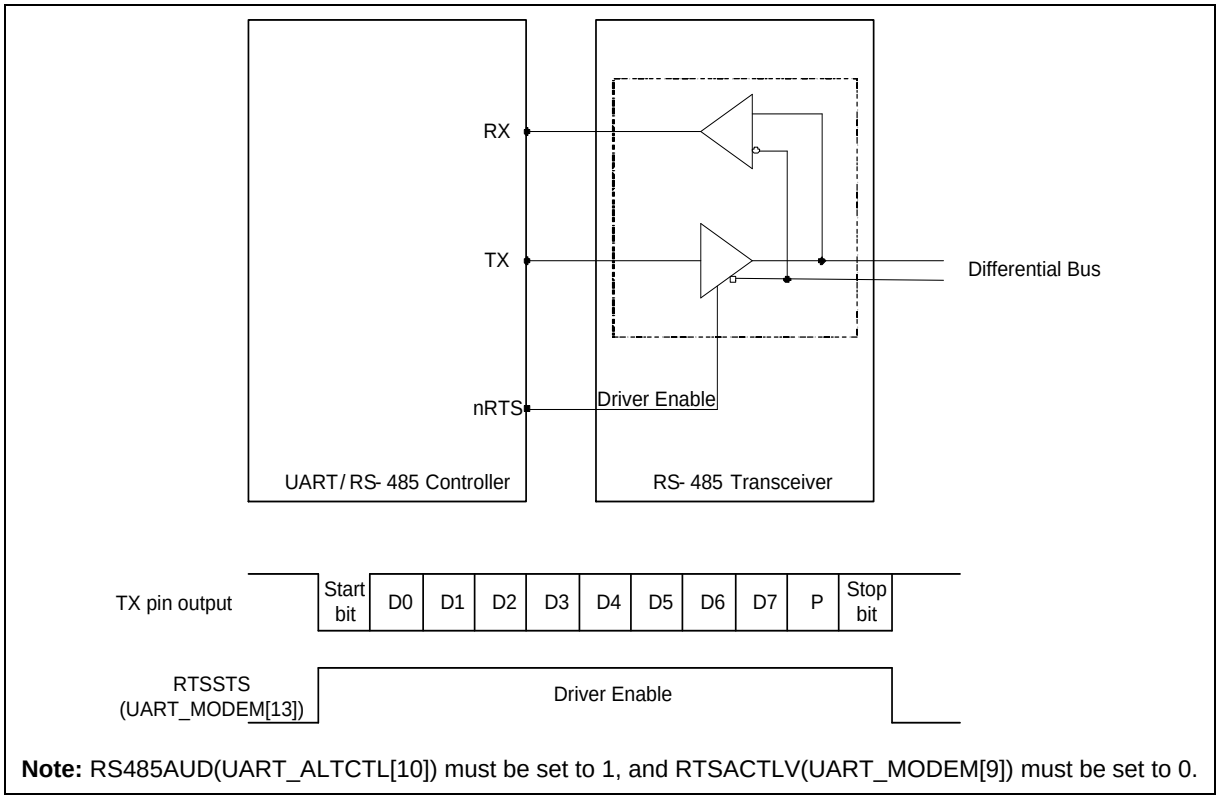


Note: RS485AUD (UART_ALTCTL[10]) = 0, support nRTS control bit by software control only.

Figure 6.27-25 RS-485 nRTS Driving Level with Software Control

Programming Sequence Example:

1. Program FUNCSEL in UART_FUNCSEL to select RS-485 function.
2. Program the RXOFF (UART_FIFO[8]) to determine enable or disable the receiver RS-485 receiver.
3. Program the RS485NMM (UART_ALTCTL[8]) or RS485AAD (UART_ALTCTL[9]) mode.
4. If the RS485AAD (UART_ALTCTL[9]) mode is selected, the ADDRMV (UART_ALTCTL[31:24]) is programmed for auto address match value.
5. Determine auto direction control by programming RS485AUD (UART_ALTCTL[10]).



Note: RS485AUD(UART_ALTCTL[10]) must be set to 1, and RTSACTLV(UART_MODEM[9]) must be set to 0.

Figure 6.27-26 Structure of RS-485 Frame

6.27.5.12 UART Single-wire Half Duplex

The UART controller provides single-wire half duplex function in UART function mode. Setting FUNCSEL (UART_FUNCSEL[2:0]) to '100' to enable the UART Single-wire function. The single-wire bus is idle (RXIDLE (UART_FIFOSTS[29]) = 1) when in RX state. Before writing data to transmit buffer (UART_DAT[7:0]), the bus state should be checked in idle (RXIDLE (UART_FIFOSTS[29]). By writing data to transmit buffer, the bus state transfers to TX state immediately. After the transmission, the bus state transfers from TX state to RX state.

The UART will not allowed to receive data when in single-wire half duplex function TX mode. If nRTS is asserted in TX mode, nRTS will make the docking UART device send out data and cause bus confliction. To reduce the bus confliction, the UART controller supports flow control function and bit error detection but does not support auto-flow control function. The nRTS is automatically inactivated in single-wire TX state. When in TX state, the UART controller will monitor bus state. If the bus state is not equal to TX state, the SWBEIF (UART_INTSTS[16]) is set.

6.27.5.13 PDMA Transfer Function

The UART controller supports PDMA transfer function.

By configuring PDMA parameter and set UART_DAT as the PDMA destination address. When TXPDMAEN (UART_INTEN[14]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

By configuring PDMA parameter and set UART_DAT as the PDMA source address. When RXPDMAEN (UART_INTEN[15]) is set to 1, the controller will start the PDMA reception process. UART controller will issue request to PDMA controller automatically when there is data in the RX FIFO buffer.

6.27.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
UART Base Address: **UARTx_BA = 0x4007_0000 + (0x1000 * x)** **x=0,1,2,3,4,5,6,7**				
UARTx_DAT **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x00	R/W	UART Receive/Transmit Buffer Register	0xXXXX_XXXX
UARTx_INTEN **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000
UARTx_FIFO **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101
UARTx_LINE **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000
UARTx_MODEM **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200
UARTx_MODEMSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110
UARTx_FIFOSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000
UARTx_INTSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002
UARTx_TOUT **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000
UARTx_BAUD **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000
UARTx_IRDA **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040
UARTx_ALTCTL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C
UARTx_FUNCSEL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000
UARTx_LINCTL **x=0,1**	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000
UARTx_LINSTS **x=0,1**	UARTx_BA+0x38	R/W	UART LIN Status Register	0x0000_0000
UARTx_BRCOMP **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000

UARTx_WKCTL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x40	R/W	UART Wake-up Control Register	0x0000_0000
UARTx_WKSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x44	R/W	UART Wake-up Status Register	0x0000_0000
UARTx_DWKCOMP **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x48	R/W	UART Incoming Data Wake-up Compensation Register	0x0000_0000
UARTx_RS485DD **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x4C	R/W	UART RS485 Transceiver Deactivate Delay Register	0x0000_0000

6.27.7 Register Description

UART Receive/Transmit Buffer Register (UART_DAT)

Register	Offset	R/W	Description	Reset Value
UARTx_DAT **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x00	R/W	UART Receive/Transmit Buffer Register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**PARITY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAT							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**PARITY**	**PARITY Bit Receive/Transmit Buffer** Write Operation: By writing to this bit, the PARITY bit will be stored in transmitter FIFO. If PBE (UART_LINE[3]) and PSS (UART_LINE[7]) are set, the UART controller will send out this bit follow the DAT (UART_DAT[7:0]) through the UART_TXD. Read Operation: If PBE (UART_LINE[3]) and PSS (UART_LINE[7]) are enabled, the PARITY bit can be read by this bit. **Note:** This bit has effect only when PBE (UART_LINE[3]) and PSS (UART_LINE[7]) are set.
[7:0]	**DAT**	**Data Receive/Transmit Buffer** Write Operation: By writing one byte to this register, the data byte will be stored in transmitter FIFO. The UART controller will send out the data stored in transmitter FIFO top location through the UART_TXD. Read Operation: By reading this register, the UART controller will return an 8-bit data received from receiver FIFO.

UART Interrupt Enable Register (UART_INTEN)

Register	Offset	R/W	Description	Reset Value
UARTx_INTEN **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	TXENDIEN	Reserved			ABRIEN	Reserved	SWBEIEN
15	14	13	12	11	10	9	8
RXPDMAEN	TXPDMAEN	ATOCTSEN	ATORTSEN	TOCNTEN	Reserved		LINIEN
7	6	5	4	3	2	1	0
Reserved	WKIEN	BUFERRIEN	RXTOIEN	MODEMIEN	RLSIEN	THREIEN	RDAIEN

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22]	**TXENDIEN**	**Transmitter Empty Interrupt Enable Bit** If TXENDIEN (UART_INTEN[22]) is enabled, the Transmitter Empty interrupt TXENDINT (UART_INTSTS[30]) will be generated when TXENDIF (UART_INTSTS[22]) is set (TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted). 0 = Transmitter empty interrupt Disabled. 1 = Transmitter empty interrupt Enabled.
[21:19]	**Reserved**	Reserved.
[18]	**ABRIEN**	**Auto-baud Rate Interrupt Enable Bit** 0 = Auto-baud rate interrupt Disabled. 1 = Auto-baud rate interrupt Enabled.
[17]	**Reserved**	Reserved.
[16]	**SWBEIEN**	**Single-wire Bit Error Detection Interrupt Enable Bit** Set this bit, the Single-wire Half Duplex Bit Error Detection Interrupt SWBEINT(UART_INTSTS[24]) is generated when Single-wire Bit Error Detection SWBEIF(UART_INTSTS[16]) is set. 0 = Single-wire Bit Error Detect Interrupt Disabled. 1 = Single-wire Bit Error Detect Interrupt Enabled. **Note:** This bit is valid when FUNCSEL (UART_FUNCSEL[2:0]) is select UART Single-wire mode.
[15]	**RXPDMAEN**	**RX PDMA Enable Bit** This bit can enable or disable RX PDMA service. 0 = RX PDMA Disabled. 1 = RX PDMA Enabled. **Note:** If RLSIEN (UART_INTEN[2]) is enabled and HWRLSINT (UART_INTSTS[26]) is set to 1, the RLS (Receive Line Status) Interrupt is caused. If RLS interrupt is caused by Break Error Flag BIF(UART_FIFOSTS[6]), Frame Error Flag FEF(UART_FIFO[5]) or Parity Error Flag PEF(UART_FIFOSTS[4]), UART PDMA receive request operation is stopped. Clear Break Error Flag BIF or Frame Error Flag FEF or Parity Error Flag PEF by

		writing “1” to corresponding BIF, FEF and PEF to make UART PDMA receive request operation continue.
[14]	**TXPDMAEN**	**TX PDMA Enable Bit** 0 = TX PDMA Disabled. 1 = TX PDMA Enabled. **Note:** If RLSIEN (UART_INTEN[2]) is enabled and HWRLSINT (UART_INTSTS[26]) is set to 1, the RLS (Receive Line Status) Interrupt is caused. If RLS interrupt is caused by Break Error Flag BIF(UART_FIFOSTS[6]), Frame Error Flag FEF(UART_FIFO[5]) or Parity Error Flag PEF(UART_FIFOSTS[4]), UART PDMA transmit request operation is stopped. Clear Break Error Flag BIF or Frame Error Flag FEF or Parity Error Flag PEF by writing “1” to corresponding BIF, FEF and PEF to make UART PDMA transmit request operation continue.
[13]	**ATOCTSEN**	**nCTS Auto-flow Control Enable Bit** 0 = nCTS auto-flow control Disabled. 1 = nCTS auto-flow control Enabled. **Note:** When nCTS auto-flow is enabled, the UART will send data to external device if nCTS input assert (UART will not send data to device until nCTS is asserted).
[12]	**ATORTSEN**	**nRTS Auto-flow Control Enable Bit** 0 = nRTS auto-flow control Disabled. 1 = nRTS auto-flow control Enabled. **Note:** When nRTS auto-flow is enabled, if the number of bytes in the RX FIFO equals the RTSTRGLV (UART_FIFO[19:16]), the UART will de-assert nRTS signal.
[11]	**TOCNTEN**	**Receive Buffer Time-out Counter Enable Bit** 0 = Receive Buffer Time-out counter Disabled. 1 = Receive Buffer Time-out counter Enabled.
[10:9]	**Reserved**	Reserved.
[8]	**LINIEN**	**LIN Bus Interrupt Enable Bit** 0 = LIN bus interrupt Disabled. 1 = LIN bus interrupt Enabled. **Note:** This bit is used for LIN function mode.
[7]	**Reserved**	Reserved.
[6]	**WKIEN**	**Wake-up Interrupt Enable Bit** 0 = Wake-up Interrupt Disabled. 1 = Wake-up Interrupt Enabled.
[5]	**BUFERRIEN**	**Buffer Error Interrupt Enable Bit** 0 = Buffer error interrupt Disabled. 1 = Buffer error interrupt Enabled.
[4]	**RXTOIEN**	**RX Time-out Interrupt Enable Bit** 0 = RX time-out interrupt Disabled. 1 = RX time-out interrupt Enabled.
[3]	**MODEMIEN**	**Modem Status Interrupt Enable Bit** 0 = Modem status interrupt Disabled. 1 = Modem status interrupt Enabled.
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable Bit** 0 = Receive Line Status interrupt Disabled. 1 = Receive Line Status interrupt Enabled.
[1]	**THREIEN**	**Transmit Holding Register Empty Interrupt Enable Bit**

		0 = Transmit holding register empty interrupt Disabled. 1 = Transmit holding register empty interrupt Enabled.
[0]	**RDAIEN**	**Receive Data Available Interrupt Enable Bit** 0 = Receive data available interrupt Disabled. 1 = Receive data available interrupt Enabled.

UART FIFO Control Register (UART_FIFO)

Register	Offset	R/W	Description	Reset Value
UARTx_FIFO **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**RTSTRGLV**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**RXOFF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RFITL				**Reserved**	**TXRST**	**RXRST**	**Reserved**

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**RTSTRGLV**	**nRTS Trigger Level for Auto-flow Control** 0000 = nRTS Trigger Level is 1 byte. 0001 = nRTS Trigger Level is 4 bytes. 0010 = nRTS Trigger Level is 8 bytes. 0011 = nRTS Trigger Level is 14 bytes. Others = Reserved. **Note:** This field is used for auto nRTS flow control.
[15:9]	**Reserved**	Reserved.
[8]	**RXOFF**	**Receiver Disable Bit** The receiver is disabled or not (set 1 to disable receiver). 0 = Receiver Enabled. 1 = Receiver Disabled. **Note:** This bit is used for RS-485 Normal Multi-drop mode. It should be programmed before RS485NMM (UART_ALTCTL[8]) is programmed.
[7:4]	**RFITL**	**RX FIFO Interrupt Trigger Level** When the number of bytes in the receive FIFO equals the RFITL, the RDAIF (UART_INTSTS[0]) will be set (if RDAIEN (UART_INTEN[0]) enabled, and an interrupt will be generated). 0000 = RX FIFO Interrupt Trigger Level is 1 byte. 0001 = RX FIFO Interrupt Trigger Level is 4 bytes. 0010 = RX FIFO Interrupt Trigger Level is 8 bytes. 0011 = RX FIFO Interrupt Trigger Level is 14 bytes. Others = Reserved.
[3]	**Reserved**	Reserved.
[2]	**TXRST**	**TX Field Software Reset** When TXRST (UART_FIFO[2]) is set, all the byte in the transmit FIFO and TX internal

		state machine are cleared. 0 = No effect. 1 = Reset the TX internal state machine and pointers. **Note 1:** This bit will automatically clear at least 3 UART peripheral clock cycles. **Note 2:** Before setting this bit, it should wait for the TXEMPTYF (UART_FIFOSTS[28]) be set.
[1]	**RXRST**	**RX Field Software Reset** When RXRST (UART_FIFO[1]) is set, all the byte in the receiver FIFO and RX internal state machine are cleared. 0 = No effect. 1 = Reset the RX internal state machine and pointers. **Note 1:** This bit will automatically clear at least 3 UART peripheral clock cycles. **Note 2:** Before setting this bit, it should wait for the RXIDLE (UART_FIFOSTS[29]) be set.
[0]	**Reserved**	Reserved.

UART Line Control Register (UART_LINE)

Register	Offset	R/W	Description	Reset Value
UARTx_LINE **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**RXDINV**	**TXDINV**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSS	**BCB**	**SPE**	**EPE**	**PBE**	**NSB**	**WLS**	

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**RXDINV**	**RX Data Inverted** 0 = Received data signal inverted Disabled. 1 = Received data signal inverted Enabled. **Note 1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[2:0]) is select UART, LIN or RS485 function.
[8]	**TXDINV**	**TX Data Inverted** 0 = Transmitted data signal inverted Disabled. 1 = Transmitted data signal inverted Enabled. **Note 1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[2:0]) is select UART, LIN or RS485 function.
[7]	**PSS**	**PARITY Bit Source Selection** The PARITY bit can be selected to be generated and checked automatically or by software. 0 = PARITY bit is generated by EPE (UART_LINE[4]) and SPE (UART_LINE[5]) setting and checked automatically. 1 = PARITY bit generated and checked by software. **Note 1:** This bit has effect only when PBE (UART_LINE[3]) is set. **Note 2:** If PSS is 0, the PARITY bit is transmitted and checked automatically. If PSS is 1, the transmitted PARITY bit value can be determined by writing PARITY (UART_DAT[8]) and the PARITY bit can be read by reading PARITY (UART_DAT[8]).
[6]	**BCB**	**Break Control Bit** 0 = Break Control Disabled.

		1 = Break Control Enabled. **Note:** When this bit is set to logic 1, the transmitted serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX line and has no effect on the transmitter logic.
[5]	**SPE**	**Stick Parity Enable Bit** 0 = Stick parity Disabled. 1 = Stick parity Enabled. **Note:** If PBE (UART_LINE[3]) and EPE (UART_LINE[4]) are logic 1, the PARITY bit is transmitted and checked as logic 0. If PBE (UART_LINE[3]) is 1 and EPE (UART_LINE[4]) is 0 then the PARITY bit is transmitted and checked as 1.
[4]	**EPE**	**Even Parity Enable Bit** 0 = Odd number of logic 1's is transmitted and checked in each word. 1 = Even number of logic 1's is transmitted and checked in each word. **Note:** This bit has effect only when PBE (UART_LINE[3]) is set.
[3]	**PBE**	**PARITY Bit Enable Bit** 0 = PARITY bit generated Disabled. 1 = PARITY bit generated Enabled. **Note:** PARITY bit is generated on each outgoing character and is checked on each incoming data.
[2]	**NSB**	**Number of "STOP Bit"** 0 = One "STOP bit" is generated in the transmitted data. 1 = When select 5-bit word length, 1.5 "STOP bit" is generated in the transmitted data. When select 6-, 7- and 8-bit word length, 2 "STOP bit" is generated in the transmitted data.
[1:0]	**WLS**	**Word Length Selection** This field sets UART word length. 00 = 5 bits. 01 = 6 bits. 10 = 7 bits. 11 = 8 bits.

UART Modem Control Register (UART_MODEM)

Register	Offset	R/W	Description	Reset Value
UARTx_MODEM **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**RTSSTS**	**Reserved**			**RTSACTLV**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**RTS**	**Reserved**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**RTSSTS**	**nRTS Pin Status (Read Only)** This bit mirror from nRTS pin output of voltage logic status. 0 = nRTS pin output is low level voltage logic state. 1 = nRTS pin output is high level voltage logic state.
[12:10]	**Reserved**	Reserved.
[9]	**RTSACTLV**	**nRTS Pin Active Level** This bit defines the active level state of nRTS pin output. 0 = nRTS pin output is high level active. 1 = nRTS pin output is low level active. (Default) **Note 1:** Refer to Figure 6.28-13 and Figure 6.28-14 for UART function mode. **Note 2:** Refer to Figure 6.28-15 and Figure 6.28-16 for RS-485 function mode. **Note 3:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller.
[8:2]	**Reserved**	Reserved.
[1]	**RTS**	**nRTS Signal Control** This bit is direct control internal nRTS (Request-to-send) signal active or not, and then drive the nRTS pin output with RTSACTLV bit configuration. 0 = nRTS signal is active. 1 = nRTS signal is inactive. **Note 1:** The nRTS signal control bit is not effective when nRTS auto-flow control is enabled in UART function mode. **Note 2:** The nRTS signal control bit is not effective when RS-485 auto direction mode (AUD) is enabled in RS-485 function mode. **Note 3:** Single-wire mode is support this feature.
[0]	**Reserved**	Reserved.

UART Modem Status Register (UART_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
UARTx_MODEMSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**CTSACTLV**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**CTSSTS**	**Reserved**			**CTSDETF**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CTSACTLV**	**nCTS Pin Active Level** This bit defines the active level state of nCTS pin input. 0 = nCTS pin input is high level active. 1 = nCTS pin input is low level active. (Default) **Note:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller.
[7:5]	**Reserved**	Reserved.
[4]	**CTSSTS**	**nCTS Pin Status (Read Only)** This bit mirror from nCTS pin input of voltage logic status. 0 = nCTS pin input is low level voltage logic state. 1 = nCTS pin input is high level voltage logic state. **Note:** This bit echoes when UART controller peripheral clock is enabled, and nCTS multi-function port is selected.
[3:1]	**Reserved**	Reserved.
[0]	**CTSDETF**	**Detect nCTS State Change Flag** This bit is set whenever nCTS input has change state, and it will generate Modem interrupt to CPU when MODEMIEN (UART_INTEN[3]) is set to 1. 0 = nCTS input has not change state. 1 = nCTS input has change state. **Note:** This bit can be cleared by writing “1” to it.

UART FIFO Status Register (UART_FIFOSTS)

Register	Offset	R/W	Description	Reset Value
UARTx_FIFOSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000

31	30	29	28	27	26	25	24
TXRXACT	Reserved	RXIDLE	TXEMPTYF	Reserved			TXOVIF
23	22	21	20	19	18	17	16
TXFULL	TXEMPTY	TXPTR					
15	14	13	12	11	10	9	8
RXFULL	RXEMPTY	RXPTR					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	ADDRDETF	ABRDTOIF	ABRDIF	RXOVIF

Bits	Description	
[31]	**TXRXACT**	**TX and RX Active Status (Read Only)** This bit indicates TX and RX are active or inactive. 0 = TX and RX are inactive. 1 = TX and RX are active. (Default) **Note:** When TXRXDIS (UART_FUNCSEL[3]) is set and both TX and RX are in idle state, this bit is cleared. The UART controller cannot transmit or receive data at this moment. Otherwise this bit is set.
[30]	**Reserved**	Reserved.
[29]	**RXIDLE**	**RX Idle Status (Read Only)** This bit is set by hardware when RX is idle. 0 = RX is busy. 1 = RX is idle. (Default)
[28]	**TXEMPTYF**	**Transmitter Empty Flag (Read Only)** This bit is set by hardware when TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted. 0 = TX FIFO is not empty or the STOP bit of the last byte has been not transmitted. 1 = TX FIFO is empty and the STOP bit of the last byte has been transmitted. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[27:25]	**Reserved**	Reserved.
[24]	**TXOVIF**	**TX Overflow Error Interrupt Flag** If TX FIFO (UART_DAT) is full, an additional write to UART_DAT will cause this bit to logic 1. 0 = TX FIFO is not overflow. 1 = TX FIFO is overflow. **Note:** This bit can be cleared by writing “1” to it.
[23]	**TXFULL**	**Transmitter FIFO Full (Read Only)**

		This bit indicates TX FIFO full or not. 0 = TX FIFO is not full. 1 = TX FIFO is full. **Note:** This bit is set when the number of usage in TX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[22]	**TXEMPTY**	**Transmitter FIFO Empty (Read Only)** This bit indicates TX FIFO empty or not. 0 = TX FIFO is not empty. 1 = TX FIFO is empty. **Note:** When the last byte of TX FIFO has been transferred to Transmitter Shift Register, hardware sets this bit high. It will be cleared when writing data into UART_DAT (TX FIFO not empty).
[21:16]	**TXPTR**	**TX FIFO Pointer (Read Only)** This field indicates the TX FIFO Buffer Pointer. When CPU writes one byte into UART_DAT, TXPTR increases one. When one byte of TX FIFO is transferred to Transmitter Shift Register, TXPTR decreases one. The Maximum value shown in TXPTR is 15. When the using level of TX FIFO Buffer equal to 16, the TXFULL bit is set to 1 and TXPTR will show 0. As one byte of TX FIFO is transferred to Transmitter Shift Register, the TXFULL bit is cleared to 0 and TXPTR will show 15.
[15]	**RXFULL**	**Receiver FIFO Full (Read Only)** This bit initiates RX FIFO full or not. 0 = RX FIFO is not full. 1 = RX FIFO is full. **Note:** This bit is set when the number of usage in RX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[14]	**RXEMPTY**	**Receiver FIFO Empty (Read Only)** This bit initiate RX FIFO empty or not. 0 = RX FIFO is not empty. 1 = RX FIFO is empty. **Note:** When the last byte of RX FIFO has been read by CPU, hardware sets this bit high. It will be cleared when UART receives any new data.
[13:8]	**RXPTR**	**RX FIFO Pointer (Read Only)** This field indicates the RX FIFO Buffer Pointer. When UART receives one byte from external device, RXPTR increases one. When one byte of RX FIFO is read by CPU, RXPTR decreases one. The Maximum value shown in RXPTR is 15. When the using level of RX FIFO Buffer equal to 16, the RXFULL bit is set to 1 and RXPTR will show 0. As one byte of RX FIFO is read by CPU, the RXFULL bit is cleared to 0 and RXPTR will show 15.
[7]	**Reserved**	Reserved.
[6]	**BIF**	**Break Interrupt Flag** This bit is set to logic 1 whenever the received data input (RX) is held in the “spacing state” (logic 0) for longer than a full word transmission time (that is, the total time of “START bit” + data bits + parity + STOP bits). 0 = No Break interrupt is generated. 1 = Break interrupt is generated. **Note:** This bit can be cleared by writing “1” to it.
[5]	**FEF**	**Framing Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “STOP bit” (that is, the STOP bit following the last data bit or PARITY bit is detected as logic 0). 0 = No framing error is generated.

		1 = Framing error is generated. **Note:** This bit can be cleared by writing “1” to it.
[4]	**PEF**	**Parity Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “PARITY bit”. 0 = No parity error is generated. 1 = Parity error is generated. **Note:** This bit can be cleared by writing “1” to it.
[3]	**ADDRDETF**	**RS-485 Address Byte Detect Flag** 0 = Receiver detects a data that is not an address bit (bit 9 =’0’). 1 = Receiver detects a data that is an address bit (bit 9 =’1’). **Note 1:** This field is used for RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1 to enable Address detection mode. **Note 2:** This bit can be cleared by writing “1” to it.
[2]	**ABRDTOIF**	**Auto-baud Rate Detect Time-out Interrupt Flag** This bit is set to logic “1” in Auto-baud Rate Detect mode when the baud rate counter is overflow. 0 = Auto-baud rate counter is underflow. 1 = Auto-baud rate counter is overflow. **Note:** This bit can be cleared by writing “1” to it.
[1]	**ABRDIF**	**Auto-baud Rate Detect Interrupt Flag** This bit is set to logic “1” when auto-baud rate detect function is finished. 0 = Auto-baud rate detect function is not finished. 1 = Auto-baud rate detect function is finished. **Note:** This bit can be cleared by writing “1” to it.
[0]	**RXOVIF**	**RX Overflow Error Interrupt Flag** This bit is set when RX FIFO overflow. If the number of bytes of received data is greater than RX_FIFO (UART_DAT) size 16 bytes, this bit will be set. 0 = RX FIFO is not overflow. 1 = RX FIFO is overflow. **Note:** This bit can be cleared by writing “1” to it.

UART Interrupt Status Register (UART_INTSTS)

Register	Offset	R/W	Description	Reset Value
UARTx_INTSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002

31	30	29	28	27	26	25	24
ABRINT	TXENDINT	HWBUFEINT	HWTOINT	HWMODINT	HWRLSINT	Reserved	SWBEINT
23	22	21	20	19	18	17	16
Reserved	TXENDIF	HWBUFEIF	HWTOIF	HWMODIF	HWRLSIF	Reserved	SWBEIF
15	14	13	12	11	10	9	8
LININT	WKINT	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THREINT	RDAINT
7	6	5	4	3	2	1	0
LINIF	WKIF	BUFERRIF	RXTOIF	MODEMIF	RLSIF	THREIF	RDAIF

Bits	Description	
[31]	**ABRINT**	**Auto-baud Rate Interrupt Indicator (Read Only)** This bit is set if ABRIEN (UART_INTEN[18]) and ABRIF (UART_ALTCTL[17]) are both set to 1. 0 = No Auto-baud Rate interrupt is generated. 1 = The Auto-baud Rate interrupt is generated.
[30]	**TXENDINT**	**Transmitter Empty Interrupt Indicator (Read Only)** This bit is set if TXENDIEN (UART_INTEN[22]) and TXENDIF(UART_INTSTS[22]) are both set to 1. 0 = No Transmitter Empty interrupt is generated. 1 = Transmitter Empty interrupt is generated.
[29]	**HWBUFEINT**	**PDMA Mode Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN (UART_INTEN[5]) and HWBUFEIF (UART_INTSTS[21]) are both set to 1. 0 = No buffer error interrupt is generated in PDMA mode. 1 = Buffer error interrupt is generated in PDMA mode.
[28]	**HWTOINT**	**PDMA Mode RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (UART_INTEN[4]) and HWTOIF(UART_INTSTS[20]) are both set to 1. 0 = No RX time-out interrupt is generated in PDMA mode. 1 = RX time-out interrupt is generated in PDMA mode.
[27]	**HWMODINT**	**PDMA Mode MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN (UART_INTEN[3]) and HWMODIF(UART_INTSTS[19]) are both set to 1. 0 = No Modem interrupt is generated in PDMA mode. 1 = Modem interrupt is generated in PDMA mode.
[26]	**HWRLSINT**	**PDMA Mode Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (UART_INTEN[2]) and HWRLSIF(UART_INTSTS[18]) are both set to 1.

		0 = No RLS interrupt is generated in PDMA mode. 1 = RLS interrupt is generated in PDMA mode.
[25]	**Reserved**	Reserved.
[24]	**SWBEINT**	**Single-wire Bit Error Detect Interrupt Indicator (Read Only)** This bit is set if SWBEIEN (UART_INTEN[16]) and SWBEIF (UART_INTSTS[16]) are both set to 1. 0 = No Single-wire Bit Error Detection Interrupt generated. 1 = Single-wire Bit Error Detection Interrupt generated.
[23]	**Reserved**	Reserved.
[22]	**TXENDIF**	**Transmitter Empty Interrupt Flag** This bit is set when TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted (TXEMPTYF (UART_FIFOSTS[28]) is set). If TXENDIEN (UART_INTEN[22]) is enabled, the Transmitter Empty interrupt will be generated. 0 = No transmitter empty interrupt flag is generated. 1 = Transmitter empty interrupt flag is generated. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[21]	**HWBUFEIF**	**PDMA Mode Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX or RX FIFO overflows (TXOVIF (UART_FIFOSTS[24]) or RXOVIF (UART_FIFOSTS[0]) is set). When BUFERRIF (UART_INTSTS[5]) is set, the transfer maybe is not correct. If BUFERRIEN (UART_INTEN[5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated in PDMA mode. 1 = Buffer error interrupt flag is generated in PDMA mode. **Note:** This bit is cleared when both TXOVIF (UART_FIFOSTS[24]]) and RXOVIF (UART_FIFOSTS[0]) are cleared.
[20]	**HWTOIF**	**PDMA Mode RX Time-out Interrupt Flag (Read Only)** This bit is set when the RX FIFO is not empty and no activities occurred in the RX FIFO and the time-out counter equal to TOIC (UART_TOUT[7:0]). If RXTOIEN (UART_INTEN[4]) is enabled, the RX time-out interrupt will be generated. 0 = No RX time-out interrupt flag is generated in PDMA mode. 1 = RX time-out interrupt flag is generated in PDMA mode. **Note:** This bit is read only and user can read UART_DAT (RX is in active) to clear it.
[19]	**HWMODIF**	**PDMA Mode MODEM Interrupt Flag (Read Only)** This bit is set when the nCTS pin has state change (CTSDETF (UART_MODEMSTS[0] =1)). If MODEMIEN (UART_INTEN[3]) is enabled, the Modem interrupt will be generated. 0 = No Modem interrupt flag is generated in PDMA mode. 1 = Modem interrupt flag is generated in PDMA mode. **Note:** This bit is read only and reset to 0 when the bit CTSDETF (UART_MODEMSTS[0]) is cleared by writing 1 on CTSDETF (UART_MODEMSTS[0]).
[18]	**HWRLSIF**	**PDMA Mode Receive Line Status Flag (Read Only)** This bit is set when the RX receive data have parity error, frame error or break error (at least one of 3 bits, BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]) and PEF (UART_FIFOSTS[4]) is set). If RLSIEN (UART_INTEN[2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated in PDMA mode. 1 = RLS interrupt flag is generated in PDMA mode. **Note 1:** In RS-485 function mode, this field include “receiver detect any address byte received address byte character (bit9 = ‘1’) bit". **Note 2:** In UART function mode, this bit is read only and reset to 0 when all bits of BIF(UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]) and PEF(UART_FIFOSTS[4]) are

		cleared. **Note 3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF(UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]), PEF(UART_FIFOSTS[4]) and ADDRDETF (UART_FIFOSTS[3]) are cleared.
[17]	**Reserved**	Reserved.
[16]	**SWBEIF**	**Single-wire Bit Error Detection Interrupt Flag** This bit is set when the single wire bus state not equals to UART controller TX state in Single-wire mode. 0 = No single-wire bit error detection interrupt flag is generated. 1 = Single-wire bit error detection interrupt flag is generated. **Note 1:** This bit is active when FUNCSEL (UART_FUNCSEL[2:0]) is select UART Single-wire mode. **Note 2:** This bit can be cleared by writing “1” to it.
[15]	**LININT**	**LIN Bus Interrupt Indicator (Read Only)** This bit is set if LINIEN (UART_INTEN[8]) and LINIF(UART_INTSTS[7]) are both set to 1. 0 = No LIN Bus interrupt is generated. 1 = The LIN Bus interrupt is generated.
[14]	**WKINT**	**UART Wake-up Interrupt Indicator (Read Only)** This bit is set if WKIEN (UART_INTEN[6]) and WKIF (UART_INTSTS[6]) are both set to 1. 0 = No UART wake-up interrupt is generated. 1 = UART wake-up interrupt is generated.
[13]	**BUFERRINT**	**Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN(UART_INTEN[5]) and BUFERRIF(UART_ INTSTS[5]) are both set to 1. 0 = No buffer error interrupt is generated. 1 = Buffer error interrupt is generated.
[12]	**RXTOINT**	**RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (UART_INTEN[4]) and RXTOIF(UART_INTSTS[4]) are both set to 1. 0 = No RX time-out interrupt is generated. 1 = RX time-out interrupt is generated.
[11]	**MODEMINT**	**MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN(UART_INTEN[3]) and MODEMIF(UART_INTSTS[3]) are both set to 1 0 = No Modem interrupt is generated. 1 = Modem interrupt is generated..
[10]	**RLSINT**	**Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (UART_INTEN[2]) and RLSIF(UART_INTSTS[2]) are both set to 1. 0 = No RLS interrupt is generated. 1 = RLS interrupt is generated.
[9]	**THREINT**	**Transmit Holding Register Empty Interrupt Indicator (Read Only)** This bit is set if THREIEN (UART_INTEN[1]) and THREIF(UART_INTSTS[1]) are both set to 1. 0 = No THRE interrupt is generated. 1 = THRE interrupt is generated.

[8]	**RDAINT**	**Receive Data Available Interrupt Indicator (Read Only)** This bit is set if RDAIEN (UART_INTEN[0]) and RDAIF (UART_INTSTS[0]) are both set to 1. 0 = No RDA interrupt is generated. 1 = RDA interrupt is generated.
[7]	**LINIF**	**LIN Bus Interrupt Flag** This bit is set when LIN slave header detect (SLVHDETF (UART_LINSTS[0] =1)), LIN break detect (BRKDETF(UART_LINSTS[8]=1)), bit error detect (BITEF(UART_LINSTS[9]=1)), LIN slave ID parity error (SLVIDPEF(UART_LINSTS[2] = 1)) or LIN slave header error detect (SLVHEF (UART_LINSTS[1])). If LINIEN (UART_INTEN[8]) is enabled the LIN interrupt will be generated. 0 = None of SLVHDETF, BRKDETF, BITEF, SLVIDPEF and SLVHEF is generated. 1 = At least one of SLVHDETF, BRKDETF, BITEF, SLVIDPEF and SLVHEF is generated. **Note:** This bit is cleared when SLVHDETF(UART_LINSTS[0]), BRKDETF(UART_LINSTS[8]), BITEF(UART_LINSTS[9]), SLVIDPEF (UART_LINSTS[2]) and SLVHEF(UART_LINSTS[1]) all are cleared and software writing '1' to LINIF(UART_INTSTS[7]).
[6]	**WKIF**	**UART Wake-up Interrupt Flag (Read Only)** This bit is set when TOUTWKF (UART_WKSTS[4]), RS485WKF (UART_WKSTS[3]), RFRTWKF (UART_WKSTS[2]), DATWKF (UART_WKSTS[1]) or CTSWKF(UART_WKSTS[0]) is set to 1. 0 = No UART wake-up interrupt flag is generated. 1 = UART wake-up interrupt flag is generated. **Note:** This bit is cleared if all of TOUTWKF, RS485WKF, RFRTWKF, DATWKF and CTSWKF are cleared to 0 by writing 1 to the corresponding interrupt flag.
[5]	**BUFERRIF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX FIFO or RX FIFO overflows (TXOVIF (UART_FIFOSTS[24]) or RXOVIF (UART_FIFOSTS[0]) is set). When BUFERRIF (UART_INTSTS[5]) is set, the transfer is not correct. If BUFERRIEN (UART_INTEN[5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated. 1 = Buffer error interrupt flag is generated. **Note:** This bit is cleared if both of RXOVIF(UART_FIFOSTS[0]) and TXOVIF(UART_FIFOSTS[24]) are cleared to 0 by writing 1 to RXOVIF(UART_FIFOSTS[0]) and TXOVIF(UART_FIFOSTS[24]).
[4]	**RXTOIF**	**RX Time-out Interrupt Flag (Read Only)** This bit is set when the RX FIFO is not empty and no activities occurred in the RX FIFO and the time-out counter equal to TOIC (UART_TOUT[7:0]). If RXTOIEN (UART_INTEN[4]) is enabled, the RX time-out interrupt will be generated. 0 = No RX time-out interrupt flag is generated. 1 = RX time-out interrupt flag is generated. **Note:** This bit is read only and user can read UART_DAT (RX is in active) to clear it.
[3]	**MODEMIF**	**MODEM Interrupt Flag (Read Only)** This bit is set when the nCTS pin has state change (CTSDETF (UART_MODEMSTS[0]) = 1). If MODEMIEN (UART_INTEN[3]) is enabled, the Modem interrupt will be generated. 0 = No Modem interrupt flag is generated. 1 = Modem interrupt flag is generated. **Note:** This bit is read only and reset to 0 when bit CTSDETF is cleared by a write 1 on CTSDETF(UART_MODEMSTS[0]).
[2]	**RLSIF**	**Receive Line Interrupt Flag (Read Only)** This bit is set when the RX receive data have parity error, frame error or break error (at least one of 3 bits, BIF(UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]) and

		PEF(UART_FIFOSTS[4]), is set). If RLSIEN (UART_INTEN[2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated. 1 = RLS interrupt flag is generated. **Note 1:** In RS-485 function mode, this field is set include “receiver detect and received address byte character (bit9 = ‘1’) bit". At the same time, the bit of ADDRDETF (UART_FIFOSTS[3]) is also set. **Note 2:** This bit is read only and reset to 0 when all bits of BIF (UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]) and PEF(UART_FIFOSTS[4]) are cleared. **Note 3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF (UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]), PEF(UART_FIFOSTS[4]) and ADDRDETF (UART_FIFOSTS[3]) are cleared.
[1]	**THREIF**	**Transmit Holding Register Empty Interrupt Flag (Read Only)** This bit is set when the last data of TX FIFO is transferred to Transmitter Shift Register. If THREIEN (UART_INTEN[1]) is enabled, the THRE interrupt will be generated. 0 = No THRE interrupt flag is generated. 1 = THRE interrupt flag is generated. **Note:** This bit is read only and will be cleared when writing data into UART_DAT (TX FIFO is not empty).
[0]	**RDAIF**	**Receive Data Available Interrupt Flag (Read Only)** When the number of bytes in the RX FIFO equals the RFITL then the RDAIF(UART_INTSTS[0]) will be set. If RDAIEN (UART_INTEN[0]) is enabled, the RDA interrupt will be generated. 0 = No RDA interrupt flag is generated. 1 = RDA interrupt flag is generated. **Note:** This bit is read only and will be cleared when the number of unread bytes of RX FIFO drops below the threshold level (RFITL(UART_FIFO[7:4]).

UART Time-out Register (UART_TOUT)

Register	Offset	R/W	Description	Reset Value
UARTx_TOUT **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000

31	30	29	28	27	26	25	24
BITOMEN	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DLY							
7	6	5	4	3	2	1	0
TOIC							

Bits	Description	
[31]	**BITOMEN**	**Bus Idle Time-out Mode Enable Bit** If BITOMEN (UART_TOUT[31]) is enabled, the reset conditions of the time-out counter and RXTOIF (UART_INTSTS[4]) will be changed to detect the bus idle. When BITOMEN (UART_TOUT[31]) is disabled, the time-out counter and RXTOIF (UART_INTSTS[4]) maintain reset value whenever the RX FIFO is empty. In addition, a new incoming data word will also clear RXTOIF (UART_INTSTS[4]). On the other hand, when BITOMEN (UART_TOUT[31]) is enabled, the RX FIFO empty state will not reset the time-out counter and RXTOIF (UART_INTSTS[4]), and the new incoming data word event will not clear RXTOIF (UART_INTSTS[4]). 0 = Bus idle time-out mode Disabled. 1 = Bus idle time-out mode Enabled.
[30:16]	**Reserved**	Reserved.
[15:8]	**DLY**	**TX Delay Time Value** This field is used to program the transfer delay time between the last STOP bit and next START bit. The unit is bit time.
[7:0]	**TOIC**	**Time-out Interrupt Comparator** The time-out counter resets and starts counting (the counting clock = baud rate) whenever the RX FIFO receives a new data word if time out counter is enabled by setting TOCNTEN (UART_INTEN[11]). Once the content of time-out counter is equal to that of time-out interrupt comparator (TOIC (UART_TOUT[7:0])), a receiver time-out interrupt (RXTOINT(UART_INTSTS[12])) is generated if RXTOIEN (UART_INTEN[4]) enabled. A new incoming data word or RX FIFO empty will clear RXTOIF (UART_INTSTS[4]). In order to avoid receiver time-out interrupt generation immediately during one character is being received, TOIC value should be set between 40 and 255. So, for example, if TOIC is set with 40, the time-out interrupt is generated after four characters are not received when 1 STOP bit and no parity check is set for UART transfer.

UART Baud Rate Divider Register (UART_BAUD)

Register	Offset	R/W	Description	Reset Value
UARTx_BAUD **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000

31	30	29	28	27	26	25	24
Reserved	BRFDEN	BAUDM1	BAUDM0	EDIVM1			
23	22	21	20	19	18	17	16
BRFD							
15	14	13	12	11	10	9	8
BRD							
7	6	5	4	3	2	1	0
BRD							

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**BRFDEN**	**Baud Rate Fractional Divider Enable Bit** 0 = Baud Rate Fractional Divider Disabled. 1 = Baud Rate Fractional Divider Enabled. **Note:** This bit has effect only at baud rate mode 2 (BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) must be 1).
[29]	**BAUDM1**	**BAUD Rate Mode Selection Bit 1** This bit is baud rate mode selection bit 1. UART provides three baud rate calculation modes. This bit combines with BAUDM0 (UART_BAUD[28]) to select baud rate calculation mode. The detail description is shown in Table 6.28-4. **Note:** In IrDA mode must be operated in mode 0.
[28]	**BAUDM0**	**BAUD Rate Mode Selection Bit 0** This bit is baud rate mode selection bit 0. UART provides three baud rate calculation modes. This bit combines with BAUDM1 (UART_BAUD[29]) to select baud rate calculation mode. The detail description is shown in Table 6.28-4.
[27:24]	**EDIVM1**	**Extra Divider for BAUD Rate Mode 1** This field is used for baud rate calculation in mode 1 and has no effect for baud rate calculation in mode 0 and mode 2. The detail description is shown in Table 6.28-4.
[23:16]	**BRFD**	**Baud Rate Fractional Divider** This field is the fractional part of the baud rate divisor. When BRFDEN (UART_BAUD[30]) is set, the Baud Rate Equation goes to UART_CLK / ((BRD+2) + (BRFD/256)). For instance, if BRD = 0x8 and BRFD = 0x40, the baud rate would be UART_CLK / 10.25. In practical, this represent 1 bit = 10 clocks, and for every 4 bits, the bit length becomes 11 clocks. As another example, if BRD = 0x8 and BRFD = 0x20, the baud rate would be UART_CLK / 10.125, which means 1 bit = 10 clocks, and for every 8 bits, the bit length becomes 11 clocks.
[15:0]	**BRD**	**Baud Rate Divider**

		The field indicates the baud rate divider. This filed is used in baud rate calculation. The detail description is shown in Table 6.28-4.

UART IrDA Control Register (UART_IRDA)

Register	Offset	R/W	Description	Reset Value
UARTx_IRDA **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**RXINV**	**TXINV**	**Reserved**			**TXEN**	**Reserved**

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**RXINV**	**IrDA Inverse Receive Input Signal** 0 = None inverse receiving input signal. 1 = Inverse receiving input signal. (Default) **Note 1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[2:0]) is select IrDA function.
[5]	**TXINV**	**IrDA Inverse Transmitting Output Signal** 0 = None inverse transmitting signal. (Default). 1 = Inverse transmitting output signal. **Note 1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[2:0]) is select IrDA function.
[4:2]	**Reserved**	Reserved.
[1]	**TXEN**	**IrDA Receiver/Transmitter Selection Enable Bit** 0 = IrDA Transmitter Disabled and Receiver Enabled. (Default) 1 = IrDA Transmitter Enabled and Receiver Disabled.
[0]	**Reserved**	Reserved.

UART Alternate Control/Status Register (UART_ALTCTL)

Register	Offset	R/W	Description	Reset Value
UARTx_ALTCTL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C

31	30	29	28	27	26	25	24
ADDRMV							
23	22	21	20	19	18	17	16
Reserved			ABRDBITS		ABRDEN	ABRIF	Reserved
15	14	13	12	11	10	9	8
ADDRDEN	Reserved				RS485AUD	RS485AAD	RS485NMM
7	6	5	4	3	2	1	0
LINTXEN	LINRXEN	Reserved		BRKFL			

Bits	Description	
[31:24]	**ADDRMV**	**Address Match Value** This field contains the RS-485 address match values. **Note:** This field is used for RS-485 auto address detection mode.
[23:21]	**Reserved**	Reserved.
[20:19]	**ABRDBITS**	**Auto-baud Rate Detect Bit Length** 00 = 1-bit time from START bit to the 1st rising edge. The input pattern shall be 0x01. 01 = 2-bit time from START bit to the 1st rising edge. The input pattern shall be 0x02. 10 = 4-bit time from START bit to the 1st rising edge. The input pattern shall be 0x08. 11 = 8-bit time from START bit to the 1st rising edge. The input pattern shall be 0x80. **Note :** The calculation of bit number includes the START bit.
[18]	**ABRDEN**	**Auto-baud Rate Detect Enable Bit** 0 = Auto-baud rate detect function Disabled. 1 = Auto-baud rate detect function Enabled. **Note :** This bit is cleared automatically after auto-baud detection is finished.
[17]	**ABRIF**	**Auto-baud Rate Interrupt Flag (Read Only)** This bit is set when auto-baud rate detection function finished or the auto-baud rate counter was overflow and if ABRIEN(UART_INTEN[18]) is set then the auto-baud rate interrupt will be generated. 0 = No auto-baud rate interrupt flag is generated. 1 = Auto-baud rate interrupt flag is generated. **Note:** This bit is read only, but it can be cleared by writing “1” to ABRDTOIF (UART_FIFOSTS[2]) and ABRDIF(UART_FIFOSTS[1]).
[16]	**Reserved**	Reserved.
[15]	**ADDRDEN**	**RS-485 Address Detection Enable Bit** This bit is used to enable RS-485 Address Detection mode. 0 = Address detection mode Disabled. 1 = Address detection mode Enabled.

		Note: This bit is used for RS-485 any operation mode.
[14:11]	**Reserved**	Reserved.
[10]	**RS485AUD**	**RS-485 Auto Direction Function** 0 = RS-485 Auto Direction Operation function (AUD) Disabled. 1 = RS-485 Auto Direction Operation function (AUD) Enabled. **Note:** It can be active with RS-485_AAD or RS-485_NMM operation mode.
[9]	**RS485AAD**	**RS-485 Auto Address Detection Operation Mode** 0 = RS-485 Auto Address Detection Operation mode (AAD) Disabled. 1 = RS-485 Auto Address Detection Operation mode (AAD) Enabled. **Note:** It cannot be active with RS-485_NMM operation mode.
[8]	**RS485NMM**	**RS-485 Normal Multi-drop Operation Mode** 0 = RS-485 Normal Multi-drop Operation mode (NMM) Disabled. 1 = RS-485 Normal Multi-drop Operation mode (NMM) Enabled. **Note:** It cannot be active with RS-485_AAD operation mode.
[7]	**LINTXEN**	**LIN TX Break Mode Enable Bit** 0 = LIN TX Break mode Disabled. 1 = LIN TX Break mode Enabled. **Note:** When TX break field transfer operation finished, this bit will be cleared automatically.
[6]	**LINRXEN**	**LIN RX Enable Bit** 0 = LIN RX mode Disabled. 1 = LIN RX mode Enabled.
[5:4]	**Reserved**	Reserved.
[3:0]	**BRKFL**	**UART LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note 1:** This break field length is BRKFL + 1. **Note 2:** According to LIN spec, the reset value is 0xC (break field length = 13).

UART Function Select Register (UART_FUNCSEL)

Register	Offset	R/W	Description	Reset Value
UARTx_FUNCSEL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXRXSWP	**DGE**	**Reserved**		**TXRXDIS**	**FUNCSEL**		

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**TXRXSWP**	**TX and RX Swap Enable Bit** Setting this bit Swaps TX pin and RX pin. 0 = TX and RX Swap Disabled. 1 = TX and RX Swap Enabled.
[6]	**DGE**	**Deglitch Enable Bit** 0 = Deglitch Disabled. 1 = Deglitch Enabled. **Note 1:** When this bit is set to logic 1, any pulse width less than about 150 ns will be considered a glitch and will be removed in the serial data input (RX). This bit acts only on RX line and has no effect on the transmitter logic. **Note 2:** It is recommended to set this bit only when operating at baud rate under 2.5 Mbps.
[5:4]	**Reserved**	Reserved.
[3]	**TXRXDIS**	**TX and RX Disable Bit** Setting this bit can disable TX and RX. 0 = TX and RX Enabled. 1 = TX and RX Disabled. **Note:** The TX and RX will not be disabled immediately when this bit is set. The TX and RX complete current task before TX and RX are disabled. When TX and RX are disabled, the TXRXACT (UART_FIFOSTS[31]) is cleared.
[2:0]	**FUNCSEL**	**Function Select** 000 = UART function. 001 = LIN function. 010 = IrDA function. 011 = RS-485 function. 100 = UART Single-wire function. Others = Reserved.

UART LIN Control Register (UART_LINCTL)

Register	Offset	R/W	Description	Reset Value
UARTx_LINCTL **x=0,1**	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000

31	30	29	28	27	26	25	24
PID							
23	22	21	20	19	18	17	16
HSEL		BSL		BRKFL			
15	14	13	12	11	10	9	8
Reserved			BITERREN	LINRXOFF	BRKDETEN	IDPEN	SENDH
7	6	5	4	3	2	1	0
Reserved			MUTE	SLVDUEN	SLVAREN	SLVHDEN	SLVEN

Bits	Description	
[31:24]	**PID**	**LIN PID Bits** This field contains the LIN frame ID value in LIN function mode, and the frame ID parity can be generated by software or hardware depends on IDPEN (UART_LINCTL[9]) = 1. If the parity generated by hardware, user fill ID0~ID5 (PID[29:24]), hardware will calculate P0 (PID[30]) and P1 (PID[31]), otherwise user must filled frame ID and parity in this field. **Note 1:** User can fill any 8-bit value to this field and the bit 24 indicates ID0 (LSB first). **Note 2:** This field can be used for LIN master mode or slave mode.
[23:22]	**HSEL**	**LIN Header Select** 00 = The LIN header includes “break field”. 01 = The LIN header includes “break field” and “sync field”. 10 = The LIN header includes “break field”, “sync field” and “frame ID field”. 11 = Reserved. **Note:** This bit is used to master mode for LIN to send header field (SENDH (UART_LINCTL[8]) = 1) or used to slave to indicate exit from mute mode condition (MUTE (UART_LINCTL[4] = 1).
[21:20]	**BSL**	**LIN Break/Sync Delimiter Length** 00 = The LIN break/sync delimiter length is 1-bit time. 01 = The LIN break/sync delimiter length is 2-bit time. 10 = The LIN break/sync delimiter length is 3-bit time. 11 = The LIN break/sync delimiter length is 4-bit time. **Note:** This bit used for LIN master to sending header field.
[19:16]	**BRKFL**	**LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note 1:** These registers are shadow registers of BRKFL (UART_ALTCTL[3:0]). User can read/write it by setting BRKFL (UART_ALTCTL[3:0]) or BRKFL (UART_LINCTL[19:16]). **Note 2:** This break field length is BRKFL + 1. **Note 3:** According to LIN spec, the reset value is 12 (break field length = 13).
[15:13]	**Reserved**	Reserved.

[12]	**BITERREN**	**Bit Error Detect Enable Bit** 0 = Bit error detection function Disabled. 1 = Bit error detection function Enabled. **Note:** In LIN function mode, when occur bit error, the BITEF (UART_LINSTS[9]) flag will be asserted. If the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated.
[11]	**LINRXOFF**	**LIN Receiver Disable Bit** If the receiver is enabled (LINRXOFF (UART_LINCTL[11]) = 0), all received byte data will be accepted and stored in the RX FIFO, and if the receiver is disabled (LINRXOFF (UART_LINCTL[11] = 1), all received byte data will be ignore. 0 = LIN receiver Enabled. 1 = LIN receiver Disabled. **Note:** This bit is only valid when operating in LIN function mode (FUNCSEL (UART_FUNCSEL[2:0]) = 001).
[10]	**BRKDETEN**	**LIN Break Detection Enable Bit** When detect consecutive dominant greater than 11 bits, and are followed by a delimiter character, the BRKDETF (UART_LINSTS[8]) flag is set at the end of break field. If the LINIEN (UART_INTEN[8])=1, an interrupt will be generated. 0 = LIN break detection Disabled. 1 = LIN break detection Enabled.
[9]	**IDPEN**	**LIN ID Parity Enable Bit** 0 = LIN frame ID parity Disabled. 1 = LIN frame ID parity Enabled. **Note 1:** This bit can be used for LIN master to sending header field (SENDH (UART_LINCTL[8])) = 1 and HSEL (UART_LINCTL[23:22]) = 10 or be used for enable LIN slave received frame ID parity checked. **Note 2:** This bit is only used when the operation header transmitter is in HSEL (UART_LINCTL[23:22]) = 10.
[8]	**SENDH**	**LIN TX Send Header Enable Bit** The LIN TX header can be “break field” or “break and sync field” or “break, sync and frame ID field”, it is depend on setting HSEL (UART_LINCTL[23:22]). 0 = Send LIN TX header Disabled. 1 = Send LIN TX header Enabled. **Note 1:** This bit is shadow bit of LINTXEN (UART_ALTCTL[7]); user can read/write it by setting LINTXEN (UART_ALTCTL[7]) or SENDH (UART_LINCTL[8]). **Note 2:** When transmitter header field (it may be “break” or “break + sync” or “break + sync + frame ID” selected by HSEL (UART_LINCTL[23:22]) field) transfer operation finished, this bit will be cleared automatically.
[7:5]	**Reserved**	Reserved.
[4]	**MUTE**	**LIN Mute Mode Enable Bit** 0 = LIN mute mode Disabled. 1 = LIN mute mode Enabled. **Note:** The exit from mute mode condition and each control and interactions of this field are explained in 6.27.5.10 (LIN slave mode).
[3]	**SLVDUEN**	**LIN Slave Divider Update Method Enable Bit** 0 = UART_BAUD updated is written by software (if no automatic resynchronization update occurs at the same time). 1 = UART_BAUD is updated at the next received character. User must set the bit before checksum reception. **Note 1:** This bit only is valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note 2:** This bit used for LIN Slave Automatic Resynchronization mode. (for Non-Automatic Resynchronization mode, this bit should be kept cleared)

		Note 3: The control and interactions of this field are explained in 6.27.5.10(Slave mode with automatic resynchronization).
[2]	**SLVAREN**	**LIN Slave Automatic Resynchronization Mode Enable Bit** 0 = LIN automatic resynchronization Disabled. 1 = LIN automatic resynchronization Enabled. **Note 1:** This bit only is valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note 2:** When operation in Automatic Resynchronization mode, the baud rate setting must be mode 2 (BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) must be 1). **Note 3:** The control and interactions of this field are explained in 6.27.5.10((Slave mode with automatic resynchronization).
[1]	**SLVHDEN**	**LIN Slave Header Detection Enable Bit** 0 = LIN slave header detection Disabled. 1 = LIN slave header detection Enabled. **Note 1:** This bit is only valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note 2:** In LIN function mode, when detect header field (break + sync + frame ID), SLVHDETF (UART_LINSTS[0]) flag will be asserted. If the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated.
[0]	**SLVEN**	**LIN Slave Mode Enable Bit** 0 = LIN slave mode Disabled. 1 = LIN slave mode Enabled.

UART LIN Status Register (UART_LINSTS)

Register	Offset	R/W	Description	Reset Value
UARTx_LINSTS **x=0,1**	UARTx_BA+0x38	R/W	UART LIN Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						BITEF	BRKDETF
7	6	5	4	3	2	1	0
Reserved				SLVSYNCF	SLVIDPEF	SLVHEF	SLVHDETF

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**BITEF**	**Bit Error Detect Status Flag** At TX transfer state, hardware will monitor the bus state, if the input pin (UART_RXD) state not equals to the output pin (UART_TXD) state, BITEF (UART_LINSTS[9]) will be set. When occur bit error, if the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated. 0 = Bit error not detected. 1 = Bit error detected. **Note 1:** This bit can be cleared by writing 1 to it. **Note 2:** This bit is only valid when enable bit error detection function (BITERREN (UART_LINCTL[12]) = 1).
[8]	**BRKDETF**	**LIN Break Detection Flag** This bit is set by hardware when a break is detected and be cleared by writing 1 to it through software. 0 = LIN break not detected. 1 = LIN break detected. **Note 1:** This bit can be cleared by writing 1 to it. **Note 2:** This bit is only valid when LIN break detection function is enabled (BRKDETEN (UART_LINCTL[10]) =1).
[7:4]	**Reserved**	Reserved.
[3]	**SLVSYNCF**	**LIN Slave Sync Field** This bit indicates that the LIN sync field is being analyzed in Automatic Resynchronization mode. When the receiver header have some error been detect, user must reset the internal circuit to re-search new frame header by writing 1 to this bit. 0 = The current character is not at LIN sync state. 1 = The current character is at LIN sync state. **Note 1:** This bit is only valid in LIN Slave mode (SLVEN(UART_LINCTL[0]) = 1). **Note 2:** This bit can be cleared by writing 1 to it. **Note 3:** When writing 1 to it, hardware will reload the initial baud rate and re-search a new

		frame header.
[2]	**SLVIDPEF**	**LIN Slave ID Parity Error Flag** This bit is set by hardware when receipted frame ID parity is not correct. 0 = No active. 1 = Receipted frame ID parity is not correct. **Note 1:** This bit can be cleared by writing 1 to it. **Note 2:** This bit is only valid in LIN slave mode (SLVEN (UART_LINCTL[0])= 1) and enable LIN frame ID parity check function IDPEN (UART_LINCTL[9]).
[1]	**SLVHEF**	**LIN Slave Header Error Flag** This bit is set by hardware when a LIN header error is detected in LIN slave mode and be cleared by writing 1 to it. The header errors include “break delimiter is too short (less than 0.5 bit time)”, “frame error in sync field or Identifier field”, “sync field data is not 0x55 in Non-Automatic Resynchronization mode”, “sync field deviation error with Automatic Resynchronization mode”, “sync field measure time-out with Automatic Resynchronization mode” and “LIN header reception time-out”. 0 = LIN header error not detected. 1 = LIN header error detected. **Note 1:** This bit can be cleared by writing 1 to it. **Note 2:** This bit is only valid when UART is operated in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1) and enables LIN slave header detection function (SLVHDEN (UART_LINCTL[1])).
[0]	**SLVHDETF**	**LIN Slave Header Detection Flag** This bit is set by hardware when a LIN header is detected in LIN slave mode and be cleared by writing 1 to it. 0 = LIN header not detected. 1 = LIN header detected (break + sync + frame ID). **Note 1:** This bit can be cleared by writing 1 to it. **Note 2:** This bit is only valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1) and enable LIN slave header detection function (SLVHDEN (UART_LINCTL[1])). **Note 3:** When enable ID parity check IDPEN (UART_LINCTL[9]), if hardware detect complete header (“break + sync + frame ID”), the SLVHDETF will be set whether the frame ID correct or not.

UART Baud Rate Compensation Register (UART_BRCOMP)

Register	Offset	R/W	Description	Reset Value
UARTx_BRCOMP **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000

31	30	29	28	27	26	25	24
BRCOMPDEC	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BRCOMP
7	6	5	4	3	2	1	0
BRCOMP							

Bits	Description	
[31]	**BRCOMPDEC**	**Baud Rate Compensation Decrease** 0 = Positive (increase one module clock) compensation for each compensated bit. 1 = Negative (decrease one module clock) compensation for each compensated bit.
[30:9]	**Reserved**	Reserved.
[8:0]	**BRCOMP**	**Baud Rate Compensation Patten** These 9-bits are used to define the relative bit is compensated or not. BRCOMP[7:0] is used to define the compensation of DAT (UART_DAT[7:0]) and BRCOM[8] is used to define PARITY (UART_DAT[8]).

UART Wake-up Control Register (UART_WKCTL)

Register	Offset	R/W	Description	Reset Value
UARTx_WKCTL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x40	R/W	UART Wake-up Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**WKTOUTEN**	**WKRS485EN**	**WKRFRTEN**	**WKDATEN**	**WKCTSEN**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**WKTOUTEN**	**Received Data FIFO Reached Threshold Time-out Wake-up Enable Bit** 0 = Received Data FIFO reached threshold time-out wake-up system function Disabled. 1 = Received Data FIFO reached threshold time-out wake-up system function Enabled. **Note 1:** When the system is in Power-down mode, Received Data FIFO reached threshold time-out will wake up system from Power-down mode. **Note 2:** It is suggested the function is enabled when the WKRFRTEN (UART_WKCTL[2]) is set to 1.
[3]	**WKRS485EN**	**RS-485 Address Match Wake-up Enable Bit** 0 = RS-485 Address Match (AAD mode) wake-up system function Disabled. 1 = RS-485 Address Match (AAD mode) wake-up system function Enabled. **Note 1:** When the system is in Power-down mode, RS-485 Address Match will wake-up system from Power-down mode. **Note 2:** This bit is used for RS-485 Auto Address Detection (AAD) mode in RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1.
[2]	**WKRFRTEN**	**Received Data FIFO Reached Threshold Wake-up Enable Bit** 0 = Received Data FIFO reached threshold wake-up system function Disabled. 1 = Received Data FIFO reached threshold wake-up system function Enabled. **Note:** When the system is in Power-down mode, Received Data FIFO reached threshold will wake-up system from Power-down mode.
[1]	**WKDATEN**	**Incoming Data Wake-up Enable Bit** 0 = Incoming data wake-up system function Disabled. 1 = Incoming data wake-up system function Enabled. **Note:** When the system is in Power-down mode, incoming data will wake-up system from Power-down mode.
[0]	**WKCTSEN**	**nCTS Wake-up Enable Bit** 0 = nCTS Wake-up system function Disabled. 1 = nCTS Wake-up system function Enabled.

		Note: When the system is in Power-down mode, an external nCTS change will wake up system from Power-down mode.

UART Wake-up Status Register (UART_WKSTS)

Register	Offset	R/W	Description	Reset Value
UARTx_WKSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x44	R/W	UART Wake-up Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**TOUTWKF**	**RS485WKF**	**RFRTWKF**	**DATWKF**	**CTSWKF**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**TOUTWKF**	**Received Data FIFO Threshold Time-out Wake-up Flag** This bit is set if chip wake-up from power-down state by Received Data FIFO Threshold Time-out wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Received Data FIFO reached threshold time-out. **Note 1:** If WKTOUTEN (UART_WKCTL[4]) is enabled, the Received Data FIFO reached threshold time-out wake-up cause this bit is set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.
[3]	**RS485WKF**	**RS-485 Address Match Wake-up Flag** This bit is set if chip wake-up from power-down state by RS-485 Address Match (AAD mode). 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by RS-485 Address Match (AAD mode) wake-up. **Note 1:** If WKRS485EN (UART_WKCTL[3]) is enabled, the RS-485 Address Match (AAD mode) wake-up cause this bit is set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.
[2]	**RFRTWKF**	**Received Data FIFO Reached Threshold Wake-up Flag** This bit is set if chip wake-up from power-down state by Received Data FIFO reached threshold wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Received Data FIFO Reached Threshold wake-up. **Note 1:** If WKRFRTEN (UART_WKCTL[2]) is enabled, the Received Data FIFO Reached Threshold wake-up cause this bit is set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.

[1]	**DATWKF**	**Incoming Data Wake-up Flag** This bit is set if chip wake-up from power-down state by data wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Incoming Data wake-up. **Note 1:** If WKDATEN (UART_WKCTL[1]) is enabled, the Incoming Data wake-up cause this bit is set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.
[0]	**CTSWKF**	**nCTS Wake-up Flag** This bit is set if chip wake-up from power-down state by nCTS wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by nCTS wake-up. **Note 1:** If WKCTSEN (UART_WKCTL[0]) is enabled, the nCTS wake-up cause this bit is set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.

UART Incoming Data Wake-up Compensation Register (UART_DWKCOMP)

Register	Offset	R/W	Description	Reset Value
UARTx_DWKCOMP **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x48	R/W	UART Incoming Data Wake-up Compensation Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
STCOMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STCOMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**STCOMP**	**START Bit Compensation Value** These bits field indicate how many clock cycle selected by UART_CLK do the UART controller can get the 1st bit (START bit) when the device is wake-up from Power-down mode. **Note:** It is valid only when WKDATEN (UART_WKCTL[1]) is set.

UART RS485 Transceiver Deactivate Delay Register (UART_RS485DD)

Register	Offset	R/W	Description	Reset Value
UARTx_RS485DD **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x4C	R/W	UART RS485 Transceiver Deactivate Delay Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RTSDDLY							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RTSDDLY							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RTSDDLY**	**RS485 Transceiver Deactivate Delay Value** These bits field indicate how many clock cycles selected by UART_CLK do the UART controller delay the RS485 transceiver state trancing when the state trancing of RS485 transceiver is from TX to RX state. These bits field have no effect when the state trancing of RS485 transceiver is from RX to TX state. **Note:** It is valid only when RS485AUD (UART_ALTCTL[10]) is set.

6.28 Low Power UART Interface Controller (LPUART)

6.28.1 Overview

The chip provides one channel of Low Power Universal Asynchronous Receiver/Transmitters (LPUART). The LPUART controller performs serial-to-parallel conversion on data received from the peripheral and parallel-to-serial conversion on data transmitted from the CPU. Each LPUART controller channel supports ten types of interrupts. The LPUART controller supports flow control function. The LPUART controller also supports RS-485 and Single-wire function modes and auto-baud rate measuring function.

6.28.2 Features

- Full-duplex asynchronous communications
- Separates receive and transmit 16/16 bytes entry FIFO for data payloads
- Supports hardware auto-flow control
- Programmable receiver buffer trigger level
- Supports programmable baud rate generator for each channel individually
- Supports nCTS, incoming data, Received Data FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function
- Supports 8-bit receiver buffer time-out detection function
- Programmable transmitting data delay time between the last stop and the next START bit by setting DLY (LPUART_TOUT[15:8])
- Supports Auto-Baud Rate measurement and baud rate compensation function
  - 9600 bps for LPUART_CLK is selected LXT.
- Supports break error, frame error, parity error and receive/transmit buffer overflow detection function
- Fully programmable serial-interface characteristics
  - Programmable number of data bit, 5-, 6-, 7-, 8- bit character
  - Programmable PARITY bit, even, odd, no parity or stick PARITY bit generation and detection
  - Programmable STOP bit, 1, 1.5, or 2 STOP bit generation
- Supports RS-485 function mode
  - RS-485 9-bit mode
  - Hardware or software enabled to program nRTS pin to control RS-485 transmission direction
- Supports LPPDMA transfer function
- Supports Single-wire function mode.
- Supports Automatic Operation

LPUART Feature	LPUART0	UART0/ UART1	UART2 ~ UART7	USCI-UART
FIFO	16 Bytes	16 Bytes	16 Bytes	TX: 1byte RX: 2byte
Automatic Operation	√	-	-	-
Auto Flow Control	√	√	√	√

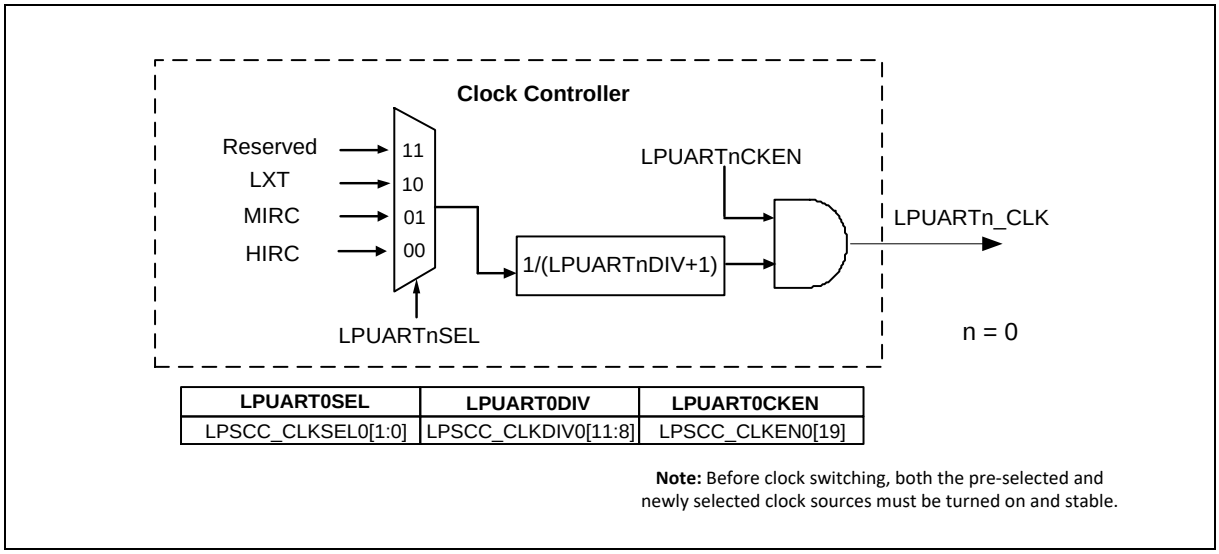
(CTS/RTS)				
RS-485 Function Mode	√	√	√	√
nCTS Wake-up	√	√	√	√
Incoming Data Wake-up	√	√	√	√
Received Data FIFO reached threshold Wake-up	√	√	√	-
RS-485 Address Match (AAD mode) Wake-up	√	√	√	-
Received Data FIFO reached threshold Time-out Wake-up	√	√	√	-
Baud Rate Compensation	√	√	√	-
Auto-Baud Rate Measurement	√	√	√	√
STOP bit Length	1, 1.5, 2 bit	1, 1.5, 2 bit	1, 1.5, 2 bit	1, 2 bit
Word Length	5, 6, 7, 8 bits	5, 6, 7, 8 bits	5, 6, 7, 8 bits	6~13 bits
Even / Odd Parity	√	√	√	√
Stick Bit	√	√	√	-

Table 6.28-1 M2L31 Series LPUART Features

6.28.3 Block Diagram

The LPUART clock control and block diagram are shown in Figure 6.28-1 and Figure 6.28-2 respectively.

Note: The frequency of LPUARTx_CLK should not be greater than 30 times HCLK.



LPUART0SEL	LPUART0DIV	LPUART0CKEN
LPSCC_CLKSEL0[1:0]	LPSCC_CLKDIV0[11:8]	LPSCC_CLKEN0[19]

Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 6.28-1 LPUART Clock Control Diagram

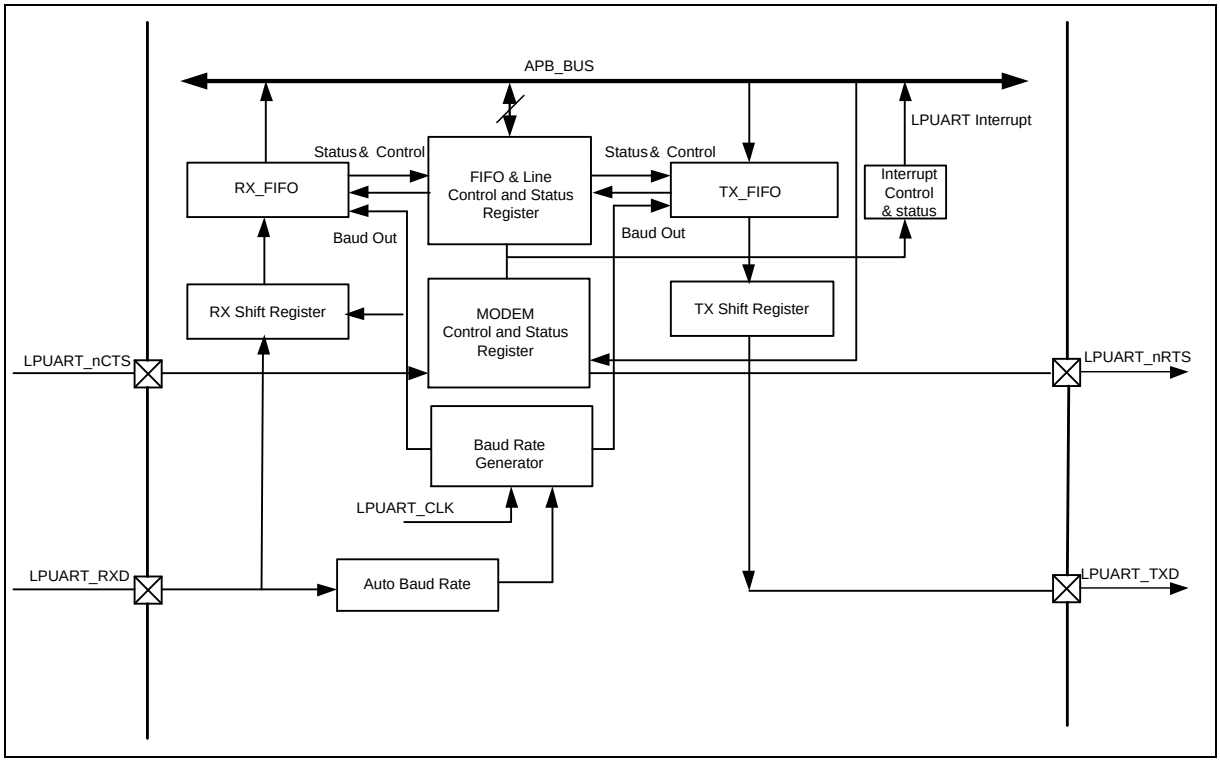


Figure 6.28-2 LPUART Block Diagram

Each block is described in detail as follows:

TX_FIFO

The transmitter is buffered with a 16 bytes FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO

The receiver is buffered with a 16 bytes FIFO (plus three error bits, BIF (LPUART_FIFOSTS[6]), FEF (LPUART_FIFOSTS[5]), PEF (LPUART_FIFOSTS[4])) to reduce the number of interrupts presented to the CPU.

TX Shift Register

This block is responsible for shifting out the transmitting data serially.

RX Shift Register

This block is responsible for shifting in the receiving data serially.

Modem Control and Status Register

This register controls the interface to the MODEM or data set (or a peripheral device emulating a MODEM).

Baud Rate Generator

Divide the external clock by the divisor to get the desired baud rate clock. Refer to baud rate equation.

FIFO & Line Control and Status Register

This field is register set that including the FIFO control register (LPUART_FIFO), FIFO status register (LPUART_FIFOSTS), and line control register (LPUART_LINE) for transmitter and receiver. The time-out register (LPUART_TOUT) identifies the condition of time-out interrupt.

Auto-Baud Rate Measurement

This block is responsible for auto-baud rate measurement.

Interrupt Control and Status Register

There are ten types of interrupts. Interrupt enable register (LPUART_INTEN) enable or disable the responding interrupt and interrupt status register (LPUART_INTSTS) identifying the occurrence of the responding interrupt.

Interrupt	Description
RDAINT	Receive Data Available Interrupt.
THREINT	Transmit Holding Register Empty Interrupt.
TXENDINT	Transmitter Empty Interrupt.
RLSINT	Receive Line Status Interrupt (parity error or frame error or break error).
MODEMINT	MODEM Status Interrupt.
RXTOINT	Receiver Buffer Time-out Interrupt.
BUFERRINT	Buffer Error Interrupt.
WKINT	Wake-up Interrupt.
ABRINT	Auto-Baud Rate Interrupt.
SWBEINT	Single-wire Bit Error Detect Interrupt.

Table 6.28-2 LPUART Interrupt

6.28.4 Basic Configuration

6.28.4.1 LPUART0 Basic Configuration

- Clock Source Configuration

- Select the source of LPUART0 peripheral clock on LPUART0SEL (LPSCC_CLKSEL0[1:0]).
- Select the clock divider number of LPUART0 peripheral clock on LPUART0DIV (LPSCC_CLKDIV0[11:8]).
- Enable LPUART0 peripheral clock in LPUART0CKEN (LPSCC_CLKEN0[19]).

- Reset LPUART0 controller in LPUART0RST (LPSCC_IPRST0[19]).
-

LPUART Interface Controller Pin description is shown in Table 6.28-3:

Pin	Type	Description
LPUARTx_TXD	Output	LPUARTx transmit
LPUARTx_RXD	Input	LPUARTx receive
LPUARTx_nCTS	Input	LPUARTx modem clear to send
LPUARTx_nRTS	Output	LPUARTx modem request to send

Table 6.28-3 LPUART Interface Controller Pin

6.28.5 Functional Description

The LPUART controller supports four function modes including LPUART, RS-485 and Single-wire mode. User can select a function by setting the LPUART_FUNCSEL register. The four function modes will be described in following section.

6.28.5.1 LPUART Controller Baud Rate Generator

The LPUART controller includes a programmable baud rate generator capable of dividing clock input by divisors to produce the serial clock that transmitter and receiver need. Table 6.28-4 list the LPUART baud rate equations in the various conditions. Table 6.28-5 and Table 6.28-6 list the LPUART baud rate parameter and register setting example. More detailed register description is shown in LPUART_BAUD register. There are three setting modes. Mode 0 is set by LPUART_BAUD[29:28] with 00. Mode 1 is set by LPUART_BAUD[29:28] with 10. Mode 2 is set by LPUART_BAUD[29:28] with 11.

Mode	BAUDM1	BAUDM0	Baud Rate Equation
Mode 0	0	0	LPUART_CLK / [16 * (BRD+2)].
Mode 1	1	0	LPUART_CLK / [(EDIVM1+1) * (BRD+2)], EDIVM1 must >= 8.
Mode 2	1	1	LPUART_CLK / (BRD+2) If LPUART_CLK <= 3*HCLK, BRD must >= 8. If LPUART_CLK > 3*HCLK, BRD must >= 3*N – 1. N is the smallest integer larger than or equal to the ratio of LPUART_CLK /HCLK. For example, if 2*HCLK < LPUART_CLK =< 3*HCLK, BRD must >=11. if 4*HCLK < LPUART_CLK =< 5*HCLK, BRD must >=14. (If the LPUART_CLK is selected from LXT, BRD can be greater than or equal to 1)

Table 6.28-4 LPUART controller Baud Rate Equation Table

LPUART Peripheral Clock = 12 MHz			
Baud Rate	Mode 0	Mode 1	Mode 2
921600	Not support	Not recommended	BRD=11

460800	Not recommended	BRD=0, EDIVM1 =12	BRD=24
230400	Not recommended	BRD =2, EDIVM1 =12	BRD =50
115200	Not recommended	BRD =6, EDIVM1 =12	BRD =102
57600	BRD =11	BRD =14, EDIVM1 =12	BRD =206
38400	BRD =18	BRD =22, EDIVM1 =12	BRD =311
19200	BRD =37	BRD =61, EDIVM1 =9	BRD =623
9600	BRD =76	BRD =123, EDIVM1 =9	BRD =1248
4800	BRD =154	BRD =248, EDIVM1 =9	BRD =2498

Table 6.28-5 LPUART controller Baud Rate Parameter Setting Example Table

LPUART Peripheral Clock = 12 MHz			
Baud Rate	LPUART_BAUD Value		
	Mode 0	Mode 1	Mode 2
921600	Not support	Not recommended	0x3000_000B
460800	Not recommended	0x2C00_0000	0x3000_0018
230400	Not recommended	0x2C00_0002	0x3000_0032
115200	Not recommended	0x2C00_0006	0x3000_0066
57600	0x0000_000B	0x2C00_000E	0x3000_00CE
38400	0x0000_0012	0x2C00_0016	0x3000_0137
19200	0x0000_0025	0x2900_003D	0x3000_026F
9600	0x0000_004C	0x2900_007B	0x3000_04E0
4800	0x0000_009A	0x2900_00F8	0x3000_09C2

Table 6.28-6 LPUART controller Baud Rate Register Setting Example Table

6.28.5.2 LPUART Controller Baud Rate Compensation

The LPUART controller supports baud rate compensation function. It is used to optimize the precision in each bit. The precision of the compensation is half of LPUART module clock because there is BRCOMDEC (LPUART_BRCOMP[31]) to define the positive or negative compensation in each bit. If the BRCOMDEC (LPUART_BRCOMP[31]) = 0, it is positive compensation for each bit, one more module clock will be append in the compensated bit. If the BRCOMPDEC (LPUART_BRCOMP[31]) = 1, it is negative compensation for each bit, decrease one module clock in the compensated bit.

There is 9-bits location, BRCOMP[8:0] (LPUART_BRCOMP[8:0]), can be configured by user to define the relative bit is compensated or not. BRCOMP[7:0] is used to define the compensation of DAT (LPUART_DAT[7:0]) and BRCOMP[8] is used to define PARITY (LPUART_DAT[8]).

Example:

1. LPUART's peripheral clock = 32.768 kHz and baud rate is 9600 bps

Baud rate is 9600 bps, LPUART peripheral clock is 32.768 kHz → 3.413 peripheral clock/bit

If the baud divider is set 1 (3 peripheral clock/bit), the inaccuracy of each bit is -0.413 peripheral clock and BRCOMPDEC (LPUART_BRCOMP[31]) = 0, so that the BRCOMP (LPUART_BRCOMP[8:0]) can be set as 9'b010100101 = 0xa5.

Bit	Name	Total INACCURACY	BRCOMP Compensated	Final Inaccuracy
0	START	-0.413	x	-0.413
1	LPUART_DAT[0]	-0.826(-0.413-0.413)	1	0.174
2	LPUART_DAT[1]	-0.239(0.174-0.413)	0	-0.239
3	LPUART_DAT[2]	-0.652(-0.239-0.413)	1	0.348
4	LPUART_DAT[3]	-0.065(0.348-0.413)	0	-0.065
5	LPUART_DAT[4]	-0.478(-0.065-0.413)	0	-0.478
6	LPUART_DAT[5]	-0.891(-0.478-0.413)	1	0.109
7	LPUART_DAT[6]	-0.304(0.109-0.413)	0	-0.304
8	LPUART_DAT[7]	-0.717(-0.304-0.413)	1	0.283
9	PARITY	-0.130(0.283-0.413)	0	-0.13

Table 6.28-7 Baud Rate Compensation Example Table 1

2. LPUART's peripheral clock = 32.768 kHz and baud rate is 4800

Baud rate is 4800, LPUART peripheral clock is 32.768 kHz → 6.827 peripheral clock/bit

If the baud divider is set 5 (7 peripheral clock/bit), the inaccuracy of each bit is 0.173 peripheral clock and BRCOMPDEC (LPUART_BRCOMP[31]) = 1, so that the BRCOMP (LPUART_BRCOMP[8:0]) can be set as 9'b010000010 = 0x82.

Bit	Name	Total INACCURACY	BRCOMP Compensated	Final Inaccuracy
0	START	0.173	x	0.173
1	LPUART_DAT[0]	0.346(0.173+0.173)	0	0.346
2	LPUART_DAT[1]	0.519(0.346+0.173)	1	-0.481
3	LPUART_DAT[2]	-0.308(-0.481+0.173)	0	-0.308
4	LPUART_DAT[3]	-0.135(-0.308+0.173)	0	-0.135
5	LPUART_DAT[4]	-0.038(-0.135+0.173)	0	0.038
6	LPUART_DAT[5]	0.211(0.038+0.173)	0	0.211
7	LPUART_DAT[6]	0.384(0.211+0.173)	0	0.384
8	LPUART_DAT[7]	0.557(0.384+0.173)	1	-0.443
9	PARITY	-0.270(-0.443+0.173)	0	-0.270

Table 6.28-8 Baud Rate Compensation Example Table 2

6.28.5.3 LPUART Controller Auto-Baud Rate Function Mode:

Auto-Baud Rate function can measure baud rate of receiving data from LPUART RX pin automatically. When the Auto-Baud Rate measurement is finished, the measuring baud rate is loaded to BRD (LPUART_BAUD[15:0]). Both of the BAUDM1 (LPUART_BAUD[29]) and BAUDM0 (LPUART_BAUD[28]) are set to 1 automatically. LPUART RX data from START bit to 1st rising edge time is set by $2^{ABRDBITS}$ bit time in Auto-Baud Rate function detection frame.

$2^{ABRDBITS}$ bit time from START bit to the 1st rising edge is calculated by setting ABRDBITS

(LPUART_ALTCTL[20:19]). Set ABRDEN (LPUART_ALTCTL[18]) to enable auto-baud rate function. In the beginning when no data transfer, the LPUART RX is held high. Once the falling edge is detected, START bit is received. The auto-baud rate counter will be reset and then start counting. The auto-baud rate counter will be stop when the 1st rising edge is detected. Then, the auto-baud rate counter value divided by ABRDBITS (LPUART_ALTCTL[20:19]) is loaded to BRD (LPUART_BAUD[15:0]) automatically. ABRDEN (LPUART_ALTCTL[18]) is cleared. The Auto-Baud measurement is shown in Figure 6.28-3. Once the auto-baud rate measurement is finished, the ABRDIF (LPUART_FIFOSTS[1]) is set. When auto-baud rate counter is overflow, ABRDTOIF (LPUART_FIFOSTS[2]) is set. If ABRDIF (LPUART_FIFOSTS[1]) or ABRDTOIF (LPUART_FIFOSTS[2]) is set, the auto-baud rate flag ABRIF (LPUART_ALTCTL[17]) generated. If ABRIEN (LPUART_INTEN[18]) is enabled, the auto-baud rate interrupt ABRINT (LPUART_INTSTS[31]) is generated when ABRIF (LPUART_ALTCTL[17]) is set.

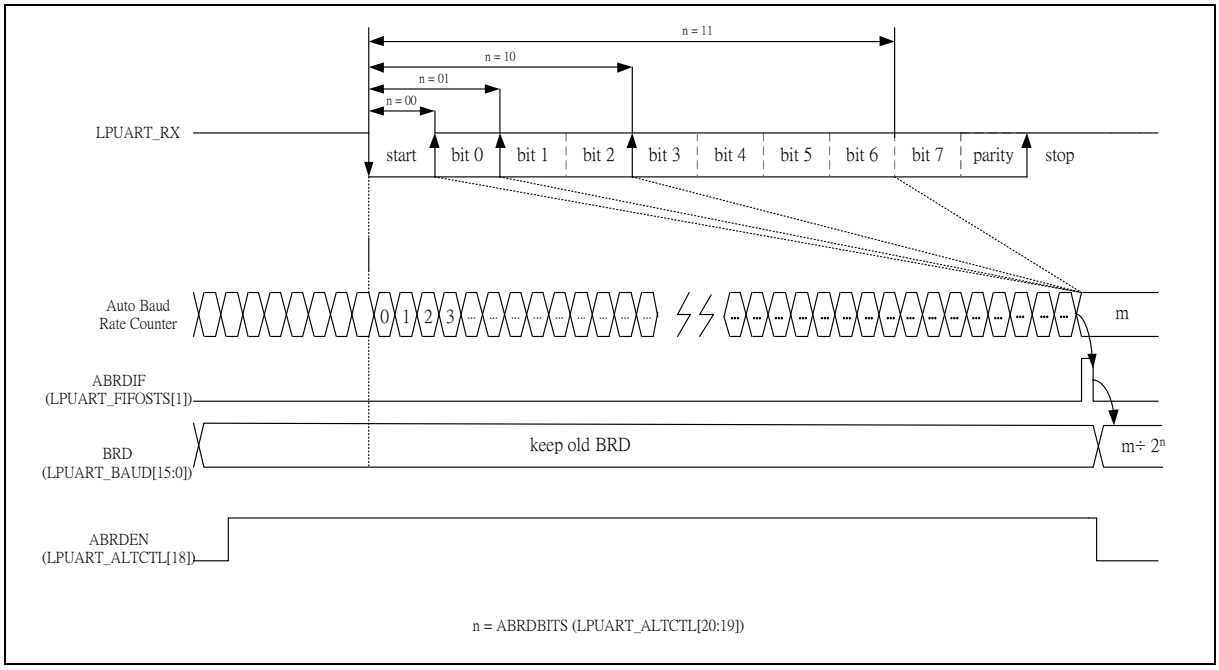


Figure 6.28-3 Auto-Baud Rate Measurement

Programming Sequence Example:

1. Program ABRDBITS (LPUART_ALTCTL[20:19]) to determine LPUART RX data 1st rising edge time from Start by $2^{ABRDBITS}$ bit time.
2. Set ABRIEN (LPUART_INTEN[18]) to enable auto-baud rate function interrupt.
3. Set ABRDEN (LPUART_ALTCTL[18]) to enable auto-baud rate function.
4. ABRDIF (LPUART_FIFOSTS[1]) is set, the auto-baud rate measurement is finished.
5. Operate LPUART transmit and receive action.
6. ABRDTOIF (LPUART_FIFOSTS[2]) is set, if auto-baud rate counter is overflow.
7. Go to Step 3.

6.28.5.4 LPUART Controller Transmit Delay Time Value

The LPUART controller programs DLY (LPUART_TOUT[15:8]) to control the transfer delay time between the last STOP bit and next START bit in transmission. The unit is baud. The operation is shown in Figure 6.28-4.

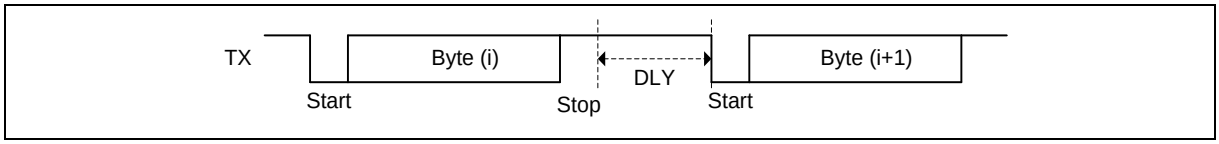


Figure 6.28-4 Transmit Delay Time Operation

6.28.5.5 LPUART Controller FIFO Control and Status

The LPUART controller is built-in with a 16 bytes transmitter FIFO (TX_FIFO) and a 16 bytes receiver FIFO (RX_FIFO) that reduces the number of interrupts presented to the CPU. The CPU can read the status of the LPUART at any time during operation. The reported status information includes condition of the transfer operations being performed by the LPUART, as well as 3 error conditions (parity error, framing error, break interrupt) occur if receiving data has parity, frame or break error. LPUART and RS-485 mode support FIFO control and status function.

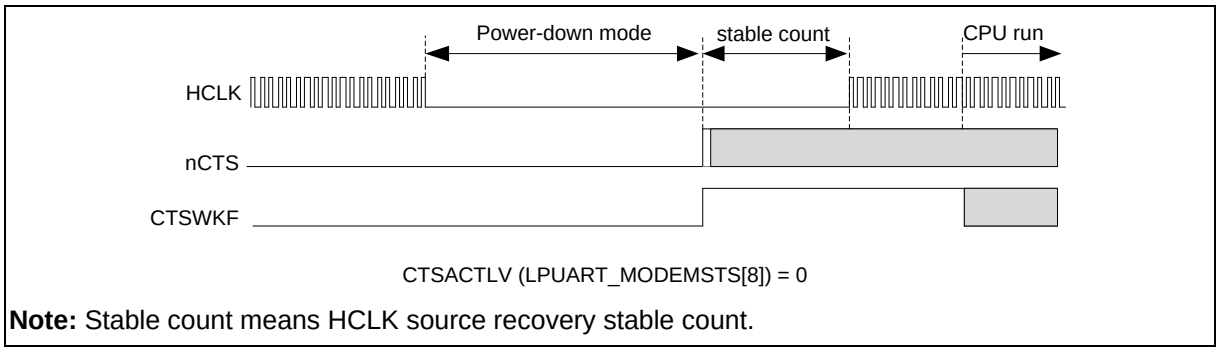
6.28.5.6 LPUART Controller Wake-up Function

The LPUART controller supports wake-up system function. The wake-up function includes nCTS pin, incoming data wake-up, Received Data FIFO reached threshold wake-up, RS-485 Address Match (AAD mode) wake-up and Received Data FIFO threshold time-out wake-up function. CTSWKF (LPUART_WKSTS[0]), DATWKF (LPUART_WKSTS[1]), RFRTWKF (LPUART_WKSTS[2]), RS485WKF (LPUART_WKSTS[3]) or TOUTWKF (LPUART_WKSTS[4]) cause the wake-up interrupt flag WKIF(LPUART_INTSTS[6]) generated. If the WKIEN (LPUART_INTEN[6]) is enabled, the wake-up interrupt flag WKIF(LPUART_INTSTS[6]) cause the wake-up interrupt WKINT (LPUART_INTSTS[14]) generated.

nCTS pin wake-up :

When the system is in Power-down mode and WKCTSEN (LPUART_WKCTL[0]) is set, the toggle of nCTS pin can wake-up system. If the WKCTSEN (LPUART_WKCTL[0]) is enabled, the toggle of nCTS pin cause the nCTS wake-up flag CTSWKF (LPUART_WKSTS[0]) generated. The nCTS wake-up is shown in Figure 6.28-5 and Figure 6.28-6.

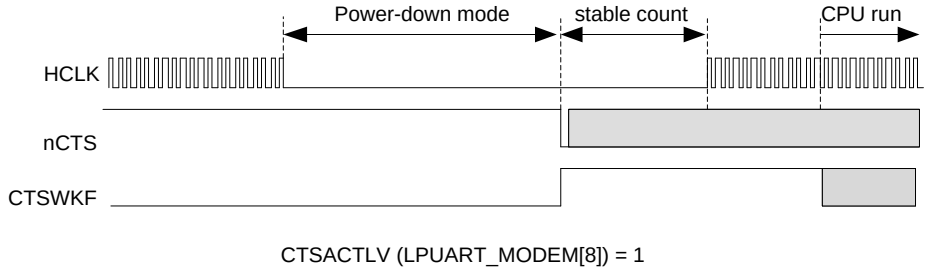
nCTS Wake-up Case 1 (nCTS transition from low to high)



Note: Stable count means HCLK source recovery stable count.

Figure 6.28-5 LPUART nCTS Wake-up Case1

nCTS Wake-up Case 2 (nCTS transition from high to low)



Note: Stable count means HCLK source recovery stable count.

Figure 6.28-6 LPUART nCTS Wake-up Case2

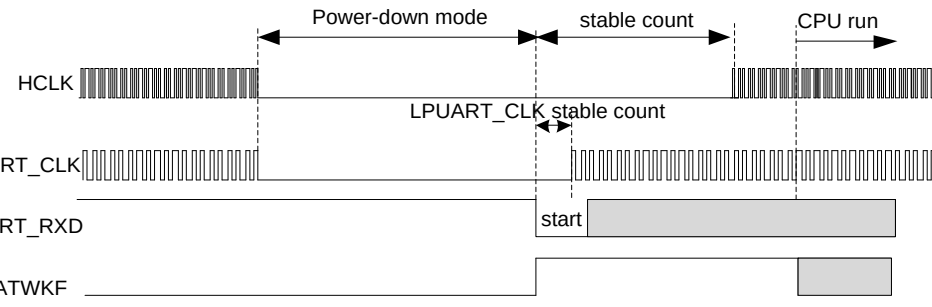
Incoming Data Wake-up

When system is in Power-down mode and the WKDATEN (LPUART_WKCTL[1]) is set, the toggle of incoming data (LPUART_RXD) pin can wake-up the system. In order to receive the incoming data after the system wake-up, the STCOMP (LPUART_DWKCOMP[15:0]) shall be set. These bits field of STCOMP indicate how many clock cycle selected by LPUART_CLK do the LPUART controller can get the 1st bit (START bit) when the system is wakeup from Power-down mode.

When incoming data wakes system up, the incoming data will be received and stored in FIFO. If the WKDATEN (LPUART_WKCTL[1]) is enabled, the toggle of incoming data (LPUART_RXD) pin cause the incoming data wake-up flag DATWKF (LPUART_WKSTS[1]) generated. The incoming data wake-up is shown in Figure 6.28-7.

Note 1: The LPUART controller clock source should be selected as HIRC, and the compensation time for START bit that refer to the M2L31 Datasheet for detailed information about wakeup time electrical characteristics. The STCOMP (LPUART_DWKCOMP[15:0]) = (wake-up stable time) * (HIRC frequency).

Note 2: The value of BRD(LPUART_BAUD[15:0]) should be greater than STCOMP (LPUART_DWKCOMP[15:0]).



Note 1: Stable count means HCLK source recovery stable count.

Note 2: LPUART_CLK stable count means LPUART clock source recovery stable count.

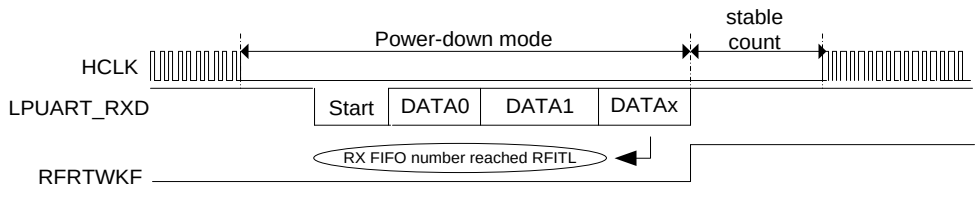
Figure 6.28-7 LPUART Data Wake-up

Received Data FIFO Reached Threshold Wake-up

The received data FIFO threshold reached wake-up function is enabled by setting WKRFRTEN (LPUART_WKCTL[2]). In Power-down mode, when the number of received data in RX FIFO reaches the threshold value RFITL (LPUART_FIFO[7:4]), it can wake-up the system. If the WKRFRTEN (LPUART_WKCTL[2]) is enabled, the number of received data in RX FIFO reaches the threshold value RFITL (LPUART_FIFO[7:4]) cause the received data FIFO reached threshold wake-up flag RFRTWKF (LPUART_WKSTS[2]) generated. The Received Data FIFO reached threshold wake-up is shown in Figure 6.28-8.

Note 1: The LPUART controller clock source should be selected as LXT in Power-down mode to receive data.

Note 2: Before using this wake-up, LPUART0KEEP (LPSCC_CLKKEEP0[19]) should be set to 1.



Note: Stable count means HCLK source recovery stable count.

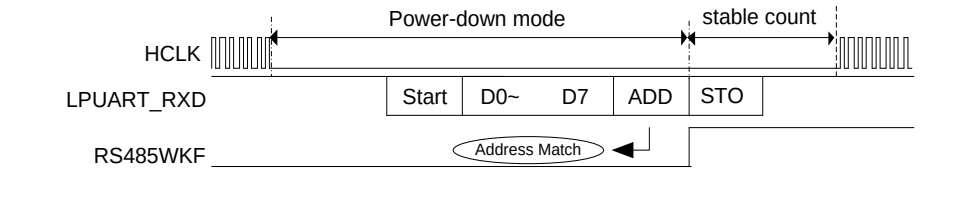
Figure 6.28-8 LPUART Received Data FIFO reached threshold wake-up

RS-485 Address Match (AAD Mode) Wake-up

The RS-485 address match wake-up function is enabled by setting WKRFRTEN (LPUART_WKCTL[2]) and WKRS485EN (LPUART_WKCTL[3]). This function is used for RS-485 Auto Address Detection (AAD) mode in RS-485 function mode and ADDRDEN (LPUART_ALTCTL[15]) is set to 1. In Power-down mode, when an address byte is detected and matches the ADDRMV (LPUART_ALTCTL[31:24]) or the number of received data in RX FIFO reaches the threshold value RFITL (LPUART_FIFO[7:4]), it can wake-up the system. If the WKRS485EN (LPUART_WKCTL[3]) is enabled, when an address byte is detected and matches the ADDRMV (LPUART_ALTCTL[31:24]), the RS485 address match (AAD mode) wake-up flag RS485WKF (LPUART_WKSTS[3]) generated. The RS-485 Address Match (AAD mode) wake-up is shown in Figure 6.28-9.

Note 1: The LPUART controller clock source should be selected as LXT in Power-down mode to receive data.

Note 2: Before using this wake-up, LPUART0KEEP (LPSCC_CLKKEEP0[19]) should be set to 1.



Note: Stable count means HCLK source recovery stable count.

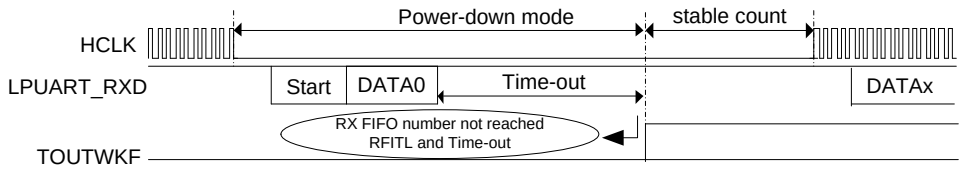
Figure 6.28-9 LPUART RS-485 AAD Mode Address Match Wake-up

Received Data FIFO Threshold Time-out Wake-up

The received data FIFO threshold time-out wake-up function is enabled by setting WKRFRTEN (LPUART_WKCTL[2]) and WKTOUTEN (LPUART_WKCTL[4]). Setting TOCNTEN (LPUART_INTEN[11]) to enable receiver buffer time-out counter. In Power-down mode, when the number of received data in RX FIFO does not reach the threshold value RFITL (LPUART_FIFO[7:4]) and the time-out counter equals to the time-out value TOIC (LPUART_TOUT[7:0]), it can wake-up the system. If the WKTOUTEN (LPUART_WKCTL[4]) is enabled, when the time-out counter equals to the time-out value TOIC (LPUART_TOUT[7:0]), the Received Data FIFO threshold time-out wake-up flag TOUTWKF (LPUART_WKSTS[4]) generated. The Received Data FIFO threshold time-out wake-up is shown in Figure 6.28-10.

Note 1: The LPUART controller clock source should be selected as LXT in Power-down mode to receive data.

Note 2: Before using this wake-up, LPUART0KEEP (LPSCC_CLKKEEP0[19]) should be set to 1.



Note: Stable count means HCLK source recovery stable count.

Figure 6.28-10 LPUART Received Data FIFO threshold time-out wake-up

6.28.5.7 LPUART Controller Interrupt and Status

Each LPUART controller supports ten types of interrupts including:

- Receive Data Available Interrupt (RDAINT)
- Transmit Holding Register Empty Interrupt (THREINT)
- Transmitter Empty Interrupt (TXENDIF)
- Receive Line Status Interrupt (RLSINT)
  - Break Interrupt Flag (BIF)
  - Framing Error Flag (FEF)
  - Parity Error Flag (PEF)
  - RS-485 Address Byte Detect Flag (ADDRDETF)
- MODEM Status Interrupt (MODEMINT)
  - Detect nCTS State Change Flag (CTSDETF)
- Receiver Buffer Time-out Interrupt (RXTOINT)
- Buffer Error Interrupt (BUFERRINT)
  - TX Overflow Error Interrupt Flag (TXOVIF)
  - RX Overflow Error Interrupt Flag (RXOVIF)
- Wake-up Interrupt (WKINT)
  - nCTS Wake-up Flag (CTSWKF)
  - Incoming Data Wake-up Flag (DATWKF)
  - Received Data FIFO Reached Threshold Wake-up Flag (RFRTWKF)
  - RS-485 Address Match (AAD mode) Wake-up Flag (RS485WKF)
  - Received Data FIFO Threshold Time-out Wake-up Flag (TOUTWKF)
- Auto-Baud Rate Interrupt (ABRINT)
  - Auto-baud Rate Detect Interrupt Flag (ABRDIF)
  - Auto-baud Rate Detect Time-out Interrupt Flag (ABRDTOIF)
- Single-wire Bit Error Detect Interrupt (SWBEINT)

Table 6.28-9 describes the interrupt sources and flags. The interrupt is generated when the interrupt flag is generated and the interrupt enable bit is set. User must clear the interrupt flag after the interrupt is generated.

Interrupt Source	Interrupt Indicator	Interrupt Enable Bit	Interrupt Flag	Flag Caused By	Flag Cleared By

Receive Data Available Interrupt	RDAINT	RDAIEN	RDAIF	N/A	Read LPUART_DAT
Transmit Holding Register Empty Interrupt	THREINT	THREIEN	THREIF	N/A	Write LPUART_DAT
Transmitter Empty Interrupt	TXENDINT	TXENDIEN	TXENDIF	N/A	Write LPUART_DAT
Receive Line Status Interrupt	RLSINT	RLSIEN	RLSIF	RLSIF = BIF	Write '1' to BIF
				RLSIF = FEF	Write '1' to FEF
				RLSIF = PEF	Write '1' to PEF
				RLSIF = ADDRDETF	Write '1' to ADDRDETF
Modem Status Interrupt	MODEMINT	MODEMIEN	MODEMIF	MODEMIF = CTSDETF	Write '1' to CTSDETF
Receiver Buffer Time-out Interrupt	RXTOINT	RXTOIEN	RXTOIF	N/A	Read LPUART_DAT
Buffer Error Interrupt	BUFERRINT	BUFERRIEN	BUFERRIF	BUFERRIF = TXOVIF	Write '1' to TXOVIF
				BUFERRIF = RXOVIF	Write '1' to RXOVIF
Wake-up Interrupt	WKINT	WKIEN	WKIF	WKIF = CTSWKF	Write '1' to CTSWKF
				WKIF = DATWKF	Write '1' to DATWKF
				WKIF = RFRTWKF	Write '1' to RFRTWKF
				WKIF = RS485WKF	Write '1' to RS485WKF
				WKIF = TOUTWKF	Write '1' to TOUTWKF
Auto-Baud Rate Interrupt	ABRINT	ABRIEN	ABRIF	ABRIF = ABRDIF	Write '1' to ABRDIF
				ABRIF = ABRDTOIF	Write '1' to ABRDTOIF
Single-wire Bit Error Detect Interrupt	SWBEINT	SWBEIEN	SWBEIF	N/A	Writing '1' to SWBEIF

Table 6.28-9 LPUART controller Interrupt Source and Flag List

6.28.5.8 LPUART Function Mode

The LPUART controller provides LPUART function (Setting FUNCSEL (LPUART_FUNCSEL[2:0]) to '000' to enable LPUART function mode). The LPUART baud rate is up to 10 Mbps.

The LPUART provides full-duplex and asynchronous communications. The transmitter and receiver contain 16 bytes FIFO for payloads. User can program receiver buffer trigger level and receiver buffer time-out detection for receiver. The transmitting data delay time between the last stop and the next START bit can be programed by setting DLY (LPUART_TOUT[15:8]) register. The LPUART supports hardware auto-flow control that provides programmable nRTS flow control trigger level. The number of data bytes in RX FIFO is equal to or greater than RTSTRGLV (LPUART_FIFO[19:16]), the nRTS is de-asserted.

LPUART Line Control Function

The LPUART controller supports fully programmable serial-interface characteristics by setting the LPUART_LINE register. User can program LPUART_LINE register for the word length, STOP bit and

PARITY bit setting. Table 6.28-10 and Table 6.28-11 list the LPUART word, STOP bit length and the PARITY bit settings.

NSB (LPUART_LINE[2])	WLS (LPUART_LINE[1:0])	Word Length (Bit)	Stop Length (Bit)
0	00	5	1
0	01	6	1
0	10	7	1
0	11	8	1
1	00	5	1.5
1	01	6	2
1	10	7	2
1	11	8	2

Table 6.28-10 LPUART Line Control of Word and Stop Length Setting

Parity Type	SPE (LPUART_LINE[5])	EPE (LPUART_LINE[4])	PSS (LPUART_LINE[7])	PBE (LPUART_LINE[3])	Description
No Parity	x	x	x	0	No PARITY bit output.
Parity source from LPUART_DAT	x	x	1	1	PARITY bit is generated and checked by software.
Odd Parity	0	0	0	1	Odd Parity is calculated by adding all the “1’s” in a data stream and adding a PARITY bit to the total bits, to make the total count an odd number.
Even Parity	0	1	0	1	Even Parity is calculated by adding all the “1’s” in a data stream and adding a PARITY bit to the total bits, to make the count an even number.
Forced Mask Parity	1	0	0	1	PARITY bit always logic 1. PARITY bit on the serial byte is set to “1” regardless of total number of “1’s” (even or odd counts).
Forced Space Parity	1	1	0	1	PARITY bit always logic 0. PARITY bit on the serial byte is set to “0” regardless of total number of “1’s” (even or odd counts).

Table 6.28-11 LPUART Line Control of PARITY Bit Setting

LPUART Auto-Flow Control Function

The LPUART supports auto-flow control function that uses two signals, nCTS (clear-to-send) and nRTS (request-to-send), to control the flow of data transfer between the LPUART and external devices (e.g. Modem). When auto-flow is enabled, the LPUART is not allowed to receive data until the LPUART asserts nRTS to external device. When the number of bytes stored in the RX FIFO equals the value of RTSTRGLV (LPUART_FIFO[19:16]), the nRTS is de-asserted. The LPUART sends data out when LPUART detects nCTS is asserted from external device. If the valid asserted nCTS is not detected, the LPUART will not send data out. The auto flow control block diagram is shown in Figure 6.28-11.

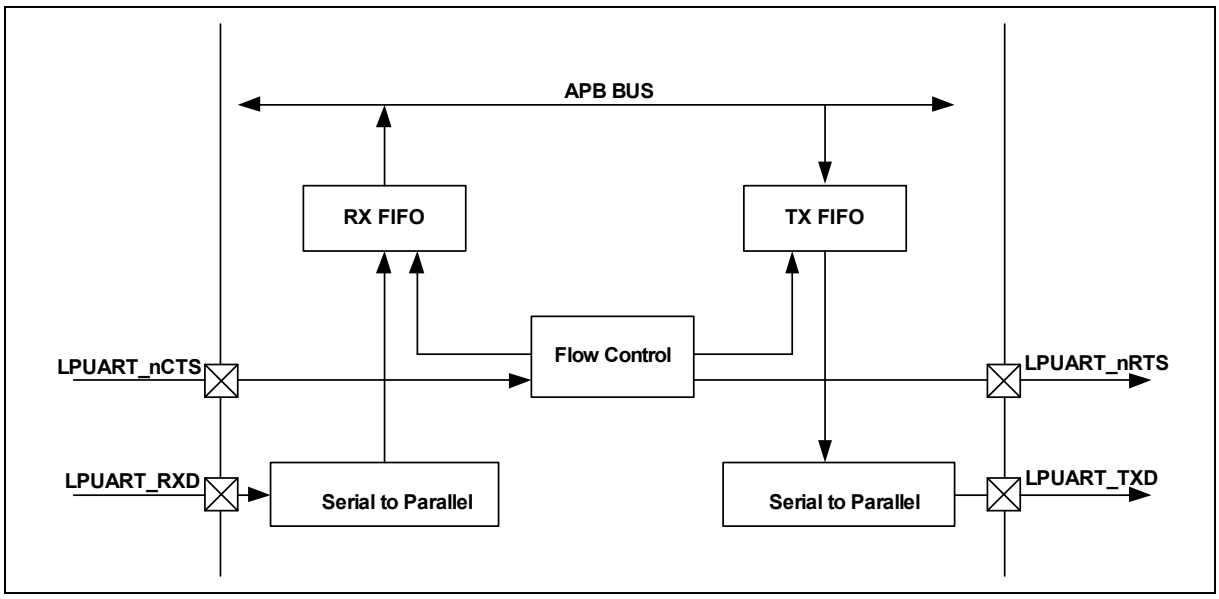


Figure 6.28-11 Auto-Flow Control Block Diagram

Figure 6.28-12 demonstrates the nCTS auto-flow control of LPUART function mode. User must set ATOCTSEN (LPUART_INTEN[13]) to enable nCTS auto-flow control function. The CTSACTLV (LPUART_MODEMSTS[8]) can set nCTS pin input active state. The CTSDETF (LPUART_MODEMSTS[0]) is set when any state change of nCTS pin input has occurred, and then TX data will be automatically transmitted from TX FIFO.

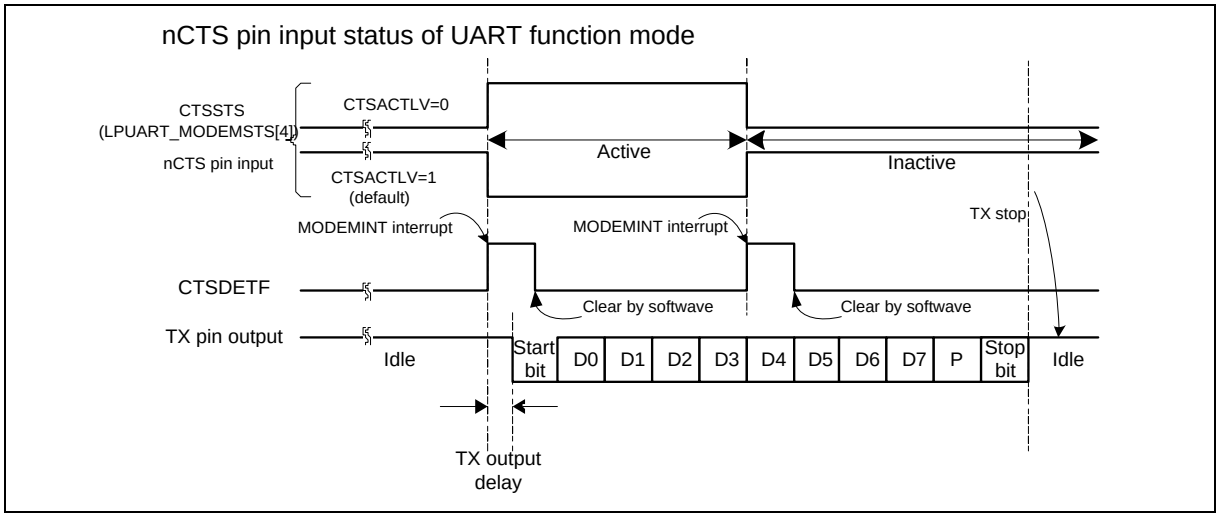


Figure 6.28-12 LPUART nCTS Auto-Flow Control Enabled

As shown in Figure 6.28-13, in LPUART nRTS auto-flow control mode (ATORTSEN (LPUART_INTEN[12]) = 1), the nRTS internal signal is controlled by LPUART FIFO controller with RTSTRGLV (LPUART_FIFO[19:16]) trigger level.

Setting RTSACTLV (LPUART_MODEM[9]) can control the nRTS pin output is inverse or non-inverse from nRTS signal. User can read the RTSSTS (LPUART_MODEM[13]) bit to get real nRTS pin output voltage logic status.

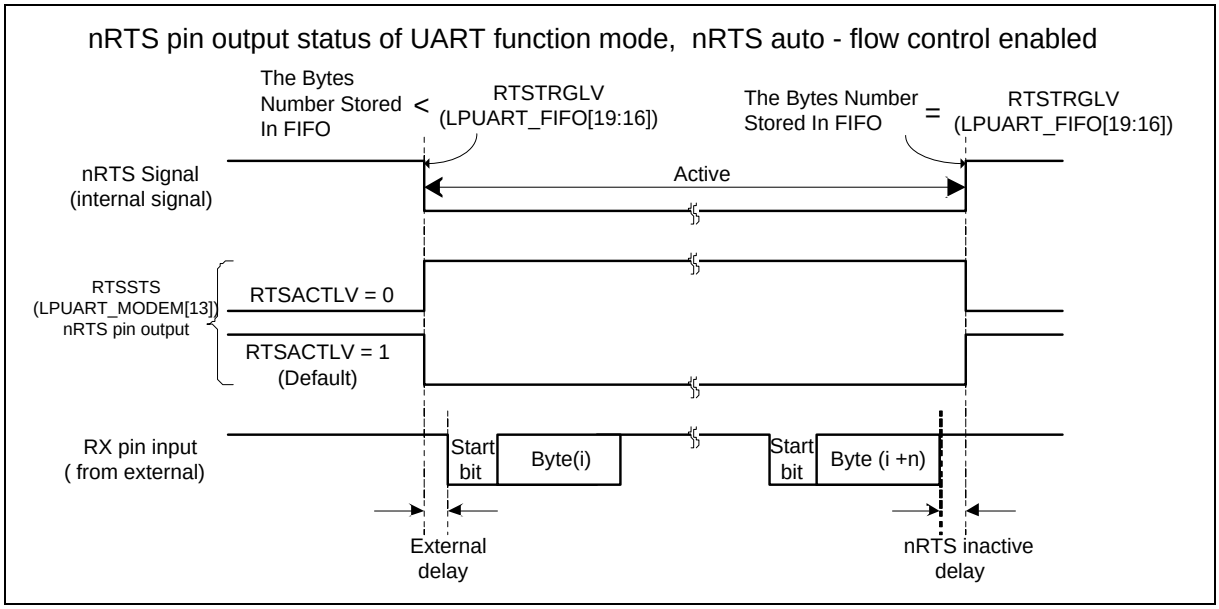


Figure 6.28-13 LPUART nRTS Auto-Flow Control Enabled

As shown in Figure 6.28-14, in software mode (ATORTSEN (LPUART_INTEN[12]) = 0), the nRTS flow is directly controlled by software programming of RTS (LPUART_MODEM[1]) control bit.

Setting RTSACTLV (LPUART_MODEM[9]) can control the nRTS pin output is inverse or non-inverse from RTS (LPUART_MODEM[1]) control bit. User can read the RTSSTS (LPUART_MODEM[13]) bit to get real nRTS pin output voltage logic status.

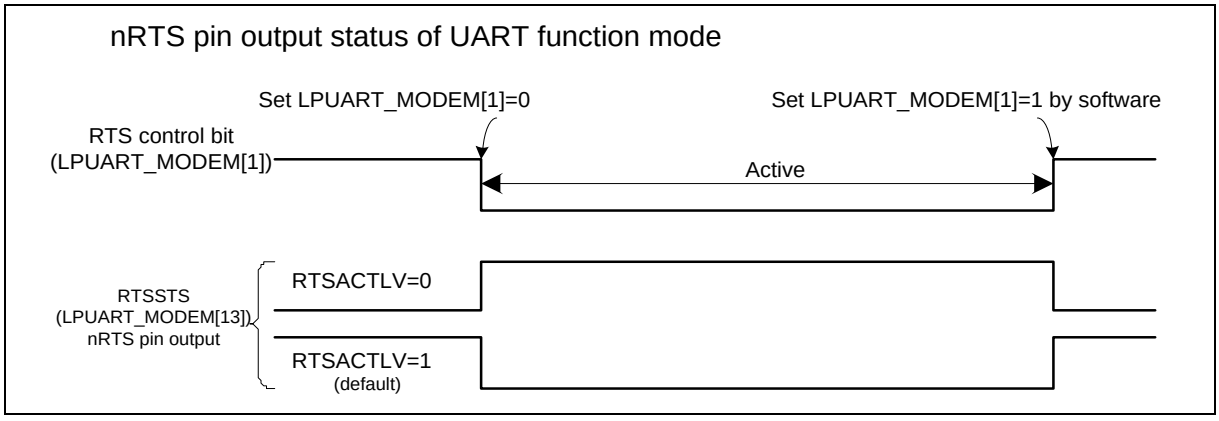


Figure 6.28-14 LPUART nRTS Auto-Flow with Software Control

6.28.5.9 RS-485 Function Mode

Another alternate function of LPUART controller is RS-485 function (user must set FUNCSEL (LPUART_FUNCSEL[2:0]) to '011' to enable RS-485 function), and direction control provided by nRTS pin from an asynchronous serial port. The RS-485 transceiver control is implemented by using the nRTS control signal to enable the RS-485 driver. Many characteristics of the RX and TX are the same as LPUART in RS-485 mode.

The LPUART controller can be configured as a RS-485 addressable slave or a RS-485 master. RS-485 master transmitter will identify an address character by setting the parity (9th bit) to 1. For data

characters, the parity is set to 0. Software can use LPUART_LINE register to control the 9th bit (When the PBE, EPE and SPE are set, the 9th bit is transmitted 0 and when PBE and SPE are set and EPE is cleared, the 9th bit is transmitted 1).

The controller supports three operation modes: RS-485 Normal Multidrop Operation Mode (NMM), RS-485 Auto Address Detection Operation Mode (AAD) and RS-485 Auto Direction Control Operation Mode (AUD). Software can choose any operation mode by programming the LPUART_ALTCTL register, and drive the transfer delay time between the last STOP bit leaving the TX FIFO and the de-assertion of by setting DLY (LPUART_TOUT[15:8]) register.

RS-485 Normal Multidrop Operation Mode (NMM)

In RS-485 Normal Multidrop Operation Mode (RS485NMM (LPUART_ALTCTL[8]) = 1), software must first decide the data which before the address byte be detected will be stored in RX FIFO or not. To ignore any data before address byte is detected, set RXOFF (LPUART_FIFO[8]) to 1 and enable RS485NMM (LPUART_ALTCTL[8]). The receiver will ignore any data until an address byte is detected (bit 9 = 1) and store the detected address byte data in the RX FIFO. To receive any data before address byte is detected, set RXOFF (LPUART_FIFO[8]) to 0 and enable RS485NMM (LPUART_ALTCTL[8]).

If an address byte is detected (bit 9 = 1), it will generate an interrupt to CPU and RXOFF (LPUART_FIFO[8]) can decide whether accepting the following data bytes are stored in the RX FIFO. If software disables receiver by setting RXOFF (LPUART_FIFO[8]) register, when a next address byte is detected, the controller will clear the RXOFF (LPUART_FIFO[8]) bit and the address byte data will be stored in the RX FIFO.

RS-485 Auto Address Detection Operation Mode (AAD)

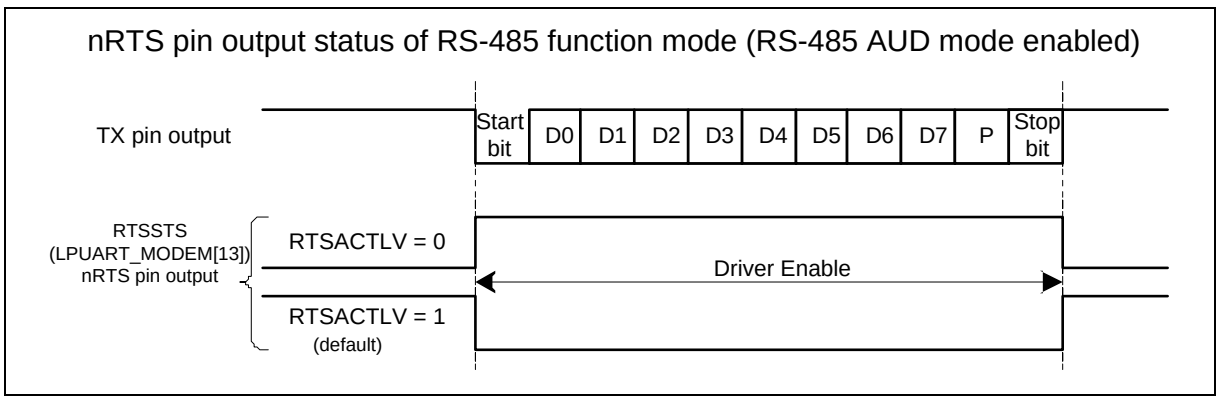
In RS-485 Auto Address Detection Operation Mode (RS485AAD (LPUART_ALTCTL[9]) = 1), the receiver will ignore any data until an address byte is detected (bit 9 = 1) and the address byte data matches the ADDRMV (LPUART_ALTCTL[31:24]) value. The address byte data will be stored in the RX FIFO. The all received byte data will be accepted and stored in the RX FIFO until an address byte data not match the ADDRMV (LPUART_ALTCTL[31:24]) value.

RS-485 Auto Direction Function (AUD)

RS-485 controllers supports RS-485 auto direction control function (RS485AUD (LPUART_ALTCTL[10]) = 1). The RS-485 transceiver control is implemented by using the nRTS control signal from an asynchronous serial port. The nRTS line is connected to the RS-485 transceiver enable pin such that setting the nRTS line to high (logic 1) enables the RS-485 transceiver. Setting the nRTS line to low (logic 0) puts the transceiver into the tri-state condition. User can set RTSACTLV in LPUART_MODEM register to change the nRTS driving level.

Figure 6.28-15 demonstrates the RS-485 nRTS driving level in AUD mode. The nRTS pin will be automatically driven during TX data transmission.

Setting RTSACTLV (LPUART_MODEM[9]) can control nRTS pin output driving level. User can read the RTSSTS (LPUART_MODEM[13]) bit to get real nRTS pin output voltage logic status.

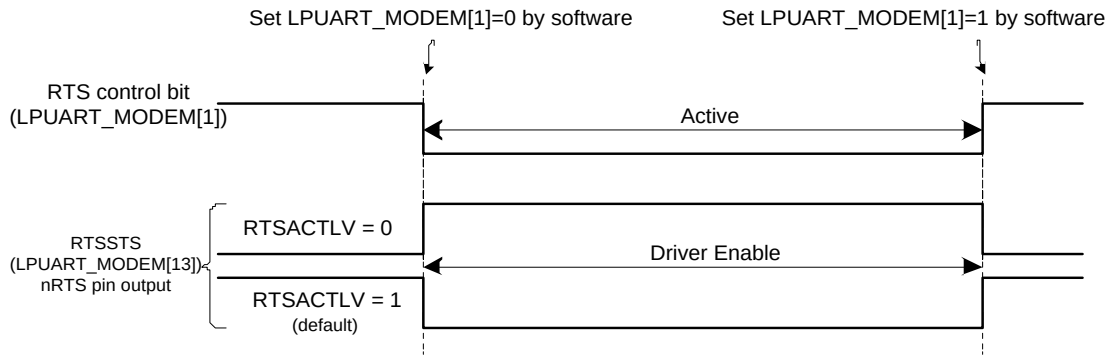


Note: RS485AUD (LPUART_ALTCTL[10]) = 1, the nRTS pin output by hardware control only.

Figure 6.28-15 RS-485 nRTS Driving Level in Auto Direction Mode

Figure 6.28-16 demonstrates the RS-485 nRTS driving level in software control (RS485AUD (LPUART_ALTCTL[10]) = 0). The nRTS driving level is controlled by programing the RTS (LPUART_MODEM[1]) control bit.

Setting RTSACTLV (LPUART_MODEM[9]) can control the nRTS pin output is inverse or non-inverse from RTS (LPUART_MODEM[1]) control bit. User can read the RTSSTS (LPUART_MODEM[13]) bit to get real nRTS pin output voltage logic status. The structure of RS-485 frame is shown in Figure 6.28-17.

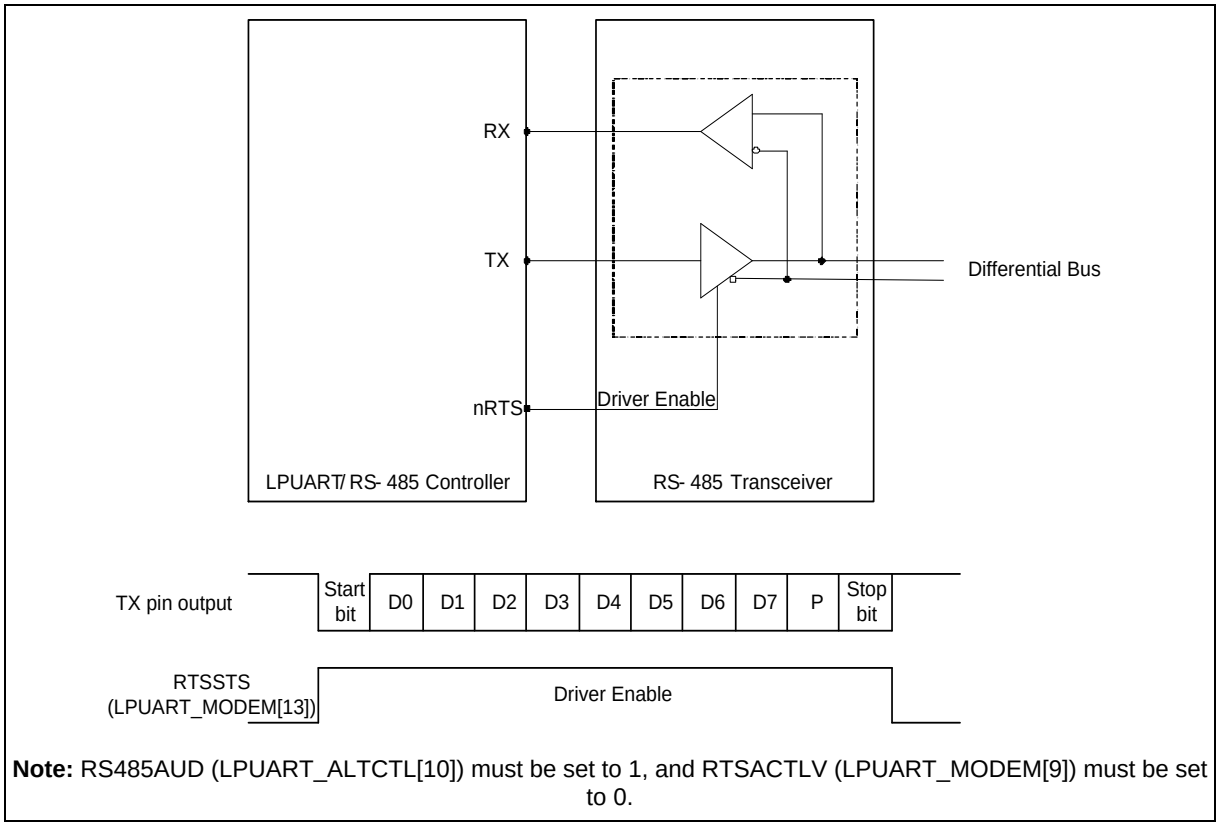


Note: RS485AUD (LPUART_ALTCTL[10]) = 0, support nRTS control bit by software control only.

Figure 6.28-16 RS-485 nRTS Driving Level with Software Control

Programming Sequence Example

1. Program FUNCSEL in LPUART_FUNCSEL to select RS-485 function.
2. Program the RXOFF (LPUART_FIFO[8]) to determine enable or disable the receiver RS-485 receiver.
3. Program the RS485NMM (LPUART_ALTCTL[8]) or RS485AAD (LPUART_ALTCTL[9]) mode.
4. If the RS485AAD (LPUART_ALTCTL[9]) mode is selected, the ADDRMV (LPUART_ALTCTL[31:24]) is programmed for auto address match value.
5. Determine auto direction control by programming RS485AUD (LPUART_ALTCTL[10]).



Note: RS485AUD (LPUART_ALTCTL[10]) must be set to 1, and RTSACTLV (LPUART_MODEM[9]) must be set to 0.

Figure 6.28-17 Structure of RS-485 Frame

6.28.5.10 LPUART Single-wire Half Duplex

The LPUART controller provides single-wire half duplex function in LPUART function mode. Setting FUNCSEL (LPUART_FUNCSEL[2:0]) to '100' to enable the LPUART Single-wire function. The single-wire bus is idle (RXIDLE (LPUART_FIFOSTS[29]) = 1) when in RX state. Before writing data to transmit buffer (LPUART_DAT[7:0]), the bus state should be checked in idle (RXIDLE (LPUART_FIFOSTS[29]). By writing data to transmit buffer, the bus state transfers to TX state immediately. After the transmission, the bus state transfers from TX state to RX state.

The LPUART will not allowed to receive data when in single-wire half duplex function TX mode. If nRTS is asserted in TX mode, nRTS will make the docking LPUART device send out data and cause bus confliction. To reduce the bus confliction, the LPUART controller supports flow control function and bit error detection but does not support auto-flow control function. The nRTS is automatically inactivated in single-wire TX state. When in TX state, the LPUART controller will monitor bus state. If the bus state is not equal to TX state, the SWBEIF (LPUART_INTSTS[16]) is set.

6.28.5.11 LPPDMA Transfer Function

The LPUART controller supports LPPDMA transfer function.

By configuring LPPDMA parameter and set LPUART_DAT as the LPPDMA destination address. When TXPDMAEN (LPUART_INTEN[14]) is set to 1, the controller will issue request to LPPDMA controller to start the LPPDMA transmission process automatically.

By configuring LPPDMA parameter and set LPUART_DAT as the LPPDMA source address. When RXPDMAEN (LPUART_INTEN[15]) is set to 1, the controller will start the LPPDMA reception process. LPUART controller will issue request to LPPDMA controller automatically when there is data in the RX FIFO buffer.

6.28.5.12 Auto-operation Mode

The LPUART controller supports auto-operation mode when the chip is in NPD0/1/2/3/4 mode. The block diagram of LPUART auto-operation mode TX state shows in Figure 6.28-19 . Set the AOEN (LPUART_AUTOCTL[31]) to 1 can enable LPUART auto-operation mode. In auto-operation mode, the LPUART can be triggered to TX state when TRIGEN (LPUART_AUTOCTL[4]) is set to 1, even chip is under NPD0/1/2/3/4 mode. The LPUART auto-operation mode trigger source include LPTMR0/1, TTMR0/1, WKIOA0, WKIOB0, WKIOC0, and WKIOD0. The trigger source can be selected by TRIGSEL (LPUART_AUTOCTL[3:0]). User can also trigger LPUART to TX state in auto-operation mode by set SWTRIG (LPUART_AUTOCTL[9]) to 1. When the LPUART is triggered to TX state, the auto-operation circuit will automaticaly turn on the clocks, request LPPDMA for TX data, transmit the TX data out, and turn off the clocks after the LPPDMA transmissions are completed (LPPDMA table finishes).

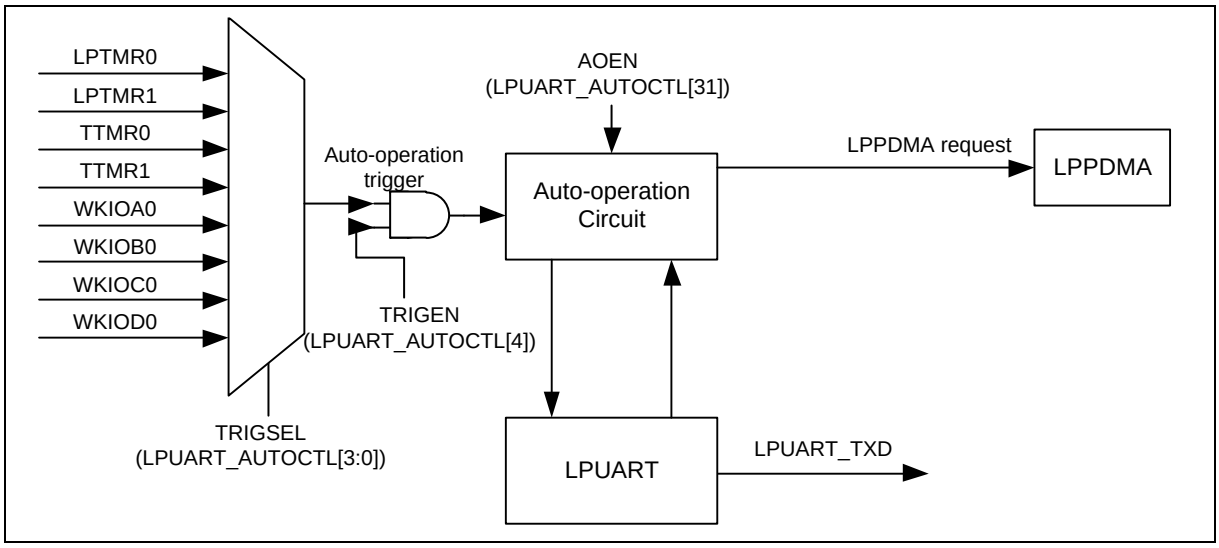


Figure 6.28-18 LPUART Auto-operation mode TX state Block Diagram

LPUART also supports RX auto-operations. The block diagram of LPUART auto-operation mode RX state shows in Figure 6.28-19 . In auto-operation mode, the LPUART can be triggered to RX state when receive incoming data, even chip is under NPD0/1/2/3/4 mode. When the LPUART is triggered to RX state, the auto-operation circuit will automaticaly turn on the clocks, receive the RX data, request LPPDMA to transmit RX data, and turn off the clocks after RX time-out (HWTOIF (LPUART_INTSTS[20]) is set). If WKAOTOEN (LPUART_AUTOCTL[8]) is set to 1, the RX time-out event will also wake-up the system and set AOTOWKF (LPUART_AUTOSTS[0]).

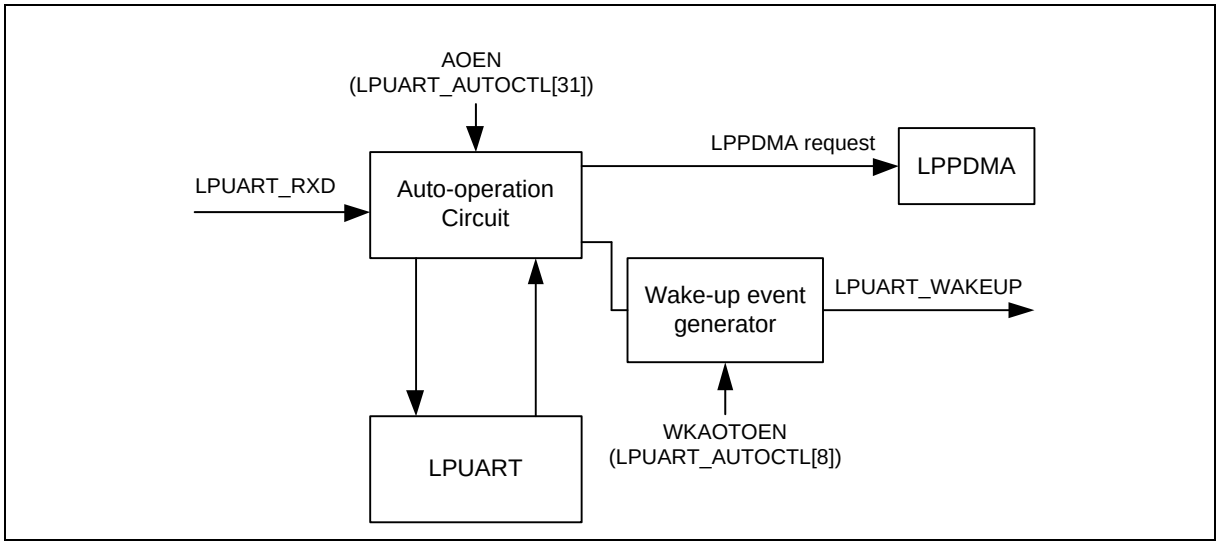


Figure 6.28-19 LPUART Auto-operation mode RX state Block Diagram

6.28.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
LPUART Base Address: **LPUARTx_BA = 0x400E_0000** **x=0**				
LPUART_DAT	LPUARTx_BA+0x00	R/W	LPUART Receive/Transmit Buffer Register	0xXXXX_XXXX
LPUART_INTEN	LPUARTx_BA+0x04	R/W	LPUART Interrupt Enable Register	0x0000_0000
LPUART_FIFO	LPUARTx_BA+0x08	R/W	LPUART FIFO Control Register	0x0000_0101
LPUART_LINE	LPUARTx_BA+0x0C	R/W	LPUART Line Control Register	0x0000_0000
LPUART_MODEM	LPUARTx_BA+0x10	R/W	LPUART Modem Control Register	0x0000_0200
LPUART_MODEMSTS	LPUARTx_BA+0x14	R/W	LPUART Modem Status Register	0x0000_0110
LPUART_FIFOSTS	LPUARTx_BA+0x18	R/W	LPUART FIFO Status Register	0xB040_4000
LPUART_INTSTS	LPUARTx_BA+0x1C	R/W	LPUART Interrupt Status Register	0x0040_0002
LPUART_TOUT	LPUARTx_BA+0x20	R/W	LPUART Time-out Register	0x0000_0000
LPUART_BAUD	LPUARTx_BA+0x24	R/W	LPUART Baud Rate Divider Register	0x0F00_0000
LPUART_ALTCTL	LPUARTx_BA+0x2C	R/W	LPUART Alternate Control/Status Register	0x0000_000C
LPUART_FUNCSEL	LPUARTx_BA+0x30	R/W	LPUART Function Select Register	0x0000_0000
LPUART_BRCOMP	LPUARTx_BA+0x3C	R/W	LPUART Baud Rate Compensation Register	0x0000_0000
LPUART_WKCTL	LPUARTx_BA+0x40	R/W	LPUART Wake-up Control Register	0x0000_0000
LPUART_WKSTS	LPUARTx_BA+0x44	R/W	LPUART Wake-up Status Register	0x0000_0000
LPUART_DWKCOMP	LPUARTx_BA+0x48	R/W	LPUART Incoming Data Wake-up Compensation Register	0x0000_0000

LPUART_RS485DD	LPUARTx_BA+0x4C	R/W	LPUART RS485 Transceiver Deactivate Delay Register	0x0000_0000
LPUART_AUTOCTL	LPUARTx_BA+0x58	R/W	LPUART Automatic Operation Control Register	0x0000_0000
LPUART_AUTOSTS	LPUARTx_BA+0x5C	R/W	LPUART Automatic Operation Status Register	0x0000_0000

6.28.7 Register Description

LPUART Receive/Transmit Buffer Register (LPUART_DAT)

Register	Offset	R/W	Description	Reset Value
LPUART_DAT	LPUARTx_BA+0x00	R/W	LPUART Receive/Transmit Buffer Register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**PARITY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAT							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**PARITY**	**PARITY Bit Receive/Transmit Buffer** Write Operation: By writing to this bit, the PARITY bit will be stored in transmitter FIFO. If PBE (LPUART_LINE[3]) and PSS (LPUART_LINE[7]) are set, the LPUART controller will send out this bit follow the DAT (LPUART_DAT[7:0]) through the LPUART_TXD. Read Operation: If PBE (LPUART_LINE[3]) and PSS (LPUART_LINE[7]) are enabled, the PARITY bit can be read by this bit. **Note:** This bit has effect only when PBE (LPUART_LINE[3]) and PSS (LPUART_LINE[7]) are set.
[7:0]	**DAT**	**Data Receive/Transmit Buffer** Write Operation: By writing one byte to this register, the data byte will be stored in transmitter FIFO. The LPUART controller will send out the data stored in transmitter FIFO top location through the LPUART_TXD. Read Operation: By reading this register, the LPUART controller will return an 8-bit data received from receiver FIFO.

LPUART Interrupt Enable Register (LPUART_INTEN)

Register	Offset	R/W	Description	Reset Value
LPUART_INTEN	LPUARTx_BA+0x04	R/W	LPUART Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	TXENDIEN	Reserved			ABRIEN	Reserved	SWBEIEN
15	14	13	12	11	10	9	8
RXPDMAEN	TXPDMAEN	ATOCTSEN	ATORTSEN	TOCNTEN	Reserved		
7	6	5	4	3	2	1	0
Reserved	WKIEN	BUFERRIEN	RXTOIEN	MODEMIEN	RLSIEN	THREIEN	RDAIEN

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22]	**TXENDIEN**	**Transmitter Empty Interrupt Enable Bit** If TXENDIEN (LPUART_INTEN[22]) is enabled, the Transmitter Empty interrupt TXENDINT (LPUART_INTSTS[30]) will be generated when TXENDIF (LPUART_INTSTS[22]) is set (TX FIFO (LPUART_DAT) is empty and the STOP bit of the last byte has been transmitted). 0 = Transmitter empty interrupt Disabled. 1 = Transmitter empty interrupt Enabled.
[21:19]	**Reserved**	Reserved.
[18]	**ABRIEN**	**Auto-baud Rate Interrupt Enable Bit** 0 = Auto-baud rate interrupt Disabled. 1 = Auto-baud rate interrupt Enabled.
[17]	**Reserved**	Reserved.
[16]	**SWBEIEN**	**Single-wire Bit Error Detection Interrupt Enable Bit** Set this bit, the Single-wire Half Duplex Bit Error Detection Interrupt SWBEINT(LPUART_INTSTS[24]) is generated when Single-wire Bit Error Detection SWBEIF(LPUART_INTSTS[16]) is set. 0 = Single-wire Bit Error Detect Interrupt Disabled. 1 = Single-wire Bit Error Detect Interrupt Enabled. **Note:** This bit is valid when FUNCSEL (LPUART_FUNCSEL[2:0]) is select LPUART Single-wire mode.
[15]	**RXPDMAEN**	**RX LPPDMA Enable Bit** This bit can enable or disable RX LPPDMA service. 0 = RX LPPDMA Disabled. 1 = RX LPPDMA Enabled. **Note:** If RLSIEN (LPUART_INTEN[2]) is enabled and HWRLSINT (LPUART_INTSTS[26]) is set to 1, the RLS (Receive Line Status) Interrupt is caused. If RLS interrupt is caused by Break Error Flag BIF(LPUART_FIFOSTS[6]), Frame Error Flag FEF(LPUART_FIFO[5]) or Parity Error Flag PEF(LPUART_FIFOSTS[4]), LPUART LPPDMA receive request

		operation is stopped. Clear Break Error Flag BIF or Frame Error Flag FEF or Parity Error Flag PEF by writing “1” to corresponding BIF, FEF and PEF to make LPUART LPPDMA receive request operation continue.
[14]	**TXPDMAEN**	**TX LPPDMA Enable Bit** 0 = TX LPPDMA Disabled. 1 = TX LPPDMA Enabled. **Note:** If RLSIEN (LPUART_INTEN[2]) is enabled and HWRLSINT (LPUART_INTSTS[26]) is set to 1, the RLS (Receive Line Status) Interrupt is caused. If RLS interrupt is caused by Break Error Flag BIF(LPUART_FIFOSTS[6]), Frame Error Flag FEF(LPUART_FIFO[5]) or Parity Error Flag PEF(LPUART_FIFOSTS[4]), LPUART LPPDMA transmit request operation is stopped. Clear Break Error Flag BIF or Frame Error Flag FEF or Parity Error Flag PEF by writing “1” to corresponding BIF, FEF and PEF to make LPUART LPPDMA transmit request operation continue.
[13]	**ATOCTSEN**	**nCTS Auto-flow Control Enable Bit** 0 = nCTS auto-flow control Disabled. 1 = nCTS auto-flow control Enabled. **Note:** When nCTS auto-flow is enabled, the LPUART will send data to external device if nCTS input assert (LPUART will not send data to device until nCTS is asserted).
[12]	**ATORTSEN**	**nRTS Auto-flow Control Enable Bit** 0 = nRTS auto-flow control Disabled. 1 = nRTS auto-flow control Enabled. **Note:** When nRTS auto-flow is enabled, if the number of bytes in the RX FIFO equals the RTSTRGLV (LPUART_FIFO[19:16]), the LPUART will de-assert nRTS signal.
[11]	**TOCNTEN**	**Receive Buffer Time-out Counter Enable Bit** 0 = Receive Buffer Time-out counter Disabled. 1 = Receive Buffer Time-out counter Enabled.
[10:7]	**Reserved**	Reserved.
[6]	**WKIEN**	**Wake-up Interrupt Enable Bit** 0 = Wake-up Interrupt Disabled. 1 = Wake-up Interrupt Enabled.
[5]	**BUFERRIEN**	**Buffer Error Interrupt Enable Bit** 0 = Buffer error interrupt Disabled. 1 = Buffer error interrupt Enabled.
[4]	**RXTOIEN**	**RX Time-out Interrupt Enable Bit** 0 = RX time-out interrupt Disabled. 1 = RX time-out interrupt Enabled.
[3]	**MODEMIEN**	**Modem Status Interrupt Enable Bit** 0 = Modem status interrupt Disabled. 1 = Modem status interrupt Enabled.
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable Bit** 0 = Receive Line Status interrupt Disabled. 1 = Receive Line Status interrupt Enabled.
[1]	**THREIEN**	**Transmit Holding Register Empty Interrupt Enable Bit** 0 = Transmit holding register empty interrupt Disabled. 1 = Transmit holding register empty interrupt Enabled.
[0]	**RDAIEN**	**Receive Data Available Interrupt Enable Bit** 0 = Receive data available interrupt Disabled.

		1 = Receive data available interrupt Enabled.

LPUART FIFO Control Register (LPUART_FIFO)

Register	Offset	R/W	Description	Reset Value
LPUART_FIFO	LPUARTx_BA+0x08	R/W	LPUART FIFO Control Register	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				RTSTRGLV			
15	14	13	12	11	10	9	8
Reserved							RXOFF
7	6	5	4	3	2	1	0
RFITL				Reserved	TXRST	RXRST	Reserved

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**RTSTRGLV**	**nRTS Trigger Level for Auto-flow Control** 0000 = nRTS Trigger Level is 1 byte. 0001 = nRTS Trigger Level is 4 bytes. 0010 = nRTS Trigger Level is 8 bytes. 0011 = nRTS Trigger Level is 14 bytes. Others = Reserved. **Note:** This field is used for auto nRTS flow control.
[15:9]	**Reserved**	Reserved.
[8]	**RXOFF**	**Receiver Disable Bit** The receiver is disabled or not (set 1 to disable receiver). 0 = Receiver Enabled. 1 = Receiver Disabled. **Note:** This bit is used for RS-485 Normal Multi-drop mode. It should be programmed before RS485NMM (LPUART_ALTCTL[8]) is programmed.
[7:4]	**RFITL**	**RX FIFO Interrupt Trigger Level** When the number of bytes in the receive FIFO equals the RFITL, the RDAIF (LPUART_INTSTS[0]) will be set (if RDAIEN (LPUART_INTEN[0]) enabled, and an interrupt will be generated). 0000 = RX FIFO Interrupt Trigger Level is 1 byte. 0001 = RX FIFO Interrupt Trigger Level is 4 bytes. 0010 = RX FIFO Interrupt Trigger Level is 8 bytes. 0011 = RX FIFO Interrupt Trigger Level is 14 bytes. Others = Reserved.
[3]	**Reserved**	Reserved.
[2]	**TXRST**	**TX Field Software Reset** When TXRST (LPUART_FIFO[2]) is set, all the byte in the transmit FIFO and TX internal

		state machine are cleared. 0 = No effect. 1 = Reset the TX internal state machine and pointers. **Note 1:** This bit will automatically clear at least 3 LPUART peripheral clock cycles. **Note 2:** Before setting this bit, it should wait for the TXEMPTYF (LPUART_FIFOSTS[28]) be set.
[1]	**RXRST**	**RX Field Software Reset** When RXRST (LPUART_FIFO[1]) is set, all the byte in the receiver FIFO and RX internal state machine are cleared. 0 = No effect. 1 = Reset the RX internal state machine and pointers. **Note 1:** This bit will automatically clear at least 3 LPUART peripheral clock cycles. **Note 2:** Before setting this bit, it should wait for the RXIDLE (LPUART_FIFOSTS[29]) be set.
[0]	**Reserved**	Reserved.

LPUART Line Control Register (LPUART_LINE)

Register	Offset	R/W	Description	Reset Value
LPUART_LINE	LPUARTx_BA+0x0C	R/W	LPUART Line Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						RXDINV	TXDINV
7	6	5	4	3	2	1	0
PSS	BCB	SPE	EPE	PBE	NSB	WLS	

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**RXDINV**	**RX Data Inverted** 0 = Received data signal inverted Disabled. 1 = Received data signal inverted Enabled. **Note 1:** Before setting this bit, TXRXDIS (LPUART_FUNCSEL[3]) should be set then waited for TXRXACT (LPUART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (LPUART_FUNCSEL[3]) to activate LPUART controller. **Note 2:** This bit is valid when FUNCSEL (LPUART_FUNCSEL[2:0]) is select LPUART or RS485 function.
[8]	**TXDINV**	**TX Data Inverted** 0 = Transmitted data signal inverted Disabled. 1 = Transmitted data signal inverted Enabled. **Note 1:** Before setting this bit, TXRXDIS (LPUART_FUNCSEL[3]) should be set then waited for TXRXACT (LPUART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (LPUART_FUNCSEL[3]) to activate LPUART controller. **Note 2:** This bit is valid when FUNCSEL (LPUART_FUNCSEL[2:0]) is select LPUART or RS485 function.
[7]	**PSS**	**PARITY Bit Source Selection** The PARITY bit can be selected to be generated and checked automatically or by software. 0 = PARITY bit is generated by EPE (LPUART_LINE[4]) and SPE (LPUART_LINE[5]) setting and checked automatically. 1 = PARITY bit generated and checked by software. **Note 1:** This bit has effect only when PBE (LPUART_LINE[3]) is set. **Note 2:** If PSS is 0, the PARITY bit is transmitted and checked automatically. If PSS is 1, the transmitted PARITY bit value can be determined by writing PARITY (LPUART_DAT[8]) and the PARITY bit can be read by reading PARITY (LPUART_DAT[8]).
[6]	**BCB**	**Break Control Bit** 0 = Break Control Disabled.

		1 = Break Control Enabled. **Note:** When this bit is set to logic 1, the transmitted serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX line and has no effect on the transmitter logic.
[5]	**SPE**	**Stick Parity Enable Bit** 0 = Stick parity Disabled. 1 = Stick parity Enabled. **Note:** If PBE (LPUART_LINE[3]) and EPE (LPUART_LINE[4]) are logic 1, the PARITY bit is transmitted and checked as logic 0. If PBE (LPUART_LINE[3]) is 1 and EPE (LPUART_LINE[4]) is 0 then the PARITY bit is transmitted and checked as 1.
[4]	**EPE**	**Even Parity Enable Bit** 0 = Odd number of logic 1's is transmitted and checked in each word. 1 = Even number of logic 1's is transmitted and checked in each word. **Note:** This bit has effect only when PBE (LPUART_LINE[3]) is set.
[3]	**PBE**	**PARITY Bit Enable Bit** 0 = PARITY bit generated Disabled. 1 = PARITY bit generated Enabled. **Note:** PARITY bit is generated on each outgoing character and is checked on each incoming data.
[2]	**NSB**	**Number of "STOP Bit"** 0 = One "STOP bit" is generated in the transmitted data. 1 = When select 5-bit word length, 1.5 "STOP bit" is generated in the transmitted data. When select 6-, 7- and 8-bit word length, 2 "STOP bit" is generated in the transmitted data.
[1:0]	**WLS**	**Word Length Selection** This field sets LPUART word length. 00 = 5 bits. 01 = 6 bits. 10 = 7 bits. 11 = 8 bits.

LPUART Modem Control Register (LPUART_MODEM)

Register	Offset	R/W	Description	Reset Value
LPUART_MODEM	LPUARTx_BA+0x10	R/W	LPUART Modem Control Register	0x0000_0200

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		RTSSTS	Reserved			RTSACTLV	Reserved
7	6	5	4	3	2	1	0
Reserved						RTS	Reserved

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**RTSSTS**	**nRTS Pin Status (Read Only)** This bit mirror from nRTS pin output of voltage logic status. 0 = nRTS pin output is low level voltage logic state. 1 = nRTS pin output is high level voltage logic state.
[12:10]	**Reserved**	Reserved.
[9]	**RTSACTLV**	**nRTS Pin Active Level** This bit defines the active level state of nRTS pin output. 0 = nRTS pin output is high level active. 1 = nRTS pin output is low level active. (Default) **Note 1:** Refer to Figure 6.28-13 and Figure 6.28-14 for LPUART function mode. **Note 2:** Refer to Figure 6.28-15 and Figure 6.28-16 for RS-485 function mode. **Note 3:** Before setting this bit, TXRXDIS (LPUART_FUNCSEL[3]) should be set then waited for TXRXACT (LPUART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (LPUART_FUNCSEL[3]) to activate LPUART controller.
[8:2]	**Reserved**	Reserved.
[1]	**RTS**	**nRTS Signal Control** This bit is direct control internal nRTS (Request-to-send) signal active or not, and then drive the nRTS pin output with RTSACTLV bit configuration. 0 = nRTS signal is active. 1 = nRTS signal is inactive. **Note 1:** The nRTS signal control bit is not effective when nRTS auto-flow control is enabled in LPUART function mode. **Note 2:** The nRTS signal control bit is not effective when RS-485 auto direction mode (AUD) is enabled in RS-485 function mode. **Note 3:** Single-wire mode is support this feature.
[0]	**Reserved**	Reserved.

LPUART Modem Status Register (LPUART_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
LPUART_MODEMSTS	LPUARTx_BA+0x14	R/W	LPUART Modem Status Register	0x0000_0110

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							CTSACTLV
7	6	5	4	3	2	1	0
Reserved			CTSSTS	Reserved			CTSDETF

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CTSACTLV**	**nCTS Pin Active Level** This bit defines the active level state of nCTS pin input. 0 = nCTS pin input is high level active. 1 = nCTS pin input is low level active. (Default) **Note:** Before setting this bit, TXRXDIS (LPUART_FUNCSEL[3]) should be set then waited for TXRXACT (LPUART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (LPUART_FUNCSEL[3]) to activate LPUART controller.
[7:5]	**Reserved**	Reserved.
[4]	**CTSSTS**	**nCTS Pin Status (Read Only)** This bit mirror from nCTS pin input of voltage logic status. 0 = nCTS pin input is low level voltage logic state. 1 = nCTS pin input is high level voltage logic state. **Note:** This bit echoes when LPUART controller peripheral clock is enabled, and nCTS multi-function port is selected.
[3:1]	**Reserved**	Reserved.
[0]	**CTSDETF**	**Detect nCTS State Change Flag** This bit is set whenever nCTS input has change state, and it will generate Modem interrupt to CPU when MODEMIEN (LPUART_INTEN[3]) is set to 1. 0 = nCTS input has not change state. 1 = nCTS input has change state. **Note:** This bit can be cleared by writing “1” to it.

LPUART FIFO Status Register (LPUART_FIFOSTS)

Register	Offset	R/W	Description	Reset Value
LPUART_FIFOSTS	LPUARTx_BA+0x18	R/W	LPUART FIFO Status Register	0xB040_4000

31	30	29	28	27	26	25	24
TXRXACT	Reserved	RXIDLE	TXEMPTYF	Reserved			TXOVIF
23	22	21	20	19	18	17	16
TXFULL	TXEMPTY	TXPTR					
15	14	13	12	11	10	9	8
RXFULL	RXEMPTY	RXPTR					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	ADDRDETF	ABRDTOIF	ABRDIF	RXOVIF

Bits	Description	
[31]	**TXRXACT**	**TX and RX Active Status (Read Only)** This bit indicates TX and RX are active or inactive. 0 = TX and RX are inactive. 1 = TX and RX are active. (Default) **Note:** When TXRXDIS (LPUART_FUNCSEL[3]) is set and both TX and RX are in idle state, this bit is cleared. The LPUART controller cannot transmit or receive data at this moment. Otherwise this bit is set.
[30]	**Reserved**	Reserved.
[29]	**RXIDLE**	**RX Idle Status (Read Only)** This bit is set by hardware when RX is idle. 0 = RX is busy. 1 = RX is idle. (Default)
[28]	**TXEMPTYF**	**Transmitter Empty Flag (Read Only)** This bit is set by hardware when TX FIFO (LPUART_DAT) is empty and the STOP bit of the last byte has been transmitted. 0 = TX FIFO is not empty or the STOP bit of the last byte has been not transmitted. 1 = TX FIFO is empty and the STOP bit of the last byte has been transmitted. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[27:25]	**Reserved**	Reserved.
[24]	**TXOVIF**	**TX Overflow Error Interrupt Flag** If TX FIFO (LPUART_DAT) is full, an additional write to LPUART_DAT will cause this bit to logic 1. 0 = TX FIFO is not overflow. 1 = TX FIFO is overflow. **Note:** This bit can be cleared by writing “1” to it.
[23]	**TXFULL**	**Transmitter FIFO Full (Read Only)**

		This bit indicates TX FIFO full or not. 0 = TX FIFO is not full. 1 = TX FIFO is full. **Note:** This bit is set when the number of usage in TX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[22]	**TXEMPTY**	**Transmitter FIFO Empty (Read Only)** This bit indicates TX FIFO empty or not. 0 = TX FIFO is not empty. 1 = TX FIFO is empty. **Note:** When the last byte of TX FIFO has been transferred to Transmitter Shift Register, hardware sets this bit high. It will be cleared when writing data into LPUART_DAT (TX FIFO not empty).
[21:16]	**TXPTR**	**TX FIFO Pointer (Read Only)** This field indicates the TX FIFO Buffer Pointer. When CPU writes one byte into LPUART_DAT, TXPTR increases one. When one byte of TX FIFO is transferred to Transmitter Shift Register, TXPTR decreases one. The Maximum value shown in TXPTR is 15. When the using level of TX FIFO Buffer equal to 16, the TXFULL bit is set to 1 and TXPTR will show 0. As one byte of TX FIFO is transferred to Transmitter Shift Register, the TXFULL bit is cleared to 0 and TXPTR will show 15.
[15]	**RXFULL**	**Receiver FIFO Full (Read Only)** This bit initiates RX FIFO full or not. 0 = RX FIFO is not full. 1 = RX FIFO is full. **Note:** This bit is set when the number of usage in RX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[14]	**RXEMPTY**	**Receiver FIFO Empty (Read Only)** This bit initiate RX FIFO empty or not. 0 = RX FIFO is not empty. 1 = RX FIFO is empty. **Note:** When the last byte of RX FIFO has been read by CPU, hardware sets this bit high. It will be cleared when LPUART receives any new data.
[13:8]	**RXPTR**	**RX FIFO Pointer (Read Only)** This field indicates the RX FIFO Buffer Pointer. When LPUART receives one byte from external device, RXPTR increases one. When one byte of RX FIFO is read by CPU, RXPTR decreases one. The Maximum value shown in RXPTR is 15. When the using level of RX FIFO Buffer equal to 16, the RXFULL bit is set to 1 and RXPTR will show 0. As one byte of RX FIFO is read by CPU, the RXFULL bit is cleared to 0 and RXPTR will show 15.
[7]	**Reserved**	Reserved.
[6]	**BIF**	**Break Interrupt Flag** This bit is set to logic 1 whenever the received data input (RX) is held in the “spacing state” (logic 0) for longer than a full word transmission time (that is, the total time of “START bit” + data bits + parity + STOP bits). 0 = No Break interrupt is generated. 1 = Break interrupt is generated. **Note:** This bit can be cleared by writing “1” to it.
[5]	**FEF**	**Framing Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “STOP bit” (that is, the STOP bit following the last data bit or PARITY bit is detected as logic 0). 0 = No framing error is generated.

		1 = Framing error is generated. **Note:** This bit can be cleared by writing “1” to it.
[4]	**PEF**	**Parity Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “PARITY bit”. 0 = No parity error is generated. 1 = Parity error is generated. **Note:** This bit can be cleared by writing “1” to it.
[3]	**ADDRDETF**	**RS-485 Address Byte Detect Flag** 0 = Receiver detects a data that is not an address bit (bit 9 =’0’). 1 = Receiver detects a data that is an address bit (bit 9 =’1’). **Note 1:** This field is used for RS-485 function mode and ADDRDEN (LPUART_ALTCTL[15]) is set to 1 to enable Address detection mode. **Note 2:** This bit can be cleared by writing “1” to it.
[2]	**ABRDTOIF**	**Auto-baud Rate Detect Time-out Interrupt Flag** This bit is set to logic “1” in Auto-baud Rate Detect mode when the baud rate counter is overflow. 0 = Auto-baud rate counter is underflow. 1 = Auto-baud rate counter is overflow. **Note:** This bit can be cleared by writing “1” to it.
[1]	**ABRDIF**	**Auto-baud Rate Detect Interrupt Flag** This bit is set to logic “1” when auto-baud rate detect function is finished. 0 = Auto-baud rate detect function is not finished. 1 = Auto-baud rate detect function is finished. **Note:** This bit can be cleared by writing “1” to it.
[0]	**RXOVIF**	**RX Overflow Error Interrupt Flag** This bit is set when RX FIFO overflow. If the number of bytes of received data is greater than RX_FIFO (LPUART_DAT) size 16 bytes, this bit will be set. 0 = RX FIFO is not overflow. 1 = RX FIFO is overflow. **Note:** This bit can be cleared by writing “1” to it.

LPUART Interrupt Status Register (LPUART_INTSTS)

Register	Offset	R/W	Description	Reset Value
LPUART_INTSTS	LPUARTx_BA+0x1C	R/W	LPUART Interrupt Status Register	0x0040_0002

31	30	29	28	27	26	25	24
ABRINT	TXENDINT	HWBUFEINT	HWTOINT	HWMODINT	HWRLSINT	Reserved	SWBEINT
23	22	21	20	19	18	17	16
Reserved	TXENDIF	HWBUFEIF	HWTOIF	HWMODIF	HWRLSIF	Reserved	SWBEIF
15	14	13	12	11	10	9	8
Reserved	WKINT	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THREINT	RDAINT
7	6	5	4	3	2	1	0
Reserved	WKIF	BUFERRIF	RXTOIF	MODEMIF	RLSIF	THREIF	RDAIF

Bits	Description	
[31]	**ABRINT**	**Auto-baud Rate Interrupt Indicator (Read Only)** This bit is set if ABRIEN (LPUART_INTEN[18]) and ABRIF (LPUART_ALTCTL[17]) are both set to 1. 0 = No Auto-baud Rate interrupt is generated. 1 = The Auto-baud Rate interrupt is generated.
[30]	**TXENDINT**	**Transmitter Empty Interrupt Indicator (Read Only)** This bit is set if TXENDIEN (LPUART_INTEN[22]) and TXENDIF(LPUART_INTSTS[22]) are both set to 1. 0 = No Transmitter Empty interrupt is generated. 1 = Transmitter Empty interrupt is generated.
[29]	**HWBUFEINT**	**LPPDMA Mode Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN (LPUART_INTEN[5]) and HWBUFEIF (LPUART_INTSTS[21]) are both set to 1. 0 = No buffer error interrupt is generated in LPPDMA mode. 1 = Buffer error interrupt is generated in LPPDMA mode.
[28]	**HWTOINT**	**LPPDMA Mode RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (LPUART_INTEN[4]) and HWTOIF(LPUART_INTSTS[20]) are both set to 1. 0 = No RX time-out interrupt is generated in LPPDMA mode. 1 = RX time-out interrupt is generated in LPPDMA mode.
[27]	**HWMODINT**	**LPPDMA Mode MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN (LPUART_INTEN[3]) and HWMODIF(LPUART_INTSTS[19]) are both set to 1. 0 = No Modem interrupt is generated in LPPDMA mode. 1 = Modem interrupt is generated in LPPDMA mode.
[26]	**HWRLSINT**	**LPPDMA Mode Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (LPUART_INTEN[2]) and HWRLSIF(LPUART_INTSTS[18]) are both set to 1.

		0 = No RLS interrupt is generated in LPPDMA mode. 1 = RLS interrupt is generated in LPPDMA mode.
[25]	**Reserved**	Reserved.
[24]	**SWBEINT**	**Single-wire Bit Error Detect Interrupt Indicator (Read Only)** This bit is set if SWBEIEN (LPUART_INTEN[16]) and SWBEIF (LPUART_INTSTS[16]) are both set to 1. 0 = No Single-wire Bit Error Detection Interrupt generated. 1 = Single-wire Bit Error Detection Interrupt generated.
[23]	**Reserved**	Reserved.
[22]	**TXENDIF**	**Transmitter Empty Interrupt Flag** This bit is set when TX FIFO (LPUART_DAT) is empty and the STOP bit of the last byte has been transmitted (TXEMPTYF (LPUART_FIFOSTS[28]) is set). If TXENDIEN (LPUART_INTEN[22]) is enabled, the Transmitter Empty interrupt will be generated. 0 = No transmitter empty interrupt flag is generated. 1 = Transmitter empty interrupt flag is generated. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[21]	**HWBUFEIF**	**LPPDMA Mode Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX or RX FIFO overflows (TXOVIF (LPUART_FIFOSTS[24]) or RXOVIF (LPUART_FIFOSTS[0]) is set). When BUFERRIF (LPUART_INTSTS[5]) is set, the transfer maybe is not correct. If BUFERRIEN (LPUART_INTEN[5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated in LPPDMA mode. 1 = Buffer error interrupt flag is generated in LPPDMA mode. **Note:** This bit is cleared when both TXOVIF (LPUART_FIFOSTS[24]]) and RXOVIF (LPUART_FIFOSTS[0]) are cleared.
[20]	**HWTOIF**	**LPPDMA Mode RX Time-out Interrupt Flag** This bit is set when the RX FIFO is not empty and no activities occurred in the RX FIFO and the time-out counter equal to TOIC (LPUART_TOUT[7:0]). If RXTOIEN (LPUART_INTEN[4]) is enabled, the RX time-out interrupt will be generated. 0 = No RX time-out interrupt flag is generated in LPPDMA mode. 1 = RX time-out interrupt flag is generated in LPPDMA mode. **Note:** This bit can be cleared by writing '1' to it. When BITOMEN (LPUART_TOUT[31]) is reset, this bit can also be cleared by reading LPUART_DAT.
[19]	**HWMODIF**	**LPPDMA Mode MODEM Interrupt Flag (Read Only)** This bit is set when the nCTS pin has state change (CTSDETF (LPUART_MODEMSTS[0] =1)). If MODEMIEN (LPUART_INTEN[3]) is enabled, the Modem interrupt will be generated. 0 = No Modem interrupt flag is generated in LPPDMA mode. 1 = Modem interrupt flag is generated in LPPDMA mode. **Note:** This bit is read only and reset to 0 when the bit CTSDETF (LPUART_MODEMSTS[0]) is cleared by writing 1 on CTSDETF (LPUART_MODEMSTS[0]).
[18]	**HWRLSIF**	**LPPDMA Mode Receive Line Status Flag (Read Only)** This bit is set when the RX receive data have parity error, frame error or break error (at least one of 3 bits, BIF (LPUART_FIFOSTS[6]), FEF (LPUART_FIFOSTS[5]) and PEF (LPUART_FIFOSTS[4]) is set). If RLSIEN (LPUART_INTEN[2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated in LPPDMA mode. 1 = RLS interrupt flag is generated in LPPDMA mode. **Note 1:** In RS-485 function mode, this field include "receiver detect any address byte received address byte character (bit9 = '1') bit".

		Note 2: In LPUART function mode, this bit is read only and reset to 0 when all bits of BIF(LPUART_FIFOSTS[6]), FEF(LPUART_FIFOSTS[5]) and PEF(LPUART_FIFOSTS[4]) are cleared. **Note 3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF(LPUART_FIFOSTS[6]), FEF(LPUART_FIFOSTS[5]), PEF(LPUART_FIFOSTS[4]) and ADDRDETF (LPUART_FIFOSTS[3]) are cleared.
[17]	**Reserved**	Reserved.
[16]	**SWBEIF**	**Single-wire Bit Error Detection Interrupt Flag** This bit is set when the single wire bus state not equals to LPUART controller TX state in Single-wire mode. 0 = No single-wire bit error detection interrupt flag is generated. 1 = Single-wire bit error detection interrupt flag is generated. **Note 1:** This bit is active when FUNCSEL (LPUART_FUNCSEL[2:0]) is select LPUART Single-wire mode. **Note 2:** This bit can be cleared by writing “1” to it.
[15]	**Reserved**	Reserved.
[14]	**WKINT**	**LPUART Wake-up Interrupt Indicator (Read Only)** This bit is set if WKIEN (LPUART_INTEN[6]) and WKIF (LPUART_INTSTS[6]) are both set to 1. 0 = No LPUART wake-up interrupt is generated. 1 = LPUART wake-up interrupt is generated.
[13]	**BUFERRINT**	**Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN(LPUART_INTEN[5]) and BUFERRIF(LPUART_ INTSTS[5]) are both set to 1. 0 = No buffer error interrupt is generated. 1 = Buffer error interrupt is generated.
[12]	**RXTOINT**	**RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (LPUART_INTEN[4]) and RXTOIF(LPUART_INTSTS[4]) are both set to 1. 0 = No RX time-out interrupt is generated. 1 = RX time-out interrupt is generated.
[11]	**MODEMINT**	**MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN(LPUART_INTEN[3]) and MODEMIF(LPUART_INTSTS[3]) are both set to 1 0 = No Modem interrupt is generated. 1 = Modem interrupt is generated..
[10]	**RLSINT**	**Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (LPUART_INTEN[2]) and RLSIF(LPUART_INTSTS[2]) are both set to 1. 0 = No RLS interrupt is generated. 1 = RLS interrupt is generated.
[9]	**THREINT**	**Transmit Holding Register Empty Interrupt Indicator (Read Only)** This bit is set if THREIEN (LPUART_INTEN[1]) and THREIF(LPUART_INTSTS[1]) are both set to 1. 0 = No THRE interrupt is generated. 1 = THRE interrupt is generated.
[8]	**RDAINT**	**Receive Data Available Interrupt Indicator (Read Only)** This bit is set if RDAIEN (LPUART_INTEN[0]) and RDAIF (LPUART_INTSTS[0]) are both set to 1.

		0 = No RDA interrupt is generated. 1 = RDA interrupt is generated.
[7]	**Reserved**	Reserved.
[6]	**WKIF**	**LPUART Wake-up Interrupt Flag (Read Only)** This bit is set when TOUTWKF (LPUART_WKSTS[4]), RS485WKF (LPUART_WKSTS[3]), RFRTWKF (LPUART_WKSTS[2]), DATWKF (LPUART_WKSTS[1]) or CTSWKF(LPUART_WKSTS[0]) is set to 1. 0 = No LPUART wake-up interrupt flag is generated. 1 = LPUART wake-up interrupt flag is generated. **Note:** This bit is cleared if all of TOUTWKF, RS485WKF, RFRTWKF, DATWKF and CTSWKF are cleared to 0 by writing 1 to the corresponding interrupt flag.
[5]	**BUFERRIF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX FIFO or RX FIFO overflows (TXOVIF (LPUART_FIFOSTS[24]) or RXOVIF (LPUART_FIFOSTS[0]) is set). When BUFERRIF (LPUART_INTSTS[5]) is set, the transfer is not correct. If BUFERRIEN (LPUART_INTEN[5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated. 1 = Buffer error interrupt flag is generated. **Note:** This bit is cleared if both of RXOVIF(LPUART_FIFOSTS[0]) and TXOVIF(LPUART_FIFOSTS[24]) are cleared to 0 by writing 1 to RXOVIF(LPUART_FIFOSTS[0]) and TXOVIF(LPUART_FIFOSTS[24]).
[4]	**RXTOIF**	**RX Time-out Interrupt Flag** This bit is set when the RX FIFO is not empty and no activities occurred in the RX FIFO and the time-out counter equal to TOIC (LPUART_TOUT[7:0]). If RXTOIEN (LPUART_INTEN[4]) is enabled, the RX time-out interrupt will be generated. 0 = No RX time-out interrupt flag is generated. 1 = RX time-out interrupt flag is generated. **Note:** This bit can be cleared by writing '1' to it. When BITOMEN (LPUART_TOUT[31]) is reset, this bit can also be cleared by reading LPUART_DAT.
[3]	**MODEMIF**	**MODEM Interrupt Flag (Read Only)** This bit is set when the nCTS pin has state change (CTSDETF (LPUART_MODEMSTS[0]) = 1). If MODEMIEN (LPUART_INTEN[3]) is enabled, the Modem interrupt will be generated. 0 = No Modem interrupt flag is generated. 1 = Modem interrupt flag is generated. **Note:** This bit is read only and reset to 0 when bit CTSDETF is cleared by a write 1 on CTSDETF(LPUART_MODEMSTS[0]).
[2]	**RLSIF**	**Receive Line Interrupt Flag (Read Only)** This bit is set when the RX receive data have parity error, frame error or break error (at least one of 3 bits, BIF(LPUART_FIFOSTS[6]), FEF(LPUART_FIFOSTS[5]) and PEF(LPUART_FIFOSTS[4]), is set). If RLSIEN (LPUART_INTEN[2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated. 1 = RLS interrupt flag is generated. **Note 1:** In RS-485 function mode, this field is set include "receiver detect and received address byte character (bit9 = '1') bit". At the same time, the bit of ADDRDETF (LPUART_FIFOSTS[3]) is also set. **Note 2:** This bit is read only and reset to 0 when all bits of BIF (LPUART_FIFOSTS[6]), FEF(LPUART_FIFOSTS[5]) and PEF(LPUART_FIFOSTS[4]) are cleared. **Note 3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF (LPUART_FIFOSTS[6]), FEF(LPUART_FIFOSTS[5]), PEF(LPUART_FIFOSTS[4]) and ADDRDETF (LPUART_FIFOSTS[3]) are cleared.
[1]	**THREIF**	**Transmit Holding Register Empty Interrupt Flag (Read Only)**

		This bit is set when the last data of TX FIFO is transferred to Transmitter Shift Register. If THREIEN (LPUART_INTEN[1]) is enabled, the THRE interrupt will be generated. 0 = No THRE interrupt flag is generated. 1 = THRE interrupt flag is generated. **Note:** This bit is read only and will be cleared when writing data into LPUART_DAT (TX FIFO is not empty).
[0]	**RDAIF**	**Receive Data Available Interrupt Flag (Read Only)** When the number of bytes in the RX FIFO equals the RFITL then the RDAIF(LPUART_INTSTS[0]) will be set. If RDAIEN (LPUART_INTEN[0]) is enabled, the RDA interrupt will be generated. 0 = No RDA interrupt flag is generated. 1 = RDA interrupt flag is generated. **Note:** This bit is read only and will be cleared when the number of unread bytes of RX FIFO drops below the threshold level (RFITL(LPUART_FIFO[7:4]).

LPUART Time-out Register (LPUART_TOUT)

Register	Offset	R/W	Description	Reset Value
LPUART_TOUT	LPUARTx_BA+0x20	R/W	LPUART Time-out Register	0x0000_0000

31	30	29	28	27	26	25	24
BITOMEN	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DLY							
7	6	5	4	3	2	1	0
TOIC							

Bits	Description	
[31]	**BITOMEN**	**Bus Idle Time-out Mode Enable Bit** If BITOMEN (LPUART_TOUT[31]) is enabled, the reset conditions of the time-out counter and RXTOIF (LPUART_INTSTS[4]) will be changed to detect the bus idle. When BITOMEN (LPUART_TOUT[31]) is disabled, the time-out counter and RXTOIF (LPUART_INTSTS[4]) maintain reset value whenever the RX FIFO is empty. In addition, a new incoming data word will also clear RXTOIF (LPUART_INTSTS[4]). On the other hand, when BITOMEN (LPUART_TOUT[31]) is enabled, the RX FIFO empty state will not reset the time-out counter and RXTOIF (LPUART_INTSTS[4]), and the new incoming data word event will not clear RXTOIF (LPUART_INTSTS[4]). 0 = Bus idle time-out mode Disabled. 1 = Bus idle time-out mode Enabled.
[30:16]	**Reserved**	Reserved.
[15:8]	**DLY**	**TX Delay Time Value** This field is used to program the transfer delay time between the last STOP bit and next START bit. The unit is bit time.
[7:0]	**TOIC**	**Time-out Interrupt Comparator** The time-out counter resets and starts counting (the counting clock = baud rate) whenever the RX FIFO receives a new data word if time out counter is enabled by setting TOCNTEN (LPUART_INTEN[11]). Once the content of time-out counter is equal to that of time-out interrupt comparator (TOIC (LPUART_TOUT[7:0])), a receiver time-out interrupt (RXTOINT(LPUART_INTSTS[12])) is generated if RXTOIEN (LPUART_INTEN[4]) enabled. A new incoming data word or RX FIFO empty will clear RXTOIF (LPUART_INTSTS[4]). In order to avoid receiver time-out interrupt generation immediately during one character is being received, TOIC value should be set between 40 and 255. So, for example, if TOIC is set with 40, the time-out interrupt is generated after four characters are not received when 1 STOP bit and no parity check is set for LPUART transfer.

LPUART Baud Rate Divider Register (LPUART_BAUD)

Register	Offset	R/W	Description	Reset Value
LPUART_BAUD	LPUARTx_BA+0x24	R/W	LPUART Baud Rate Divider Register	0x0F00_0000

31	30	29	28	27	26	25	24
Reserved		BAUDM1	BAUDM0	EDIVM1			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BRD							
7	6	5	4	3	2	1	0
BRD							

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**BAUDM1**	**BAUD Rate Mode Selection Bit 1** This bit is baud rate mode selection bit 1. LPUART provides three baud rate calculation modes. This bit combines with BAUDM0 (LPUART_BAUD[28]) to select baud rate calculation mode. The detail description is shown in Table 6.28-4.
[28]	**BAUDM0**	**BAUD Rate Mode Selection Bit 0** This bit is baud rate mode selection bit 0. LPUART provides three baud rate calculation modes. This bit combines with BAUDM1 (LPUART_BAUD[29]) to select baud rate calculation mode. The detail description is shown in Table 6.28-4.
[27:24]	**EDIVM1**	**Extra Divider for BAUD Rate Mode 1** This field is used for baud rate calculation in mode 1 and has no effect for baud rate calculation in mode 0 and mode 2. The detail description is shown in Table 6.28-4.
[23:16]	**Reserved**	Reserved.
[15:0]	**BRD**	**Baud Rate Divider** The field indicates the baud rate divider. This filed is used in baud rate calculation. The detail description is shown in Table 6.28-4.

LPUART Alternate Control/Status Register (LPUART_ALTCTL)

Register	Offset	R/W	Description	Reset Value
LPUART_ALTCTL	LPUARTx_BA+0x2C	R/W	LPUART Alternate Control/Status Register	0x0000_000C

31	30	29	28	27	26	25	24
ADDRMV							
23	22	21	20	19	18	17	16
Reserved			ABRDBITS		ABRDEN	ABRIF	Reserved
15	14	13	12	11	10	9	8
ADDRDEN	Reserved				RS485AUD	RS485AAD	RS485NMM
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:24]	**ADDRMV**	**Address Match Value** This field contains the RS-485 address match values. **Note:** This field is used for RS-485 auto address detection mode.
[23:21]	**Reserved**	Reserved.
[20:19]	**ABRDBITS**	**Auto-baud Rate Detect Bit Length** 00 = 1-bit time from START bit to the 1st rising edge. The input pattern shall be 0x01. 01 = 2-bit time from START bit to the 1st rising edge. The input pattern shall be 0x02. 10 = 4-bit time from START bit to the 1st rising edge. The input pattern shall be 0x08. 11 = 8-bit time from START bit to the 1st rising edge. The input pattern shall be 0x80. **Note :** The calculation of bit number includes the START bit.
[18]	**ABRDEN**	**Auto-baud Rate Detect Enable Bit** 0 = Auto-baud rate detect function Disabled. 1 = Auto-baud rate detect function Enabled. **Note :** This bit is cleared automatically after auto-baud detection is finished.
[17]	**ABRIF**	**Auto-baud Rate Interrupt Flag (Read Only)** This bit is set when auto-baud rate detection function finished or the auto-baud rate counter was overflow and if ABRIEN(LPUART_INTEN[18]) is set then the auto-baud rate interrupt will be generated. 0 = No auto-baud rate interrupt flag is generated. 1 = Auto-baud rate interrupt flag is generated. **Note:** This bit is read only, but it can be cleared by writing “1” to ABRDTOIF (LPUART_FIFOSTS[2]) and ABRDIF(LPUART_FIFOSTS[1]).
[16]	**Reserved**	Reserved.
[15]	**ADDRDEN**	**RS-485 Address Detection Enable Bit** This bit is used to enable RS-485 Address Detection mode. 0 = Address detection mode Disabled. 1 = Address detection mode Enabled.

		Note: This bit is used for RS-485 any operation mode.
[14:11]	**Reserved**	Reserved.
[10]	**RS485AUD**	**RS-485 Auto Direction Function** 0 = RS-485 Auto Direction Operation function (AUD) Disabled. 1 = RS-485 Auto Direction Operation function (AUD) Enabled. **Note:** It can be active with RS-485_AAD or RS-485_NMM operation mode.
[9]	**RS485AAD**	**RS-485 Auto Address Detection Operation Mode** 0 = RS-485 Auto Address Detection Operation mode (AAD) Disabled. 1 = RS-485 Auto Address Detection Operation mode (AAD) Enabled. **Note:** It cannot be active with RS-485_NMM operation mode.
[8]	**RS485NMM**	**RS-485 Normal Multi-drop Operation Mode** 0 = RS-485 Normal Multi-drop Operation mode (NMM) Disabled. 1 = RS-485 Normal Multi-drop Operation mode (NMM) Enabled. **Note:** It cannot be active with RS-485_AAD operation mode.
[7:0]	**Reserved**	Reserved.

LPUART Function Select Register (LPUART_FUNCSEL)

Register	Offset	R/W	Description	Reset Value
LPUART_FUNCSEL	LPUARTx_BA+0x30	R/W	LPUART Function Select Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
TXRXSWP	DGE	Reserved		TXRXDIS	FUNCSEL		

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**TXRXSWP**	**TX and RX Swap Enable Bit** Setting this bit Swaps TX pin and RX pin. 0 = TX and RX Swap Disabled. 1 = TX and RX Swap Enabled.
[6]	**DGE**	**Deglitch Enable Bit** 0 = Deglitch Disabled. 1 = Deglitch Enabled. **Note 1:** When this bit is set to logic 1, any pulse width less than about 150 ns will be considered a glitch and will be removed in the serial data input (RX). This bit acts only on RX line and has no effect on the transmitter logic. **Note 2:** It is recommended to set this bit only when operating at baud rate under 2.5 Mbps.
[5:4]	**Reserved**	Reserved.
[3]	**TXRXDIS**	**TX and RX Disable Bit** Setting this bit can disable TX and RX. 0 = TX and RX Enabled. 1 = TX and RX Disabled. **Note:** The TX and RX will not be disabled immediately when this bit is set. The TX and RX complete current task before TX and RX are disabled. When TX and RX are disabled, the TXRXACT (LPUART_FIFOSTS[31]) is cleared.
[2:0]	**FUNCSEL**	**Function Select** 000 = LPUART function. 011 = RS-485 function. 100 = LPUART Single-wire function. Others = Reserved.

LPUART Baud Rate Compensation Register (LPUART_BRCOMP)

Register	Offset	R/W	Description	Reset Value
LPUART_BRCOMP	LPUARTx_BA+0x3C	R/W	LPUART Baud Rate Compensation Register	0x0000_0000

31	30	29	28	27	26	25	24
BRCOMPDEC	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BRCOMP
7	6	5	4	3	2	1	0
BRCOMP							

Bits	Description	
[31]	**BRCOMPDEC**	**Baud Rate Compensation Decrease** 0 = Positive (increase one module clock) compensation for each compensated bit. 1 = Negative (decrease one module clock) compensation for each compensated bit.
[30:9]	**Reserved**	Reserved.
[8:0]	**BRCOMP**	**Baud Rate Compensation Patten** These 9-bits are used to define the relative bit is compensated or not. BRCOMP[7:0] is used to define the compensation of DAT (LPUART_DAT[7:0]) and BRCOM[8] is used to define PARITY (LPUART_DAT[8]).

LPUART Wake-up Control Register (LPUART_WKCTL)

Register	Offset	R/W	Description	Reset Value
LPUART_WKCTL	LPUARTx_BA+0x40	R/W	LPUART Wake-up Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			WKTOUTEN	WKRS485EN	WKRFRTEN	WKDATEN	WKCTSEN

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**WKTOUTEN**	**Received Data FIFO Reached Threshold Time-out Wake-up Enable Bit** 0 = Received Data FIFO reached threshold time-out wake-up system function Disabled. 1 = Received Data FIFO reached threshold time-out wake-up system function Enabled. **Note 1:** When the system is in Power-down mode, Received Data FIFO reached threshold time-out will wake up system from Power-down mode. **Note 2:** It is suggested the function is enabled when the WKRFRTEN (LPUART_WKCTL[2]) is set to 1.
[3]	**WKRS485EN**	**RS-485 Address Match Wake-up Enable Bit** 0 = RS-485 Address Match (AAD mode) wake-up system function Disabled. 1 = RS-485 Address Match (AAD mode) wake-up system function Enabled. **Note 1:** When the system is in Power-down mode, RS-485 Address Match will wake-up system from Power-down mode. **Note 2:** This bit is used for RS-485 Auto Address Detection (AAD) mode in RS-485 function mode and ADDRDEN (LPUART_ALTCTL[15]) is set to 1.
[2]	**WKRFRTEN**	**Received Data FIFO Reached Threshold Wake-up Enable Bit** 0 = Received Data FIFO reached threshold wake-up system function Disabled. 1 = Received Data FIFO reached threshold wake-up system function Enabled. **Note:** When the system is in Power-down mode, Received Data FIFO reached threshold will wake-up system from Power-down mode.
[1]	**WKDATEN**	**Incoming Data Wake-up Enable Bit** 0 = Incoming data wake-up system function Disabled. 1 = Incoming data wake-up system function Enabled. **Note:** When the system is in Power-down mode, incoming data will wake-up system from Power-down mode.
[0]	**WKCTSEN**	**nCTS Wake-up Enable Bit** 0 = nCTS Wake-up system function Disabled. 1 = nCTS Wake-up system function Enabled.

		Note: When the system is in Power-down mode, an external nCTS change will wake up system from Power-down mode.

LPUART Wake-up Status Register (LPUART_WKSTS)

Register	Offset	R/W	Description	Reset Value
LPUART_WKSTS	LPUARTx_BA+0x44	R/W	LPUART Wake-up Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			TOUTWKF	RS485WKF	RFRTWKF	DATWKF	CTSWKF

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**TOUTWKF**	**Received Data FIFO Threshold Time-out Wake-up Flag** This bit is set if chip wake-up from power-down state by Received Data FIFO Threshold Time-out wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Received Data FIFO reached threshold time-out. **Note 1:** If WKTOUTEN (LPUART_WKCTL[4]) is enabled, the Received Data FIFO reached threshold time-out wake-up cause this bit set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.
[3]	**RS485WKF**	**RS-485 Address Match Wake-up Flag** This bit is set if chip wake-up from power-down state by RS-485 Address Match (AAD mode). 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by RS-485 Address Match (AAD mode) wake-up. **Note 1:** If WKRS485EN (LPUART_WKCTL[3]) is enabled, the RS-485 Address Match (AAD mode) wake-up cause this bit set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.
[2]	**RFRTWKF**	**Received Data FIFO Reached Threshold Wake-up Flag** This bit is set if chip wake-up from power-down state by Received Data FIFO reached threshold wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Received Data FIFO Reached Threshold wake-up. **Note 1:** If WKRFRTEN (LPUART_WKCTL[2]) is enabled, the Received Data FIFO Reached Threshold wake-up cause this bit set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.
[1]	**DATWKF**	**Incoming Data Wake-up Flag**

		This bit is set if chip wake-up from power-down state by data wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Incoming Data wake-up. **Note 1:** If WKDATEN (LPUART_WKCTL[1]) is enabled, the Incoming Data wake-up cause this bit set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.
[0]	**CTSWKF**	**nCTS Wake-up Flag** This bit is set if chip wake-up from power-down state by nCTS wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by nCTS wake-up. **Note 1:** If WKCTSEN (LPUART_WKCTL[0]) is enabled, the nCTS wake-up cause this bit set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.

LPUART Incoming Data Wake-up Compensation Register (LPUART_DWKCOMP)

Register	Offset	R/W	Description	Reset Value
LPUART_DWKCOMP	LPUARTx_BA+0x48	R/W	LPUART Incoming Data Wake-up Compensation Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
STCOMP							
7	6	5	4	3	2	1	0
STCOMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**STCOMP**	**START Bit Compensation Value** These bits field indicate how many clock cycle selected by LPUART_CLK do the LPUART controller can get the 1st bit (START bit) when the device is wake-up from Power-down mode. **Note:** It is valid only when WKDATEN (LPUART_WKCTL[1]) is set.

LPUART RS485 Transceiver Deactivate Delay Register (LPUART_RS485DD)

Register	Offset	R/W	Description	Reset Value
LPUART_RS485DD	LPUARTx_BA+0x4C	R/W	LPUART RS485 Transceiver Deactivate Delay Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RTSDDLY							
7	6	5	4	3	2	1	0
RTSDDLY							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RTSDDLY**	**RS485 Transceiver Deactivate Delay Value** These bits field indicate how many clock cycles selected by LPUART_CLK do the LPUART controller delay the RS485 transceiver state trancing when the state trancing of RS485 transceiver is from TX to RX state. These bits field have no effect when the state trancing of RS485 transceiver is from RX to TX state. **Note:** It is valid only when RS485AUD (LPUART_ALTCTL[10]) is set.

LPUART Automatic Operation Control Register (LPUART_AUTOCTL)

Register	Offset	R/W	Description	Reset Value
LPUART_AUTOCTL	LPUARTx_BA+0x58	R/W	LPUART Automatic Operation Control Register	0x0000_0000

31	30	29	28	27	26	25	24
AOEN	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						SWTRIG	WKAOTOEN
7	6	5	4	3	2	1	0
Reserved		CKAWOEN	TRIGEN	TRIGSEL			

Bits	Description	
[31]	**AOEN**	**Automatic Operation Enable Bit** 0 = Automatic Operation Disabled. 1 = Automatic Operation Enabled.
[30:10]	**Reserved**	Reserved.
[9]	**SWTRIG**	**Software Trigger (Write Only)** After AOEN (LPUART_AUTOCTL[31]) is set to 1, software can manually trigger the automatic TX operation by writing 1 to this bit.
[8]	**WKAOTOEN**	**Bus Idle Time-out Wake-up Enable Bit** 0 = Bus Idle Time-out Wake-up system function Disabled. 1 = Bus Idle Time-out Wake-up system function Enabled. **Note 1:** When the system is in Power-down mode, Bus Idle Time-out Wake-up will wake-up system from Power-down mode if WKAOTOEN (LPUART_AUTOCTL[8]) is set to '1'. **Note 2:** When WKAOTOEN (LPUART_AUTOCTL[8]) is reset to '0', Bus Idle Time-out Wake-up event will not wake up the system. Moreover, Bus Idle Time-out Wake-up event will gate the LPUART clock off if WKAOTOEN (LPUART_AUTOCTL[8]) and AOCKAWOEN (LPUART_AUTOCTL[4]) are both reset to '0'.
[7:6]	**Reserved**	Reserved.
[5]	**CKAWOEN**	**Automatic Operation Clock Always-on Enable Bit** 0 = Automatic Operation Clock Always-on Disabled. 1 = Automatic Operation Clock Always-on Enabled.
[4]	**TRIGEN**	**Automatic Operation Trigger Enable Bit** When TRIGEN (LPUART_AUTOCTL[4]) is set to '1', the automatic TX operation of LPUART will be triggered by an event sent from the trigger source selected by TRIGSEL[3:0]. 0 = LPUART Automatic Operation Trigger disabled. 1 = LPUART Automatic Operation Trigger enabled.
[3:0]	**TRIGSEL**	**Automatic Operation Trigger Source Select**

		0000 = Low Power Auto-operation Trigger Source from LPTMR0. 0001 = Low Power Auto-operation Trigger Source from LPTMR1. 0010 = Low Power Auto-operation Trigger Source from TTMR0. 0011 = Low Power Auto-operation Trigger Source from TTMR1. 0100 = Low Power Auto-operation Trigger Source from WKIOA0. 0101 = Low Power Auto-operation Trigger Source from WKIOB0. 0110 = Low Power Auto-operation Trigger Source from WKIOC0. 0111 = Low Power Auto-operation Trigger Source from WKIOD0. 1000~1111 = Reserved.

LPUART Automatic Operation Status Register (LPUART_AUTOSTS)

Register	Offset	R/W	Description	Reset Value
LPUART_AUTOSTS	LPUARTx_BA+0x5C	R/W	LPUART Automatic Operation Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							AOTOWKF

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**AOTOWKF**	**Bus Idle Time-out Wake-up Flag** This bit is set if chip wake-up from power-down state by Bus Idle Time-out Wake-up 0 = There is no Bus Idle Time-out Wake-up event. 1 = Chip wake-up from power-down state by Bus Idle Time-out Wake-up. **Note 1:** When RXPDMAEN (LPUART_INTEN[15]) or TXPDMAEN (LPUART_INTEN[14]) is enabled, this bit is the shadow bit of HWTOIF (LPUART_INTSTS[20]); user can read/clear it by reading/resetting HWTOIF (LPUART_INTSTS[20]) or AOTOWKF (LPUART_AUTOSTS[0]). **Note 2:** When RXPDMAEN (LPUART_INTEN[15]) and TXPDMAEN (LPUART_INTEN[14]) are both disabled, this bit is the shadow bit of RXTOIF (LPUART_INTSTS[4]); user can read/clear it by reading/resetting RXTOIF (LPUART_INTSTS[4]) or AOTOWKF (LPUART_AUTOSTS[0]). **Note 3:** If AOEN (LPUART_AUTOCTL[31]) and WKAOTOEN (LPUART_AUTOCTL[8]) are enabled, the Bus Idle Time-out Wake-up event will set this bit. **Note 4:** This bit can be cleared by writing '1' to it.

6.29 Serial Peripheral Interface (SPI)

6.29.1 Overview

The Serial Peripheral Interface (SPI) applies to synchronous serial data communication and allows full duplex transfer. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The chip contains up to four sets of SPI controllers performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. Each SPI controller can be configured as a master or a slave device and supports the PDMA function to access the data buffer. Each SPI controller also supports I^2S mode to connect external audio CODEC.

6.29.2 Features

- SPI Mode
  - Up to four sets of SPI controllers
  - Supports Master or Slave mode operation
  - Master mode up to 72 MHz (when chip works at V_{DD} = 2.7~3.6V)
  - Slave mode up to 36 MHz (when chip works at V_{DD} = 2.7~3.6V)
  - Configurable bit length of a transaction word from 4 to 32-bit
  - Provides separate 4-level depth transmit and receive FIFO buffers
  - Supports MSB first or LSB first transfer sequence
  - Supports Byte Reorder function
  - Supports Byte or Word Suspend mode
  - Supports PDMA transfer
  - Supports 3-Wire, no slave selection signal, bi-direction interface
  - Supports one data channel half-duplex transfer
  - Supports receive-only mode
- I^2S Mode
  - Supports Master or Slave
  - Capable of handling 8-, 16-, 24- and 32-bit word sizes
  - Each provides two 4-level FIFO data buffers, one for transmitting and the other for receiving
  - Supports monaural and stereo audio data
  - Supports PCM mode A, PCM mode B, I^2S and MSB justified data format
  - Supports two PDMA requests, one for transmitting and the other for receiving

6.29.3 Block Diagram

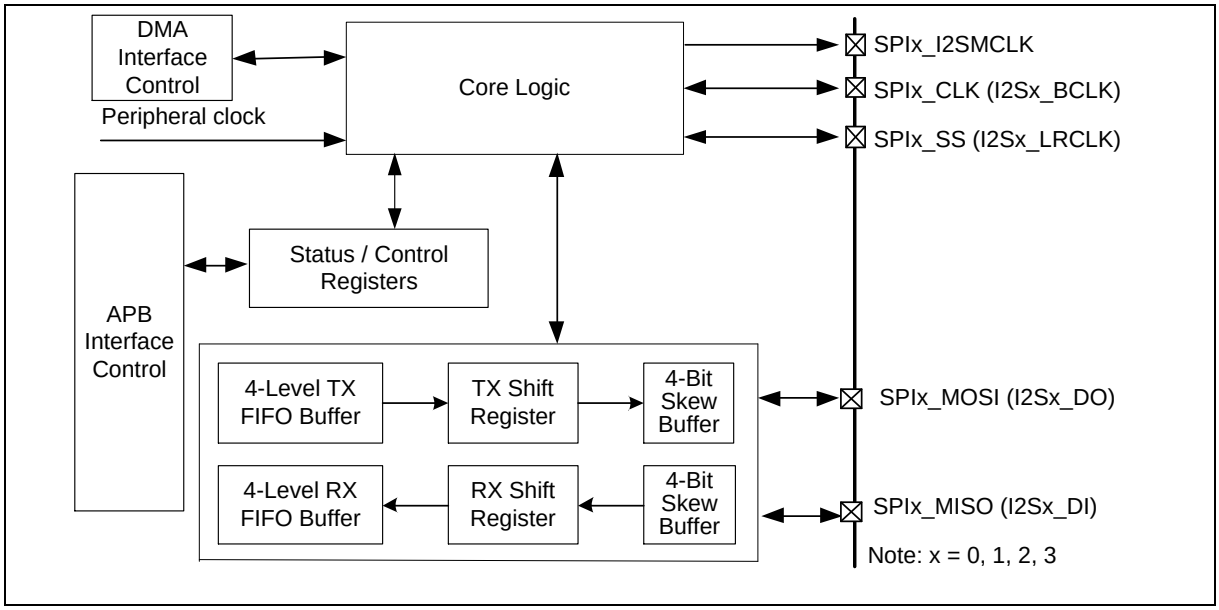


Figure 6.29-1 SPI Block Diagram

TX FIFO Buffer:

The transmit FIFO buffer is a 4-level depth, 32-bit wide, first-in, first-out register buffer. The data can be written to the transmit FIFO buffer in advance through software by writing the SPIx_TX register. In SPI mode, the transmit FIFO will be configured as 8-level while data length is set as 4~16 bits.

RX FIFO Buffer:

The receive FIFO buffer is also a 4-level depth, 32-bit wide, first-in, first-out register buffer. The receive control logic will store the receive data to this buffer. The FIFO buffer data can be read from SPIx_RX register by software. In SPI mode, the receive FIFO will be configured as 8-level while data length is set as 4~16 bits.

TX Shift Register:

The transmit shift register is a 32-bit wide register buffer. The transmit data is loaded from the TX FIFO buffer and shifted out bit-by-bit to the skew buffer.

RX Shift Register:

The receive shift register is also a 32-bit wide register buffer. The receive data is shift in bit-by-bit from the skew buffer and is loaded into RX FIFO buffer when a transaction done.

Skew Buffer:

The skew buffer is a 4-level 1-bit buffer. There are two skew buffers in transmitting and received side. In received side, it is used to shift bits into RX shift register from SPI bus. In transmitting side, it is used to shift bits into SPI bus from TX shift register.

6.29.4 Basic Configuration

6.29.4.1 SPI0 Basic Configuration

- Clock source Configuration
  - Select the source of SPI0 peripheral clock on SPI0SEL (CLK_CLKSEL2[6:4]).
  - Enable SPI0 peripheral clock in SPI0CKEN (CLK_APBCLK0[13]).
- Reset Configuration

- Reset SPI0 controller in SPI0RST (SYS_IPRST1[13]).

6.29.4.2 SPI1 Basic Configuration

- Clock source Configuration
  - Select the source of SPI1 peripheral clock on SPI1SEL (CLK_CLKSEL2[14:12]).
  - Enable SPI1 peripheral clock in SPI1CKEN (CLK_APBCLK0[14]).
- Reset Configuration
  - Reset SPI1 controller in SPI1RST (SYS_IPRST1[14]).

6.29.4.3 SPI2 Basic Configuration

- Clock source Configuration
  - Select the source of SPI2 peripheral clock on SPI2SEL (CLK_CLKSEL3[10:8]).
  - Enable SPI2 peripheral clock in SPI2CKEN (CLK_APBCLK0[15]).
- Reset Configuration
  - Reset SPI2 controller in SPI2RST (SYS_IPRST1[15]).

6.29.4.4 SPI3 Basic Configuration

- Clock source Configuration
  - Select the source of SPI3 peripheral clock on SPI3SEL (CLK_CLKSEL3[14:12]).
  - Enable SPI3 peripheral clock in SPI3CKEN (CLK_APBCLK1[6]).
- Reset Configuration
  - Reset SPI3 controller in SPI3RST (SYS_IPRST2[6]).

SPI/I^{2}S (SPI0~SPI3) Interface Controller Pin description is shown as follows:

Pin	SPI Mode	I^{2}S Mode
SPIx_SS	SPI slave selection pin	I^{2}S left/right channel synchronization clock pin (I2Sx_LRCLK)
SPIx_CLK	SPI clock pin	I^{2}S bit clock pin (I2Sx_BCLK)
SPIx_MISO	SPI master input or slave output pin	I^{2}S data input pin (I2Sx_DI)
SPIx_MOSI	SPI master output or slave input pin	I^{2}S data output pin (I2Sx_DO)
SPIx_I2SMCLK	Not available	I^{2}S Master clock output pin

Table 6.29-1 SPI/I^{2}S Interface Controller Pin Description (SPI0~SPI3)

6.29.5 Functional Description

6.29.5.1 Terminology

SPI Peripheral Clock and SPI Bus Clock

The SPI controller needs the peripheral clock to drive the SPI logic unit to perform the data transfer. The peripheral clock rate is determined by the settings of clock divisor (SPIx_CLKDIV) and the clock source which can be HXT, PLL, PCLK or HIRC (12M or 48M). SPIxSEL of CLK_CLKSEL2 or CLK_CLKSEL3 registers determine the clock source of the peripheral clock. The DIVIDER (SPIx_CLKDIV[8:0]) setting determines the divisor of the clock rate calculation.

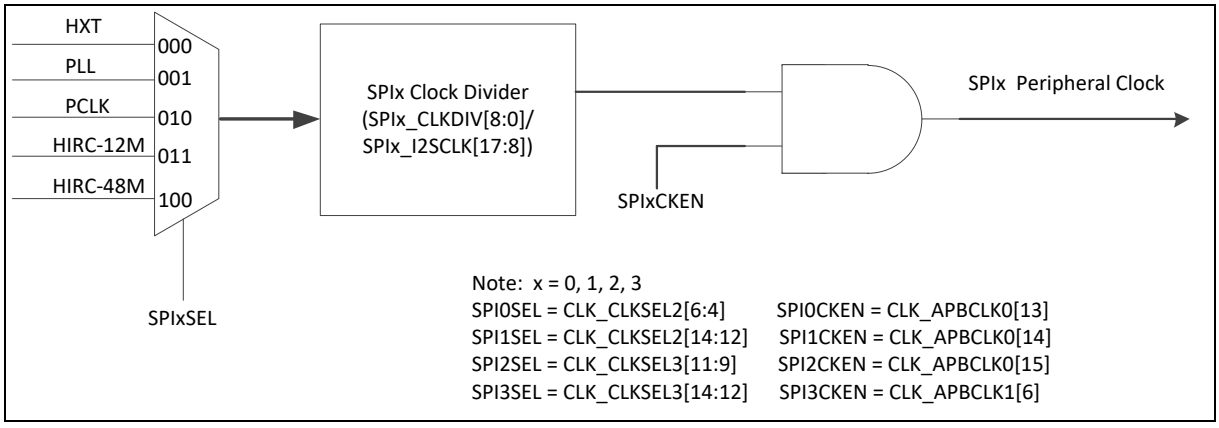


Figure 6.29-2 SPI Peripheral Clock

In Master mode, the frequency of the SPI bus clock is equal to the peripheral clock rate. In general, the SPI bus clock is denoted as SPI clock. In Slave mode, the SPI bus clock is provided by a master device. The frequency of SPI peripheral clock cannot be faster than the system clock rate regardless of Master or Slave mode. If the clock source of peripheral clock is not system clock, the frequency of SPI peripheral clock shall be slower than the system clock frequency regardless of Master or Slave mode.

In I^2S mode, the peripheral clock rate is equal to I^2S bit clock rate determined by SPIx_I2SCLK register.

Master/Slave mode

The SPI controllers can be set as Master or Slave mode by setting the SLAVE (SPIx_CTL[18]) to communicate with the off-chip SPI slave or master device. The HALFDPX (SPIx_CTL[14]) can be used to select the full-duplex or half-duplex in SPI transmission. The application block diagrams in Master and Slave mode are shown below.

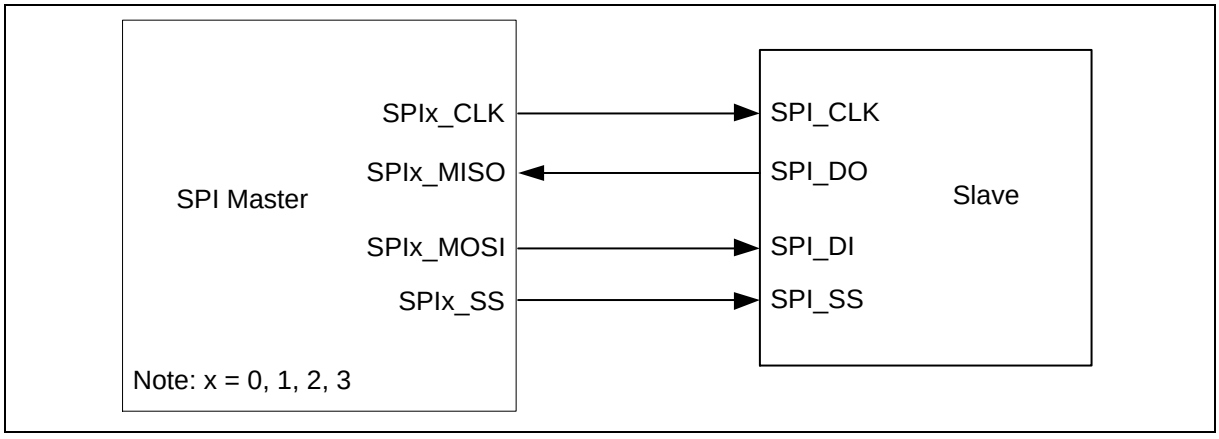


Figure 6.29-3 SPI Full-Duplex Master Mode Application Block Diagram

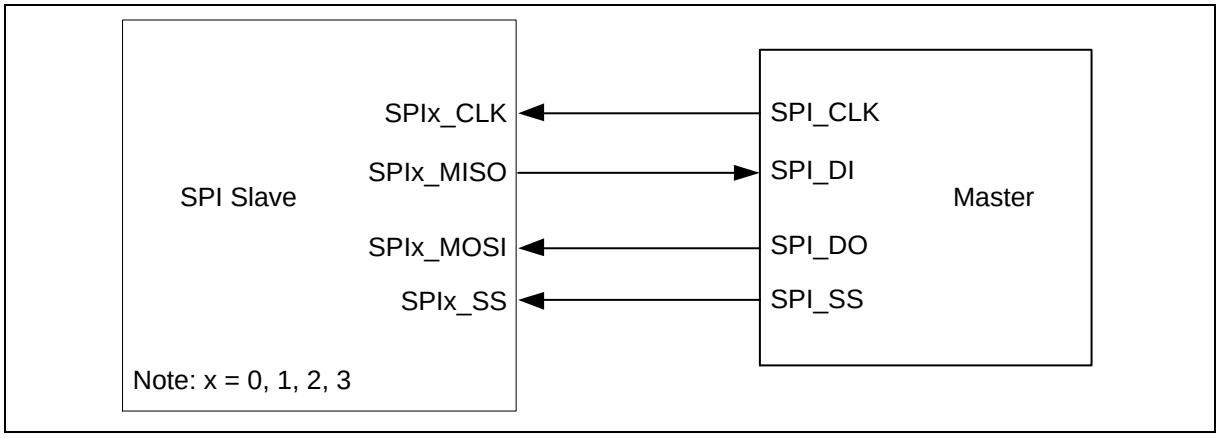


Figure 6.29-4 SPI Full-Duplex Slave Mode Application Block Diagram

Slave Selection

In Master mode, the SPI controller can drive off-chip slave device through the slave select output pin SPIx_SS. In Slave mode, the off-chip master device drives the slave selection signal from the SPIx_SS input port to this SPI controller. The duration between the slave select active edge and the first SPI clock input shall over 3 SPI peripheral clock cycles of slave.

In Master/Slave mode, the active state of slave selection signal can be programmed to low or high active in SSACTPOL (SPIx_SSCTL[2]). The selection of slave select conditions depends on what type of device is connected. In Slave mode, to recognize the inactive state of the slave selection signal, the inactive period of the slave selection signal must be larger than or equal to 3 peripheral clock cycles between two successive transactions.

In Slave mode, the SPIx_MISO signal type is controlled by the SPIx_SS pin. If the SPIx_SS pin is inactive state, the SPIx_MISO is set to tri-state to avoid interference with other Slave devices. If the SPIx_SS pin is active state, the SPIx_MISO is set to output mode for data transfer.

Timing Condition

The CLKPOL (SPIx_CTL[3]) defines the SPI clock idle state. If CLKPOL = 1, the output SPI clock is idle at high state; if CLKPOL = 0, it is idle at low state.

TXNEG (SPIx_CTL[2]) defines the data transmitted out either on negative edge or on positive edge of SPI clock. RXNEG (SPIx_CTL[1]) defines the data received either on negative edge or on positive edge of SPI clock.

Note: The settings of TXNEG and RXNEG are mutually exclusive. In other words, do not transmit and receive data at the same clock edge.

Transmit/Receive Bit Length

The bit length of a transaction word is defined in DWIDTH (SPIx_CTL[12:8]) and can be configured up to 32-bit length in a transaction word for transmitting and receiving.

When SPI controller finishes a transaction, i.e. receives or transmits a specific count of bits defined in DWIDTH (SPIx_CTL[12:8]), the unit transfer interrupt flag UNITIF (SPIx_STATUS[1]) will be set to 1.

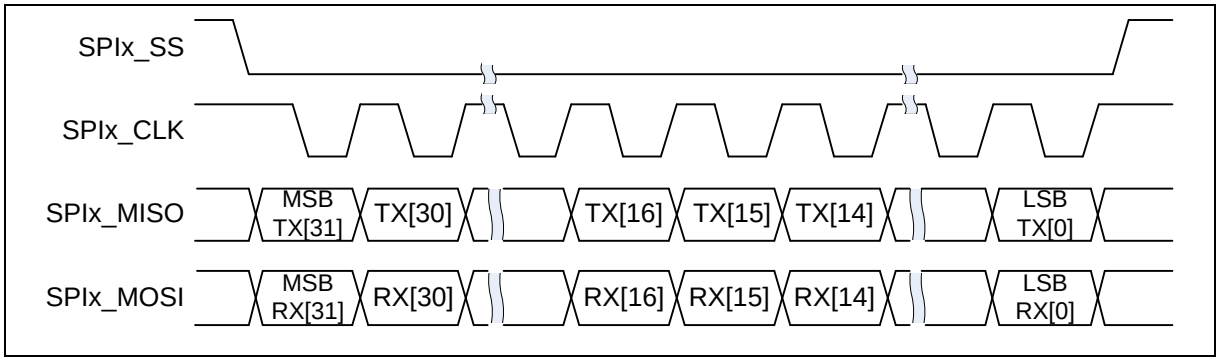


Figure 6.29-5 32-bit in One Transaction

LSB/MSB First

LSB (SPIx_CTL[13]) defines the bit transfer sequence in a transaction. If the LSB (SPIx_CTL[13]) is set to 1, the transfer sequence is LSB first. The bit 0 will be transferred firstly. If the LSB (SPIx_CTL[13]) is cleared to 0, the transfer sequence is MSB first.

Suspend Interval

SUSPITV (SPIx_CTL[7:4]) provides a configurable suspend interval, 0.5 ~ 15.5 SPI clock periods, between two successive transaction words in Master mode. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SUSPITV is 0x3 (3.5 SPI clock cycles).

6.29.5.2 Automatic Slave Selection

In Master mode, if AUTOSS (SPIx_SSCTL[3]) is set, the slave selection signal will be generated automatically and output to the SPIx_SS pin according to whether SS (SPIx_SSCTL[0]) is enabled or not. The slave selection signal will be set to active state by the SPI controller when the SPI data transfer is started by writing to FIFO. It will be set to inactive state when SPI bus is idle. If SPI bus is not idle, i.e. TX FIFO, TX shift register or TX skew buffer is not empty, the slave selection signal will be set to inactive state between transactions if the value of SUSPITV (SPIx_CTL[7:4]) is greater than or equal to 3.

In Master mode, if the value of SUSPITV is less than 3 and the AUTOSS is set as 1, the slave selection signal will be kept at active state between two successive transactions.

If the AUTOSS bit is cleared, the slave selection output signal will be determined by the SS setting. The active state of the slave selection output signal is specified in SSACTPOL (SPIx_SSCTL[2]).

The duration between the slave selection signal active edge and the first SPI bus clock edge is 1 SPI bus clock cycle and the duration between the last SPI bus clock and the slave selection signal inactive edge is 1.5 SPI bus clock cycle.

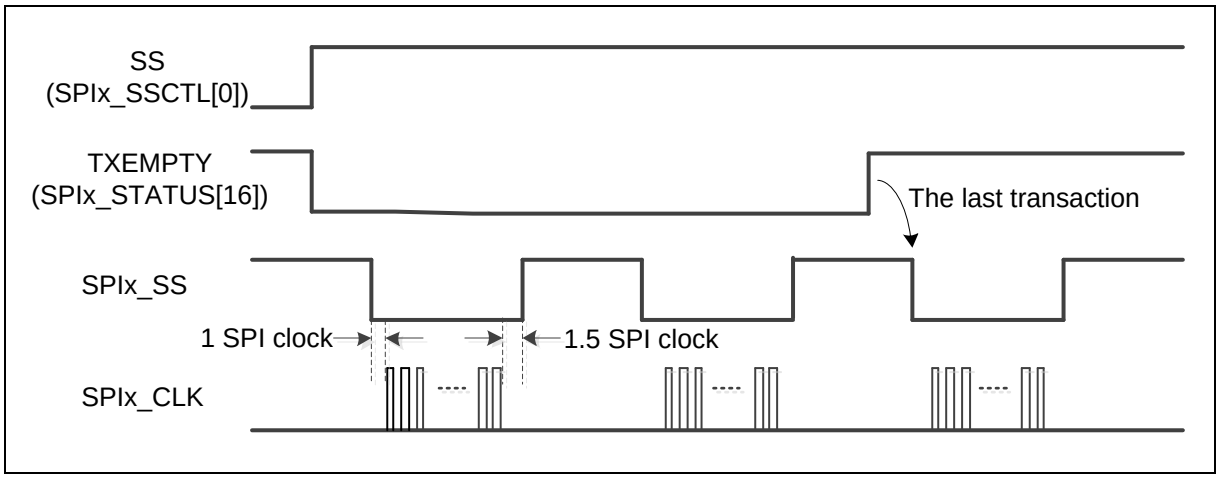


Figure 6.29-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2)

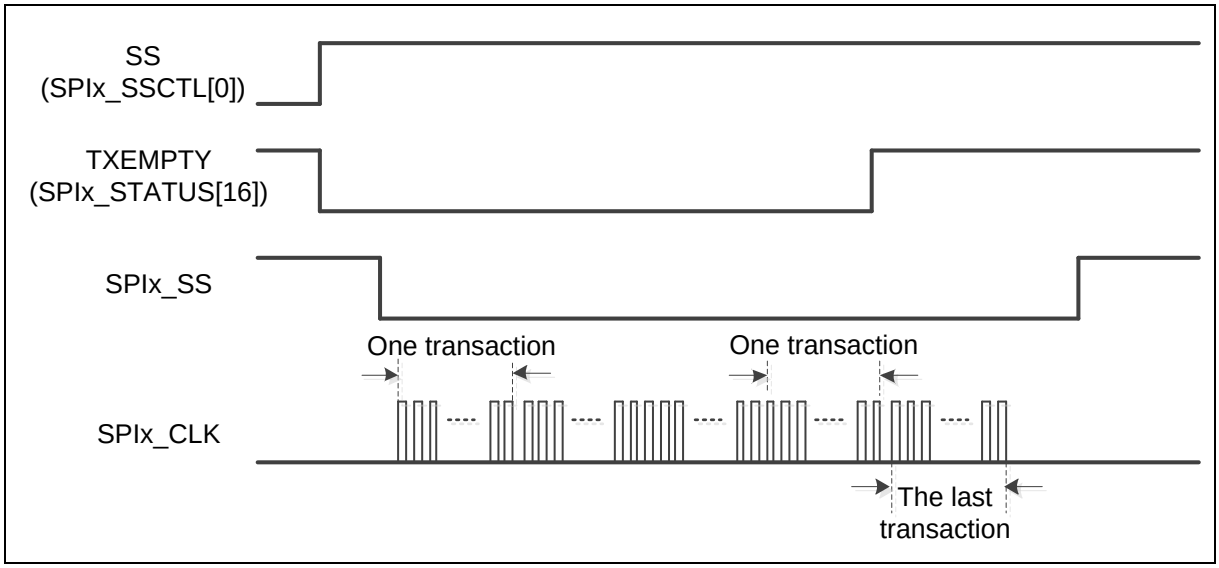


Figure 6.29-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3)

6.29.5.3 Byte Reorder and Suspend Function

When the transfer is set as MSB first (LSB = 0) and the REORDER (SPIx_CTL[19]) is set to 1, the data stored in the TX buffer and RX buffer will be rearranged in the order as [Byte0, Byte1, Byte2, Byte3] in 32-bit transfer (DWIDTH = 0). The sequence of transmitted/received data will be Byte0, Byte1, Byte2, and then Byte3. If the DWIDTH is set as 24-bit transfer mode, the data in TX buffer and RX buffer will be rearranged as [unknown byte, Byte0, Byte1, Byte2]. The SPI controller will transmit/receive data with the sequence of Byte0, Byte1 and then Byte2. Each byte will be transmitted/received with MSB first. The rule of 16-bit mode is the same as above. Byte Reorder function is only available when DWIDTH is configured as 16, 24, and 32 bits.

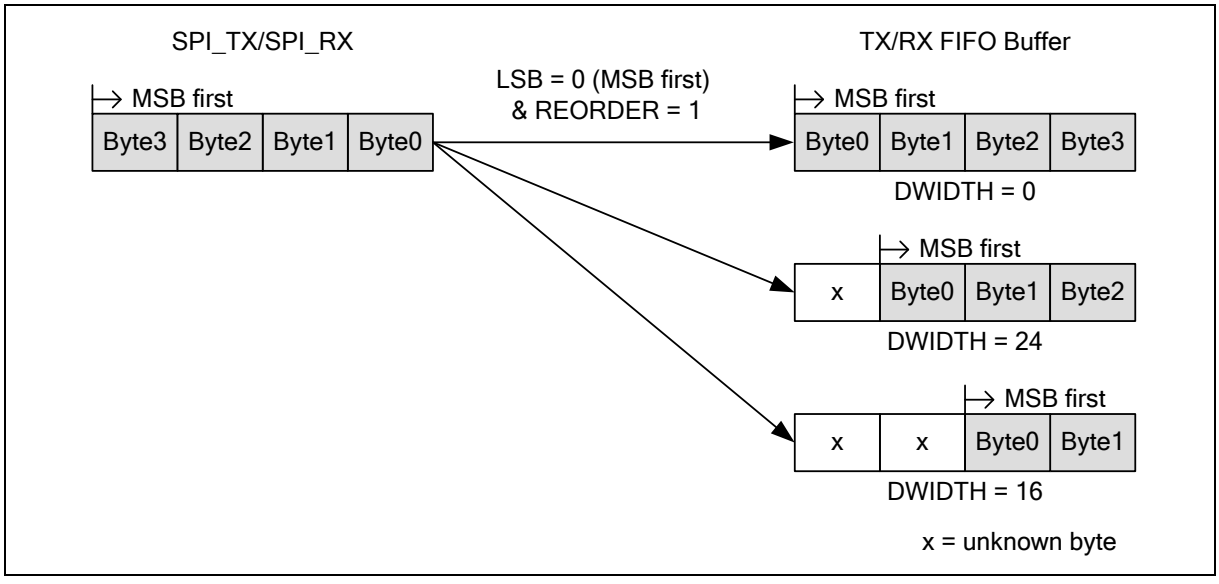


Figure 6.29-8 Byte Reorder Function

In Master mode, if REORDER (SPIx_CTL[19]) is set to 1, a suspend interval of 0.5 ~ 15.5 SPI clock periods will be inserted by hardware between two successive bytes in a transaction word. The suspend

interval is configured in SUSPITV (SPIx_CTL[7:4]).

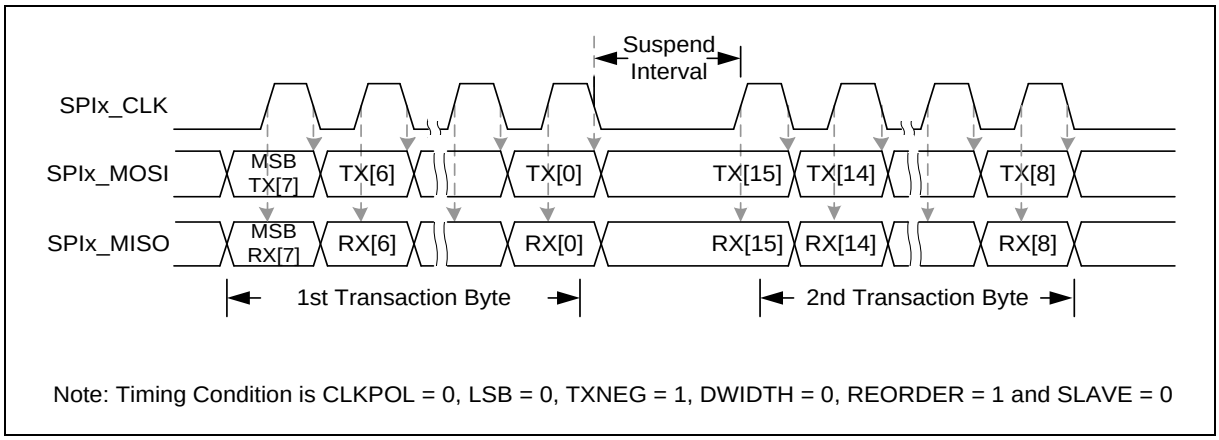


Figure 6.29-9 Timing Waveform for Byte Suspend

6.29.5.4 Half-Duplex Communication

The SPI controller can communicate in half-duplex mode by setting HALFDPX (SPIx_CTL[14]) bit. In half-duplex mode, there is only one data line for receiving or transmitting data direction which is defined by DATDIR (SPIx_CTL[20]). In half-duplex configuration, the SPIx_MISO pin is free for other applications and it can be configured as GPIO. Enabling or disabling the control bit HALFDPX (SPIx_CTL[14]) will produce TXFBCLR (SPIx_FIFOCTL[9]) and RXFBCLR (SPIx_FIFOCTL[8]) at the same time automatically.

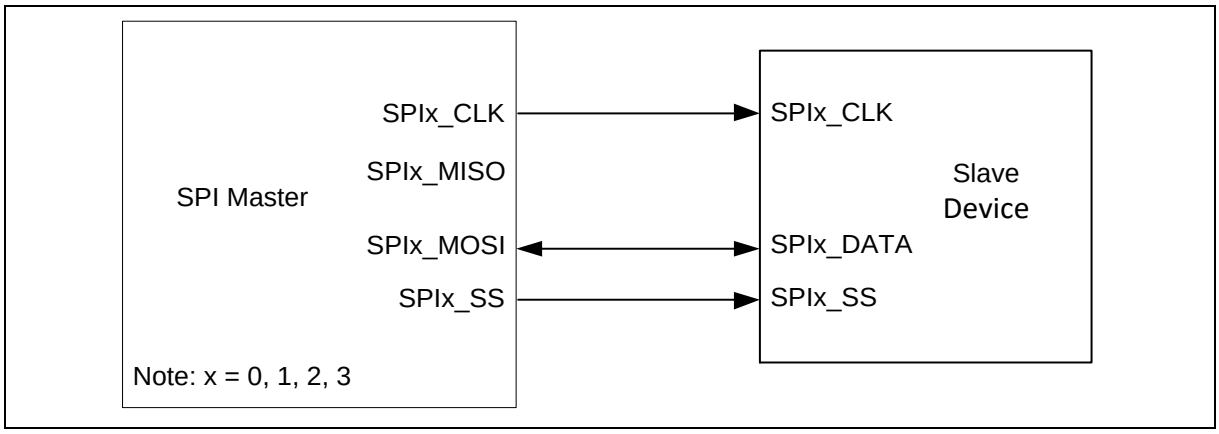


Figure 6.29-10 SPI Half-Duplex Master Mode Application Block Diagram

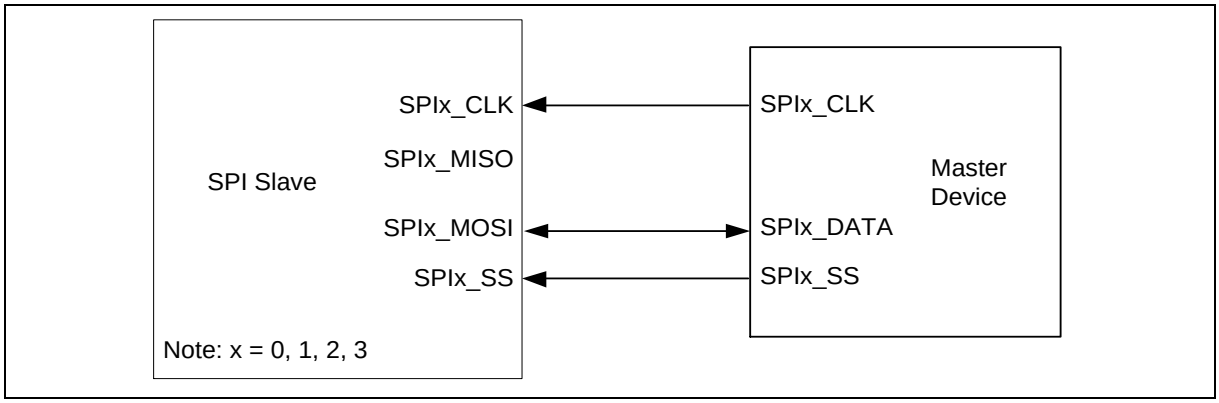


Figure 6.29-11 SPI Half-Duplex Slave Mode Application Block Diagram

6.29.5.5 Receive-Only Mode

In SPI Master device, it can communicate in receive-only mode by setting RXONLY (SPIx_CTL[15]). In this configuration, the SPI Master device will generate SPI bus clock continuously as long as the receive-only mode is enabled for receiving data bit from SPI slave device. If AUTOSS (SPIx_SSCTL[3]) is enabled in receive-only mode, SPI Master will keep activating the slave select signal.

The remaining SPIx_MOSI pin of SPI Master device is not used for communication and can be configured as GPIO. The status BUSY (SPIx_STATUS[0]) will be asserted in receive-only mode due to the generation of SPI bus clock. Entering this mode will produce the TXFBCLR (SPIx_FIFOCTL[9]) and RXFBCLR (SPIx_FIFOCTL[8]) at the same time automatically. When user enables this mode, the output SPI bus clock will be sent out after 6 peripheral clock cycles. In this mode, the data which has been written into transmit FIFO will be loaded into transmit shift register and sent out.

When user sets RXONLY (SPIx_CTL[15]) to enable, SPI RX data with data bit width of DWIDTH (SPIx_CTL[12:8]) will be received into RX FIFO and SPI clock will be sent to SPI slave device until RX FIFO is full.

For data bit width of 4~16 bits, the SPI master will send SPI output clock to SPI slave and receive RX data when RX FIFO counter RXCNT (SPIx_STATUS[27:24]) is less than or equal to 6.

For data bit width of 17~32 bits, the SPI master will send SPI output clock to SPI slave and receive RX data when RX FIFO counter RXCNT (SPIx_STATUS[27:24]) is less than or equal to 2.

6.29.5.6 Slave 3-Wire Mode

When SLV3WIRE (SPIx_SSCTL[4]) is set by software to enable the Slave 3-Wire mode, the SPI controller can work with no slave selection signal in Slave mode. The SLV3WIRE (SPIx_SSCTL[4]) only takes effect in Slave mode. Only three pins, SPIx_CLK, SPIx_MISO, and SPIx_MOSI, are required to communicate with a SPI master. The SPIx_SS pin can be configured as a GPIO. When the SLV3WIRE (SPIx_SSCTL[4]) is set to 1, the SPI slave will be ready to transmit/receive data after the SPIEN (SPIx_CTL[0]) is set to 1.

6.29.5.7 PDMA Transfer Function

SPI controller supports PDMA transfer function.

When TXPDMAEN (SPIx_PDMACTL[0]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (SPIx_PDMACTL[1]) is set to 1, the controller will start the PDMA reception process. SPI controller will issue request to PDMA controller automatically when there is data in the RX FIFO buffer.

Note: SPI supports single request PDMA (Read/Write) only; burst request PDMA is not supported.

6.29.5.8 FIFO Buffer Operation

The SPI controllers are equipped with four 32-bit wide transmit and receive FIFO buffers. The data stored in the transmit FIFO buffer will be read and sent out by the transmission control logic. If the transmit FIFO buffer is full, the TXFULL (SPIx_STATUS[17]) will be set to 1. When the SPI transmission logic unit draws out the last datum of the transmit FIFO buffer, so that the transmit FIFO buffer is empty, the TXEMPTY (SPIx_STATUS[16]) will be set to 1. Note that the TXEMPTY (SPIx_STATUS[16]) flag is set to 1 while the last transaction is still in progress. In Master mode, the BUSY (SPIx_STATUS[0]) is set to 1 when the FIFO buffer is written any data or there is any transaction on the SPI bus. (e.g. the slave selection signal is active and the SPI controller is receiving data in Slave mode). It will set to 0 when the transmit FIFO is empty and the current transaction has done. Thus, the status of BUSY (SPIx_STATUS[0]) should be checked by software to make sure whether the SPI is in idle or not.

The receive control logic will store the SPI input data into the receive FIFO buffer. There are FIFO related status bits, like RXEMPTY (SPIx_STATUS[8]) and RXFULL (SPIx_STATUS[9]), to indicate the current status of RX FIFO buffer.

The transmitting and receiving threshold can be configured by setting TXTH (SPIx_FIFOCTL[30:28])

and RXTH (SPIx_FIFOCTL[26:24]). When the count of valid data stored in transmit FIFO buffer is less than or equal to TXTH (SPIx_FIFOCTL[30:28]) setting, TXTHIF (SPIx_STATUS[18]) will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RXTH (SPIx_FIFOCTL[26:24]) setting, RXTHIF (SPIx_STATUS[10]) will be set to 1.

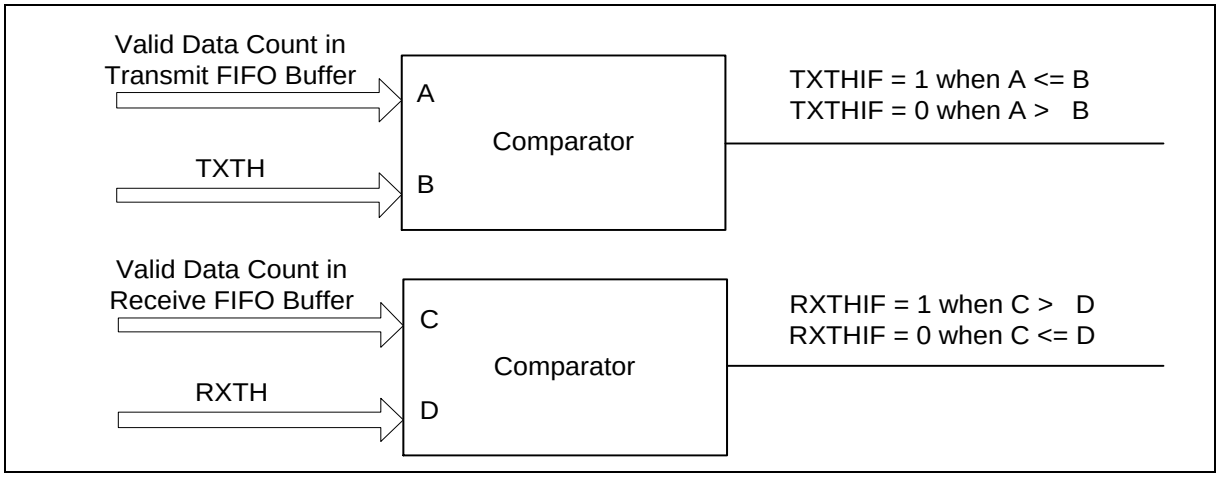


Figure 6.29-12 FIFO Threshold Comparator

In Master mode, when the first datum is written to the SPIx_TX register, the TXEMPTY flag (SPIx_STATUS[16]) will be cleared to 0. The transmission will start after 1 PCLK clock cycles and 6 peripheral clock cycles. User can write the next data into SPIx_TX register immediately. The SPI controller will insert a suspend interval between two successive transactions. The period of suspend interval is decided by the setting of SUSPITV (SPIx_CTL[7:4]). If the SUSPITV (SPIx_CTL[7:4]) equals 0, SPI controller can perform continuous transfer. User can write data into SPIx_TX register as long as the TXFULL (SPIx_STATUS[17]) is 0.

The example 1 of Figure 6.30-13 indicates the updated condition of TXEMPTY (SPIx_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer for 4~16 bits of data length. The TXEMPTY (SPIx_STATUS[16]) is set to 0 when the Data0 is written into the FIFO buffer. The Data0 will be loaded into the shift register by the core logic and the TXEMPTY (SPIx_STATUS[16]) will be 1. The Data0 in shift register will be shifted into skew buffer by bit for transmission until the transfer is done.

The Example 2 of Figure 6.30-13 indicates the updated condition of TXFULL (SPIx_STATUS[17]) when there are 8 data in the FIFO buffer and the next data of Data9 is not written into the FIFO buffer when the TXFULL = 1.

The example 1 of Figure 6.30-14 indicates the updated condition of TXEMPTY (SPIx_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer for 17~32 bits of data length.

The Example 2 of Figure 6.30-14 indicates the updated condition of TXFULL (SPIx_STATUS[17]) when there are 4 data in the FIFO buffer and the next data of Data5 is not written into the FIFO buffer when the TXFULL = 1.

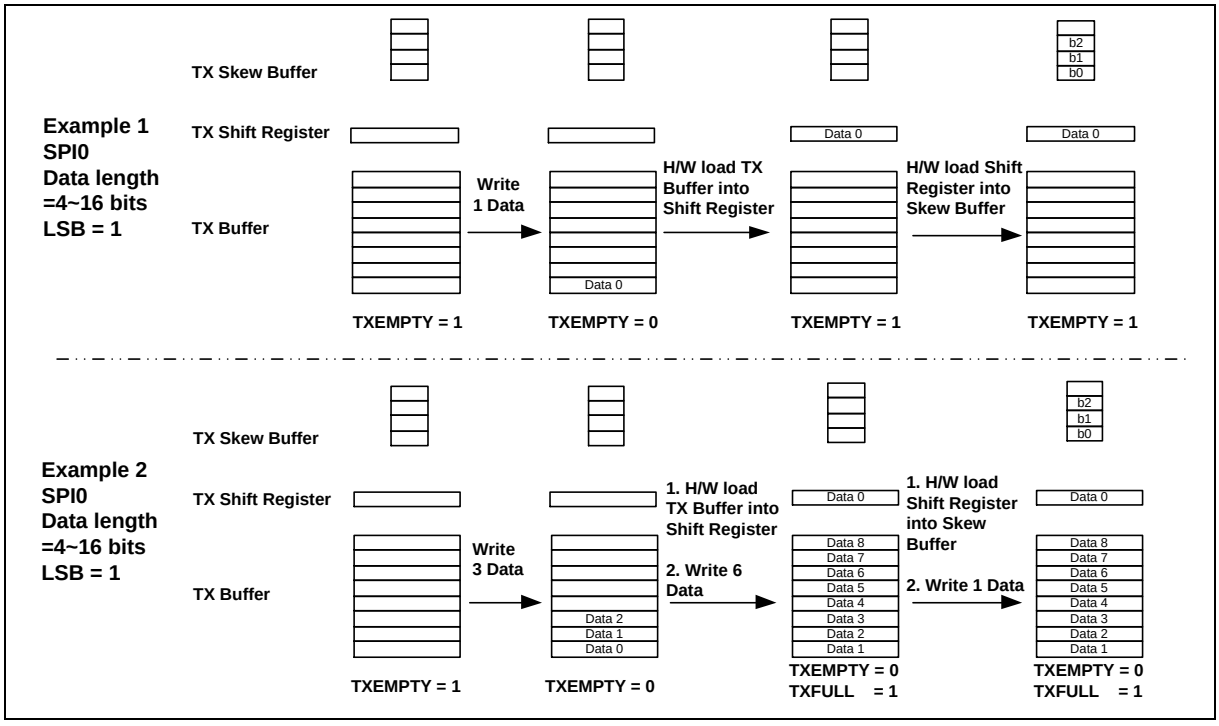


Figure 6.29-13 Transmit FIFO Buffer Example for 4~16 bits of data length

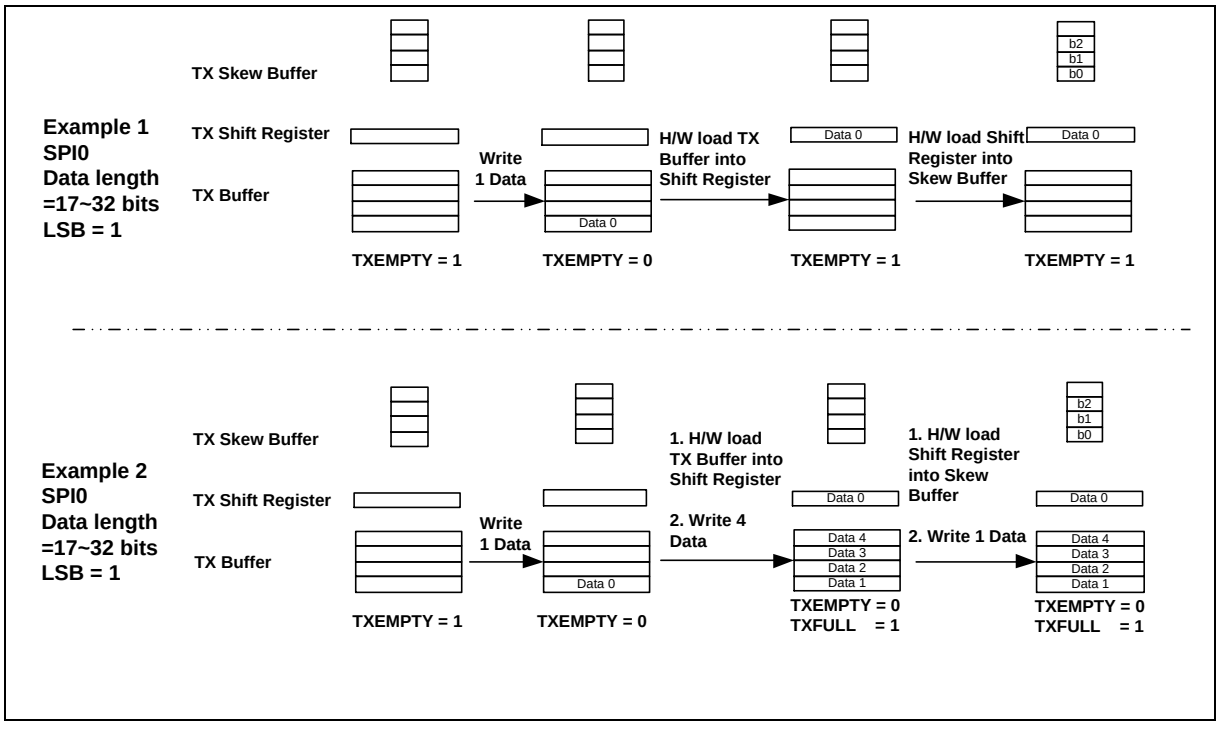


Figure 6.29-14 Transmit FIFO Buffer Example for 17~32 bits of data length

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the SPIx_TX register is not updated after all data transfer are done, the transfer will stop.

In Master mode, during receiving operation, the serial data are received from SPIx_MISO pin and stored to receive FIFO buffer.

The received data (Data0's b0, b1, …b31) is stored into skew buffer first according the serial clock (SPIx_CLK) and then it is shift into the shift register bit by bit. The core logic will load the data in shift register into FIFO buffer when the received data bit count reach the value of DWIDTH (SPIx_CTL[12:8]).

The RXEMPTY (SPIx_STATUS[8]) will be cleared to 0 while the receive FIFO buffer contains unread data (see the Example 1 of Receive FIFO Buffer Example in Figure 6.30-15). The received data can be read by software from SPIx_RX register as long as the RXEMPTY (SPIx_STATUS[8]) is 0. If the receive FIFO buffer contains 8 unread data, the RXFULL (SPIx_STATUS[9]) will be set to 1 (see the Example 2 of Receive FIFO Buffer Example in Figure 6.30-15).

The RXEMPTY (SPIx_STATUS[8]) will be cleared to 0 while the receive FIFO buffer contains unread data (see the Example 1 of Receive FIFO Buffer Example in Figure 6.30-16). The received data can be read by software from SPIx_RX register as long as the RXEMPTY (SPIx_STATUS[8]) is 0. If the receive FIFO buffer contains 4 unread data, the RXFULL (SPIx_STATUS[9]) will be set to 1 (see the Example 2 of Receive FIFO Buffer Example in Figure 6.30-16).

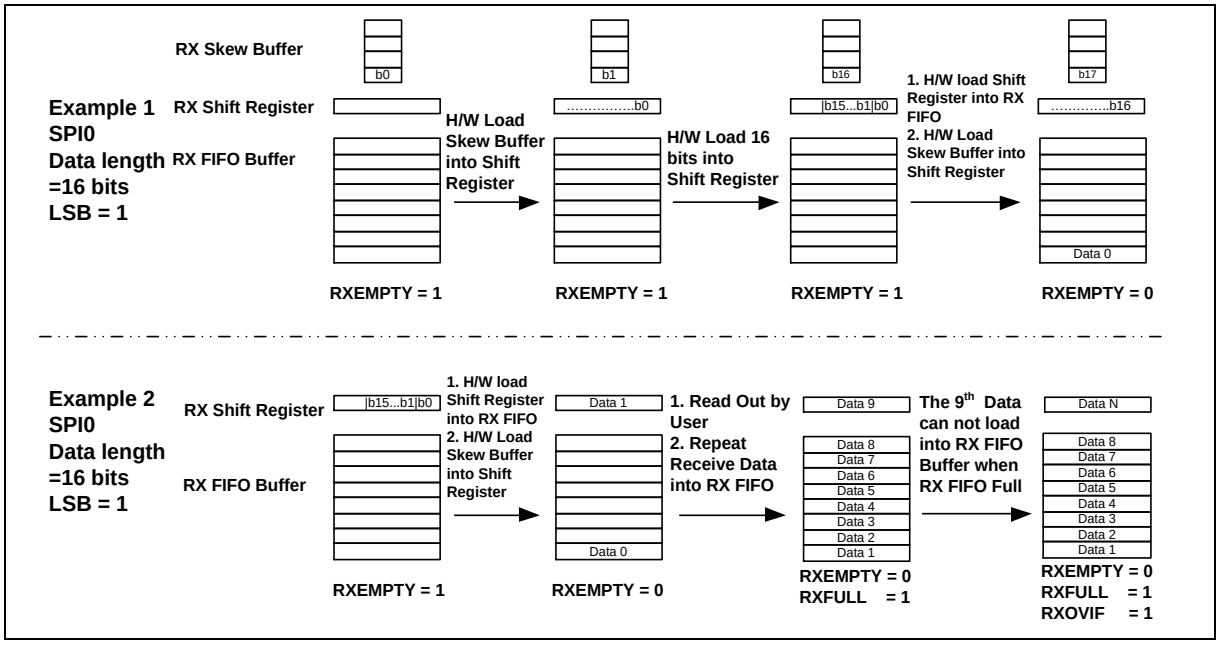


Figure 6.29-15 Receive FIFO Buffer Example for 16 bits of Data Length

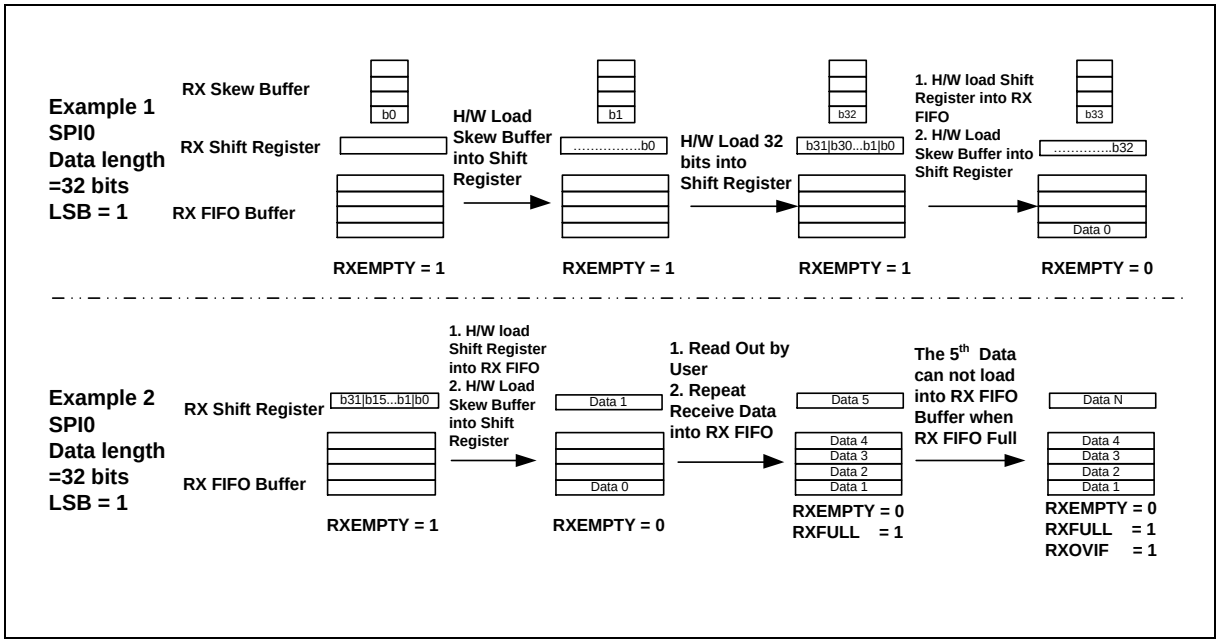


Figure 6.29-16 Receive FIFO Buffer Example for 32 bits of Data Length

In Slave mode, during transmission operation, when data is written to the SPIx_TX register by software, the data will be loaded into transmit FIFO buffer and the TXEMPTY (SPIx_STATUS[16]) will be set to 0. The transmission will start when the slave device receives clock signal from master. Data can be written to SPIx_TX register as long as the TXFULL (SPIx_STATUS[17]) is 0. After all data have been drawn out by the SPI transmission logic unit and the SPIx_TX register is not updated by software, the TXEMPTY (SPIx_STATUS[16]) will be set to 1.

If there is no any data written to the SPIx_TX register, the transmit underflow interrupt flag, TXUFIF (SPIx_STATUS[19]) will be set to 1 when the slave selection signal is active. The output data will be held by TXUFPOL (SPIx_FIFOCTL[6]) setting during this transfer until the slave selection signal goes to inactive state. When the transmit underflow event occurs, the slave under run interrupt flag, SLVURIF (SPIx_STATUS[7]), will be set to 1 as SPIx_SS goes to inactive state.

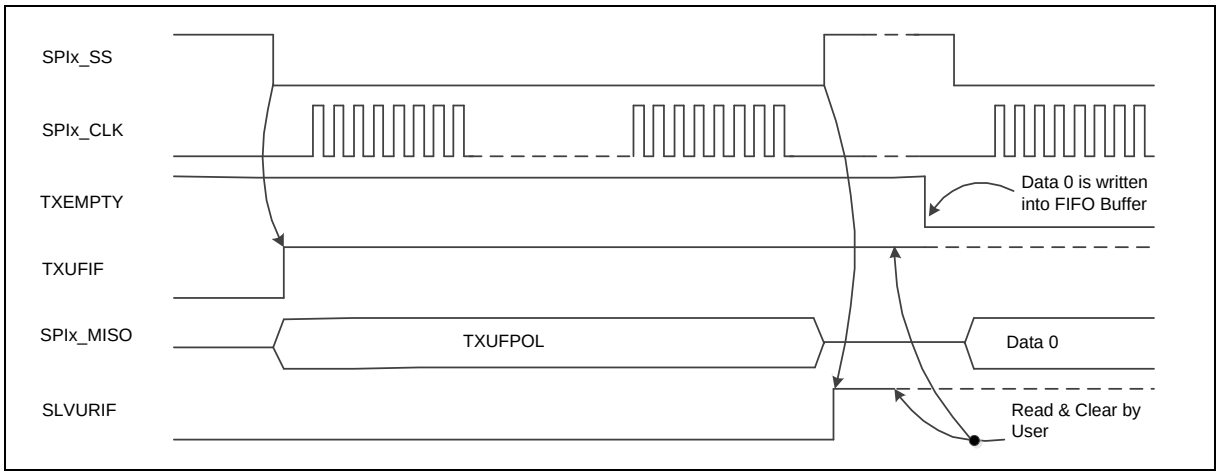


Figure 6.29-17 TX Underflow Event and Slave Under Run Event

In SPI Slave 3-Wire mode, the first 2-bit data is un-predicted (kept on the level of last bit in previously transfer) if the data is written into TX FIFO among 3 peripheral clock cycles before the SPI bus clock is presented. The other bits are held by TXUFPOL (SPIx_FIFOCTL[6]) because there is TX underflow

event. The written data will be transmitted in the next transfer.

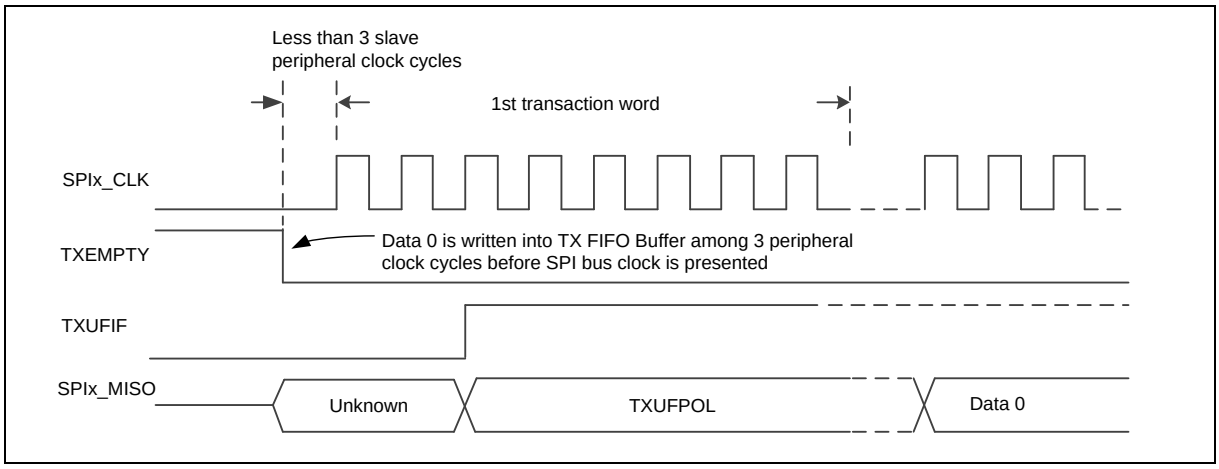


Figure 6.29-18 TX Underflow Event (SPI Slave 3-Wire Mode Enabled)

In Slave mode, during receiving operation, the serial data is received from SPIx_MOSI pin and stored to SPIx_RX register. The reception mechanism is similar to Master mode reception operation. If the receive FIFO buffer contains 4 unread data, the RXFULL (SPIx_STATUS[9]) will be set to 1 and the RXOVIF (SPIx_STATUS[11]) will be set to 1 if there is more serial data received from SPIx_MOSI and follow-up data will be dropped (refer to the Receive FIFO Buffer Example figure).

If the receive bit count mismatches with the DWIDTH (SPIx_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (SPIx_STATUS[6]) will be set to 1 in SPI Slave mode. RX data will be written into RX FIFO and SLVBENUM (SPIx_STATUS2[29:24]) will be updated when SLVBERX (SPIx_FIFOCTL[10]) is enabled and SPI slave bit count error event happened. RX data will be dropped and SLVBENUM (SPIx_STATUS2[29:24]) will be fixed to 0x0 when the control register SLVBERX (SPIx_FIFOCTL[10]) is disabled and SPI slave bit count error event happened. The status register SLVBENUM (SPIx_STATUS2[29:24]) indicates that effective bit number of uncompleted RX data when SPI slave bit count error happened. In Figure 6.29-19, the timing diagram of SPI slave bit count error and SLVBENUM is shown, and SLVBENUM will be updated to 0x8 when SPI slave device receives 8 bits data and DWIDTH (SPIx_CTL[12:8]) sets to 16 bits at SPI bit count error event. The SPI slave bit count error event (SLVBEIF) will generate RX FIFO write operation pulse to write RX data from RX shift register to RX FIFO, and RX FIFO count (RXCNT) will be updated to one.

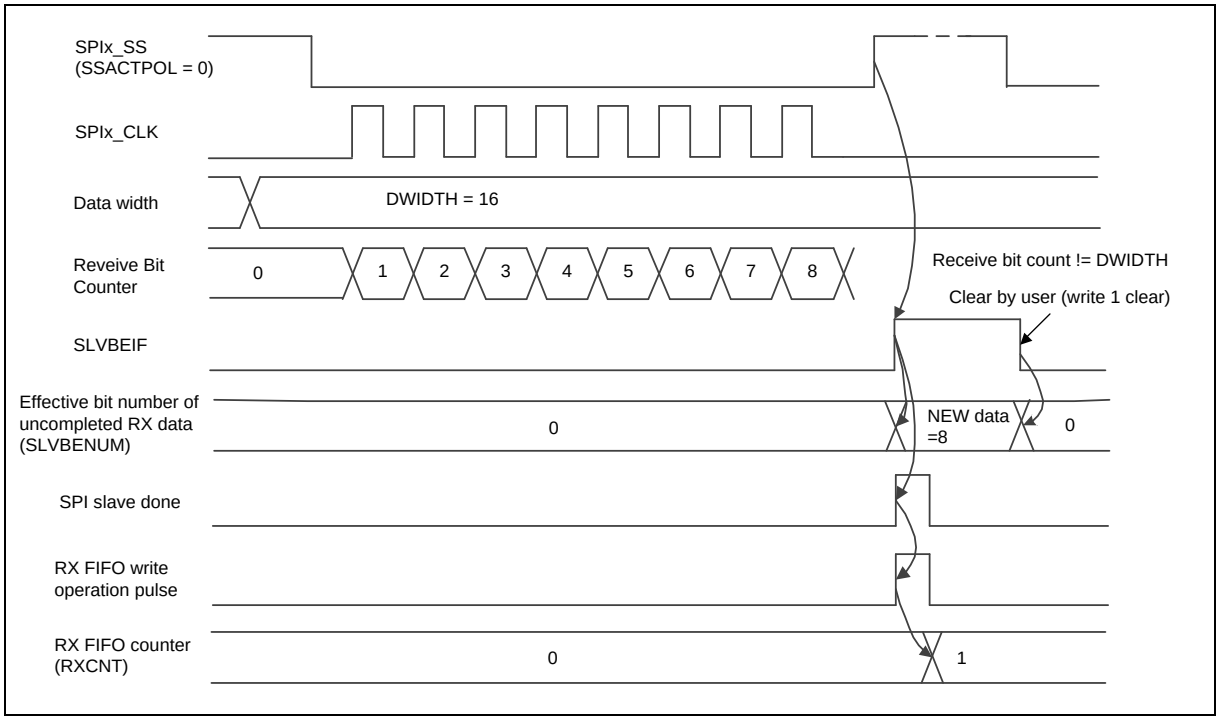


Figure 6.29-19 Slave Mode Bit Count Error and Effective Bit Number of Uncompleted RX data

A receive time-out function is built in this controller. When the receive FIFO is not empty and no read operation in receive FIFO over 64 SPI peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode, the receive time-out occurs and the RXTOIF (SPIx_STATUS[12]) will be set to 1. When the receive FIFO is read by user, the time-out status will be cleared automatically.

6.29.5.9 Interrupt

- SPI unit transfer interrupt

  As the SPI controller finishes a unit transfer, the unit transfer interrupt flag UNITIF (SPIx_STATUS[1]) will be set to 1. The unit transfer interrupt event will generate an interrupt to CPU if the unit transfer interrupt enable bit UNITIEN (SPIx_CTL[17]) is set. The unit transfer interrupt flag can be cleared only by writing 1 to it.

- SPI slave selection active/inactive interrupt

  In Slave mode, the slave selection active/inactive interrupt flag, SSACTIF (SPIx_STATUS[2]) and SSINAIF (SPIx_STATUS[3]), will be set to 1 when the SPIEN (SPIx_CTL[0]) and SLAVE (SPIx_CTL[18]) are set to 1 and the slave selection signal goes to active/inactive state. The SPI controller will issue an interrupt if the SSINAIEN (SPIx_SSCTL[13]) or SSACTIEN (SPIx_SSCTL[12]), is set to 1.

- Slave bit count error interrupt

  In Slave mode, if the transmit/receive bit count mismatches with the DWIDTH (SPIx_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (SPIx_STATUS[6]) will be set to 1. The uncompleted transaction will be dropped from TX shift register. When the control register SLVBERX (SPIx_FIFOCTL[10]) is disabled and SPI slave bit count error event happened, the uncompleted transaction data will be dropped from RX shift registers. When control register SLVBERX (SPIx_FIFOCTL[10]) is enabled and SPI slave bit count error event happened, the uncompleted transaction data will be written from RX shift registers into RX FIFO. The SPI controller will issue an interrupt if the SLVBEIEN (SPIx_SSCTL[8]) is set to 1.

Note: If the slave selection signal is active but there is no any serial clock input, the SLVBEIF (SPIx_STATUS[6]) will be set to 1 when the slave selection signal goes to inactive state.

- TX underflow interrupt

  In SPI Slave mode, if there is no any data is written to the SPIx_TX register, the TXUFIF (SPIx_STATUS[19]) will be set to 1 when the slave selection signal is active. The SPI controller will issue a TX underflow interrupt if the TXUFIEN (SPIx_FIFOCTL[7]) is set to 1.

  Note: If underflow event occurs in SPI Slave mode, there are two conditions which make SPI Slave mode return to idle state and then goes for next transfer: (1) set TXRST to 1 (2) slave select signal is changed to inactive state while SLV3WIRE=0.

- Slave TX under run interrupt

  If the TX underflow event occurs, the SLVURIF (SPIx_STATUS[7]) will be set to 1 when SPIx_SS goes to inactive state. The SPI controller will issue a TX under run interrupt if the SLVURIEN (SPIx_SSCTL[9]) is set to 1.

  Note: In Slave 3-Wire mode, the slave selection signal is considered active all the time so that user shall poll the TXUFIF (SPIx_STATUS[19]) to know if there is TX underflow event or not.

- Receive Overrun interrupt

  In Slave mode, if the receive FIFO buffer contains 4 unread data for 17~32-bits of data length, the RXFULL (SPIx_STATUS[9]) will be set to 1 and the RXOVIF (SPIx_STATUS[11]) will be set to 1 if there is more serial data received from SPI bus and follow-up data will be dropped. The SPI controller will issue an interrupt if the RXOVIEN (SPIx_FIFOCTL[5]) is set to 1.

  If the receive FIFO buffer contains 8 unread data for 4~16-bits of data length, the RXFULL (SPIx_STATUS[9]) will be set to 1 and the RXOVIF (SPIx_STATUS[11]) will be set to 1 if there is more serial data received from SPI bus and follow-up data will be dropped.The SPI controller will issue an interrupt if the RXOVIEN (SPIx_FIFOCTL[5]) is set to 1.

- Receive FIFO time-out interrupt

  If there is a received data in the FIFO buffer and it is not read by software over 64 SPI peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode, it will send a RX time-out interrupt to the system if the RX time-out interrupt enable bit, RXTOIEN (SPIx_FIFOCTL[4]), is set to 1.

- Transmit FIFO interrupt

  In FIFO mode, if the valid data count of the transmit FIFO buffer is less than or equal to the setting value of TXTH (SPIx_FIFOCTL[30:28]), the transmit FIFO interrupt flag TXTHIF (SPIx_STATUS[18]) will be set to 1. The SPI controller will generate a transmit FIFO interrupt to the system if the transmit FIFO interrupt enable bit, TXTHIEN (SPIx_FIFOCTL[3]), is set to 1.

- Receive FIFO interrupt

  In FIFO mode, if the valid data count of the receive FIFO buffer is larger than the setting value of RXTH (SPIx_FIFOCTL[26:24]), the receive FIFO interrupt flag RXTHIF (SPIx_STATUS[10]) will be set to 1. The SPI controller will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, RXTHIEN (SPIx_FIFOCTL[2]), is set to 1.

6.29.5.10 I^2S Mode

The SPI0~SPI3 controllers support I^2S mode with PCM mode A, PCM mode B, MSB justified and I^2S data format. The bit count of an audio channel is determined by WDWIDTH (SPIx_I2SCTL[5:4]). The

transfer sequence is always first from the most significant bit, MSB. Data are read on rising clock edge and are driven on falling clock edge.

In I²S data format, the MSB is sent and latched on the second clock of an audio channel. The I2Sx_LRCLK signal indicates which audio channel is in transferring.

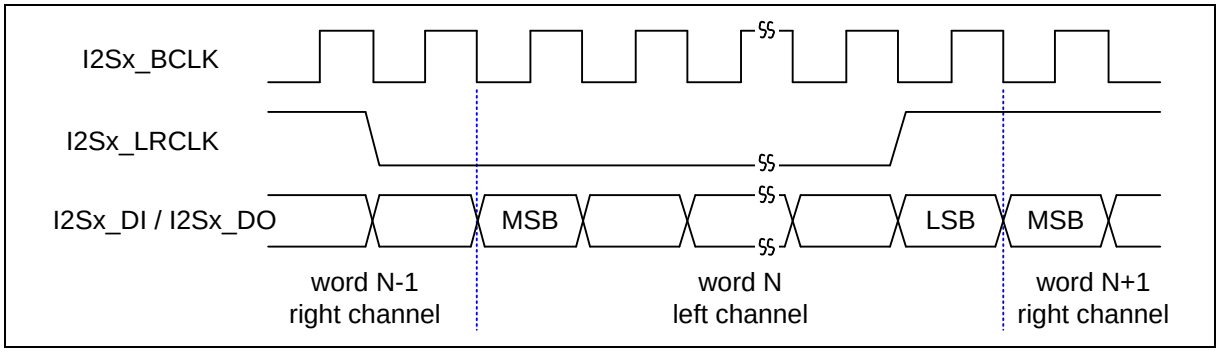


Figure 6.29-20 I²S Data Format Timing Diagram

In MSB justified data format, the MSB is sent and latched on the first clock of an audio channel.

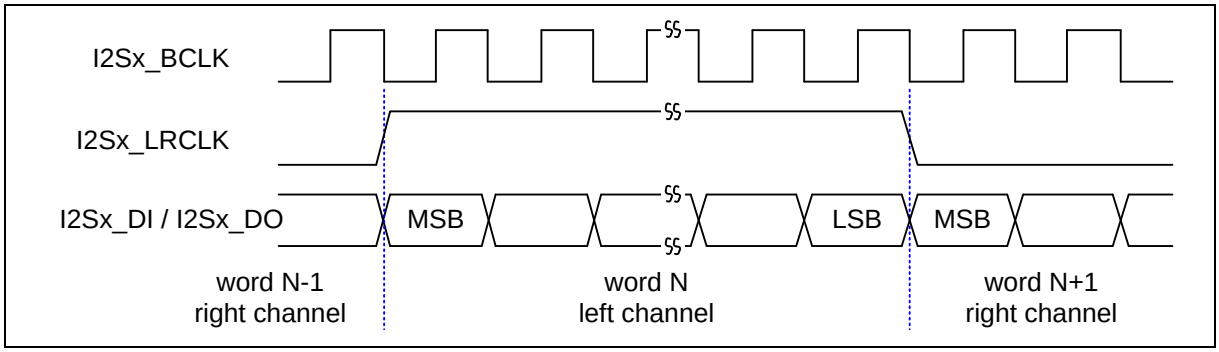


Figure 6.29-21 MSB Justified Data Format Timing Diagram

The I2Sx_LRCLK signal also supports PCM mode A and PCM mode B. The I2Sx_LRCLK signal in PCM mode indicates the beginning of an audio frame.

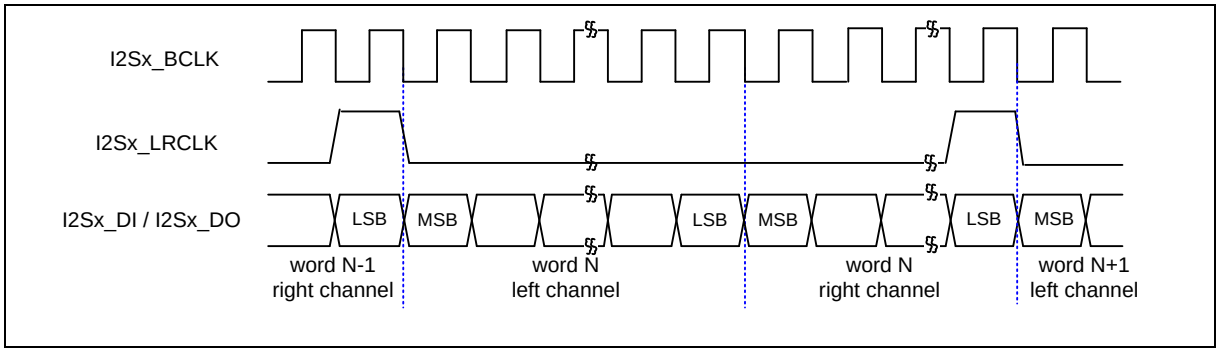


Figure 6.29-22 PCM Mode A Timing Diagram

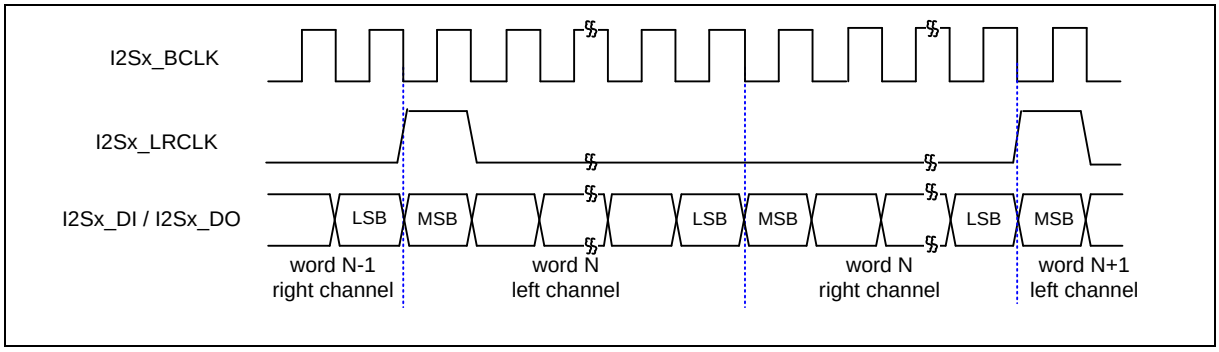


Figure 6.29-23 PCM Mode B Timing Diagram

6.29.5.11 I²S Mode FIFO Operation

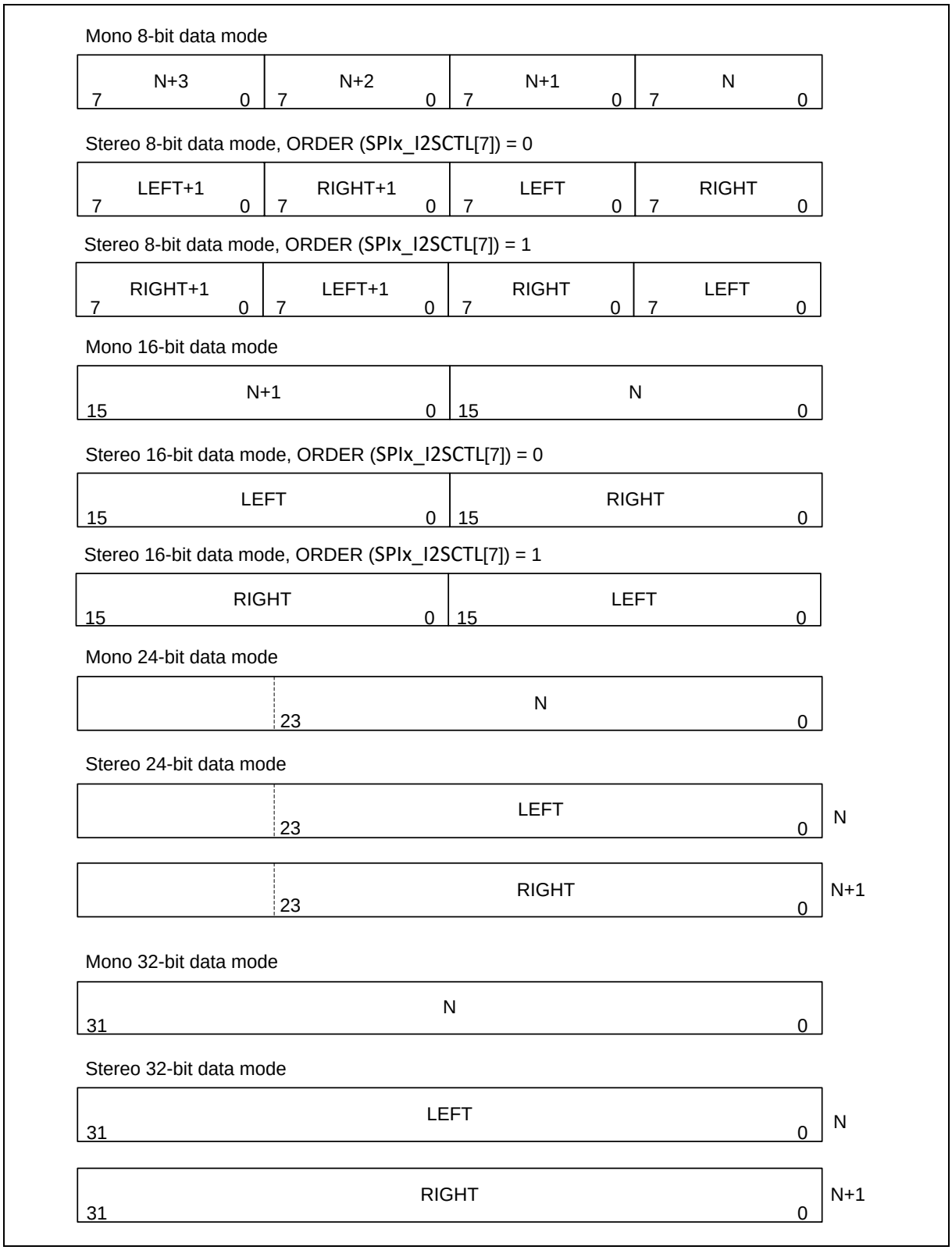


Figure 6.29-24 FIFO Contents for Various I²S Modes

6.29.5.12 Dummy Data Number for I²S / PCM Master mode and Monaural Mode

Before I²S / PCM master starts to send TX data to an external slave device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, and write TX data to TX FIFO. After master sends dummy data (data with zero value) to an external slave device, master will send TX FIFO data to an external slave device. Table 6.29-2 shows the number of dummy data for monaural mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	0
16 bits	0
24 bits	1
32 bits	1

Table 6.29-2 Dummy Data Number for I²S / PCM Master Mode and Monaural Mode

6.29.5.13 Dummy Data Number for I²S / PCM Master mode and Stereo Mode

Before I²S / PCM master starts to send TX data to an external slave device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, and write TX data to TX FIFO. After master sends dummy data (data with zero value) to an external slave device, master will send TX FIFO data to an external slave device. Table 6.29-3 shows the number of dummy data for stereo mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	0
16 bits	0
24 bits	1
32 bits	1

Table 6.29-3 Dummy Data Number for I²S / PCM Master Mode and Stereo Mode

6.29.5.14 Dummy Data Number for I²S Slave mode and Monaural Mode

Before I²S / PCM slave starts to send TX data to an external master device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, SLAVE mode (SPIx_I2SCTL[8]), and write TX data to TX FIFO. After slave sends dummy data (data with zero value) to an external master device, slave will send TX FIFO data to an external master device. Table 6.29-4 shows the number of dummy data for I²S slave monaural mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	3
16 bits	2
24 bits	2
32 bits	2

Table 6.29-4 Dummy Data Number for I²S Slave Mode and Monaural Mode

6.29.5.15 Dummy Data Number for PCM Slave Mode and Monaural Mode

Before I²S / PCM slave starts to send TX data to an external master device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, SLAVE mode (SPIx_I2SCTL[8]), and

write TX data to TX FIFO. After slave sends dummy data (data with zero value) to an external master device, slave will send TX FIFO data to an external master device. Table 6.29-5 shows the number of dummy data for PCM slave monaural mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	2
16 bits	1
24 bits	1
32 bits	1

Table 6.29-5 Dummy Data Number for PCM Slave Mode and Monaural Mode

6.29.5.16 Dummy Data Number for I²S Slave mode and Stereo Mode

Before I²S / PCM slave starts to send TX data to an external master device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, SLAVE mode (SPIx_I2SCTL[8]), and write TX data to TX FIFO. After slave sends dummy data (data with zero value) to an external master device, slave will send TX FIFO data to an external master device. Table 6.29-6 shows number of dummy data for I²S slave stereo mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	3
16 bits	2
24 bits	2
32 bits	2

Table 6.29-6 Dummy Data Number for I²S Slave Mode and Stereo Mode

6.29.5.17 Dummy Data Number for PCM Slave mode and Stereo Mode

Before I²S / PCM slave starts to send TX data to an external master device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, SLAVE mode (SPIx_I2SCTL[8]), and write TX data to TX FIFO. After slave sends dummy data (data with zero value) to an external master device, slave will send TX FIFO data to an external master device. Table 6.29-7 shows number of dummy data for PCM slave stereo mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	2
16 bits	1
24 bits	1
32 bits	1

Table 6.29-7 Dummy Data Number for PCM Slave Mode and Stereo Mode

6.29.6 Timing Diagram

The active state of slave selection signal can be defined by setting the SSACTPOL (SPIx_SSCTL[2]). The SPI clock which is in idle state can be configured as high or low state by setting the CLKPOL (SPIx_CTL[3]). It also provides the bit length of a transaction word in DWIDTH (SPIx_CTL[12:8]), and transmitting/receiving data from MSB or LSB first in LSB (SPIx_CTL[13]). User can also select which

edge of SPI clock to transmit/receive data in TXNEG/RXNEG (SPIx_CTL[2:1]). Four SPI timing diagrams for master/slave operations and the related settings are shown below.

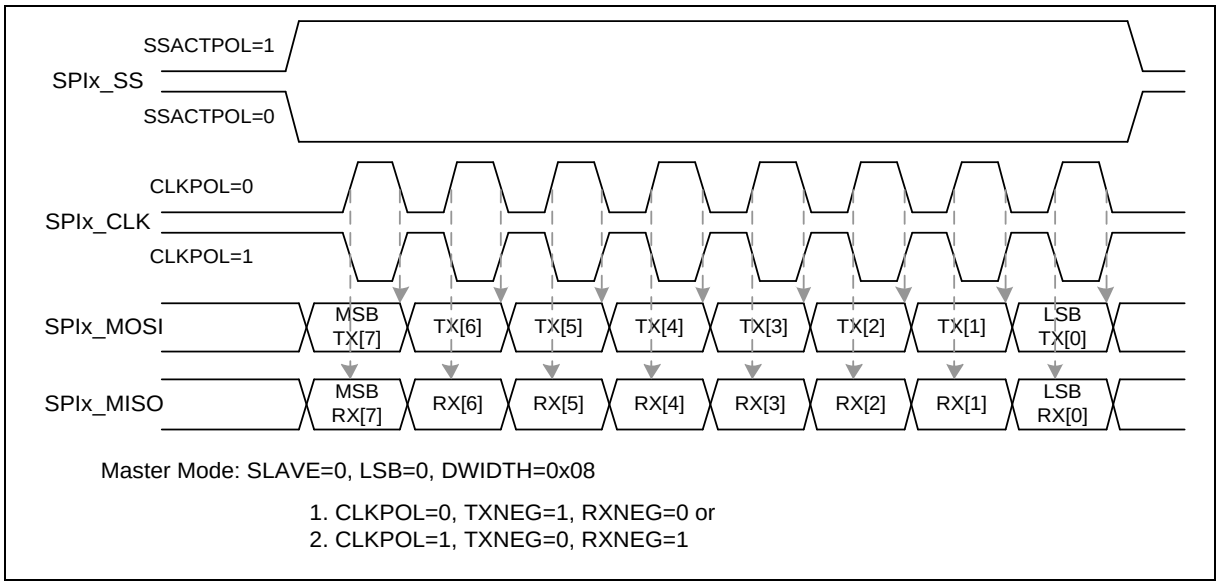


Figure 6.29-25 SPI Timing in Master Mode

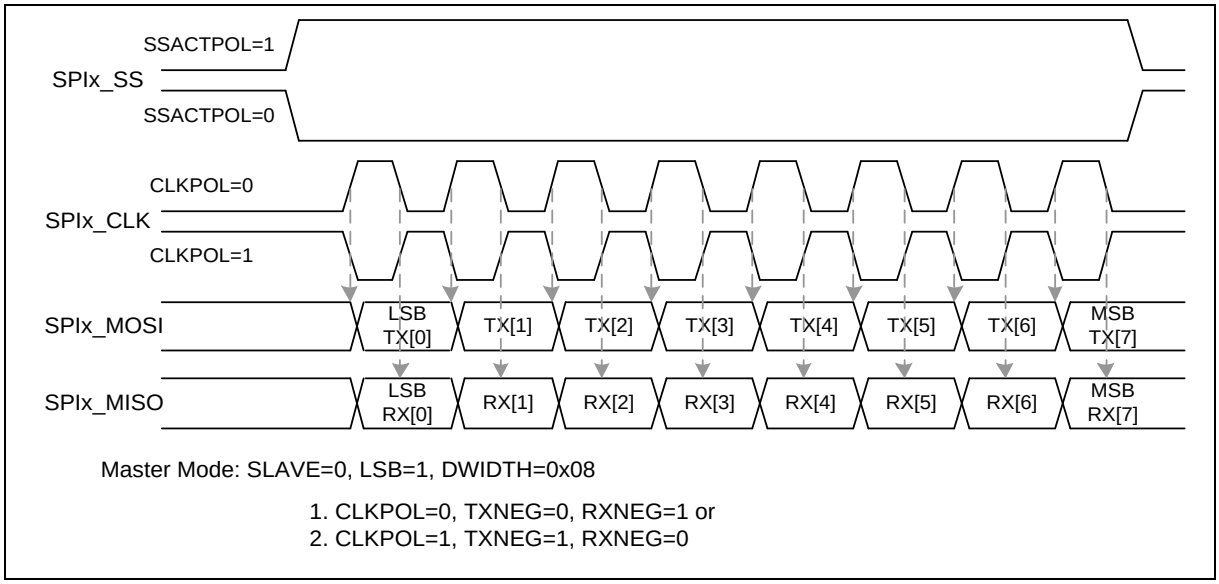


Figure 6.29-26 SPI Timing in Master Mode (Alternate Phase of SPIx_CLK)

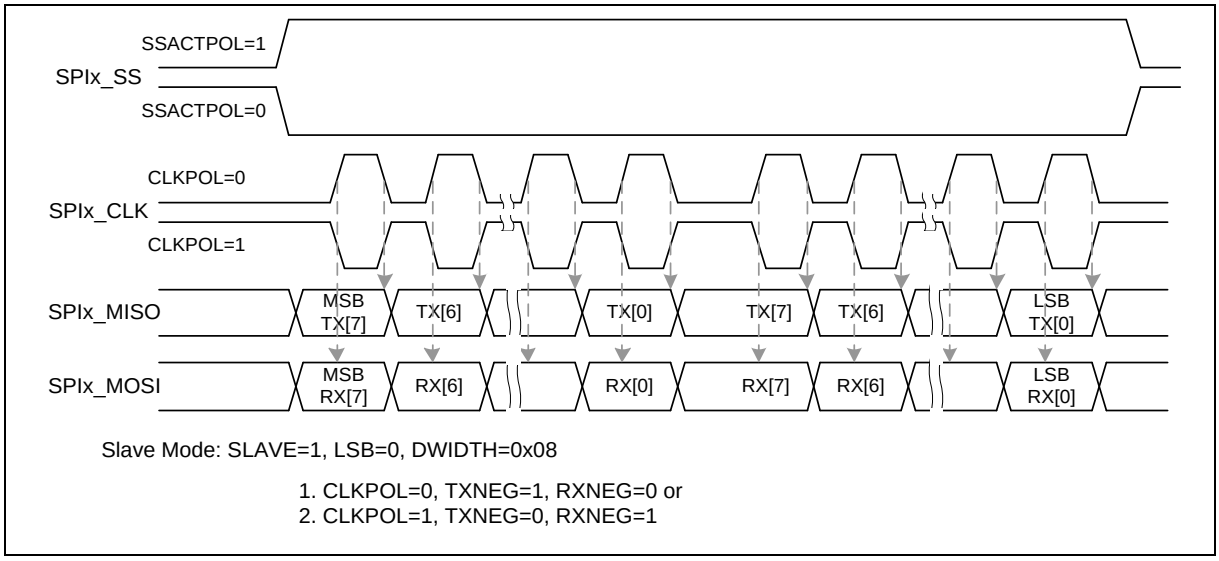


Figure 6.29-27 SPI Timing in Slave Mode

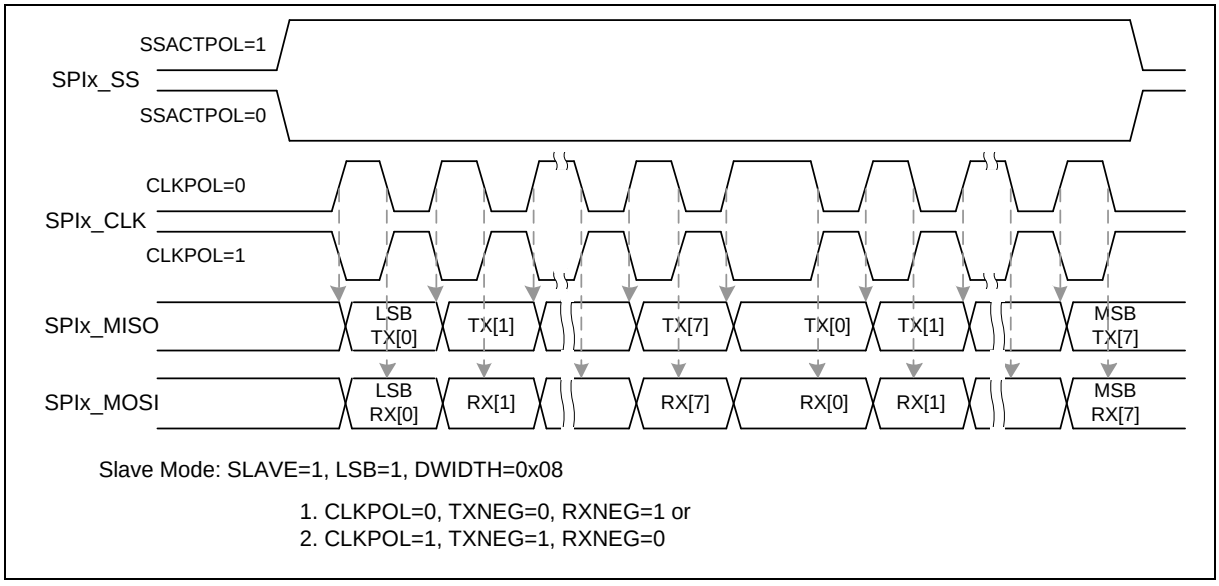


Figure 6.29-28 SPI Timing in Slave Mode (Alternate Phase of SPIx_CLK)

6.29.7 Programming Examples

Example 1:

The SPI controller is set as a full-duplex master to access an off-chip slave device with the following specifications:

- Data bit is latched on positive edge of SPI bus clock.
- Data bit is driven on negative edge of SPI bus clock.
- Data is transferred from MSB first.
- SPI bus clock is idle at low state.
- Only one byte of data to be transmitted/received in a transaction.
- Uses the first SPI slave select pin to connect with an off-chip slave device. The slave

selection signal is active low.

The operation flow is as follows:

1. Set DIVIDER (SPIx_CLKDIV[8:0]) to determine the output frequency of SPI clock.
2. Write the SPIx_SSCTL register a proper value for the related settings of Master mode:
   1) Clear AUTOSS (SPIx_SSCTL[3]) to 0 to disable the Automatic Slave Selection function.
   2) Configure slave selection signal as active low by clearing SSACTPOL (SPIx_SSCTL[2]) to 0.
   3) Enable slave selection signal by setting SS (SPIx_SSCTL[0]) to 1 to activate the off-chip slave device.
3. Write the related settings into the SPIx_CTL register to control the SPI master actions.
   1) Configure this SPI controller as master device by setting SLAVE (SPIx_CTL[18]) to 0.
   2) Force the SPI clock idle state at low by clearing CLKPOL (SPIx_CTL[3]) to 0.
   3) Select data transmitted on negative edge of SPI bus clock by setting TXNEG (SPIx_CTL[2]) to 1.
   4) Select data latched on positive edge of SPI bus clock by clearing RXNEG (SPIx_CTL[1]) to 0.
   5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (SPIx_CTL[12:8] = 0x08).
   6) Set MSB transfer first by clearing LSB (SPIx_CTL[13]) to 0.
4. Set SPIEN (SPIx_CTL[0]) to 1 to enable the data transfer with the SPI interface.
5. If this SPI master attempts to transmit (write) one byte data to the off-chip slave device, write the byte data that will be transmitted into the SPIx_TX register.
6. Waiting for SPI interrupt if the UNITIEN (SPIx_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (SPIx_STATUS[1]).
7. Read out the received one byte data from SPIx_RX register.
8. Go to 5 to continue another data transfer or set SS (SPIx_SSCTL[0]) to 0 to inactivate the off-chip slave device.

Example 2:

The SPI controller is set as a full-duplex slave device and connects with an off-chip master device. The off-chip master device communicates with the on-chip SPI slave controller through the SPI interface with the following specifications:

- Data bit is latched on positive edge of SPI bus clock.
- Data bit is driven on negative edge of SPI bus clock.
- Data is transferred from LSB first.
- SPI bus clock is idle at high state.
- Only one byte of data to be transmitted/received in a transaction.
- Slave selection signal is active high.

The operation flow is as follows:

1. Write the SPIx_SSCTL register a proper value for the related settings of Slave mode.
2. Select high level for the input of slave selection signal by setting SSACTPOL (SPIx_SSCTL[2]) to 1.
3. Write the related settings into the SPIx_CTL register to control this SPI slave actions

1) Set the SPI controller as slave device by setting SLAVE (SPIx_CTL[18]) to 1.
2) Select the SPI clock idle state at high by setting CLKPOL (SPIx_CTL[3]) to 1.
3) Select data transmitted on negative edge of SPI bus clock by setting TXNEG (SPIx_CTL[2]) to 1.
4) Select data latched on positive edge of SPI bus clock by clearing RXNEG (SPIx_CTL[1]) to 0.
5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (SPIx_CTL[12:8] = 0x08).

4. Set LSB transfer first by setting LSB (SPIx_CTL[13]) to 1.
5. Set the SPIEN (SPIx_CTL[0]) to 1. Wait for the slave select trigger input and SPI clock input from the off-chip master device to start the data transfer.
6. If this SPI slave attempts to transmit (be read) one byte data to the off-chip master device, write the byte data that will be transmitted into the SPIx_TX register.
7. If this SPI slave just only attempts to receive (be written) one byte data from the off-chip master device and does not care what data will be transmitted, the SPIx_TX register does not need to be updated by software.
8. Waiting for SPI interrupt if the UNITIEN (SPIx_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (SPIx_STATUS[1]).
9. Read out the received one byte data from SPIx_RX register.
10. Go to 6 to continue another data transfer or stop data transfer.

6.29.8 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
SPI Base Address: **SPIx_BA = 0x4006_1000 + (0x0000_1000 * x)** **x=0, 1, 2, 3**				
SPIx_CTL	SPIx_BA+0x00	R/W	SPI Control Register	0x0000_0034
SPIx_CLKDIV	SPIx_BA+0x04	R/W	SPI Clock Divider Register	0x0000_0000
SPIx_SSCTL	SPIx_BA+0x08	R/W	SPI Slave Select Control Register	0x0000_0000
SPIx_PDMACTL	SPIx_BA+0x0C	R/W	SPI PDMA Control Register	0x0000_0000
SPIx_FIFOCTL	SPIx_BA+0x10	R/W	SPI FIFO Control Register	0x2200_0000
SPIx_STATUS	SPIx_BA+0x14	R/W	SPI Status Register	0x0005_0110
SPIx_STATUS2	SPIx_BA+0x18	R	SPI Status2 Register	0x0000_0000
SPIx_TX	SPIx_BA+0x20	W	SPI Data Transmit Register	0x0000_0000
SPIx_RX	SPIx_BA+0x30	R	SPI Data Receive Register	0x0000_0000
SPIx_I2SCTL	SPIx_BA+0x60	R/W	I^2S Control Register	0x0000_0000
SPIx_I2SCLK	SPIx_BA+0x64	R/W	I^2S Clock Divider Control Register	0x0000_0000
SPIx_I2SSTS	SPIx_BA+0x68	R/W	I^2S Status Register	0x0005_0100

6.29.9 Register Description

SPI Control Register (SPIx_CTL)

Register	Offset	R/W	Description	Reset Value
SPIx_CTL	SPIx_BA+0x00	R/W	SPI Control Register	0x0000_0034

Note: Not supported in I^2S mode.

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**DATDIR**	**REORDER**	**SLAVE**	**UNITIEN**	**Reserved**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXONLY	**HALFDPX**	**LSB**	**DWIDTH**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SUSPITV				**CLKPOL**	**TXNEG**	**RXNEG**	**SPIEN**

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**DATDIR**	**Data Port Direction Control** This bit is used to select the data input/output direction in half-duplex transfer and Dual/Quad transfer 0 = SPI data is input direction. 1 = SPI data is output direction.
[19]	**REORDER**	**Byte Reorder Function Enable Bit** 0 = Byte Reorder function Disabled. 1 = Byte Reorder function Enabled. A byte suspend interval will be inserted among each byte. The period of the byte suspend interval depends on the setting of SUSPITV. **Note:** Byte Reorder function is only available if DWIDTH is defined as 16, 24, and 32 bits.
[18]	**SLAVE**	**Slave Mode Control** 0 = Master mode. 1 = Slave mode.
[17]	**UNITIEN**	**Unit Transfer Interrupt Enable Bit** 0 = SPI unit transfer interrupt Disabled. 1 = SPI unit transfer interrupt Enabled.
[16]	**Reserved**	Reserved.
[15]	**RXONLY**	**Receive-only Mode Enable Bit** This bit field is only available in Master mode. In receive-only mode, SPI Master will generate SPI bus clock continuously for receiving data bit from SPI slave device and assert the BUSY status. 0 = Receive-only mode Disabled. 1 = Receive-only mode Enabled.
[14]	**HALFDPX**	**SPI Half-duplex Transfer Enable Bit** This bit is used to select full-duplex or half-duplex for SPI transfer. The bit field DATDIR

		(SPIx_CTL[20]) can be used to set the data direction in half-duplex transfer. 0 = SPI operates in full-duplex transfer. 1 = SPI operates in half-duplex transfer.
[13]	**LSB**	**Send LSB First** 0 = The MSB, which bit of transmit/receive register depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, bit 0 of the SPI TX register, is sent first to the SPI data output pin, and the first bit received from the SPI data input pin will be put in the LSB position of the RX register (bit 0 of SPI_RX).
[12:8]	**DWIDTH**	**Data Width** This field specifies how many bits can be transmitted/received in one transaction. The minimum bit length is 4 bits and can up to 32 bits. DWIDTH = 0x04 .... 4 bits. DWIDTH = 0x05 .... 5 bits. DWIDTH = 0x06 .... 6 bits. DWIDTH = 0x07 .... 7 bits. DWIDTH = 0x08 .... 8 bits. DWIDTH = 0x09 .... 9 bits. ..... DWIDTH = 0x1F .... 31 bits. DWIDTH = 0x00 .... 32 bits. **Note:** This bit field will decide the depth of TX/RX FIFO configuration in SPI mode**.** Therefore, changing this bit field will clear TX/RX FIFO by hardware automatically.
[7:4]	**SUSPITV**	**Suspend Interval** The four bits provide configurable suspend interval between two successive transmit/receive transaction in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. (SUSPITV[3:0] + 0.5) * period of SPI_CLK clock cycle Example: SUSPITV = 0x0 .... 0.5 SPI_CLK clock cycle. SUSPITV = 0x1 .... 1.5 SPI_CLK clock cycle. ..... SUSPITV = 0xE .... 14.5 SPI_CLK clock cycle. SUSPITV = 0xF .... 15.5 SPI_CLK clock cycle.
[3]	**CLKPOL**	**Clock Polarity** 0 = SPI bus clock is idle low. 1 = SPI bus clock is idle high.
[2]	**TXNEG**	**Transmit on Negative Edge** 0 = Transmitted data output signal is changed on the rising edge of SPI bus clock. 1 = Transmitted data output signal is changed on the falling edge of SPI bus clock.
[1]	**RXNEG**	**Receive on Negative Edge** 0 = Received data input signal is latched on the rising edge of SPI bus clock. 1 = Received data input signal is latched on the falling edge of SPI bus clock.

[0]	**SPIEN**	**SPI Transfer Control Enable Bit** In Master mode, the transfer will start when there is data in the FIFO buffer after this bit is set to 1. In Slave mode, this device is ready to receive data when this bit is set to 1. 0 = Transfer control Disabled. 1 = Transfer control Enabled. **Note:** Before changing the configurations of SPIx_CTL, SPIx_CLKDIV, SPIx_SSCTL and SPIx_FIFOCTL registers, user shall clear the SPIEN (SPIx_CTL[0]) and confirm the SPIENSTS (SPIx_STATUS[15]) is 0.

SPI Clock Divider Register (SPIx_CLKDIV)

Register	Offset	R/W	Description	Reset Value
SPIx_CLKDIV	SPIx_BA+0x04	R/W	SPI Clock Divider Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**DIVIDER**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**DIVIDER**	**Clock Divider** The value in this field is the frequency divider for generating the peripheral clock, f_{spi_eclk}, and the SPI bus clock of SPI Master. The frequency is obtained according to the following equation. $f_{spi_eclk} = \frac{f_{spi_clock_src}}{(DIVIDER + 1)}$ where $f_{spi_clock_src}$ is the peripheral clock source, which is defined in the clock control register, CLK_CLKSEL2 or CLK_CLKSEL3. **Note 1:** Not supported in I²S mode. **Note 2:** The time interval must be larger than or equal to 8 peripheral clock cycles between releasing SPI IP software reset and setting this clock divider register.

Note: DIVIDER should be set carefully because the peripheral clock frequency must be slower than or equal to system frequency.

SPI Slave Select Control Register (SPIx_SSCTL)

Register	Offset	R/W	Description	Reset Value
SPIx_SSCTL	SPIx_BA+0x08	R/W	SPI Slave Select Control Register	0x0000_0000

Note: Not supported in I²S mode.

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVURIEN	SLVBEIEN
7	6	5	4	3	2	1	0
Reserved			SLV3WIRE	AUTOSS	SSACTPOL	Reserved	SS

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable Bit** 0 = Slave select inactive interrupt Disabled. 1 = Slave select inactive interrupt Enabled.
[12]	**SSACTIEN**	**Slave Select Active Interrupt Enable Bit** 0 = Slave select active interrupt Disabled. 1 = Slave select active interrupt Enabled.
[11:10]	**Reserved**	Reserved.
[9]	**SLVURIEN**	**Slave Mode TX Under Run Interrupt Enable Bit** 0 = Slave mode TX under run interrupt Disabled. 1 = Slave mode TX under run interrupt Enabled.
[8]	**SLVBEIEN**	**Slave Mode Bit Count Error Interrupt Enable Bit** 0 = Slave mode bit count error interrupt Disabled. 1 = Slave mode bit count error interrupt Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**SLV3WIRE**	**Slave 3-wire Mode Enable Bit** In Slave 3-wire mode, the SPI controller can work with 3-wire interface including SPIx_CLK, SPIx_MISO and SPIx_MOSI pins. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface. **Note:** The value of this register equals to control register SLAVE (SPIx_I2SCTL[8]) when I²S mode is enabled in **SPI Slave mode Only**.
[3]	**AUTOSS**	**Automatic Slave Selection Function Enable Bit** 0 = Automatic slave selection function Disabled. Slave selection signal will be asserted/de-asserted according to SS (SPIx_SSCTL[0]). 1 = Automatic slave selection function Enabled (Master Only).

[2]	**SSACTPOL**	**Slave Selection Active Polarity** This bit defines the active polarity of slave selection signal (SPIx_SS). 0 = The slave selection signal SPIx_SS is active low. 1 = The slave selection signal SPIx_SS is active high.
[1]	**Reserved**	Reserved.
[0]	**SS**	**Slave Selection Control** If AUTOSS bit is cleared to 0, 0 = set the SPIx_SS line to inactive state. 1 = set the SPIx_SS line to active state. If the AUTOSS bit is set to 1, 0 = Keep the SPIx_SS line at inactive state. 1 = SPIx_SS line will be automatically driven to active state for the duration of data transfer, and will be driven to inactive state for the rest of the time. The active state of SPIx_SS is specified in SSACTPOL (SPIx_SSCTL[2]). **Note:** It is Master Only

SPI PDMA Control Register (SPIx_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SPIx_PDMACTL	SPIx_BA+0x0C	R/W	SPI PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					PDMARST	RXPDMAEN	TXPDMAEN

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the PDMA control logic of the SPI controller. This bit will be automatically cleared to 0.
[1]	**RXPDMAEN**	**Receive PDMA Enable Bit** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[0]	**TXPDMAEN**	**Transmit PDMA Enable Bit** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled. **Note 1:** In SPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, RX PDMA function cannot be enabled prior to TX PDMA function. User can enable TX PDMA function firstly or enable both functions simultaneously. **Note 2:** In SPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, TX PDMA function cannot be disabled prior to RX PDMA function. User can disable RX PDMA function firstly or disable both functions simultaneously.

SPI FIFO Control Register (SPIx_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
SPIx_FIFOCTL	SPIx_BA+0x10	R/W	SPI FIFO Control Register	0x2200_0000

31	30	29	28	27	26	25	24
Reserved	TXTH			Reserved	RXTH		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					SLVBERX	TXFBCLR	RXFBCLR
7	6	5	4	3	2	1	0
TXUFIEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TXTH**	**Transmit FIFO Threshold** If the valid data count of the transmit FIFO buffer is less than or equal to the TXTH setting, the TXTHIF bit will be set to 1, else the TXTHIF bit will be cleared to 0. The MSB of this bit field is only meaningful while SPI mode 4~16 bits of data length.
[27]	**Reserved**	Reserved.
[26:24]	**RXTH**	**Receive FIFO Threshold** If the valid data count of the receive FIFO buffer is larger than the RXTH setting, the RXTHIF bit will be set to 1, else the RXTHIF bit will be cleared to 0. The MSB of this bit field is only meaningful while SPI mode 4~16 bits of data length.
[23:11]	**Reserved**	Reserved.
[10]	**SLVBERX**	**RX FIFO Write Data Enable Bit When Slave Mode Bit Count Error** 0 = Uncompleted RX data will be dropped from RX FIFO when bit count error event happened in SPI Slave mode. 1 = Uncompleted RX data will be written into RX FIFO when bit count error event happened in SPI Slave mode. User can read SLVBENUM (SPIx_STATUS2[29:24]) to know that the effective bit number of uncompleted RX data when SPI slave bit count error happened. **(SPI Slave mode Only)**
[9]	**TXFBCLR**	**Transmit FIFO Buffer Clear** 0 = No effect. 1 = Clear transmit FIFO pointer. The TXFULL bit will be cleared to 0 and the TXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 1 system clock after it is set to 1. **Note:** The TX shift register will not be cleared.
[8]	**RXFBCLR**	**Receive FIFO Buffer Clear** 0 = No effect. 1 = Clear receive FIFO pointer. The RXFULL bit will be cleared to 0 and the RXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 1 system clock after it is set to 1. **Note:** The RX shift register will not be cleared.

[7]	**TXUFIEN**	**TX Underflow Interrupt Enable Bit** When TX underflow event occurs in Slave mode, TXUFIF (SPIx_STATUS[19]) will be set to 1. This bit is used to enable the TX underflow interrupt. 0 = Slave TX underflow interrupt Disabled. 1 = Slave TX underflow interrupt Enabled.
[6]	**TXUFPOL**	**TX Underflow Data Polarity** 0 = The SPI data out is keep 0 if there is TX underflow event in Slave mode. 1 = The SPI data out is keep 1 if there is TX underflow event in Slave mode. **Note 1:** The TX underflow event occurs if there is no any data in TX FIFO when the slave selection signal is active. **Note 2:** This bit should be set as 0 in I^2S mode. **Note 3:** When TX underflow event occurs, SPIx_MISO pin state will be determined by this setting even though TX FIFO is not empty afterward. Data stored in TX FIFO will be sent through SPIx_MISO pin in the next transfer frame.
[5]	**RXOVIEN**	**Receive FIFO Overrun Interrupt Enable Bit** 0 = Receive FIFO overrun interrupt Disabled. 1 = Receive FIFO overrun interrupt Enabled.
[4]	**RXTOIEN**	**Receive Time-out Interrupt Enable Bit** 0 = Receive time-out interrupt Disabled. 1 = Receive time-out interrupt Enabled.
[3]	**TXTHIEN**	**Transmit FIFO Threshold Interrupt Enable Bit** 0 = TX FIFO threshold interrupt Disabled. 1 = TX FIFO threshold interrupt Enabled.
[2]	**RXTHIEN**	**Receive FIFO Threshold Interrupt Enable Bit** 0 = RX FIFO threshold interrupt Disabled. 1 = RX FIFO threshold interrupt Enabled.
[1]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset transmit FIFO pointer and transmit circuit. The TXFULL bit will be cleared to 0 and the TXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (SPIx_STATUS[23]) to check if reset is accomplished or not. **Note:** If TX underflow event occurs in SPI Slave mode, this bit can be used to make SPI return to idle state.
[0]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset receive FIFO pointer and receive circuit. The RXFULL bit will be cleared to 0 and the RXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (SPIx_STATUS[23]) to check if reset is accomplished or not.

SPI Status Register (SPIx_STATUS)

Register	Offset	R/W	Description	Reset Value
SPIx_STATUS	SPIx_BA+0x14	R/W	SPI Status Register	0x0005_0110

Note: Not supported in I²S mode.

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	Reserved	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

Bits	Description	
[31:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**TX or RX Reset Status (Read Only)** 0 = The reset function of TXRST or RXRST is done. 1 = Doing the reset function of TXRST or RXRST. **Note:** Both the reset operations of TXRST and RXRST need 3 system clock cycles + 2 peripheral clock cycles. User can check the status of this bit to monitor the reset function is doing or done.
[22:20]	**Reserved**	Reserved.
[19]	**TXUFIF**	**TX Underflow Interrupt Flag** When the TX underflow event occurs, this bit will be set to 1, the state of data output pin depends on the setting of TXUFPOL. 0 = No effect. 1 = No data in Transmit FIFO and TX shift register when the slave selection signal is active. **Note 1:** This bit will be cleared by writing 1 to it. **Note 2:** If reset slave's transmission circuit when slave selection signal is active, this flag will be set to 1 after 2 peripheral clock cycles + 3 system clock cycles since the reset operation is done.
[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TXTH. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TXTH.
[17]	**TXFULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.

[16]	**TXEMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[15]	**SPIENSTS**	**SPI Enable Status (Read Only)** 0 = SPI controller Disabled. 1 = SPI controller Enabled. **Note:** The SPI peripheral clock is asynchronous with the system clock. In order to make sure the SPI control logic is disabled, this bit indicates the real status of SPI controller.
[14:13]	**Reserved**	Reserved.
[12]	**RXTOIF**	**Receive Time-out Interrupt Flag** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 SPI peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. **Note:** This bit will be cleared by writing 1 to it.
[11]	**RXOVIF**	**Receive FIFO Overrun Interrupt Flag** When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1. 0 = No FIFO is overrun. 1 = Receive FIFO is overrun. **Note:** This bit will be cleared by writing 1 to it.
[10]	**RXTHIF**	**Receive FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the receive FIFO buffer is smaller than or equal to the setting value of RXTH. 1 = The valid data count within the receive FIFO buffer is larger than the setting value of RXTH.
[9]	**RXFULL**	**Receive FIFO Buffer Full Indicator (Read Only)** 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[8]	**RXEMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[7]	**SLVURIF**	**Slave Mode TX Under Run Interrupt Flag** In Slave mode, if TX underflow event occurs and the slave select line goes to inactive state, this interrupt flag will be set to 1. 0 = No Slave TX under run event. 1 = Slave TX under run event occurred. **Note:** This bit will be cleared by writing 1 to it.
[6]	**SLVBEIF**	**Slave Mode Bit Count Error Interrupt Flag** In Slave mode, when the slave select line goes to inactive state, if bit counter is mismatch with DWIDTH, this interrupt flag will be set to 1. 0 = No Slave mode bit count error event. 1 = Slave mode bit count error event occurred. **Note:** If the slave select active but there is no any bus clock input, the SLVBEIF also active when the slave select goes to inactive state. This bit will be cleared by writing 1 to it.
[5]	**Reserved**	Reserved.
[4]	**SSLINE**	**Slave Select Line Bus Status (Read Only)**

		0 = The slave select line status is 0. 1 = The slave select line status is 1. **Note:** This bit is only available in Slave mode. If SSACTPOL (SPIx_SSCTL[2]) is set 0, and the SSLINE is 1, the SPI slave select is in inactive status.
[3]	**SSINAIF**	**Slave Select Inactive Interrupt Flag** 0 = Slave select inactive interrupt was cleared or not occurred. 1 = Slave select inactive interrupt event occurred. **Note:** Only available in Slave mode. This bit will be cleared by writing 1 to it.
[2]	**SSACTIF**	**Slave Select Active Interrupt Flag** 0 = Slave select active interrupt was cleared or not occurred. 1 = Slave select active interrupt event occurred. **Note:** Only available in Slave mode. This bit will be cleared by writing 1 to it.
[1]	**UNITIF**	**Unit Transfer Interrupt Flag** 0 = No transaction has been finished since this bit was cleared to 0. 1 = SPI controller has finished one unit transfer. **Note:** This bit will be cleared by writing 1 to it.
[0]	**BUSY**	**Busy Status (Read Only)** 0 = SPI controller is in idle state. 1 = SPI controller is in busy state. The following lists the bus busy conditions: a. SPIEN (SPIx_CTL[0]) = 1 and TXEMPTY = 0. b. For SPI Master mode, SPIEN (SPIx_CTL[0]) = 1 and TXEMPTY = 1 but the current transaction is not finished yet. c. For SPI Master mode, SPIEN (SPIx_CTL[0]) = 1 and RXONLY = 1. d. For SPI Slave mode, SPIEN (SPIx_CTL[0]) = 1 and there is serial clock input into the SPI core logic when slave select is active. **e.** For SPI Slave mode, SPIEN (SPIx_CTL[0]) = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive. **Note:** By applications, this SPI busy flag should be used with other status registers in SPIx_STATUS such as TXCNT, RXCNT, TXTHIF, TXFULL, TXEMPTY, RXTHIF, RXFULL, RXEMPTY, and UNITIF. Therefore the SPI transfer done events of TX/RX operations can be obtained at correct timing point.

SPI Status2 Register (SPIx_STATUS2)

Register	Offset	R/W	Description	Reset Value
SPIx_STATUS2	SPIx_BA+0x18	R	SPI Status2 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		**SLVBENUM**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**SLVBENUM**	**Effective Bit Number of Uncompleted RX data** This status register indicates that effective bit number of uncompleted RX data when SLVBERX (SPIx_FIFOCTL[10]) is enabled and RX bit count error event happen in SPI Slave mode. This status register will be fixed to 0x0 when SLVBERX (SPIx_FIFOCTL[10]) is disabled. **Note:** This register will be cleared to 0x0 when user writes 0x1 to SLVBEIF (SPIx_STATUS[6]) and **SPI Slave mode Only**
[23:0]	**Reserved**	Reserved.

SPI Data Transmit Register (SPIx_TX)

Register	Offset	R/W	Description	Reset Value
SPIx_TX	SPIx_BA+0x20	W	SPI Data Transmit Register	0x0000_0000

31	30	29	28	27	26	25	24
TX							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TX							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TX							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX							

Bits	Description	
[31:0]	**TX**	**Data Transmit Register** The data transmit registers pass through the transmitted data into the 4-level transmit FIFO buffers. The number of valid bits depends on the setting of DWIDTH (SPIx_CTL[12:8]) in SPI mode or WDWIDTH (SPIx_I2SCTL[5:4]) in I^2S mode. In SPI mode, if DWIDTH is set to 0x08, the bits TX[7:0] will be transmitted. If DWIDTH is set to 0x00, the SPI controller will perform a 32-bit transfer. In I^2S mode, if WDWIDTH (SPIx_I2SCTL[5:4]) is set to 0x2, the data width of audio channel is 24-bit and corresponding to TX[23:0]. If WDWIDTH is set as 0x0, 0x1, or 0x3, all bits of this field are valid and referred to the data arrangement in I^2S mode FIFO operation section **Note:** In Master mode, SPI controller will start to transfer the SPI bus clock after 1 APB clock and 6 peripheral clock cycles after user writes to this register.

SPI Data Receive Register (SPIx_RX)

Register	Offset	R/W	Description	Reset Value
SPIx_RX	SPIx_BA+0x30	R	SPI Data Receive Register	0x0000_0000

31	30	29	28	27	26	25	24
RX							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RX							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RX							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX							

Bits	Description	
[31:0]	**RX**	**Data Receive Register (Read Only)** There are 4-level FIFO buffers in this controller. The data receive register holds the data received from SPI data input pin. If the RXEMPTY (SPIx_STATUS[8] or SPIx_I2SSTS[8]) is not set to 1, the receive FIFO buffers can be accessed through software by reading this register.

I²S Control Register (SPIx_I2SCTL)

Register	Offset	R/W	Description	Reset Value
SPIx_I2SCTL	SPIx_BA+0x60	R/W	I²S Control Register	0x0000_0000

Note: Not supported in SPI mode.

31	30	29	28	27	26	25	24
SLVERRIEN	Reserved	FORMAT		Reserved		LZCIEN	RZCIEN
23	22	21	20	19	18	17	16
RXLCH	Reserved					LZCEN	RZCEN
15	14	13	12	11	10	9	8
MCLKEN	Reserved						SLAVE
7	6	5	4	3	2	1	0
ORDER	MONO	WDWIDTH		MUTE	RXEN	TXEN	I2SEN

Bits	Description	
[31]	**SLVERRIEN**	**Bit Number Error Interrupt Enable Bit for Slave Mode** Interrupt occurs if this bit is set to 1 and bit number error event occurs. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[30]	**Reserved**	Reserved.
[29:28]	**FORMAT**	**Data Format Selection** 00 = I²S data format. 01 = MSB justified data format. 10 = PCM mode A. 11 = PCM mode B.
[27:26]	**Reserved**	Reserved.
[25]	**LZCIEN**	**Left Channel Zero Cross Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and left channel zero cross event occurs. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[24]	**RZCIEN**	**Right Channel Zero Cross Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and right channel zero cross event occurs. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[23]	**RXLCH**	**Receive Left Channel Enable Bit** When monaural format is selected (MONO = 1), I²S controller will receive right channel data if RXLCH is set to 0, and receive left channel data if RXLCH is set to 1. 0 = Receive right channel data in Mono mode. 1 = Receive left channel data in Mono mode.
[22:18]	**Reserved**	Reserved.

[17]	**LZCEN**	**Left Channel Zero Cross Detection Enable Bit** If this bit is set to 1, when left channel data sign bit changes or next shift data bits are all 0 then LZCIF flag in SPIx_I2SSTS register is set to 1. This function is only available in transmit operation. 0 = Left channel zero cross detection Disabled. 1 = Left channel zero cross detection Enabled.
[16]	**RZCEN**	**Right Channel Zero Cross Detection Enable Bit** If this bit is set to 1, when right channel data sign bit change or next shift data bits are all 0 then RZCIF flag in SPIx_I2SSTS register is set to 1. This function is only available in transmit operation. 0 = Right channel zero cross detection Disabled. 1 = Right channel zero cross detection Enabled.
[15]	**MCLKEN**	**Master Clock Enable Bit** If MCLKEN is set to 1, I^2S controller will generate master clock on SPIx_I2SMCLK pin for external audio devices. 0 = Master clock Disabled. 1 = Master clock Enabled.
[14:9]	**Reserved**	Reserved.
[8]	**SLAVE**	**Slave Mode** I^2S can operate as master or slave. For Master mode, I2Sx_BCLK and I2Sx_LRCLK pins are output mode and send bit clock from this chip to audio CODEC chip. In Slave mode, I2Sx_BCLK and I2Sx_LRCLK pins are input mode and I2Sx_BCLK and I2Sx_LRCLK signals are received from outer audio CODEC chip. 0 = Master mode. 1 = Slave mode.
[7]	**ORDER**	**Stereo Data Order in FIFO** 0 = Left channel data at high byte. 1 = Left channel data at low byte.
[6]	**MONO**	**Monaural Data** 0 = Data is stereo format. 1 = Data is monaural format.
[5:4]	**WDWIDTH**	**Word Width** 00 = data size is 8-bit. 01 = data size is 16-bit. 10 = data size is 24-bit. 11 = data size is 32-bit.
[3]	**MUTE**	**Transmit Mute Enable Bit** 0 = Transmit data is shifted from buffer. 1 = Transmit channel zero.
[2]	**RXEN**	**Receive Enable Bit** 0 = Data receive Disabled. 1 = Data receive Enabled.
[1]	**TXEN**	**Transmit Enable Bit** 0 = Data transmit Disabled. 1 = Data transmit Enabled.

[0]	**I2SEN**	**I^2S Controller Enable Bit** 0 = I^2S mode Disabled. 1 = I^2S mode Enabled. **Note 1:** If enabling this bit, I2Sx_BCLK will start to output in Master mode. **Note 2:** Before changing the configurations of SPIx_I2SCTL, SPIx_I2SCLK, and SPIx_FIFOCTL registers, user shall clear the I2SEN (SPIx_I2SCTL[0]) and confirm the I2SENSTS (SPIx_I2SSTS[15]) is 0.

I²S Clock Divider Control Register (SPIx_I2SCLK)

Register	Offset	R/W	Description	Reset Value
SPIx_I2SCLK	SPIx_BA+0x64	R/W	I²S Clock Divider Control Register	0x0000_0000

Note: Not supported in SPI mode.

31	30	29	28	27	26	25	24
Reserved						I2SSLAVE	I2SMODE
23	22	21	20	19	18	17	16
Reserved						BCLKDIV	
15	14	13	12	11	10	9	8
BCLKDIV							
7	6	5	4	3	2	1	0
Reserved	MCLKDIV						

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**I2SSLAVE**	**I²S Clock Divider Number Selection for I²S Slave Mode and I²S Master Mode** User sets I2SSLAVE to set frequency of peripheral clock of I²S Master mode and I²S Slave mode when BCLKDIV (SPIx_I2SCLK[17:8]) is set. I2SSLAVE needs to set before I2SEN (SPIx_I2SCTL[0]) is enabled. 0 = The frequency of peripheral clock is set to I²S Master mode. 1 = The frequency of peripheral clock is set to I²S Slave mode.
[24]	**I2SMODE**	**I²S Clock Divider Number Selection for I²S Mode and SPI Mode** User sets I2SMODE to set frequency of peripheral clock of I²S mode or SPI mode when BCLKDIV (SPIx_I2SCLK[17:8]) or DIVIDER (SPIx_CLKDIV[8:0]) is set. User needs to set I2SMODE before I2SEN (SPIx_I2SCTL[0]) or SPIEN (SPIx_CTL[0]) is enabled. 0 = The frequency of peripheral clock is set to SPI mode. 1 = The frequency of peripheral clock is set to I²S mode.
[23:18]	**Reserved**	Reserved.

[17:8]	**BCLKDIV**	**Bit Clock Divider** The I²S controller will generate bit clock in Master mode. The clock frequency of bit clock , f_{BCLK}, is determined by the following expression: $f_{BCLK} = \frac{f_{i2s_clock_src}}{2 \times (BCLKDIV + 1)}$ where $f_{i2s_clock_src}$ is the frequency of I²S peripheral clock source, which is defined in the clock control register, CLK_CLKSEL2 or CLK_CLKSEL3. In I²S Slave mode, this field is used to define the frequency of peripheral clock and it's determined by $f_{i2s_clock_src} \div \left(\frac{BCLKDIV}{2} + 1\right)$. The peripheral clock frequency in I²S Slave mode must be equal to or faster than 6 times of input bit clock. **Note:** The time interval must be larger than or equal to 8 peripheral clock cycles between releasing SPI IP software reset and setting this clock divider register.
[7]	**Reserved**	Reserved.
[6:0]	**MCLKDIV**	**Master Clock Divider** If MCLKEN is set to 1, I²S controller will generate master clock for external audio devices. The frequency of master clock, f_{MCLK}, is determined by the following expressions: If MCLKDIV >= 1,. $f_{MCLK} = \frac{f_{i2s_clock_src}}{2 \times MCLKDIV}$ If MCLKDIV = 0,. $f_{MCLK} = f_{i2s_clock_src}$ where $f_{i2s_clock_src}$ is the frequency of I²S peripheral clock source, which is defined in the clock control register, CLK_CLKSEL2 or CLK_CLKSEL3. In general, the master clock rate is 256 times sampling clock rate.

Note: BCLKDIV should be set carefully because the peripheral clock frequency must be slower than or equal to system frequency.

I²S Status Register (SPIx_I2SSTS)

Register	Offset	R/W	Description	Reset Value
SPIx_I2SSTS	SPIx_BA+0x68	R/W	I²S Status Register	0x0005_0100

Note: Not supported in SPI mode.

31	30	29	28	27	26	25	24
Reserved	TXCNT			Reserved	RXCNT		
23	22	21	20	19	18	17	16
TXRXRST	SLVERRIF	LZCIF	RZCIF	TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
I2SENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
Reserved			RIGHT	Reserved			

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27]	**Reserved**	Reserved.
[26:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**TX or RX Reset Status (Read Only)** 0 = The reset function of TXRST or RXRST is done. 1 = Doing the reset function of TXRST or RXRST. **Note:** Both the reset operations of TXRST and RXRST need 3 system clock cycles + 2 peripheral clock cycles. User can check the status of this bit to monitor the reset function is doing or done.
[22]	**SLVERRIF**	**Bit Number Error Interrupt Flag for Slave Mode** 0 = No bit number error event occurred. 1 = Bit number error event occurred. **Note:** This bit will be cleared by writing 1 to it.
[21]	**LZCIF**	**Left Channel Zero Cross Interrupt Flag** 0 = No zero cross event occurred on left channel. 1 = Zero cross event occurred on left channel. **Note:** This bit will be cleared by writing 1 to it.
[20]	**RZCIF**	**Right Channel Zero Cross Interrupt Flag** 0 = No zero cross event occurred on right channel. 1 = Zero cross event occurred on right channel. **Note:** This bit will be cleared by writing 1 to it.
[19]	**TXUFIF**	**Transmit FIFO Underflow Interrupt Flag** When the transmit FIFO buffer is empty and there is no datum written into the FIFO buffer, if there is more bus clock input, this bit will be set to 1.

		Note: This bit will be cleared by writing 1 to it.
[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TXTH. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TXTH. **Note:** If TXTHIEN = 1 and TXTHIF = 1, the SPI/I^{2}S controller will generate a SPI interrupt request.
[17]	**TXFULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.
[16]	**TXEMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[15]	**I2SENSTS**	**I^{2}S Enable Status (Read Only)** 0 = The SPI/I^{2}S control logic is disabled. 1 = The SPI/I^{2}S control logic is enabled. **Note:** The SPI peripheral clock is asynchronous with the system clock. In order to make sure the SPI/I^{2}S control logic is disabled, this bit indicates the real status of SPI/I^{2}S control logic for user.
[14:13]	**Reserved**	Reserved.
[12]	**RXTOIF**	**Receive Time-out Interrupt Flag** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 SPI peripheral clock period in Master mode or over 576 SPI peripheral clock period in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. **Note:** This bit will be cleared by writing 1 to it.
[11]	**RXOVIF**	**Receive FIFO Overrun Interrupt Flag** When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1. **Note:** This bit will be cleared by writing 1 to it.
[10]	**RXTHIF**	**Receive FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the receive FIFO buffer is smaller than or equal to the setting value of RXTH. 1 = The valid data count within the receive FIFO buffer is larger than the setting value of RXTH. **Note:** If RXTHIEN = 1 and RXTHIF = 1, the SPI/I^{2}S controller will generate a SPI interrupt request.
[9]	**RXFULL**	**Receive FIFO Buffer Full Indicator (Read Only)** 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[8]	**RXEMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[7:5]	**Reserved**	Reserved.
[4]	**RIGHT**	**Right Channel (Read Only)** This bit indicates the current transmit data is belong to which channel.

		0 = Left channel. 1 = Right channel.
[3:0]	**Reserved**	Reserved.

6.30 Low Power Serial Peripheral Interface (LPSPI)

6.30.1 Overview

The Low Power Serial Peripheral Interface (LPSPI) applies to synchronous serial data communication and allows full duplex transfer. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The chip contains one set of LPSPI controller performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. The LPSPI controller can be configured as a master device and supports the PDMA function to access the data buffer, or configured as a slave device and monitor the signal level change of SS to wake up system.

6.30.2 Features

- SPI Mode
  - One set of LPSPI controller
  - Supports Master or Slave mode operation
  - Master mode up to 12 MHz (when chip works at V_{DD} = 2.7~3.6V)
  - Slave mode up to 6 MHz when SPI master device supports adjustment function of RX data sampling clock (when chip works at V_{DD} = 2.7~3.6V)
  - Slave mode up to 6 MHz when SPI master device does not support adjustment function of RX data sampling clock (when chip works at V_{DD} = 2.7~3.6V)
  - Configurable bit length of a transaction word from 4 to 32-bit
  - Provides separate 4-level depth transmit and receive FIFO buffers
  - Supports MSB first or LSB first transfer sequence
  - Supports Byte Reorder function
  - Supports Byte or Word Suspend mode
  - Supports PDMA transfer
  - Supports 3-Wire, no slave selection signal, bi-direction interface
  - Supports one data channel half-duplex transfer
  - Supports receive-only mode
  - Supports automatic operation mode

6.30.3 Block Diagram

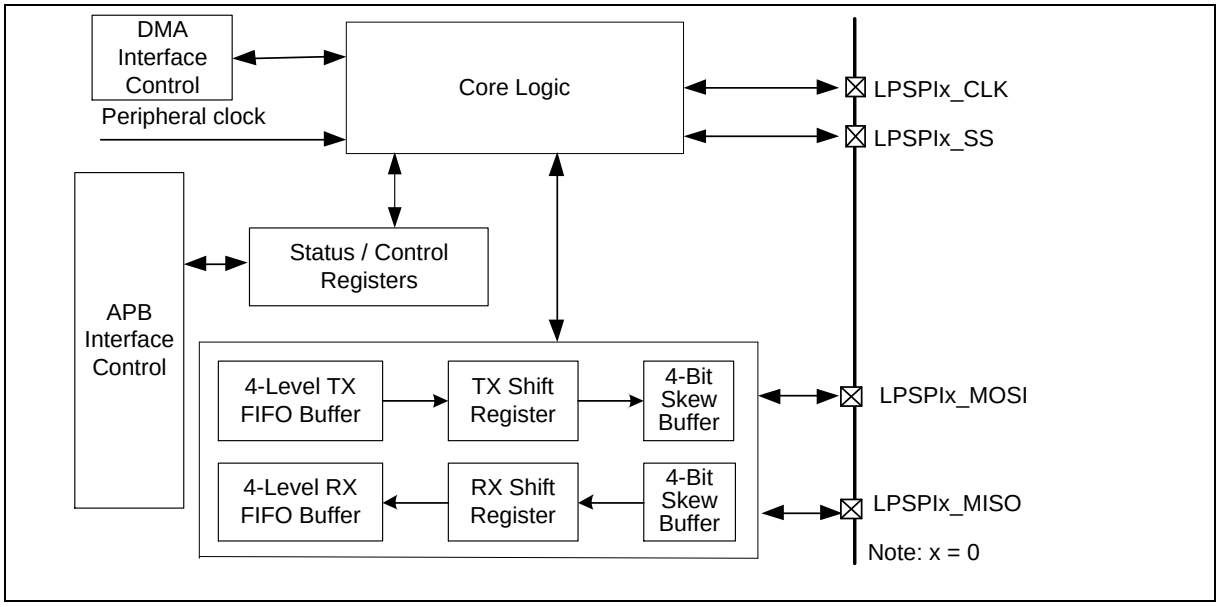


Figure 6.30-1 LPSPI Block Diagram

TX FIFO Buffer

The transmit FIFO buffer is a 4-level depth, 32-bit wide, first-in, first-out register buffer. The data can be written to the transmit FIFO buffer in advance through software by writing the LPSPI_TX register. In SPI mode, the transmit FIFO will be configured as 8-level while data length is set as 4~16 bits.

RX FIFO Buffer

The receive FIFO buffer is also a 4-level depth, 32-bit wide, first-in, first-out register buffer. The receive control logic will store the receive data to this buffer. The FIFO buffer data can be read from LPSPI_RX register by software. In SPI mode, the receive FIFO will be configured as 8-level while data length is set as 4~16 bits.

TX Shift Register

The transmit shift register is a 32-bit wide register buffer. The transmit data is loaded from the TX FIFO buffer and shifted out bit-by-bit to the skew buffer.

RX Shift Register

The receive shift register is also a 32-bit wide register buffer. The receive data is shift in bit-by-bit from the skew buffer and is loaded into RX FIFO buffer when a transaction done.

Skew Buffer

The skew buffer is a 4-level 1-bit buffer. There are two skew buffers in transmitting and received side. In received side, it is used to shift bits into RX shift register from SPI bus. In transmitting side, it is used to shift bits into SPI bus from TX shift register.

6.30.4 Basic Configuration

6.30.4.1 LPSPI0 Basic Configuration

- Clock source Configuration
  - Select the source of LPSPI0 peripheral clock on LPSPI0SEL (LPSCC_CLKSEL0[3:2]).
  - Enable LPSPI0 peripheral clock in LPSPI0CKEN (LPSCC_CLKEN0[17]).
  - Select HCLK1 clock source (CLK_CLKSEL0[7:6]).

- Set APB2 Clock Divider & HCLK1DIV (LPSCC_CLKDIV0[6:4] & LPSCC_CLKDIV0[3:0]).
- Keep LPSPI0 peripheral clock while in NPDx mode in LPSPI0KEEP (LPSCC_CLKKEEP0[17]).

- Reset Configuration
  - Reset LPSPI0 controller in LPSPI0RST (LPSCC_IPRST0[17]).
-

LPSPI (LPSPI0) Interface Controller Pin description is shown as follows:

Pin	SPI Mode
LPSPI_SS	SPI slave selection pin
LPSPI_CLK	SPI clock pin
LPSPI_MISO	SPI master input or slave output pin
LPSPI_MOSI	SPI master output or slave input pin

Table 6.30-1 LPSPI Interface Controller Pin Description (LPSPI0)

6.30.5 Functional Description

6.30.5.1 Terminology

SPI Peripheral Clock and SPI Bus Clock

The LPSPI controller needs the peripheral clock to drive the SPI logic unit to perform the data transfer. The peripheral clock rate is determined by the settings of clock divisor (LPSPI_CLKDIV) and the clock source which can be HIRC or MIRC. LPSPISEL of LPSCC_CLKSEL0 register determine the clock source of the peripheral clock. The DIVIDER (LPSPI_CLKDIV[8:0]) setting determines the divisor of the clock rate calculation.

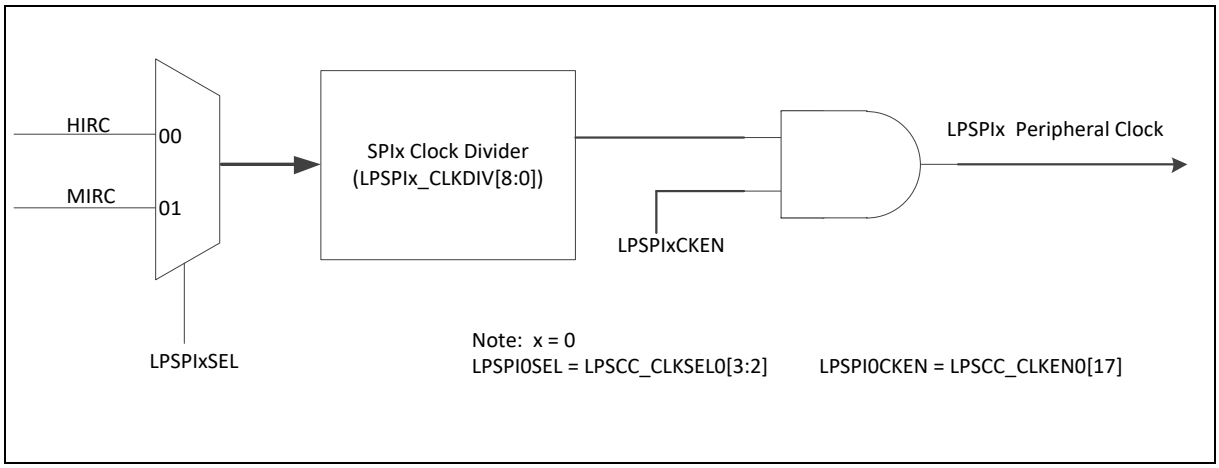


Figure 6.30-2 LPSPI Peripheral Clock

In Master mode, the frequency of the SPI bus clock is equal to the peripheral clock rate. In general, the SPI bus clock is denoted as SPI clock. In Slave mode, the SPI bus clock is provided by a master device. The frequency of SPI peripheral clock cannot be faster than the system clock rate regardless of Master or Slave mode. If the clock source of peripheral clock is not system clock, the frequency of SPI peripheral clock shall be slower than the system clock frequency regardless of Master or Slave mode.

Master/Slave Mode

The LPSPI controllers can be set as Master or Slave mode by setting the SLAVE (LPSPI_CTL[18]) to communicate with the off-chip SPI slave or master device. The HALFDPX (LPSPI_CTL[14]) can be used to select the full-duplex or half-duplex in SPI transmission. The application block diagrams in Master and Slave mode are shown below.

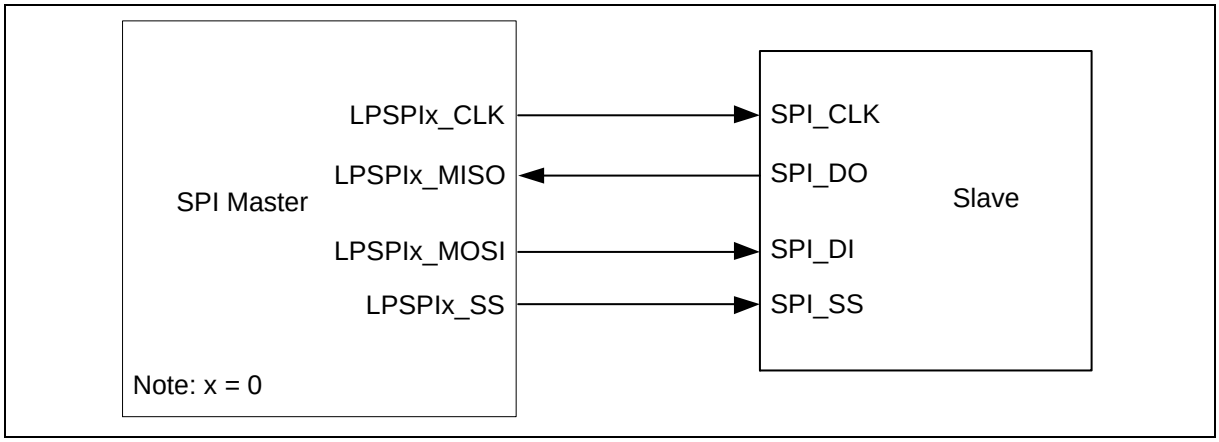


Figure 6.30-3 LPSPI Full-Duplex Master Mode Application Block Diagram

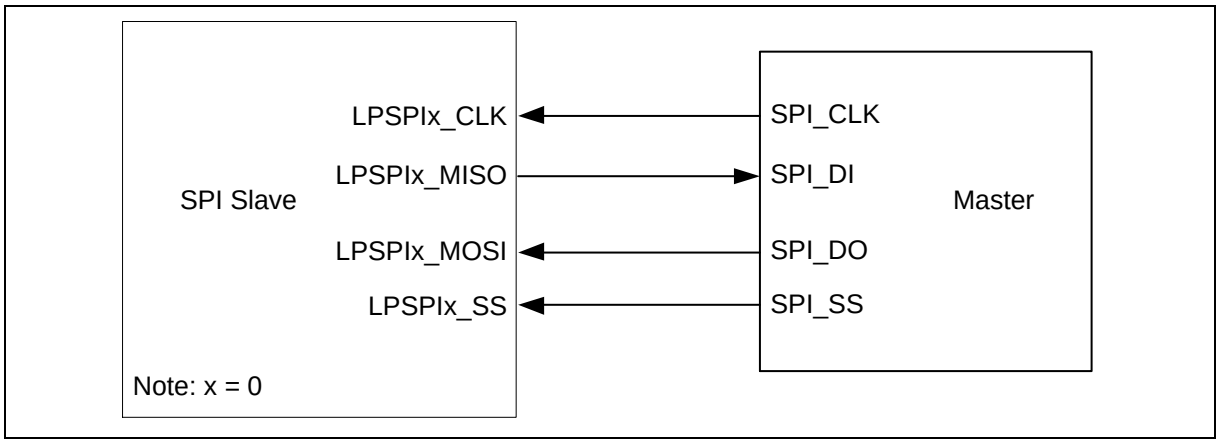


Figure 6.30-4 LPSPI Full-Duplex Slave Mode Application Block Diagram

Slave Selection

In Master mode, the LPSPI controller can drive off-chip slave device through the slave select output pin LPSPI_SS. In Slave mode, the off-chip master device drives the slave selection signal from the LPSPI_SS input port to this LPSPI controller. The duration between the slave select active edge and the first SPI clock input shall over 3 SPI peripheral clock cycles of slave.

In Master/Slave mode, the active state of slave selection signal can be programmed to low or high active in SSACTPOL (LPSPI_SSCTL[2]). The selection of slave select conditions depends on what type of device is connected. In Slave mode, to recognize the inactive state of the slave selection signal, the inactive period of the slave selection signal must be larger than or equal to 3 peripheral clock cycles between two successive transactions.

In Slave mode, the LPSPI_MISO signal type is controlled by the LPSPI_SS pin. If the LPSPI_SS pin is inactive state, the LPSPI_MISO is set to tri-state to avoid interference with other Slave devices. If the LPSPI_SS pin is active state, the LPSPI_MISO is set to output mode for data transfer.

In Slave mode, the LPSPI controller can monitor the SS active state changing to wake up system by setting SSWKEN (LPSPI_AUTOCTL[7]) and SPIEN (LPSPI_CTL[0]) while chip is in NPDx mode.

Timing Condition

The CLKPOL (LPSPI_CTL[3]) defines the SPI clock idle state. If CLKPOL = 1, the output SPI clock is idle at high state; if CLKPOL = 0, it is idle at low state.

TXNEG (LPSPI_CTL[2]) defines the data transmitted out either on negative edge or on positive edge of SPI clock. RXNEG (LPSPI_CTL[1]) defines the data received either on negative edge or on positive edge of SPI clock.

Note: The settings of TXNEG and RXNEG are mutually exclusive. In other words, do not transmit and receive data at the same clock edge.

Transmit/Receive Bit Length

The bit length of a transaction word is defined in DWIDTH (LPSPI_CTL[12:8]) and can be configured up to 32-bit length in a transaction word for transmitting and receiving.

When LPSPI controller finishes a transaction, i.e. receives or transmits a specific count of bits defined in DWIDTH (LPSPI_CTL[12:8]), the unit transfer interrupt flag UNITIF (LPSPI_STATUS[1]) will be set to 1.

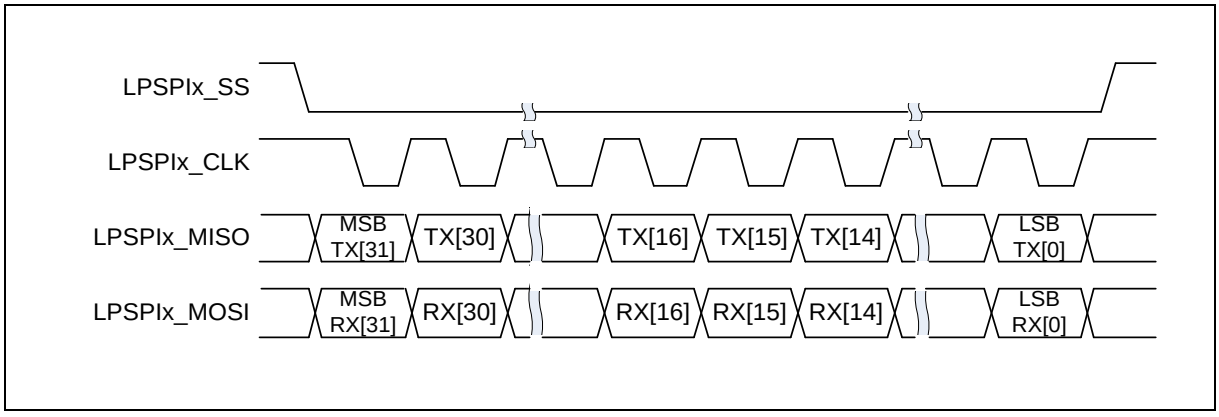


Figure 6.30-5 32-bit in One Transaction

LSB/MSB First

LSB (LPSPI_CTL[13]) defines the bit transfer sequence in a transaction. If the LSB (LPSPI_CTL[13]) is set to 1, the transfer sequence is LSB first. The bit 0 will be transferred firstly. If the LSB (LPSPI_CTL[13]) is cleared to 0, the transfer sequence is MSB first.

Suspend Interval

SUSPITV (LPSPI_CTL[7:4]) provides a configurable suspend interval, 0.5 ~ 15.5 SPI clock periods, between two successive transaction words in Master mode. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SUSPITV is 0x3 (3.5 SPI clock cycles).

6.30.5.2 Automatic Slave Selection

In Master mode, if AUTOSS (LPSPI_SSCTL[3]) is set, the slave selection signal will be generated automatically and output to the LPSPI_SS pin according to whether SS (LPSPI_SSCTL[0]) is enabled or not. The slave selection signal will be set to active state by the LPSPI controller when the SPI data transfer is started by writing to FIFO. It will be set to inactive state when SPI bus is idle. If SPI bus is not idle, i.e. TX FIFO, TX shift register or TX skew buffer is not empty, the slave selection signal will be set to inactive state between transactions if the value of SUSPITV (LPSPI_CTL[7:4]) is greater than or equal to 3.

In Master mode, if the value of SUSPITV is less than 3 and the AUTOSS is set as 1, the slave selection signal will be kept at active state between two successive transactions.

If the AUTOSS bit is cleared, the slave selection output signal will be determined by the SS setting. The active state of the slave selection output signal is specified in SSACTPOL (LPSPI_SSCTL[2]).

The duration between the slave selection signal active edge and the first SPI bus clock edge is 1 SPI bus clock cycle and the duration between the last SPI bus clock and the slave selection signal inactive edge is 1.5 SPI bus clock cycle.

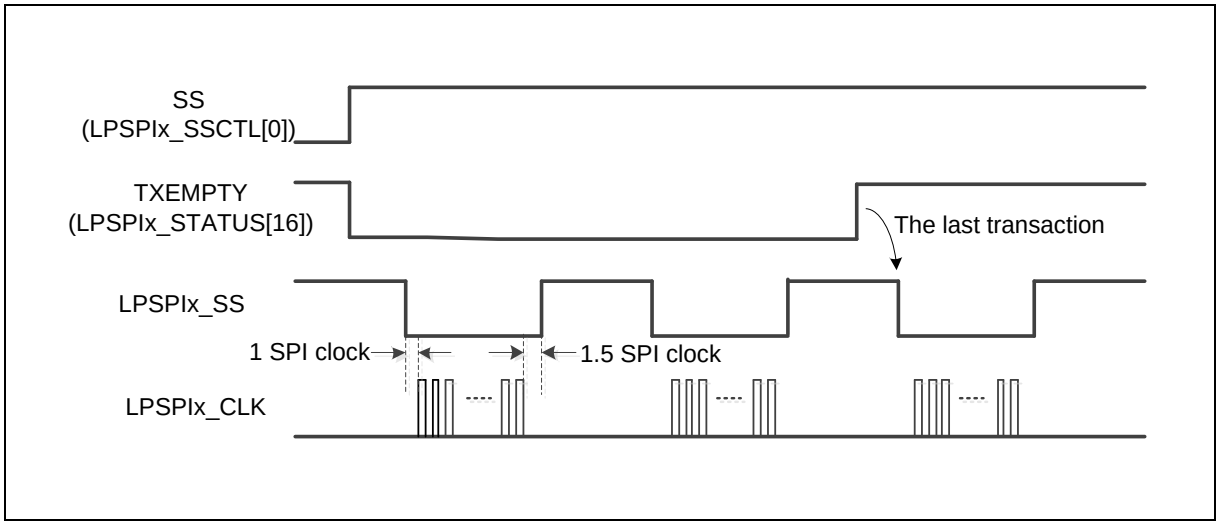


Figure 6.30-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2)

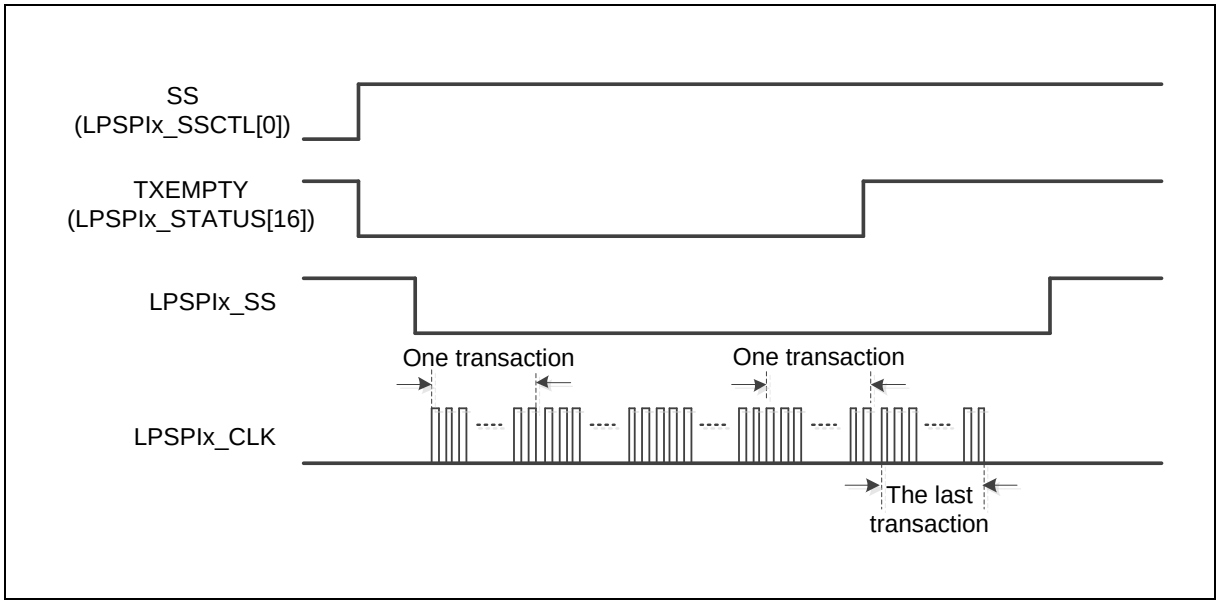


Figure 6.30-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3)

6.30.5.3 Byte Reorder and Suspend Function

When the transfer is set as MSB first (LSB = 0) and the REORDER (LPSPI_CTL[19]) is set to 1, the data stored in the TX buffer and RX buffer will be rearranged in the order as [Byte0, Byte1, Byte2, Byte3] in 32-bit transfer (DWIDTH = 0). The sequence of transmitted/received data will be Byte0, Byte1, Byte2, and then Byte3. If the DWIDTH is set as 24-bit transfer mode, the data in TX buffer and RX buffer will be rearranged as [unknown byte, Byte0, Byte1, Byte2]. The LPSPI controller will transmit/receive data with the sequence of Byte0, Byte1 and then Byte2. Each byte will be transmitted/received with MSB first. The rule of 16-bit mode is the same as above. Byte Reorder function is only available when DWIDTH is configured as 16, 24, and 32 bits.

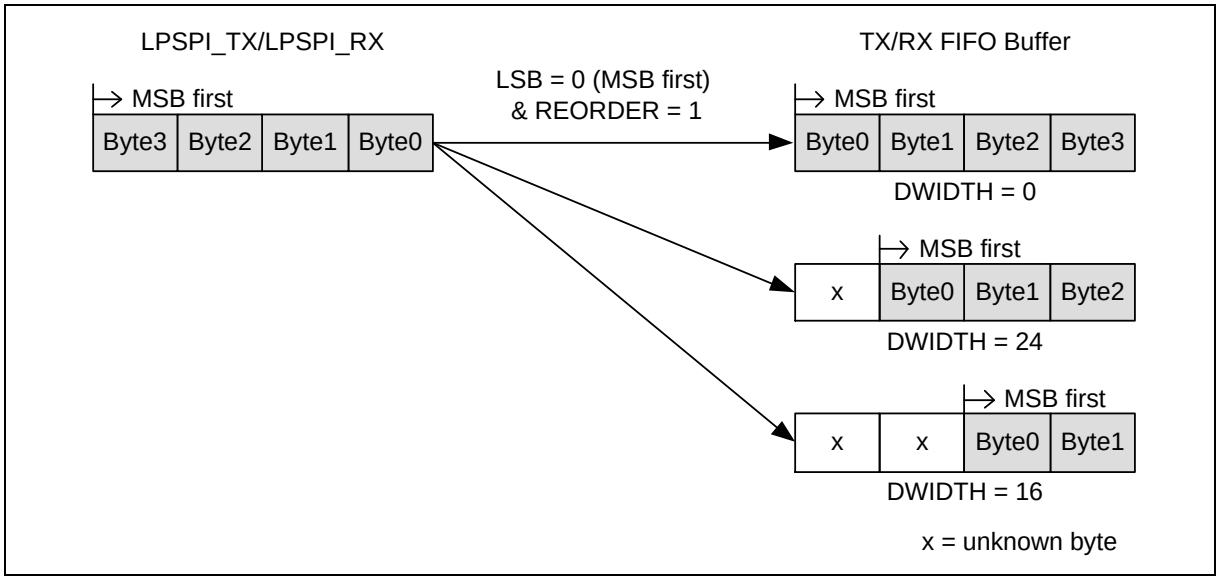


Figure 6.30-8 Byte Reorder Function

In Master mode, if REORDER (LPSPI_CTL[19]) is set to 1, a suspend interval of 0.5 ~ 15.5 SPI clock periods will be inserted by hardware between two successive bytes in a transaction word. The suspend interval is configured in SUSPITV (LPSPI_CTL[7:4]).

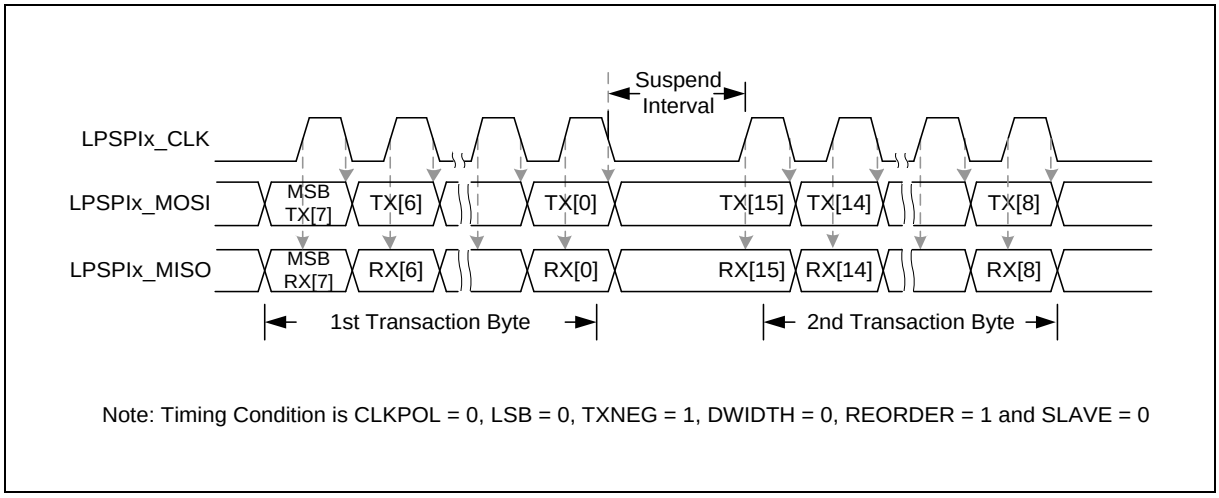


Figure 6.30-9 Timing Waveform for Byte Suspend

6.30.5.4 Half-Duplex Communication

The LPSPI controller can communicate in half-duplex mode by setting HALFDPX (LPSPI_CTL[14]) bit. In half-duplex mode, there is only one data line for receiving or transmitting data direction which is defined by DATDIR (LPSPI_CTL[20]). In half-duplex configuration, the LPSPI_MISO pin is free for other applications and it can be configured as GPIO. Enabling or disabling the control bit HALFDPX (LPSPI_CTL[14]) will produce TXFBCLR (LPSPI_FIFOCTL[9]) and RXFBCLR (LPSPI_FIFOCTL[8]) at the same time automatically.

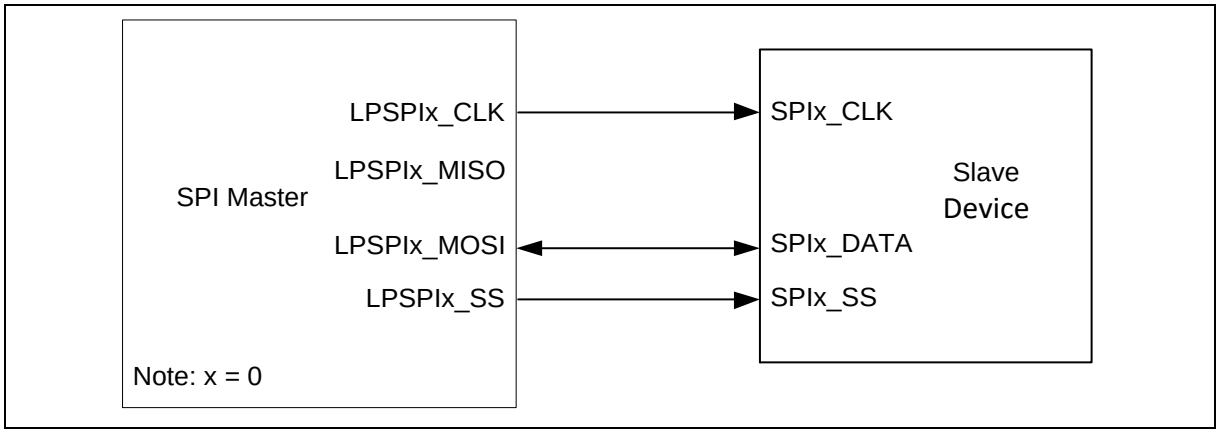


Figure 6.30-10 SPI Half-Duplex Master Mode Application Block Diagram

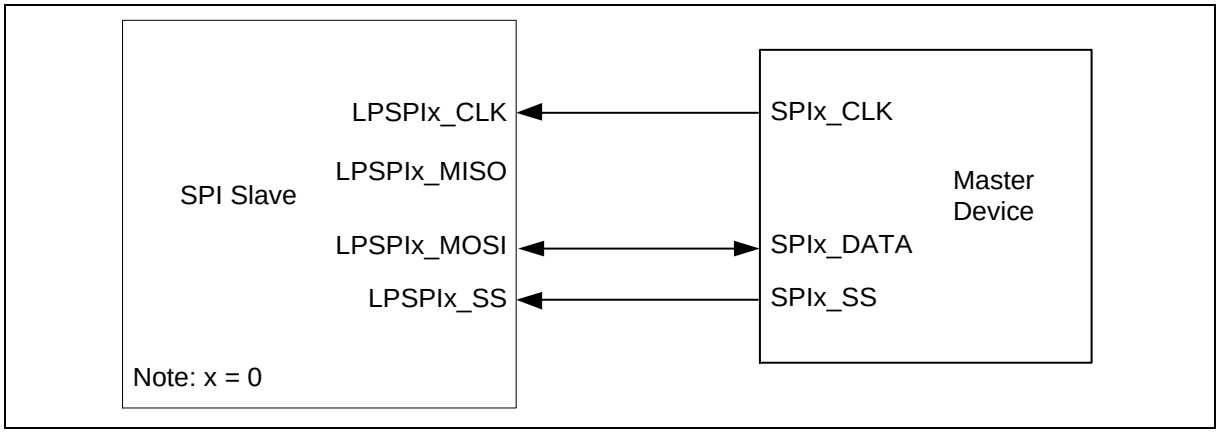


Figure 6.30-11 SPI Half-Duplex Slave Mode Application Block Diagram

6.30.5.5 Receive-Only Mode

In SPI Master device, it can communicate in receive-only mode by setting RXONLY (LPSPI_CTL[15]). In this configuration, the SPI Master device will generate SPI bus clock continuously as long as the receive-only mode is enabled for receiving data bit from SPI slave device. If AUTOSS (LPSPI_SSCTL[3]) is enabled in receive-only mode, SPI Master will keep activating the slave select signal.

The remaining LPSPI_MOSI pin of SPI Master device is not used for communication and can be configured as GPIO. The status BUSY (LPSPI_STATUS[0]) will be asserted in receive-only mode due to the generation of SPI bus clock. Entering this mode will produce the TXFBCLR (LPSPI_FIFOCTL[9]) and RXFBCLR (LPSPI_FIFOCTL[8]) at the same time automatically. When user enables this mode, the output SPI bus clock will be sent out after 6 peripheral clock cycles. In this mode, the data which has been written into transmit FIFO will be loaded into transmit shift register and sent out.

When user sets RXONLY (LPSPI_CTL[15]) to enable, SPI RX data with data bit width of DWIDTH (LPSPI_CTL[12:8]) will be received into RX FIFO and SPI clock will be sent to SPI slave device until RX FIFO is full.

For data bit width of 4~16 bits, the SPI master will send SPI output clock to SPI slave and receive RX data when RX FIFO counter RXCNT (LPSPI_STATUS[27:24]) is less than or equal to 6.

For data bit width of 17~32 bits, the SPI master will send SPI output clock to SPI slave and receive RX data when RX FIFO counter RXCNT (LPSPI_STATUS[27:24]) is less than or equal to 2.

6.30.5.6 Slave 3-Wire Mode

When SLV3WIRE (LPSPI_SSCTL[4]) is set by software to enable the Slave 3-Wire mode, the LPSPI controller can work with no slave selection signal in Slave mode. The SLV3WIRE (LPSPI_SSCTL[4]) only takes effect in Slave mode. Only three pins, LPSPI_CLK, LPSPI_MISO, and LPSPI_MOSI, are required to communicate with a SPI master. The LPSPI_SS pin can be configured as a GPIO. When the SLV3WIRE (LPSPI_SSCTL[4]) is set to 1, the SPI slave will be ready to transmit/receive data after the SPIEN (LPSPI_CTL[0]) is set to 1.

6.30.5.7 PDMA Transfer Function

LPSPI supports PDMA transfer function.

When TXPDMAEN (LPSPI_PDMACTL[0]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (LPSPI_PDMACTL[1]) is set to 1, the controller will start the PDMA reception process. LPSPI will issue request to PDMA controller automatically when there is data in the RX FIFO buffer.

Note: LPSPI supports single request PDMA (Read/Write) only; burst request PDMA is not supported.

Note: LPSPI controller can only co-work with LPPDMA but not normal PDMA.

6.30.5.8 FIFO Buffer Operation

LPSPI is equipped with four 32-bit wide transmit and receive FIFO buffers. The data stored in the transmit FIFO buffer will be read and sent out by the transmission control logic. If the transmit FIFO buffer is full, the TXFULL (LPSPI_STATUS[17]) will be set to 1. When the SPI transmission logic unit draws out the last datum of the transmit FIFO buffer, so that the transmit FIFO buffer is empty, the TXEMPTY (LPSPI_STATUS[16]) will be set to 1. Note that the TXEMPTY (LPSPI_STATUS[16]) flag is set to 1 while the last transaction is still in progress. In Master mode, the BUSY (LPSPI_STATUS[0]) is set to 1 when the FIFO buffer is written any data or there is any transaction on the SPI bus. (e.g. the slave selection signal is active and LPSPI is receiving data in Slave mode). It will set to 0 when the transmit FIFO is empty and the current transaction has done. Thus, the status of BUSY (LPSPI_STATUS[0]) should be checked by software to make sure whether the SPI is in idle or not.

The receive control logic will store the SPI input data into the receive FIFO buffer. There are FIFO related status bits, like RXEMPTY (LPSPI_STATUS[8]) and RXFULL (LPSPI_STATUS[9]), to indicate the current status of RX FIFO buffer.

The transmitting and receiving threshold can be configured by setting TXTH (LPSPI_FIFOCTL[30:28]) and RXTH (LPSPI_FIFOCTL[26:24]). When the count of valid data stored in transmit FIFO buffer is less than or equal to TXTH (LPSPI_FIFOCTL[30:28]) setting, TXTHIF (LPSPI_STATUS[18]) will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RXTH (LPSPI_FIFOCTL[26:24]) setting, RXTHIF (LPSPI_STATUS[10]) will be set to 1.

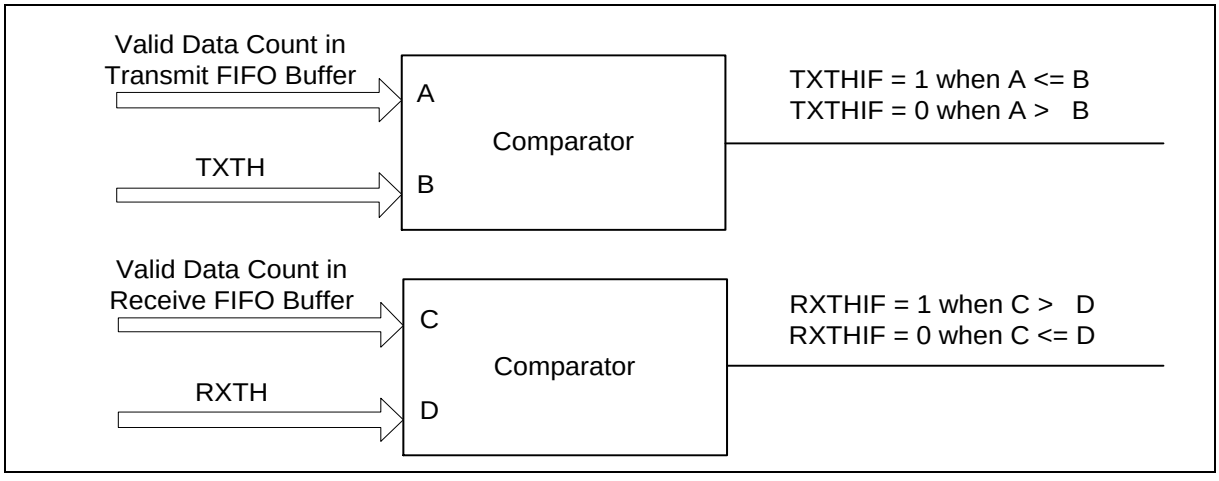


Figure 6.30-12 FIFO Threshold Comparator

In Master mode, when the first datum is written to the LPSPI_TX register, the TXEMPTY flag (LPSPI_STATUS[16]) will be cleared to 0. The transmission will start after 1 PCLK clock cycles and 6 peripheral clock cycles. User can write the next data into LPSPI_TX register immediately. LPSPI will insert a suspend interval between two successive transactions. The period of suspend interval is decided by the setting of SUSPITV (LPSPI_CTL[7:4]). If the SUSPITV (LPSPI_CTL[7:4]) equals 0, LPSPI can perform continuous transfer. User can write data into LPSPI_TX register as long as the TXFULL (LPSPI_STATUS[17]) is 0.

The example 1 of indicates the updated condition of TXEMPTY (LPSPI_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer for 4~16 bits of data length. The TXEMPTY (LPSPI_STATUS[16]) is set to 0 when the Data0 is written into the FIFO buffer. The Data0 will be loaded into the shift register by the core logic and the TXEMPTY (LPSPI_STATUS[16]) will be 1. The Data0 in shift register will be shifted into skew buffer by bit for transmission until the transfer is done.

The Example 2 of Figure 6.30-13 indicates the updated condition of TXFULL (LPSPI_STATUS[17]) when there are 8 data in the FIFO buffer and the next data of Data9 is not written into the FIFO buffer when the TXFULL = 1.

The example 1 of indicates the updated condition of TXEMPTY (LPSPI_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer for 17~32 bits of data length.

The Example 2 of Figure 6.30-14 indicates the updated condition of TXFULL (LPSPI_STATUS[17]) when there are 4 data in the FIFO buffer and the next data of Data5 is not written into the FIFO buffer when the TXFULL = 1.

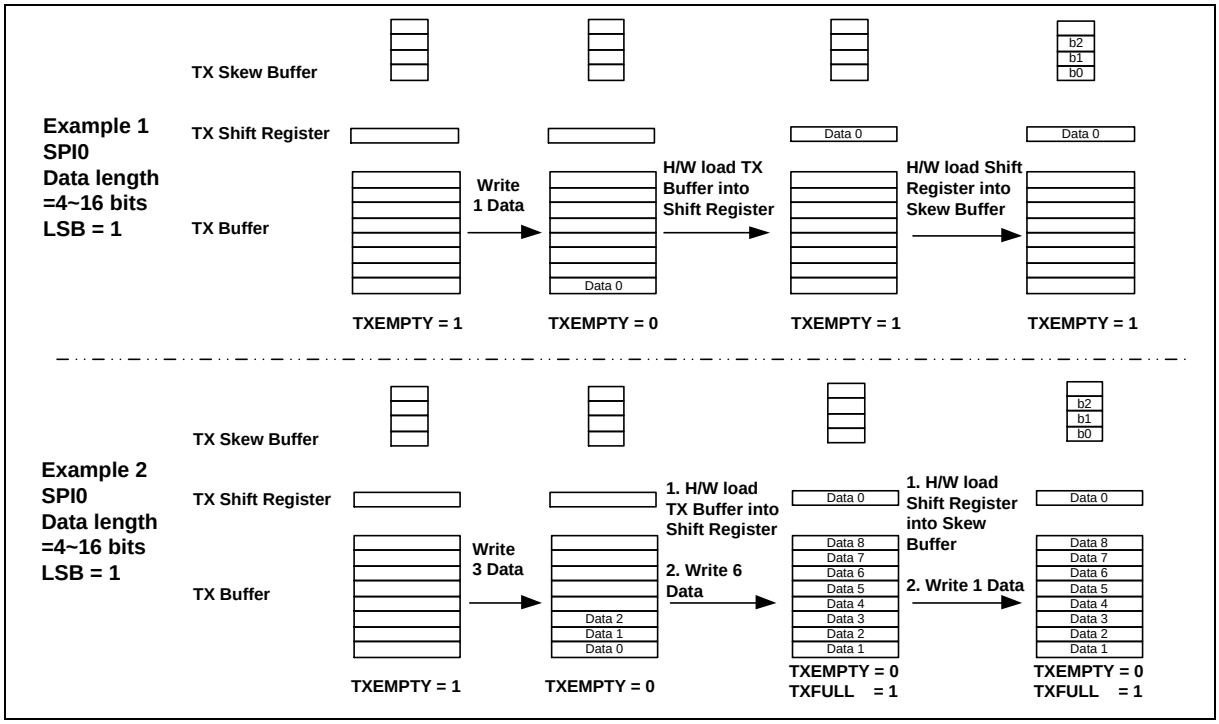


Figure 6.30-13 Transmit FIFO Buffer Example for 4~16 bits of data length

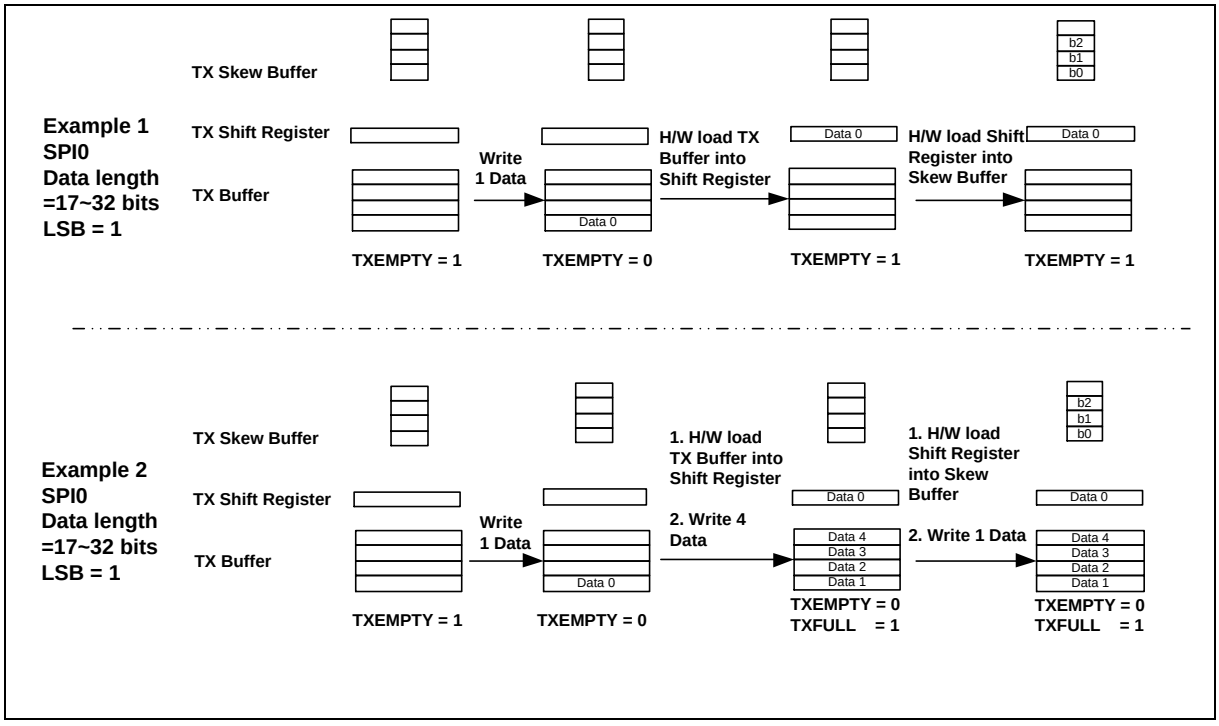


Figure 6.30-14 Transmit FIFO Buffer Example for 17~32 bits of data length

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the LPSPI_TX register is not updated after all data transfer are done, the transfer will stop.

In Master mode, during receiving operation, the serial data are received from LPSPI_MISO pin and stored to receive FIFO buffer.

The received data (Data0's b0, b1, …b31) is stored into skew buffer first according the serial clock (LPSPI_CLK) and then it is shift into the shift register bit by bit. The core logic will load the data in shift register into FIFO buffer when the received data bit count reach the value of DWIDTH (LPSPI_CTL[12:8]).

The RXEMPTY (LPSPI_STATUS[8]) will be cleared to 0 while the receive FIFO buffer contains unread data (see the Example 1 of Receive FIFO Buffer Example in Figure 6.30-15). The received data can be read by software from LPSPI_RX register as long as the RXEMPTY (LPSPI_STATUS[8]) is 0. If the receive FIFO buffer contains 8 unread data, the RXFULL (LPSPI_STATUS[9]) will be set to 1 (see the Example 2 of Receive FIFO Buffer Example in Figure 6.30-15).

The RXEMPTY (LPSPI_STATUS[8]) will be cleared to 0 while the receive FIFO buffer contains unread data (see the Example 1 of Receive FIFO Buffer Example in Figure 6.30-16). The received data can be read by software from LPSPI_RX register as long as the RXEMPTY (LPSPI_STATUS[8]) is 0. If the receive FIFO buffer contains 4 unread data, the RXFULL (LPSPI_STATUS[9]) will be set to 1 (see the Example 2 of Receive FIFO Buffer Example in Figure 6.30-16).

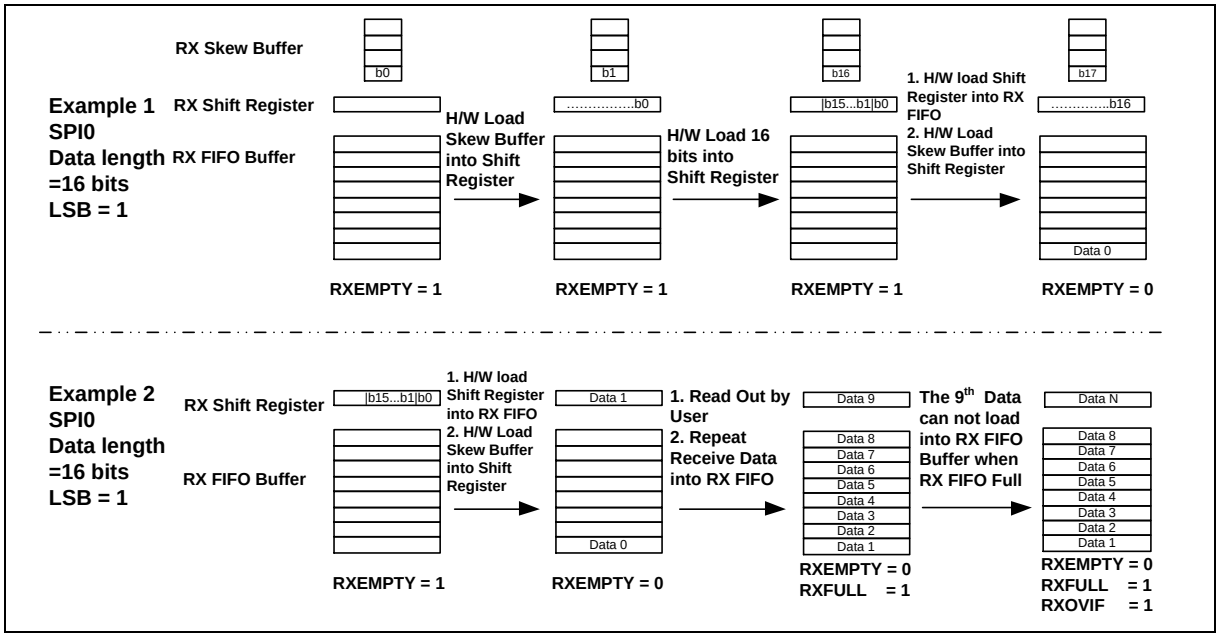


Figure 6.30-15 Receive FIFO Buffer Example for 16 bits of Data Length

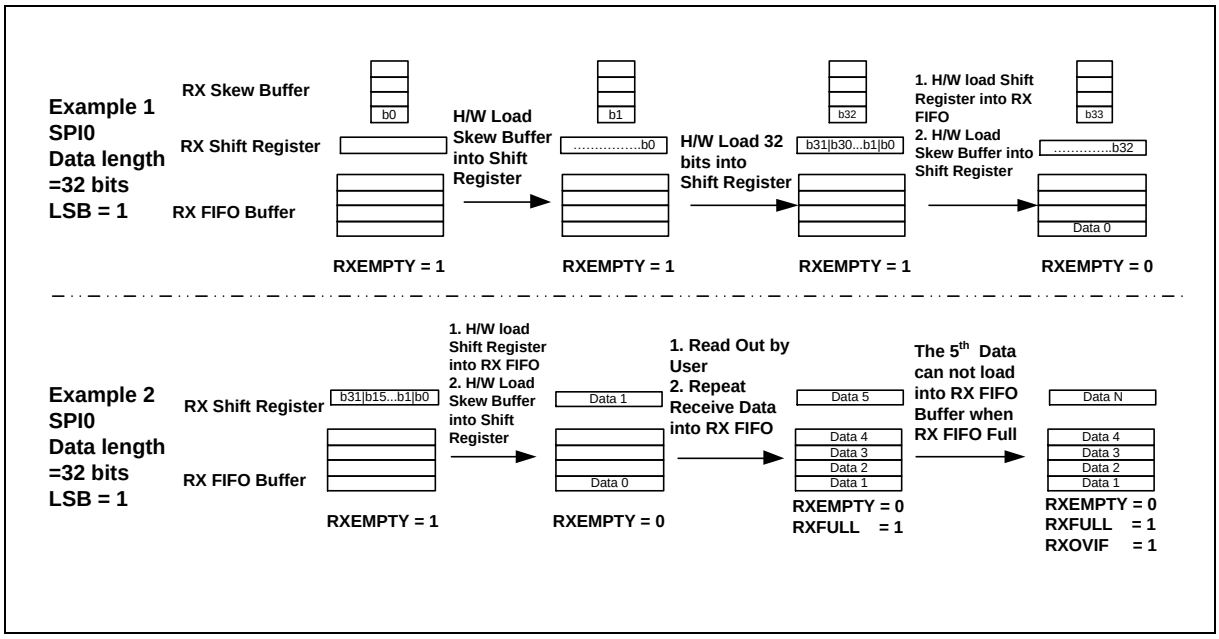


Figure 6.30-16 Receive FIFO Buffer Example for 32 bits of Data Length

In Slave mode, during transmission operation, when data is written to the LPSPI_TX register by software, the data will be loaded into transmit FIFO buffer and the TXEMPTY (LPSPI_STATUS[16]) will be set to 0. The transmission will start when the slave device receives clock signal from master. Data can be written to LPSPI_TX register as long as the TXFULL (LPSPI_STATUS[17]) is 0. After all data have been drawn out by the SPI transmission logic unit and the LPSPI_TX register is not updated by software, the TXEMPTY (LPSPI_STATUS[16]) will be set to 1.

If there is no any data written to the LPSPI_TX register, the transmit underflow interrupt flag, TXUFIF (LPSPI_STATUS[19]) will be set to 1 when the slave selection signal is active. The output data will be held by TXUFPOL (LPSPI_FIFOCTL[6]) setting during this transfer until the slave selection signal goes to inactive state. When the transmit underflow event occurs, the slave under run interrupt flag, SLVURIF

(LPSPI_STATUS[7]), will be set to 1 as LPSPI_SS goes to inactive state.

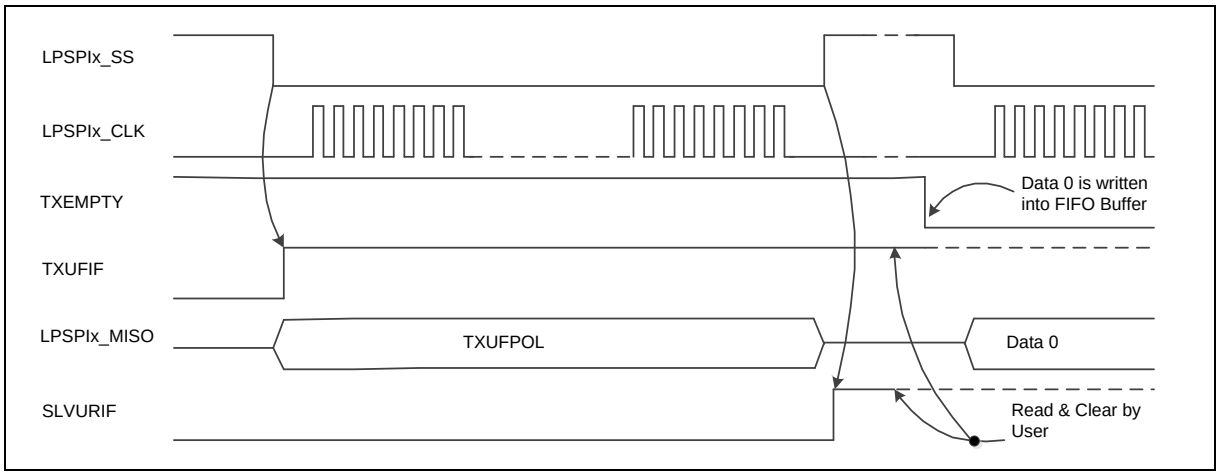


Figure 6.30-17 TX Underflow Event and Slave Under Run Event

In SPI Slave 3-Wire mode, the first 2-bit data is un-predicted (kept on the level of last bit in previously transfer) if the data is written into TX FIFO among 3 peripheral clock cycles before the SPI bus clock is presented. The other bits are held by TXUFPOL (LPSPI_FIFOCTL[6]) because there is TX underflow event. The written data will be transmitted in the next transfer.

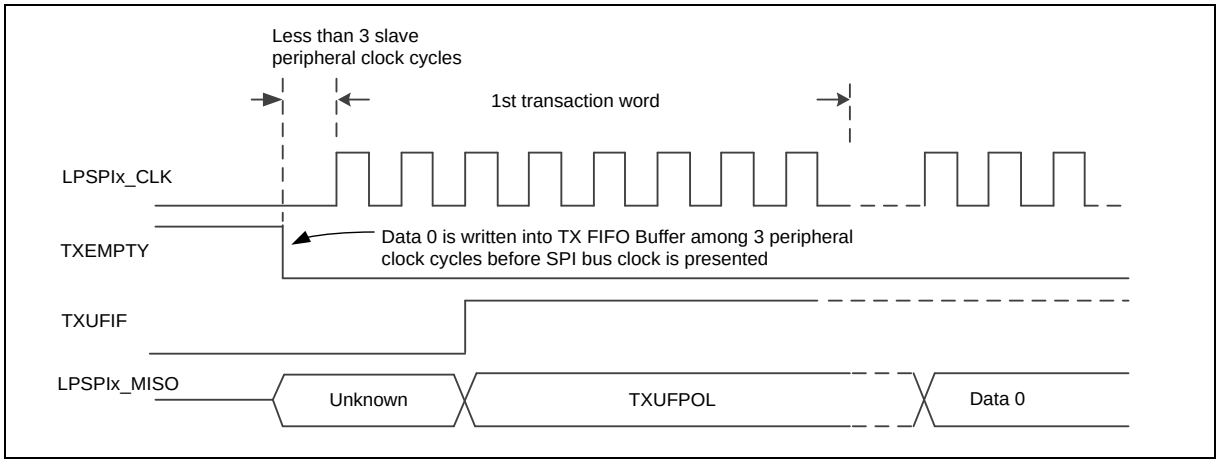


Figure 6.30-18 TX Underflow Event (SPI Slave 3-Wire Mode Enabled)

In Slave mode, during receiving operation, the serial data is received from LPSPI_MOSI pin and stored to LPSPI_RX register. The reception mechanism is similar to Master mode reception operation. If the receive FIFO buffer contains 4 unread data, the RXFULL (LPSPI_STATUS[9]) will be set to 1 and the RXOVIF (LPSPI_STATUS[11]) will be set to 1 if there is more serial data received from LPSPI_MOSI and follow-up data will be dropped (refer to the Receive FIFO Buffer Example figure).

If the receive bit count mismatches with the DWIDTH (LPSPI_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (LPSPI_STATUS[6]) will be set to 1 in SPI Slave mode. RX data will be written into RX FIFO and SLVBENUM (LPSPI_STATUS2[29:24]) will be updated when SLVBERX (LPSPI_FIFOCTL[10]) is enabled and SPI slave bit count error event happened. RX data will be dropped and SLVBENUM (LPSPI_STATUS2[29:24]) will be fixed to 0x0 when the control register SLVBERX (LPSPI_FIFOCTL[10]) is disabled and SPI slave bit count error event happened. The status register SLVBENUM (LPSPI_STATUS2[29:24]) indicates that effective bit number of uncompleted RX data when SPI slave bit count error happened. In Figure 6.30-19, the timing diagram of SPI slave bit count error and SLVBENUM is shown, and SLVBENUM will be updated to 0x8 when SPI slave device receives 8 bits data and DWIDTH (LPSPI_CTL[12:8]) sets to 16 bits at SPI bit count error event. The

SPI slave bit count error event (SLVBEIF) will generate RX FIFO write operation pulse to write RX data from RX shift register to RX FIFO, and RX FIFO count (RXCNT) will be updated to one.

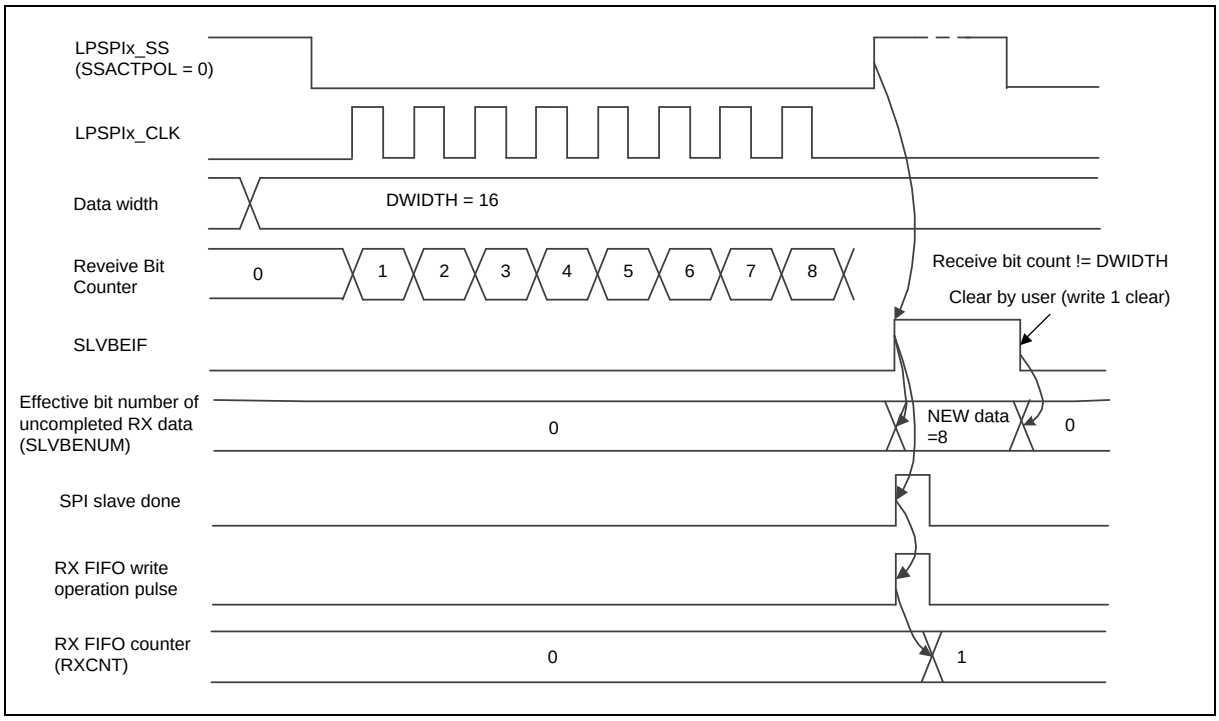


Figure 6.30-19 Slave Mode Bit Count Error and Effective Bit Number of Uncompleted RX data

A receive time-out function is built in this controller. When the receive FIFO is not empty and no read operation in receive FIFO over 64 SPI peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode, the receive time-out occurs and the RXTOIF (LPSPI_STATUS[12]) will be set to 1. When the receive FIFO is read by user, the time-out status will be cleared automatically.

6.30.5.9 Interrupt

- SPI unit transfer interrupt

  As LPSPI finishes a unit transfer, the unit transfer interrupt flag UNITIF (LPSPI_STATUS[1]) will be set to 1. The unit transfer interrupt event will generate an interrupt to CPU if the unit transfer interrupt enable bit UNITIEN (LPSPI_CTL[17]) is set. The unit transfer interrupt flag can be cleared only by writing 1 to it.

- SPI slave selection active/inactive interrupt

  In Slave mode, the slave selection active/inactive interrupt flag, SSACTIF (LPSPI_STATUS[2]) and SSINAIF (LPSPI_STATUS[3]), will be set to 1 when the SPIEN (LPSPI_CTL[0]) and SLAVE (LPSPI_CTL[18]) are set to 1 and the slave selection signal goes to active/inactive state. LPSPI will issue an interrupt if the SSINAIEN (LPSPI_SSCTL[13]) or SSACTIEN (LPSPI_SSCTL[12]), is set to 1.

- Slave bit count error interrupt

  In Slave mode, if the transmit/receive bit count mismatches with the DWIDTH (LPSPI_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (LPSPI_STATUS[6]) will be set to 1. The uncompleted transaction will be dropped from TX shift register. When the control register SLVBERX (LPSPI_FIFOCTL[10]) is disabled and SPI slave bit count error event happened, the uncompleted transaction data will be dropped from RX shift registers. When control register SLVBERX (LPSPI_FIFOCTL[10]) is enabled and SPI slave bit count error event happened, the uncompleted transaction data will be

written from RX shift registers into RX FIFO. LPSPI will issue an interrupt if the SLVBEIEN (LPSPI_SSCTL[8]) is set to 1.

Note: If the slave selection signal is active but there is no any serial clock input, the SLVBEIF (LPSPI_STATUS[6]) will be set to 1 when the slave selection signal goes to inactive state.

- TX underflow interrupt

  In SPI Slave mode, if there is no any data is written to the LPSPI_TX register, the TXUFIF (LPSPI_STATUS[19]) will be set to 1 when the slave selection signal is active. LPSPI will issue a TX underflow interrupt if the TXUFIEN (LPSPI_FIFOCTL[7]) is set to 1.

  Note: If underflow event occurs in SPI Slave mode, there are two conditions which make SPI Slave mode return to idle state and then goes for next transfer: (1) set TXRST to 1 (2) slave select signal is changed to inactive state while SLV3WIRE=0.

- Slave TX under run interrupt

  If the TX underflow event occurs, the SLVURIF (LPSPI_STATUS[7]) will be set to 1 when LPSPI_SS goes to inactive state. LPSPI will issue a TX under run interrupt if the SLVURIEN (LPSPI_SSCTL[9]) is set to 1.

  Note: In Slave 3-Wire mode, the slave selection signal is considered active all the time so that user shall poll the TXUFIF (LPSPI_STATUS[19]) to know if there is TX underflow event or not.

- Receive Overrun interrupt

  In Slave mode, if the receive FIFO buffer contains 4 unread data for 17~32-bits of data length, the RXFULL (LPSPI_STATUS[9]) will be set to 1 and the RXOVIF (LPSPI_STATUS[11]) will be set to 1 if there is more serial data received from SPI bus and follow-up data will be dropped. LPSPI will issue an interrupt if the RXOVIEN (LPSPI_FIFOCTL[5]) is set to 1.

  If the receive FIFO buffer contains 8 unread data for 4~16-bits of data length, the RXFULL (LPSPI_STATUS[9]) will be set to 1 and the RXOVIF (LPSPI_STATUS[11]) will be set to 1 if there is more serial data received from SPI bus and follow-up data will be dropped. LPSPI will issue an interrupt if the RXOVIEN (LPSPI_FIFOCTL[5]) is set to 1.

- Receive FIFO time-out interrupt

  If there is a received data in the FIFO buffer and it is not read by software over 64 SPI peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode, it will send a RX time-out interrupt to the system if the RX time-out interrupt enable bit, RXTOIEN (LPSPI_FIFOCTL[4]), is set to 1.

- Transmit FIFO interrupt

  In FIFO mode, if the valid data count of the transmit FIFO buffer is less than or equal to the setting value of TXTH (LPSPI_FIFOCTL[30:28]), the transmit FIFO interrupt flag TXTHIF (LPSPI_STATUS[18]) will be set to 1. LPSPI will generate a transmit FIFO interrupt to the system if the transmit FIFO interrupt enable bit, TXTHIEN (LPSPI_FIFOCTL[3]), is set to 1.

- Receive FIFO interrupt

  In FIFO mode, if the valid data count of the receive FIFO buffer is larger than the setting value of RXTH (LPSPI_FIFOCTL[26:24]), the receive FIFO interrupt flag RXTHIF (LPSPI_STATUS[10]) will be set to 1. LPSPI will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, RXTHIEN (LPSPI_FIFOCTL[2]), is set to 1.

6.30.5.10 Automatic Operaion Mode

For SPI Master mode, LPSPI supports three automatic operation modes while chip in NPD0/1/2/3/4 mode. The automatic operaion Master modes allow LPSPI to do SPI transmit/receive operation automatically by LPPDMA without the intervening by CPU. The block diagram of LPSPI automatic operation mode shows in Figure 6.30-20. Setting the AUTOEN (LPSPI_AUTOCTL[8]) to 1 can enable LPSPI auto-operation mode. In auto-operation mode, LPSPI can be triggered to do transfer when TRIGEN (LPSPI_AUTOCTL[4]) is set to 1, even if chip under NPD0/1/2/3/4 mode. The trigger sources of LPSPI include LPTMR0/1, TTMR0/1, WKIOA0, WKIOB0, WKIOC0, WKIOD0. The trigger source can be selected by TRIGSEL (LPSPI_AUTOCTL[3:0]). User can also trigger LPSPI directly by setting SWTRIG (LPSPI_AUTOCTL[9]) to 1. User also need to properly configure LPPDMA according the following different types of auto-operation mode, and chip can be waked up by LPPDMA from NPD0/1/2/3/4 mode if all the data transfer down.

These auto-operation modes are used as a specific purpose to communicate with external SPI slave device, which allows some TX transfers followed by some RX transfers. The overall transfer operation is combined with a TX phase and a RX phase, that the TX phase sends commands to external device, and the RX phase receives the data read from external device. To confugure the data for TX phase, user need to use scatter-gather mode of LPPDMA to arrange total TX data, with each table containing the TX data amount to be transferred for each trigger event. If there is no any table for TX data, LPSPI will not enter TX phase. On the other hand, to configure the data for RX phase, user just need to use basic mode of LPPDMA to arrange total RX data, which there is one table containing all the data amount needed being received. Moreover, for the RX phase, user also need to configure the TCNT (LPSPI_AUTOCTL[23:16]) to determine the received data amount while in RX phase for each trigger event. If there is no any table for RX data or TCNT (LPSPI_AUTOCTL[23:16]) is set to 0, LPSPI will not enter RX phase.

For the condition that LPSPI has received the data amount equaling to TCNT (LPSPI_AUTOCTL[23:16]), chip can be waked up if correlated wake-up enable control bits CNTWKEN (LPSPI_AUTOCTL[10]) and CNTIEN (LPSPI_AUTOCTL[5]) are set to 1.

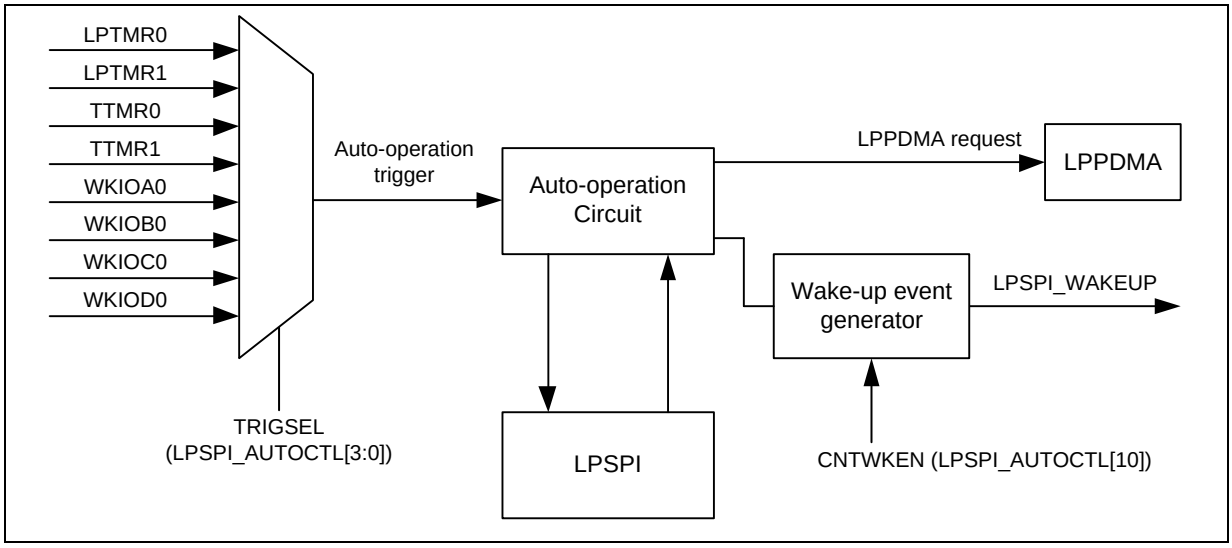


Figure 6.30-20 LPSPI Auto-operation Mode Block Diagram

The auto-operation Master mode can be achieved by full-duplex mode or half-duplex mode; moreover, there is a control bit named FULLRXEN (LPSPI_AUTOCTL[6]) that can be used to determine whether LPPDMA should receive the RX data in TX phase of full-duplex mode. The Auto Full-Duplex Master Mode communicate external device by full-duplex Master mode and LPPDMA will not move the RX data received in TX phase. The Auto Full-Duplex Full-RX Master Mode communicate external device by full-duplex Master mode and LPPDMA will move the RX data received in TX phase. The Auto Half-Duplex Master Mode communicate external device by half-duplex Master mode, and the MISO pin is not used.

To set and enable automatic operation Master mode, software can do the following configuration before entering NPDx mode:

- LPPDMA Descriptor Table Configuration for LPSPI TX/RX transfer
  - Use scatter-gather mode for TX data, and each table contains the amount of data for transfer by one trigger
  - Use basic mode for RX data, and the table contains the amount of all the data needed to be received
- LPSPI Clock & Pin Configuration
  - Follow LPSPI0 Basic Configuration to do setting.
- LPSPI Configuration
  - Set SPIEN, SLAVE, RXONLY, DATDIR to 0 in LPSPI_CTL register.
  - Set SS to 0 in LPSPI_SSCTL register.
  - Configure other general bits in LPSPI_CTL/LPSPI_SSCTL register.
  - Set TXPDMAEN, RXPDMAEN to 0 in LPSPI_PDMACTL register.
  - Configure HALFDPX in LPSPI_CTL register and FULLRXEN in LPSPI_AUTOCTL register to select Auto-Operation Mode.
  - Configure TCNT[7:0], TRIGSEL[2:0], TRIGEN, CNTIEN, CNTWKEN, FULLRXEN in LPSPI_AUTOCTL register.
  - Set AUTOEN to 1 in LPSPI_AUTOCTL register.
- Trigger Source Configuration
  - According to slected trigger source (TRIGSEL[2:0] in LPSPI_AUTOCTL register) do proper configuration.
  - SWTRIG in LPSPI_AUTOCTL register is provided as an alternative trigger source by user control (when Auto-Operation Mode used in non-NPDx mode).

The following registers cannot be read after AUTOEN (LPSPI_AUTOCTL[8]) is set to 1:

- LPSPI_RX register

The following registers cannot be writen after AUTOEN (LPSPI_AUTOCTL[8]) is set to 1:

- LPSPI_CTL, LPSPI_CKDIV, LPSPI_SSCTL, LPSPI_PDMACTL, LPSPI_FIFOCTL, LPSPI_TX, LPSPI_AUTOCTL, register

After all the data has been trasmitted/received, before doing any configuration, set AUTOEN (LPSPI_AUTOCTL[8]) to 0 in advance.

Below truth table shows the setting of each Master mode for automatic operation:

LPSPI Auto-Operation Master Mode	AUTOEN	TRIGEN	HALFDPX	FULLRXEN	SLAVE
Normal (non-auto) Operation Master Mode	0	0	X	X	0
Auto Full-Duplex Master Mode	1	1	0	0	0
Auto Full-Duplex Full-RX Master Mode	1	1	0	1	0
Auto Half-Duplex Master Mode	1	1	1	0	0

The waveform of Auto Full-Duplex Master Mode is the same as Auto Full-Duplex Full-RX Master mode,
But the RX data received in TX Phase will be discarded in Auto Full-Duplex Master Mode.

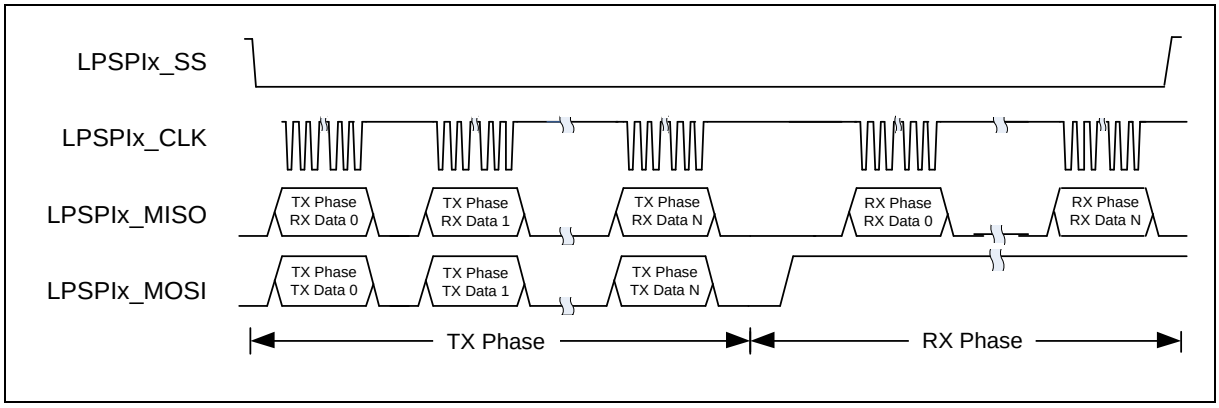


Figure 6.30-21 LPSPI Auto Full-Duplex Master Mode when AUTOSS set to 0

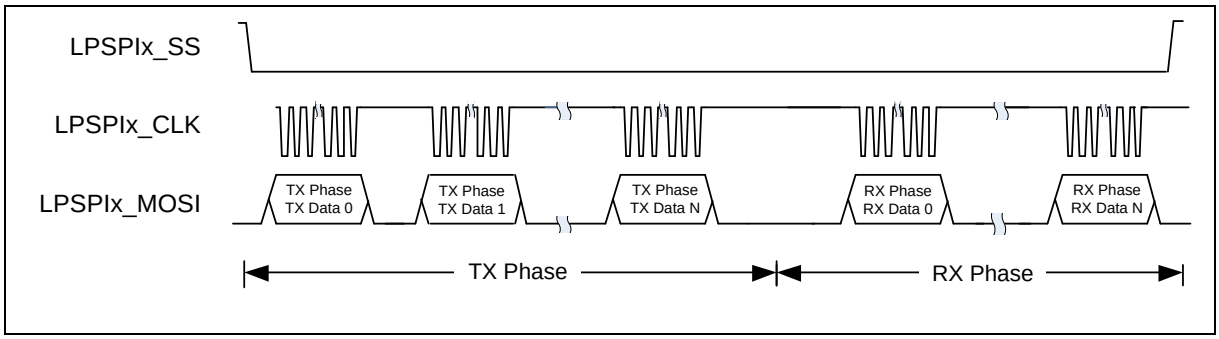


Figure 6.30-22 LPSPI Auto Half-Duplex Master Mode when AUTOSS set to 0

Moreover, below figures show the condition when AUTOSS (LPSPI_SSCTL[3]) is set. LPSPIx_SS is de-asserted automatically when SPI bus is idle.

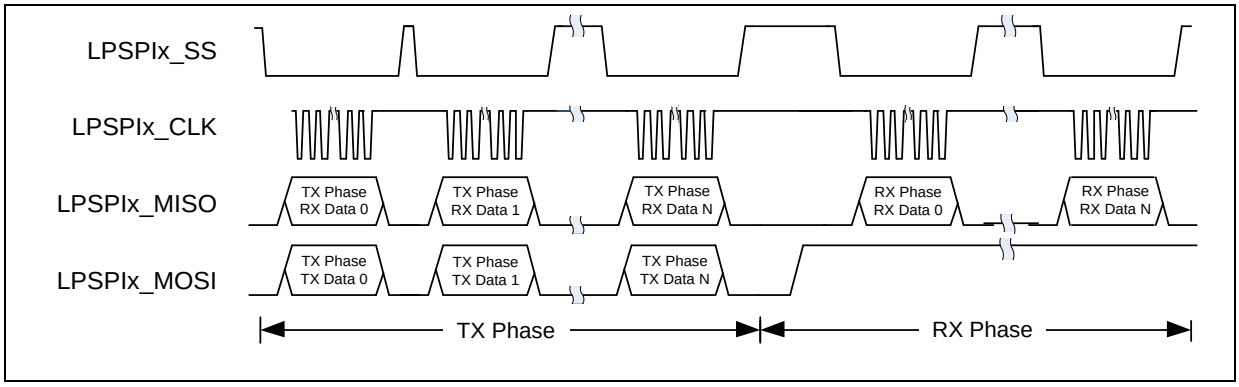


Figure 6.30-23 LPSPI Auto Full-Duplex Master Mode when AUTOSS set to 1

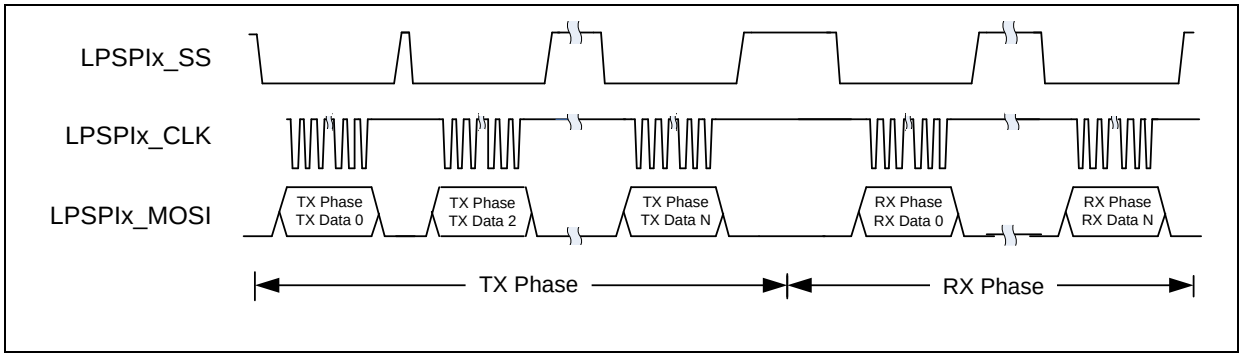


Figure 6.30-24 LPSPI Auto Half-Duplex Master Mode when AUTOSS set to 1

6.30.6 Timing Diagram

The active state of slave selection signal can be defined by setting the SSACTPOL (LPSPI_SSCTL[2]). The SPI clock which is in idle state can be configured as high or low state by setting the CLKPOL (LPSPI_CTL[3]). It also provides the bit length of a transaction word in DWIDTH (LPSPI_CTL[12:8]), and transmitting/receiving data from MSB or LSB first in LSB (LPSPI_CTL[13]). User can also select which edge of SPI clock to transmit/receive data in TXNEG/RXNEG (LPSPI_CTL[2:1]). Four LPSPI timing diagrams for master/slave operations and the related settings are shown below.

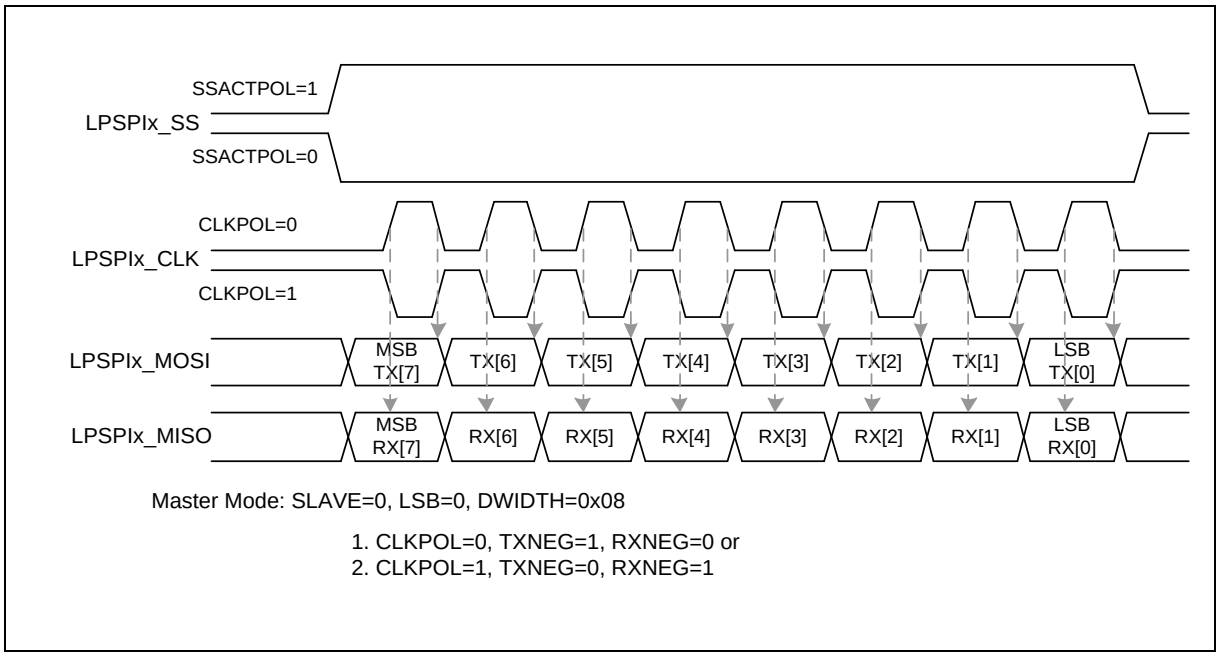


Figure 6.30-25 LPSPI Timing in Master Mode

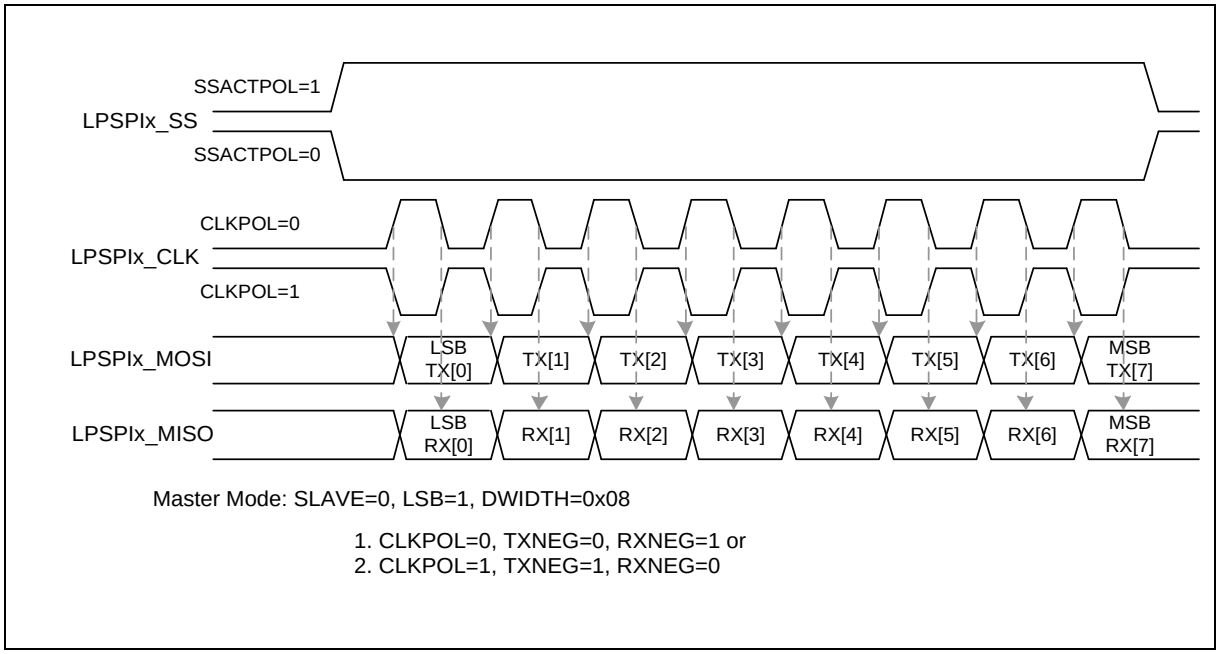


Figure 6.30-26 LPSPI Timing in Master Mode (Alternate Phase of LPSPI_CLK)

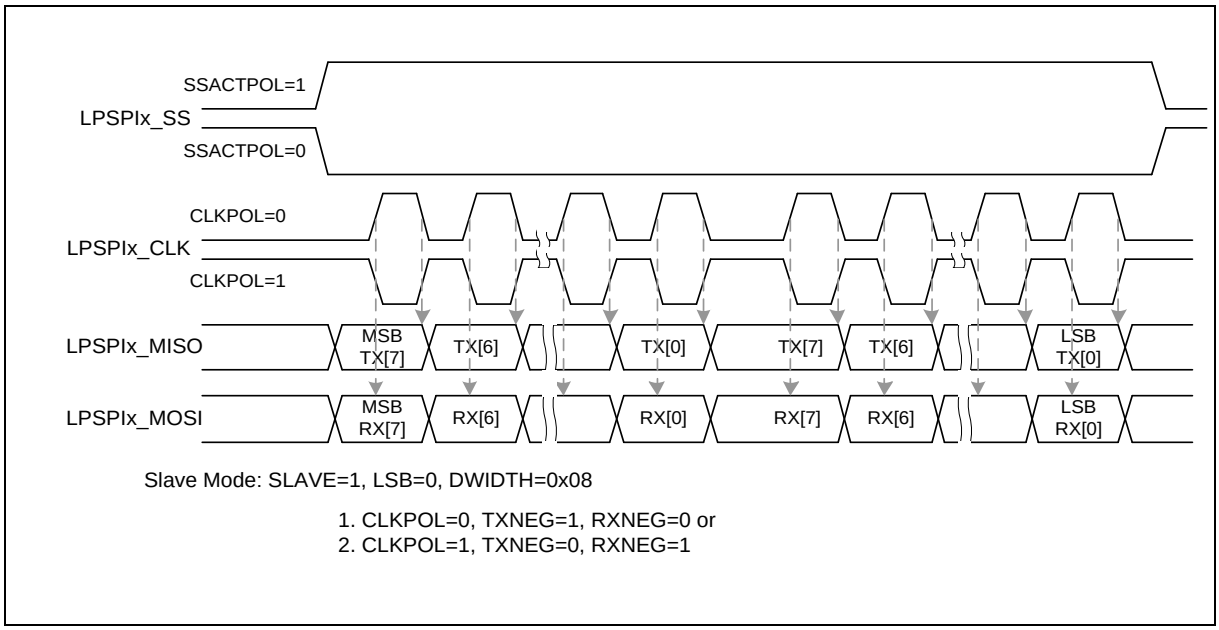


Figure 6.30-27 LPSPI Timing in Slave Mode

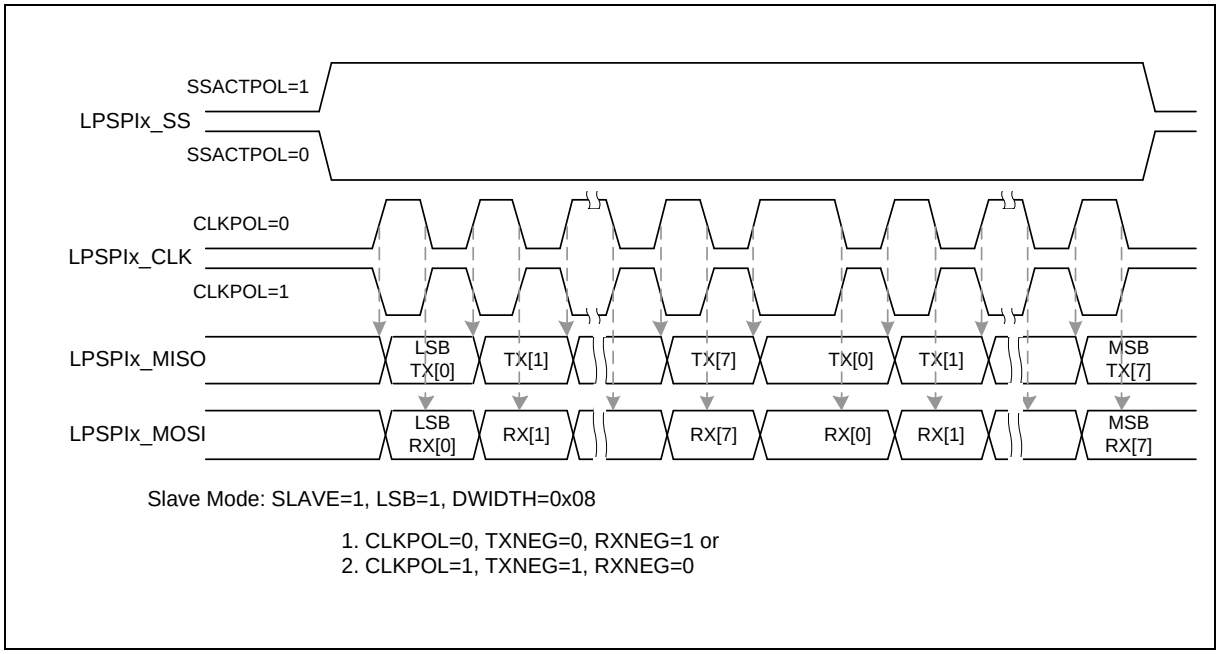


Figure 6.30-28 LPSPI Timing in Slave Mode (Alternate Phase of LPSPI_CLK)

6.30.7 Programming Examples

Example 1

LPSPI is set as a full-duplex master to access an off-chip slave device with the following specifications:

- Data bit is latched on positive edge of SPI bus clock.
- Data bit is driven on negative edge of SPI bus clock.
- Data is transferred from MSB first.
- SPI bus clock is idle at low state.
- Only one byte of data to be transmitted/received in a transaction.
- Uses the first SPI slave select pin to connect with an off-chip slave device. The slave selection signal is active low.

The operation flow is as follows:

1. Set DIVIDER (LPSPI_CLKDIV[8:0]) to determine the output frequency of SPI clock.
2. Write the LPSPI_SSCTL register a proper value for the related settings of Master mode:
   1) Clear AUTOSS (LPSPI_SSCTL[3]) to 0 to disable the Automatic Slave Selection function.
   2) Configure slave selection signal as active low by clearing SSACTPOL (LPSPI_SSCTL[2]) to 0.
   3) Enable slave selection signal by setting SS (LPSPI_SSCTL[0]) to 1 to activate the off-chip slave device.
3. Write the related settings into the LPSPI_CTL register to control the SPI master actions.
   1) Configure this LPSPI controller as master device by setting SLAVE (LPSPI_CTL[18]) to 0.
   2) Force the SPI clock idle state at low by clearing CLKPOL (LPSPI_CTL[3]) to 0.
   3) Select data transmitted on negative edge of SPI bus clock by setting TXNEG

(LPSPI_CTL[2]) to 1.

4) Select data latched on positive edge of SPI bus clock by clearing RXNEG (LPSPI_CTL[1]) to 0.

5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (LPSPI_CTL[12:8] = 0x08).

6) Set MSB transfer first by clearing LSB (LPSPI_CTL[13]) to 0.

4. Set SPIEN (LPSPI_CTL[0]) to 1 to enable the data transfer with the SPI interface.
5. If this SPI master attempts to transmit (write) one byte data to the off-chip slave device, write the byte data that will be transmitted into the LPSPI_TX register.
6. Waiting for SPI interrupt if the UNITIEN (LPSPI_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (LPSPI_STATUS[1]).
7. Read out the received one byte data from LPSPI_RX register.
8. Go to 5 to continue another data transfer or set SS (LPSPI_SSCTL[0]) to 0 to inactivate the off-chip slave device.

Example 2

The LPSPI controller is set as a full-duplex slave device and connects with an off-chip master device. The off-chip master device communicates with the on-chip SPI slave controller through the SPI interface with the following specifications:

- Data bit is latched on positive edge of SPI bus clock.
- Data bit is driven on negative edge of SPI bus clock.
- Data is transferred from LSB first.
- SPI bus clock is idle at high state.
- Only one byte of data to be transmitted/received in a transaction.
- Slave selection signal is active high.

The operation flow is as follows:

1. Write the LPSPI_SSCTL register a proper value for the related settings of Slave mode.
2. Select high level for the input of slave selection signal by setting SSACTPOL (LPSPI_SSCTL[2]) to 1.
3. Write the related settings into the LPSPI_CTL register to control this SPI slave actions
   1) Set the LPSPI controller as slave device by setting SLAVE (LPSPI_CTL[18]) to 1.
   2) Select the SPI clock idle state at high by setting CLKPOL (LPSPI_CTL[3]) to 1.
   3) Select data transmitted on negative edge of SPI bus clock by setting TXNEG (LPSPI_CTL[2]) to 1.
   4) Select data latched on positive edge of SPI bus clock by clearing RXNEG (LPSPI_CTL[1]) to 0.
   5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (LPSPI_CTL[12:8] = 0x08).
4. Set LSB transfer first by setting LSB (LPSPI_CTL[13]) to 1.
5. Set the SPIEN (LPSPI_CTL[0]) to 1. Wait for the slave select trigger input and SPI clock input from the off-chip master device to start the data transfer.
6. If this SPI slave attempts to transmit (be read) one byte data to the off-chip master device, write the byte data that will be transmitted into the LPSPI_TX register.
7. If this SPI slave just only attempts to receive (be written) one byte data from the off-chip

master device and does not care what data will be transmitted, the LPSPI_TX register does not need to be updated by software.

8. Waiting for SPI interrupt if the UNITIEN (LPSPI_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (LPSPI_STATUS[1]).
9. Read out the received one byte data from LPSPI_RX register.
10. Go to 6 to continue another data transfer or stop data transfer.

6.30.8 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
LPSPI Base Address: **LPSPI_BA = 0x400E_1000**				
LPSPI_CTL	LPSPI_BA+0x00	R/W	SPI Control Register	0x0000_0034
LPSPI_CLKDIV	LPSPI_BA+0x04	R/W	SPI Clock Divider Register	0x0000_0000
LPSPI_SSCTL	LPSPI_BA+0x08	R/W	SPI Slave Select Control Register	0x0000_0000
LPSPI_PDMACTL	LPSPI_BA+0x0C	R/W	SPI PDMA Control Register	0x0000_0000
LPSPI_FIFOCTL	LPSPI_BA+0x10	R/W	SPI FIFO Control Register	0x2200_0000
LPSPI_STATUS	LPSPI_BA+0x14	R/W	SPI Status Register	0x0005_0110
LPSPI_STATUS2	LPSPI_BA+0x18	R	SPI Status2 Register	0x0000_0000
LPSPI_TX	LPSPI_BA+0x20	W	SPI Data Transmit Register	0x0000_0000
LPSPI_RX	LPSPI_BA+0x30	R	SPI Data Receive Register	0x0000_0000
LPSPI_AUTOCTL	LPSPI_BA+0x50	R/W	LPSPI Automatic Operation Control Register	0x0000_0000
LPSPI_AUTOSTS	LPSPI_BA+0x54	R/W	LPSPI Automatic Operation Status Register	0x0000_0000

6.30.9 Register Description

LPSPI Control Register (LPSPI_CTL)

Register	Offset	R/W	Description	Reset Value
LPSPI_CTL	LPSPI_BA+0x00	R/W	SPI Control Register	0x0000_0034

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			DATDIR	REORDER	SLAVE	UNITIEN	Reserved
15	14	13	12	11	10	9	8
RXONLY	HALFDPX	LSB	DWIDTH				
7	6	5	4	3	2	1	0
SUSPITV				CLKPOL	TXNEG	RXNEG	SPIEN

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**DATDIR**	**Data Port Direction Control** This bit is used to select the data input/output direction in half-duplex transfer and Dual/Quad transfer 0 = SPI data is input direction. 1 = SPI data is output direction.
[19]	**REORDER**	**Byte Reorder Function Enable Bit** 0 = Byte Reorder function Disabled. 1 = Byte Reorder function Enabled. A byte suspend interval will be inserted among each byte. The period of the byte suspend interval depends on the setting of SUSPITV. **Note:** Byte Reorder function is only available if DWIDTH is defined as 16, 24, and 32 bits.
[18]	**SLAVE**	**Slave Mode Control** 0 = Master mode. 1 = Slave mode.
[17]	**UNITIEN**	**Unit Transfer Interrupt Enable Bit** 0 = SPI unit transfer interrupt Disabled. 1 = SPI unit transfer interrupt Enabled.
[16]	**Reserved**	Reserved.
[15]	**RXONLY**	**Receive-only Mode Enable Bit** This bit field is only available in Master mode. In receive-only mode, SPI Master will generate SPI bus clock continuously for receiving data bit from SPI slave device and assert the BUSY status. 0 = Receive-only mode Disabled. 1 = Receive-only mode Enabled. **Note:** This bit is for Master Mode only.
[14]	**HALFDPX**	**SPI Half-duplex Transfer Enable Bit**

		This bit is used to select full-duplex or half-duplex for SPI transfer. The bit field DATDIR (LPSPI_CTL[20]) can be used to set the data direction in half-duplex transfer. 0 = SPI operates in full-duplex transfer. 1 = SPI operates in half-duplex transfer.
[13]	**LSB**	**Send LSB First** 0 = The MSB, which bit of transmit/receive register depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, bit 0 of the SPI TX register, is sent first to the SPI data output pin, and the first bit received from the SPI data input pin will be put in the LSB position of the RX register (bit 0 of LPSPI_RX).
[12:8]	**DWIDTH**	**Data Width** This field specifies how many bits can be transmitted/received in one transaction. The minimum bit length is 4 bits and can up to 32 bits. DWIDTH = 0x04 .... 4 bits. DWIDTH = 0x05 .... 5 bits. DWIDTH = 0x06 .... 6 bits. DWIDTH = 0x07 .... 7 bits. DWIDTH = 0x08 .... 8 bits. DWIDTH = 0x09 .... 9 bits. ..... DWIDTH = 0x1F .... 31 bits. DWIDTH = 0x00 .... 32 bits. **Note:** This bit field will decide the depth of TX/RX FIFO configuration in SPI mode**.** Therefore, changing this bit field will clear TX/RX FIFO by hardware automatically.
[7:4]	**SUSPITV**	**Suspend Interval** The four bits provide configurable suspend interval between two successive transmit/receive transaction in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. (SUSPITV[3:0] + 0.5) * period of LPSPI_CLK clock cycle Example: SUSPITV = 0x0 .... 0.5 LPSPI_CLK clock cycle. SUSPITV = 0x1 .... 1.5 LPSPI_CLK clock cycle. ..... SUSPITV = 0xE .... 14.5 LPSPI_CLK clock cycle. SUSPITV = 0xF .... 15.5 LPSPI_CLK clock cycle. **Note:** These bits are for Master Mode only.
[3]	**CLKPOL**	**Clock Polarity** 0 = SPI bus clock is idle low. 1 = SPI bus clock is idle high.
[2]	**TXNEG**	**Transmit on Negative Edge** 0 = Transmitted data output signal is changed on the rising edge of SPI bus clock. 1 = Transmitted data output signal is changed on the falling edge of SPI bus clock.
[1]	**RXNEG**	**Receive on Negative Edge** 0 = Received data input signal is latched on the rising edge of SPI bus clock. 1 = Received data input signal is latched on the falling edge of SPI bus clock.

[0]	**SPIEN**	**SPI Transfer Control Enable Bit** In Master mode, the transfer will start when there is data in the FIFO buffer after this bit is set to 1. In Slave mode, this device is ready to receive data when this bit is set to 1. 0 = Transfer control Disabled. 1 = Transfer control Enabled. **Note:** Before changing the configurations of LPSPI_CTL, LPSPI_CLKDIV, LPSPI_SSCTL and LPSPI_FIFOCTL registers, user shall clear the SPIEN (LPSPI_CTL[0]) and confirm the SPIENSTS (LPSPI_STATUS[15]) is 0.

LPSPI Clock Divider Register (LPSPI_CLKDIV)

Register	Offset	R/W	Description	Reset Value
LPSPI_CLKDIV	LPSPI_BA+0x04	R/W	SPI Clock Divider Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**DIVIDER**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**DIVIDER**	**Clock Divider** The value in this field is the frequency divider for generating the peripheral clock, f_{spi_eclk}, and the SPI bus clock of SPI Master. The frequency is obtained according to the following equation. $f_{spi_eclk} = \frac{f_{spi_clock_src}}{(DIVIDER + 1)}$ where $f_{spi_clock_src}$ is the peripheral clock source, which is defined in the clock control register, CLK_CLKSEL2 or CLK_CLKSEL3. **Note 1:** The time interval must be larger than or equal to 8 peripheral clock cycles between releasing SPI IP software reset and setting this clock divider register.

Note: DIVIDER should be set carefully because the peripheral clock frequency must be slower than or equal to system frequency.

LPSPI Slave Select Control Register (LPSPI_SSCTL)

Register	Offset	R/W	Description	Reset Value
LPSPI_SSCTL	LPSPI_BA+0x08	R/W	SPI Slave Select Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVURIEN	SLVBEIEN
7	6	5	4	3	2	1	0
Reserved			SLV3WIRE	AUTOSS	SSACTPOL	Reserved	SS

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable Bit** 0 = Slave select inactive interrupt Disabled. 1 = Slave select inactive interrupt Enabled.
[12]	**SSACTIEN**	**Slave Select Active Interrupt Enable Bit** 0 = Slave select active interrupt Disabled. 1 = Slave select active interrupt Enabled.
[11:10]	**Reserved**	Reserved.
[9]	**SLVURIEN**	**Slave Mode TX Under Run Interrupt Enable Bit** 0 = Slave mode TX under run interrupt Disabled. 1 = Slave mode TX under run interrupt Enabled.
[8]	**SLVBEIEN**	**Slave Mode Bit Count Error Interrupt Enable Bit** 0 = Slave mode bit count error interrupt Disabled. 1 = Slave mode bit count error interrupt Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**SLV3WIRE**	**Slave 3-wire Mode Enable Bit** In Slave 3-wire mode, the LPSPI controller can work with 3-wire interface including LPSPI_CLK, LPSPI_MISO and LPSPI_MOSI pins. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface. **Note:** This bit is for Slave Mode only.
[3]	**AUTOSS**	**Automatic Slave Selection Function Enable Bit** 0 = Automatic slave selection function Disabled. Slave selection signal will be asserted/de-asserted according to SS (LPSPI_SSCTL[0]). 1 = Automatic slave selection function Enabled. **Note:** This bit is for Master Mode only.

[2]	**SSACTPOL**	**Slave Selection Active Polarity** This bit defines the active polarity of slave selection signal (LPSPI_SS). 0 = The slave selection signal LPSPI_SS is active low. 1 = The slave selection signal LPSPI_SS is active high.
[1]	**Reserved**	Reserved.
[0]	**SS**	**Slave Selection Control** If AUTOSS bit is cleared to 0, 0 = set the LPSPI_SS line to inactive state. 1 = set the LPSPI_SS line to active state. If the AUTOSS bit is set to 1, 0 = Keep the LPSPI_SS line at inactive state. 1 = LPSPI_SS line will be automatically driven to active state for the duration of data transfer, and will be driven to inactive state for the rest of the time. The active state of LPSPI_SS is specified in SSACTPOL (LPSPI_SSCTL[2]). **Note:** This bit is for Master Mode only.

LPSPI PDMA Control Register (LPSPI_PDMACTL)

Register	Offset	R/W	Description	Reset Value
LPSPI_PDMACTL	LPSPI_BA+0x0C	R/W	SPI PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**PDMARST**	**RXPDMAEN**	**TXPDMAEN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the PDMA control logic of the LPSPI controller. This bit will be automatically cleared to 0.
[1]	**RXPDMAEN**	**Receive PDMA Enable Bit** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[0]	**TXPDMAEN**	**Transmit PDMA Enable Bit** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled. **Note 1:** In SPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, RX PDMA function cannot be enabled prior to TX PDMA function. User can enable TX PDMA function firstly or enable both functions simultaneously. **Note 2:** In SPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, TX PDMA function cannot be disabled prior to RX PDMA function. User can disable RX PDMA function firstly or disable both functions simultaneously.

LPSPI FIFO Control Register (LPSPI_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
LPSPI_FIFOCTL	LPSPI_BA+0x10	R/W	SPI FIFO Control Register	0x2200_0000

31	30	29	28	27	26	25	24
Reserved	TXTH			Reserved	RXTH		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					SLVBERX	TXFBCLR	RXFBCLR
7	6	5	4	3	2	1	0
TXUFIEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TXTH**	**Transmit FIFO Threshold** If the valid data count of the transmit FIFO buffer is less than or equal to the TXTH setting, the TXTHIF bit will be set to 1, else the TXTHIF bit will be cleared to 0. The MSB of this bit field is only meaningful while SPI mode 4~16 bits of data length.
[27]	**Reserved**	Reserved.
[26:24]	**RXTH**	**Receive FIFO Threshold** If the valid data count of the receive FIFO buffer is larger than the RXTH setting, the RXTHIF bit will be set to 1, else the RXTHIF bit will be cleared to 0. The MSB of this bit field is only meaningful while SPI mode 4~16 bits of data length.
[23:11]	**Reserved**	Reserved.
[10]	**SLVBERX**	**RX FIFO Write Data Enable Bit When Slave Mode Bit Count Error** 0 = Uncompleted RX data will be dropped from RX FIFO when bit count error event happened in SPI Slave mode. 1 = Uncompleted RX data will be written into RX FIFO when bit count error event happened in SPI Slave mode. User can read SLVBENUM (LPSPI_STATUS2[29:24]) to know that the effective bit number of uncompleted RX data when SPI slave bit count error happened. **Note:** This bit is for Slave Mode only.
[9]	**TXFBCLR**	**Transmit FIFO Buffer Clear** 0 = No effect. 1 = Clear transmit FIFO pointer. The TXFULL bit will be cleared to 0 and the TXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 1 system clock after it is set to 1. **Note:** The TX shift register will not be cleared.
[8]	**RXFBCLR**	**Receive FIFO Buffer Clear** 0 = No effect. 1 = Clear receive FIFO pointer. The RXFULL bit will be cleared to 0 and the RXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 1 system clock after it is set to 1.

		Note: The RX shift register will not be cleared.
[7]	**TXUFIEN**	**TX Underflow Interrupt Enable Bit** When TX underflow event occurs in Slave mode, TXUFIF (LPSPI_STATUS[19]) will be set to 1. This bit is used to enable the TX underflow interrupt. 0 = Slave TX underflow interrupt Disabled. 1 = Slave TX underflow interrupt Enabled.
[6]	**TXUFPOL**	**TX Underflow Data Polarity** 0 = The SPI data out is keep 0 if there is TX underflow event in Slave mode. 1 = The SPI data out is keep 1 if there is TX underflow event in Slave mode. **Note 1:** The TX underflow event occurs if there is no any data in TX FIFO when the slave selection signal is active. **Note 2:** When TX underflow event occurs, LPSPI_MISO pin state will be determined by this setting even though TX FIFO is not empty afterward. Data stored in TX FIFO will be sent through LPSPI_MISO pin in the next transfer frame.
[5]	**RXOVIEN**	**Receive FIFO Overrun Interrupt Enable Bit** 0 = Receive FIFO overrun interrupt Disabled. 1 = Receive FIFO overrun interrupt Enabled.
[4]	**RXTOIEN**	**Receive Time-out Interrupt Enable Bit** 0 = Receive time-out interrupt Disabled. 1 = Receive time-out interrupt Enabled.
[3]	**TXTHIEN**	**Transmit FIFO Threshold Interrupt Enable Bit** 0 = TX FIFO threshold interrupt Disabled. 1 = TX FIFO threshold interrupt Enabled.
[2]	**RXTHIEN**	**Receive FIFO Threshold Interrupt Enable Bit** 0 = RX FIFO threshold interrupt Disabled. 1 = RX FIFO threshold interrupt Enabled.
[1]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset transmit FIFO pointer and transmit circuit. The TXFULL bit will be cleared to 0 and the TXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (LPSPI_STATUS[23]) to check if reset is accomplished or not. **Note:** If TX underflow event occurs in SPI Slave mode, this bit can be used to make SPI return to idle state.
[0]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset receive FIFO pointer and receive circuit. The RXFULL bit will be cleared to 0 and the RXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (LPSPI_STATUS[23]) to check if reset is accomplished or not.

LPSPI Status Register (LPSPI_STATUS)

Register	Offset	R/W	Description	Reset Value
LPSPI_STATUS	LPSPI_BA+0x14	R/W	SPI Status Register	0x0005_0110

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	Reserved	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

Bits	Description	
[31:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**TX or RX Reset Status (Read Only)** 0 = The reset function of TXRST or RXRST is done. 1 = Doing the reset function of TXRST or RXRST. **Note:** Both the reset operations of TXRST and RXRST need 3 system clock cycles + 2 peripheral clock cycles. User can check the status of this bit to monitor the reset function is doing or done.
[22:20]	**Reserved**	Reserved.
[19]	**TXUFIF**	**TX Underflow Interrupt Flag** When the TX underflow event occurs, this bit will be set to 1, the state of data output pin depends on the setting of TXUFPOL. 0 = No effect. 1 = No data in Transmit FIFO and TX shift register when the slave selection signal is active. **Note 1:** This bit will be cleared by writing 1 to it. **Note 2:** If reset slave's transmission circuit when slave selection signal is active, this flag will be set to 1 after 2 peripheral clock cycles + 3 system clock cycles since the reset operation is done.
[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TXTH. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TXTH.
[17]	**TXFULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.

[16]	**TXEMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[15]	**SPIENSTS**	**SPI Enable Status (Read Only)** 0 = LPSPI controller Disabled. 1 = LPSPI controller Enabled. **Note:** The SPI peripheral clock is asynchronous with the system clock. In order to make sure the SPI control logic is disabled, this bit indicates the real status of LPSPI controller.
[14:13]	**Reserved**	Reserved.
[12]	**RXTOIF**	**Receive Time-out Interrupt Flag** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 SPI peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. **Note:** This bit will be cleared by writing 1 to it.
[11]	**RXOVIF**	**Receive FIFO Overrun Interrupt Flag** When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1. 0 = No FIFO is overrun. 1 = Receive FIFO is overrun. **Note:** This bit will be cleared by writing 1 to it.
[10]	**RXTHIF**	**Receive FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the receive FIFO buffer is smaller than or equal to the setting value of RXTH. 1 = The valid data count within the receive FIFO buffer is larger than the setting value of RXTH.
[9]	**RXFULL**	**Receive FIFO Buffer Full Indicator (Read Only)** 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[8]	**RXEMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[7]	**SLVURIF**	**Slave Mode TX Under Run Interrupt Flag** In Slave mode, if TX underflow event occurs and the slave select line goes to inactive state, this interrupt flag will be set to 1. 0 = No Slave TX under run event. 1 = Slave TX under run event occurred. **Note:** This bit will be cleared by writing 1 to it.
[6]	**SLVBEIF**	**Slave Mode Bit Count Error Interrupt Flag** In Slave mode, when the slave select line goes to inactive state, if bit counter is mismatch with DWIDTH, this interrupt flag will be set to 1. 0 = No Slave mode bit count error event. 1 = Slave mode bit count error event occurred. **Note:** If the slave select active but there is no any bus clock input, the SLVBEIF also active when the slave select goes to inactive state. This bit will be cleared by writing 1 to it.
[5]	**Reserved**	Reserved.
[4]	**SSLINE**	**Slave Select Line Bus Status (Read Only)**

		0 = The slave select line status is 0. 1 = The slave select line status is 1. **Note:** This bit is only available in Slave mode. If SSACTPOL (LPSPI_SSCTL[2]) is set 0, and the SSLINE is 1, the SPI slave select is in inactive status.
[3]	**SSINAIF**	**Slave Select Inactive Interrupt Flag** 0 = Slave select inactive interrupt was cleared or not occurred. 1 = Slave select inactive interrupt event occurred. **Note:** Only available in Slave mode. This bit will be cleared by writing 1 to it.
[2]	**SSACTIF**	**Slave Select Active Interrupt Flag** 0 = Slave select active interrupt was cleared or not occurred. 1 = Slave select active interrupt event occurred. **Note:** Only available in Slave mode. This bit will be cleared by writing 1 to it.
[1]	**UNITIF**	**Unit Transfer Interrupt Flag** 0 = No transaction has been finished since this bit was cleared to 0. 1 = LPSPI controller has finished one unit transfer. **Note:** This bit will be cleared by writing 1 to it.
[0]	**BUSY**	**Busy Status (Read Only)** 0 = LPSPI controller is in idle state. 1 = LPSPI controller is in busy state. The following lists the bus busy conditions: f. SPIEN (LPSPI_CTL[0]) = 1 and TXEMPTY = 0. g. For SPI Master mode, SPIEN (LPSPI_CTL[0]) = 1 and TXEMPTY = 1 but the current transaction is not finished yet. h. For SPI Master mode, SPIEN (LPSPI_CTL[0]) = 1 and RXONLY = 1. i. For SPI Slave mode, SPIEN (LPSPI_CTL[0]) = 1 and there is serial clock input into the SPI core logic when slave select is active. **j.** For SPI Slave mode, SPIEN (LPSPI_CTL[0]) = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive. **Note:** By applications, this SPI busy flag should be used with other status registers in LPSPI_STATUS such as TXCNT, RXCNT, TXTHIF, TXFULL, TXEMPTY, RXTHIF, RXFULL, RXEMPTY, and UNITIF. Therefore the SPI transfer done events of TX/RX operations can be obtained at correct timing point.

LPSPI Status2 Register (LPSPI_STATUS2)

Register	Offset	R/W	Description	Reset Value
LPSPI_STATUS2	LPSPI_BA+0x18	R	SPI Status2 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		**SLVBENUM**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**SLVBENUM**	**Effective Bit Number of Uncompleted RX data** This status register indicates that effective bit number of uncompleted RX data when SLVBERX (LPSPI_FIFOCTL[10]) is enabled and RX bit count error event happen in SPI Slave mode. This status register will be fixed to 0x0 when SLVBERX (LPSPI_FIFOCTL[10]) is disabled. **Note:** This register will be cleared to 0x0 when user writes 0x1 to SLVBEIF (LPSPI_STATUS[6]). **Note:** This bit is for Slave Mode only.
[23:0]	**Reserved**	Reserved.

LPSPI Data Transmit Register (LPSPI_TX)

Register	Offset	R/W	Description	Reset Value
LPSPI_TX	LPSPI_BA+0x20	W	SPI Data Transmit Register	0x0000_0000

31	30	29	28	27	26	25	24
TX							
23	22	21	20	19	18	17	16
TX							
15	14	13	12	11	10	9	8
TX							
7	6	5	4	3	2	1	0
TX							

Bits	Description	
[31:0]	**TX**	**Data Transmit Register** The data transmit registers pass through the transmitted data into the 4-level transmit FIFO buffers. The number of valid bits depends on the setting of DWIDTH (LPSPI_CTL[12:8]) in SPI mode. In SPI mode, if DWIDTH is set to 0x08, the bits TX[7:0] will be transmitted. If DWIDTH is set to 0x00, the LPSPI controller will perform a 32-bit transfer. **Note:** In Master mode, LPSPI controller will start to transfer the SPI bus clock after 1 APB clock and 6 peripheral clock cycles after user writes to this register.

LPSPI Data Receive Register (LPSPI_RX)

Register	Offset	R/W	Description	Reset Value
LPSPI_RX	LPSPI_BA+0x30	R	SPI Data Receive Register	0x0000_0000

31	30	29	28	27	26	25	24
RX							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RX							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RX							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX							

Bits	Description	
[31:0]	**RX**	**Data Receive Register (Read Only)** There are 4-level FIFO buffers in this controller. The data receive register holds the data received from SPI data input pin. If the RXEMPTY (LPSPI_STATUS[8]) is not set to 1, the receive FIFO buffers can be accessed through software by reading this register.

LPSPI Automatic Operation Control Register (LPSPI_AUTOCTL)

Register	Offset	R/W	Description	Reset Value
LPSPI_AUTOCTL	LPSPI_BA+0x50	R/W	LPSPI Automatic Operation Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TCNT							
15	14	13	12	11	10	9	8
Reserved					CNTWKEN	SWTRIG	AUTOEN
7	6	5	4	3	2	1	0
SSWKEN	FULLRXEN	CNTIEN	TRIGEN	TRIGSEL			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**TCNT**	**Auomatic Operation RX Transfer Count** In Auomatic Operation Master mode, TCNT represents the required number of RX data received from external SPI slave device while in RX Phase. The maximum transfer count is 255.
[15:11]	**Reserved**	Reserved.
[10]	**CNTWKEN**	**TCNT Count Match Wake Up Enable Bit** 0 = TCNT count match wake-up function Disabled. 1 = TCNT count match wake-up function Enabled. **Note:** If TCNT is set to 0, TCNT count match wake-up function will remain disabled even if CNTWKEN is set to 1.
[9]	**SWTRIG**	**Software Trigger (Write Only)** After AUTOEN set to 1, software can manually trigger the Automatic Operation Master mode by writing 1 to this bit.
[8]	**AUTOEN**	**Automatic Operation Mode Enable Bit** 0 = Automatic operation master mode Disabled. 1 = Automatic operation master mode Enabled.
[7]	**SSWKEN**	**Slave Select Wake Up Enable Bit** In Slave mode, the CPU will be woken up by SS falling or rising edge in NPDx mode after this bit is set to 1. 0 = LPSPI SS wake-up function Disabled. 1 = LPSPI SS wake-up function Enabled.
[6]	**FULLRXEN**	**Full RX Data Acception Enable Bit** In Auomatic Operation Master mode and LPSPI operates in full-duplex mode, the RX data will be saved from RX buffer by LPPDMA while in TX Phase after this bit is set to 1.
[5]	**CNTIEN**	**TCNT Count Match Interrupt Enable** 0 = TCNT count match interrupt Disabled. 1 = TCNT count match interrupt Enabled.

[4]	**TRIGEN**	**Automatic Operation Trigger Enable Bit** In Auomatic Operation Master mode, the automatic operation of LPSPI will be triggered by an event sent from the trigger source selected by TRIGSEL[3:0] after this bit is set to 1. 0 = LPSPI Automatic Operation Trigger Disabled. 1 = LPSPI Automatic Operation Trigger Enabled.
[3:0]	**TRIGSEL**	**Automatic Operation Trigger Source Select** 0000 = Low Power Auto-operation Trigger Source from LPTMR0. 0001 = Low Power Auto-operation Trigger Source from LPTMR1. 0010 = Low Power Auto-operation Trigger Source from TTMR0. 0011 = Low Power Auto-operation Trigger Source from TTMR1. 0100 = Low Power Auto-operation Trigger Source from LPGPIO0 (PA.0). 0101 = Low Power Auto-operation Trigger Source from LPGPIO1 (PB.0). 0110 = Low Power Auto-operation Trigger Source from LPGPIO2 (PC.0). 0111 = Low Power Auto-operation Trigger Source from LPGPIO3 (PD.0). 1000~1111 = Reserved.

LPSPI Automatic Operation Status Register (LPSPI_AUTOSTS)

Register	Offset	R/W	Description	Reset Value
LPSPI_AUTOSTS	LPSPI_BA+0x54	R/W	LPSPI Automatic Operation Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				CNTWKF	AOBUSY	SSWKF	CNTIF

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**CNTWKF**	**TCNT Count Match Wake Up Flag** When chip is woken up due to the received data count matching the setting value of TCNT, this bit will be set to 1. Software can write 1 to clear this bit.
[2]	**AOBUSY**	**Automatic Operation Busy Flag** When there were one or more requests from trigger sources during auto operation, this bit will be set to 1. Software can write 1 to clear this bit.
[1]	**SSWKF**	**Slave Select Wake Up Flag** In Slave mode, when chip is woken up from NPDx mode by LPSPI, this bit is set to 1. Software can write 1 to clear this bit.
[0]	**CNTIF**	**TCNT Count Match Interrupt Flag** When the received data count matchs the setting value of TCNT, this bit will be set to 1. Software can write 1 to clear this bit. 0 = The received data count is less than the setting value of TCNT. 1 = The received data count is equal than the setting value of TCNT.

6.31 Quad Serial Peripheral Interface (QSPI)

6.31.1 Overview

The Quad Serial Peripheral Interface (QSPI) applies to synchronous serial data communication and allows full duplex transfer. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The chip contains one QSPI controller performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device.

The QSPI controller supports 2-bit Transfer mode to perform full-duplex 2-bit data transfer and also supports Dual and Quad I/O Transfer mode and the controller supports the PDMA function to access the data buffer.

6.31.2 Features

- Supports Master or Slave mode operation
- Master mode up to 36 MHz (when chip works at V_{DD} = 2.7V~3.6V)
- Slave mode up to 24 MHz (when chip works at V_{DD} = 2.7V~3.6V)
- Supports 2-bit Transfer mode
- Supports Dual and Quad I/O Transfer mode
- Configurable bit length of a transaction word from 8 to 32-bit
- Provides separate 8-level depth transmit and receive FIFO buffers
- Supports MSB first or LSB first transfer sequence
- Supports Byte Reorder function
- Supports Byte or Word Suspend mode
- Supports PDMA transfer
- Supports 3-Wire, no slave selection signal, bi-direction interface
- Supports one data channel half-duplex transfer
- Supports Transmit Double Transfer Rate Mode (TX DTR mode)
- Supports receive-only mode

6.31.3 Block Diagram

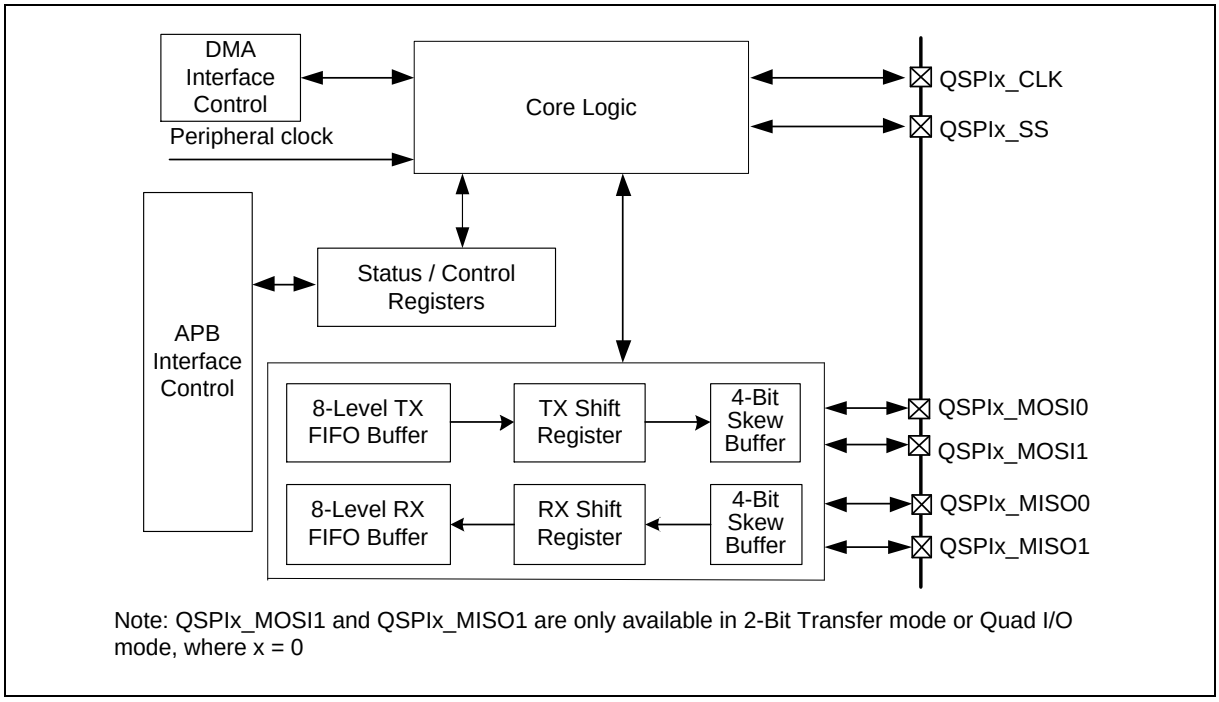


Figure 6.31-1 QSPI Block Diagram

TX FIFO Buffer

The transmit FIFO buffer is a 8-level depth, 32-bit wide, first-in, first-out register buffer. The data can be written to the transmit FIFO buffer in advance through software by writing the QSPIx_TX register.

RX FIFO Buffer

The receive FIFO buffer is also a 8-level depth, 32-bit wide, first-in, first-out register buffer. The receive control logic will store the receive data to this buffer. The FIFO buffer data can be read from QSPIx_RX register by software.

TX Shift Register

The transmit shift register is a 32-bit wide register buffer. The transmit data is loaded from the TX FIFO buffer and shifted out bit-by-bit to the skew buffer.

RX Shift Register

The receive shift register is also a 32-bit wide register buffer. The receive data is shift in bit-by-bit from the skew buffer and is loaded into RX FIFO buffer when a transaction done.

Skew Buffer

The skew buffer is a 4-level 1-bit buffer. There are two skew buffers in transmitting and received side. In received side, it is used to shift bits into RX shift register from QSPI bus. In transmitting side, it is used to shift bits into QSPI bus from TX shift register.

6.31.4 Basic Configuration

6.31.4.1 QSPI0 Basic Configuration

- Clock source Configuration
  - Select the source of QSPI0 peripheral clock on QSPI0SEL (CLK_CLKSEL2[3:2]).
  - Enable QSPI0 peripheral clock in QSPI0CKEN (CLK_APBCLK0[12]).

- Reset Configuration
  - Reset QSPI0 controller in QSPI0RST (SYS_IPRST1[12]).

6.31.5 Functional Description

6.31.5.1 Terminology

QSPI Peripheral Clock and QSPI Bus Clock

The QSPI controller needs the peripheral clock to drive the QSPI logic unit to perform the data transfer. The peripheral clock rate is determined by the settings of clock divisor (QSPIx_CLKDIV) and the clock source which can be HXT, PLL, PCLK or HIRC. QSPIxSEL of CLK_CLKSEL2 register determines the clock source of the peripheral clock. The DIVIDER (QSPIx_CLKDIV[8:0]) setting determines the divisor of the clock rate calculation.

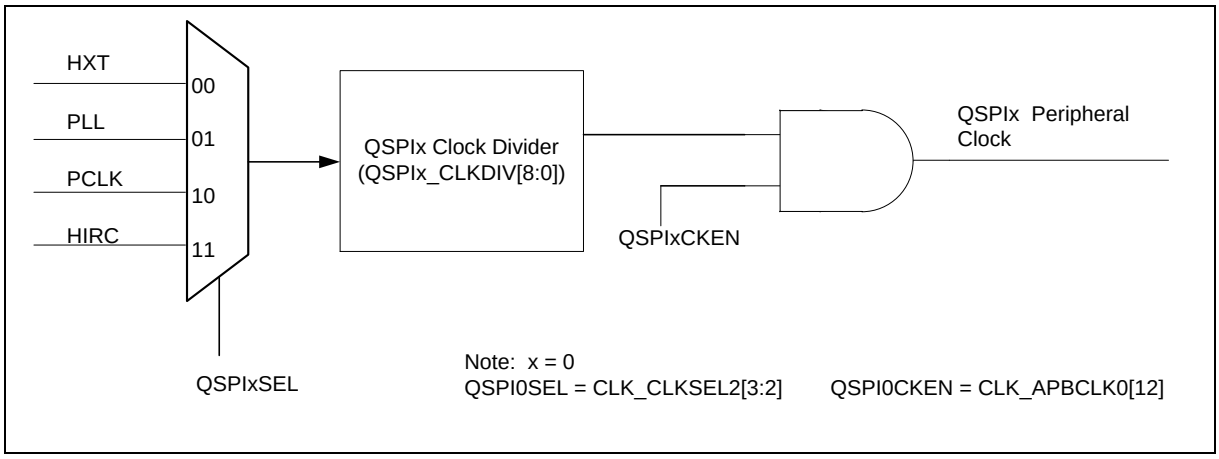


Figure 6.31-2 QSPI Peripheral Clock

In Master mode, the frequency of the QSPI bus clock is equal to the peripheral clock rate. In general, the QSPI bus clock is denoted as QSPI clock. In Slave mode, the QSPI bus clock is provided by a master device. The frequency of QSPI peripheral clock cannot be faster than the system clock rate regardless of Master or Slave mode. If the clock source of peripheral clock is not system clock, the frequency of QSPI peripheral clock shall be slower than the system clock frequency regardless of Master or Slave mode.

Master/Slave mode

The QSPI controllers can be set as Master or Slave mode by setting the SLAVE (QSPIx_CTL[18]) to communicate with the off-chip SPI slave or master device. The HALFDPX (QSPIx_CTL[14]) can be used to select the full-duplex or half-duplex in QSPI transmission. The application block diagrams in Master and Slave mode are shown below.

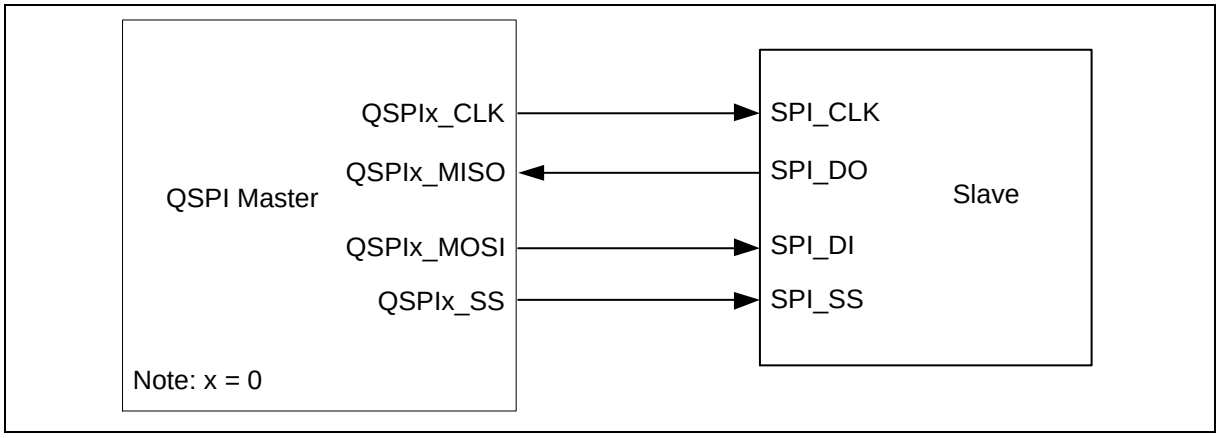


Figure 6.31-3 QSPI Full-Duplex Master Mode Application Block Diagram

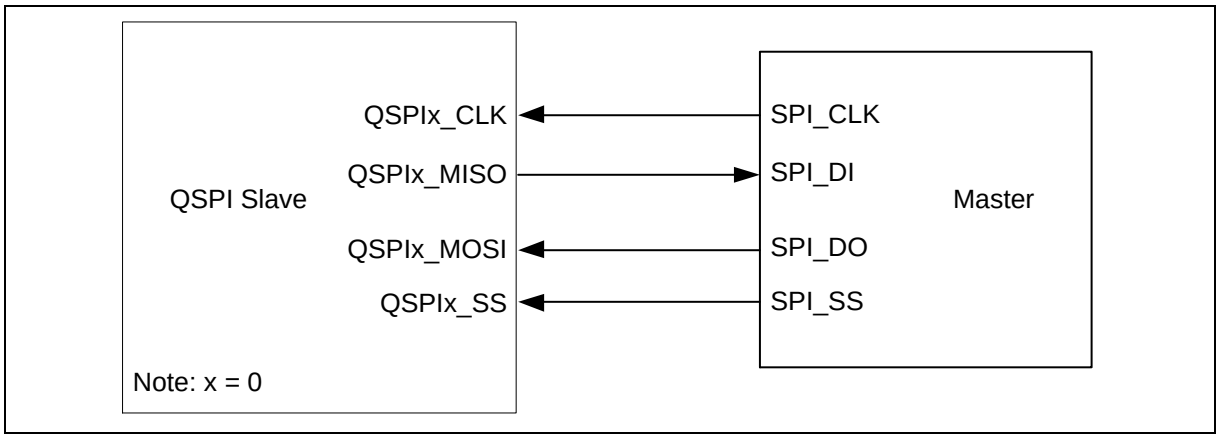


Figure 6.31-4 QSPI Full-Duplex Slave Mode Application Block Diagram

Slave Selection

In Master mode, the QSPI controller can drive off-chip slave device through the slave select output pin QSPIx_SS. In Slave mode, the off-chip master device drives the slave selection signal from the QSPIx_SS input port to this QSPI controller. The duration between the slave select active edge and the first QSPI clock input shall over 3 QSPI peripheral clock cycles of slave.

In Master/Slave mode, the active state of slave selection signal can be programmed to low or high active in SSACTPOL (QSPIx_SSCTL[2]). The selection of slave select conditions depends on what type of device is connected. In Slave mode, to recognize the inactive state of the slave selection signal, the inactive period of the slave selection signal must be larger than or equal to 3 peripheral clock cycles between two successive transactions.

In Slave mode, the QSPIx_MISO signal type is controlled by the QSPIx_SS pin. If the QSPIx_SS pin is inactive state, the QSPIx_MISO is set to tri-state to avoid interference with other Slave devices. If the QSPIx_SS pin is active state, the QSPIx_MISO is set to output mode for data transfer.

Timing Condition

The CLKPOL (QSPIx_CTL[3]) defines the QSPI clock idle state. If CLKPOL = 1, the output QSPI clock is idle at high state; if CLKPOL = 0, it is idle at low state.

TXNEG (QSPIx_CTL[2]) defines the data transmitted out either on negative edge or on positive edge of QSPI clock. RXNEG (QSPIx_CTL[1]) defines the data received either on negative edge or on positive edge of QSPI clock.

Note: The settings of TXNEG and RXNEG are mutually exclusive. In other words, do not transmit and

receive data at the same clock edge.

Transmit/Receive Bit Length

The bit length of a transaction word is defined in DWIDTH (QSPIx_CTL[12:8]) and can be configured up to 32-bit length in a transaction word for transmitting and receiving.

When QSPI controller finishes a transaction, i.e. receives or transmits a specific count of bits defined in DWIDTH (QSPIx_CTL[12:8]), the unit transfer interrupt flag UNITIF (QSPIx_STATUS[1]) will be set to 1.

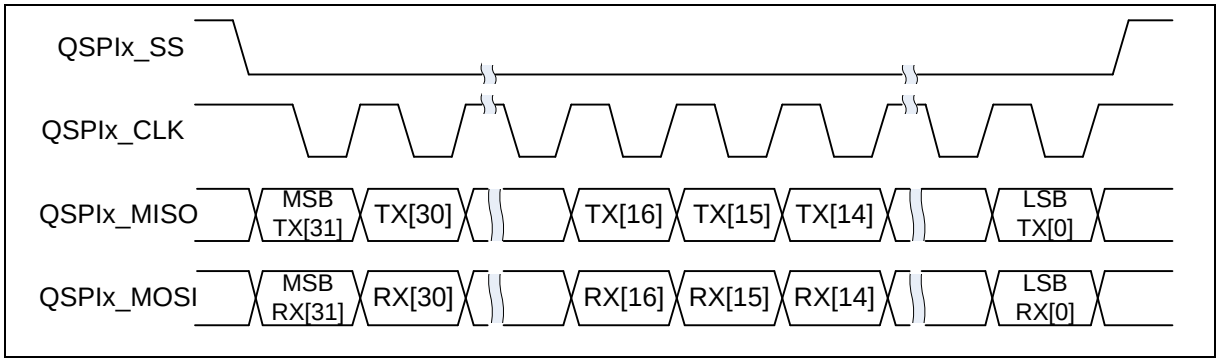


Figure 6.31-5 32-bit in One Transaction

LSB/MSB First

LSB (QSPIx_CTL[13]) defines the bit transfer sequence in a transaction. If the LSB (QSPIx_CTL[13]) is set to 1, the transfer sequence is LSB first. The bit 0 will be transferred firstly. If the LSB (QSPIx_CTL[13]) is cleared to 0, the transfer sequence is MSB first.

Suspend Interval

SUSPITV (QSPIx_CTL[7:4]) provides a configurable suspend interval, 0.5 ~ 15.5 QSPI clock periods, between two successive transaction words in Master mode. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SUSPITV is 0x3 (3.5 QSPI clock cycles).

6.31.5.2 Automatic Slave Selection

In Master mode, if AUTOSS (QSPIx_SSCTL[3]) is set, the slave selection signal will be generated automatically and output to the QSPIx_SS pin according to whether SS (QSPIx_SSCTL[0]) is enabled or not. The slave selection signal will be set to active state by the QSPI controller when the QSPI data transfer is started by writing to FIFO. It will be set to inactive state when QSPI bus is idle. If QSPI bus is not idle, i.e. TX FIFO, TX shift register or TX skew buffer is not empty, the slave selection signal will be set to inactive state between transactions if the value of SUSPITV (QSPIx_CTL[7:4]) is greater than or equal to 3.

In Master mode, if the value of SUSPITV is less than 3 and the AUTOSS is set as 1, the slave selection signal will be kept at active state between two successive transactions.

If the AUTOSS bit is cleared, the slave selection output signal will be determined by the SS setting. The active state of the slave selection output signal is specified in SSACTPOL (QSPIx_SSCTL[2]).

The duration between the slave selection signal active edge and the first QSPI bus clock edge is 1 QSPI bus clock cycle and the duration between the last QSPI bus clock and the slave selection signal inactive edge is 1.5 QSPI bus clock cycle.

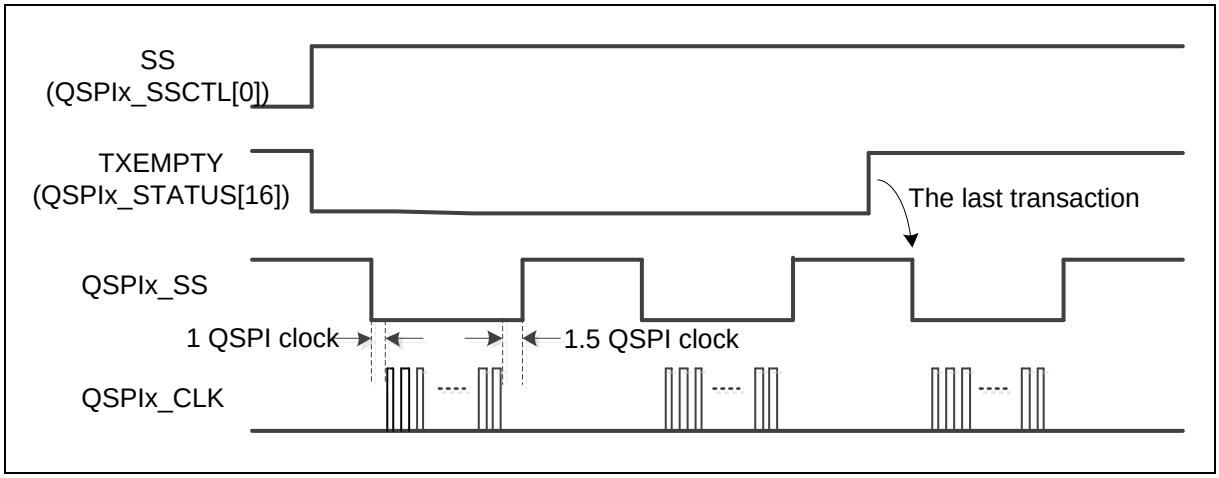


Figure 6.31-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2)

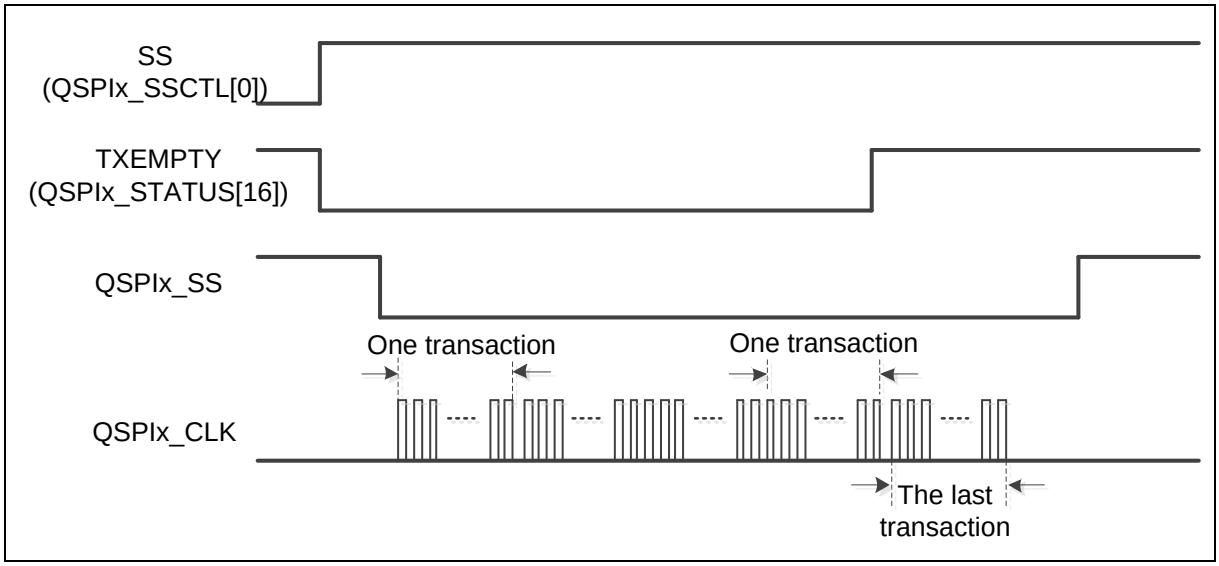


Figure 6.31-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3)

6.31.5.3 Byte Reorder and Suspend Function

When the transfer is set as MSB first (LSB = 0) and the REORDER (QSPIx_CTL[19]) is set to 1, the data stored in the TX buffer and RX buffer will be rearranged in the order as [Byte0, Byte1, Byte2, Byte3] in 32-bit transfer (DWIDTH = 0). The sequence of transmitted/received data will be Byte0, Byte1, Byte2, and then Byte3. If the DWIDTH is set as 24-bit transfer mode, the data in TX buffer and RX buffer will be rearranged as [unknown byte, Byte0, Byte1, Byte2]. The QSPI controller will transmit/receive data with the sequence of Byte0, Byte1 and then Byte2. Each byte will be transmitted/received with MSB first. The rule of 16-bit mode is the same as above. Byte Reorder function is only available when DWIDTH is configured as 16, 24, and 32 bits.

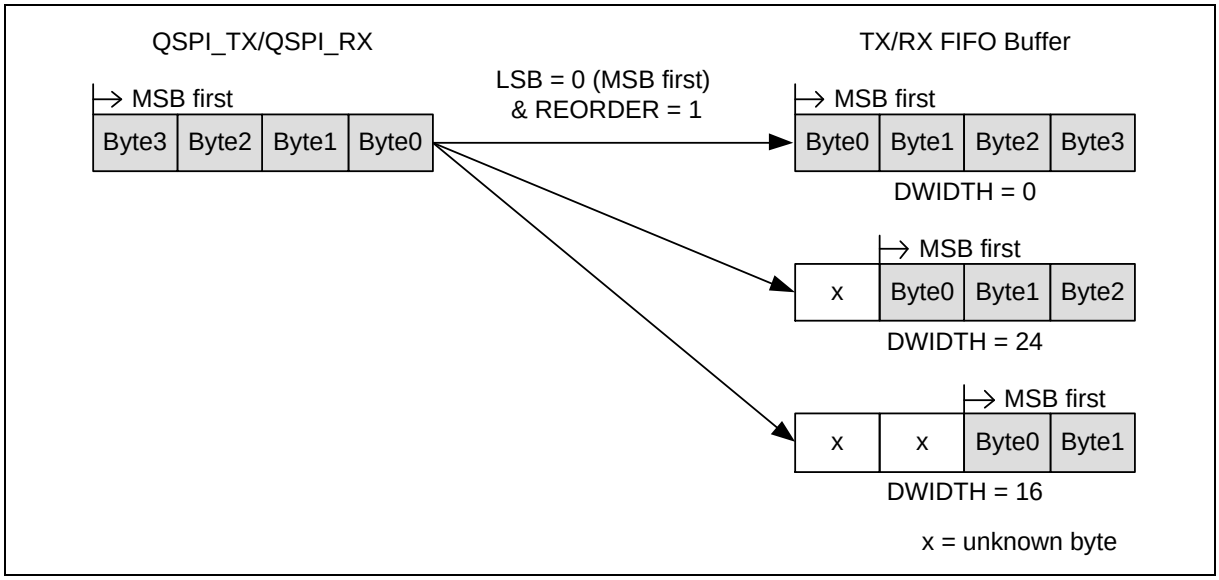


Figure 6.31-8 Byte Reorder Function

In Master mode, if REORDER (QSPIx_CTL[19]) is set to 1, a suspend interval of 0.5 ~ 15.5 QSPI clock periods will be inserted by hardware between two successive bytes in a transaction word. The suspend interval is configured in SUSPITV (QSPIx_CTL[7:4]).

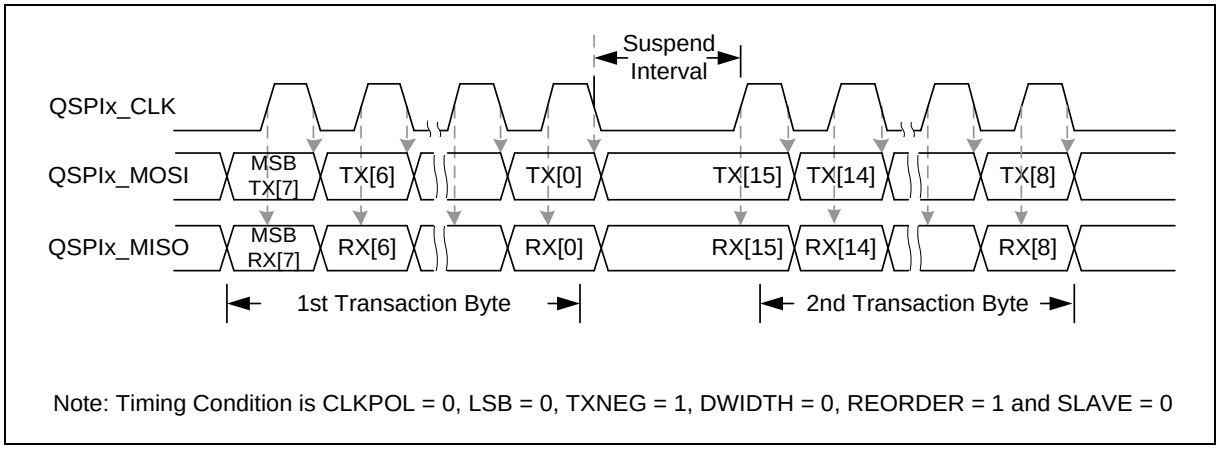


Figure 6.31-9 Timing Waveform for Byte Suspend

6.31.5.4 Half-Duplex Communication

The QSPI controller can communicate in half-duplex mode by setting HALFDPX (QSPIx_CTL[14]) bit. In half-duplex mode, there is only one data line for receiving or transmitting data direction which is defined by DATDIR (QSPIx_CTL[20]). In half-duplex configuration, the QSPIx_MISO pin is free for other applications and it can be configured as GPIO. Enabling or disabling the control bit HALFDPX (QSPIx_CTL[14]) will produce TXFBCLR (QSPIx_FIFOCTL[9]) and RXFBCLR (QSPIx_FIFOCTL[8]) at the same time automatically.

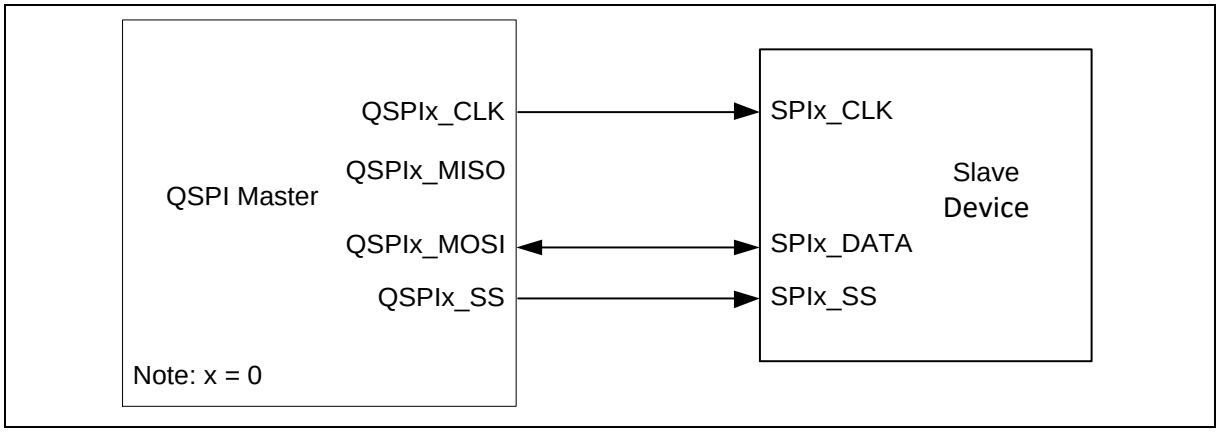


Figure 6.31-10 QSPI Half-Duplex Master Mode Application Block Diagram

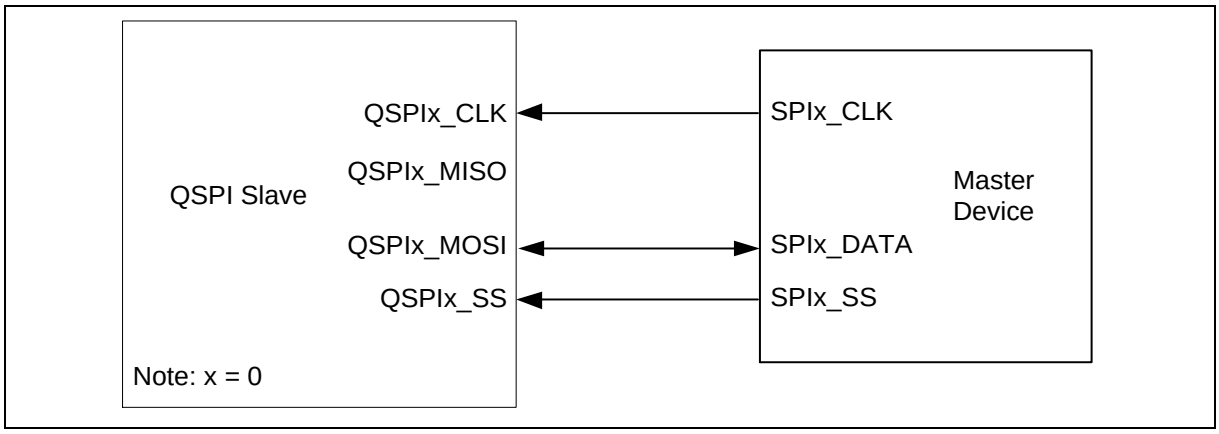


Figure 6.31-11 QSPI Half-Duplex Slave Mode Application Block Diagram

6.31.5.5 Receive-Only Mode

In QSPI Master device, it can communicate in receive-only mode by setting RXONLY (QSPIx_CTL[15]). In this configuration, the QSPI Master device will generate QSPI bus clock continuously as long as the receive-only mode is enabled for receiving data bit from SPI slave device. If AUTOSS (QSPIx_SSCTL[3]) is enabled in receive-only mode, QSPI Master will keep activating the slave select signal.

The remaining QSPIx_MOSI pin of QSPI Master device is not used for communication and can be configured as GPIO. The status BUSY (QSPIx_STATUS[0]) will be asserted in receive-only mode due to the generation of QSPI bus clock. Entering this mode will produce the TXFBCLR (QSPIx_FIFOCTL[9]) and RXFBCLR (QSPIx_FIFOCTL[8]) at the same time automatically. When user enables this mode, the output QSPI bus clock will be sent out after 6 peripheral clock cycles. In this mode, the data which has been written into transmit FIFO will be loaded into transmit shift register and sent out.

When user sets RXONLY (QSPIx_CTL[15]) to enable, QSPI RX data with data bit width of DWIDTH (QSPIx_CTL[12:8]) will be received into RX FIFO and QSPI clock will be sent to SPI slave device until RX FIFO is full.

For two-bit transfer mode TWOBIT (QSPIx_CTL[16]) is enabled, the QSPI master will send QSPI output clock to SPI slave and receive RX data until RX FIFO is full. After user reads RX data from RX FIFO, QSPI master will send QSPI output clock to SPI slave and receive RX data again when RX FIFO counter RXCNT is less than or equal to 4.

6.31.5.6 Slave 3-Wire Mode

When SLV3WIRE (QSPIx_SSCTL[4]) is set by software to enable the Slave 3-Wire mode, the QSPI controller can work with no slave selection signal in Slave mode. The SLV3WIRE (QSPIx_SSCTL[4]) only takes effect in Slave mode. Only three pins, QSPIx_CLK, QSPIx_MISO, and QSPIx_MOSI, are required to communicate with a SPI master. The QSPIx_SS pin can be configured as a GPIO. When the SLV3WIRE (QSPIx_SSCTL[4]) is set to 1, the QSPI slave will be ready to transmit/receive data after the SPIEN (QSPIx_CTL[0]) is set to 1.

6.31.5.7 PDMA Transfer Function

QSPI controller supports PDMA transfer function.

When TXPDMAEN (QSPIx_PDMACTL[0]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (QSPIx_PDMACTL[1]) is set to 1, the controller will start the PDMA reception process. QSPI controller will issue request to PDMA controller automatically when there is data in the RX FIFO buffer.

Note: QSPI supports single request PDMA (Read/Write) only, burst request PDMA is not supported.

6.31.5.8 Two-bit Transfer Mode

The QSPI controller also supports 2-bit Transfer mode when setting TWOBIT (QSPIx_CTL[16]) to 1. In 2-bit Transfer mode, the QSPI controller performs full duplex data transfer. In other words, the two serial data bits can be transmitted and received simultaneously.

For example, in Master mode, the even data (TX Data (n)) stored in the TX FIFO of QSPIx will be transmitted through the QSPIx_MOSI0 pin and the odd data (TX Data (n+1)) stored in the QSPIx_TX register will be transmitted through the QSPIx_MOSI1 pin respectively. In the meanwhile, the even data received from QSPIx_MISO0 pin will be written to RX FIFO prior to the odd data received from QSPIx_MISO1 pin.

In Slave mode, the even and odd data stored in the QSPIx_TX register will be transmitted through the QSPIx_MISO0 pin and QSPIx_MISO1 pin respectively. In the meanwhile, the RX FIFO of QSPIx will store the even data received from the QSPIx_MOSI0 pin and the odd data from QSPIx_MOSI1 pin respectively. The data sequence of FIFO buffers is the same as the Master mode.

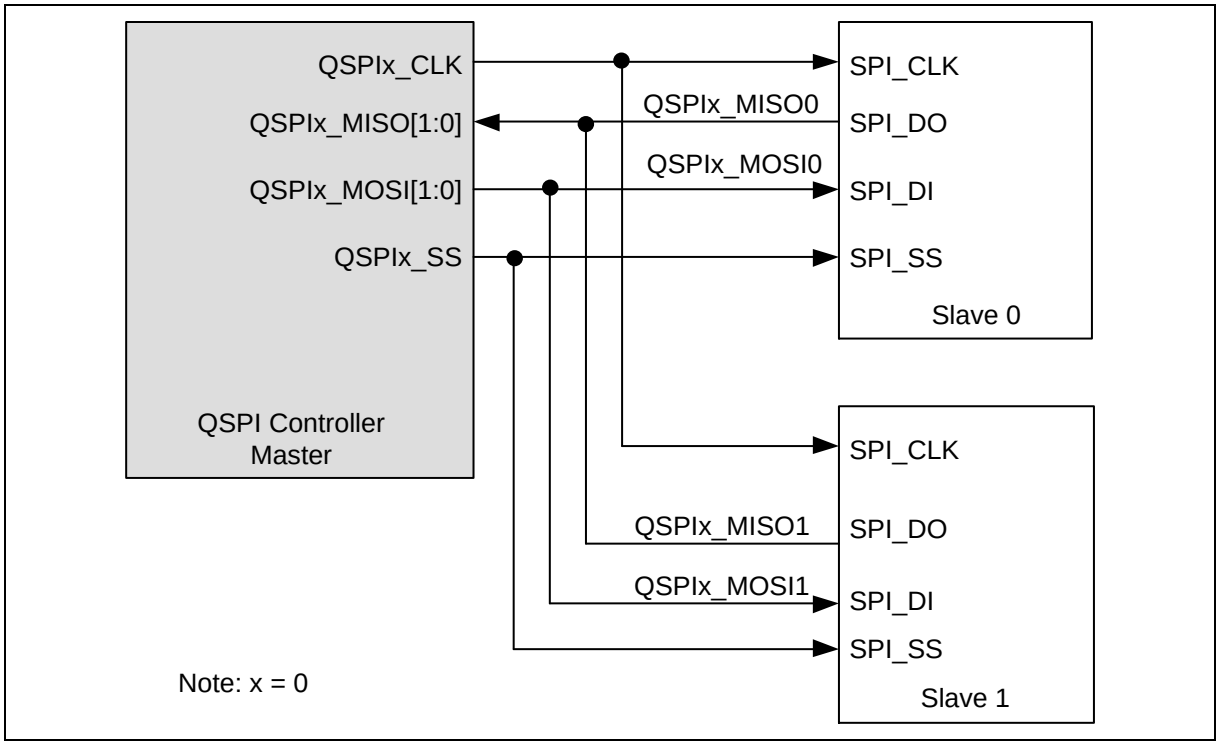


Figure 6.31-12 Two-bit Transfer Mode System Architecture

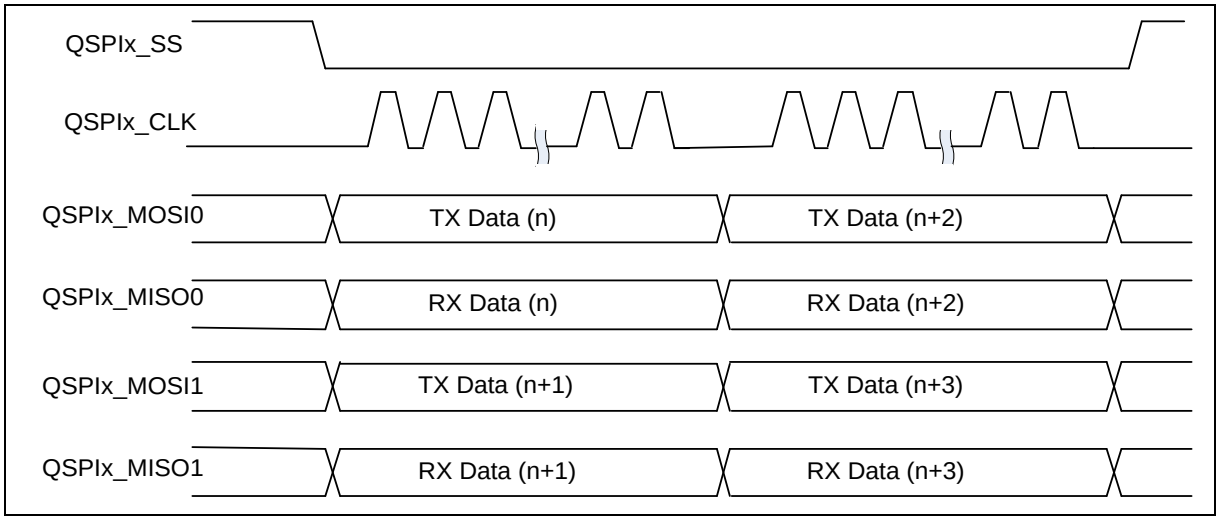


Figure 6.31-13 Two-bit Transfer Mode Timing (Master Mode)

6.31.5.9 Dual I/O Mode

The QSPI controller also supports Dual I/O transfer when setting the DUALIOEN (QSPIx_CTL[21]) to 1. Many general SPI Flashes support Dual I/O transfer. The DATDIR (QSPIx_CTL[20]) is used to define the direction of the transfer data. When the DATDIR bit is set to 1, the controller will send the data to external device. When the DATDIR bit is set to 0, the controller will read the data from the external device. This function supports 8, 16, 24, and 32 bits of length.

The Dual I/O mode is not supported when the Slave 3-Wire mode or the Byte Reorder function is enabled.

For Dual I/O mode, if both the DUALIOEN (QSPIx_CTL[21]) and DATDIR (QSPIx_CTL[20]) are set as

1, the QSPIx_MOSI0 is the even bit data output and the QSPIx_MISO0 will be set as the odd bit data output. If the DUALIOEN (QSPIx_CTL[21]) is set as 1 and DATDIR (QSPIx_CTL[20]) is set as 0, both the QSPIx_MISO0 and QSPIx_MOSI0 will be set as data input ports.

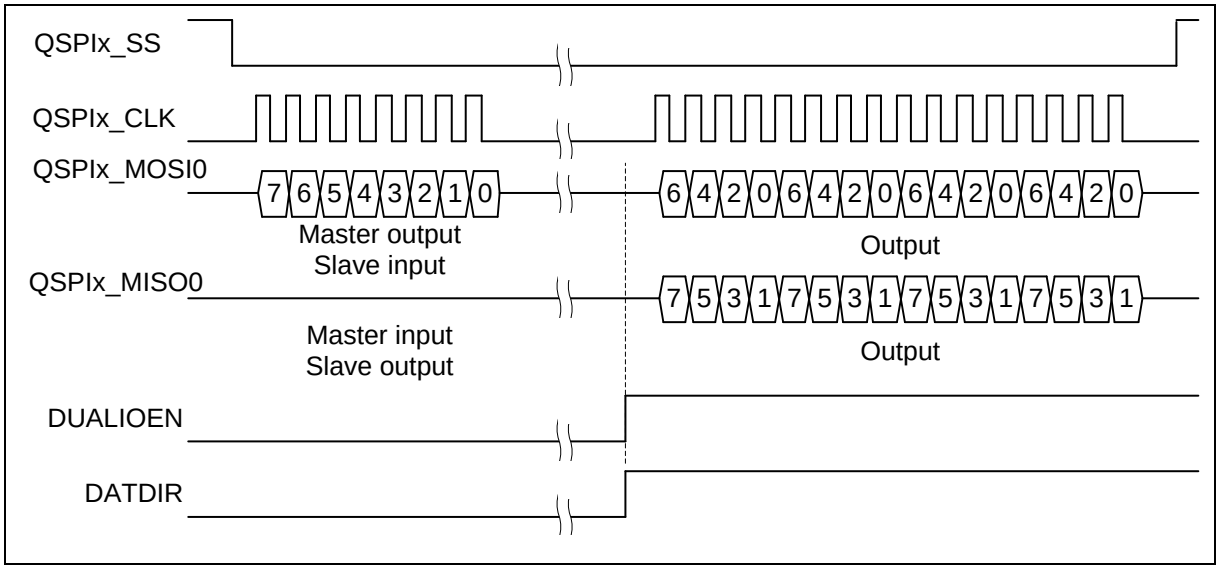


Figure 6.31-14 Bit Sequence of Dual Output Mode

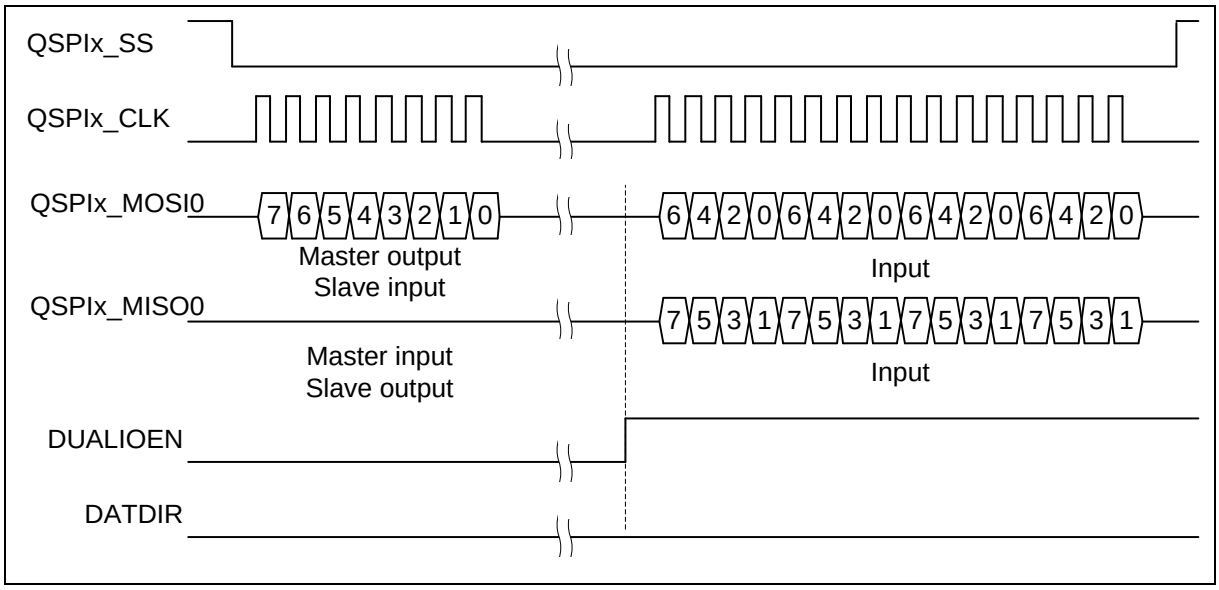


Figure 6.31-15 Bit Sequence of Dual Input Mode

6.31.5.10 Quad I/O Mode

The QSPI controller also supports Quad I/O transfer when setting the QUADIOEN (QSPIx_CTL[22]) to 1. Many general SPI Flashes support Quad I/O transfer. The DATDIR bit (QSPIx_CTL[20]) is used to define the direction of the transfer data. When the DATDIR (QSPIx_CTL[20]) is set to 1, the controller will send the data to external device. When the DATDIR (QSPIx_CTL[20]) is set to 0, the controller will read the data from the external device. This function supports 8, 16, 24, and 32 bits of length.

The Quad I/O mode is not supported when the Slave 3-Wire mode or the Byte Reorder function is enabled. The DUALIOEN (QSPIx_CTL[21]) and QUADIOEN (QSPIx_CTL[22]) shall not be set to 1 simultaneously.

For Quad I/O mode, if both the QUADIOEN (QSPIx_CTL[22]) and DATDIR (QSPIx_CTL[20]) are set as 1, the QSPIx_MOSI0 and QSPIx_MOSI1 are the even bit data output and the QSPIx_MISO0 and QSPIx_MISO1 will be set as the odd bit data output. If the QUADIOEN (QSPIx_CTL[22]) is set as 1 and DATDIR (QSPIx_CTL[20]) is set as 0, all the QSPIx_MISO0, QSPIx_MISO1, QSPIx_MOSI0 and QSPIx_MOSI1 pins will be set as data input ports.

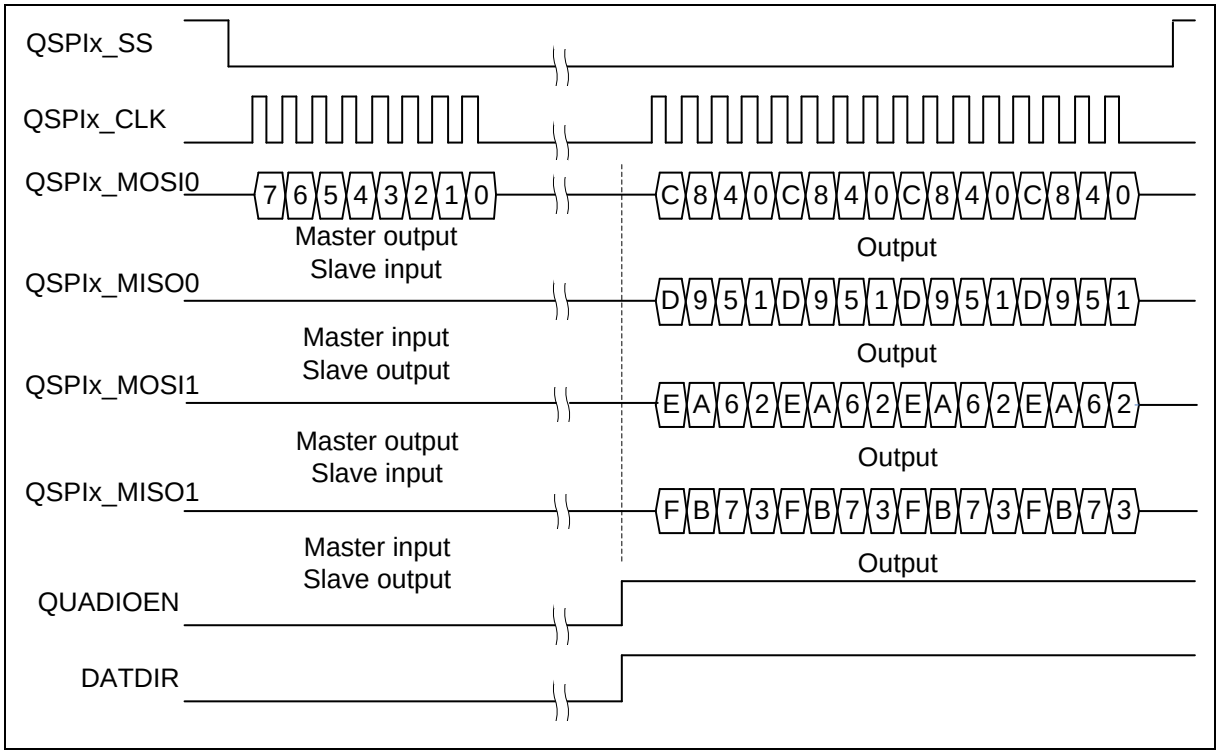


Figure 6.31-16 Bit Sequence of Quad Output Mode

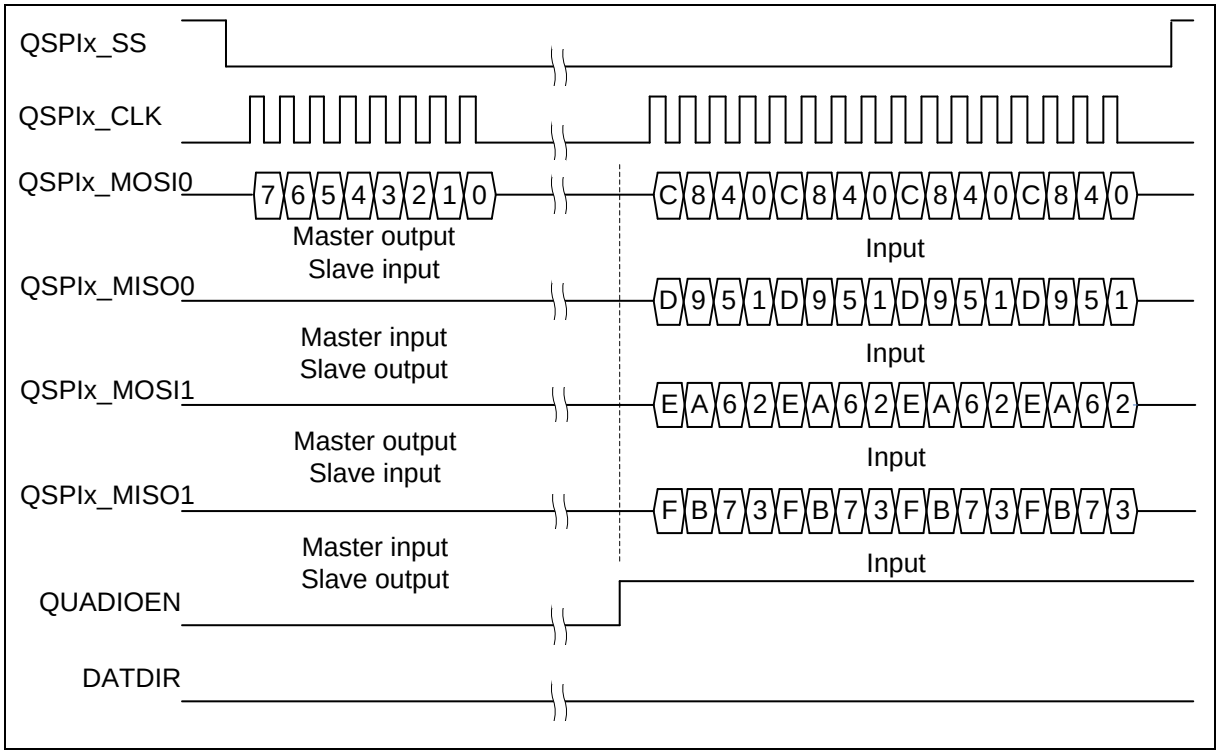


Figure 6.31-17 Bit Sequence of Quad Input Mode

6.31.5.11 Transmit Double Transfer Rate Mode

QSPI Master mode supports TX DTR mode (Transmit Double Transfer Rate Mode) when user sets control register TXDTREN (QSPIx_CTL[23]) to 1. The timing diagram of TX DTR mode is shown in Figure 6.31-18.

In TX DTR mode, the frequency of QSPI bus clock (QSPIx_CLK) equals to the half frequency of QSPI peripheral clock (QSPI_ECLK). TX output data of QSPI in TX DTR mode will be sent near rising edge and falling edge of QSPIx_CLK, where the rising edge and falling edge of QSPIx_CLK are the half period of QSPI peripheral clock (QSPI_ECLK) behind data driving edge of QSPI output data QSPIx_MOSI.

User can select idle polarity of QSPIx_CLK in TX DTR mode using the control register CLKPOL (QSPIx_CTL[3]). When user sets CLKPOL to 0, the idle polarity of QSPIx_CLK is 0. When user sets CLKPOL to 1, the idle polarity of QSPIx_CLK is 1. In TX DTR mode, TXNEG (QSPIx_CTL[2]) equals to CLKPOL (QSPIx_CTL[3]), and SUSPITV (QSPIx_CTL[7:4]) equals to 0. In TX DTR mode of SPI master, user does not use RX data that is received from SPI slave device.

Note: QSPI Master mode supports TXDTR mode, and QSPI Slave mode does not support this mode. There is extra half clock period between the slave select and serial clock output than the TXDTREN (QSPIx_CTL[23]) is disabled.

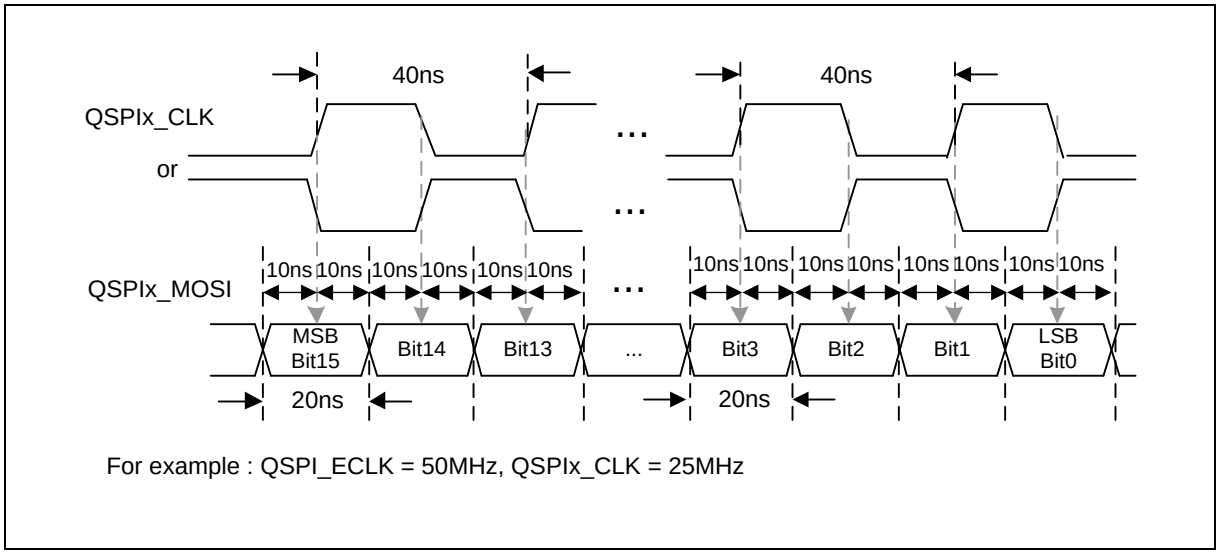


Figure 6.31-18 Timing Diagram of QSPI Output Data for Transmit Double Transfer Rate Mode

6.31.5.12 FIFO Buffer Operation

The QSPI controllers are equipped with eight 32-bit wide transmit and receive FIFO buffers. The data stored in the transmit FIFO buffer will be read and sent out by the transmission control logic. If the transmit FIFO buffer is full, the TXFULL (QSPIx_STATUS[17]) will be set to 1. When the QSPI transmission logic unit draws out the last datum of the transmit FIFO buffer, so that the transmit FIFO buffer is empty, the TXEMPTY (QSPIx_STATUS[16]) will be set to 1. Note that the TXEMPTY (QSPIx_STATUS[16]) flag is set to 1 while the last transaction is still in progress. In Master mode, the BUSY (QSPIx_STATUS[0]) is set to 1 when the FIFO buffer is written any data or there is any transaction on the QSPI bus. (e.g. the slave selection signal is active and the QSPI controller is receiving data in Slave mode). It will set to 0 when the transmit FIFO is empty and the current transaction has done. Thus, the status of BUSY (QSPIx_STATUS[0]) should be checked by software to make sure whether the QSPI is in idle or not.

The receive control logic will store the QSPI input data into the receive FIFO buffer. There are FIFO related status bits, like RXEMPTY (QSPIx_STATUS[8]) and RXFULL (QSPIx_STATUS[9]), to indicate the current status of RX FIFO buffer.

The transmitting and receiving threshold can be configured by setting TXTH (QSPIx_FIFOCTL[30:28]) and RXTH (QSPIx_FIFOCTL[26:24]). When the count of valid data stored in transmit FIFO buffer is less than or equal to TXTH (QSPIx_FIFOCTL[30:28]) setting, TXTHIF (QSPIx_STATUS[18]) will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RXTH (QSPIx_FIFOCTL[26:24]) setting, RXTHIF (QSPIx_STATUS[10]) will be set to 1.

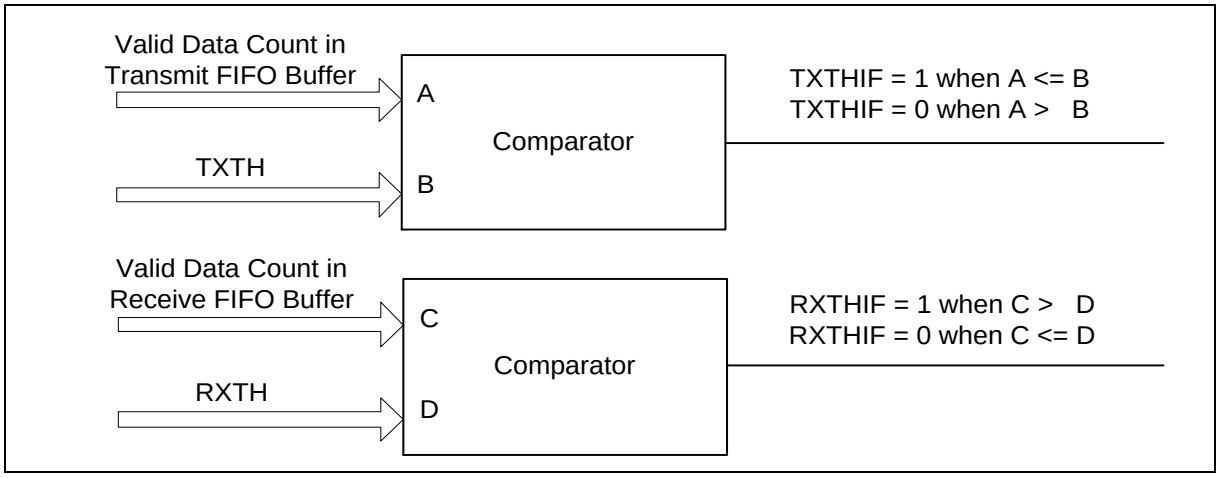


Figure 6.31-19 FIFO Threshold Comparator

In Master mode, when the first datum is written to the QSPIx_TX register, the TXEMPTY flag (QSPIx_STATUS[16]) will be cleared to 0. The transmission will start after 1 PCLK clock cycles and 6 peripheral clock cycles. User can write the next data into QSPIx_TX register immediately. The QSPI controller will insert a suspend interval between two successive transactions. The period of suspend interval is decided by the setting of SUSPITV (QSPIx_CTL[7:4]). If the SUSPITV (QSPIx_CTL[7:4]) equals 0, QSPI controller can perform continuous transfer. User can write data into QSPIx_TX register as long as the TXFULL (QSPIx_STATUS[17]) is 0.

In the example 1 of Figure 6.30-13, it indicates the updated condition of TXEMPTY (QSPIx_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer. The TXEMPTY (QSPIx_STATUS[16]) is set to 0 when the Data0 is written into the FIFO buffer. The Data0 will be loaded into the shift register by the core logic and the TXEMPTY (QSPIx_STATUS[16]) will be to 1. The Data0 in shift register will be shift into skew buffer by bit for transmission until the transfer is done.

In the Example 2, it indicates the updated condition of TXFULL (QSPIx_STATUS[17]) when there are 8 data in the FIFO buffer and the next data of Data9 does not be written into the FIFO buffer when the TXFULL = 1.

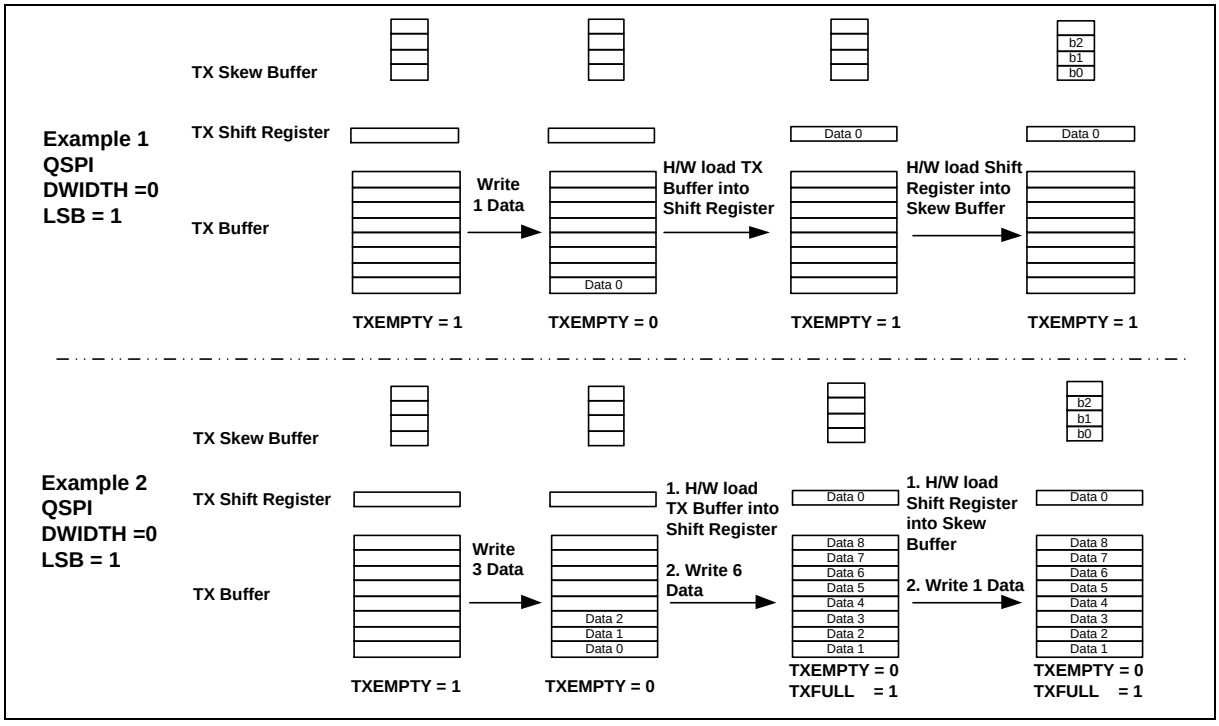


Figure 6.31-20 Transmit FIFO Buffer Example

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the QSPIx_TX register is not updated after all data transfer are done, the transfer will stop.

In Master mode, during receiving operation, the serial data are received from QSPIx_MISO pin and stored to receive FIFO buffer.

The received data (Data0's b0, b1, …b31) is stored into skew buffer first according the serial clock (QSPIx_CLK) and then it is shift into the shift register bit by bit. The core logic will load the data in shift register into FIFO buffer when the received data bit count reach the value of DWIDTH (QSPIx_CTL[12:8]). The RXEMPTY (QSPIx_STATUS[8]) will be cleared to 0 while the receive FIFO buffer contains unread data (see the Example 1 of Receive FIFO Buffer Example in Figure 6.31-21). The received data can be read by software from QSPIx_RX register as long as the RXEMPTY (QSPIx_STATUS[8]) is 0. If the receive FIFO buffer contains 8 unread data, the RXFULL (QSPIx_STATUS[9]) will be set to 1 (see the Example 2 of Receive FIFO Buffer Example).

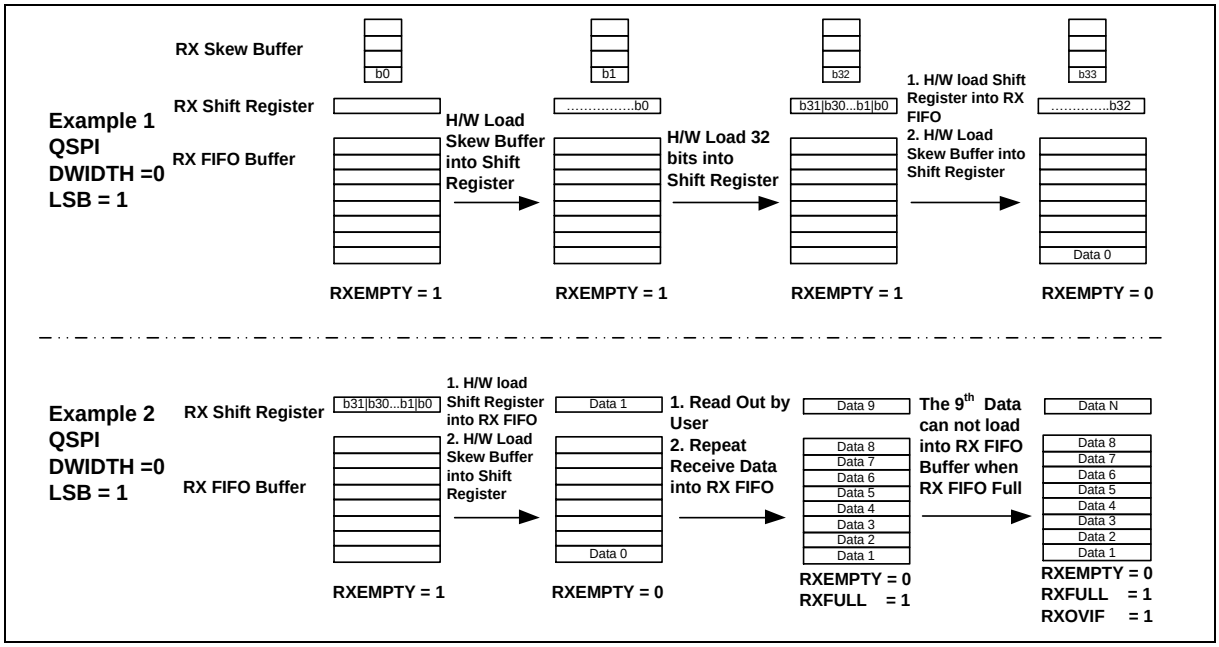


Figure 6.31-21 Receive FIFO Buffer Example

In Slave mode, during transmission operation, when data is written to the QSPIx_TX register by software, the data will be loaded into transmit FIFO buffer and the TXEMPTY (QSPIx_STATUS[16]) will be set to 0. The transmission will start when the slave device receives clock signal from master. Data can be written to QSPIx_TX register as long as the TXFULL (QSPIx_STATUS[17]) is 0. After all data have been drawn out by the QSPI transmission logic unit and the QSPIx_TX register is not updated by software, the TXEMPTY (QSPIx_STATUS[16]) will be set to 1.

If there is no any data written to the QSPIx_TX register, the transmit underflow interrupt flag, TXUFIF (QSPIx_STATUS[19]) will be set to 1 when the slave selection signal is active. The output data will be held by TXUFPOL (QSPIx_FIFOCTL[6]) setting during this transfer until the slave selection signal goes to inactive state. When the transmit underflow event occurs, the slave under run interrupt flag, SLVURIF (QSPIx_STATUS[7]), will be set to 1 as QSPIx_SS goes to inactive state.

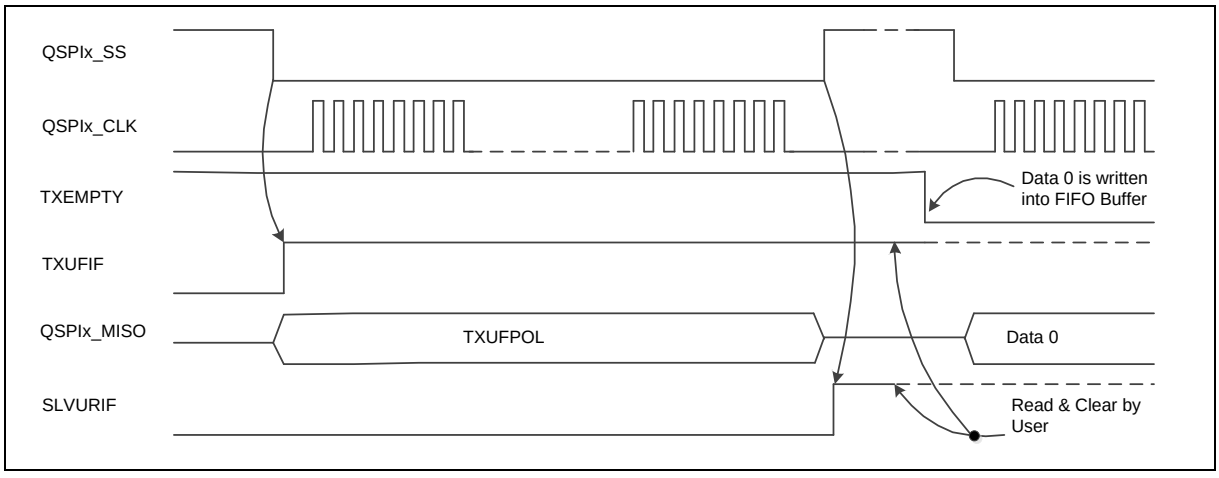


Figure 6.31-22 TX Underflow Event and Slave Under Run Event

In 2-bit Transfer mode, the transmit data is loaded into shift register after 2 datum have been written into the TX FIFO buffer. It uses two shift registers and two 4-level skew buffers concurrently. The detail timing of 2-bit Transfer mode, please refer to the section of Two-bit Transfer mode.

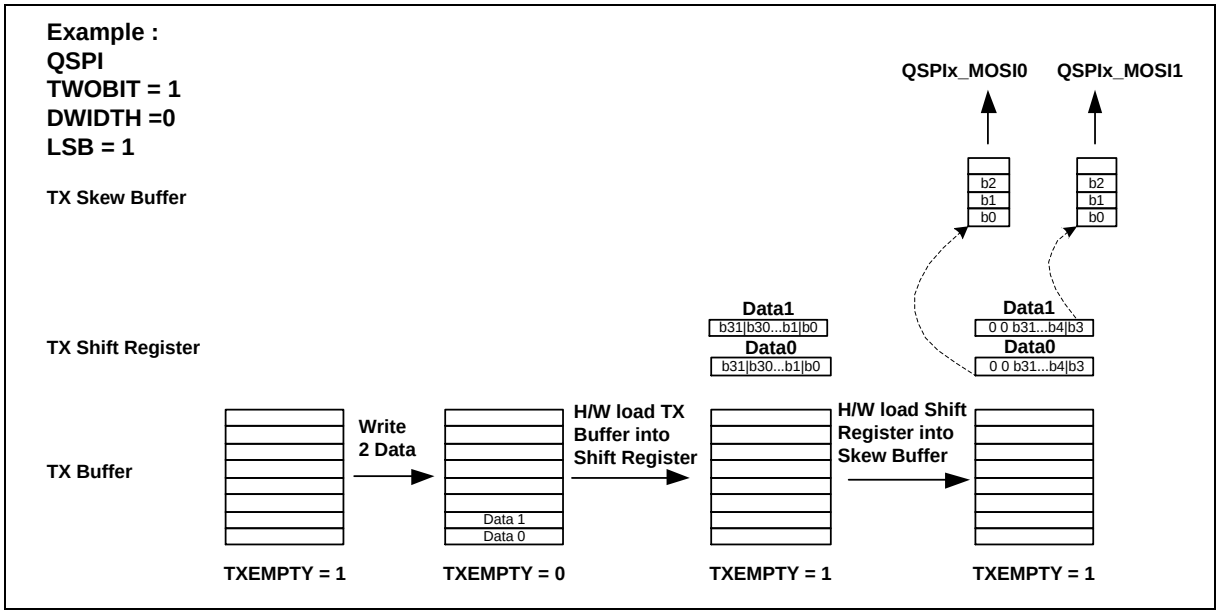


Figure 6.31-23 Two-bit Transfer Mode FIFO Buffer Example

In QSPI Slave 3-Wire mode, the first 2-bit data is un-predicted (keep on the level of last bit in previously transfer) if the data is written into TX FIFO among 3 peripheral clock cycles before the QSPI bus clock is presented. The other bits are held by TXUFPOL (QSPIx_FIFOCTL[6]) because there is TX underflow event. The written data will be transmitted in the next transfer.

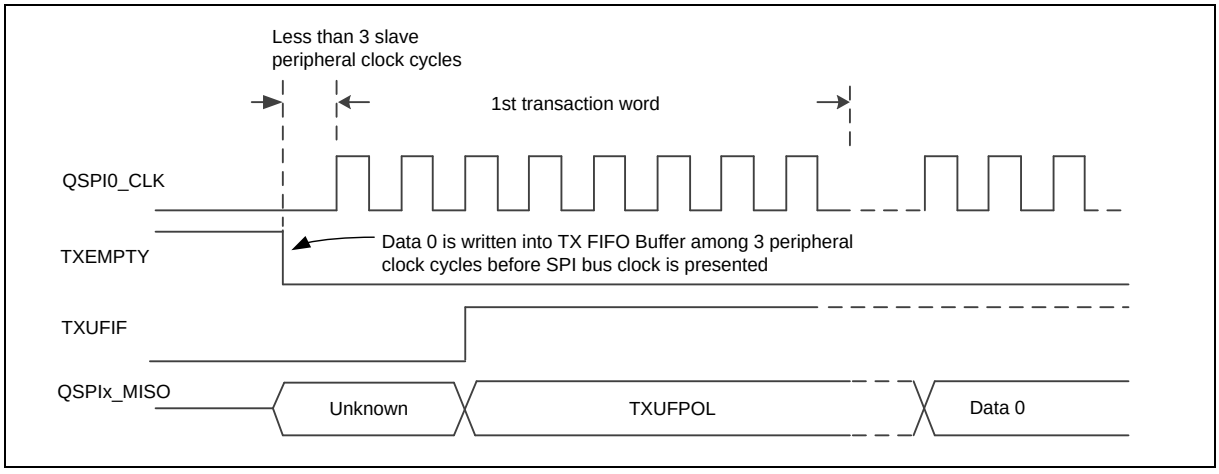


Figure 6.31-24 TX Underflow Event (QSPI Slave 3-Wire Mode Enabled)

In Slave mode, during receiving operation, the serial data is received from QSPIx_MOSI pin and stored to QSPIx_RX register. The reception mechanism is similar to Master mode reception operation. If the receive FIFO buffer contains 8 unread data, the RXFULL (QSPIx_STATUS[9]) will be set to 1 and the RXOVIF (QSPIx_STATUS[11]) will be set to 1 if there is more serial data received from QSPIx_MOSI and follow-up data will be dropped (refer to the Receive FIFO Buffer Example figure).

If the receive bit count mismatch with the DWIDTH (QSPIx_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (QSPIx_STATUS[6]) will be set to 1 in QSPI Slave mode. RX data will be written into RX FIFO and SLVBENUM (QSPIx_STATUS2[29:24]) will be updated when SLVBERX (QSPIx_FIFOCTL[10]) is enabled and QSPI slave bit count error event happened. RX data will be dropped and SLVBENUM (QSPIx_STATUS2[29:24]) will be fixed to 0x0 when the control register SLVBERX (QSPIx_FIFOCTL[10]) is disabled and QSPI slave bit count error event happened. The status register SLVBENUM (QSPIx_STATUS2[29:24]) indicates that effective bit number of uncompleted RX

data when QSPI slave bit count error happened. In Figure 6.31-25, the timing diagram of QSPI slave bit count error and SLVBENUM is shown, and SLVBENUM will be updated to 0x8 when QSPI slave device receives 8 bits data and DWIDTH (QSPIx_CTL[12:8]) is set to 16 bits at QSPI bit count error event. The QSPI slave bit count error event (SLVBEIF) will generates RX FIFO write operation pulse to write RX data from RX shift register to RX FIFO, and RX FIFO count (RXCNT) will update to one.

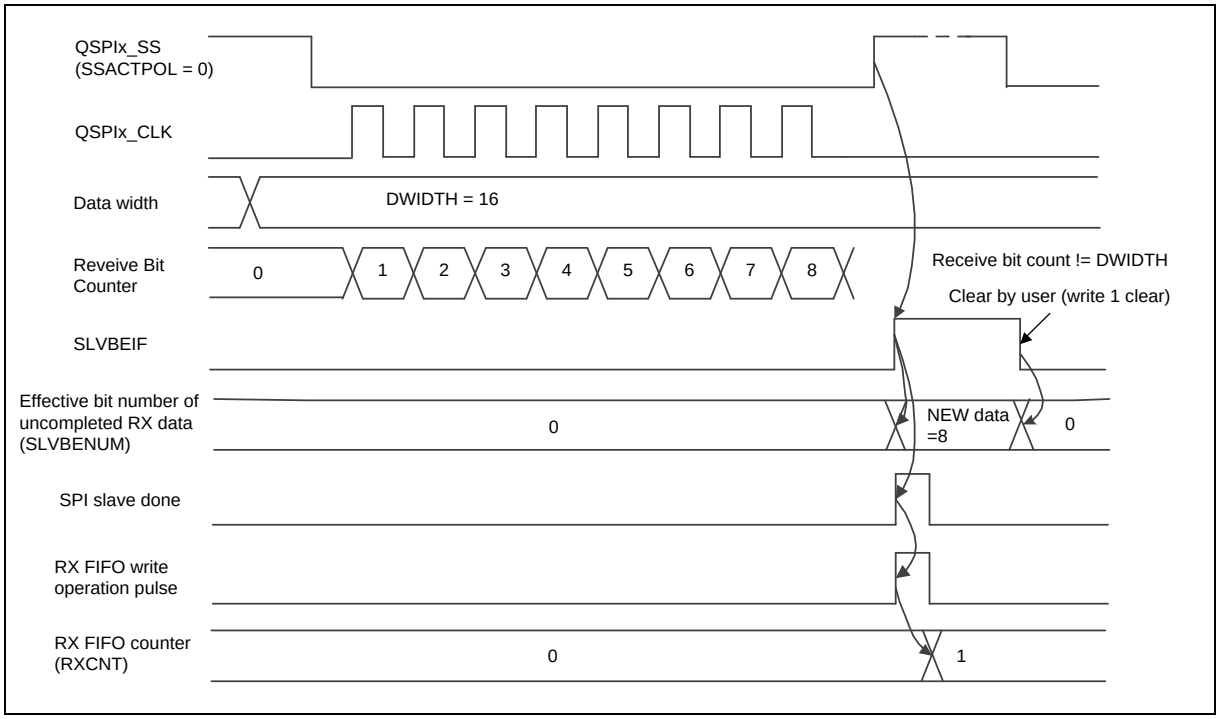


Figure 6.31-25 Slave Mode Bit Count Error and Effective Bit Number of Uncompleted RX Data

When the Slave select signal is active and the value of SLVTOCNT (QSPIx_SSCTL[31:16]) is not 0, the Slave time-out counter in the QSPI controller logic will start after the serial clock input. This counter will be cleared after one transaction done or the SLVTOCNT is set to 0. If the value of the time-out counter is equal to the value of SLVTOCNT before one transaction done, the slave time-out event occurs and the SLVTOIF (QSPIx_STATUS[5]) will be set to 1.

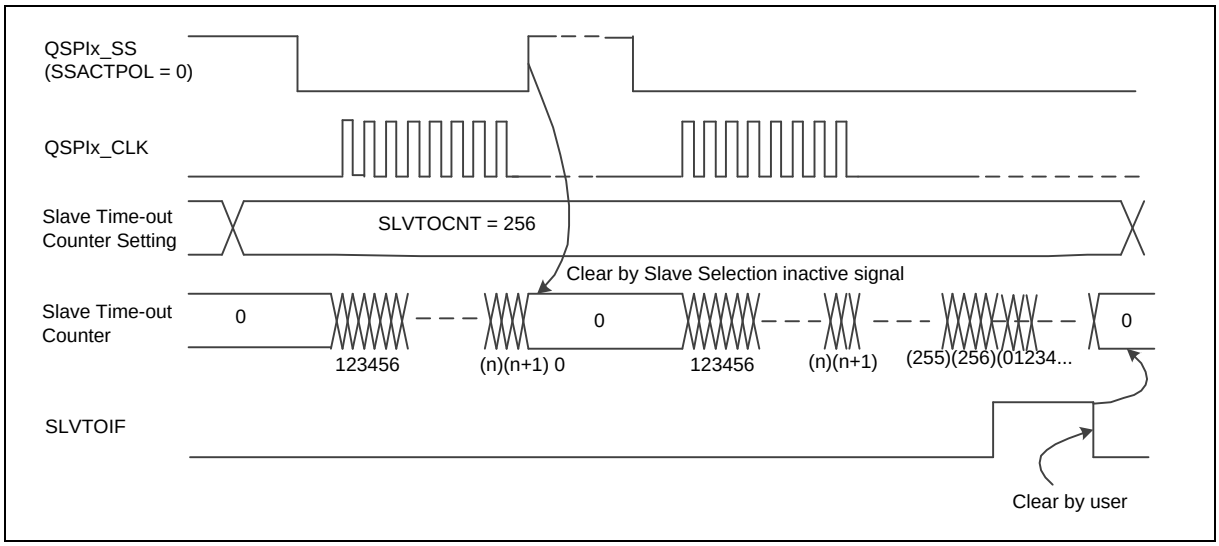


Figure 6.31-26 Slave Time-out Event

A receive time-out function is built-in in this controller. When the receive FIFO is not empty and no read operation in receive FIFO over 64 QSPI peripheral clock periods in Master mode or over 576 QSPI peripheral clock periods in Slave mode, the receive time-out occurs and the RXTOIF (QSPIx_STATUS[12]) will be set to 1. When the receive FIFO is read by user, the time-out status will be cleared automatically.

6.31.5.13 Interrupt

- QSPI unit transfer interrupt

  As the QSPI controller finishes a unit transfer, the unit transfer interrupt flag UNITIF (QSPIx_STATUS[1]) will be set to 1. The unit transfer interrupt event will generate an interrupt to CPU if the unit transfer interrupt enable bit UNITIEN (QSPIx_CTL[17]) is set. The unit transfer interrupt flag can be cleared only by writing 1 to it.

- QSPI slave selection active/inactive interrupt

  In Slave mode, the slave selection active/inactive interrupt flag, SSACTIF (QSPIx_STATUS[2]) and SSINAIF (QSPIx_STATUS[3]), will be set to 1 when the SPIEN (QSPIx_CTL[0]) and SLAVE (QSPIx_CTL[18]) are set to 1 and the slave selection signal goes to active/inactive state. The QSPI controller will issue an interrupt if the SSINAIEN (QSPIx_SSCTL[13]) or SSACTIEN (QSPIx_SSCTL[12]), is set to 1.

- Slave time-out interrupt

  In Slave mode, there is slave time-out function for user to know that there is serial clock input but one transaction is not finished over the period of SLVTOCNT (QSPIx_SSCTL[31:16]) basing on Slave peripheral clock.

  When the slave selection signal is active and the value of SLVTOCNT (QSPIx_SSCTL[31:16]) is not 0, the slave time-out counter in the QSPI controller logic will start after the serial clock input. This counter will be cleared after one transaction done or the SLVTOCNT (QSPIx_SSCTL[31:16]) is set to 0. If the value of the time-out counter is greater than or equal to the value of SLVTOCNT (QSPIx_SSCTL[31:16]) before one transaction done, the slave time-out event occurs and the SLVTOIF (QSPIx_STATUS[5]) will be set to 1. The QSPI controller will issue an interrupt if the SLVTOIEN (QSPIx_SSCTL[5]) is set to 1.

- Slave bit count error interrupt

  In Slave mode, if the transmit/receive bit count mismatch with the DWIDTH (QSPIx_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (QSPIx_STATUS[6]) will be set to 1. The uncompleted transaction will be dropped from TX shift register. When the control register SLVBERX (QSPIx_FIFOCTL[10]) is disabled and QSPI slave bit count error event happened, the uncompleted transaction data will be dropped from RX shift registers. When control register SLVBERX (QSPIx_FIFOCTL[10]) is enabled and QSPI slave bit count error event happened, the uncompleted transaction data will be written from RX shift registers into RX FIFO. The QSPI controller will issue an interrupt if the SLVBEIEN (QSPIx_SSCTL[8]) is set to 1.

  Note: If the slave selection signal is active but there is no any serial clock input, the SLVBEIF (QSPIx_STATUS[6]) will be set to 1 when the slave selection signal goes to inactive state.

- TX underflow interrupt

  In QSPI Slave mode, if there is no any data is written to the QSPIx_TX register, the TXUFIF (QSPIx_STATUS[19]) will be set to 1 when the slave selection signal is active. The QSPI controller will issue a TX underflow interrupt if the TXUFIEN (QSPIx_FIFOCTL[7]) is set to 1.

  Note: If underflow event occurs in QSPI Slave mode, there are two conditions which make QSPI Slave mode return to idle state and then goes for next transfer: (1) set TXRST to 1

(2) slave select signal is changed to inactive state while SLV3WIRE=0.

- Slave TX under run interrupt

  If the TX underflow event occurs, the SLVURIF (QSPIx_STATUS[7]) will be set to 1 when QSPIx_SS goes to inactive state. The QSPI controller will issue a TX under run interrupt if the SLVURIEN (QSPIx_SSCTL[9]) is set to 1.

  Note: In Slave 3-Wire mode, the slave selection signal is considered active all the time so that user shall poll the TXUFIF (QSPIx_STATUS[19]) to know if there is TX underflow event or not.

- Receive Overrun interrupt

  In Slave mode, if the receive FIFO buffer contains 8 unread data, the RXFULL (QSPIx_STATUS[9]) will be set to 1 and the RXOVIF (QSPIx_STATUS[11]) will be set to 1 if there is more serial data received from QSPI bus and follow-up data will be dropped. The QSPI controller will issue an interrupt if the RXOVIEN (QSPIx_FIFOCTL[5]) is set to 1.

- Receive FIFO time-out interrupt

  If there is a received data in the FIFO buffer and it is not read by software over 64 QSPI peripheral clock periods in Master mode or over 576 QSPI peripheral clock periods in Slave mode, it will send a RX time-out interrupt to the system if the RX time-out interrupt enable bit, RXTOIEN (QSPIx_FIFOCTL[4]), is set to 1.

- Transmit FIFO interrupt

  In FIFO mode, if the valid data count of the transmit FIFO buffer is less than or equal to the setting value of TXTH (QSPIx_FIFOCTL[30:28]), the transmit FIFO interrupt flag TXTHIF (QSPIx_STATUS[18]) will be set to 1. The QSPI controller will generate a transmit FIFO interrupt to the system if the transmit FIFO interrupt enable bit, TXTHIEN (QSPIx_FIFOCTL[3]), is set to 1.

- Receive FIFO interrupt

  In FIFO mode, if the valid data count of the receive FIFO buffer is larger than the setting value of RXTH (QSPIx_FIFOCTL[26:24]), the receive FIFO interrupt flag RXTHIF (QSPIx_STATUS[10]) will be set to 1. The QSPI controller will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, RXTHIEN (QSPIx_FIFOCTL[2]), is set to 1.

6.31.6 Timing Diagram

The active state of slave selection signal can be defined by setting the SSACTPOL (QSPIx_SSCTL[2]). The QSPI clock which is in idle state can be configured as high or low state by setting the CLKPOL (QSPIx_CTL[3]). It also provides the bit length of a transaction word in DWIDTH (QSPIx_CTL[12:8]), and transmitting/receiving data from MSB or LSB first in LSB (QSPIx_CTL[13]). User can also select which edge of QSPI clock to transmit/receive data in TXNEG/RXNEG (QSPIx_CTL[2:1]). Four QSPI timing diagrams for master/slave operations and the related settings are shown below.

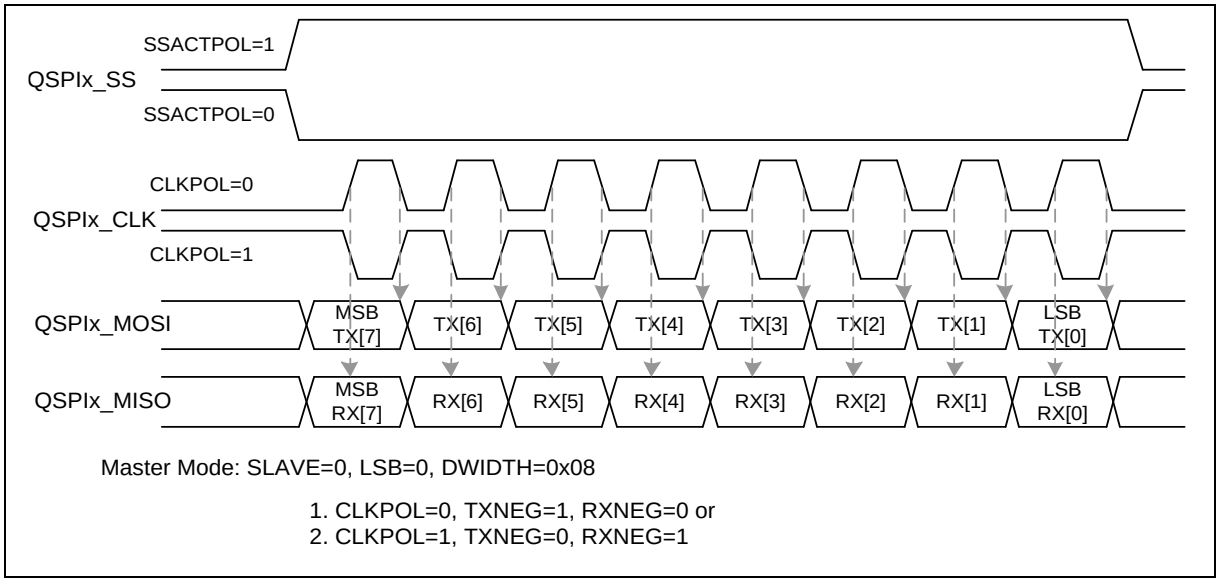


Figure 6.31-27 QSPI Timing in Master Mode

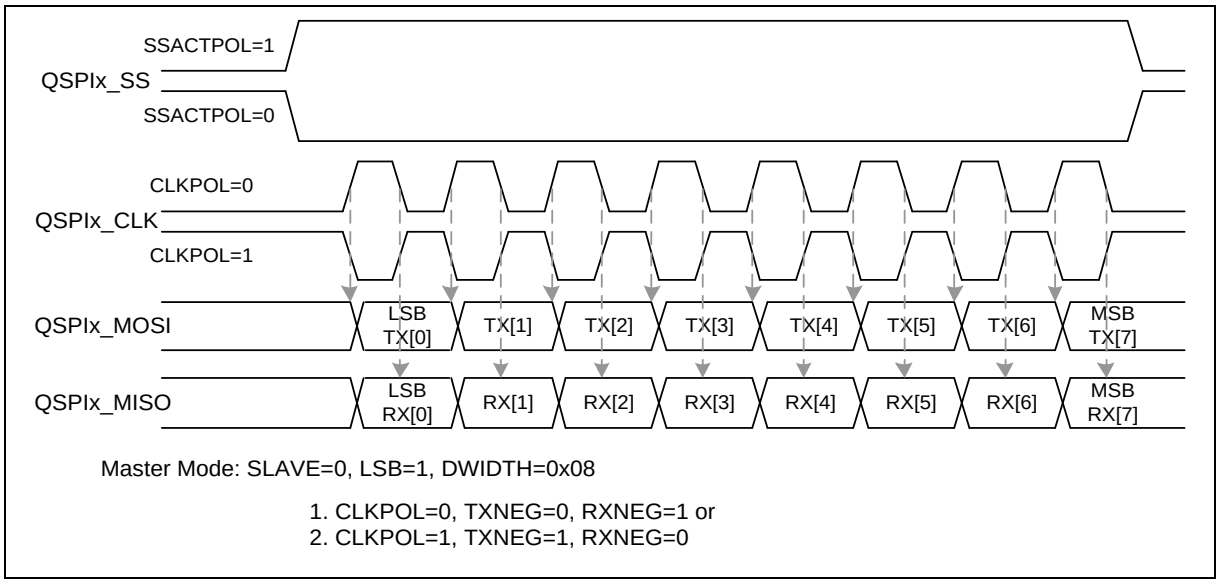


Figure 6.31-28 QSPI Timing in Master Mode (Alternate Phase of QSPIx_CLK)

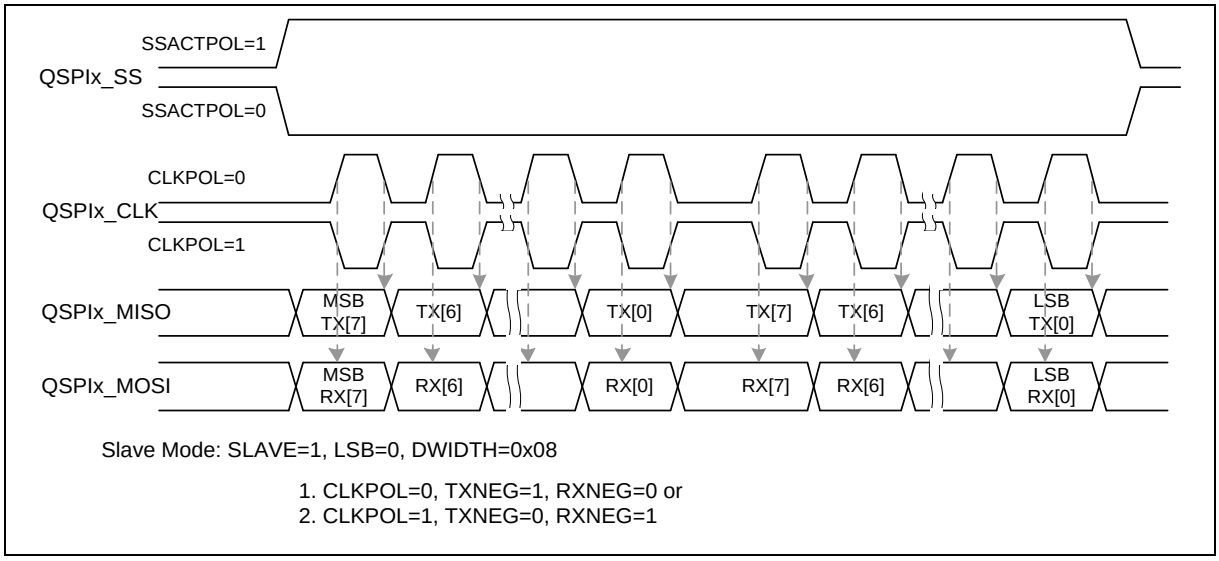


Figure 6.31-29 QSPI Timing in Slave Mode

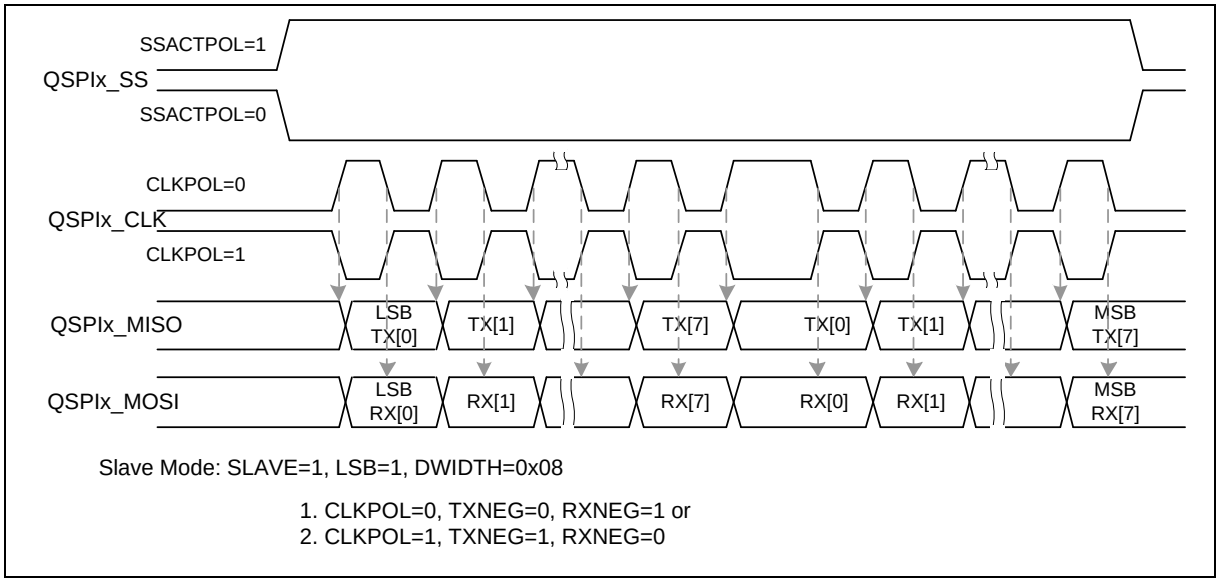


Figure 6.31-30 QSPI Timing in Slave Mode (Alternate Phase of QSPIx_CLK)

6.31.7 Programming Examples

Example 1

The QSPI controller is set as a full-duplex master to access an off-chip slave device with the following specifications:

- Data bit is latched on positive edge of QSPI bus clock.
- Data bit is driven on negative edge of QSPI bus clock.
- Data is transferred from MSB first.
- QSPI bus clock is idle at low state.
- Only one byte of data to be transmitted/received in a transaction.
- Uses the QSPI slave select pin to connect with an off-chip slave device. The slave

selection signal is active low.

The operation flow is as follows:

1. Set DIVIDER (QSPIx_CLKDIV[8:0]) to determine the output frequency of QSPI clock.
2. Write the QSPIx_SSCTL register a proper value for the related settings of Master mode:
   1) Clear AUTOSS (QSPIx_SSCTL[3]) to 0 to disable the Automatic Slave Selection function.
   2) Configure slave selection signal as active low by clearing SSACTPOL (QSPIx_SSCTL[2]) to 0.
   3) Enable slave selection signal by setting SS (QSPIx_SSCTL[0]) to 1 to activate the off-chip slave device.
3. Write the related settings into the QSPIx_CTL register to control the QSPI master actions.
   1) Configure this QSPI controller as master device by setting SLAVE (QSPIx_CTL[18]) to 0.
   2) Force the QSPI clock idle state at low by clearing CLKPOL (QSPIx_CTL[3]) to 0.
   3) Select data transmitted on negative edge of QSPI bus clock by setting TXNEG (QSPIx_CTL[2]) to 1.
   4) Select data latched on positive edge of QSPI bus clock by clearing RXNEG (QSPIx_CTL[1]) to 0.
   5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (QSPIx_CTL[12:8] = 0x08).
   6) Set MSB transfer first by clearing LSB (QSPIx_CTL[13]) to 0.
4. Set SPIEN (QSPIx_CTL[0]) to 1 to enable the data transfer with the QSPI interface.
5. If this QSPI master attempts to transmit (write) one byte data to the off-chip slave device, write the byte data that will be transmitted into the QSPIx_TX register.
6. Waiting for QSPI interrupt if the UNITIEN (QSPIx_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (QSPIx_STATUS[1]).
7. Read out the received one byte data from QSPIx_RX register.
8. Go to 5 to continue with another data transfer or set SS (QSPIx_SSCTL[0]) to 0 to inactivate the off-chip slave device.

Example 2

The QSPI controller is set as a full-duplex slave device and connects with an off-chip master device. The off-chip master device communicates with the on-chip QSPI slave controller through the QSPI interface with the following specifications:

- Data bit is latched on positive edge of QSPI bus clock.
- Data bit is driven on negative edge of QSPI bus clock.
- Data is transferred from LSB first.
- QSPI bus clock is idle at high state.
- Only one byte of data to be transmitted/received in a transaction.
- Slave selection signal is active high.

The operation flow is as follows:

1. Write the QSPIx_SSCTL register a proper value for the related settings of Slave mode.
2. Select high level for the input of slave selection signal by setting SSACTPOL (QSPIx_SSCTL[2]) to 1.
3. Write the related settings into the QSPIx_CTL register to control this QSPI slave actions

1) Set the QSPI controller as slave device by setting SLAVE (QSPIx_CTL[18]) to 1.
2) Select the QSPI clock idle state at high by setting CLKPOL (QSPIx_CTL[3]) to 1.
3) Select data transmitted on negative edge of QSPI bus clock by setting TXNEG (QSPIx_CTL[2]) to 1.
4) Select data latched on positive edge of QSPI bus clock by clearing RXNEG (QSPIx_CTL[1]) to 0.
5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (QSPIx_CTL[12:8] = 0x08).

4. Set LSB transfer first by setting LSB (QSPIx_CTL[13]) to 1.
5. Set the SPIEN (QSPIx_CTL[0]) to 1. Wait for the slave select trigger input and QSPI clock input from the off-chip master device to start the data transfer.
6. If this QSPI slave attempts to transmit (be read) one byte data to the off-chip master device, write the byte data that will be transmitted into the QSPIx_TX register.
7. If this QSPI slave just only attempts to receive (be written) one byte data from the off-chip master device and does not care what data will be transmitted, the QSPIx_TX register does not need to be updated by software.
8. Waiting for QSPI interrupt if the UNITIEN (QSPIx_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (QSPIx_STATUS[1]).
9. Read out the received one byte data from QSPIx_RX register.
10. Go to 6 to continue another data transfer or stop data transfer.

6.31.8 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
QSPI Base Address: **QSPIx_BA = 0x4006_0000 + (0x0000_9000 *x)** **x = 0**				
QSPIx_CTL	QSPIx_BA+0x00	R/W	QSPI Control Register	0x0000_0034
QSPIx_CLKDIV	QSPIx_BA+0x04	R/W	QSPI Clock Divider Register	0x0000_0000
QSPIx_SSCTL	QSPIx_BA+0x08	R/W	QSPI Slave Select Control Register	0x0000_0000
QSPIx_PDMACTL	QSPIx_BA+0x0C	R/W	QSPI PDMA Control Register	0x0000_0000
QSPIx_FIFOCTL	QSPIx_BA+0x10	R/W	QSPI FIFO Control Register	0x4400_0000
QSPIx_STATUS	QSPIx_BA+0x14	R/W	QSPI Status Register	0x0005_0110
QSPIx_STATUS2	QSPIx_BA+0x18	R	QSPI Status2 Register	0x0000_0000
QSPIx_TX	QSPIx_BA+0x20	W	QSPI Data Transmit Register	0x0000_0000
QSPIx_RX	QSPIx_BA+0x30	R	QSPI Data Receive Register	0x0000_0000

6.31.9 Register Description

QSPI Control Register (QSPIx_CTL)

Register	Offset	R/W	Description	Reset Value
QSPIx_CTL	QSPIx_BA+0x00	R/W	QSPI Control Register	0x0000_0034

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TXDTREN	**QUADIOEN**	**DUALIOEN**	**DATDIR**	**REORDER**	**SLAVE**	**UNITIEN**	**TWOBIT**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXONLY	**HALFDPX**	**LSB**	**DWIDTH**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SUSPITV				**CLKPOL**	**TXNEG**	**RXNEG**	**SPIEN**

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**TXDTREN**	**Transmit Double Transfer Rate Mode Enable Bit** 0 = TX DTR (Transmit Double Transfer Rate) mode Disabled. 1 = TX DTR (Transmit Double Transfer Rate) mode Enabled. **Note:** QSPI Master mode supports TXDTR mode, and QSPI Slave mode does not support this mode.
[22]	**QUADIOEN**	**Quad I/O Mode Enable Bit** 0 = Quad I/O mode Disabled. 1 = Quad I/O mode Enabled.
[21]	**DUALIOEN**	**Dual I/O Mode Enable Bit** 0 = Dual I/O mode Disabled. 1 = Dual I/O mode Enabled.
[20]	**DATDIR**	**Data Port Direction Control** This bit is used to select the data input/output direction in half-duplex transfer and Dual/Quad transfer 0 = QSPI data is input direction. 1 = QSPI data is output direction.
[19]	**REORDER**	**Byte Reorder Function Enable Bit** 0 = Byte Reorder function Disabled. 1 = Byte Reorder function Enabled. A byte suspend interval will be inserted among each byte. The period of the byte suspend interval depends on the setting of SUSPITV. **Note:** Byte Reorder function is only available if DWIDTH is defined as 16, 24, and 32 bits.
[18]	**SLAVE**	**Slave Mode Control** 0 = Master mode. 1 = Slave mode.

[17]	**UNITIEN**	**Unit Transfer Interrupt Enable Bit** 0 = QSPI unit transfer interrupt Disabled. 1 = QSPI unit transfer interrupt Enabled.
[16]	**TWOBIT**	**2-bit Transfer Mode Enable Bit** 0 = 2-bit Transfer mode Disabled. 1 = 2-bit Transfer mode Enabled. **Note:** When 2-bit Transfer mode is enabled, the first serial transmitted bit data is from the first FIFO buffer data, and the 2nd serial transmitted bit data is from the second FIFO buffer data. As the same as transmitted function, the first received bit data is stored into the first FIFO buffer and the 2nd received bit data is stored into the second FIFO buffer at the same time.
[15]	**RXONLY**	**Receive-only Mode Enable Bit** This bit field is only available in Master mode. In receive-only mode, QSPI Master will generate QSPI bus clock continuously for receiving data bit from SPI slave device and assert the BUSY status. 0 = Receive-only mode Disabled. 1 = Receive-only mode Enabled.
[14]	**HALFDPX**	**QSPI Half-duplex Transfer Enable Bit** This bit is used to select full-duplex or half-duplex for QSPI transfer. The bit field DATDIR (QSPIx_CTL[20]) can be used to set the data direction in half-duplex transfer. 0 = QSPI operates in full-duplex transfer. 1 = QSPI operates in half-duplex transfer.
[13]	**LSB**	**Send LSB First** 0 = The MSB, which bit of transmit/receive register depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, bit 0 of the QSPIx TX register, is sent first to the QSPI data output pin, and the first bit received from the QSPI data input pin will be put in the LSB position of the RX register (bit 0 of QSPIx_RX).
[12:8]	**DWIDTH**	**Data Width** This field specifies how many bits can be transmitted/received in one transaction. The minimum bit length is 8 bits and can up to 32 bits. DWIDTH = 0x08 …. 8 bits. DWIDTH = 0x09 …. 9 bits. …… DWIDTH = 0x1F …. 31 bits. DWIDTH = 0x00 …. 32 bits.
[7:4]	**SUSPITV**	**Suspend Interval** The four bits provide configurable suspend interval between two successive transmit/receive transaction in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. (SUSPITV[3:0] + 0.5) * period of QSPI_CLK clock cycle Example: SUSPITV = 0x0 …. 0.5 QSPI_CLK clock cycle. SUSPITV = 0x1 …. 1.5 QSPI_CLK clock cycle. …… SUSPITV = 0xE …. 14.5 QSPI_CLK clock cycle. SUSPITV = 0xF …. 15.5 QSPI_CLK clock cycle. **Note:** Master Only and In TX DTR mode, SUSPITV equals to 0x0.
[3]	**CLKPOL**	**Clock Polarity**

		0 = QSPI bus clock is idle low. 1 = QSPI bus clock is idle high.
[2]	**TXNEG**	**Transmit on Negative Edge** 0 = Transmitted data output signal is changed on the rising edge of QSPI bus clock. 1 = Transmitted data output signal is changed on the falling edge of QSPI bus clock. **Note:** In TX DTR mode, TXNEG equals to CLKPOL (QSPIx_CTL[3]).
[1]	**RXNEG**	**Receive on Negative Edge** 0 = Received data input signal is latched on the rising edge of QSPI bus clock. 1 = Received data input signal is latched on the falling edge of QSPI bus clock.
[0]	**SPIEN**	**QSPI Transfer Control Enable Bit** In Master mode, the transfer will start when there is data in the FIFO buffer after this bit is set to 1. In Slave mode, this device is ready to receive data when this bit is set to 1. 0 = Transfer control Disabled. 1 = Transfer control Enabled. **Note:** Before changing the configurations of QSPIx_CTL, QSPIx_CLKDIV, QSPIx_SSCTL and QSPIx_FIFOCTL registers, user shall clear the SPIEN (QSPIx_CTL[0]) and confirm the SPIENSTS (QSPIx_STATUS[15]) is 0.

QSPI Clock Divider Register (QSPIx_CLKDIV)

Register	Offset	R/W	Description	Reset Value
QSPIx_CLKDIV	QSPIx_BA+0x04	R/W	QSPI Clock Divider Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							DIVIDER
7	6	5	4	3	2	1	0
DIVIDER							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**DIVIDER**	**Clock Divider** The value in this field is the frequency divider for generating the peripheral clock, f_{spi_eclk}, and the QSPI bus clock of QSPI Master. The frequency is obtained according to the following equation. $f_{spi_eclk} = \frac{f_{spi_clock_src}}{(DIVIDER + 1)}$ where $f_{spi_clock_src}$ is the peripheral clock source, which is defined in the clock control register, CLK_CLKSEL2. **Note:** The time interval must be larger than or equal to 8 peripheral clock cycles between releasing QSPI IP software reset and setting this clock divider register.

Note: DIVIDER should be set carefully because the peripheral clock frequency must be slower than or equal to system frequency.

QSPI Slave Select Control Register (QSPIx_SSCTL)

Register	Offset	R/W	Description	Reset Value
QSPIx_SSCTL	QSPIx_BA+0x08	R/W	QSPI Slave Select Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SLVTOCNT							
23	22	21	20	19	18	17	16
SLVTOCNT							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVURIEN	SLVBEIEN
7	6	5	4	3	2	1	0
Reserved	SLVTORST	SLVTOIEN	SLV3WIRE	AUTOSS	SSACTPOL	Reserved	SS

Bits	Description	
[31:16]	**SLVTOCNT**	**Slave Mode Time-out Period** In Slave mode, these bits indicate the time-out period when there is bus clock input during slave select active. The clock source of the time-out counter is Slave peripheral clock. If the value is 0, it indicates the slave mode time-out function is disabled.
[15:14]	**Reserved**	Reserved.
[13]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable Bit** 0 = Slave select inactive interrupt Disabled. 1 = Slave select inactive interrupt Enabled.
[12]	**SSACTIEN**	**Slave Select Active Interrupt Enable Bit** 0 = Slave select active interrupt Disabled. 1 = Slave select active interrupt Enabled.
[11:10]	**Reserved**	Reserved.
[9]	**SLVURIEN**	**Slave Mode TX Under Run Interrupt Enable Bit** 0 = Slave mode TX under run interrupt Disabled. 1 = Slave mode TX under run interrupt Enabled.
[8]	**SLVBEIEN**	**Slave Mode Bit Count Error Interrupt Enable Bit** 0 = Slave mode bit count error interrupt Disabled. 1 = Slave mode bit count error interrupt Enabled.
[7]	**Reserved**	Reserved.
[6]	**SLVTORST**	**Slave Mode Time-out Reset Control** 0 = When Slave mode time-out event occurs, the TX and RX control circuit will not be reset. 1 = When Slave mode time-out event occurs, the TX and RX control circuit will be reset by hardware.
[5]	**SLVTOIEN**	**Slave Mode Time-out Interrupt Enable Bit** 0 = Slave mode time-out interrupt Disabled. 1 = Slave mode time-out interrupt Enabled.

[4]	**SLV3WIRE**	**Slave 3-wire Mode Enable Bit** In Slave 3-wire mode, the QSPI controller can work with 3-wire interface including QSPIx_CLK, QSPIx_MISO and QSPIx_MOSI pins. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface.
[3]	**AUTOSS**	**Automatic Slave Selection Function Enable Bit** 0 = Automatic slave selection function Disabled. Slave selection signal will be asserted/de-asserted according to SS (QSPIx_SSCTL[0]). 1 = Automatic slave selection function Enabled (Master only).
[2]	**SSACTPOL**	**Slave Selection Active Polarity** This bit defines the active polarity of slave selection signal (QSPIx_SS). 0 = The slave selection signal QSPIx_SS is active low. 1 = The slave selection signal QSPIx_SS is active high.
[1]	**Reserved**	Reserved.
[0]	**SS**	**Slave Selection Control** If AUTOSS bit is cleared to 0, 0 = Set the QSPIx_SS line to inactive state. 1 = Set the QSPIx_SS line to active state. If the AUTOSS bit is set to 1, 0 = Keep the QSPIx_SS line at inactive state. 1 = QSPIx_SS line will be automatically driven to active state for the duration of data transfer, and will be driven to inactive state for the rest of the time. The active state of QSPIx_SS is specified in SSACTPOL (QSPIx_SSCTL[2]). **Note:** Master Only.

QSPI PDMA Control Register (QSPIx_PDMACTL)

Register	Offset	R/W	Description	Reset Value
QSPIx_PDMACTL	QSPIx_BA+0x0C	R/W	QSPI PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					PDMARST	RXPDMAEN	TXPDMAEN

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the PDMA control logic of the QSPI controller. This bit will be automatically cleared to 0.
[1]	**RXPDMAEN**	**Receive PDMA Enable Bit** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[0]	**TXPDMAEN**	**Transmit PDMA Enable Bit** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled. **Note 1:** In QSPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, RX PDMA function cannot be enabled prior to TX PDMA function. User can enable TX PDMA function firstly or enable both functions simultaneously. **Note 2:** In QSPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, TX PDMA function cannot be disabled prior to RX PDMA function. User can disable RX PDMA function firstly or disable both functions simultaneously.

QSPI FIFO Control Register (QSPIx_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
QSPIx_FIFOCTL	QSPIx_BA+0x10	R/W	QSPI FIFO Control Register	0x4400_0000

31	30	29	28	27	26	25	24
Reserved	TXTH			Reserved	RXTH		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					SLVBERX	TXFBCLR	RXFBCLR
7	6	5	4	3	2	1	0
TXUFIEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TXTH**	**Transmit FIFO Threshold** If the valid data count of the transmit FIFO buffer is less than or equal to the TXTH setting, the TXTHIF bit will be set to 1, else the TXTHIF bit will be cleared to 0.
[27]	**Reserved**	Reserved.
[26:24]	**RXTH**	**Receive FIFO Threshold** If the valid data count of the receive FIFO buffer is larger than the RXTH setting, the RXTHIF bit will be set to 1, else the RXTHIF bit will be cleared to 0.
[23:11]	**Reserved**	Reserved.
[10]	**SLVBERX**	**RX FIFO Write Data Enable Bit When Slave Mode Bit Count Error** 0 = Uncompleted RX data will be dropped from RX FIFO when bit count error event happen in QSPI slave mode. 1 = Uncompleted RX data will be written into RX FIFO when bit count error event happen in QSPI slave mode. User can read SLVBENUM (QSPIx_STATUS2[29:24]) to know that the effective bit number of uncompleted RX data when SPI slave bit count error happened. **Note:** Slave mode Only.
[9]	**TXFBCLR**	**Transmit FIFO Buffer Clear** 0 = No effect. 1 = Clear transmit FIFO pointer. The TXFULL bit will be cleared to 0 and the TXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 1 system clock after it is set to 1. **Note:** The TX shift register will not be cleared.
[8]	**RXFBCLR**	**Receive FIFO Buffer Clear** 0 = No effect. 1 = Clear receive FIFO pointer. The RXFULL bit will be cleared to 0 and the RXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 1 system clock after it is set to 1. **Note:** The RX shift register will not be cleared.

[7]	**TXUFIEN**	**TX Underflow Interrupt Enable Bit** When TX underflow event occurs in Slave mode, TXUFIF (QSPIx_STATUS[19]) will be set to 1. This bit is used to enable the TX underflow interrupt. 0 = Slave TX underflow interrupt Disabled. 1 = Slave TX underflow interrupt Enabled.
[6]	**TXUFPOL**	**TX Underflow Data Polarity** 0 = The QSPI data out is kept 0 if there is TX underflow event in Slave mode. 1 = The QSPI data out is kept 1 if there is TX underflow event in Slave mode. **Note 1:** The TX underflow event occurs if there is no any data in TX FIFO when the slave selection signal is active. **Note 2:** When TX underflow event occurs, QSPIx_MISO pin state will be determined by this setting even though TX FIFO is not empty afterward. Data stored in TX FIFO will be sent through QSPIx_MISO pin in the next transfer frame.
[5]	**RXOVIEN**	**Receive FIFO Overrun Interrupt Enable Bit** 0 = Receive FIFO overrun interrupt Disabled. 1 = Receive FIFO overrun interrupt Enabled.
[4]	**RXTOIEN**	**Receive Time-out Interrupt Enable Bit** 0 = Receive time-out interrupt Disabled. 1 = Receive time-out interrupt Enabled.
[3]	**TXTHIEN**	**Transmit FIFO Threshold Interrupt Enable Bit** 0 = TX FIFO threshold interrupt Disabled. 1 = TX FIFO threshold interrupt Enabled.
[2]	**RXTHIEN**	**Receive FIFO Threshold Interrupt Enable Bit** 0 = RX FIFO threshold interrupt Disabled. 1 = RX FIFO threshold interrupt Enabled.
[1]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset transmit FIFO pointer and transmit circuit. The TXFULL bit will be cleared to 0 and the TXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (QSPIx_STATUS[23]) to check if reset is accomplished or not. **Note:** If TX underflow event occurs in QSPI Slave mode, this bit can be used to make QSPI return to idle state.
[0]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset receive FIFO pointer and receive circuit. The RXFULL bit will be cleared to 0 and the RXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (QSPIx_STATUS[23]) to check if reset is accomplished or not.

QSPI Status Register (QSPIx_STATUS)

Register	Offset	R/W	Description	Reset Value
QSPIx_STATUS	QSPIx_BA+0x14	R/W	QSPI Status Register	0x0005_0110

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	SLVTOIF	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

Bits	Description	
[31:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**TX or RX Reset Status (Read Only)** 0 = The reset function of TXRST or RXRST is done. 1 = Doing the reset function of TXRST or RXRST. **Note:** Both the reset operations of TXRST and RXRST need 3 system clock cycles + 2 peripheral clock cycles. User can check the status of this bit to monitor the reset function is doing or done.
[22:20]	**Reserved**	Reserved.
[19]	**TXUFIF**	**TX Underflow Interrupt Flag** When the TX underflow event occurs, this bit will be set to 1, the state of data output pin depends on the setting of TXUFPOL. 0 = No effect. 1 = No data in Transmit FIFO and TX shift register when the slave selection signal is active. **Note 1:** This bit will be cleared by writing 1 to it. **Note 2:** If reset slave's transmission circuit when slave selection signal is active, this flag will be set to 1 after 2 peripheral clock cycles + 3 system clock cycles since the reset operation is done.
[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TXTH. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TXTH.
[17]	**TXFULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.

[16]	**TXEMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[15]	**SPIENSTS**	**QSPI Enable Status (Read Only)** 0 = QSPI controller Disabled. 1 = QSPI controller Enabled. **Note:** The QSPI peripheral clock is asynchronous with the system clock. In order to make sure the QSPI control logic is disabled, this bit indicates the real status of QSPI controller.
[14:13]	**Reserved**	Reserved.
[12]	**RXTOIF**	**Receive Time-out Interrupt Flag** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 QSPI peripheral clock periods in Master mode or over 576 QSPI peripheral clock periods in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. **Note:** This bit will be cleared by writing 1 to it.
[11]	**RXOVIF**	**Receive FIFO Overrun Interrupt Flag** When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1. 0 = No FIFO is overrun. 1 = Receive FIFO is overrun. **Note 1:** This bit will be cleared by writing 1 to it. **Note 2**: If the RXONLY is enabled, the SPI bus length for one transaction shall greater than 5 ECLK + 2 PCLK, otherwise this flag will be set. (ex. 8-bit length quad mode, there is 2 bit bus cycle only and it is less than the limitation above).
[10]	**RXTHIF**	**Receive FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the receive FIFO buffer is smaller than or equal to the setting value of RXTH. 1 = The valid data count within the receive FIFO buffer is larger than the setting value of RXTH.
[9]	**RXFULL**	**Receive FIFO Buffer Full Indicator (Read Only)** 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[8]	**RXEMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[7]	**SLVURIF**	**Slave Mode TX Under Run Interrupt Flag** In Slave mode, if TX underflow event occurs and the slave select line goes to inactive state, this interrupt flag will be set to 1. 0 = No Slave TX under run event. 1 = Slave TX under run event occurred. **Note:** This bit will be cleared by writing 1 to it.
[6]	**SLVBEIF**	**Slave Mode Bit Count Error Interrupt Flag** In Slave mode, when the slave select line goes to inactive state, if bit counter is mismatch with DWIDTH, this interrupt flag will be set to 1. 0 = No Slave mode bit count error event. 1 = Slave mode bit count error event occurred. **Note:** If the slave select active but there is no any bus clock input, the SLVBEIF also active when the slave select goes to inactive state. This bit will be cleared by writing 1 to it.

[5]	**SLVTOIF**	**Slave Time-out Interrupt Flag** When the slave select is active and the value of SLVTOCNT is not 0, if the bus clock is detected, the slave time-out counter in QSPI controller logic will be started. When the value of time-out counter is greater than or equal to the value of SLVTOCNT (QSPIx_SSCTL[31:16]) before one transaction is done, the slave time-out interrupt event will be asserted. 0 = Slave time-out is not active. 1 = Slave time-out is active. **Note:** This bit will be cleared by writing 1 to it.
[4]	**SSLINE**	**Slave Select Line Bus Status (Read Only)** 0 = The slave select line status is 0. 1 = The slave select line status is 1. **Note:** This bit is only available in Slave mode. If SSACTPOL (QSPIx_SSCTL[2]) is set 0, and the SSLINE is 1, the QSPI slave select is in inactive status.
[3]	**SSINAIF**	**Slave Select Inactive Interrupt Flag** 0 = Slave select inactive interrupt was cleared or not occurred. 1 = Slave select inactive interrupt event occurred. **Note:** Only available in Slave mode. This bit will be cleared by writing 1 to it.
[2]	**SSACTIF**	**Slave Select Active Interrupt Flag** 0 = Slave select active interrupt was cleared or not occurred. 1 = Slave select active interrupt event occurred. **Note:** Only available in Slave mode. This bit will be cleared by writing 1 to it.
[1]	**UNITIF**	**Unit Transfer Interrupt Flag** 0 = No transaction has been finished since this bit was cleared to 0. 1 = QSPI controller has finished one unit transfer. **Note:** This bit will be cleared by writing 1 to it.
[0]	**BUSY**	**Busy Status (Read Only)** 0 = QSPI controller is in idle state. 1 = QSPI controller is in busy state. The following lists the bus busy conditions: ● SPIEN (QSPIx_CTL[0]) = 1 and TXEMPTY = 0. ● For QSPI Master mode, SPIEN (QSPIx_CTL[0]) = 1 and TXEMPTY = 1 but the current transaction is not finished yet. ● For QSPI Master mode, SPIEN (QSPIx_CTL[0]) = 1 and RXONLY = 1. ● For QSPI Slave mode, SPIEN (QSPIx_CTL[0]) = 1 and there is serial clock input into the QSPI core logic when slave select is active. ● For QSPI Slave mode, SPIEN (QSPIx_CTL[0]) = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive. **Note:** By applications, this QSPI busy flag should be used with other status registers in QSPIx_STATUS such as TXCNT, RXCNT, TXTHIF, TXFULL, TXEMPTY, RXTHIF, RXFULL, RXEMPTY, and UNITIF. Therefore the QSPI transfer done events of TX/RX operations can be obtained at correct timing point.

QSPI Status2 Register (QSPIx_STATUS2)

Register	Offset	R/W	Description	Reset Value
QSPIx_STATUS2	QSPIx_BA+0x18	R	QSPI Status2 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		**SLVBENUM**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**SLVBENUM**	**Effective Bit Number of Uncompleted RX data** This status register indicates that effective bit number of uncompleted RX data when SLVBERX (QSPIx_FIFOCTL[10]) is enabled and RX bit count error event happened in QSPI slave mode. This status register will be fixed to 0x0 when SLVBERX (QSPIx_FIFOCTL[10]) is disabled. **Note:** Slave mode Only and this register will be cleared to 0x0 when user writes 0x1 to SLVBEIF (QSPIx_STATUS[6]).
[23:0]	**Reserved**	Reserved.

QSPI Data Transmit Register (QSPIx_TX)

Register	Offset	R/W	Description	Reset Value
QSPIx_TX	QSPIx_BA+0x20	W	QSPI Data Transmit Register	0x0000_0000

31	30	29	28	27	26	25	24
TX							
23	22	21	20	19	18	17	16
TX							
15	14	13	12	11	10	9	8
TX							
7	6	5	4	3	2	1	0
TX							

Bits	Description	
[31:0]	**TX**	**Data Transmit Register** The data transmit registers pass through the transmitted data into the 8-level transmit FIFO buffers. The number of valid bits depends on the setting of DWIDTH (QSPIx_CTL[12:8]) in QSPI mode. In QSPI mode, if DWIDTH is set to 0x08, the bits TX[7:0] will be transmitted. If DWIDTH is set to 0x00, the QSPI controller will perform a 32-bit transfer. **Note:** In Master mode, QSPI controller will start to transfer the QSPI bus clock after 1 APB clock and 6 peripheral clock cycles after user writes to this register.

QSPI Data Receive Register (QSPIx_RX)

Register	Offset	R/W	Description	Reset Value
QSPIx_RX	QSPIx_BA+0x30	R	QSPI Data Receive Register	0x0000_0000

31	30	29	28	27	26	25	24
RX							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RX							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RX							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX							

Bits	Description	
[31:0]	**RX**	**Data Receive Register (Read Only)** There are 8-level FIFO buffers in this controller. The data receive register holds the data received from QSPI data input pin. If the RXEMPTY (QSPIx_STATUS[8) is not set to 1, the receive FIFO buffers can be accessed through software by reading this register.

6.32 USB 2.0 Full-Speed Device Controller (USBD)

6.32.1 Overview

There is one set of USB 2.0 full-speed device controller and transceiver in this device. It is compliant with USB 2.0 full-speed device specification and supports Control/Bulk/Interrupt/Isochronous transfer types.

In this device controller, there are two main interfaces: the APB bus and USB bus which comes from the USB PHY transceiver. For the APB bus, the CPU can program control registers through it. There are 1 Kbyte sizes internal SRAM as data buffer in this controller. For IN or OUT transfer, it is necessary to write data to SRAM or read data from SRAM through the APB interface or SIE. User needs to set the effective starting address of SRAM for each endpoint buffer through buffer segmentation register (USBD_BUFSEG0~18).

There are 19 endpoints in this controller. Each of the endpoint can be configured as IN or OUT endpoint. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. The block of “Endpoint Control” is also used to manage the data sequential synchronization, endpoint states, current start address, transaction status, and data buffer status for each endpoint.

There are four different interrupt events in this controller. They are the no-event-wake-up, device plug-in or plug-out event, USB events, like IN ACK, OUT ACK and BUS events, like suspend and resume. Any event will cause an interrupt, and users just need to check the related event flags in interrupt event status register (USBD_INTSTS) to acknowledge what kind of interrupt occurring, and then check the related USB Endpoint Status Register (USBD_EPSTS0~3) to acknowledge what kind of event occurring in this endpoint.

A software-disconnect function is also supported for this USB controller. It is used to simulate the disconnection of this device from the host. If user enables SE0 bit (USBD_SE0), the USB controller will force the output of USB_D+ and USB_D- to level low and its function is disabled. After disabling the SE0 bit, host will enumerate the USB device again.

For more information on the Universal Serial Bus, please refer to *Universal Serial Bus Specification Revision 1.1.*

6.32.2 Features

- Compliant with USB 2.0 Full-Speed specification
- Provides 1 interrupt vector with 5 different interrupt events (NEVWK, VBDET, USB, BUS and SOF)
- Supports Control/Bulk/Interrupt/Isochronous transfer type
- Supports suspend function when no bus activity existing for 3ms
- Supports 19 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 1 Kbyte buffer size
- Provides remote wake-up capability
- Supports double buffer function

6.32.3 Block Diagram

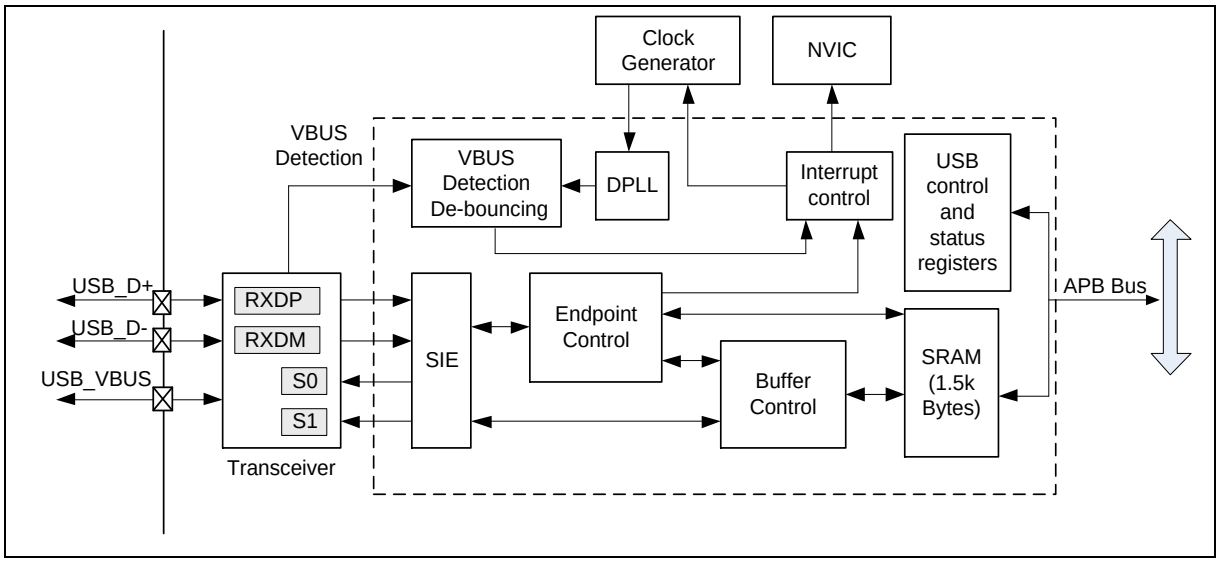


Figure 6.32-1 USB Block Diagram

6.32.4 Basic Configuration

The role of USB frame is determined by USBROLE (SYS_USBPHY[1:0]). The internal USB 3.3V LDO can be enabled by USBEN (SYS_USBPHY[8]) and wait 10 us for stable. These two configurations are write-protection bits. Before writing to these bits, user must disable the register protection function. Refer to the description of SYS_REGLCTL register for details. The USBD clock source is derived from HIRC48M or PLL. If user would like to achieve crystal-less, HIRC must be selected as USB clock source, or user has to use the PLL related configurations with crystal before USB device controller is enabled. Set the USBDCKEN (CLK_APBCLK0[27]) bit to enable USBD clock and 4-bit pre-scaler USBDIV (CLK_CLKDIV0[7:4]) to generate the proper USBD clock rate.

Note: USBD peripheral clock frequency needs to be greater than 15 MHz when USBD receives/transmits data from/to the host.

6.32.4.1 USB Device 1.1 Basic Configuration

- Clock source Configuration
  - Enable HIRC48MEN (CLK_PWRCTL[18]) or set PLL controller (CLK_PLLCTL)
  - Select the clock divider number of USBD peripheral clock on USBDIV(CLK_CLKDIV0[7:4])
  - Enable USBD peripheral clock in USBDCKEN (CLK_APBCLK0[27]).
- Reset Configuration
  - Reset USBD controller in USBDRST (SYS_IPRST1[27]).

6.32.5 Functional Description

6.32.5.1 Serial Interface Engine (SIE)

The SIE is the front-end of the device controller and handles most of the USB packet protocol. The SIE typically comprehends signaling up to the transaction level. The functions that it handles could include:

- Packet recognition and transaction sequencing
- SOP, EOP, RESET, RESUME signal detection/generation
- Clock/Data separation

- NRZI Data encoding/decoding and bit-stuffing
- CRC generation and checking (for Token and Data)
- Packet ID (PID) generation and checking/decoding
- Serial-Parallel/Parallel-Serial conversion

6.32.5.2 Endpoint Control

This controller supports 19 endpoints. Each of the endpoint can be configured as Control, Bulk, Interrupt, or Isochronous transfer type. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. It is also used to manage the data sequential synchronization, endpoint state control, current endpoint start address, current transaction status, and data buffer status in each endpoint.

6.32.5.3 Digital Phase Lock Loop (DPLL)

The bit rate of USB data is 12MHz. The DPLL uses the 48 MHz which comes from the clock controller to lock the input data RXDP and RXDM. The 12 MHz bit rate clock is also converted from DPLL.

6.32.5.4 V_{BUS} Detection De-bouncing

A USB device may be plugged-in or plugged-out from the USB host. To monitor the state of a USB device when it is detached from the USB host, the device controller provides hardware de-bouncing for USB V_{BUS} detection interrupt to avoid bounce problems on USB plug-in or plug-out. V_{BUS} detection interrupt appears about 10ms later than USB plug-in or plug-out. User can acknowledge USB plug-in/plug-out by reading USBD_VBUSDET register. The VBUSDET flag represents the current state on the bus without de-bouncing. If VBUSDET is 1, it means the USB cable is plugged-in. If user polls the flag to check USB state, software de-bouncing must be added if needed.

6.32.5.5 Interrupt Control

This USB provides 1 interrupt vector with 5 interrupt events (NEVWK, VBDET, USB, BUS and SOF). The NEVWK event occurs after waking up the system from Power-down mode (The power-down mode function is defined in system power-down control register, CLK_PWRCTL). The VBDET event is used for USB plug-in or plug-out. The USB event notifies users of some USB requests, such as IN ACK, OUT ACK. And the BUS event notifies users of some bus events, such as suspend and resume. The related bits must be set in the interrupt enable register (USBD_INTEN) of USB Device Controller to enable USB interrupts.

NEVWK interrupt is only presented when no the other USB interrupt events happened more than 20ms after the chip is waked up from Power-down mode. After the chip enters Power-down mode, any change on USB_VBUS, USB_D+ and USB_D- can wake up this chip if USB wake-up function is enabled. If this change is not intentionally, no interrupt but NEVWK interrupt will occur. After waking up by USB, this interrupt will occur when no the other USB interrupt events are presented for more than 20ms. Figure 6.32-2 shows the control flow of wake-up interrupt.

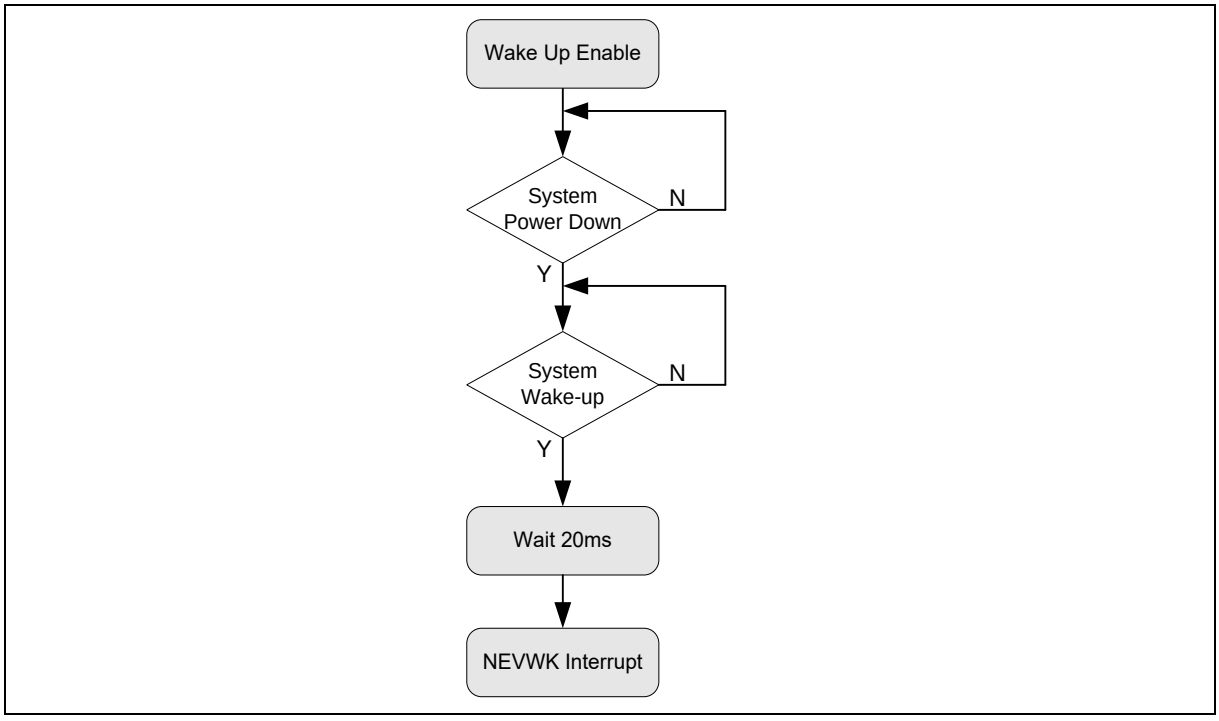


Figure 6.32-2 NEVWK Interrupt Operation Flow

The USB interrupt is used to notify users of any USB event on the bus, and user can read EPSTS (USBD_EPSTS0~3) and EPEVT0~18 (USBD_INTSTS[27:16] or USBD_EPINTSTS[18:0]) to take necessary responses.

Same as USB interrupt, BUS interrupt notifies users of some bus events, like USB reset, suspend, time-out, and resume. A user can read USBD_ATTR to acknowledge bus events.

6.32.5.6 Power Saving

User can write 0 to USBD_ATTR[4] to disable PHY under special circumstances, like suspend, to conserve power.

6.32.5.7 Buffer Control

There is 1 Kbytes SRAM in the controller and the 19 endpoints share this buffer. User shall configure each endpoint's effective starting address in the buffer segmentation register before the USB function active. The "Buffer Control" block is used to control each endpoint's effective starting address and its SRAM size is defined in the USBD_MXPLD0~18 register.

Figure 6.32-3 depicts the starting address for each endpoint according the content of USBD_BUFSEG0~18 and USBD_MXPLD0~18 registers. If the USBD_BUFSEG0 is programmed as 0x08h and USBD_MXPLD0 is set as 0x40h, the SRAM size of endpoint 0 starts from USBD_BA+0x808h and end in USBD_BA+0x848h.

Note: The USBD SRAM base is USBD_BA+0x800h.

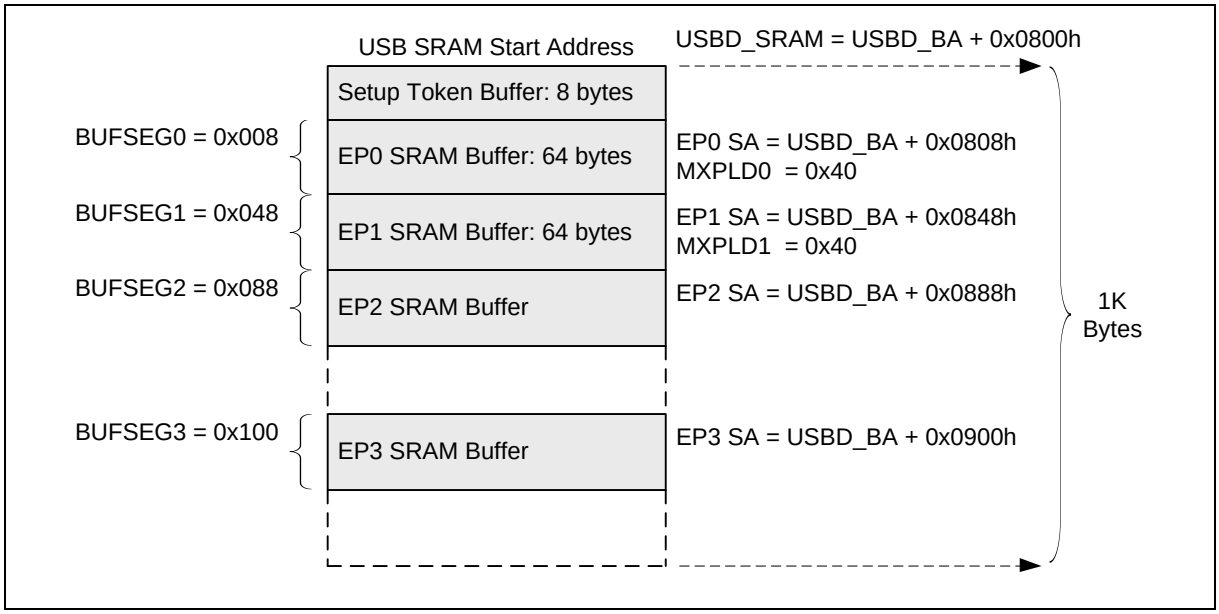


Figure 6.32-3 Endpoint SRAM Structure

6.32.5.8 Handling Transactions with USB Device Peripheral

User can use interrupt or polling USBD_INTSTS to monitor the USB transactions. When transactions occur, USBD_INTSTS will be set by hardware and send an interrupt request to CPU (if related interrupt enabled), or user can polling USBD_INTSTS to get these events without interrupt. The following is the control flow with interrupt enabled.

When USB host has requested data from a device controller, user needs to prepare related data in the specified endpoint buffer in advance. After buffering the required data, user needs to write the actual data length in the specified USBD_MXPLD0~18 register. Once this register is written, the internal signal “In_Rdy” will be asserted and the buffering data will be transmitted immediately after receiving associated IN token from Host. Note that after transferring the specified data, the signal “In_Rdy” will de-assert automatically by hardware.

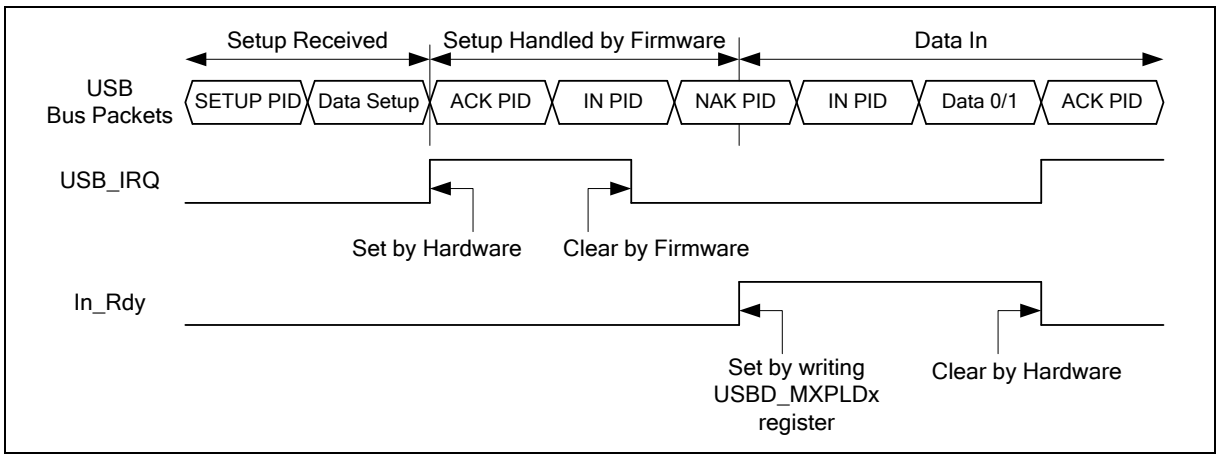


Figure 6.32-4 Setup Transaction Followed by Data IN Transaction

Alternatively, when USB host wants to transmit data to the OUT endpoint in the device controller, hardware will buffer these data to the specified endpoint buffer. After this transaction is completed, hardware will record the data length in specified USBD_MXPLD0~18 register and de-assert the internal signal “Out_Rdy”. This will avoid hardware accepting next transaction until user moves out the current data in the related endpoint buffer. Once users have processed this transaction, the specified

USBD_MXPLD0~18 register needs to be written by firmware to assert the signal “Out_Rdy” again to accept the next transaction.

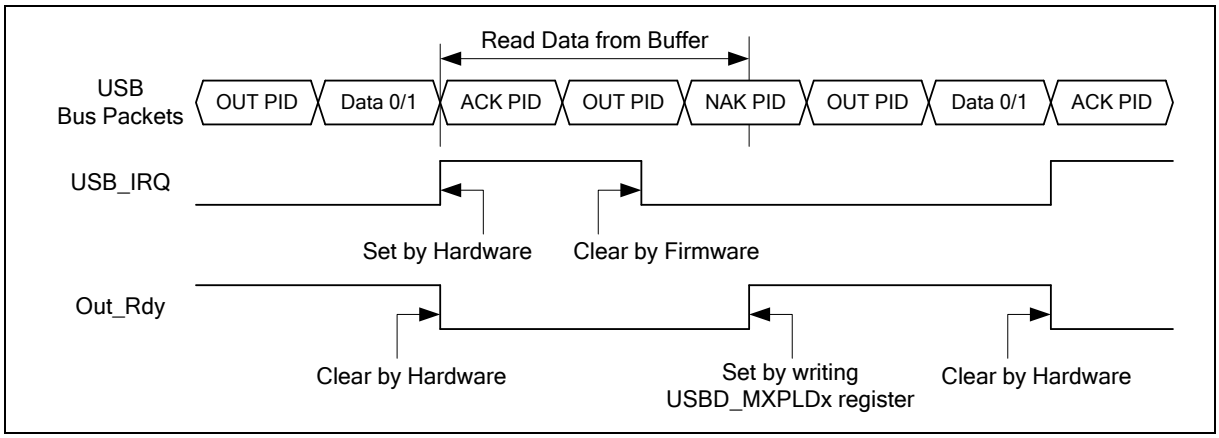


Figure 6.32-5 Data Out Transfer

6.32.5.9 Double Buffer Transfer

When user sets double buffer to continuously transfer or receive data, hardware will automatically toggle endpoint buffer to access data after transaction is finished. User has much time to process the last transaction data and set the next transaction data to reduce the chances that the device replies NAK to the host.

There are 19 endpoints buffer. User can set single buffer + double buffer * 2 = 19, and set double buffer by “DBEN”, ” DBTGACTIVE” in USB_CFG0~18. The double buffer function supports IN, OUT and ISO transaction.

The USB device receives endpoint number and token to decide which endpoint buffer will be used. If setting double buffer (DBEN = 1), active endpoint (DBTGACTIVE =1) will transfer or receive data.

When double buffer is enabled, DBTGACTIVE will be toggled automatically by hardware after successful transfer, and DSQSYNC will remain stable.

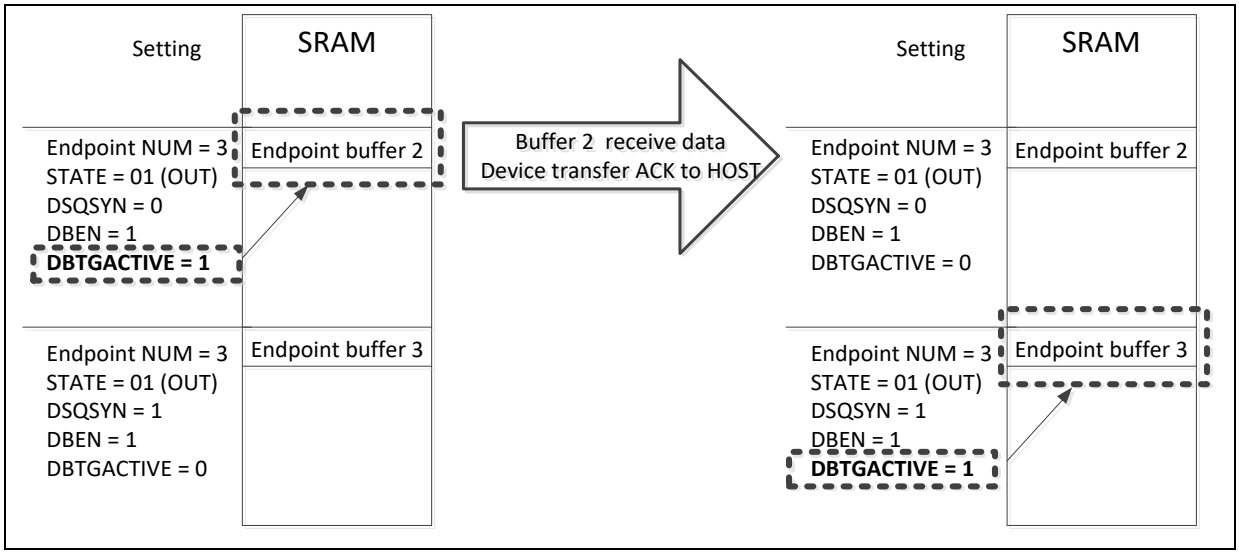


Figure 6.32-7 Double Buffer Change Process

6.32.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
USBD Base Address: **USBD_BA = 0x400C_0000**				
USBD_INTEN	USBD_BA+0x000	R/W	USB Device Interrupt Enable Register	0x0000_0000
USBD_INTSTS	USBD_BA+0x004	R/W	USB Device Interrupt Event Status Register	0x0000_0000
USBD_FADDR	USBD_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000
USBD_EPSTS	USBD_BA+0x00C	R	USB Device Endpoint Status Register	0x0000_0000
USBD_ATTR	USBD_BA+0x010	R/W	USB Device Bus Status and Attribution Register	0x0000_0040
USBD_VBUSDET	USBD_BA+0x014	R	USB Device V_{BUS} Detection Register	0x0000_0000
USBD_STBUFSEG	USBD_BA+0x018	R/W	SETUP Token Buffer Segmentation Register	0x0000_0000
USBD_EPSTS0	USBD_BA+0x020	R	USB Device Endpoint Status Register 0	0x0000_0000
USBD_EPSTS1	USBD_BA+0x024	R	USB Device Endpoint Status Register 1	0x0000_0000
USBD_EPSTS2	USBD_BA+0x028	R	USB Device Endpoint Status Register 2	0x0000_0000
USBD_EPINTSTS	USBD_BA+0x030	R/W	USB Device Endpoint Interrupt Event Status Register	0x0000_0000
USBD_FN	USBD_BA+0x08C	R	USB Frame Number Register	0x0000_0XXX
USBD_SE0	USBD_BA+0x090	R/W	USB Device Drive SE0 Control Register	0x0000_0001
USBD_BUFSEG0	USBD_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD0	USBD_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USBD_CFG0	USBD_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USBD_CFGP0	USBD_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG1	USBD_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD1	USBD_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USBD_CFG1	USBD_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USBD_CFGP1	USBD_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG2	USBD_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD2	USBD_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USBD_CFG2	USBD_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USBD_CFGP2	USBD_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG3	USBD_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD3	USBD_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000

USBD_CFG3	USBD_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USBD_CFGP3	USBD_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG4	USBD_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD4	USBD_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USBD_CFG4	USBD_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USBD_CFGP4	USBD_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG5	USBD_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD5	USBD_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USBD_CFG5	USBD_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USBD_CFGP5	USBD_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG6	USBD_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD6	USBD_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USBD_CFG6	USBD_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USBD_CFGP6	USBD_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG7	USBD_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD7	USBD_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000
USBD_CFG7	USBD_BA+0x578	R/W	Endpoint 7 Configuration Register	0x0000_0000
USBD_CFGP7	USBD_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG8	USBD_BA+0x580	R/W	Endpoint 8 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD8	USBD_BA+0x584	R/W	Endpoint 8 Maximal Payload Register	0x0000_0000
USBD_CFG8	USBD_BA+0x588	R/W	Endpoint 8 Configuration Register	0x0000_0000
USBD_CFGP8	USBD_BA+0x58C	R/W	Endpoint 8 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG9	USBD_BA+0x590	R/W	Endpoint 9 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD9	USBD_BA+0x594	R/W	Endpoint 9 Maximal Payload Register	0x0000_0000
USBD_CFG9	USBD_BA+0x598	R/W	Endpoint 9 Configuration Register	0x0000_0000
USBD_CFGP9	USBD_BA+0x59C	R/W	Endpoint 9 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG10	USBD_BA+0x5A0	R/W	Endpoint 10 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD10	USBD_BA+0x5A4	R/W	Endpoint 10 Maximal Payload Register	0x0000_0000
USBD_CFG10	USBD_BA+0x5A8	R/W	Endpoint 10 Configuration Register	0x0000_0000
USBD_CFGP10	USBD_BA+0x5AC	R/W	Endpoint 10 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG11	USBD_BA+0x5B0	R/W	Endpoint 11 Buffer Segmentation Register	0x0000_0000

USBD_MXPLD11	USBD_BA+0x5B4	R/W	Endpoint 11 Maximal Payload Register	0x0000_0000
USBD_CFG11	USBD_BA+0x5B8	R/W	Endpoint 11 Configuration Register	0x0000_0000
USBD_CFGP11	USBD_BA+0x5BC	R/W	Endpoint 11 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG12	USBD_BA+0x5C0	R/W	Endpoint 12 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD12	USBD_BA+0x5C4	R/W	Endpoint 12 Maximal Payload Register	0x0000_0000
USBD_CFG12	USBD_BA+0x5C8	R/W	Endpoint 12 Configuration Register	0x0000_0000
USBD_CFGP12	USBD_BA+0x5CC	R/W	Endpoint 12 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG13	USBD_BA+0x5D0	R/W	Endpoint 13 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD13	USBD_BA+0x5D4	R/W	Endpoint 13 Maximal Payload Register	0x0000_0000
USBD_CFG13	USBD_BA+0x5D8	R/W	Endpoint 13 Configuration Register	0x0000_0000
USBD_CFGP13	USBD_BA+0x5DC	R/W	Endpoint 13 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG14	USBD_BA+0x5E0	R/W	Endpoint 14 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD14	USBD_BA+0x5E4	R/W	Endpoint 14 Maximal Payload Register	0x0000_0000
USBD_CFG14	USBD_BA+0x5E8	R/W	Endpoint 14 Configuration Register	0x0000_0000
USBD_CFGP14	USBD_BA+0x5EC	R/W	Endpoint 14 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG15	USBD_BA+0x5F0	R/W	Endpoint 15 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD15	USBD_BA+0x5F4	R/W	Endpoint 15 Maximal Payload Register	0x0000_0000
USBD_CFG15	USBD_BA+0x5F8	R/W	Endpoint 15 Configuration Register	0x0000_0000
USBD_CFGP15	USBD_BA+0x5FC	R/W	Endpoint 15 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG16	USBD_BA+0x600	R/W	Endpoint 16 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD16	USBD_BA+0x604	R/W	Endpoint 16 Maximal Payload Register	0x0000_0000
USBD_CFG16	USBD_BA+0x608	R/W	Endpoint 16 Configuration Register	0x0000_0000
USBD_CFGP16	USBD_BA+0x60C	R/W	Endpoint 16 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG17	USBD_BA+0x610	R/W	Endpoint 17 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD17	USBD_BA+0x614	R/W	Endpoint 17 Maximal Payload Register	0x0000_0000
USBD_CFG17	USBD_BA+0x618	R/W	Endpoint 17 Configuration Register	0x0000_0000
USBD_CFGP17	USBD_BA+0x61C	R/W	Endpoint 17 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG18	USBD_BA+0x620	R/W	Endpoint 18 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD18	USBD_BA+0x624	R/W	Endpoint 18 Maximal Payload Register	0x0000_0000
USBD_CFG18	USBD_BA+0x628	R/W	Endpoint 18 Configuration Register	0x0000_0000
USBD_CFGP18	USBD_BA+0x62C	R/W	Endpoint 18 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

Memory Type	Address	Size	Description
USBD_BA = 0x400C_0000			
USBD_SRAM	USBD_BA+0x800 ~ USBD_BA+0xBFF	1 Kbyte	The SRAM is used for the entire endpoints buffer. Refer to section 6.32.5.7 for the endpoint SRAM structure and its description.

6.32.7 Register Description

USB Interrupt Enable Register (USBD_INTEN)

Register	Offset	R/W	Description	Reset Value
USBD_INTEN	USBD_BA+0x000	R/W	USB Device Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
INNAKEN	**Reserved**						**WKEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**SOFIEN**	**NEVWKIEN**	**VBDETIEN**	**USBIEN**	**BUSIEN**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**INNAKEN**	**Active NAK Function and Its Status in IN Token** 0 = When device responds NAK after receiving IN token, IN NAK status will not be updated to USBD_EPSTS0 and USBD_EPSTS1register, so that the USB interrupt event will not be asserted. 1 = IN NAK status will be updated to USBD_EPSTS0 and USBD_EPSTS1 register and the USB interrupt event will be asserted, when the device responds NAK after receiving IN token.
[14:9]	**Reserved**	Reserved.
[8]	**WKEN**	**Wake-up Function Enable Bit** 0 = USB wake-up function Disabled. 1 = USB wake-up function Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**SOFIEN**	**Start of Frame Interrupt Enable Bit** 0 = SOF Interrupt Disabled. 1 = SOF Interrupt Enabled.
[3]	**NEVWKIEN**	**USB No-event-wake-up Interrupt Enable Bit** 0 = No-event-wake-up Interrupt Disabled. 1 = No-event-wake-up Interrupt Enabled.
[2]	**VBDETIEN**	**V_{BUS} Detection Interrupt Enable Bit** 0 = V_{BUS} detection Interrupt Disabled. 1 = V_{BUS} detection Interrupt Enabled.
[1]	**USBIEN**	**USB Event Interrupt Enable Bit** 0 = USB event interrupt Disabled. 1 = USB event interrupt Enabled.

[0]	**BUSIEN**	**Bus Event Interrupt Enable Bit** 0 = BUS event interrupt Disabled. 1 = BUS event interrupt Enabled.

USB Interrupt Event Status Register (USBD_INTSTS)

Register	Offset	R/W	Description	Reset Value
USBD_INTSTS	USBD_BA+0x004	R/W	USB Device Interrupt Event Status Register	0x0000_0000

31	30	29	28	27	26	25	24
SETUP	Reserved			EPEVT11	EPEVT10	EPEVT9	EPEVT8
23	22	21	20	19	18	17	16
EPEVT7	EPEVT6	EPEVT5	EPEVT4	EPEVT3	EPEVT2	EPEVT1	EPEVT0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			SOFIF	NEVWKIF	VBDETIF	USBIF	BUSIF

Bits	Description	
[31]	**SETUP**	**Setup Event Status** 0 = No Setup event. 1 = Setup event occurred, cleared by writing 1 to USBD_INTSTS[31].
[30:28]	**Reserved**	Reserved.
[27]	**EPEVT11**	**Endpoint 11's USB Event Status** 0 = No event occurred in endpoint 11. 1 = USB event occurred on Endpoint 11. Check USBD_EPSTS1[15:12] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[27] or USBD_EPINTSTS[11] or USBD_INTSTS[1].
[26]	**EPEVT10**	**Endpoint 10's USB Event Status** 0 = No event occurred in endpoint 10. 1 = USB event occurred on Endpoint 10. Check USBD_EPSTS1[11:8] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[26] or USBD_EPINTSTS[10] or USBD_INTSTS[1].
[25]	**EPEVT9**	**Endpoint 9's USB Event Status** 0 = No event occurred in endpoint 9. 1 = USB event occurred on Endpoint 9. Check USBD_EPSTS1[7:4] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[25] or USBD_EPINTSTS[9] or USBD_INTSTS[1].
[24]	**EPEVT8**	**Endpoint 8's USB Event Status** 0 = No event occurred in endpoint 8. 1 = USB event occurred on Endpoint 8. Check USBD_EPSTS1[3:0] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[24] or USBD_EPINTSTS[8] or USBD_INTSTS[1].
[23]	**EPEVT7**	**Endpoint 7's USB Event Status** 0 = No event occurred in endpoint 7. 1 = USB event occurred on Endpoint 7. Check USBD_EPSTS0[31:28] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[23] or USBD_EPINTSTS[7] or USBD_INTSTS[1].

[22]	**EPEVT6**	**Endpoint 6's USB Event Status** 0 = No event occurred in endpoint 6. 1 = USB event occurred on Endpoint 6. Check USBD_EPSTS0[27:24] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[22] or USBD_EPINTSTS[6] or USBD_INTSTS[1].
[21]	**EPEVT5**	**Endpoint 5's USB Event Status** 0 = No event occurred in endpoint 5. 1 = USB event occurred on Endpoint 5. Check USBD_EPSTS0[23:20] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[21] or USBD_EPINTSTS[5] or USBD_INTSTS[1].
[20]	**EPEVT4**	**Endpoint 4's USB Event Status** 0 = No event occurred in endpoint 4. 1 = USB event occurred on Endpoint 4. Check USBD_EPSTS0[19:16] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[20] or USBD_EPINTSTS[4] or USBD_INTSTS[1].
[19]	**EPEVT3**	**Endpoint 3's USB Event Status** 0 = No event occurred in endpoint 3. 1 = USB event occurred on Endpoint 3. Check USBD_EPSTS0[15:12] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[19] or USBD_EPINTSTS[3] or USBD_INTSTS[1].
[18]	**EPEVT2**	**Endpoint 2's USB Event Status** 0 = No event occurred in endpoint 2. 1 = USB event occurred on Endpoint 2. Check USBD_EPSTS0[11:8] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[18] or USBD_EPINTSTS[2] or USBD_INTSTS[1].
[17]	**EPEVT1**	**Endpoint 1's USB Event Status** 0 = No event occurred in endpoint 1. 1 = USB event occurred on Endpoint 1. Check USBD_EPSTS0[7:4] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[17] or USBD_EPINTSTS[1] or USBD_INTSTS[1].
[16]	**EPEVT0**	**Endpoint 0's USB Event Status** 0 = No event occurred in endpoint 0. 1 = USB event occurred on Endpoint 0. Check USBD_EPSTS0[3:0] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[16] or USBD_EPINTSTS[0] or USBD_INTSTS[1].
[15:5]	**Reserved**	Reserved.
[4]	**SOFIF**	**Start of Frame Interrupt Status** 0 = SOF event did not occur. 1 = SOF event occurred, cleared by writing 1 to USBD_INTSTS[4].
[3]	**NEVWKIF**	**No-event-wake-up Interrupt Status** 0 = NEVWK event did not occur. 1 = No-event-wake-up event occurred, cleared by writing 1 to USBD_INTSTS[3].
[2]	**VBDETIF**	**V_{BUS} Detection Interrupt Status** 0 = There is not attached/detached event in the USB. 1 = There is attached/detached event in the USB bus and it is cleared by writing 1 to USBD_INTSTS[2].
[1]	**USBIF**	**USB Event Interrupt Status** The USB event includes the SETUP Token, IN Token, OUT ACK, ISO IN, or ISO OUT events in the bus.

		0 = No USB event occurred. 1 = USB event occurred. Check EPSTS0~25 in USBD_EPSTS0~3 to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[1] or USBD_INTSTS[27:16] or USBD_EPINTSTS[24:0] and SETUP (USBD_INTSTS[31]).
[0]	**BUSIF**	**BUS Interrupt Status** The BUS event means that there is one of the suspense or the resume function in the bus. 0 = No BUS event occurred. 1 = Bus event occurred; check USBD_ATTR[3:0] to know which kind of bus event was occurred, cleared by writing 1 to USBD_INTSTS[0].

USB Device Function Address Register (USBD_FADDR)

A 7-bit value is used as the address of a device on the USB BUS.

Register	Offset	R/W	Description	Reset Value
USBD_FADDR	USBD_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**FADDR**						

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:0]	**FADDR**	USB device function address

USB Endpoint Status Register (USBD_EPSTS)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS	USBD_BA+0x00C	R	USB Device Endpoint Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OV	**Reserved**						

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**OV**	**Overrun** It indicates that the received data is over the maximum payload number or not. 0 = No overrun. 1 = Out Data is more than the Max Payload in MXPLD register or the Setup Data is more than 8 bytes.
[6:0]	**Reserved**	Reserved.

USB Bus Status and Attribution Register (USBD_ATTR)

Register	Offset	R/W	Description	Reset Value
USBD_ATTR	USBD_BA+0x010	R/W	USB Device Bus Status and Attribution Register	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BYTEM	PWRDN	DPPUEN
7	6	5	4	3	2	1	0
USBEN	Reserved	RWAKEUP	PHYEN	TOUT	RESUME	SUSPEND	USBRST

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**BYTEM**	**CPU Access USB SRAM Size Mode Selection** 0 = Word mode: The size of the transfer from CPU to USB SRAM can be Word only. 1 = Byte mode: The size of the transfer from CPU to USB SRAM can be Byte only.
[9]	**PWRDN**	**Power-down PHY Transceiver, Low Active** 0 = Power-down related circuit of PHY transceiver. 1 = Turn-on related circuit of PHY transceiver.
[8]	**DPPUEN**	**Pull-up Resistor on USB_DP Enable Bit** 0 = Pull-up resistor in USB_D+ bus Disabled. 1 = Pull-up resistor in USB_D+ bus Active.
[7]	**USBEN**	**USB Controller Enable Bit** 0 = USB Controller Disabled. 1 = USB Controller Enabled.
[6]	**Reserved**	Reserved.
[5]	**RWAKEUP**	**Remote Wake-up** 0 = Release the USB bus from K state. 1 = Force USB bus to K (USB_D+ low, USB_D- high) state, used for remote wake-up.
[4]	**PHYEN**	**PHY Transceiver Function Enable Bit** 0 = PHY transceiver function Disabled. 1 = PHY transceiver function Enabled.
[3]	**TOUT**	**Time-out Status (Read Only)** 0 = No time-out. 1 = No Bus response more than 18 bits time.
[2]	**RESUME**	**Resume Status (Read Only)** 0 = No bus resume.

		1 = Resume from suspend.
[1]	**SUSPEND**	**Suspend Status (Read Only)** 0 = Bus no suspend. 1 = Bus idle more than 3ms, either cable is plugged-out or host is sleeping.
[0]	**USBRST**	**USB Reset Status (Read Only)** 0 = Bus no reset. 1 = Bus reset when SE0 (single-ended 0) more than 2.5us.

USB Device V_{BUS} Detection Register (USBD_VBUSDET)

Register	Offset	R/W	Description	Reset Value
USBD_VBUSDET	USBD_BA+0x014	R	USB Device V_{BUS} Detection Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**VBUSDET**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**VBUSDET**	**Device V_{BUS} Detection** 0 = Controller is not attached to the USB host. 1 = Controller is attached to the USB host.

USB SETUP Token Buffer Segmentation Register (USBD_STBUFSEG)

Register	Offset	R/W	Description	Reset Value
USBD_STBUFSEG	USBD_BA+0x018	R/W	SETUP Token Buffer Segmentation Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**STBUFSEG**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STBUFSEG					**Reserved**		

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:3]	**STBUFSEG**	**SETUP Token Buffer Segmentation** It is used to indicate the offset address for the SETUP token with the USB Device SRAM starting address. The effective starting address is USBD_SRAM address + {STBUFSEG, 3'b000} Where the USBD_SRAM address = USBD_BA+0x800h. **Note:** It is used for SETUP token only.
[2:0]	**Reserved**	Reserved.

USB Endpoint Status Register 0 (USBD_EPSTS0)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS0	USBD_BA+0x020	R	USB Device Endpoint Status Register 0	0x0000_0000

31	30	29	28	27	26	25	24
EPSTS7				EPSTS6			
23	22	21	20	19	18	17	16
EPSTS5				EPSTS4			
15	14	13	12	11	10	9	8
EPSTS3				EPSTS2			
7	6	5	4	3	2	1	0
EPSTS1				EPSTS0			

Bits	Description	
[31:28]	**EPSTS7**	**Endpoint 7 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[27:24]	**EPSTS6**	**Endpoint 6 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[23:20]	**EPSTS5**	**Endpoint 5 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

[19:16]	**EPSTS4**	**Endpoint 4 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous Transfer End.
[15:12]	**EPSTS3**	**Endpoint 3 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[11:8]	**EPSTS2**	**Endpoint 2 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[7:4]	**EPSTS1**	**Endpoint 1 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[3:0]	**EPSTS0**	**Endpoint 0 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

USB Endpoint Status Register 1 (USBD_EPSTS1)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS1	USBD_BA+0x024	R	USB Device Endpoint Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
EPSTS15				EPSTS14			
23	22	21	20	19	18	17	16
EPSTS13				EPSTS12			
15	14	13	12	11	10	9	8
EPSTS11				EPSTS10			
7	6	5	4	3	2	1	0
EPSTS9				EPSTS8			

Bits	Description	
[31:28]	**EPSTS15**	**Endpoint 15 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[27:24]	**EPSTS14**	**Endpoint 14 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[23:20]	**EPSTS13**	**Endpoint 13 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[19:16]	**EPSTS12**	**Endpoint 12 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

[15:12]	**EPSTS11**	**Endpoint 11 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[11:8]	**EPSTS10**	**Endpoint 10 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[7:4]	**EPSTS9**	**Endpoint 9 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[3:0]	**EPSTS8**	**Endpoint 8 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

USB Endpoint Status Register 2 (USBD_EPSTS2)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS2	USBD_BA+0x028	R	USB Device Endpoint Status Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**EPSTS18**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPSTS17				**EPSTS16**			

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**EPSTS18**	**Endpoint 18 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[7:4]	**EPSTS17**	**Endpoint 17 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[3:0]	**EPSTS16**	**Endpoint 16 Status** These bits are used to indicate the current status of this endpoint. 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

USB Endpoint Interrupt Event Status Register (USBD_EPINTSTS)

Register	Offset	R/W	Description	Reset Value
USBD_EPINTSTS	USBD_BA+0x030	R/W	USB Device Endpoint Interrupt Event Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**EPEVT18**	**EPEVT17**	**EPEVT16**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
EPEVT15	**EPEVT14**	**EPEVT13**	**EPEVT12**	**EPEVT11**	**EPEVT10**	**EPEVT9**	**EPEVT8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPEVT7	**EPEVT6**	**EPEVT5**	**EPEVT4**	**EPEVT3**	**EPEVT2**	**EPEVT1**	**EPEVT0**

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**EPEVT18**	**Endpoint 18's USB Event Status** 0 = No event occurred in endpoint 18. 1 = USB event occurred on Endpoint 18. Check USBD_EPSTS2[11:8] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[18] or USBD_INTSTS[1].
[17]	**EPEVT17**	**Endpoint 17's USB Event Status** 0 = No event occurred in endpoint 17. 1 = USB event occurred on Endpoint 17. Check USBD_EPSTS2[7:4] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[17] or USBD_INTSTS[1].
[16]	**EPEVT16**	**Endpoint 16's USB Event Status** 0 = No event occurred in endpoint 16. 1 = USB event occurred on Endpoint 16. Check USBD_EPSTS2[3:0] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[16] or USBD_INTSTS[1].
[15]	**EPEVT15**	**Endpoint 15's USB Event Status** 0 = No event occurred in endpoint 15. 1 = USB event occurred on Endpoint 15. Check USBD_EPSTS1[31:28] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[15] or USBD_INTSTS[1].
[14]	**EPEVT14**	**Endpoint 14's USB Event Status** 0 = No event occurred in endpoint 14. 1 = USB event occurred on Endpoint 14. Check USBD_EPSTS1[27:24] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[14] or USBD_INTSTS[1].
[13]	**EPEVT13**	**Endpoint 13's USB Event Status** 0 = No event occurred in endpoint 13. 1 = USB event occurred on Endpoint 13. Check USBD_EPSTS1[23:20] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[13] or

		USBD_INTSTS[1].
[12]	**EPEVT12**	**Endpoint 12's USB Event Status** 0 = No event occurred in endpoint 12. 1 = USB event occurred on Endpoint 12. Check USBD_EPSTS1[19:16] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[12] or USBD_INTSTS[1].
[11]	**EPEVT11**	**Endpoint 11's USB Event Status** 0 = No event occurred in endpoint 11. 1 = USB event occurred on Endpoint 11. Check USBD_EPSTS1[15:12] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[11] or USBD_INTSTS[27] or USBD_INTSTS[1].
[10]	**EPEVT10**	**Endpoint 10's USB Event Status** 0 = No event occurred in endpoint 10. 1 = USB event occurred on Endpoint 10. Check USBD_EPSTS1[11:8] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[10] or USBD_INTSTS[26] or USBD_INTSTS[1].
[9]	**EPEVT9**	**Endpoint 9's USB Event Status** 0 = No event occurred in endpoint 9. 1 = USB event occurred on Endpoint 9. Check USBD_EPSTS1[7:4] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[9] or USBD_INTSTS[25] or USBD_INTSTS[1].
[8]	**EPEVT8**	**Endpoint 8's USB Event Status** 0 = No event occurred in endpoint 8. 1 = USB event occurred on Endpoint 8. Check USBD_EPSTS1[3:0] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[8] or USBD_INTSTS[24] or USBD_INTSTS[1].
[7]	**EPEVT7**	**Endpoint 7's USB Event Status** 0 = No event occurred in endpoint 7. 1 = USB event occurred on Endpoint 7. Check USBD_EPSTS0[31:28] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[7] or USBD_INTSTS[23] or USBD_INTSTS[1].
[6]	**EPEVT6**	**Endpoint 6's USB Event Status** 0 = No event occurred in endpoint 6. 1 = USB event occurred on Endpoint 6. Check USBD_EPSTS0[27:24] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[6] or USBD_INTSTS[22] or USBD_INTSTS[1].
[5]	**EPEVT5**	**Endpoint 5's USB Event Status** 0 = No event occurred in endpoint 5. 1 = USB event occurred on Endpoint 5. Check USBD_EPSTS0[23:20] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[5] or USBD_INTSTS[21] or USBD_INTSTS[1].
[4]	**EPEVT4**	**Endpoint 4's USB Event Status** 0 = No event occurred in endpoint 4. 1 = USB event occurred on Endpoint 4. Check USBD_EPSTS0[19:16] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[4] or USBD_INTSTS[20] or USBD_INTSTS[1].
[3]	**EPEVT3**	**Endpoint 3's USB Event Status** 0 = No event occurred in endpoint 3. 1 = USB event occurred on Endpoint 3. Check USBD_EPSTS0[15:12] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[3] or USBD_INTSTS[19] or USBD_INTSTS[1].

[2]	**EPEVT2**	**Endpoint 2's USB Event Status** 0 = No event occurred in endpoint 2. 1 = USB event occurred on Endpoint 2. Check USBD_EPSTS0[11:8] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[2] or USBD_EPINTSTS[2] or USBD_INTSTS[18] or USBD_INTSTS[1].
[1]	**EPEVT1**	**Endpoint 1's USB Event Status** 0 = No event occurred in endpoint 1. 1 = USB event occurred on Endpoint 1. Check USBD_EPSTS0[7:4] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[1] or USBD_INTSTS[17] or USBD_INTSTS[1].
[0]	**EPEVT0**	**Endpoint 0's USB Event Status** 0 = No event occurred in endpoint 0. 1 = USB event occurred on Endpoint 0. Check USBD_EPSTS0[3:0] to know which kind of USB event was occurred, cleared by writing 1 to USBD_EPINTSTS[0] or USBD_INTSTS[16] or USBD_INTSTS[1].

USB Frame Number Register (USBD_FN)

Register	Offset	R/W	Description	Reset Value
USBD_FN	USBD_BA+0x08C	R	USB Frame Number Register	0x0000_0XXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**FN**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FN							

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**FN**	**Frame Number** These bits contain the 11-bits frame number in the last received SOF packet.

USB Drive SE0 Register (USBD_SE0)

Register	Offset	R/W	Description	Reset Value
USBD_SE0	USBD_BA+0x090	R/W	USB Device Drive SE0 Control Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SE0**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SE0**	**Drive Single Ended Zero in USB Bus** The Single Ended Zero (SE0) is when both lines (USB_D+ and USB_D-) are being pulled low. 0 = Normal operation. 1 = Force USB PHY transceiver to drive SE0.

USB Buffer Segmentation Register (USBD_BUFSEG0~18)

Register	Offset	R/W	Description	Reset Value
USBD_BUFSEG0	USBD_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG1	USBD_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG2	USBD_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG3	USBD_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG4	USBD_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG5	USBD_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG6	USBD_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG7	USBD_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG8	USBD_BA+0x580	R/W	Endpoint 8 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG9	USBD_BA+0x590	R/W	Endpoint 9 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG10	USBD_BA+0x5A0	R/W	Endpoint 10 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG11	USBD_BA+0x5B0	R/W	Endpoint 11 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG12	USBD_BA+0x5C0	R/W	Endpoint 12 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG13	USBD_BA+0x5D0	R/W	Endpoint 13 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG14	USBD_BA+0x5E0	R/W	Endpoint 14 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG15	USBD_BA+0x5F0	R/W	Endpoint 15 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG16	USBD_BA+0x600	R/W	Endpoint 16 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG17	USBD_BA+0x610	R/W	Endpoint 17 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG18	USBD_BA+0x620	R/W	Endpoint 18 Buffer Segmentation Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**BUFSEG**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BUFSEG					**Reserved**		

Bits	Description

[31:10]	**Reserved**	Reserved.
[9:3]	**BUFSEG**	**Endpoint Buffer Segmentation** It is used to indicate the offset address for each endpoint with the USB SRAM starting address The effective starting address of the endpoint is USBD_SRAM address + {BUFSEG, 3'b000} Where the USBD_SRAM address = USBD_BA+0x800h. Refer to the section 6.32.5.7 for the endpoint SRAM structure and its description.
[2:0]	**Reserved**	Reserved.

USB Maximal Payload Register (USBD_MXPLD0~18)

Register	Offset	R/W	Description	Reset Value
USBD_MXPLD0	USBD_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USBD_MXPLD1	USBD_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USBD_MXPLD2	USBD_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USBD_MXPLD3	USBD_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USBD_MXPLD4	USBD_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USBD_MXPLD5	USBD_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USBD_MXPLD6	USBD_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USBD_MXPLD7	USBD_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000
USBD_MXPLD8	USBD_BA+0x584	R/W	Endpoint 8 Maximal Payload Register	0x0000_0000
USBD_MXPLD9	USBD_BA+0x594	R/W	Endpoint 9 Maximal Payload Register	0x0000_0000
USBD_MXPLD10	USBD_BA+0x5A4	R/W	Endpoint 10 Maximal Payload Register	0x0000_0000
USBD_MXPLD11	USBD_BA+0x5B4	R/W	Endpoint 11 Maximal Payload Register	0x0000_0000
USBD_MXPLD12	USBD_BA+0x5C4	R/W	Endpoint 12 Maximal Payload Register	0x0000_0000
USBD_MXPLD13	USBD_BA+0x5D4	R/W	Endpoint 13 Maximal Payload Register	0x0000_0000
USBD_MXPLD14	USBD_BA+0x5E4	R/W	Endpoint 14 Maximal Payload Register	0x0000_0000
USBD_MXPLD15	USBD_BA+0x5F4	R/W	Endpoint 15 Maximal Payload Register	0x0000_0000
USBD_MXPLD16	USBD_BA+0x604	R/W	Endpoint 16 Maximal Payload Register	0x0000_0000
USBD_MXPLD17	USBD_BA+0x614	R/W	Endpoint 17 Maximal Payload Register	0x0000_0000
USBD_MXPLD18	USBD_BA+0x624	R/W	Endpoint 18 Maximal Payload Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**MXPLD**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MXPLD							

Bits	Description

[31:10]	**Reserved**	Reserved.
[9:0]	**MXPLD**	**Maximal Payload** Define the data length which is transmitted to host (IN token) or the actual data length which is received from the host (OUT token). It also used to indicate that the endpoint is ready to be transmitted in IN token or received in OUT token. (1) When the register is written by CPU, For IN token, the value of MXPLD is used to define the data length to be transmitted and indicate the data buffer is ready. For OUT token, it means that the controller is ready to receive data from the host and the value of MXPLD is the maximal data length comes from host. (2) When the register is read by CPU, For IN token, the value of MXPLD is indicated by the data length be transmitted to host For OUT token, the value of MXPLD is indicated the actual data length receiving from host. **Note:** Once MXPLD is written, the data packets will be transmitted/received immediately after IN/OUT token arrived.

USB Configuration Register (USBD_CFG0~18)

Register	Offset	R/W	Description	Reset Value
USBD_CFG0	USBD_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USBD_CFG1	USBD_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USBD_CFG2	USBD_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USBD_CFG3	USBD_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USBD_CFG4	USBD_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USBD_CFG5	USBD_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USBD_CFG6	USBD_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USBD_CFG7	USBD_BA+0x578	R/W	Endpoint 7 Configuration Register	0x0000_0000
USBD_CFG8	USBD_BA+0x588	R/W	Endpoint 8 Configuration Register	0x0000_0000
USBD_CFG9	USBD_BA+0x598	R/W	Endpoint 9 Configuration Register	0x0000_0000
USBD_CFG10	USBD_BA+0x5A8	R/W	Endpoint 10 Configuration Register	0x0000_0000
USBD_CFG11	USBD_BA+0x5B8	R/W	Endpoint 11 Configuration Register	0x0000_0000
USBD_CFG12	USBD_BA+0x5C8	R/W	Endpoint 12 Configuration Register	0x0000_0000
USBD_CFG13	USBD_BA+0x5D8	R/W	Endpoint 13 Configuration Register	0x0000_0000
USBD_CFG14	USBD_BA+0x5E8	R/W	Endpoint 14 Configuration Register	0x0000_0000
USBD_CFG15	USBD_BA+0x5F8	R/W	Endpoint 15 Configuration Register	0x0000_0000
USBD_CFG16	USBD_BA+0x608	R/W	Endpoint 16 Configuration Register	0x0000_0000
USBD_CFG17	USBD_BA+0x618	R/W	Endpoint 17 Configuration Register	0x0000_0000
USBD_CFG18	USBD_BA+0x628	R/W	Endpoint 18 Configuration Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**DBEN**	**DBTGACTIVE**	**CSTALL**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DSQSYNC	**STATE**		**ISOCH**	**EPNUM**			

Bits	Description

[31:12]	**Reserved**	Reserved.
[11]	**DBEN**	**Double Buffer Enable Bit** 0 = Single buffer mode. 1 = Double buffer mode.
[10]	**DBTGACTIVE**	**Double Buffer Toggle Active Bit** 0 = Inactive in double buffer mode. 1 = Active in double buffer mode. When DBEN = 1 and endpoint successful transfer, DBTGACTIVE of each double buffer will be toggled automatically by hardware.
[9]	**CSTALL**	**Clear STALL Response** 0 = Disable the device to clear the STALL handshake in setup stage. 1 = Clear the device to response STALL handshake in setup stage.
[8]	**Reserved**	Reserved.
[7]	**DSQSYNC**	**Data Sequence Synchronization** 0 = DATA0 PID. 1 = DATA1 PID. For IN token, DSQSYNC specify DATA0 or DATA1 PID in transfer data packet For OUT token, DSQSYNC specify DATA0 or DATA1 PID in received data packet. DSQSYNC will be toggled automatically by hardware when IN or OUT token transfer successfully in single buffer mode, but will not be toggled in double buffer mode. **Note 1:** When double buffer is enabled, hardware will automatically write 0 to DSQSYNC with active double buffer and write 1 to DSQSYNC with inactive double buffer. **Note 2:** It will not be toggled by hardware when DBEN = 1. USB data toggle will be guaranteed by changing endpoint.
[6:5]	**STATE**	**Endpoint State** 00 = Endpoint is Disabled. 01 = Out endpoint. 10 = IN endpoint. 11 = Undefined.
[4]	**ISOCH**	**Isochronous Endpoint** This bit is used to set the endpoint as Isochronous endpoint, no handshake. 0 = No Isochronous endpoint. 1 = Isochronous endpoint.
[3:0]	**EPNUM**	**Endpoint Number** These bits are used to define the endpoint number of the current endpoint.

USB Extra Configuration Register (USBD_CFGP0~18)

Register	Offset	R/W	Description	Reset Value
USBD_CFGP0	USBD_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP1	USBD_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP2	USBD_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP3	USBD_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP4	USBD_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP5	USBD_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP6	USBD_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP7	USBD_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP8	USBD_BA+0x58C	R/W	Endpoint 8 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP9	USBD_BA+0x59C	R/W	Endpoint 9 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP10	USBD_BA+0x5AC	R/W	Endpoint 10 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP11	USBD_BA+0x5BC	R/W	Endpoint 11 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP12	USBD_BA+0x5CC	R/W	Endpoint 12 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP13	USBD_BA+0x5DC	R/W	Endpoint 13 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP14	USBD_BA+0x5EC	R/W	Endpoint 14 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP15	USBD_BA+0x5FC	R/W	Endpoint 15 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP16	USBD_BA+0x60C	R/W	Endpoint 16 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP17	USBD_BA+0x61C	R/W	Endpoint 17 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP18	USBD_BA+0x62C	R/W	Endpoint 18 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**SSTALL**	**CLRRDY**

Bits	Description

[31:2]	**Reserved**	Reserved.
[1]	**SSTALL**	**Set STALL Bit** 0 = Disable the device to response STALL. 1 = Set the device to respond STALL automatically.
[0]	**CLRRDY**	**Clear Ready Bit** When the USBD_MXPLD0~18 register is set by user, it means that the endpoint is ready to transmit or receive data. If the user wants to disable this transaction before the transaction start, users can set this bit to 1 to disable it and it is automatically clear to 0. For IN token, write '1' to clear the IN token had ready to transmit the data to USB. For OUT token, write '1' to clear the OUT token had ready to receive the data from USB. This bit is written 1 only and is always 0 when it is read back.

6.33 USB 1.1 Host Controller (USBH)

6.33.1 Overview

This chip is equipped with a USB 1.1 Host Controller (USBH) that supports Open Host Controller Interface (OpenHCI, OHCI) Specification, a register-level description of a host controller, to manage the devices and data transfer of Universal Serial Bus (USB).

The USBH supports an integrated Root Hub with a USB port, a DMA for real-time data transfer between system memory and USB bus, port power control and port overcurrent detection.

The USBH is responsible for detecting the connect and disconnect of USB devices, managing data transfer, collecting status and activity of USB bus, providing power control and detecting overcurrent of attached USB devices.

6.33.2 Features

- Compliant with Universal Serial Bus (USB) Specification Revision 1.1.
- Supports Open Host Controller Interface (OpenHCI) Specification Revision 1.0.
- Supports both full-speed (12Mbps) and low-speed (1.5Mbps) USB devices.
- Supports Control, Bulk, Interrupt and Isochronous transfers.
- Supports an integrated Root Hub.
- Supports a USB host port shared with USB device (OTG function).
- Supports port power control and port overcurrent detection.
- Supports DMA for real-time data transfer.

6.33.3 Block Diagram

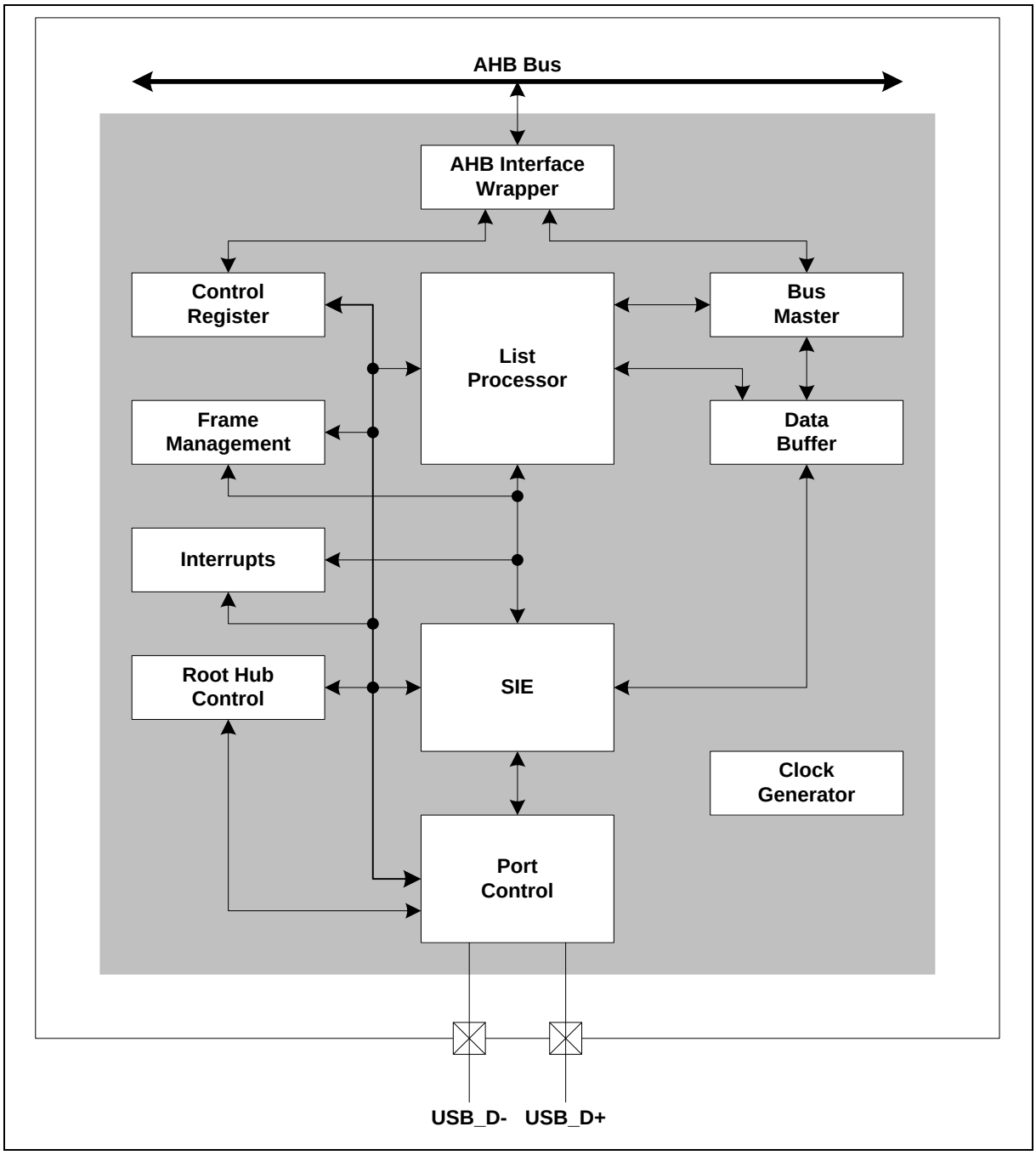


Figure 6.33-1 USB 1.1 Host Controller Block Diagram

6.33.4 Basic Configuration

6.33.4.1 *USBH OHCI Controller Basic Configuration*

- Clock Source Configuration
  - Select the source of USBH engine clock on USBSEL (CLK_CLKSEL0[8]).
  - Select the clock divider number of USBH engine clock on USBDIV (CLK_CLKDIV0[7:4]).
  - Enable USBH engine clock in USBHCKEN (CLK_AHBCLK[16]).
  - Select HCLK clock source to be more than or equal to 24 MHz.
- Reset Configuration
  - Reset USBH controller in USBHRST (SYS_IPRST0[4]).
-

6.33.5 Functional Description

6.33.5.1 *OHCI Controller*

AHB Interface

The OpenHCI Host Controller is connected to the system by the AHB bus. The design requires both master and slave bus operations. As a master, the Host Controller is responsible for running cycles on the AHB bus to access EDs and TDs as well as transferring data between memory and the local data buffer. As a slave, the Host Controller monitors the cycles on the AHB bus and determines when to respond to these cycles. Configuration and non-real-time control access to the Host Controller operational registers are through the AHB bus slave interface.

Host Controller

The host controller includes 5 functional blocks, including List Processing, Frame Management, Interrupt Processing, Host Controller Bus Master and Data Buffer.

The List Processor manages the data structures from the Host Controller Driver and coordinates all activity within the Host Controller.

The Frame Management is responsible for managing the frame specific tasks required by the USB specification and the OpenHCI specification. These tasks are:

- Management of the OpenHCI frame specific Operational Registers
- Operation of the Largest Data Packet Counter.
- Performing frame qualifications on USB Transaction requests to the SI.E.
- Generate SOF token requests to the SI.E.

Interrupts are the communication method for HC-initiated communication with the Host Controller Driver. There are several events that may trigger an interrupt from the Host Controller. Each specific event sets a specific bit in the HcInterruptStatus register.

The Host Controller Bus Master is the central block in the data path. The Host Controller Bus Master coordinates all access to the AHB Interface. There are two sources of bus mastering within Host Controller: the List Processor and the Data Buffer Engine.

The Data Buffer serves as the data interface between the Bus Master and the SI.E. It is a combination of a 64-byte latched based bi-directional asynchronous FIFO and a single DWORD AHB Holding Register.

USB Interface

The USB interface includes the integrated Root Hub with an USB port, Port 1 as well as the Serial

Interface Engine (SIE) and USB clock generator. The interface combines responsibility for executing bus transactions requested by the HC as well as the hub and port management specified by USB.

The SIE is responsible for managing all transactions to the USB. It controls the bus protocol, packet generation/extraction, data parallel-to-serial conversion, CRC coding, bit stuffing, and NRZI encoding. All transactions on the USB are requested from the List Processor and Frame Manager.

The Root Hub is a collection of ports that are individually controlled and a hub that maintains control/status over functions common to all ports.

6.33.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
USBH Base Address: **USBH_BA = 0x4000_9000**				
HcRevision	USBH_BA+0x000	R	Host Controller Revision Register	0x0000_0110
HcControl	USBH_BA+0x004	R/W	Host Controller Control Register	0x0000_0000
HcCommandStatus	USBH_BA+0x008	R/W	Host Controller Command Status Register	0x0000_0000
HcInterruptStatus	USBH_BA+0x00C	R/W	Host Controller Interrupt Status Register	0x0000_0000
HcInterruptEnable	USBH_BA+0x010	R/W	Host Controller Interrupt Enable Register	0x0000_0000
HcInterruptDisable	USBH_BA+0x014	R/W	Host Controller Interrupt Disable Register	0x0000_0000
HcHCCA	USBH_BA+0x018	R/W	Host Controller Communication Area Register	0x0000_0000
HcPeriodCurrentED	USBH_BA+0x01C	R	Host Controller Period Current ED Register	0x0000_0000
HcControlHeadED	USBH_BA+0x020	R/W	Host Controller Control Head ED Register	0x0000_0000
HcControlCurrentED	USBH_BA+0x024	R/W	Host Controller Control Current ED Register	0x0000_0000
HcBulkHeadED	USBH_BA+0x028	R/W	Host Controller Bulk Head ED Register	0x0000_0000
HcBulkCurrentED	USBH_BA+0x02C	R/W	Host Controller Bulk Current ED Register	0x0000_0000
HcDoneHead	USBH_BA+0x030	R/W	Host Controller Done Head Register	0x0000_0000
HcFmInterval	USBH_BA+0x034	R/W	Host Controller Frame Interval Register	0x0000_2EDF
HcFmRemaining	USBH_BA+0x038	R	Host Controller Frame Remaining Register	0xXXXX_XXXX
HcFmNumber	USBH_BA+0x03C	R	Host Controller Frame Number Register	0x0000_0000
HcPeriodicStart	USBH_BA+0x040	R/W	Host Controller Periodic Start Register	0x0000_0000
HcLSThreshold	USBH_BA+0x044	R/W	Host Controller Low-speed Threshold Register	0x0000_0628
HcRhDescriptorA	USBH_BA+0x048	R/W	Host Controller Root Hub Descriptor A Register	0x0100_0902
HcRhDescriptorB	USBH_BA+0x04C	R/W	Host Controller Root Hub Descriptor B Register	0x0000_0000
HcRhStatus	USBH_BA+0x050	R/W	Host Controller Root Hub Status Register	0x0000_0000
HcRhPortStatus1	USBH_BA+0x058	R/W	Host Controller Root Hub Port Status [1]	0x0000_0000
HcPhyControl	USBH_BA+0x200	R/W	Host Controller PHY Control Register	0x080F_0000
HcMiscControl	USBH_BA+0x204	R/W	Host Controller Miscellaneous Control Register	0x0000_0000

6.33.7 Register Description

Host Controller Revision Register (HcRevision)

Register	Offset	R/W	Description	Reset Value
HcRevision	USBH_BA+0x000	R	Host Controller Revision Register	0x0000_0110

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REV							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**REV**	**Revision Number** Indicates the Open HCI Specification revision number implemented by the Hardware. Host Controller supports 1.1 specification. (X.Y = XYh).

Host Controller Control Register (HcControl)

Register	Offset	R/W	Description	Reset Value
HcControl	USBH_BA+0x004	R/W	Host Controller Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HCFS		**BLE**	**CLE**	**IE**	**PLE**	**CBSR**	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:6]	**HCFS**	**Host Controller Functional State** This field sets the Host Controller state. The Controller may force a state change from USBSUSPEND to USBRESUME after detecting resume signaling from a downstream port. States are: 00 = USBRESET. 01 = USBRESUME. 10 = USBOPERATIONAL. 11 = USBSUSPEND.
[5]	**BLE**	**Bulk List Enable Bit** 0 = Processing of the Bulk list after next SOF (Start-Of-Frame) Disabled. 1 = Processing of the Bulk list in the next frame Enabled.
[4]	**CLE**	**Control List Enable Bit** 0 = Processing of the Control list after next SOF (Start-Of-Frame) Disabled. 1 = Processing of the Control list in the next frame Enabled.
[3]	**IE**	**Isochronous List Enable Bit** Both IE and PLE (HcControl[2]) high enables Host Controller to process the Isochronous list. Either IE or PLE (HcControl[2]) is low disables Host Controller to process the Isochronous list. 0 = Processing of the Isochronous list after next SOF (Start-Of-Frame) Disabled. 1 = Processing of the Isochronous list in the next frame Enabled, if the PLE (HcControl[2]) is high, too.
[2]	**PLE**	**Periodic List Enable Bit** When set, this bit enables processing of the Periodic (interrupt and isochronous) list. The Host Controller checks this bit prior to attempting any periodic transfers in a frame. 0 = Processing of the Periodic (Interrupt and Isochronous) list after next SOF (Start-Of-Frame) Disabled. 1 = Processing of the Periodic (Interrupt and Isochronous) list in the next frame Enabled. **Note:** To enable the processing of the Isochronous list, user has to set both PLE and IE (HcControl[3]) high.

[1:0]	**CBSR**	**Control Bulk Service Ratio** This specifies the service ratio between Control and Bulk EDs. Before processing any of the non-periodic lists, HC must compare the ratio specified with its internal count on how many nonempty Control EDs have been processed, in determining whether to continue serving another Control ED or switching to Bulk EDs. The internal count will be retained when crossing the frame boundary. In case of reset, HCD is responsible for restoring this value. 00 = Number of Control EDs over Bulk EDs served is 1:1. 01 = Number of Control EDs over Bulk EDs served is 2:1. 10 = Number of Control EDs over Bulk EDs served is 3:1. 11 = Number of Control EDs over Bulk EDs served is 4:1.

Host Controller Command Status Register (HcCommandStatus)

Register	Offset	R/W	Description	Reset Value
HcCommandStatus	USBH_BA+0x008	R/W	Host Controller Command Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**SOC**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**BLF**	**CLF**	**HCR**

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**SOC**	**Schedule Overrun Count (Read Only)** These bits are incremented on each scheduling overrun error. It is initialized to 00b and wraps around at 11b. This will be incremented when a scheduling overrun is detected even if SO (HcInterruptStatus[0]) has already been set.
[15:3]	**Reserved**	Reserved.
[2]	**BLF**	**Bulk List Filled** Set high to indicate there is an active TD on the Bulk list. This bit may be set by either software or the Host Controller and cleared by the Host Controller each time it begins processing the head of the Bulk list. 0 = No active TD found or Host Controller begins to process the head of the Bulk list. 1 = An active TD added or found on the Bulk list.
[1]	**CLF**	**Control List Filled** Set high to indicate there is an active TD on the Control List. It may be set by either software or the Host Controller and cleared by the Host Controller each time it begins processing the head of the Control List. 0 = No active TD found or Host Controller begins to process the head of the Control list. 1 = An active TD added or found on the Control list.
[0]	**HCR**	**Host Controller Reset** This bit is set to initiate the software reset of Host Controller. This bit is cleared by the Host Controller, upon completed of the reset operation. This bit, when set, didn't reset the Root Hub and no subsequent reset signaling be asserted to its downstream ports. 0 = Host Controller is not in software reset state. 1 = Host Controller is in software reset state.

Host Controller Interrupt Status Register (HcInterruptStatus)

Register	Offset	R/W	Description	Reset Value
HcInterruptStatus	USBH_BA+0x00C	R/W	Host Controller Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**RHSC**	**FNO**	**Reserved**	**RD**	**SF**	**WDH**	**SO**

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**RHSC**	**Root Hub Status Change** This bit is set when the content of HcRhStatus or the content of HcRhPortStatus1 register has changed. 0 = The content of HcRhStatus and the content of HcRhPortStatus1 register didn't change. 1 = The content of HcRhStatus or the content of HcRhPortStatus1 register has changed. **Note:** This bit is cleared by writing '1Fh' to HcRhPortStatus1[20:16].
[5]	**FNO**	**Frame Number Overflow** This bit is set when bit 15 of Frame Number changes from 1 to 0 or from 0 to 1. 0 = The bit 15 of Frame Number didn't change. 1 = The bit 15 of Frame Number changes from 1 to 0 or from 0 to 1. **Note:** This bit is cleared by writing 1 to it.
[4]	**Reserved**	Reserved.
[3]	**RD**	**Resume Detected** Set when Host Controller detects resume signaling on a downstream port. 0 = No resume signaling detected on a downstream port. 1 = Resume signaling detected on a downstream port. **Note:** This bit is cleared by writing 1 to it.
[2]	**SF**	**Start of Frame** Set when the Frame Management functional block signals a 'Start of Frame' event. Host Control generates a SOF token at the same time. 0 = Not the start of a frame. 1 = Indicate the start of a frame and Host Controller generates a SOF token. **Note:** This bit is cleared by writing 1 to it.

[1]	**WDH**	**Write Back Done Head** Set after the Host Controller has written HcDoneHead to HccaDoneHead. Further updates of the HccaDoneHead will not occur until this bit has been cleared. 0 = Host Controller didn't update HccaDoneHead. 1 = Host Controller has written HcDoneHead to HccaDoneHead. **Note:** This bit is cleared by writing 1 to it.
[0]	**SO**	**Scheduling Overrun** Set when the List Processor determines a Schedule Overrun has occurred. 0 = Schedule Overrun didn't occur. 1 = Schedule Overrun has occurred. **Note:** This bit is cleared by writing 1 to it.

Host Controller Interrupt Enable Register (HcInterruptEnable)

Register	Offset	R/W	Description	Reset Value
HcInterruptEnable	USBH_BA+0x010	R/W	Host Controller Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
MIE	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RHSC	FNO	Reserved	RD	SF	WDH	SO

Bits	Description	
[31]	**MIE**	**Master Interrupt Enable Bit** This bit is a global interrupt enable. A write of '1' allows interrupts to be enabled via the specific enable bits listed above. Write Operation: 0 = No effect. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) Enabled if the corresponding bit in HcInterruptEnable is high. Read Operation: 0 = Interrupt generation due to RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) Disabled even if the corresponding bit in HcInterruptEnable is high. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) Enabled if the corresponding bit in HcInterruptEnable is high.
[30:7]	**Reserved**	Reserved.
[6]	**RHSC**	**Root Hub Status Change Enable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]) Enabled. Read Operation: 0 = Interrupt generation due to RHSC (HcInterruptStatus[6]) Disabled. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]) Enabled.
[5]	**FNO**	**Frame Number Overflow Enable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to FNO (HcInterruptStatus[5]) Enabled. Read Operation: 0 = Interrupt generation due to FNO (HcInterruptStatus[5]) Disabled. 1 = Interrupt generation due to FNO (HcInterruptStatus[5]) Enabled.

[4]	**Reserved**	Reserved.
[3]	**RD**	**Resume Detected Enable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to RD (HcInterruptStatus[3]) Enabled. Read Operation: 0 = Interrupt generation due to RD (HcInterruptStatus[3]) Disabled. 1 = Interrupt generation due to RD (HcInterruptStatus[3]) Enabled.
[2]	**SF**	**Start of Frame Enable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to SF (HcInterruptStatus[2]) Enabled. Read Operation: 0 = Interrupt generation due to SF (HcInterruptStatus[2]) Disabled. 1 = Interrupt generation due to SF (HcInterruptStatus[2]) Enabled.
[1]	**WDH**	**Write Back Done Head Enable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to WDH (HcInterruptStatus[1]) Enabled. Read Operation: 0 = Interrupt generation due to WDH (HcInterruptStatus[1]) Disabled. 1 = Interrupt generation due to WDH (HcInterruptStatus[1]) Enabled.
[0]	**SO**	**Scheduling Overrun Enable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to SO (HcInterruptStatus[0]) Enabled. Read Operation: 0 = Interrupt generation due to SO (HcInterruptStatus[0]) Disabled. 1 = Interrupt generation due to SO (HcInterruptStatus[0]) Enabled.

Host Controller Interrupt Disable Register (HcInterruptDisable)

Register	Offset	R/W	Description	Reset Value
HcInterruptDisable	USBH_BA+0x014	R/W	Host Controller Interrupt Disable Register	0x0000_0000

31	30	29	28	27	26	25	24
MIE	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**RHSC**	**FNO**	**Reserved**	**RD**	**SF**	**WDH**	**SO**

Bits	Description	
[31]	**MIE**	**Master Interrupt Disable Bit** Global interrupt disable. Writing '1' to disable all interrupts. Write Operation: 0 = No effect. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) Disabled if the corresponding bit in HcInterruptEnable is high. Read Operation: 0 = Interrupt generation due to RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) Disabled even if the corresponding bit in HcInterruptEnable is high. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) Enabled if the corresponding bit in HcInterruptEnable is high.
[30:7]	**Reserved**	Reserved.
[6]	**RHSC**	**Root Hub Status Change Disable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]) Disabled. Read Operation: 0 = Interrupt generation due to RHSC (HcInterruptStatus[6]) Disabled. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]) Enabled.
[5]	**FNO**	**Frame Number Overflow Disable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to FNO (HcInterruptStatus[5]) Disabled. Read Operation: 0 = Interrupt generation due to FNO (HcInterruptStatus[5]) Disabled. 1 = Interrupt generation due to FNO (HcInterruptStatus[5]) Enabled.

[4]	**Reserved**	Reserved.
[3]	**RD**	**Resume Detected Disable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to RD (HcInterruptStatus[3]) Disabled. Read Operation: 0 = Interrupt generation due to RD (HcInterruptStatus[3]) Disabled. 1 = Interrupt generation due to RD (HcInterruptStatus[3]) Enabled.
[2]	**SF**	**Start of Frame Disable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to SF (HcInterruptStatus[2]) Disabled. Read Operation: 0 = Interrupt generation due to SF (HcInterruptStatus[2]) Disabled. 1 = Interrupt generation due to SF (HcInterruptStatus[2]) Enabled.
[1]	**WDH**	**Write Back Done Head Disable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to WDH (HcInterruptStatus[1]) Disabled. Read Operation: 0 = Interrupt generation due to WDH (HcInterruptStatus[1]) Disabled. 1 = Interrupt generation due to WDH (HcInterruptStatus[1]) Enabled.
[0]	**SO**	**Scheduling Overrun Disable Bit** Write Operation: 0 = No effect. 1 = Interrupt generation due to SO (HcInterruptStatus[0]) Disabled. Read Operation: 0 = Interrupt generation due to SO (HcInterruptStatus[0]) Disabled. 1 = Interrupt generation due to SO (HcInterruptStatus[0]) Enabled.

Host Controller Communication Area Register (HcHCCA)

Register	Offset	R/W	Description	Reset Value
HcHCCA	USBH_BA+0x018	R/W	Host Controller Communication Area Register	0x0000_0000

31	30	29	28	27	26	25	24
HCCA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HCCA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HCCA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:8]	**HCCA**	**Host Controller Communication Area** Pointer to indicate the base address of the Host Controller Communication Area (HCCA).
[7:0]	**Reserved**	Reserved.

Host Controller Period Current ED Register (HcPeriodCurrentED)

Register	Offset	R/W	Description	Reset Value
HcPeriodCurrentED	USBH_BA+0x01C	R	Host Controller Period Current ED Register	0x0000_0000

31	30	29	28	27	26	25	24
PCED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PCED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PCED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PCED				**Reserved**			

Bits	Description	
[31:4]	**PCED**	**Periodic Current ED** Pointer to indicate the physical address of the current Isochronous or Interrupt Endpoint Descriptor.
[3:0]	**Reserved**	Reserved.

Host Controller Control Head ED Register (HcControlHeadED)

Register	Offset	R/W	Description	Reset Value
HcControlHeadED	USBH_BA+0x020	R/W	Host Controller Control Head ED Register	0x0000_0000

31	30	29	28	27	26	25	24
CHED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CHED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CHED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHED				**Reserved**			

Bits	Description	
[31:4]	**CHED**	**Control Head ED** Pointer to indicate the physical address of the first Endpoint Descriptor of the Control list.
[3:0]	**Reserved**	Reserved.

Host Controller Control Current ED Register (HcControlCurrentED)

Register	Offset	R/W	Description	Reset Value
HcControlCurrentED	USBH_BA+0x024	R/W	Host Controller Control Current ED Register	0x0000_0000

31	30	29	28	27	26	25	24
CCED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CCED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CCED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCED				**Reserved**			

Bits	Description	
[31:4]	**CCED**	**Control Current Head ED** Pointer to indicate the physical address of the current Endpoint Descriptor of the Control list.
[3:0]	**Reserved**	Reserved.

Host Controller Bulk Head ED Register (HcBulkHeadED)

Register	Offset	R/W	Description	Reset Value
HcBulkHeadED	USBH_BA+0x028	R/W	Host Controller Bulk Head ED Register	0x0000_0000

31	30	29	28	27	26	25	24
BHED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BHED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BHED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BHED				**Reserved**			

Bits	Description	
[31:4]	**BHED**	**Bulk Head ED** Pointer to indicate the physical address of the first Endpoint Descriptor of the Bulk list.
[3:0]	**Reserved**	Reserved.

Host Controller Bulk Current ED Register (HcBulkCurrentED)

Register	Offset	R/W	Description	Reset Value
HcBulkCurrentED	USBH_BA+0x02C	R/W	Host Controller Bulk Current ED Register	0x0000_0000

31	30	29	28	27	26	25	24
BCED							
23	22	21	20	19	18	17	16
BCED							
15	14	13	12	11	10	9	8
BCED							
7	6	5	4	3	2	1	0
BCED				Reserved			

Bits	Description	
[31:4]	**BCED**	**Bulk Current Head ED** Pointer to indicate the physical address of the current Endpoint Descriptor of the Bulk list.
[3:0]	**Reserved**	Reserved.

Host Controller Done Head Register (HcDoneHead)

Register	Offset	R/W	Description	Reset Value
HcDoneHead	USBH_BA+0x030	R/W	Host Controller Done Head Register	0x0000_0000

31	30	29	28	27	26	25	24
DH							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DH							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DH							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DH				**Reserved**			

Bits	Description	
[31:4]	**DH**	**Done Head** Pointer to indicate the physical address of the last completed Transfer Descriptor that was added to the Done queue.
[3:0]	**Reserved**	Reserved.

Host Controller Frame Interval Register (HcFmInterval)

Register	Offset	R/W	Description	Reset Value
HcFmInterval	USBH_BA+0x034	R/W	Host Controller Frame Interval Register	0x0000_2EDF

31	30	29	28	27	26	25	24
FIT	**Reserved**	**FSMPS**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
FSMPS							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**FI**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FI							

Bits	Description	
[31]	**FIT**	**Frame Interval Toggle** This bit is toggled by Host Controller Driver when it loads a new value into FI (HcFmInterval[13:0]). 0 = Host Controller Driver didn't load new value into FI (HcFmInterval[13:0]). 1 = Host Controller Driver loads a new value into FI (HcFmInterval[13:0]).
[30]	**Reserved**	Reserved.
[29:16]	**FSMPS**	**FS Largest Data Packet** This field specifies a value that is loaded into the Largest Data Packet Counter at the beginning of each frame.
[15:14]	**Reserved**	Reserved.
[13:0]	**FI**	**Frame Interval** This field specifies the length of a frame as (bit times - 1). For 12,000 bit times in a frame, a value of 11,999 is stored here.

Host Controller Frame Remaining Register (HcFmRemaining)

Register	Offset	R/W	Description	Reset Value
HcFmRemaining	USBH_BA+0x038	R	Host Controller Frame Remaining Register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
FRT	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**FR**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FR							

Bits	Description	
[31]	**FRT**	**Frame Remaining Toggle** This bit is loaded from the FIT (HcFmInterval[31]) whenever FR (HcFmRemaining[13:0]) reaches 0.
[30:14]	**Reserved**	Reserved.
[13:0]	**FR**	**Frame Remaining** When the Host Controller is in the USBOPERATIONAL state, this 14-bit field decrements each 12 MHz clock period. When the count reaches 0, (end of frame) the counter reloads with Frame Interval. In addition, the counter loads when the Host Controller transitions into USBOPERATIONAL.

Host Controller Frame Number Register (HcFmNumber)

Register	Offset	R/W	Description	Reset Value
HcFmNumber	USBH_BA+0x03C	R	Host Controller Frame Number Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FN**	**Frame Number** This 16-bit incrementing counter field is incremented coincident with the re-load of FR (HcFmRemaining[13:0]). The count rolls over from ‘FFFFh’ to ‘0h.’

Host Controller Periodic Start Register (HcPeriodicStart)

Register	Offset	R/W	Description	Reset Value
HcPeriodicStart	USBH_BA+0x040	R/W	Host Controller Periodic Start Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**PS**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PS							

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:0]	**PS**	**Periodic Start** This field contains a value used by the List Processor to determine where in a frame the Periodic List processing must begin.

Host Controller Low-speed Threshold Register (HcLSThreshold)

Register	Offset	R/W	Description	Reset Value
HcLSThreshold	USBH_BA+0x044	R/W	Host Controller Low-speed Threshold Register	0x0000_0628

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**LST**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LST							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**LST**	**Low-speed Threshold** This field contains a value which is compared to the FR (HcFmRemaining[13:0]) field prior to initiating a Low-speed transaction. The transaction is started only if FR (HcFmRemaining[13:0]) >= this field. The value is calculated by Host Controller Driver with the consideration of transmission and setup overhead.

Host Controller Root Hub Descriptor A Register (HcRhDescriptorA)

Register	Offset	R/W	Description	Reset Value
HcRhDescriptorA	USBH_BA+0x048	R/W	Host Controller Root Hub Descriptor A Register	0x0100_0902

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**NOCP**	**OCPM**	**Reserved**		**PSM**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NDP							

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**NOCP**	**No Overcurrent Protection** This bit describes how the overcurrent status for the Root Hub ports reported. 0 = Overcurrent status is reported. 1 = Overcurrent status is not reported.
[11]	**OCPM**	**Overcurrent Protection Mode** This bit describes how the overcurrent status for the Root Hub ports reported. This bit is only valid when NOCP (HcRhDescriptorA[12]) is cleared. 0 = Global overcurrent. 1 = Individual overcurrent.
[10:9]	**Reserved**	Reserved.
[8]	**PSM**	**Power Switching Mode** This bit is used to specify how the power switching of the Root Hub ports is controlled. 0 = Global switching. 1 = Individual switching.
[7:0]	**NDP**	**Number Downstream Ports** USB host control supports two downstream ports and only one port is available in this series of chip. **Note:** NDP = 2 in this chip. However port 1 is reserved and only port 2 is available.

Host Controller Root Hub Descriptor B Register (HcRhDescriptorB)

Register	Offset	R/W	Description	Reset Value
HcRhDescriptorB	USBH_BA+0x04C	R/W	Host Controller Root Hub Descriptor B Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**PPCM**	**Reserved**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**PPCM**	**Port Power Control Mask** Global power switching. This field is only valid if Power Switching Mode is set (individual port switching). When set, the port only responds to individual port power switching commands (Set/Clear Port Power). When cleared, the port only responds to global power switching commands (Set/Clear Global Power). 0 = Port power controlled by global power switching. 1 = Port power controlled by port power switching.
[17:0]	**Reserved**	Reserved.

Host Controller Root Hub Status Register (HcRhStatus)

Register	Offset	R/W	Description	Reset Value
HcRhStatus	USBH_BA+0x050	R/W	Host Controller Root Hub Status Register	0x0000_0000

31	30	29	28	27	26	25	24
CRWE	Reserved						
23	22	21	20	19	18	17	16
Reserved						OCIC	LPSC
15	14	13	12	11	10	9	8
DRWE	Reserved						
7	6	5	4	3	2	1	0
Reserved						OCI	LPS

Bits	Description	
[31]	**CRWE**	**Clear Remote Wake-up Enable Bit** This bit is use to clear DRWE (HcRhStatus[15]). This bit is always read as 0. Write Operation: 0 = No effect. 1 = Clear DRWE (HcRhStatus[15]).
[30:18]	**Reserved**	Reserved.
[17]	**OCIC**	**Overcurrent Indicator Change** This bit is set by hardware when a change has occurred in OCI (HcRhStatus[1]). Write 1 to clear this bit to 0. 0 = OCI (HcRhStatus[1]) didn't change. 1 = OCI (HcRhStatus[1]) changed.
[16]	**LPSC**	**Set Global Power** In global power mode (PSM (HcRhDescriptorA[8]) = 0), this bit is written to one to enable power to all ports. This bit is always read as 0. Write Operation: 0 = No effect. 1 = Set global power.
[15]	**DRWE**	**Device Remote Wakeup Enable Bit** This bit controls if port's Connect Status Change as a remote wake-up event. Write Operation: 0 = No effect. 1 = Connect Status Change as a remote wake-up event Enabled. Read Operation: 0 = Connect Status Change as a remote wake-up event Disabled. 1 = Connect Status Change as a remote wake-up event Enabled.
[14:2]	**Reserved**	Reserved.

[1]	**OCI**	**Overcurrent Indicator (Read Only)** This bit reflects the state of the overcurrent status pin. This field is only valid if NOCP (HcRhDescriptorA[12]) and OCPM (HcRhDescriptorA[11]) are cleared. 0 = No overcurrent condition. 1 = Overcurrent condition.
[0]	**LPS**	**Clear Global Power** In global power mode (PSM (HcRhDescriptorA[8]) = 0), this bit is written to one to clear all ports' power. This bit is always read as 0. Write Operation: 0 = No effect. 1 = Clear global power.

Host Controller Root Hub Port Status [1] (HcRhPortStatus1)

Register	Offset	R/W	Description	Reset Value
HcRhPortStatus1	USBH_BA+0x058	R/W	Host Controller Root Hub Port Status [1]	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			PRSC	OCIC	PSSC	PESC	CSC
15	14	13	12	11	10	9	8
Reserved						LSDA	PPS
7	6	5	4	3	2	1	0
Reserved			PRS	POCI	PSS	PES	CCS

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**PRSC**	**Port Reset Status Change** This bit indicates that the port reset signal has completed. Write 1 to clear this bit to 0. 0 = Port reset is not complete. 1 = Port reset is complete.
[19]	**OCIC**	**Port Overcurrent Indicator Change** This bit is set when POCI (HcRhPortStatus1[3]) changes. Write 1 to clear this bit to 0. 0 = POCI (HcRhPortStatus1[3]) didn't change. 1 = POCI (HcRhPortStatus1[3]) changed.
[18]	**PSSC**	**Port Suspend Status Change** This bit indicates the completion of the selective resume sequence for the port. Write 1 to clear this bit to 0. 0 = Port resume is not complete. 1 = Port resume is complete.
[17]	**PESC**	**Port Enable Status Change** This bit indicates that the port has been disabled (PES (HcRhPortStatus1[1]) cleared) due to a hardware event. Write 1 to clear this bit to 0. 0 = PES (HcRhPortStatus1[1]) didn't change. 1 = PES (HcRhPortStatus1[1]) changed.
[16]	**CSC**	**Connect Status Change** This bit indicates connect or disconnect event has been detected (CCS (HcRhPortStatus1[0]) changed). Write 1 to clear this bit to 0. 0 = No connect/disconnect event (CCS (HcRhPortStatus1[0]) didn't change). 1 = Hardware detection of connect/disconnect event (CCS (HcRhPortStatus1[0]) changed).

[15:10]	**Reserved**	Reserved.
[9]	**LSDA**	**Low Speed Device Attached** This bit defines the speed (and bus idle) of the attached device. It is only valid when CCS (HcRhPortStatus1[0]) is set. This bit is also used to clear port power. Write Operation: 0 = No effect. 1 = Clear PPS (HcRhPortStatus1[8]). Read Operation: 0 = Full Speed device. 1 = Low-speed device.
[8]	**PPS**	**Port Power Status** This bit reflects the power state of the port regardless of the power switching mode. Write Operation: 0 = No effect. 1 = Port Power Enabled. Read Operation: 0 = Port power is Disabled. 1 = Port power is Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**PRS**	**Port Reset Status** This bit reflects the reset state of the port. Write Operation: 0 = No effect. 1 = Set port reset. Read Operation 0 = Port reset signal is not active. 1 = Port reset signal is active.
[3]	**POCI**	**Port Overcurrent Indicator** This bit reflects the state of the overcurrent status pin dedicated to this port. This field is only valid if NOCP (HcRhDescriptorA[12]) is cleared and OCPM (HcRhDescriptorA[11]) is set. This bit is also used to initiate the selective result sequence for the port. Write Operation: 0 = No effect. 1 = Clear port suspend. Read Operation: 0 = No overcurrent condition. 1 = Overcurrent condition.
[2]	**PSS**	**Port Suspend Status** This bit indicates the port is suspended Write Operation: 0 = No effect. 1 = Set port suspend. Read Operation: 0 = Port is not suspended. 1 = Port is selectively suspended.

[1]	**PES**	**Port Enable Status** Write Operation: 0 = No effect. 1 = Set port enable. Read Operation: 0 = Port Disabled. 1 = Port Enabled.
[0]	**CCS**	**Current Connect Status** Write Operation: 0 = No effect. 1 = Clear port enable. Read Operation: 0 = No device connected. 1 = Device connected.

Host Controller PHY Control Register (HcPhyControl)

Register	Offset	R/W	Description	Reset Value
HcPhyControl	USBH_BA+0x200	R/W	Host Controller PHY Control Register	0x080F_0000

31	30	29	28	27	26	25	24
Reserved				**STBYEN**	**Reserved**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**STBYEN**	**USB Transceiver Standby Enable Bit** This bit controls if USB transceiver could enter the standby mode to reduce power consumption. 0 = The USB transceiver would never enter the standby mode. 1 = The USB transceiver will enter standby mode while port is in power off state (port power is inactive).
[26:0]	**Reserved**	Reserved.

Host Controller Miscellaneous Control Register (HcMiscControl)

Register	Offset	R/W	Description	Reset Value
HcMiscControl	USBH_BA+0x204	R/W	Host Controller Miscellaneous Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**DPRT1**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**PPCAL**	**OCAL**	**Reserved**	**ABORT**	**Reserved**

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**DPRT1**	**Disable Port 1** This bit controls if the connection between USB host controller and transceiver of port 1 is disabled. If the connection is disabled, the USB host controller will not recognize any event of USB bus. Set this bit high, the transceiver of port 1 will also be forced into the standby mode no matter what USB host controller operation is. 0 = The connection between USB host controller and transceiver of port 1 Enabled. 1 = The connection between USB host controller and transceiver of port 1 Disabled and the transceiver of port 1 will also be forced into the standby mode.
[15:5]	**Reserved**	Reserved.
[4]	**PPCAL**	**Port Power Control Active Low** This bit controls the polarity of port power control to external power IC. 0 = Port power control is high active. 1 = Port power control is low active.
[3]	**OCAL**	**Overcurrent Active Low** This bit controls the polarity of overcurrent flag from external power IC. 0 = Overcurrent flag is high active. 1 = Overcurrent flag is low active.
[2]	**Reserved**	Reserved.
[1]	**ABORT**	**AHB Bus Error Response** This bit indicates there is an Error response received in AHB bus. 0 = No Error response received. 1 = Error response received. **Note:** This bit is cleared by writing 1 to it.
[0]	**Reserved**	Reserved.

6.34 USB On-The-Go (OTG)

6.34.1 Overview

The OTG controller interfaces to USB PHY and USB controllers which consist of a USB 1.1 host controller and a USB 2.0 FS device controller. The OTG controller supports HNP and SRP protocols defined in the “On-The-Go and Embedded Host Supplement to the USB 2.0 Revision 2.0 Specification”.

A USB frame, including USB host, USB device, and OTG controller, can be configured as Host-only, Device-only, ID dependent or OTG device mode defined in USBROLE (SYS_USBPHY[1:0]). In Host-only mode, the USB frame acts as USB host. The USB frame can support both full-speed and low-speed transfer. In Device-only mode, the USB frame acts as USB device. USB device support full-speed transfer. In ID dependent mode, the USB frame can be USB Host or USB device depending on USB_ID pin state. In OTG device mode, the role of USB frame depends on the definition of OTG specification. OTG specification support full-speed transfer whether OTG device acts as a host or peripheral.

6.34.2 Features

- Built-in USB PHY
- Configurable to operate as:
  - Host-only
  - Device-only
  - ID dependent: The role of USB frame is only dependent on USB_ID pin value--as USB Host (USB_ID pin is low) or USB Device (USB_ID pin is high). HNP or SRP protocol is not supported.
  - OTG device: dependent on USB_ID pin status to be A-device (USB_ID pin is low) or B-device (USB_ID pin is high). HNP and SRP protocols are supported.

6.34.3 Block Diagram

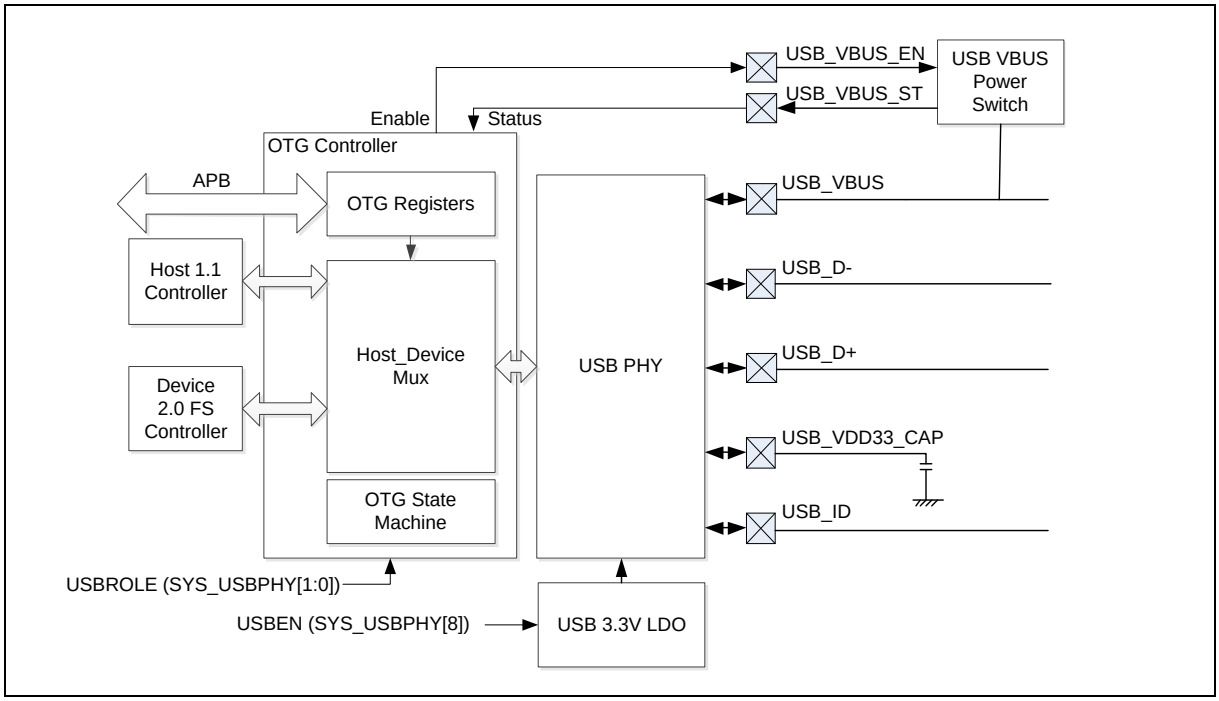


Figure 6.34-1 USB OTG Block Diagram

6.34.4 Basic Configuration

6.34.4.1 OTG Controller Basic Configuration

- Clock Source Configuration
  - Enable OTG peripheral clock on OTGCKEN (CLK_APBCLK0[26]).
- OTG Configuration
  - Refer description of SYS_REGLCTL register to disable the register protection function.
  - Setting role of USB frame is on USBROLE (SYS_USBPHY[1:0]).
  - Enable PHY on USBEN(SYS_USBPHY[8]) and wait 10 us for stable.
  - Write 1 clear all interrupt status(OTG_INTSTS)

6.34.5 Functional Description

The role of USB frame depends on the setting of USBROLE (SYS_USBPHY[1:0]) and USB_ID pin status. The USBROLE configuration has precedence over USB_ID pin status. User can configure the OTG controller to USB Host mode, USB Device mode, ID dependent mode or OTG device mode. In USB Host mode, the host controller will interact with USB PHY directly. In USB Device mode, the device controller will interact with USB PHY directly. In these cases, the OTG controller is used simply as a multiplexer. In ID dependent mode, USB_ID pin status will decide USB frame to act as USB host or USB device. If the USB_ID pin is FALSE state (low level), the USB frame will act as USB host. If the USB_ID pin is TRUE state (high level), the USB frame will act as USB device. In OTG device mode, the OTG controller will handle OTG HNP and SRP protocols. If the USB_ID pin is FALSE state (low level), the OTG controller will act as an OTG A-device. If the USB_ID pin is TRUE state (high level), the OTG controller will act as an OTG B-device.

6.34.5.1 The Role of USB Frame

USB Device Mode

When USBROLE (SYS_USBPHY[1:0]) is set to 0, the USB frame acts as USB device. The USB host function is not available.

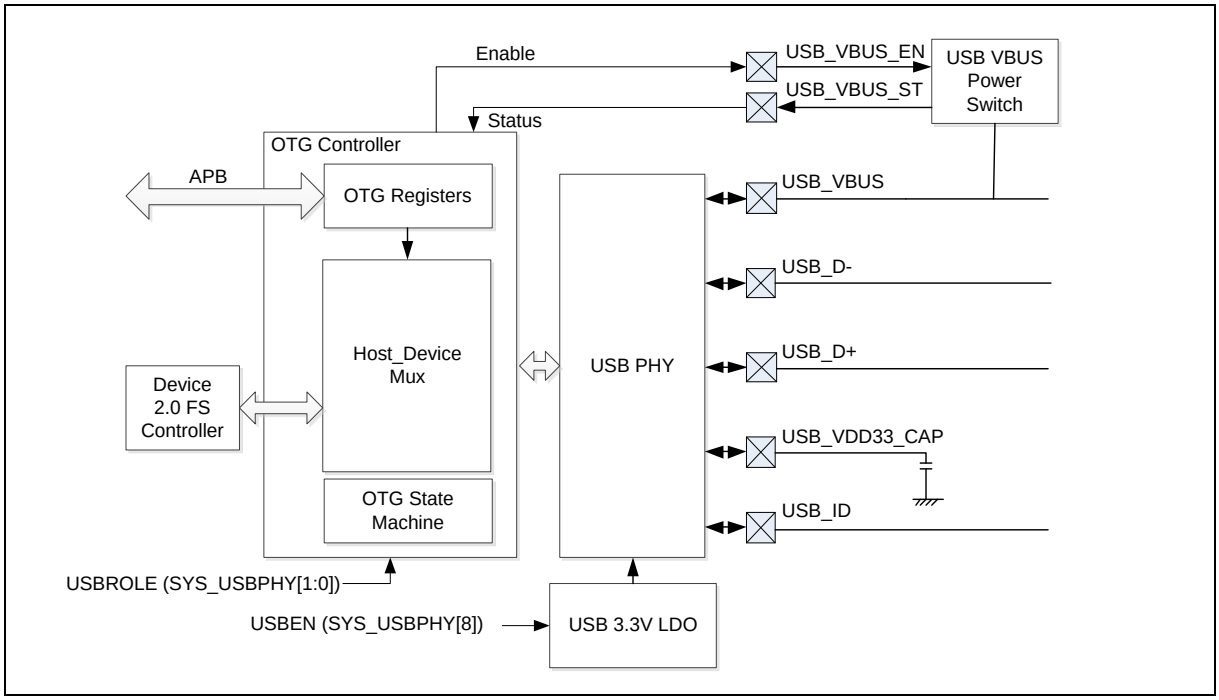


Figure 6.34-2 USB Device Mode

USB Host Mode

When USBROLE (SYS_USBPHY[1:0]) is set to 1, the USB frame acts as USB host. The USB device function is not available.

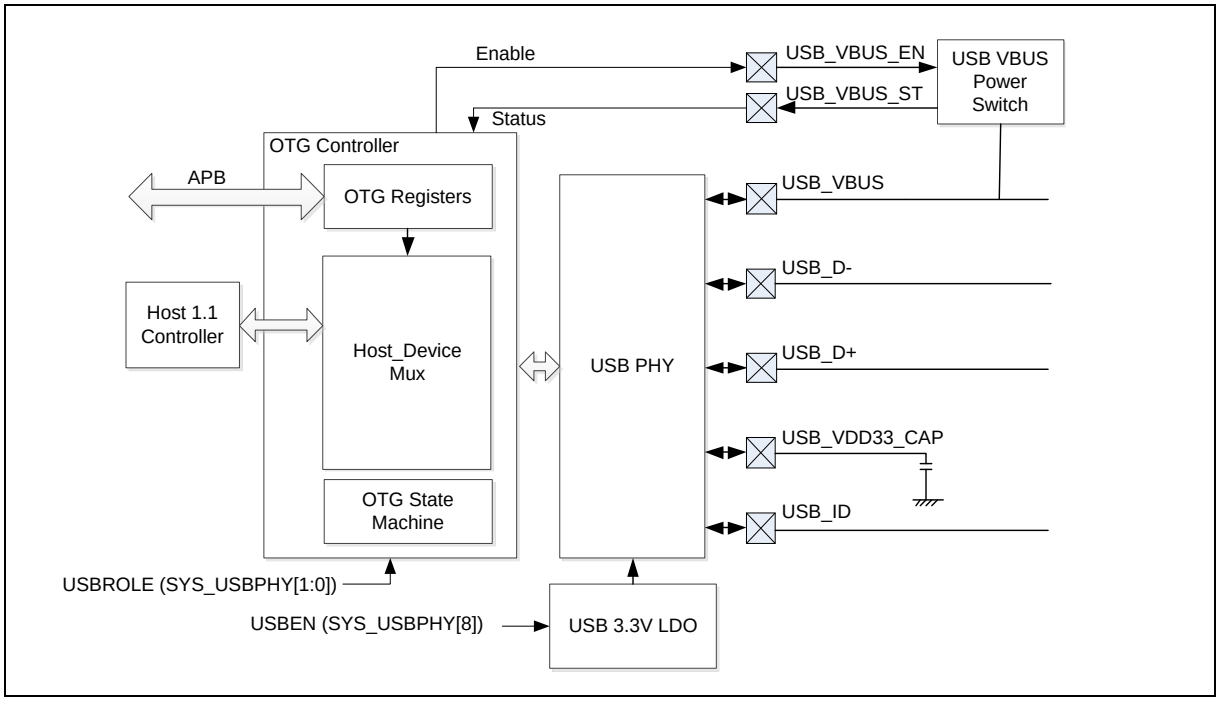


Figure 6.34-3 USB Host Mode

ID Dependent Mode

When USBROLE (SYS_USBPHY[1:0]) is set to 2, the role of USB frame depends on USB_ID pin status.

The ID detection function can be enabled by set IDDETEN (OTG_PHYCTL[1]) to 1. The USB_ID pin status reflects on IDSTS (OTG_STATUS[1]). When the USB frame acts as USB host (USB_ID pin is low level), the function is the same as USB Host mode. When the USB frame acts as USB device (USB_ID pin is high level), the function is the same as USB Device mode.

OTG Device Mode

When USBROLE (SYS_USBPHY[1:0]) is set to 3, the role of USB frame depends on USB_ID pin status. The ID detection function can be enabled by set IDDETEN (OTG_PHYCTL[1]) to 1. The USB_ID pin status reflects on IDSTS (OTG_STATUS[1]). When USB_ID pin status is low level, the OTG controller acts as OTG A-device. When USB_ID pin status is high level, the OTG controller acts as OTG B-device. Please refer to OTG specification to get detailed behavior of A-device and B-device.

6.34.5.2 Session Request Protocol (SRP)

When the USB frame is configured as OTG device mode, the OTG controller supports SRP to conserve power. Refer to OTG specification for details of SRP.

A-Device Session Request Protocol

1. A-device turns off USB bus power to conserve power. B-device recognizes such condition by checking V_{BUS} status.
2. B-device requests A-device to supply USB bus power by data line pulsing when B-device wants to connect to A-device.
3. A-device recognizes USB bus power request through checking SRPDETIF (OTG_INTSTS[13]).
4. A-device starts to drive V_{BUS} by setting BUSREQ (OTG_CTL[1]) to 1 once SRPDETIF (OTG_INTSTS[13]) is set to 1 by hardware. If V_{BUS} reaches valid level in specific time interval and B-device is connected, A-device will become USB host, HOSTIF (OTG_INTSTS[7]) will be set to 1. If V_{BUS} cannot reach valid level in specific time interval, it means overcurrent condition occurs. Then V_{BUS} error bit, VBEIF (OTG_INTSTS[1]), will be set to 1.

B-device Session Request Protocol

1. A-device turns off USB bus power to conserve power. B-device recognizes such condition by checking VBUSVLD (OTG_STATUS[5]).
2. B-device can request A-device to supply USB bus power by setting BUSREQ(OTG_CTL[1]) to 1.
3. B-device will generate data line pulsing as defined in OTG specification.
4. A-device will start to drive V_{BUS} after detecting data line pulsing and B-device can recognize such condition by checking VBUSVLD (OTG_STATUS[5]). If A-device drives V_{BUS} to valid level in specific time interval, B-device becomes USB peripheral, PDEVIF (OTG_INTSTS[6]) will be set to 1. If A-device does not drive V_{BUS} to valid level in specific time interval, SRP failure flag, SRPFIF (OTG_INTSTS[2]), will be set to 1 and B-device will go to idle state defined in OTG specification.

6.34.5.3 Host Negotiation Protocol (HNP)

When the USB frame is configured as OTG device mode, the host function can be transferred between two directly connected OTG device without changing the cable connection. Refer to OTG specification for details of HNP.

A-device Host Negotiation Protocol

1. A-Host defined in OTG specification sends SetFeature b_hnp_enable command to enable B-device HNP capability. B-device responses ACK to indicate B-device supports HNP. User needs to set HNPREQEN (OTG_CTL[2]) to 1 to enable HNP protocol.
2. A-Host goes to a_suspend state by setting BUSREQ (OTG_CTL[1]) to 0 and put USB bus into

J-state (USB_D+ high and USB_D- low) when A-Host has finished all desired operations.

3. A-Host becomes A-Peripheral if A-Host detects B-peripheral disconnected by checking USB_D+ and USB_D- low in specific time interval. If A-Host cannot detect B-Peripheral disconnected in specific time interval, A-Host will back to idle state.

B-device Host Negotiation Protocol

1. After B-Peripheral receives SetFeature b_hnp_enable command successfully, user enables B-peripheral HNP function by setting HNPREQEN (OTG_CTL[2]) to 1.
2. User sets BUSREQ (OTG_CTL[1]) to 1 after detecting USB bus in J-state(USB_D+ high and USB_D- low). Then USB_D+ pull high resistor will be removed to cause USB disconnect state (USB_D+ low and USB_D- low).
3. If B-device detects A-device is connected (USB_D+ high) in specific time interval, B-device will become B-Host. If B-device cannot detect A-device is connected (USB_D+ high) in specific time interval, HNP failure flag, HNPFIF (OTG_INTSTS[3]), will be set to 1.

6.34.6 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
OTG Base Address: **OTG_BA = 0x4004_D000**				
OTG_CTL	OTG_BA+0x00	R/W	OTG Control Register	0x0000_0000
OTG_PHYCTL	OTG_BA+0x04	R/W	OTG PHY Control Register	0x0000_0000
OTG_INTEN	OTG_BA+0x08	R/W	OTG Interrupt Enable Register	0x0000_0000
OTG_INTSTS	OTG_BA+0x0C	R/W	OTG Interrupt Status Register	0x0000_0XX0
OTG_STATUS	OTG_BA+0x10	R	OTG Status Register	0x0000_000X

6.34.7 Register Description

OTG Control Register (OTG_CTL)

Register	Offset	R/W	Description	Reset Value
OTG_CTL	OTG_BA+0x00	R/W	OTG Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**WKEN**	**OTGEN**	**Reserved**	**HNPREQEN**	**BUSREQ**	**VBUSDROP**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**WKEN**	**OTG ID Pin Wake-up Enable Bit** 0 = OTG ID pin status change wake-up function Disabled. 1 = OTG ID pin status change wake-up function Enabled.
[4]	**OTGEN**	**OTG Function Enable Bit** User needs to set this bit to enable OTG function while the USB frame configured as OTG device. When the USB frame is not configured as OTG device, this bit must be low. 0 = OTG function Disabled. 1 = OTG function Enabled.
[3]	**Reserved**	Reserved.
[2]	**HNPREQEN**	**OTG HNP Request Enable Bit** When the USB frame acts as A-device, set this bit when A-device allows to process HNP protocol—A-device changes role from Host to Peripheral. This bit will be cleared when OTG state changes from a_suspend to a_peripheral or goes back to a_idle state. When the USB frame acts as B-device, set this bit after the OTG A-device successfully sends a SetFeature (b_hnp_enable) command to the OTG B-device to start role change — B-device changes role from Peripheral to Host. This bit will be cleared when OTG state changes from b_peripheral to b_wait_acon or goes back to b_idle state. 0 = HNP request Disabled. 1 = HNP request Enabled (A-device can change role from Host to Peripheral or B-device can change role from Peripheral to Host). **Note:** Refer to OTG specification to get a_suspend, a_peripheral, a_idle and b_idle state.
[1]	**BUSREQ**	**OTG Bus Request** If OTG A-device wants to do data transfers via USB bus, setting this bit will drive V_{BUS} high to detect USB device connection. If user won't use the bus any more, clearing this bit will drop V_{BUS} to save power. This bit will be cleared when A-device goes to A_wait_vfall state. This bit will be also cleared if VBUSDROP (OTG_CTL[0]) bit is set or IDSTS (OTG_STATUS[1]) changed. If user of an OTG-B Device wants to request V_{BUS}, setting this bit will run SRP protocol.

Bits	Description	
		This bit will be cleared if SRP failure (OTG A-device does not provide V_{BUS} after B-device issues SRP in specified interval, defined in OTG specification). This bit will be also cleared if VBUSDROP (OTG_CTL[0]) bit is set or IDSTS (OTG_STATUS[1]) changed. 0 = Not launch V_{BUS} in OTG A-device or not request SRP in OTG B-device. 1 = Launch V_{BUS} in OTG A-device or request SRP in OTG B-device.
[0]	**VBUSDROP**	**Drop V_{BUS} Control** If user application running on this OTG A-device wants to conserve power, set this bit to drop V_{BUS}. BUSREQ (OTG_CTL[1]) will be also cleared no matter A-device or B-device. 0 = Not drop the V_{BUS}. 1 = Drop the V_{BUS}.

OTG PHY Control Register (OTG_PHYCTL)

Register	Offset	R/W	Description	Reset Value
OTG_PHYCTL	OTG_BA+0x04	R/W	OTG PHY Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		VBSTSPOL	VBENPOL	Reserved		IDDETEN	OTGPHYEN

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**VBSTSPOL**	**Off-chip USB V_{BUS} Power Switch Status Polarity** The polarity of off-chip USB V_{BUS} power switch valid signal depends on the selected component. A USB_VBUS_ST pin is used to monitor the valid signal of the off-chip USB V_{BUS} power switch. Set this bit as following according to the polarity of off-chip USB V_{BUS} power switch. 0 = The polarity of off-chip USB V_{BUS} power switch valid status is high. 1 = The polarity of off-chip USB V_{BUS} power switch valid status is low.
[4]	**VBENPOL**	**Off-chip USB V_{BUS} Power Switch Enable Polarity** The OTG controller will enable off-chip USB V_{BUS} power switch to provide V_{BUS} power when need. A USB_VBUS_EN pin is used to control the off-chip USB V_{BUS} power switch. The polarity of enabling off-chip USB V_{BUS} power switch (high active or low active) depends on the selected component. Set this bit as following according to the polarity of off-chip USB V_{BUS} power switch. 0 = The off-chip USB V_{BUS} power switch enable is active high. 1 = The off-chip USB V_{BUS} power switch enable is active low.
[3:2]	**Reserved**	Reserved.
[1]	**IDDETEN**	**ID Detection Enable Bit** 0 = Detect ID pin status Disabled. 1 = Detect ID pin status Enabled.
[0]	**OTGPHYEN**	**OTG PHY Enable Bit** When the USB frame is configured as either OTG device or ID dependent, user needs to set this bit before using OTG function. If device is configured as neither OTG device nor ID dependent, this bit is “don’t care”. 0 = OTG PHY Disabled. 1 = OTG PHY Enabled.

OTG Interrupt Enable Register (OTG_INTEN)

Register	Offset	R/W	Description	Reset Value
OTG_INTEN	OTG_BA+0x08	R/W	OTG Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SRPDETIEN	Reserved	SECHGIEN	VBCHGIEN	AVLDCHGIEN	BVLDCHGIEN
7	6	5	4	3	2	1	0
HOSTIEN	PDEVIEN	IDCHGIEN	GOIDLEIEN	HNPFIEN	SRPFIEN	VBEIEN	ROLECHGIEN

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**SRPDETIEN**	**SRP Detected Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[12]	**Reserved**	Reserved.
[11]	**SECHGIEN**	**SESSEND Status Changed Interrupt Enable Bit** If this bit is set to 1 and SESSEND (OTG_STATUS[2]) status is changed from high to low or from low to high, an interrupt will be asserted. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[10]	**VBCHGIEN**	**VBUSVLD Status Changed Interrupt Enable Bit** If this bit is set to 1 and VBUSVLD (OTG_STATUS[5]) status is changed from high to low or from low to high, an interrupt will be asserted. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[9]	**AVLDCHGIEN**	**A-device Session Valid Status Changed Interrupt Enable Bit** If this bit is set to 1 and AVLD (OTG_STATUS[4]) status is changed from high to low or from low to high, an interrupt will be asserted. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[8]	**BVLDCHGIEN**	**B-device Session Valid Status Changed Interrupt Enable Bit** If this bit is set to 1 and BVLD (OTG_STATUS[3]) status is changed from high to low or from low to high, an interrupt will be asserted. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[7]	**HOSTIEN**	**Act As Host Interrupt Enable Bit** If this bit is set to 1 and the device is changed as a host, an interrupt will be asserted.

Bits	Description	
		0 = This device as a host interrupt Disabled. 1 = This device as a host interrupt Enabled.
[6]	**PDEVIEN**	**Act As Peripheral Interrupt Enable Bit** If this bit is set to 1 and the device is changed as a peripheral, an interrupt will be asserted. 0 = This device as a peripheral interrupt Disabled. 1 = This device as a peripheral interrupt Enabled.
[5]	**IDCHGIEN**	**IDSTS Changed Interrupt Enable Bit** If this bit is set to 1 and IDSTS (OTG_STATUS[1]) status is changed from high to low or from low to high, an interrupt will be asserted. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[4]	**GOIDLEIEN**	**OTG Device Going to IDLE State Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** Going to idle state means going to a_idle or b_idle state. Please refer to A-device state diagram and B-device state diagram in OTG specification.
[3]	**HNPFIEN**	**HNP Fail Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[2]	**SRPFIEN**	**SRP Fail Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[1]	**VBEIEN**	**V_{BUS} Error Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** V_{BUS} error means going to a_vbus_err state. Please refer to A-device state diagram in OTG specification.
[0]	**ROLECHGIEN**	**Role Changed Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note: Role is Host or Peripheral.**

OTG Interrupt Status Register (OTG_INTSTS)

Register	Offset	R/W	Description	Reset Value
OTG_INTSTS	OTG_BA+0x0C	R/W	OTG Interrupt Status Register	0x0000_0XX0

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SRPDETIF	Reserved	SECHGIF	VBCHGIF	AVLDCHGIF	BVLDCHGIF
7	6	5	4	3	2	1	0
HOSTIF	PDEVIF	IDCHGIF	GOIDLEIF	HNPFIF	SRPFIF	VBEIF	ROLECHGIF

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**SRPDETIF**	**SRP Detected Interrupt Status** 0 = SRP not detected. 1 = SRP detected. **Note:** Write 1 to clear this status.
[12]	**Reserved**	**Note:**
[11]	**SECHGIF**	**SESSEND State Change Interrupt Status** 0 = SESSEND (OTG_STATUS[2]) not toggled. 1 = SESSEND (OTG_STATUS[2]) from high to low or from low to high. **Note:** Write 1 to clear this flag.
[10]	**VBCHGIF**	**VBUSVLD State Change Interrupt Status** 0 = VBUSVLD (OTG_STATUS[5]) not toggled. 1 = VBUSVLD (OTG_STATUS[5]) from high to low or from low to high. **Note:** Write 1 to clear this status.
[9]	**AVLDCHGIF**	**A-device Session Valid State Change Interrupt Status** 0 = AVLD (OTG_STATUS[4]) not toggled. 1 = AVLD (OTG_STATUS[4]) from high to low or low to high. **Note:** Write 1 to clear this status.
[8]	**BVLDCHGIF**	**B-device Session Valid State Change Interrupt Status** 0 = BVLD (OTG_STATUS[3]) not toggled. 1 = BVLD (OTG_STATUS[3]) from high to low or low to high. **Note:** Write 1 to clear this status.
[7]	**HOSTIF**	**Act As Host Interrupt Status** 0= This device does not act as a host. 1 = This device acts as a host. **Note:** Write 1 to clear this flag.

Bits	Description	
[6]	**PDEVIF**	**Act As Peripheral Interrupt Status** 0 = This device does not act as a peripheral. 1 = This device acts as a peripheral. **Note:** Write 1 to clear this flag.
[5]	**IDCHGIF**	**ID State Change Interrupt Status** 0 = IDSTS (OTG_STATUS[1]) not toggled. 1 = IDSTS (OTG_STATUS[1]) from high to low or from low to high. **Note:** Write 1 to clear this flag.
[4]	**GOIDLEIF**	**OTG Device Goes to IDLE Interrupt Status** Flag is set if the OTG device transfers from non-idle state to idle state. The OTG device will be neither a host nor a peripheral. 0 = OTG device does not go back to idle state (a_idle or b_idle). 1 = OTG device goes back to idle state(a_idle or b_idle). **Note 1:** Going to idle state means going to a_idle or b_idle state. Please refer to OTG specification. **Note 2:** Write 1 to clear this flag.
[3]	**HNPFIF**	**HNP Fail Interrupt Status** When A-device has granted B-device to be host and USB bus is in SE0 (both USB_D+ and USB_D- low) state, this bit will be set when A-device does not connect after specified interval expires. 0 = A-device connects to B-device before specified interval expires. 1 = A-device does not connect to B-device before specified interval expires. **Note:** Write 1 to clear this flag.
[2]	**SRPFIF**	**SRP Fail Interrupt Status** After initiating SRP, an OTG B-device will wait for the OTG A-device to drive V_{BUS} high at least TB_SRP_FAIL minimum, defined in OTG specification. This flag is set when the OTG B-device does not get V_{BUS} high after this interval. 0 = OTG B-device gets V_{BUS} high before this interval. 1 = OTG B-device does not get V_{BUS} high before this interval. **Note:** Write 1 to clear this flag.
[1]	**VBEIF**	**V_{BUS} Error Interrupt Status** This bit will be set when voltage on V_{BUS} cannot reach a minimum valid threshold 4.4V within a maximum time of 100ms after OTG A-device starting to drive V_{BUS} high. 0 = OTG A-device drives V_{BUS} over threshold voltage before this interval expires. 1 = OTG A-device cannot drive V_{BUS} over threshold voltage before this interval expires. **Note:** Write 1 to clear this flag and recover from the V_{BUS} error state.
[0]	**ROLECHGIF**	**OTG Role Change Interrupt Status** This flag is set when the role of an OTG device changed from a host to a peripheral, or changed from a peripheral to a host while USB_ID pin status does not change. 0 = OTG device role not changed. 1 = OTG device role changed. **Note:** Write 1 to clear this flag.

OTG Functional Status Register (OTG_STATUS)

Register	Offset	R/W	Description	Reset Value
OTG_STATUS	OTG_BA+0x10	R	OTG Status Register	0x0000_000X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ASHOST	ASPERI	VBUSVLD	AVLD	BVLD	SESSEND	IDSTS	OVERCUR

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**ASHOST**	**As Host Status** When OTG acts as Host, this bit is set. 0 = OTG not as Host. 1 = OTG as Host.
[6]	**ASPERI**	**As Peripheral Status** When OTG acts as peripheral, this bit is set. 0 = OTG not as peripheral. 1 = OTG as peripheral.
[5]	**VBUSVLD**	**V_{BUS} Valid Status** When V_{BUS} is larger than 4.7V, this bit will be set to 1. 0 = V_{BUS} is not valid. 1 = V_{BUS} is valid.
[4]	**AVLD**	**A-Device Session Valid Status** 0 = A-device session is not valid. 1 = A-device session is valid.
[3]	**BVLD**	**B-device Session Valid Status** 0 = B-device session is not valid. 1 = B-device session is valid.
[2]	**SESSEND**	**Session End Status** When V_{BUS} voltage is lower than 0.4V, this bit will be set to 1. Session end means no meaningful power on V_{BUS}. 0 = Session is not end. 1 = Session is end.
[1]	**IDSTS**	**USB_ID Pin State of Mini-/Micro-Plug** 0 = Mini-A/Micro-A plug is attached.

Bits	Description	
		1 = Mini-B/Micro-B plug is attached.
[0]	**OVERCUR**	**Overcurrent Condition** The voltage on V_{BUS} cannot reach a minimum V_{BUS} valid threshold, 4.4V minimum, within a maximum time of 100ms after OTG A-device drives V_{BUS} high. 0 = OTG A-device drives V_{BUS} successfully. 1 = OTG A-device cannot drives V_{BUS} high in this interval.

6.35 USB Type C Power Delivery Controller (UTCPD)

6.35.1 Overview

The UTCPD is a USB Type-C Power Delivery controller. UTCPD integrates a Power Delivery PHY with BMC encoding and decoding, protocol layer function and USB Type-C Configuration Channel (CC) logic to fulfill USB Type-C CC sensing and control and USB PD message delivery function.

UTCPD is compliant with USB Type-C 1.2 and USB Power Delivery3.0.

6.35.2 Features

- Basic Functionality
  - Compliant with USB Type-C Cable and Connector Specification Rev1.2 (TYPE-C)
  - Supports UFP, DFP and DRD data roles
  - Supports Sink, Source and DRP power roles
  - Dual-Role Power functionality with Autonomous DRP toggle
  - Supports CC1 and CC2 signals
    - Internal Rp and Rd
    - Default, 1.5A and 3A current sourcing advertising
    - Support dead battery
    - Detects cable orientation
    - Transmits and receives BMC encoded, PD messages
    - Transmits and receives Fast Role Swap (FRS) requests
    - Detects V_{BUS} presence
  - Monitors V_{BUS} voltage,using system EADC
  - Debounces attachment detection on either V_{BUS} or CCx signals
  - Detects connection removal via either V_{BUS} or CC*x* signals
- Power Delivery Functionality
  - Compliant with USB Power Delivery Specification Rev3.0, V1.2 (PD)
  - Transmits and receives serial data via CC*x* signals:
    - Limits the slew rate of the transmitted signal
    - Filters the received signal
  - Supports Power Delivery PHY Layer
    - Performs BMC and 4b5b encoding/decoding of the transmitted and received data bits stream
    - Generates and decodes Preamble, SOP and EOP
    - Appends and checks CRC
    - Generates and receives Hard Reset
    - Generates and receives Cable Reset
    - Supports Carrier Mode 2 and Test data, BIST modes
  - Supports Power Delivery Protocol Layer

- Supports SOP', SOP" and SOP messages
- Implements and checks the MessageID counter
- Supports Control messages ：Automatically generates GoodCRC response
- Supports Control messages ：Generates Hard Reset
- Supports Data messages with up to 30 data bytes：Supports Vendor Defined Messages (VDM)
- Supports Data messages with up to 30 data bytes：Supports long messages via Chunking
- Has separate transmit and receive buffers
- Supports up to three automatic retries
- Supports Data role, power role and V_{CONN} swapping
- Implements the timing critical functions of the PD protocol, such as waiting for a GoodCRC response

- System Control and Monitoring
  - Provides controls for V_{BUS} both "force" and "bleed" bulk capacitors discharge
    - Discharge signal is controlled by monitoring the actual VBus voltage level
- Wake-up from Idle mode and NPD0~2

6.35.3 Block Diagram

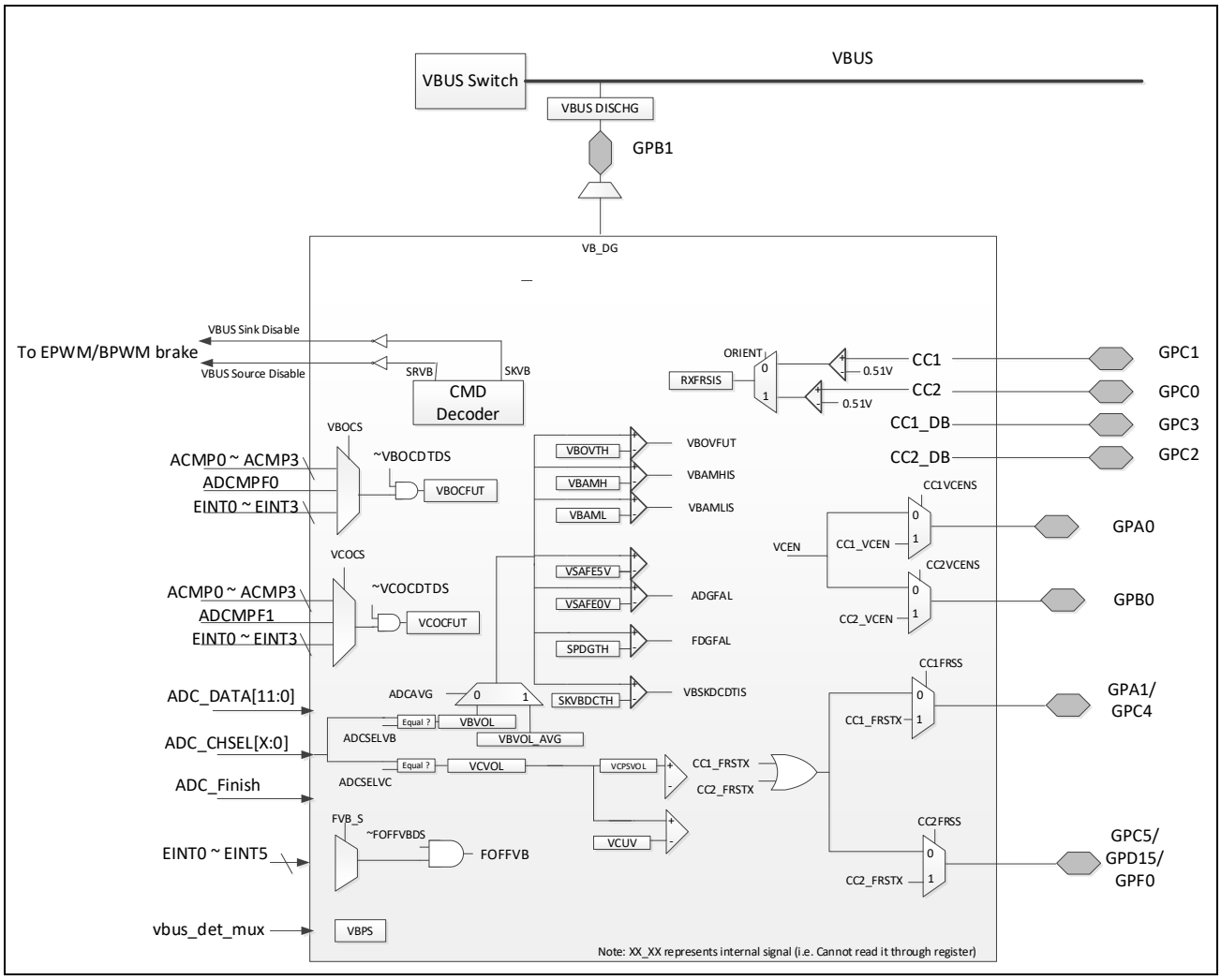


Figure 6.35-1 UTCPD System Block Diagram

6.35.4 Basic Configuration

- Clock Source Configuration
  - Enable UTCPD clock in UTCPD0CKEN (CLK_UTCPDCTL[0]).
  - Enable HIRC in HIRCEN (CLK_SRCCTL[3]).
  - Enable LIRC in LIRCEN (CLK_SRCCTL[0]).
- Reset Configuration
  - Reset UTCPD in UTCPDxRST (SYS_UTCPDRST[0]).

6.35.5 Functional Description

The three main functionalities of the UTCPD are: USB Type-C™ Port Power Control for V_{BUS} and VCONN, CC sensing and control, and USB PD Message delivery.

It implements the portion of the Physical layer, as defined in *USB Power Delivery Specification, Revision 3.0,V 1.2* (**PD**), sends and receives messages for Power Delivery negotiation, Role swapping, standard and Fast Power swapping, Alternate mode and VDM exchanges.

In addition, the UTCPD implements logic for controlling the CC pins on the USB Type-C™ Connector, as defined in *USB Type-C™ Cable and Connector Specification, Revision 1.3* (**TYPE-C**), as follows: detection and advertisement charging profiles, port connection and orientation detection, role detection and configuration and Dead Battery condition handling.

6.35.5.1 Abbreviation

Description	Abbreviation
Discharge	DG
Threshold	TH
Disconnect	DC
Detect	DT
Source	SR
Sink	SK
V_{BUS}	VB
VCONN	VC
Role	ROL
Voltage	VOL
Polarity	PL
Disable	DS
Present	PS
Cable	CAB
Fail	FAL
Change	CH
Biphase Mark Coding	BMC
Local Policy Manager	LPM
Local Policy Manager Interface	LPMI
Operating System Policy Manager	OPM
Platform Policy Manager	PPM
Platform Policy Manager Interface	PPMI
USB Type-C Port Conttroller	TCPC
USB Type-C Port Conttroller Interface	TCPCI
USB Type-C Port Manager	TCPCM
Sink CC pin above minimum vRd-Connect, per USB Type-C	SNK.Rp
Sink CC pin below maximum vRa, per USB Type-C	SNK.Open
Source CC pin above vOPEN, per USB Type-C	SRC.Ra
Source CC pin within vRd range, per USB Type-C	SRC.Rd
Source CC pin below maximum vRa, per USB Type-C	SRC.Open
Overcurrent Protection	OCP

Over-voltage Protection	OVP
Safe operating voltage at “zero volts” per USB PD	vSafe0V
Safe operating voltage at 5V per USB PD	vSafe5v

Table 6.35-1 Abbreviation

6.35.5.2 V_{BUS} detection:

In a self-powered system, it is generally desirable to be able to detect the presence of V_{BUS}. V_{BUS} indicates when a host device has been connected to or disconnected from the USB peripheral. For detailed information, please refer to **Universal Serial Bus Power Delivery Specification 3.1**. In the PD specification, which describes the reference parameters and response time of the V_{BUS} voltage threshold in detail.

Figure 6.35-2 shows the V_{BUS} detection block diagram. The external divided circuit setting (divided by 10 or 20) must reflect on VBSCALE (UTCPD_VBVOL[11:10]). User shall refer to the system V_{BUS} present voltage.

Note: The external divided circuit setting (divided by 10 or 20) must reflect on VBSCALE (UTCPD_VBVOL[11:10])

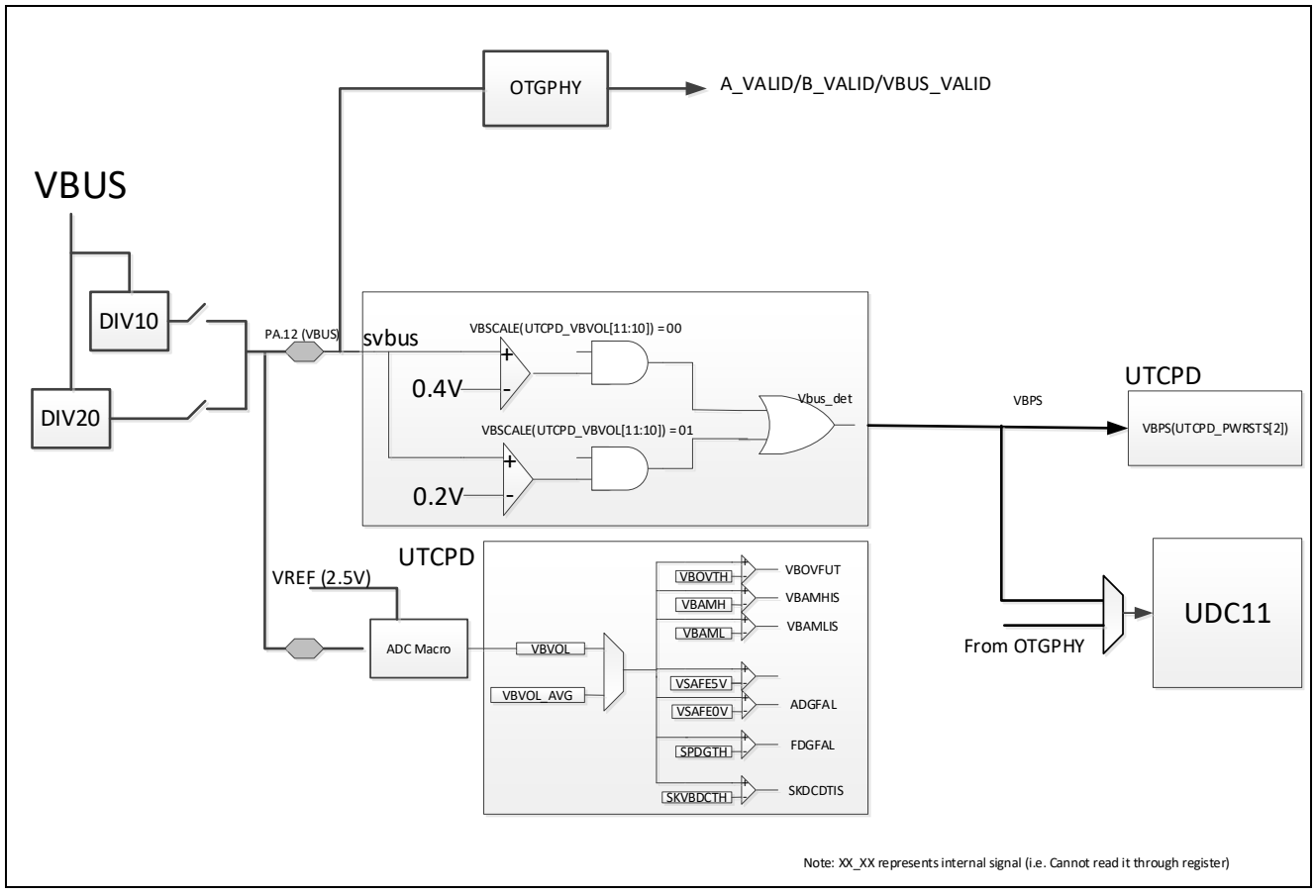


Figure 6.35-2 V_{BUS} Detection Block Diagram (Implement in design)

6.35.5.3 UTCPD as Sink

The default of V_{BUS} is 5V and default V_{BUS} voltage scaling is 10. It means PA.12 is 0.5V. Internal comparator will use 0.4V as a reference to monitor PA.12 voltage. Once PA.12 is higher than 0.4V, VBPS (UTCPD_PWRSTS[2]) will be high. After explicit power contract is established, V_{BUS} may go up

higher than 5V. If application uses V_{BUS} which is lower than 25V, the V_{BUS} voltage scale 10 should be selected. Taking V_{BUS} as 20V as example, PA.12 is 2V in this case. User needs to set relevant registers, such as VBOVTH, VBMAH, VBAML, VSAFE5V, VSAFE0V, SPDGTH, and SKVBDCTH to proper value to monitor V_{BUS} voltage. If the V_{BUS} greater than 25V is used, e.g.40V, the V_{BUS} voltage scale 20 should be selected.

6.35.5.4 Resets

Resets are necessary response to protocol or other error conditions. USB Power Delivery defines four different types of reset:

- Soft Reset, which resets protocol
- Data Reset which resets the USB communications
- Hard Reset which resets both the power supplies and protocol
- Cable Reset which resets the cable

A Soft Reset Message is used to cause a Soft Reset of protocol communication when this has broken down in some way. It Shall Not have any impact on power supper supply operation but is used to correct a Protocol Error occurring during an Atomic Message Sequence (AMS, for detail, please refer to USB Power Delivery specification). The Soft Reset May be triggered by either Port Partner in response to the Protocol Error which is any uinexpected Message during an AMS.

A Data Reset Message is used by a Port to reset its USB data connection and to exit all Alternate Modes both with its Port Partner and in the Cable Plug(s). The Data Reset process May be initiated by either Port Partner sending a Data_Reset Message.

Hard Resets are signaled by an ordered set as defined as thres RST-1 K-codes followed by one RST-2 K-code (detail information refer to USB Power Delievery specification).

Both the sender and recipient Shall cause their power supplies to return to their default states. In addition, their respective Protocol Layers Shall be reset as for the Soft Reset. This allows the Attached devices to be in a state where they can re-establish USB PD communication. **Note:** that even though VBus drops to vSafe0V during a Hard Reset a Sink will not see this as a disconnect since this is expected behavior.

Cable Reset Signaling is an ordered set of bytes sent with the purpose to be recognized by the PHY Layer. The Cable Reset Signaling ordered set is defined as the following sequence of K-codes: RST-1, Sync-1, RST-1, Sync-3.

Table 6.35-2 lists the reset conditions that affect the UTCPD_DTRXEVNT register which implies the UTCPD operation conditions.

Condition	**Hardware auto clear RECEIVE_DETECT (UTCPD_DTRXEVNT)**
Receive Hard Reset	Yes
Transmit Hard Reset	Yes
Receive Cable Reset	Yes
Transmit Cable Reset	No
Disconnect detected	Yes
Command RxOneMore and GoodCRC Transmitted	Yes
Others	No

Table 6.35-2 Reset Conditions

6.35.5.5 Interrupts

All conditions on the UTCPD bus will be monitored all the time, which includes the status of CC pin, the power status of V_{BUS} and Vcon, the response of system error events and vendor defined. All conditions will be generated interrupt if the relatively enable bit is set. Figure 6.35-3 shows all of the interrupt conditions for UTCPD. For the convenience of classification, CCSCHIS (UTCPD_IS[0]), PWRSCHIS (UTCPD_IS[1]), FUTIS(UTCPD_IS[9]) and VNDIS(UTCPD_IS[15]) are shown in Figure 6.35-4 to Figure 6.35-7, respectively.

Figure 6.35-4 shows the CC interrupt status in register UTCPD_CCSTS including CC1 state, CC2 state, connect result and look for connection. Figure 6.35-5 shows the power change status in regsiter UTCPD_PWRSTS. Figure 6.35-6 shows the fault status in register UTCPD_FUTSTS. Figure 6.35-7 shows the vendor status in register UTCPD_VNDIS.

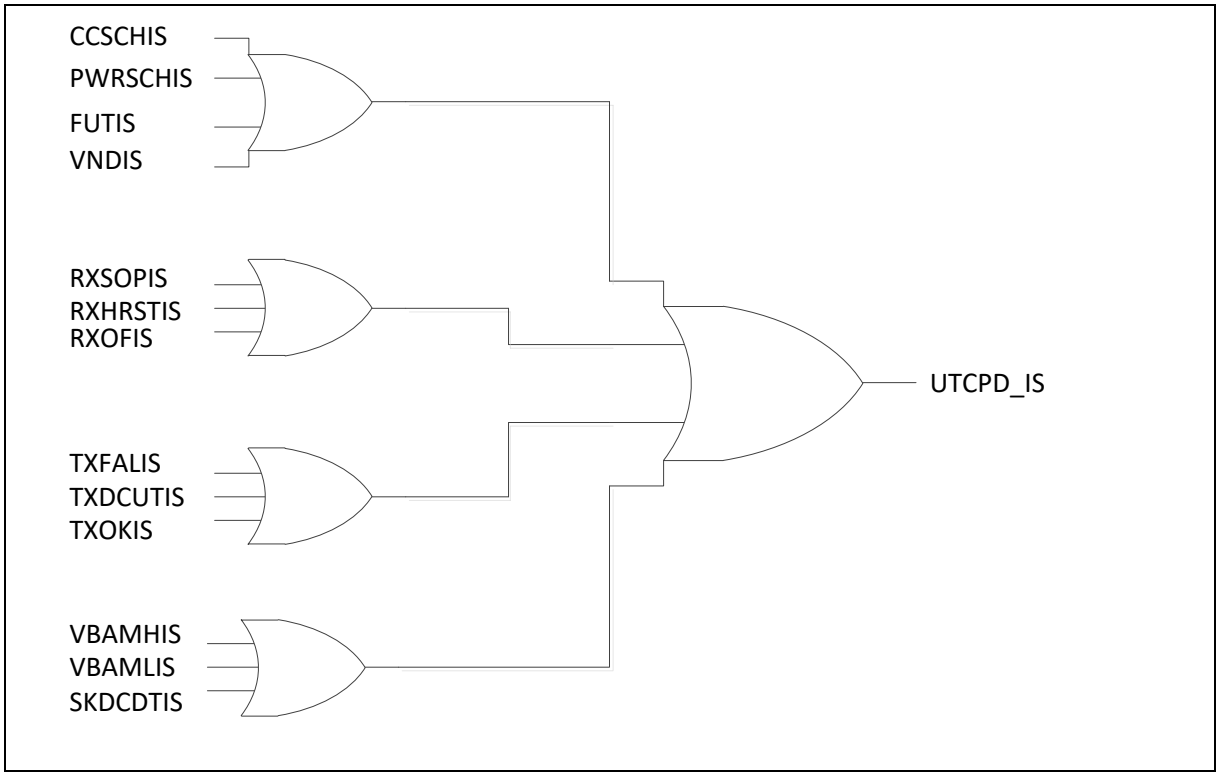


Figure 6.35-3 UTCPD Interrupt

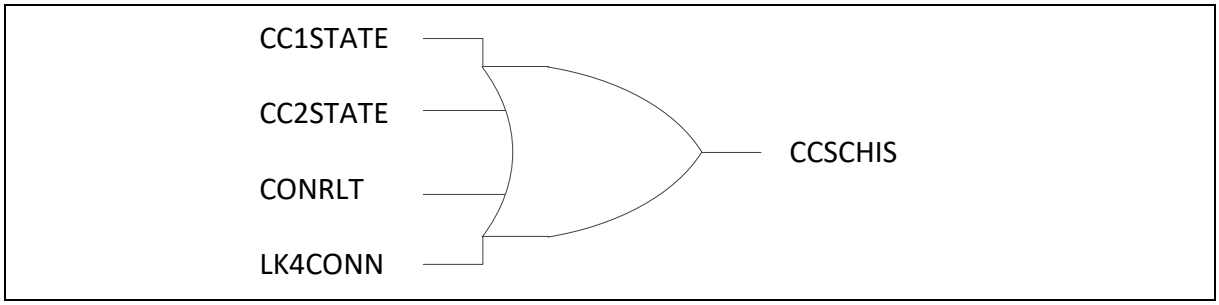


Figure 6.35-4 CC State Change Interrupt

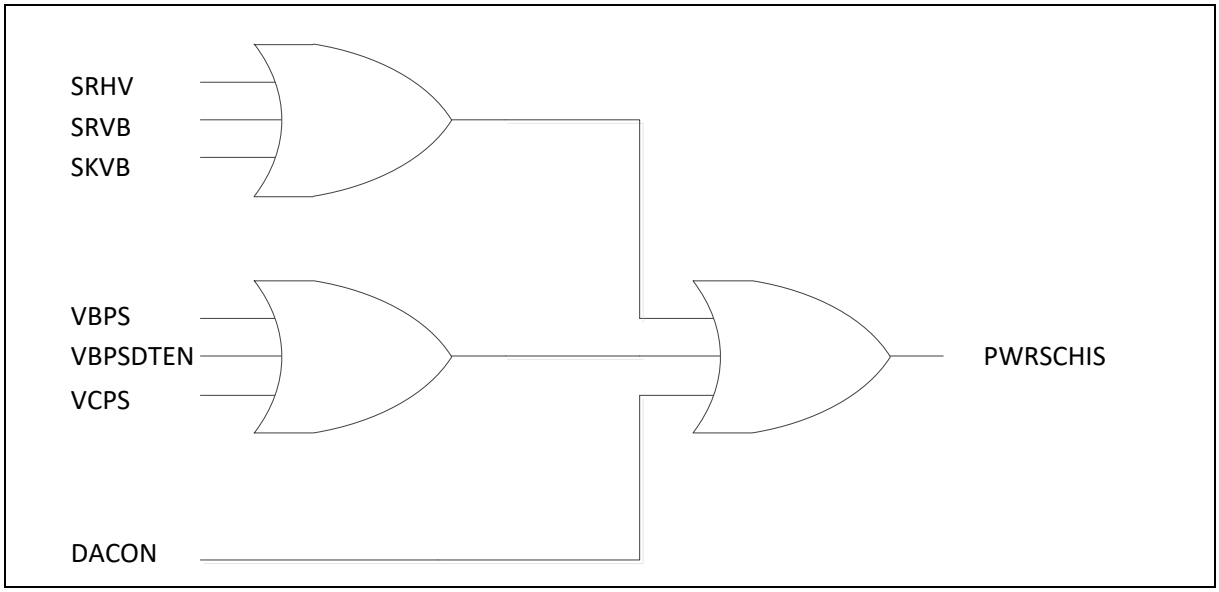


Figure 6.35-5 Power Status Change Interrupt

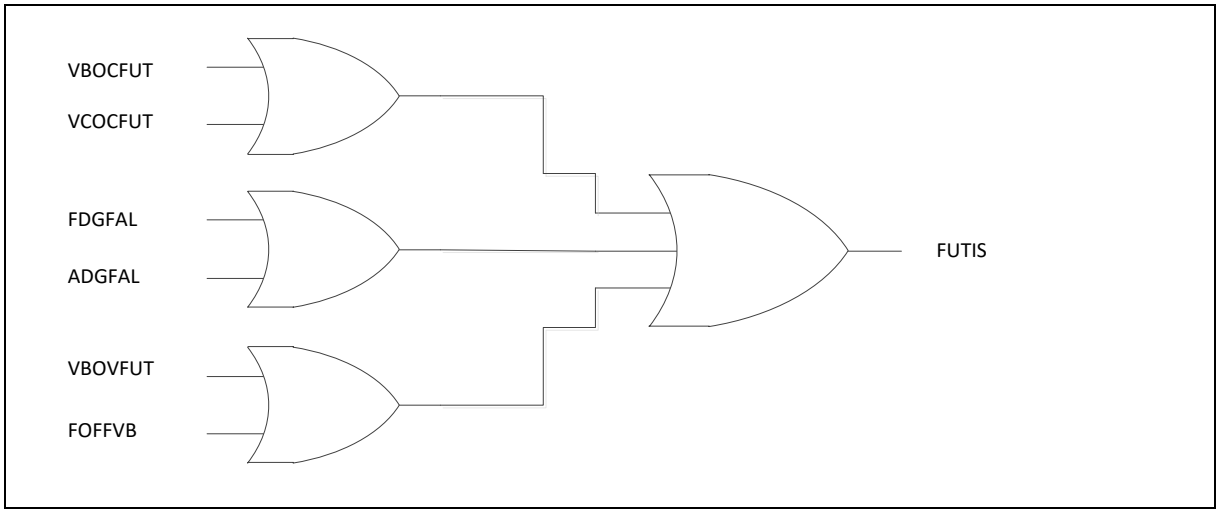


Figure 6.35-6 Fault Status Interrupt

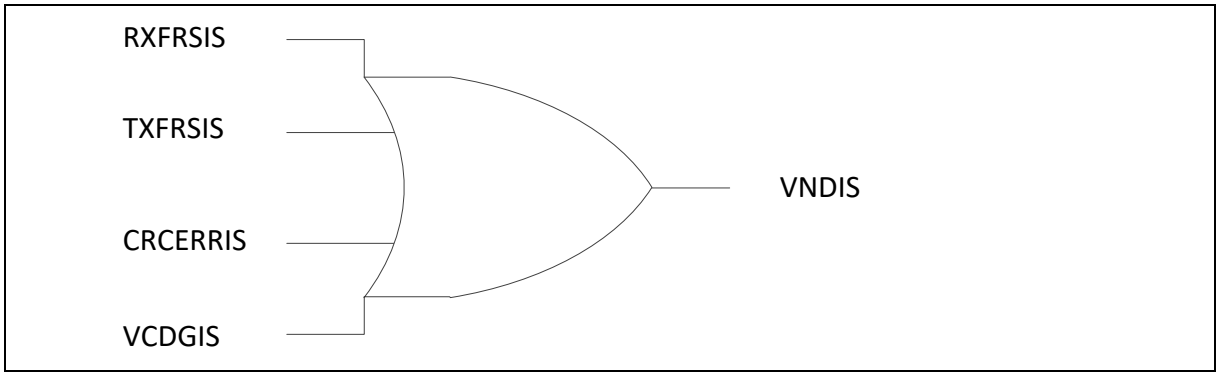


Figure 6.35-7 Vendor Status Interrupt

6.35.5.6 Disconnection detect

As Sink

A UTCPD shall use either the VBPS (UTCPD_PWRSTS[2]) transition from 1 to 0, or V_{BUS} falling below UTCPD_SKVBDCTH as a Sink disconnect indicator. The mechanism used shall be defined in CPSKDCDT (UTCPD_DVCAP2[7]). The Sink side UTCPD shall detect a cable removal within tDisconnectDetect (max. 6 ms) of the Sink disconnect indicator change and enable the automatic discharge circuitry.

6.35.5.7 Discharge

Note: Both auto discharge and force discharge share the same time-out counter. The time-out value for auto discharge is 0x5100, using 32 kHz clock, i.e. ~630ms. The time-out value for force discharge is 0x214D, using 32 kHz clock, i.e. ~260ms.

Clear VBDGTMDS (UTCPD_FUTCTL[3])to enable this timer.

V_{BUS} auto discharge:

- For a Source,
  - Auto Discharge is enabled/disabled by writing to ADGDC (UTCPD_PWRCTL[4])
  - the UTCPD shall fully discharge V_{BUS} to VSAFE5V (max) within tSAFE5V (max. 275 mS) and then to vSAFE0V within tSAFE0V (max. 650 mS) when a disconnect occurs.
- For a Sink,
  - A UTCPD shall use either the VBPS (UTCPD_PWRSTS[2]) transition from 1 to 0, or V_{BUS} falling below UTCPD_SKVBDCTH as a Sink disconnect indicator. The mechanism used shall be defined in CPSKDCDT (UTCPD_DVCAP2[7]). The Sink side UTCPD shall detect a cable removal within tDisconnectDetect (max. 6 ms) of the Sink disconnect indicator change and enable the automatic discharge circuitry.
  - When ADGDC (UTCPD_PWRCTL[4]) is 1 and in Sink mode, the UTCPD shall fully discharge V_{BUS} to VSAFE5V (max) within tSafe5V and then to VSAFE0V (max) within tSafe0V of the removal of the cable.
  - Refer to PD specification for details.

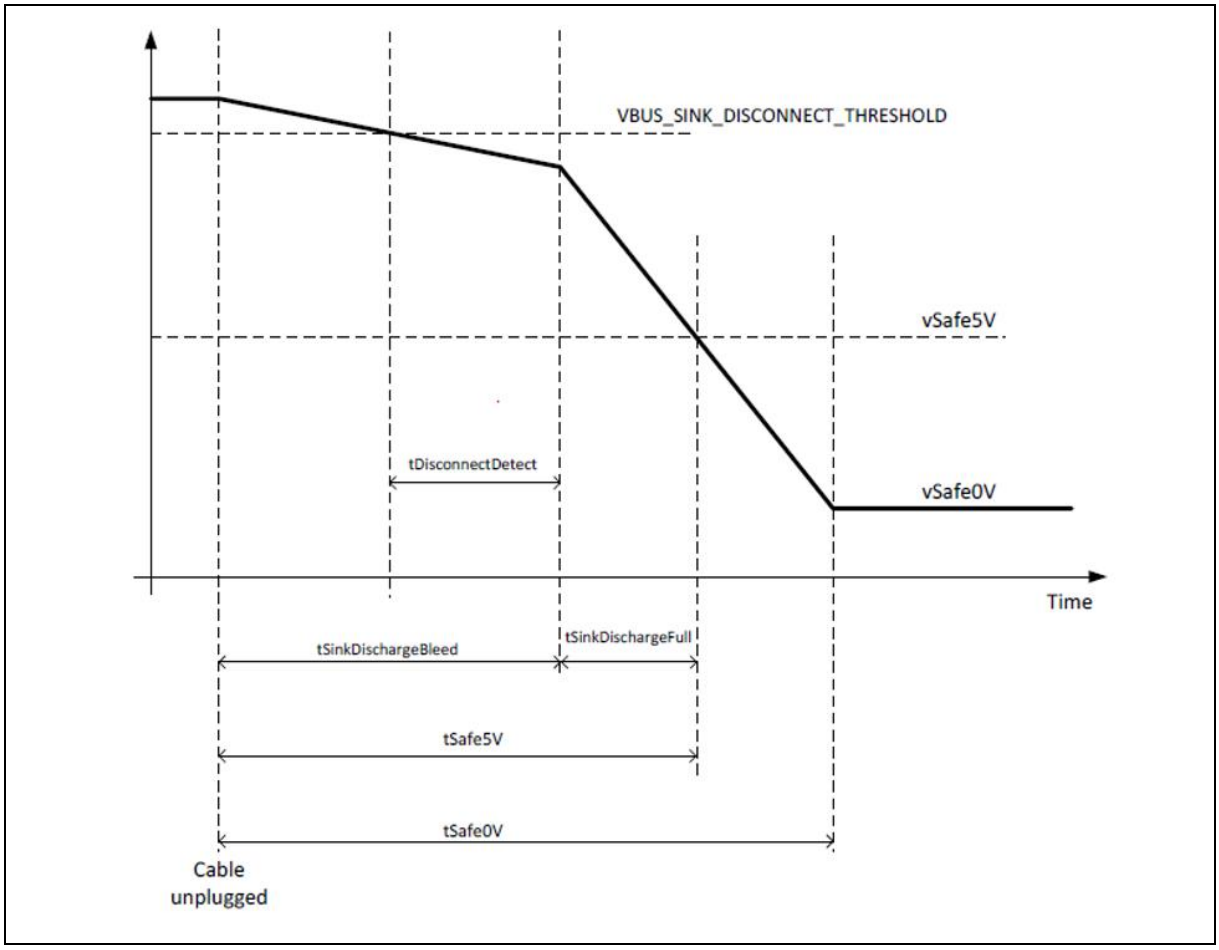


Figure 6.35-8 Discharge Time Diagram

- Auto Discharge is enabled by setting non zero to UTCPD_SKVBDCTH and setting ADGDC (UTCPD_PWRCTL[4]) to one.
- For a Sink, Auto Discharge is disabled by either setting all zeros to UTCPD_SKVBDCTH or setting ADGDC (UTCPD_PWRCTL[4])to zero.

Bleed discharge (for sink only)

When the Source is removed, the system load and optionally the bleed discharge circuit will discharge the Sink bulk capacitance cSnkBulkPd.

The bleed discharge circuit is enabled and disabled by setting BDGEN (UTCPD_PWRCTL[3]) The bleed discharge is a low current load on the V_{BUS}.

Force Discharge (for source only)

- Force discharge is only for Source port.
- Force discharge only applies to source port and when source power can source power higher than VSAFE5V.
- CPU writes to this register (UTCPD_SPDGTH) to set the threshold at which a Source shall stop the Forced Discharge circuit when FDGEN (UTCPD_PWRCTL[2]) is 1.
- A UTCPD acting as a Source shall always discharge to VSAFE0V upon a disconnect, Hard Reset, or Power Role Swap.

This register (UTCPD_SPDGTH) is required by UTCPD which act as a Source and are capable of sourcing non-default V_{BUS} voltages (i.e. voltages other than VSAFE5V).

6.35.5.8 Fast Role Swap and Vbus Auto Drive

Fast Role Swap (FRS) is great new feature of USB PD 3.0, where a device that is providing power can quickly change its power role to become a power consumer in order to maintain a constant data connection. FRS helps prevent any data loss that may occur when power is unexpectedly removed from a device.

When the device is detecting the fast role swap signal, it will compare the voltage conversion of the ADC with the Vbus safe 5V voltage. When the detected input voltage is less than the Vbus safe 5V, the device will start the Vbus Drive action.

It must be noted that since the fast role swap signal specification is realized within 150μs or less, the minimum time for this device design is 30us, so each conversion frequency of the ADC must be faster than 30us, otherwise the PD device may unable to detect fast role swap correctly.

6.35.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
UTCPD Base Address: **UTCPD_BA = 0x400c_6000**				
UTCPD_VID	UTCPD_BA+0x00	R	UTCPD Vendor ID Register	0x0000_0416
UTCPD_PID	UTCPD_BA+0x04	R/W	UTCPD Product ID Register	0x0000_XXXX
UTCPD_DID	UTCPD_BA+0x08	R	UTCPD Device ID Register	0x0000_0001
UTCPD_TCREV	UTCPD_BA+0x0C	R	UTCPD USB Type C Revision Register	0x0000_0013
UTCPD_PDREV	UTCPD_BA+0x10	R/W	UTCPD USB PD Revision Register	0x0000_3111
UTCPD_IS	UTCPD_BA+0x14	R/W	UTCPD Interrupt Status Register	0x0000_0000
UTCPD_IE	UTCPD_BA+0x18	R/W	UTCPD Interrupt Enable Register	0x0000_0FFF
UTCPD_PWRSTSIE	UTCPD_BA+0x1C	R/W	UTCPD Power Status Interrupt Enable Register	0x0000_00BF
UTCPD_FUTSTSIE	UTCPD_BA+0x20	R/W	UTCPD Fault Status Interrupt Enable Register	0x0000_007E
UTCPD_CTL	UTCPD_BA+0x24	R/W	UTCPD Control Register	0x0000_0000
UTCPD_PINPL	UTCPD_BA+0x28	R/W	UTCPD Pin Polarity Control Register	0x0000_0430
UTCPD_ROLCTL	UTCPD_BA+0x2C	R/W	UTCPD Role Control Register	0x0000_000A
UTCPD_FUTCTL	UTCPD_BA+0x30	R/W	UTCPD Fault Control Register	0x0000_0000
UTCPD_PWRCTL	UTCPD_BA+0x34	R/W	UTCPD Power Control Register	0x0000_0060
UTCPD_CCSTS	UTCPD_BA+0x38	R	UTCPD CC Status Register	0x0000_0010
UTCPD_PWRSTS	UTCPD_BA+0x3C	R	UTCPD Power Status Register	0x0000_0008
UTCPD_FUTSTS	UTCPD_BA+0x40	R/W	UTCPD Fault Status Register	0x0000_0000
UTCPD_CMD	UTCPD_BA+0x44	R/W	UTCPD Command Register	0x0000_0000
UTCPD_DVCAP1	UTCPD_BA+0x48	R/W	UTCPD Device Capabilities 1 Register	0x0000_7EDF
UTCPD_DVCAP2	UTCPD_BA+0x4C	R/W	UTCPD Device Capabilities 2 Register	0x0000_00CF
UTCPD_MSHEAD	UTCPD_BA+0x50	R/W	UTCPD Message Header Info Register	0x0000_0004
UTCPD_DTRXEVNT	UTCPD_BA+0x54	R/W	UTCPD Enable Detect RX Event Register	0x0000_0000
UTCPD_RXBCNT	UTCPD_BA+0x58	R	UTCPD RX Byte Count Register	0x0000_0000
UTCPD_RXFTYPE	UTCPD_BA+0x5C	R	UTCPD Received Frame Type Register	0x0000_0000
UTCPD_RXHEAD	UTCPD_BA+0x60	R	UTCPD Received Header Data Register	0x0000_0000
UTCPD_RXDA0	UTCPD_BA+0x70	R	UTCPD Received Data0 Register	0x0000_0000
UTCPD_RXDA1	UTCPD_BA+0x74	R	UTCPD Received Data1 Register	0x0000_0000

UTCPD_RXDA2	UTCPD_BA+0x78	R	UTCPD Received Data2 Register	0x0000_0000
UTCPD_RXDA3	UTCPD_BA+0x7C	R	UTCPD Received Data3 Register	0x0000_0000
UTCPD_RXDA4	UTCPD_BA+0x80	R	UTCPD Received Data4 Register	0x0000_0000
UTCPD_RXDA5	UTCPD_BA+0x84	R	UTCPD Received Data5 Register	0x0000_0000
UTCPD_RXDA6	UTCPD_BA+0x88	R	UTCPD Received Data6 Register	0x0000_0000
UTCPD_TXCTL	UTCPD_BA+0x90	R/W	UTCPD TX Control Register	0x0000_0000
UTCPD_TXBCNT	UTCPD_BA+0x94	R/W	UTCPD TX Byte Count Register	0x0000_0000
UTCPD_TXHEAD	UTCPD_BA+0x98	R/W	UTCPD TX Header Data Register	0x0000_0000
UTCPD_TXDA0	UTCPD_BA+0xA0	R/W	UTCPD Transmit Data0 Register	0x0000_0000
UTCPD_TXDA1	UTCPD_BA+0xA4	R/W	UTCPD Transmit Data1 Register	0x0000_0000
UTCPD_TXDA2	UTCPD_BA+0xA8	R/W	UTCPD Transmit Data2 Register	0x0000_0000
UTCPD_TXDA3	UTCPD_BA+0xAC	R/W	UTCPD Transmit Data3 Register	0x0000_0000
UTCPD_TXDA4	UTCPD_BA+0xB0	R/W	UTCPD Transmit Data4 Register	0x0000_0000
UTCPD_TXDA5	UTCPD_BA+0xB4	R/W	UTCPD Transmit Data5 Register	0x0000_0000
UTCPD_TXDA6	UTCPD_BA+0xB8	R/W	UTCPD Transmit Data6 Register	0x0000_0000
UTCPD_VBVOL	UTCPD_BA+0xC0	R/W	UTCPD V_{BUS} Voltage Register	0x0000_0000
UTCPD_SKVBDCTH	UTCPD_BA+0xC4	R/W	UTCPD V_{BUS} Sink disconnect threshold Register	0x0000_008C
UTCPD_SPDGTH	UTCPD_BA+0xC8	R/W	UTCPD V_{BUS} Stop Discharge threshold Register	0x0000_0020
UTCPD_VBAMH	UTCPD_BA+0xCC	R/W	UTCPD V_{BUS} voltage high alarm threshold Register	0x0000_0000
UTCPD_VBAML	UTCPD_BA+0xD0	R/W	UTCPD V_{BUS} voltage low alarm threshold Register	0x0000_0000
UTCPD_VNDIS	UTCPD_BA+0xD4	R/W	UTCPD Vendor defined Interrupt Status Register	0x0000_0000
UTCPD_VNDIE	UTCPD_BA+0xD8	R/W	UTCPD Vendor defined Interrupt Enable Register	0x0000_0000
UTCPD_MUXSEL	UTCPD_BA+0xDC	R/W	UTCPD Mux Select Register	0x0000_0000
UTCPD_VCDGCTL	UTCPD_BA+0xE0	R/W	UTCPD VCONN Discharge Control Register	0x0000_0000
UTCPD_PHYSLEW	UTCPD_BA+0xE4	R/W	UTCPD PHY Slew Rate Control Register	0x0000_0024
UTCPD_ADGTM	UTCPD_BA+0xE8	R/W	UTCPD Auto Discharge Time Register	0x0000_0064
UTCPD_VSAFE0V	UTCPD_BA+0xEC	R/W	UTCPD Auto Discharge VSAFE0V Register	0x0000_0000
UTCPD_VSAFE5V	UTCPD_BA+0xF0	R/W	UTCPD VSAFE5V Register	0x0000_00C8
UTCPD_RATIO	UTCPD_BA+0xF4	R/W	UTCPD DRP Toggle Ratio Register	0x0000_0000
UTCPD_INTFRAME	UTCPD_BA+0xF8	R/W	UTCPD Inter-Frame Time Register	0x0000_0019
UTCPD_VBOVTH	UTCPD_BA+0xFC	R/W	UTCPD V_{BUS} Over Voltage Threshold Register	0x0000_03FF

UTCPD_VNDINIT	UTCPD_BA+0x100	R/W	UTCPD Vendor Initial Register	0x0000_0064
UTCPD_BMCTXBP	UTCPD_BA+0x104	R/W	UTCPD BMC TX Bit Period Register	0x0000_0027
UTCPD_BMCTXDU	UTCPD_BA+0x108	R/W	UTCPD BMC TX Duty Register	0x0000_0000
UTCPD_VCPSVOL	UTCPD_BA+0x10C	R/W	VCONN Present Voltage Register	0x0000_0060
UTCPD_VCUV	UTCPD_BA+0x110	R/W	VCONN Under Voltage Register	0x0000_0020
UTCPD_BMCSLICE	UTCPD_BA+0x118	R/W	UTCPD BMC SLICE Control Register	0x0824_8423
UTCPD_PHYCTL	UTCPD_BA+0x11C	R/W	UTCPD PHY Power Control Register	0x0000_0001
UTCPD_FRSRXCTL	UTCPD_BA+0x120	R/W	UTCPD CC Fast Swap RX Control Register	0x0000_0080
UTCPD_VCVOL	UTCPD_BA+0x124	R	UTCPD VCONN Voltage Measurement Register	0x0000_0000
UTCPD_CLKINFO	UTCPD_BA+0x300	R/W	UTCPD Clock Information Register	0x0000_0000
UTCPD_PREDET	UTCPD_BA+0x320	R/W	UTCPD Preamble Detect Register	0x321D_4027
UTCPD_NIRR	UTCPD_BA+0x324	R	UTCPD Noise Immunity Record Register	0x0000_3F00
UTCPD_C0PRR	UTCPD_BA+0x328	R	UTCPD Case 0 Path Record Register	0x0000_3F00
UTCPD_C1PRR	UTCPD_BA+0x32C	R	UTCPD Case 1 Path Record Register	0x0000_3F00
UTCPD_C2PRR	UTCPD_BA+0x330	R	UTCPD Case 2 Path Record Register	0x0000_3F00
UTCPD_NBMCSLICE	UTCPD_BA+0x334	R/W	UTCPD New BMC SLICE Control Register	0x0000_0422
UTCPD_BMCDBGR	UTCPD_BA+0x338	R	UTCPD BMC SLICE Value Debug Register	0x0000_0425

6.35.7 Register Description

UTCPD Vendor ID Register (UTCPD_VID)

Register	Offset	R/W	Description	Reset Value
UTCPD_VID	UTCPD_BA+0x00	R	UTCPD Vendor ID Register	0x0000_0416

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VID							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**VID**	**UTCPD Vendor ID** Vendor identifier is used to identify the TCPC vendor, the VID is a unique 16-bit unsigned integer assigned by USB-IF to the Vendor.

UTCPD PRODUCT ID Register (UTCPD_PID)

Register	Offset	R/W	Description	Reset Value
UTCPD_PID	UTCPD_BA+0x04	R/W	UTCPD Product ID Register	0x0000_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PID							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PID**	**UTCPD Product ID** USB Product ID is used to identify the product.

UTCPD Device ID Register (UTCPD_DID)

Register	Offset	R/W	Description	Reset Value
UTCPD_DID	UTCPD_BA+0x08	R	UTCPD Device ID Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DID							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DID**	**UTCPD Device ID** USB Device ID is used to identify the release version of the product.

UTCPD USBTYPEC_REV Register (UTCPD_ TCREV)

Register	Offset	R/W	Description	Reset Value
UTCPD_TCREV	UTCPD_BA+0x0C	R	UTCPD USB Type C Revision Register	0x0000_0013

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TCREV							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**TCREV**	**UTCPD USB Type C Revision** USB Type-C Cable and Connector Specification Revision 1.3.

UTCPD USBPD_REV Register (UTCPD_ PDREV)

Register	Offset	R/W	Description	Reset Value
UTCPD_PDREV	UTCPD_BA+0x10	R/W	UTCPD USB PD Revision Register	0x0000_3111

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PDREV							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PDVER							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[15:8]	**PDREV**	**UTCPD USB PD Revision** USB Power Delivery Specification Revision 3.1.
[7:0]	**PDVER**	**UTCPD USB PD Vision** USB Power Delivery Specification Revision 1.1

UTCPD Interrupt Status Register (UTCPD_ IS)

Register	Offset	R/W	Description	Reset Value
UTCPD_IS	UTCPD_BA+0x14	R/W	UTCPD Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
VNDIS	Reserved			SKDCDTIS	RXOFIS	FUTIS	VBAMLIS
7	6	5	4	3	2	1	0
VBAMHIS	TXOKIS	TXDCIS	TXFALIS	RXHRSTIS	RXSOPIS	PWRSCHIS	CCSCHIS

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**VNDIS**	**Vendor Define Event Detected** 0 = No vendor defined interrupt status has been detected. 1 = A vendor defined interrupt status has been detected. Refer the vender defined interrupt status register. **Note:** It is cleared by software writing 1 into this bit.
[14:12]	**Reserved**	Reserved.
[11]	**SKDCDTIS**	**V_{BUS} Sink Disconnect Detected** 0 = No disconnect detected. 1 = A V_{BUS} Sink disconnect threshold crossing has been detected. This bit will be set when VBMONI (UTCPD_PWRCTL[6]) is enabled and V_{BUS} voltage drop lower than SKVBDCTH (UTCPD_SKVBDCTH[9:0]). **Note:** It is cleared by software writing 1 into this bit.
[10]	**RXOFIS**	**Rx Buffer Overflow** 0 = RX buffer is functioning properly. 1 = RX buffer has overflowed. Writing 1 to this register acknowledges the overflow. The overflow is cleared by writing 1 to RXSOPIS (UTCPD_IS[2])
[9]	**FUTIS**	**Fault Occur** 0 = No fault occurs. 1= A Fault has occurred. Read the FUT_STS register. **Note:** It is cleared by software writing 1 into this bit.
[8]	**VBAMLIS**	**V_{BUS} Voltage Alarm Low** 0 = No Low voltage alarm has occurred. 1= A Low voltage alarm has occurred. This bit will be set high when DSVBAM (UTCPD_PWRCTL[5]]) is low and V_{BUS} voltage is lower than VBAML (UTCPD_VBAML[9:0]). **Note:** It is cleared by software writing 1 into this bit.

[7]	**VBAMHIS**	**V_{BUS} Voltage Alarm High** 0 = No high voltage alarm has occurred. 1= A high voltage alarm has occurred. This bit will be set high when DSVBAM (UTCPD_PWRCTL[5]) is low and V_{BUS} voltage is higher than VBAMH (UTCPD_VBAMH[9:0]). **Note:** It is cleared by software writing 1 into this bit.
[6]	**TXOKIS**	**Transmit SOP* Message Successful** 0 = No TX SOP* transmit. 1 = Reset or SOP* message transmission successful. GoodCRC response received on SOP* message transmission. Transmit SOP* message buffer registers are empty. **Note:** It is cleared by software writing 1 into this bit.
[5]	**TXDCIS**	**Transmit SOP* Message Discarded** 0 = No TX SOP discarded. 1 = Reset or SOP* message transmission not sent due to incoming receive message. Transmit SOP* message buffer registers are empty. **Note:** It is cleared by software writing 1 into this bit.
[4]	**TXFALIS**	**Transmit SOP Fail** 0 = No Transmit SOP fail. 1 = SOP* message transmission not successful, no GoodCRC response received on SOP* message transmission. Transmit SOP* message buffer registers are empty. **Note:** It is cleared by software writing 1 into this bit.
[3]	**RXHRSTIS**	**Received Hard Reset** 0 = No Hard reset Received. 1 = Received Hard reset. **Note:** It is cleared by software writing 1 into this bit.
[2]	**RXSOPIS**	**Received SOP Message** 0 = No SOP message Received. 1 = Received SOP message UTCPD_RXBCNT being set to 0 does not set this bit. **Note:** It is cleared by software writing 1 into this bit.
[1]	**PWRSCHIS**	**Power Status Changed** 0 = Power status not change. 1 = Power status changed. **Note:** It is cleared by software writing 1 into this bit.
[0]	**CCSCHIS**	**CC Status Changed** 0 = CC status not change. 1= CC status changed. **Note:** It is cleared by software writing 1 into this bit.

UTCPD Interrupt Enbale Register (UTCPD_ IE)

Register	Offset	R/W	Description	Reset Value
UTCPD_IE	UTCPD_BA+0x18	R/W	UTCPD Interrupt Enable Register	0x0000_0FFF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
VNDIE	Reserved			SKDCDTIE	RXOFIE	FUTIE	VBAMLIE
7	6	5	4	3	2	1	0
VBAMHIE	TXSOKIE	TXDCIE	TXFAILIE	RXHRSTIE	RXSOPIE	PWRSCHIE	CCSCHIE

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**VNDIE**	**Vendor Define Event Detected Interrupt Enable** 0 = Vendor Define Event Detected Interrupt Disabled. 1 = Vendor Define Event Detected Interrupt Enabled.
[14:12]	**Reserved**	Reserved.
[11]	**SKDCDTIE**	**V_{BUS} Sink Disconnect Detected Interrupt Enable** 0 = V_{BUS} Sink Disconnect Detected Interrupt Disabled. 1 = V_{BUS} Sink Disconnect Detected Interrupt Enabled.
[10]	**RXOFIE**	**Rx Buffer Overflow Interrupt Enable** 0 = Rx Buffer Overflow Interrupt Disabled. 1 = Rx Buffer Overflow Interrupt Enabled.
[9]	**FUTIE**	**Fault Occur Interrupt Enable** 0 = Fault Occur Interrupt Disabled. 1 = Fault Occur Interrupt Enabled.
[8]	**VBAMLIE**	**V_{BUS} Voltage Alarm Low Interrupt Enable** 0 = V_{BUS} Voltage Alarm Low Interrupt Disabled. 1 = V_{BUS} Voltage Alarm Low Interrupt Enabled.
[7]	**VBAMHIE**	**V_{BUS} Voltage Alarm High Interrupt Enable** 0 = V_{BUS} Voltage Alarm High Interrupt Disabled. 1 = V_{BUS} Voltage Alarm High Interrupt Enabled.
[6]	**TXSOKIE**	**Transmit SOP* Message Successful Interrupt Enable** 0 = Transmit SOP* Message Successful Interrupt Disabled. 1 = Transmit SOP* Message Successful Interrupt Enabled.

[5]	**TXDCIE**	**Transmit SOP* Message Discarded Interrupt Enable** 0 = Transmit SOP* Message Discarded Interrupt Disabled. 1 = Transmit SOP* Message Discarded Interrupt Enabled.
[4]	**TXFAILIE**	**Transmit SOP Fail Interrupt Enable** 0 = Transmit SOP Fail Interrupt Disabled. 1 = Transmit SOP Fail Interrupt Enabled.
[3]	**RXHRSTIE**	**Received Hard Reset Interrupt Enable** 0 = Received Hard Reset Interrupt Disabled. 1 = Received Hard Reset Interrupt Enabled.
[2]	**RXSOPIE**	**Received SOP Message Interrupt Enable** 0 = Received SOP Message Interrupt Disabled. 1 = Received SOP Message Interrupt Enabled.
[1]	**PWRSCHIE**	**Power Status Changed Interrupt Enable** 0 = Power Status Changed Interrupt Disabled. 1 = Power Status Changed Interrupt Enabled.
[0]	**CCSCHIE**	**CC Status Changed Interrupt Enable** 0 = CC Status Changed Interrupt Disabled. 1 = CC Status Changed Interrupt Enabled.

UTCPD Power Status Interrupt Enable Register (UTCPD_ PWRSTSIE)

Register	Offset	R/W	Description	Reset Value
UTCPD_PWRSTSIE	UTCPD_BA+0x1C	R/W	UTCPD Power Status Interrupt Enable Register	0x0000_00BF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DACONIE	Reserved	SRHVIE	SRVBIE	VBDTDGIE	VBPSIE	VCPSIE	SKVBIE

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**DACONIE**	**Debug Accessory Connected Status Interrupt Enable** 0 = Debug Accessory Connected Status Interrupt Disabled. 1 = Debug Accessory Connected Status Interrupt Enabled.
[6]	**Reserved**	Reserved.
[5]	**SRHVIE**	**Sourcing High Voltage Status Interrupt Enable** 0 = Sourcing High Voltage Status Interrupt Disabled. 1 = Sourcing High Voltage Status Interrupt Enabled.
[4]	**SRVBIE**	**Sourcing V_{BUS} Status Interrupt Enable** 0 = Sourcing V_{BUS} Status Interrupt Disabled. 1 = Sourcing V_{BUS} Status Interrupt Enabled.
[3]	**VBDTDGIE**	**V_{BUS} Detection Status Change Interrupt Enable** 0 = V_{BUS} Detection Status Change Interrupt Disabled. 1 = V_{BUS} Detection Status Change Interrupt Enabled.
[2]	**VBPSIE**	**V_{BUS} Present Status Interrupt Enable** 0 = V_{BUS} Present Status Interrupt Disabled. 1 = V_{BUS} Present Status Interrupt Enabled.
[1]	**VCPSIE**	**VCONN Present Status Interrupt Enable** 0 = VCONN Present Status Interrupt Disabled. 1 = VCONN Present Status Interrupt Enabled.
[0]	**SKVBIE**	**Sinking V_{BUS} Status Interrupt Enable** 0 = Sinking V_{BUS} Status Interrupt Disabled. 1 = Sinking V_{BUS} Status Interrupt Enabled.

UTCPD Fault Status Interrupt Enable Register (UTCPD_FUTSTSIE)

Register	Offset	R/W	Description	Reset Value
UTCPD_FUTSTSIE	UTCPD_BA+0x20	R/W	UTCPD Fault Status Interrupt Enable Register	0x0000_007E

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	FOFFVBIE	ADGFALIE	FDGFALIE	VBOCIE	VBOVIE	VCOCIE	Reserved

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**FOFFVBIE**	**Force Off V_{BUS} Interrupt Enable** 0 = Force Off V_{BUS} Interrupt Disabled. 1 = Force Off V_{BUS} Interrupt Enabled.
[5]	**ADGFALIE**	**Auto Discharge Failed Interrupt Enable** 0 = Auto Discharge Failed Interrupt Disabled. 1 = Auto Discharge Failed Interrupt Enabled.
[4]	**FDGFALIE**	**Force Discharge Failed Interrupt Enable** 0 = Force Discharge Failed Interrupt Disabled. 1 = Force Discharge Failed Interrupt Enabled.
[3]	**VBOCIE**	**External V_{BUS} OCP Fault Interrupt Enable** 0 = External V_{BUS} OCP Fault Interrupt Disabled. 1 = External V_{BUS} OCP Fault Interrupt Enabled.
[2]	**VBOVIE**	**Internal V_{BUS} OVP Fault Interrupt Enable** 0 = Internal V_{BUS} OVP Fault Interrupt Disabled. 1 = Internal V_{BUS} OVP Fault Interrupt Enabled.
[1]	**VCOCIE**	**VCONN OCP Fault Interrupt Enable** 0 = VCONN OCP Fault Interrupt Disabled. 1 = VCONN OCP Fault Interrupt Enabled.
[0]	**Reserved**	Reserved.

UTCPD Control Register (UTCPD_CTL)

Register	Offset	R/W	Description	Reset Value
UTCPD_CTL	UTCPD_BA+0x24	R/W	UTCPD Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**BISTEN**	**ORIENT**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**BISTEN**	**BIST Test Mode** Setting this bit to 1 is intended to be used only when a USB compliance tester is using USB BIST Test Data to test the PHY layer of the UTCPD. The CPU should clear this bit when a detach is detected. 0 = Normal Operation. 1 = BIST Test Mode.
[0]	**ORIENT**	**Plug Orientation** 0 = When VCONN is enabled, apply it to the CC2 pin. Monitor the CC1 pin for BMC communications if PD messaging is enabled. 1 = When VCONN is enabled, apply it to the CC1 pin. Monitor the CC2 pin for BMC communications if PD messaging is enabled.

UTCPD Pin Polarity Control Register (UTCPD_PINPL)

Register	Offset	R/W	Description	Reset Value
UTCPD_PINPL	UTCPD_BA+0x28	R/W	UTCPD Pin Polarity Control Register	0x0000_0430

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					FOFFVBPL	VCDGENPL	VCENPL
7	6	5	4	3	2	1	0
Reserved		VBOCPL	VCOCPL	TXFRSPL	VBDGENPL	VBSKENPL	VBSRENPL

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**FOFFVBPL**	**Force Off V_{BUS} event Polarity** 0 = Low active. 1 = High active.
[9]	**VCDGENPL**	**VCONN Discharge Enable Polarity** 0 = Low active. 1 = High active.
[8]	**VCENPL**	**VCONN Enable Polarity** 0 = Low active. 1 = High active.
[7:6]	**Reserved**	Reserved.
[5]	**VBOCPL**	**V_{BUS} Overcurrent event Polarity** 0 = Low active. 1 = High active.
[4]	**VCOCPL**	**VCONN Overcurrent event Polarity** 0 = Low active. 1 = High active.
[3]	**TXFRSPL**	**Fast Role Swap TX Polarity** 0 = Low active. 1 = High active.
[2]	**VBDGENPL**	**V_{BUS} Discharge Enable Polarity** 0 = Low active. 1 = High active.

[1]	**VBSKENPL**	**V_{BUS} Sink Enable Polarity** 0 = Low active. 1 = High active.
[0]	**VBSRENPL**	**V_{BUS} Source Enable Polarity** 0 = Low active. 1 = High active.

UTCPD Role Control Register (UTCPD_ROLCTL)

Register	Offset	R/W	Description	Reset Value
UTCPD_ROLCTL	UTCPD_BA+0x2C	R/W	UTCPD Role Control Register	0x0000_000A

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	DRP	RPVALUE		CC2		CC1	

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**DRP**	**DRP** 0 = No DRP. 1 = DRP. The UTCPD toggles CC1 & CC2 after receiving LK4CON (UTCPD_CMD[5]) and until a connection is detected. Upon connection, the UTCPD shall resolve to either an Rp or Rd and report the CC1/CC2 State in the CC_STS register.
[5:4]	**RPVALUE**	**Rp Value** 00 = Rp default. 01 = Rp 1.5A. 10 = Rp 3.0A. 11 = Reserved.
[3:2]	**CC2**	**CC2** 00 = Reserved. 01 = Rp (Use Rp definition in RPVALUE). 10 = Rd. 11 = Open (Disconnect or don't care).
[1:0]	**CC1**	**CC1** 00 = Reserved. 01b = Rp (Use Rp definition in RPVALUE). 10 = Rd. 11 = Open (Disconnect or don't care).

Note: When CC1 and CC2 are set to Open and DRP = 0, the UTCPD will power down the PHY and CC Status comparators.

UTCPD Fault Control Register (UTCPD_FUTCTL)

Register	Offset	R/W	Description	Reset Value
UTCPD_FUTCTL	UTCPD_BA+0x30	R/W	UTCPD Fault Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			FOFFVBDS	VBDGTMDS	VBOCDTDS	VBOVDTDS	VCOCDTDS

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**FOFFVBDS**	**Force Off V_{BUS} Disable** 0 = Force off **V_{BUS}** Enabled. 1 = Force off **V_{BUS}** Disabled.
[3]	**VBDGTMDS**	**V_{BUS} Discharge Fault Detection Timer Disable** 0 = V_{BUS} Discharge Fault Detection Timer Enabled. 1 = V_{BUS} Discharge Fault Detection Timer Disabled. This enables the timer for both force discharge and auto discharge. No timer for bleed discharge. Once time-out, UTCPD will compare V_{BUS} voltage with VSAFE0V or SP_DGTH, depends on setting of ADGDC and FDGEN.
[2]	**VBOCDTDS**	**External V_{BUS} Overcurrent Protection Fault Detect Disable** 0 = External OCP circuit Enabled. 1 = External OCP circuit Disabled.
[1]	**VBOVDTDS**	**Internal V_{BUS} Over Voltage Protection Fault Detect Disable** 0 = Internal OVP circuit Enabled. 1 = Internal OVP circuit Disabled. **Note:** Internal V_{BUS} over voltage protection means to use ADC to measure V_{BUS} voltage.
[0]	**VCOCDTDS**	**External VCONN Overcurrent Fault Detect Disable** 0 = External VCONN OCP circuit Enabled. 1 = External VCONN OCP circuit Disabled.

UTCPD PWR Control Register (UTCPD_PWRCTL)

Register	Offset	R/W	Description	Reset Value
UTCPD_PWRCTL	UTCPD_BA+0x34	R/W	UTCPD Power Control Register	0x0000_0060

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	VBMONI	DSVBAM	ADGDC	BDGEN	FDGEN	VCPWR	VCEN

Bits	Description	
[31:4]	**Reserved**	Reserved.
[6]	**VBMONI**	**V_{BUS} Voltage Monitor Enable** 0 = V_{BUS} Voltage Monitoring Enabled. 1 = V_{BUS} Voltage Monitoring Disabled (default). Controls V_{BUS} voltage Monitoring. UTCPD_VBVOL shall report all zeroes if disabled.
[5]	**DSVBAM**	**Disable V_{BUS} Voltage Alarms** 0 = V_{BUS} Voltage Alarms Power status reporting Enabled. 1 = V_{BUS} Voltage Alarms Power status reporting Disabled (default).
[4]	**ADGDC**	**Auto Discharge Disconnect** 0 = The UTCPD shall not automatically discharge V_{BUS} based on V_{BUS} voltage (default). 1 = The UTCPD shall automatically discharge when disconnect detected. Setting this bit in a Source UTCPD triggers the following actions upon a disconnect detection: 1. Disable sourcing power over V_{BUS} 2. V_{BUS} discharge Sourcing power over V_{BUS} shall be disabled before or at the same time as starting V_{BUS} discharge. Setting this bit in a Sink UTCPD triggers the following action upon a disconnect detection: 1. V_{BUS} discharge The UTCPD shall automatically disable discharge (without clearing this bit) once the voltage on V_{BUS} is below vSafe0V (max). UTCPD shall not re-apply discharge circuit if V_{BUS} rises above vSafe0V. This bit will only be cleared by CPU.

[3]	**BDGEN**	**Enable Bleed Discharge** 0 = Disable bleed discharge (default). 1 = Enable bleed discharge of V_{BUS}. Bleed Discharge is a low current discharge to provide a minimum load current if needed 10 K Ohms or 2mA recommended. Bleed discharge is used for sink only. This bit will only be cleared by CPU.
[2]	**FDGEN**	**Enable Force Discharge** 0 = Disable forced discharge (default). 1 = Enable forced discharge of V_{BUS}. Force discharge is used for source only. This bit will only be cleared by CPU.
[1]	**VCPWR**	**VCONN Power Supported** 0 = UTCPD delivers at least 1W on VCONN. 1 = UTCPD delivers at least the power indicated in CPVCPWR (UTCPD_DVCAP2[3:1]).
[0]	**VCEN**	**Enable VCONN** 0 = Disable VCONN Source (default). 1 = Enable VCONN Source to CC.

UTCPD CC Status Register (UTCPD_CCSTS)

Register	Offset	R/W	Description	Reset Value
UTCPD_CCSTS	UTCPD_BA+0x38	R	UTCPD CC Status Register	0x0000_0010

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		LK4CONN	CONRLT	CC2STATE		CC1STATE	

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**LK4CONN**	**Looking for Connection** 0 = UTCPD is not actively looking for a connection. A transition from '1' to '0' indicates a potential connection has been found. 1 = UTCPD is looking for a connection (toggling as a DRP or looking for a connection as Sink/Source only condition).
[4]	**CONRLT**	**Connect Result (Read Only)** 0 = the UTCPD is presenting Rp. 1 = the UTCPD is presenting Rd.

[3:2]	**CC2STATE**	**CC2 State**
		If (CC2 (UTCPD_ROLCTL[3:2]=Rp) or (CONRLT (UTCPD_CCSTS[4]) = 0).
		00 = SRC.Open (Open, Rp).
		01 = SRC.Ra (below maximum vRa).
		10 = SRC.Rd (within the vRd range).
		11 = reserved.
		If (CC2 (UTCPD_ROLCTL[3:2])=Rd) or (CONRLT (UTCPD_CCSTS[4]) = 1).
		00 = SNK.Open (Below maximum vRa).
		01 = SNK.Default (Above minimum vRd-Connect).
		10 = SNK.Power1.5 (Above minimum vRd-Connect) Detects Rp 1.5A.
		11 = SNK.Power3.0 (Above minimum vRd-Connect) Detects Rp 3.0A.
		If CC2 (UTCPD_ROLCTL[3:2])=Reserved, this field is set to 00.
		If CC2 (UTCPD_ROLCTL[3:2])=Open, this field is set to 00.
		This field always returns 00 if (LK4CONN=1) or (VCEN (UTCPD_PWRCTL[0])=1 and ORIENT (UTCPD_CTL[0])=0). Otherwise, the returned value depends upon CC2 (UTCPD_ROLCTL[3:2])

[1:0]	**CC1STATE**	**CC1 State**
		If (CC1 (UTCPD_ROLCTL[1:0])=Rp) or (CONRLT (UTCPD_CCSTS[4]) = 0).
		00 = SRC.Open (Open, Rp).
		01 = SRC.Ra (below maximum vRa).
		10 = SRC.Rd (within the vRd range).
		11 = reserved.
		If (CC1 (UTCPD_ROLCTL[1:0])=Rd) or (CONRLT (UTCPD_CCSTS[4]) = 1).
		00 = SNK.Open (Below maximum vRa).
		01 = SNK.Default (Above minimum vRd-Connect).
		10 = SNK.Power1.5 (Above minimum vRd-Connect) Detects Rp 1.5A.
		11 = SNK.Power3.0 (Above minimum vRd-Connect) Detects Rp 3.0A.
		If CC1 (UTCPD_ROLCTL[1:0])=Reserved, this field is set to 00.
		If CC1 (UTCPD_ROLCTL[1:0])=Open, this field is set to 00.
		This field always returns 00 if (LK4CONN=1) or (VCEN (UTCPD_PWRCTL[0])]=1 and ORIENT (UTCPD_CTL[0])=1). Otherwise, the returned value depends upon CC1 (UTCPD_ROLCTL[1:0])

UTCPD PWR Status Register (UTCPD_PWRSTS)

Register	Offset	R/W	Description	Reset Value
UTCPD_PWRSTS	UTCPD_BA+0x3C	R	UTCPD Power Status Register	0x0000_0008

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DACON	Reserved	SRHV	SRVB	VBPSDTEN	VBPS	VCPS	SKVB

Bits	Description	
[31:7]	**Reserved**	Reserved.
[7]	**DACON**	**Debug Accessory Connected** 0 = No Debug Accessory connected (default). 1 = Debug Accessory connected.
[6]	**Reserved**	Reserved.
[5]	**SRHV**	**Sourcing High Voltage** 0 = VSAFE5V. 1 = Higher voltage. This bit does not control the power path, it just provides a monitor of the status. This bit is asserted as long as the UTCPD is sourcing nondefault voltage over V_{BUS} (i.e. not VSAFE5V) as a response to CPU writing 0x88 to CMD (Source V_{BUS} High Voltage)
[4]	**SRVB**	**Sourcing V_{BUS}** 0 = Sourcing V_{BUS} is disabled. 1 = Sourcing V_{BUS} is enabled. This bit does not control the path, just provides a monitor of the status.
[3]	**VBPSDTEN**	**V_{BUS} Present Detection Enabled** 0 = V_{BUS} Present Detection Disabled. 1 = V_{BUS} Present Detection Enabled (default).
[2]	**VBPS**	**V_{BUS} Present** 0 = V_{BUS} Disconnected. 1 = V_{BUS} Connected. The UTCPD shall report V_{BUS} present when UTCPD detects V_{BUS} rises above **4V.** The UTCPD shall report V_{BUS} is not present when UTCPD detects V_{BUS} falls below 3.5V.
[1]	**VCPS**	**VCONN Present** 0 = VCONN is not present. 1 = This bit is asserted when VCONN present CC1 or CC2. Threshold is fixed at 2.4V. If VCEN (UTCPD_PWRCTL[0])] is disabled VCONN Present should be set to 0.

[0]	**SKVB**	**Sinking V_{BUS}** 0 = Sink is Disconnected (Default). 1 = UTCPD is sinking V_{BUS} to the system load.

UTCPD Fault Status Register (UTCPD_FUTSTS)

Register	Offset	R/W	Description	Reset Value
UTCPD_FUTSTS	UTCPD_BA+0x40	R/W	UTCPD Fault Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
INVHRST	FOFFVB	ADGFAL	FDGFAL	VBOCFUT	VBOVFUT	VCOCFUT	Reserved

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**INVHRST**	**Invalid Hard Reset** 0 = No invalid hard reset event. 1 = Invalid hard reset event active.
[6]	**FOFFVB**	**Force Off V_{BUS}** 0 = No Fault Detected, no action (default and not supported). 1 = V_{BUS} Source/Sink has been forced off due to external fault. The UTCPD has disconnected V_{BUS} due to external inputs (EINT0 ~ EINT5)
[5]	**ADGFAL**	**Auto Discharge Failed** 0 = No discharge failure. 1 = Discharge commanded by the TCPM failed. If ADGDC (UTCPD_PWRCTL[4]) is set, the UTCPD shall report discharge fails if V_{BUS} is not below vSafe0V within tSafe0V
[4]	**FDGFAL**	**Force Discharge Failed** 0 = No discharge failure. 1 = Discharge commanded by the TCPM failed. If FDGEN (UTCPD_PWRCTL[2]) is set, the UTCPD shall report a discharge fails if V_{BUS} is not below vSafe0V within tSafe0V.
[3]	**VBOCFUT**	**V_{BUS} Overcurrent Protection Fault** 0 = Not in an overcurrent protection state. 1 = Overcurrent fault latched.
[2]	**VBOVFUT**	**V_{BUS} Over Voltage Protection Fault** 0 = No Fault detected. 1 = Over-voltage fault latched.
[1]	**VCOCFUT**	**VCONN Overcurrent Protection Fault** 0 = Not in an overcurrent protection state. 1 = Overcurrent fault latched.

[0]	**Reserved**	Reserved.

UTCPD Command Register (UTCPD_CMD)

Register	Offset	R/W	Description	Reset Value
UTCPD_CMD	UTCPD_BA+0x44	R/W	UTCPD Command Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMD							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**CMD**	**Command Set** 0x22 = Disable V_{BUS} Detect. 0x33 = Enable V_{BUS} Detect. 0x44 = Disable Sink V_{BUS}. 0x55h = Enable Sink V_{BUS}. 0x66 = Disable Source V_{BUS}. 0x77 = Enable Source V_{BUS} 5V. 0x88 = Source V_{BUS} High Voltage. 0x99h = Look4Connection. 0xAA = RxOneMore. Others: Reserved. The Command is issued by the CPU. The Command is cleared by the UTCPD after being acted upon. It always read as 0.

UTCPD Device Capbilities1 Register (UTCPD_DVCAP1)

Register	Offset	R/W	Description	Reset Value
UTCPD_DVCAP1	UTCPD_BA+0x48	R/W	UTCPD Device Capabilities 1 Register	0x0000_7EDF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	CPVBOCP	CPVBOVP	CPBDG	CPFDG	CPVBAM	CPSRRE	
7	6	5	4	3	2	1	0
CPROL			CPSDBG	CPSRVC	CPSKVB	CPSRHV	CPSRVB

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14]	**CPVBOCP**	**V_{BUS} OCP Reporting** 0 = V_{BUS} OCP is not reported. 1 = V_{BUS} OCP is reported.
[13]	**CPVBOVP**	**V_{BUS} OVP Reporting** 0 = V_{BUS} OVP is not reported. 1 = V_{BUS} OVP is reported.
[12]	**CPBDG**	**Bleed Discharge** 0 = No Bleed Discharge implemented. 1 = Bleed Discharge is implemented.
[11]	**CPFDG**	**Force Discharge** 0 = No Force Discharge implemented. 1 = Force Discharge is implemented.
[10]	**CPVBAM**	**V_{BUS} Measurement and Alarm Capable** 0 = No V_{BUS} voltage measurement nor V_{BUS} Alarms. 1 = V_{BUS} voltage measurement and V_{BUS} Alarms.
[9:8]	**CPSRRE**	**Source Resistor Supported** 00 = Rp default only. 01 = Rp 1.5A and default. 10 = Rp 3.0A, 1.5A, and default. 11 = Reserved.

[7:5]	**CPROL**	**Roles Supported** 000 = USB Type-C Port Manager can configure the Port as Source only or Sink only (not DRP). 001 = Source only. 010 = Sink only. 011 = Sink with accessory support. 100 = DRP only. 101 = Source, Sink, DRP, Adapter/Cable all supported. 110 = Source, Sink, DRP. 111 = Not valid.
[4]	**CPSDBG**	**SOP'_DBG/SOP"_DBG Support** 0 = All SOP* except SOP'_DBG/SOP"_DBG. 1 = All SOP* messages are supported.
[3]	**CPSRVC**	**Source VCONN** 0 = UTCPD is not capable of switching VCONN. 1 = UTCPD is capable of switching VCONN.
[2]	**CPSKVB**	**Sink V_{BUS}** 0 = UTCPD is not capable controlling the sink path to the system load. 1 = UTCPD is capable of controlling the sink path to the system load.
[1]	**CPSRHV**	**Source High Voltage V_{BUS}** 0 = UTCPD is not capable of controlling the source high voltage path to V_{BUS}. 1 = UTCPD is capable of controlling the source high voltage path to V_{BUS}.
[0]	**CPSRVB**	**Source V_{BUS}** 0 = TCPC is not capable of controlling the source path to V_{BUS}. 1 = TCPC is capable of controlling the source path to V_{BUS}.

UTCPD Device Capabilities2 Register (UTCPD_DVCAP2)

Register	Offset	R/W	Description	Reset Value
UTCPD_DVCAP2	UTCPD_BA+0x4C	R/W	UTCPD Device Capabilities 2 Register	0x0000_00CF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CPSKDCDT	CPSPDGTH	CPVBAMLS		CPVCPWR			CPVCOC

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**CPSKDCDT**	**Sink Disconnect Detection** 0 = UTCPD_SKVBDCTH not implemented. 1 = UTCPD_SKVBDCTH implemented.
[6]	**CPSPDGTH**	**Stop Discharge Threshold** 0 = UTCPD_SPDGTH not implemented. 1 = UTCPD_SPDGTH implemented.
[5:4]	**CPVBAMLS**	**V_{BUS} Voltage Alarm LSB** 00 = UTCPD has 25mV LSB for its voltage alarm and uses all 10 bits in. VB_AMH and VB_AML. Others = Reserved.
[3:1]	**CPVCPWR**	**VCONN Power Supported** 000 = 1.0W. 001 = 1.5W. 010 = 2.0W. 011 = 3W. 100 = 4W. 101 = 5W. 110 = 6W. 111 = External.
[0]	**CPVCOC**	**VCONN Overcurrent Fault Capable** 0 = UTCPD is not capable of detecting a Vconn fault. 1 = UTCPD is capable of detecting a Vconn fault.

UTCPD Message Header INFO Register (UTCPD_MSHEAD)

Register	Offset	R/W	Description	Reset Value
UTCPD_MSHEAD	UTCPD_BA+0x50	R/W	UTCPD Message Header Info Register	0x0000_0004

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			CABPLG	DAROL	PDREV		PWRROL

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**CABPLG**	**Cable Plug** 0 = Message originated from Source, Sink, or DRP. 1 = Message originated from a Cable Plug.
[3]	**DAROL**	**Data Role** 0 = UFP. 1 = DFP.
[2:1]	**PDREV**	**USB PD Specification Revision** 00 = Revision 1.0. 01 = Revision 2.0. 10 = Revision 3.0. 11 = Revision 3.1.
[0]	**PWRROL**	**Power Role** 0 = Sink. 1 = Source.

UTCPD Enable Detect RX Event Register (UTCPD_DTRXEVNT)

Register	Offset	R/W	Description	Reset Value
UTCPD_DTRXEVNT	UTCPD_BA+0x54	R/W	UTCPD Enable Detect RX Event Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	CABRSTEN	HRSTEN	SDBGPPEN	SDBGPEN	SOPPPEN	SOPPEN	SOPEN

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**CABRSTEN**	**Enable Cable Reset** 0 = UTCPD does not detect Cable Reset signaling (default). 1 = UTCPD detects Cable Reset signaling.
[5]	**HRSTEN**	**Enable Hard Reset** 0 = UTCPD does not detect Hard Reset signaling (default). 1 = UTCPD detects Hard Reset signaling.
[4]	**SDBGPPEN**	**Enable SOP_DBG'' Message** 0 = UTCPD does not detect SOP_DBG'' message (default). 1 = UTCPD detects SOP_DBG'' message.
[3]	**SDBGPEN**	**Enable SOP_DBG' Message** 0 = UTCPD does not detect SOP_DBG' message (default). 1 = UTCPD detects SOP_DBG' message.
[2]	**SOPPPEN**	**Enable SOP'' Message** 0 = UTCPD does not detect SOP'' message (default). 1 = UTCPD detects SOP'' message.
[1]	**SOPPEN**	**Enable SOP' Message** 0 = UTCPD does not detect SOP' message (default). 1 = UTCPD detects SOP' message.
[0]	**SOPEN**	**Enable SOP Message** 0 = UTCPD does not detect SOP message (default). 1 = UTCPD detects SOP message.

UTCPD RX Byte Count Register (UTCPD_RXBCNT)

Register	Offset	R/W	Description	Reset Value
UTCPD_RXBCNT	UTCPD_BA+0x58	R	UTCPD RX Byte Count Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXBCNT							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**RXBCNT**	**Receive Byte Count** Indicates number of bytes in this register that are not stale. CPU should read the first RXBCNT bytes in this register. This is the number of bytes in the UTCPD_RXDAx plus three (for the RXFTYPE and RXHEAD). The UTCPD shall clear the UTCPD_DTRXEVNT and the UTCPD_RXBCNT register to disable the PD message passing when CPU writes the UTCPD_TXCTL register requesting a Hard Reset

UTCPD Received Frame Type Register (UTCPD_RXFTYPE)

Register	Offset	R/W	Description	Reset Value
UTCPD_RXFTYPE	UTCPD_BA+0x5C	R	UTCPD Received Frame Type Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**RXFTYPE**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**RXFTYPE**	**Received Frame Type** 000 = Received SOP. 001 = Received SOP'. 010 = Received SOP". 011 = Received SOP_DBG'. 100 = Received SOP_DBG". 110 = Received Cable Reset. All others are reserved.

UTCPD Received Header Register (UTCPD_RXHEAD)

Register	Offset	R/W	Description	Reset Value
UTCPD_RXHEAD	UTCPD_BA+0x60	R	UTCPD Received Header Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXHEAD1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXHEAD0							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**RXHEAD1**	USB PD message header byte 1
[7:0]	**RXHEAD0**	USB PD message header byte 0

UTCPD Received Data0~6 Register (UTCPD_RXDAx)

Register	Offset	R/W	Description	Reset Value
UTCPD_RXDA0	UTCPD_BA+0x70	R	UTCPD Received Data0 Register	0x0000_0000
UTCPD_RXDA1	UTCPD_BA+0x74	R	UTCPD Received Data1 Register	0x0000_0000
UTCPD_RXDA2	UTCPD_BA+0x78	R	UTCPD Received Data2 Register	0x0000_0000
UTCPD_RXDA3	UTCPD_BA+0x7C	R	UTCPD Received Data3 Register	0x0000_0000
UTCPD_RXDA4	UTCPD_BA+0x80	R	UTCPD Received Data4 Register	0x0000_0000
UTCPD_RXDA5	UTCPD_BA+0x84	R	UTCPD Received Data5 Register	0x0000_0000
UTCPD_RXDA6	UTCPD_BA+0x88	R	UTCPD Received Data6 Register	0x0000_0000

31	30	29	28	27	26	25	24
RXDAB3							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXDAB2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXDAB1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAB0							

Bits	Description	
[31:24]	**RXDAB3**	USB PD Received Datax byte 3
[23:16]	**RXDAB2**	USB PD Received Datax byte 2
[15:8]	**RXDAB1**	USB PD Received Datax byte 1
[7:0]	**RXDAB0**	USB PD Received Datax byte 0

UTCPD TX Control Register (UTCPD_TXCTL)

Register	Offset	R/W	Description	Reset Value
UTCPD_TXCTL	UTCPD_BA+0x90	R/W	UTCPD TX Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		RETRYCNT		Reserved	TXSTYPE		

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:4]	**RETRYCNT**	**Retry Counter** 00 = No message retry is required. 01 = Automatically retry message transmission once. 10 = Automatically retry message transmission twice. 11 = Automatically retry message transmission three times.
[3]	**Reserved**	Reserved.
[2:0]	**TXSTYPE**	**Transmit SOP* Message** 000 = Transmit SOP. 001 = Transmit SOP'. 010 = Transmit SOP". 011 = Transmit SOP_DBG'. 100 = Transmit SOP_DBG''. 101 = Transmit Hard Reset. 110 = Transmit Cable Reset. 111 = Transmit BIST Carrier Mode 2. The UTCPD shall ignore the Retry Counter bits when transmitting a Hard Reset, Cable Reset, or BIST Carrier

UTCPD TX Byte Count Register (UTCPD_TBCNT)

Register	Offset	R/W	Description	Reset Value
UTCPD_TXBCNT	UTCPD_BA+0x94	R/W	UTCPD TX Byte Count Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXBCNT							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**TXBCNT**	**Transmit Byte Count** This is the number of bytes in the UTCPD_TXDA plus two (for the TXHEAD) If a previous transmit request has not yet completed when TX_CTL is written requesting a Hard Reset, the UTCPD shall assert the Transmission Discarded bit(TXDCIS (UTCPD_IS[5])). The UTCPD shall assert both TXOKIS (UTCPD_IS[6]) and TXFALIS (UTCPD_IS[4]) after it completes the sending of a Hard Reset.

UTCPD TX HEADER Register (UTCPD_TXHEAD)

Register	Offset	R/W	Description	Reset Value
UTCPD_TXHEAD	UTCPD_BA+0x98	R/W	UTCPD TX Header Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXHEAD1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXHEAD0							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**TXHEAD1**	Transmit Header Byte 1
[7:0]	**TXHEAD0**	Transmit Header Byte 0

UTCPD Transmit Data0~6 Register (UTCPD_TXDAx)

Register	Offset	R/W	Description	Reset Value
UTCPD_TXDA0	UTCPD_BA+0xA0	R/W	UTCPD Transmit Data0 Register	0x0000_0000
UTCPD_TXDA1	UTCPD_BA+0xA4	R/W	UTCPD Transmit Data1 Register	0x0000_0000
UTCPD_TXDA2	UTCPD_BA+0xA8	R/W	UTCPD Transmit Data2 Register	0x0000_0000
UTCPD_TXDA3	UTCPD_BA+0xAC	R/W	UTCPD Transmit Data3 Register	0x0000_0000
UTCPD_TXDA4	UTCPD_BA+0xB0	R/W	UTCPD Transmit Data4 Register	0x0000_0000
UTCPD_TXDA5	UTCPD_BA+0xB4	R/W	UTCPD Transmit Data5 Register	0x0000_0000
UTCPD_TXDA6	UTCPD_BA+0xB8	R/W	UTCPD Transmit Data6 Register	0x0000_0000

31	30	29	28	27	26	25	24
TXDAB3							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TXDAB2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXDAB1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAB0							

Bits	Description	
[31:24]	**TXDAB3**	USB PD Transmit Datax byte 3
[23:16]	**TXDAB2**	USB PD Transmit Datax byte 2
[15:8]	**TXDAB1**	USB PD Transmit Datax byte 1
[7:0]	**TXDAB0**	USB PD Transmit Datax byte 0

UTCPD V_{BUS} Voltage Register (UTCPD_VBVOL)

Register	Offset	R/W	Description	Reset Value
UTCPD_VBVOL	UTCPD_BA+0xC0	R/W	UTCPD V_{BUS} Voltage Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**VBSCALE**		**VBVOL**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VBVOL							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:10]	**VBSCALE**	**V_{BUS} Scale Factor** 00 = V_{BUS} measurement not scaled. 01 = V_{BUS} measurement divided by 2. 10 = V_{BUS} measurement divided by 4. 11 = reserved.
[9:0]	**VBVOL**	**V_{BUS} Voltage Measurement** 10-bit measurement of (V_{BUS} / Scale Factor). It is read only. CPU multiplies this value by the scale factor to obtain the voltage measurement. Voltages greater than or equal to 4V shall meet +/-2% absolute value or +/- 50mV, whichever is greater. The LSB is 25mV.

UTCPD V_{BUS} SINK Disconnect Threshold Register (UTCPD_SKVBDCTH)

Register	Offset	R/W	Description	Reset Value
UTCPD_SKVBDCTH	UTCPD_BA+0xC4	R/W	UTCPD V_{BUS} Sink disconnect threshold Register	0x0000_008C

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SKVBDCTH**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SKVBDCTH							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**SKVBDCTH**	**Sink V_{BUS} Disconnect Threshold** 10-bit for voltage threshold with 25mV LSB. (Default 3.5V) A value of zero disables this threshold.

UTCPD V_{BUS} Stop Discharge Threshold Register (UTCPD_SPDGTH)

Register	Offset	R/W	Description	Reset Value
UTCPD_SPDGTH	UTCPD_BA+0xC8	R/W	UTCPD V_{BUS} Stop Discharge threshold Register	0x0000_0020

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SPDGTH**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPDGTH							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**SPDGTH**	**V_{BUS} Stop Force Discharge Threshold** This value is used as a threshold hold for force discharge fail. The default is 0.8V. The CPU writes to this register to set the threshold at which a Source shall stop the Forced Discharge circuit when FDGEN (UTCPD_PWRCTL[2]) is 1.

UTCPD V_{BUS} voltage high alarm threshold Register (UTCPD_VBAMH)

Register	Offset	R/W	Description	Reset Value
UTCPD_VBAMH	UTCPD_BA+0xCC	R/W	UTCPD V_{BUS} voltage high alarm threshold Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**VBAMH**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VBAMH							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**VBAMH**	V_{BUS} voltage high Alarm threshold register

UTCPD V_{BUS} voltage low alarm threshold Regiser (UTCPD_VBAML)

Register	Offset	R/W	Description	Reset Value
UTCPD_VBAML	UTCPD_BA+0xD0	R/W	UTCPD V_{BUS} voltage low alarm threshold Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**VBAML**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VBAML							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**VBAML**	V_{BUS} voltage low Alarm threshold register

UTCPD Vendor Defined Interrupt Status Register (UTCPD_VNDIS)

Register	Offset	R/W	Description	Reset Value
UTCPD_VNDIS	UTCPD_BA+0xD4	R/W	UTCPD Vendor defined Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		VCDGIS	Reserved	CRCERRIS	Reserved	TXFRSIS	RXFRSIS

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**VCDGIS**	VCONN Discharge Interrupt Status
[4]	**Reserved**	Reserved.
[3]	**CRCERRIS**	**CRC Error Interrupt Status** 0 = Cleared. 1 = The event has been detected.
[2]	**Reserved**	Reserved.
[1]	**TXFRSIS**	**Fast Role Swap TX Interrupt Status** 0 = Cleared. 1 = The event has been detected.
[0]	**RXFRSIS**	**Fast Role Swap RX Interrupt Status** 0 = Cleared. 1 = The event has been detected.

UTCPD Vendor Defined Interrupt Enable Register (UTCPD_VNDIE)

Register	Offset	R/W	Description	Reset Value
UTCPD_VNDIE	UTCPD_BA+0xD8	R/W	UTCPD Vendor defined Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**VCDGIE**	**Reserved**	**CRCERRIE**	**Reserved**	**TXFRSIE**	**RXFRSIE**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**VCDGIE**	**VCONN Discharge Interrupt Enable** 0 = VCOON discharge interrupt Disabled. 1 = VCOON discharge interrupt Enabled.
[4]	**Reserved**	Reserved.
[3]	**CRCERRIE**	**CRC Error Interrupt Enable** 0 = CRC error interrupt Disabled. 1 = CRC error interrupt Enabled.
[2]	**Reserved**	Reserved.
[1]	**TXFRSIE**	**Fast Role Swap TX Interrupt Enable** 0 = Fast role swap TX interrupt Disabled. 1 = Fast role swap TX interrupt Enabled.
[0]	**RXFRSIE**	**Fast Role Swap RX Interrupt Enable** 0 = Fast role swap RX interrupt Disabled. 1 = Fast role swap RX interrupt Enabled.

UTCPD Mux Selection Register (UTCPD_MUXSEL)

Register	Offset	R/W	Description	Reset Value
UTCPD_MUXSEL	UTCPD_BA+0xDC	R/W	UTCPD Mux Select Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CC2FRSS	CC2VCENS	Reserved		CC1FRSS	CC1VCENS
23	22	21	20	19	18	17	16
Reserved		ADCSELVC					ADCSELVB
15	14	13	12	11	10	9	8
ADCSELVB				Reserved	FVBS		
7	6	5	4	3	2	1	0
VCOCS				VBOCS			

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**CC2FRSS**	**CC2 VCONN Fast Role Swap Select** 0 = Merged CC1 and CC2 Fast Role Swap signal. 1 = CC2 VCONN discharge signal.
[28]	**CC2VCENS**	**CC2 VCONN Enable Select** 0 = Merged CC1 and CC2 VCONN enable signal. 1 = CC2 VCONN enable signal.
[27:26]	**Reserved**	Reserved.
[25]	**CC1FRSS**	**CC1 VCONN Fast Role Swap Select** 0 = Merged CC1 and CC2 VCONN Fast Role Swap signal. 1 = CC1 VCONN Fast Role Swap signal.
[24]	**CC1VCENS**	**CC1 VCONN Enable Select** 0 = Merged CC1 and CC2 VCONN enable signal. 1 = CC1 VCONN enable signal.
[23:22]	**Reserved**	Reserved.
[21:17]	**ADCSELVC**	**ADC Channel Select for VCONN** ADC result will be latched into VCVOL register once ADCCSELVC matches ADC_CHSEL and ADC finishes.
[16:12]	**ADCSELVB**	**ADC Channel Select for V_{BUS}** ADC result will be latched into VBVOL register once ADCCSELVB matches ADC_CHSEL and ADC finishes.
[11]	**Reserved**	Reserved.

[10:8]	**FVBS**	**Force off V_{BUS} Select** 000 = ENIT0. 001 = EINT1. 010 = EINT2. 011 = ENIT3. 100 = EINT4. 101 = EINT5. Others = Reserved.
[7:4]	**VCOCS**	**VCONN Overcurrent Source Select** 0000 = EINT0. 0001 = EINT1. 0010 = EINT2. 0011 = EINT3. 0100 = ADCMPF1. 0101 = ACMP0. 0110 = ACMP1. 0111 = ACMP2. Others = Reserved. **Note:** For ACMP, the VCOCPL (PINPL[10]) shall be set as 1.
[3:0]	**VBOCS**	**V_{BUS} Overcurrent Source Select** 0000 = EINT0. 0001 = EINT1. 0010 = EINT2. 0011 = EINT3. 0100 = ADCMPF0. 0101 = ACMP0. 0110 = ACMP1. 0111 = ACMP2. Others = Reserved. **Note:** For ACMP, the VBOCPL (PINPL[5]) shall be set as 1.

UTCPD VCONN Discharge Control (UTCPD_VCDGC)

Register	Offset	R/W	Description	Reset Value
UTCPD_VCDGCTL	UTCPD_BA+0xE0	R/W	UTCPD VCONN Discharge Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**VCDGEN**	**VCDGDTEN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[1]	**VCDGEN**	**VCONN Discharge Enable** 0 = VCONN Discharge Disabled. 1 = VCONN Discharge Enabled.
[0]	**VCDGDTEN**	**VCONN Discharge Detect Enable** 0 = VCONN Discharge Detect Disabled. 1 = VCONN Discharge Detect Enabled.

UTCPD PHY Slew Rate Control Register (UTCPD_PHYSLEW)

Register	Offset	R/W	Description	Reset Value
UTCPD_PHYSLEW	UTCPD_BA+0xE4	R/W	UTCPD PHY Slew Rate Control Register	0x0000_0024

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	TXFTRIM			Reserved	TXRTRIM		

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**IDLESEL**	**Bus Idle Select** 00 = Digital bus no transaction over 12us. 01 = Digital and analog bus no transaction over 12us. 10 = Analog bus no transaction over 12us. 11 = CC pins no activity. Only after bus idle, the de-bounce circuit starts.
[7]	**Reserved**	Reserved.
[6:4]	**TXFTRIM**	**TX Trim Falling Slew Rate** It is for BMC eye diagram. Trim PHY TX falling slew rate. (from ROMMAP)
[3]	**Reserved**	Reserved.
[2:0]	**TXRTRIM**	**TX Trim Rising Slew Rate** It is for BMC eye diagram. Trim PHY TX Rising slew rate (from ROMMAP)

UTCPD PHY Auto Discharge Time Register (UTCPD_ADGTM)

Register	Offset	R/W	Description	Reset Value
UTCPD_ADGTM	UTCPD_BA+0xE8	R/W	UTCPD Auto Discharge Time Register	0x0000_0064

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADGTM							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**ADGTM**	**Auto Discharge Time** Default Time = 31.25us x 16 x 100 (0x16) = 49.9ms.

UTCPD Auto Discharge VSAFE0V Register (UTCPD_ADG_0V)

Register	Offset	R/W	Description	Reset Value
UTCPD_VSAFE0V	UTCPD_BA+0xEC	R/W	UTCPD Auto Discharge VSAFE0V Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**VSAFE0V**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VSAFE0V							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**VSAFE0V**	Set the vSafe0V voltage level.

UTCPD FRS VSAFE5V Register (UTCPD_VSAFE5V)

Register	Offset	R/W	Description	Reset Value
UTCPD_VSAFE5V	UTCPD_BA+0xF0	R/W	UTCPD VSAFE5V Register	0x0000_00C8

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**VSAFE5V**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VSAFE5V							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**VSAFE5V**	**Set the vSafe5V Voltage Level** For fast role swap voltage comparison.

UTCPD DRP Toggle Ratio Register (UTCPD_RATIO)

Register	Offset	R/W	Description	Reset Value
UTCPD_RATIO	UTCPD_BA+0xF4	R/W	UTCPD DRP Toggle Ratio Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**ADCAVG**	**Reserved**		**VBSEL**	**DRPRATIO**		

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**ADCAVG**	**ADC Moving Average Enable** If enabling this bit, average the ADC value for 4 times.
[5:4]	**Reserved**	Reserved.
[3]	**VBSEL**	**VBUS_VOLTAGE_SEL** Select comparing value (vbus_voltage_s) during vbus discharge determines to stop discharging or not. 0 = V_{BUS} voltage value from adc only when VBMONI (UTCPD_PWRCT[6]) is 1. 1 = V_{BUS} voltage value from ADC.
[2:0]	**DRPRATIO**	**The percent of time that a DRP shall advertise source & sink during tDRP** 000 = 50:50 (Sink: Source). 010 = 30:70 (Sink: Source). 001 = 40:60 (Sink: Source). 101 = 60:40 (Sink: Source). 110 =70:30 (Sink: Source). 111 = Reserved.

UTCPD Inter-Frame Time Register (UTCPD_INTFRAME)

Register	Offset	R/W	Description	Reset Value
UTCPD_INTFRAME	UTCPD_BA+0xF8	R/W	UTCPD Inter-Frame Time Register	0x0000_0019

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
INTFRAME							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**INTFRAME**	**USB PD Inter Frame Gap** Each unit time: 83ns (12 MHz RC) Time = 83ns x 16 x Inter frame gap time. Example = 83ns x 16 x 25(8'b00011001) = 33.2us.

UTCPD V_{BUS} Over Voltage Threshold Register (UTCPD_VBOVTH)

Register	Offset	R/W	Description	Reset Value
UTCPD_VBOVTH	UTCPD_BA+0xFC	R/W	UTCPD V_{BUS} Over Voltage Threshold Register	0x0000_03FF

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**VBOVTH**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VBOVTH							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**VBOVTH**	**V_{BUS} Over Voltage Threshold** This value defines the V_{BUS} over voltage threshold 10-bit for voltage threshold with 25mV LSB.

UTCPD Vendor Initial Register (UTCPD_VNDINIT)

Register	Offset	R/W	Description	Reset Value
UTCPD_VNDINIT	UTCPD_BA+0x100	R/W	UTCPD Vendor Initial Register	0x0000_0064

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**DVCAPDEF**			**Reserved**			

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:4]	**DVCAPDEF**	**Device Capability Default Setting** Write the Register will update the value to Device_Cap.RoleSupport, ROLE_CONTROL and MESSAGE_HEADER_INFO register, please refer the following table. (DevCap_DEF Table)
[3:0]	**Reserved**	Reserved.

DevCap_DEF	DEVICE_CAPBABILITY.RoleSupported	ROLE_CONTROL	MESSAGE_HEADER_INFO
000	Source or Sink(000b)	0Ah	02h
001	Source(001b)	05h	0Bh
010	Sink(010b)	0Ah	02h
011	Source with Accessory (011b)	0Ah	02h
100	DRP(100b)	0Ah	02h
101	Source, Sink, DRP Applies to SOP Device (101b)	0Ah	02h
110	Source, Sink, DRP (110b)	0Ah	02h
111	Reserved (111b)	0Ah	02h

UTCPD BMC TX Bit Period Register (UTCPD_BMCTXBP)

Register	Offset	R/W	Description	Reset Value
UTCPD_BMCTXBP	UTCPD_BA+0x104	R/W	UTCPD BMC TX Bit Period Register	0x0000_0027

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BMCTXBP							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**BMCTXBP**	**BMC Tx Bit Period** It's for BMC eye diagram. Example 12 MHz = 83.33ns. 83.33ns x (39 + 1) = 3.33us.

UTCPD BMC TX Duty Register (UTCPD_BMCTXDU)

Register	Offset	R/W	Description	Reset Value
UTCPD_BMCTXDU	UTCPD_BA+0x108	R/W	UTCPD BMC TX Duty Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DUOFFS1	DUOFFS2						

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**DUOFFS1**	**BMC Tx Duty Offset Parameter 1** It's for BMC eye diagram. 0 = Increase duty offset?. (+) 1 = Decrease duty offset?. (-)
[6:0]	**DUOFFS2**	**BMC Tx Duty Offset Parameter 2** It's for BMC eye diagram. Offset count value. Example 12 MHz = 83.33ns.

UTCPD VCONN Present Voltage Register (UTCPD_VCPSVOL)

Register	Offset	R/W	Description	Reset Value
UTCPD_VCPSVOL	UTCPD_BA+0x10C	R/W	VCONN Present Voltage Register	0x0000_0060

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						VCPSVOL	
7	6	5	4	3	2	1	0
VCPSVOL							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**VCPSVOL**	**VCONN Present voltage** Detect Voltage = 60h * 0.025V = 2.42V.

UTCPD VCONN Under Voltage Register (UTCPD_VCUV)

Register	Offset	R/W	Description	Reset Value
UTCPD_VCUV	UTCPD_BA+0x110	R/W	VCONN Under Voltage Register	0x0000_0020

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**VCUV**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VCUV							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**VCUV**	**VCONN under voltage comparator** Detect Voltage = 20h * 0.025V = 0.8V.

UTCPD BMC Slice Control Register (UTCPD_BMCSLICE)

Register	Offset	R/W	Description	Reset Value
UTCPD_BMCSLICE	UTCPD_BA+0x118	R/W	UTCPD BMC SLICE Control Register	0x0824_8423

31	30	29	28	27	26	25	24
VTRIM				Reserved	TRIMVBUS20		
23	22	21	20	19	18	17	16
Reserved	TRIMVBUS10			Reserved	TRIMV1P1		
15	14	13	12	11	10	9	8
TRIMRP				Reserved	TRIMRD		
7	6	5	4	3	2	1	0
Reserved	SLICEM			SLICEH		SLICEL	

Bits	Description	
[31:28]	**VTRIM**	Reference Voltage TRIM setting for TCPD PHY macro
[27]	**Reserved**	Reserved.
[26:24]	**TRIMVBUS20**	**TRIMVBUS20** Power good level trimming option of V_{BUS} when the VBSCALE[1] in UTCPD_VBVOL[10] = 1.
[23]	**Reserved**	Reserved.
[22:20]	**TRIMVBUS10**	**TRIMVBUS10** Power good level trimming option of V_{BUS} when the VBSCALE[1] in UTCPD_VBVOL[10] = 0.
[19]	**Reserved**	Reserved.
[18:16]	**TRIMV1P1**	TRIMV1P1
[15:12]	**TRIMRP**	TRIMRP
[11]	**Reserved**	Reserved.
[10:8]	**TRIMRD**	TRIMRD
[7]	**Reserved**	Reserved.
[6:4]	**SLICEM**	**TX Slice Middle Level Control** Middle level slice control (The LSB is 2mV.) 000 = 0.48V. 100 = 0.56V (Default). 111 = 0.62V. Others = Reserved.

[3:2]	**SLICEH**	**TX Slice High Level Control** High level slice control (The LSB is 2mV.) 00 = 0.84V (Default). 01 = 0.86V. 10 = 0.88V. 11 = 0.9V. Others = Reserved.
[1:0]	**SLICEL**	**TX Slice Low Level Control** Low level slice control (The LSB is 2mV.) 00 = 0.18V. 01 = 0.2V. 10 = 0.22V. 11 = 0.24V (Default). Others = Reserved.

UTCPD PHY Power Control Register (UTCPD_PHYCTL)

Register	Offset	R/W	Description	Reset Value
UTCPD_PHYCTL	UTCPD_BA+0x11C	R/W	UTCPD PHY Power Control Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						DBCTL	PHYPWR

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**DBCTL**	**Dead Battery Control** 0 = Dead Battery circuit control internal Rd/Rp. 1 = Role Control Register control internal Rd/Rp.
[0]	**PHYPWR**	**Analog PHY Power** 0 = Power down PHY. 1 = Enable PHY. Analog PHY power default is off before UTCPD clock available. Once UTCPD clock is on, the analog PHY power will be on as well.

UTCPD Fast Swap RX Control Register (UTCPD_FRSRXCTL)

Register	Offset	R/W	Description	Reset Value
UTCPD_FRSRXCTL	UTCPD_BA+0x120	R/W	UTCPD CC Fast Swap RX Control Register	0x0000_0080

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				FRSRXEN	FRSDVVB	Reserved	FRSTX

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**FRSRXEN**	**CC receive fast swap Rx Enable** 0 = CC receive fast swap Rx Disabled. 1 = CC receive fast swap Rx Enabled.
[2]	**FRSDVVB**	**CC receive fast swap and auto drive Source V_{BUS}** 0 = CC receive fast swap and auto drive Source V_{BUS} Disabled. 1 = CC receive fast swap and auto drive Source V_{BUS} Enabled.
[1]	**Reserved**	Reserved.
[0]	**FRSTX**	**CC Transmitter Fast Swap Signal** Pulse width is 85us.

UTCPD VCONN Voltage Measure Register (UTCPD_VCVOL)

Register	Offset	R/W	Description	Reset Value
UTCPD_VCVOL	UTCPD_BA+0x124	R	UTCPD VCONN Voltage Measurement Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						VCVOL	
7	6	5	4	3	2	1	0
VCVOL							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**VCVOL**	**VCONN Voltage Measurement** The LSB is 25Mv.

UTCPD_CLKINFO Register (UTCPD_CLKINFO)

Register	Offset	R/W	Description	Reset Value
UTCPD_CLKINFO	UTCPD_BA+0x300	R/W	UTCPD Clock Information Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**WKEN**	**Reserved**			**ReadyFlag**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**WKEN**	**Wakeup Enable** 0 = UTCPD wakeup function Disabled. 1 = UTCPD wakeup function Enabled.
[3:1]	**Reserved**	Reserved.
[0]	**ReadyFlag**	**RC32K Domain Ready Flag Check if the Register Value is Loaded to RC32K Domain by Reading This Flag** 0 = The RC32K signal is not ready. 1 = The RC32K signal is ready.

UTCPD Preamble Detect Register (UTCPD_PREDET)

Register	Offset	R/W	Description	Reset Value
UTCPD_PREDET	UTCPD_BA+0x320	R/W	UTCPD Preamble Detect Register	0x321D_4027

31	30	29	28	27	26	25	24
Reserved		MAXWBCN					
23	22	21	20	19	18	17	16
Reserved		MINWBCN					
15	14	13	12	11	10	9	8
PSDB		DGACT	DGEN	Reserved		BNW	
7	6	5	4	3	2	1	0
Reserved	BNICHK			RNICHK			

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**MAXWBCN**	**Maximun Width Bit Counter Number** Define the maximun width of one bit counter number among preamble period. If the counter of one bit width is over this value, this case path will be ignore to juge the bit counter number. **Note:** The default value is the bit clock counter of minimun bit rate frequency (270K) minus 10%.
[23:22]	**Reserved**	Reserved.
[21:16]	**MINWBCN**	**Minimum Width Bit Counter Number** Define the minuimun width of one bit counter number among preamble period. If the counter of one bit width is less than this value, this case path will be ignore to juge the bit counter number. **Note:** The default value is the bit clock counter of maxmun bit rate frequency (330K) plus 10%.
[15:14]	**PSDB**	**Preamble Start Detect Bit** 00 = Round start after first 3 bits be detected. 01 = Round start after first 5 bits be detected. 10 = Round start after first 6 bits be detected. 11 = Round start after first 7 bits be detected.
[13]	**DGACT**	**Analog Macro Activity Signal Source from Deglitch Circuit** 0 = Analog macro activity signal source from deglitch circuit Disabled. 1 = Analog macro activity signal source from deglitch circuit Enabled.
[12]	**DGEN**	**Analog Macro Deglitch circuit Enable** 0 = Analog macro deglitch circuit Disabled. 1 = Analog macro 3 slices comparator deglitch circuit Enabled.
[11:10]	**Reserved**	Reserved.

[9:8]	**BNW**	**Bit Number Wait for next Round** 00 = The wait bit number for next round is 1 bits. 01 = The wait bit number for next round is 2 bits. 10 = The wait bit number for next round is 3 bits. 11 = The wait bit number for next round is 4 bits.
[7]	**Reserved**	Reserved.
[6:4]	**BNICHK**	**Bit Number for Noise Immunity Check each Round** 000 = The detected bit number of one round equal to 3 bits. 001 = The detected bit number of one round equal to 4 bits. 010 = The detected bit number of one round equal to 5 bits. 011 = The detected bit number of one round equal to 6 bits. 100 = The detected bit number of one round equal to 7 bits. 101 = The detected bit number of one round equal to 8 bits. Others = Reserved.
[3:0]	**RNICHK**	**Round of Noise Immunity Check** 0000 = Noise immunity optimize Disabled. 0001 = The detect is one round. 0010 = The detect is two rounds. 0011 = The detect is three rounds. 0100 = The detect is four rounds. 0101 = The detect is five rounds. 0110 = The detect is six rounds. 0111 = The detect is seven rounds. 1000 = The detect is eight round. 1001 = The detect is nini rounds. 1010 = The detect is ten rounds. 1011 = The detect is eleven rounds. 1100 = The detect is twelve rounds. 1101 = The detect is thirteen rounds. 1110 = The detect is fourteen rounds. 1111 = The detect is fifteen rounds. **Note:** RPCHK = 3'b000, Use the original detection solution. (case path 0).

UTCPD Noise Immunity Record Register (UTCPD_NIRR)

Register	Offset	R/W	Description	Reset Value
UTCPD_NIRR	UTCPD_BA+0x324	R	UTCPD Noise Immunity Record Register	0x0000_3F00

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CSEL		**NBW**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**WBW**					

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:14]	**CSEL**	**Case Selection (Read Only)** The field indicates the selected case condition. 00 = The case 0. 01 = The case 1. 10 = The case 2. 11 = Reserved.
[13:8]	**NBW**	**Narrow Bit Width (Read Only)** The field indicates the narrow bit width (based on 12M).
[7:6]	**Reserved**	Reserved.
[5:0]	**WBW**	**Wide Bit Width (Read Only)** The field indicates the wide bit width (based on 12M).

UTCPD Case 0 Path Record Register (UTCPD_C0PRR)

Register	Offset	R/W	Description	Reset Value
UTCPD_C0PRR	UTCPD_BA+0x328	R	UTCPD Case 0 Path Record Register	0x0000_3F00

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					C0BRT		
15	14	13	12	11	10	9	8
C0PORF	Reserved	C0PNBPN					
7	6	5	4	3	2	1	0
Reserved		C0PWBPN					

Bits	Description	
[31:16]	**Reserved**	Reserved.
[18:16]	**C0BRT**	**Case 0 Path Best Round Times** The field indicates the best round times of bit width detection. **Note:** It is cleared in next preamble start or write 1.
[15]	**C0PORF**	**Case 0 Path Out of Range Flag** 0 = Case 0 isn't out of range. 1 = Case 0 is out of range. **Note:** It is cleared in next preamble start or write 1.
[14]	**Reserved**	Reserved.
[13:8]	**C0PNBPN**	**Case 0 Path Narrow Bit Period Number (Read Only)** The field indicates the final narrow bit clcok number based 12M. If the RNICHK (UTCPD_PREDET[2:0]) is 3'b000, the default value is 0x3F. **Note:** It is cleared in next preamble start or write 1.
[7:6]	**Reserved**	Reserved.
[5:0]	**C0PWBPN**	**Case 0 Path Wide Bit Period Number (Read Only)** The field indicates the final width bit clock number based 12M clock. **Note:** It is cleared in next preamble start or write 1.

UTCPD Case 1 Path Record Register (UTCPD_C1PRR)

Register	Offset	R/W	Description	Reset Value
UTCPD_C1PRR	UTCPD_BA+0x32C	R	UTCPD Case 1 Path Record Register	0x0000_3F00

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					C1BRT		
15	14	13	12	11	10	9	8
C1PORF	Reserved	C1PNBPN					
7	6	5	4	3	2	1	0
Reserved		C1PWBPN					

Bits	Description	
[31:16]	**Reserved**	Reserved.
[18:16]	**C1BRT**	**Case 1 Path Best Round bnTimes** The field indicates the best round times of bit width detection. **Note:** It is cleared in next preamble start or write 1.
[15]	**C1PORF**	**Case 1 Path Out of Range Flag** 0 = Case 1 isn't out of range. 1 = Case 1 is out of range. **Note:** It is cleared in next preamble start or write 1.
[14]	**Reserved**	Reserved.
[13:8]	**C1PNBPN**	**Case 1 Path Narrow Bit Period Number (Read Only)** The field indicates the final narrow bit clcok number based 12M. **Note:** It is cleared in next preamble start or write 1.
[7:6]	**Reserved**	Reserved.
[5:0]	**C1PWBPN**	**Case 1 Path Wide Bit Period Number (Read Only)** The field indicates the final width bit clock number based 12M clock. **Note:** It is cleared in next preamble start or write 1.

UTCPD Case 2 Path Record Register (UTCPD_C2PRR)

Register	Offset	R/W	Description	Reset Value
UTCPD_C2PRR	UTCPD_BA+0x330	R	UTCPD Case 2 Path Record Register	0x0000_3F00

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					C2BRT		
15	14	13	12	11	10	9	8
C2PORF	Reserved	C2PNBPN					
7	6	5	4	3	2	1	0
Reserved		C2PWBPN					

Bits	Description	
[31:16]	**Reserved**	Reserved.
[18:16]	**C2BRT**	**Case 2 Path Best Round Times** The field indicates the best round times of bit width detection. **Note:** It is cleared in next preamble start or write 1.
[15]	**C2PORF**	**Case 2 Path Out of Range Flag** 0 = Case 2 isn't out of range. 1 = Case 2 is out of range. **Note:** It is cleared in next preamble start or write 1.
[14]	**Reserved**	Reserved.
[13:8]	**C2PNBPN**	**Case 2 Path Narrow Bit Period Number (Read Only)** The field indicates the final narrow bit clcok number based 12M. **Note:** It is cleared in next preamble start or write 1.
[7:6]	**Reserved**	Reserved.
[5:0]	**C2PWBPN**	**Case 2 Path Wide Bit Period Number (Read Only)** The field indicates the final width bit clock number based 12M clock. **Note:** It is cleared in next preamble start or write 1.

SLICEUTCPD New BMC Slice Control Register (UTCPD_NBMCSLICE)

Register	Offset	R/W	Description	Reset Value
UTCPD_NBMCSLICE	UTCPD_BA+0x334	R/W	UTCPD New BMC SLICE Control Register	0x0000_0422

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
NBMCEN	Reserved				NSLICEM		
7	6	5	4	3	2	1	0
Reserved	NSLICEH			Reserved	NSLICEL		

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**NBCMEN**	**New BMC Trim Value Control Enable** 0 = The trim value of BMC is compatiable with MAD025. 1 = New BMC trim value in this regsiter bit field.
[14:11]	**Reserved**	Reserved.
[10:8]	**NSLICEM**	**New TX Slice Middle Level Control** Middle level slice control (The LSB is 20mV.) 000 = 0.48V. 001 = 0.50V. 010 = 0.52V. 011 = 0.54V. 100 = 0.56V (Default). 101 = 0.58V. 110 = 0.60V. 111 = 0.62V.
[7]	**Reserved**	Reserved.
[6:4]	**NSLICEH**	**New TX Slice High Level Control** High level slice control (The LSB is 20mV.) 000 = 0.80V. 001 = 0.82V. 010 = 0.84V (Default). 011 = 0.86V. 100 = 0.88V. 101 = 0.90V. 110 = 0.92V. 111 = 0.94V.

[4]	**Reserved**	Reserved.
[2:0]	**NSLICEL**	**New TX Slice Low Level Control** Low level slice control (The LSB is 20mV.) 000 = 0.14V. 001 = 0.16V. 010 = 0.18V (Default). 011 = 0.20V. 100 = 0.22V. 101 = 0.24V. 110 = 0.26V. 111 = 0.28V.

UTCPD BMC Slice Debug Register (UTCPD_BMCDBGR)

Register	Offset	R/W	Description	Reset Value
UTCPD_BMCDBGR	UTCPD_BA+0x338	R	UTCPD BMC SLICE Value Debug Register	0x0000_0425

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					SLICEM		
7	6	5	4	3	2	1	0
Reserved	SLICEH			Reserved	SLICEL		

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**SLICEM**	**TX Slice Middle Level Control (Read Only)** Middle level slice control (The LSB is 20mV.) 000 = 0.48V. 001 = 0.50V. 010 = 0.52V. 011 = 0.54V. 100 = 0.56V (Default). 101 = 0.58V. 110 = 0.60V. 111 = 0.62V. **Note:** For FPGA Debug and Read Only.
[7]	**Reserved**	Reserved.
[6:4]	**SLICEH**	**TX Slice High Level Control (Read Only)** High level slice control (The LSB is 20mV.) 000 = 0.80V. 001 = 0.82V. 010 = 0.84V (Default). 011 = 0.86V. 100 = 0.88V. 101 = 0.90V. 110 = 0.92V. 111 = 0.94V. **Note:** For FPGA Debug and Read Only.
[4]	**Reserved**	Reserved.

[2:0]	**SLICEL**	**TX Slice Low Level Control (Read Only)** Low level slice control (The LSB is 20mV.) 000 = 0.14V. 001 = 0.16V. 010 = 0.18V (Default). 011 = 0.20V. 100 = 0.22V. 101 = 0.24V. 110 = 0.26V. 111 = 0.28V. **Note:** For FPGA Debug and Read Only.

6.36 Controller Area Network with Flexible DataRate (CAN FD)

6.36.1 Overview

The CAN FD controller performs communication according to ISO 11898-1:2015 and need be connected to additional transceiver hardware for the physical layer.

The CAN FD controller consists of one CAN Core, Memory access control and arbiter, Tx Handler, Rx Handler, a shared Message RAM memory and a 32-bit AHB interface for control and configuration registers.

The message storage is intended to be a single-ported Message RAM outside of the CAN Core module. It is connected to the CAN Core via the memory control interface. The Message RAM implements filters, receive FIFOs, transmit event FIFOs and transmit FIFOs.

All functions concerning the handling of messages are implemented by the Rx Handler and the Tx Handler. The Rx Handler manages message acceptance filtering, the transfer of received messages from the CAN Core to the Message RAM as well as providing received message status information. The Tx Handler is responsible for the transfer of transmit messages from the Message RAM to the CAN Core as well as providing transmitted status information.

The controller's clock domain concept allows the separation among CAN Core clock and the AHB clock (HCLK). In order to achieve a stable function of the CAN FD controller, the AHB clock must always be faster than or equal to the CAN core clock.

6.36.2 Features

- Conform with CAN protocol version 2.0 part A, B and ISO 11898-1: 2015
- CAN FD with up to 64 data bytes supported
- CAN Error Logging
- AUTOSAR support
- SAE J1939 support
- Improved acceptance filtering
- Two configurable Receive FIFOs
- Separate signaling on reception of High Priority Messages
- Up to 64/32 Receive buffers/Transmit Buffers
- Configurable Transmit FIFO, Transmit Queue, Transmit Event FIFO
- Direct Message RAM access for CPU
- Programmable loop-back test mode
- Maskable module interrupts
- Two clock domains (CAN Core clock and AHB clock)
- Power-down support

6.36.3 Block Diagram

The controller interfaces with the AMBA AHB bus. Figure 6.36-1 shows the block diagram of the controller.

CAN Core

CAN Protocol Controller and Rx/Tx shift register. Handle all 2.0 part A, B and ISO 11898-1:2015 protocol functions. Supports 11-bit and 29-bit identifiers

Sync

Synchronizes signals from the AHB clock domain to the CAN Core clock domain and vice versa

Control and Configuration Registers

All registers and related configuration used to control and to configure the CAN Core.

Interrupt & power down control

The CAN FD controller provides two interrupt lines CANFDx_INT0 and CANFDx_INT1. By programming EINT0 and EINT1 bits in CANFD_ILE register, the interrupt lines can be enabled or disabled separately.

The CAN FD controller can be set into power down controlled by both CANFDnREQ (CLK_STOPREQ[1:0]), and CANFDnACK (CLK_STOPACK[1:0]) where n= 0,1. The specific operation flow is the same as definition of CSR (CANFD_CCCR[4]) and CSA(CANFD_CCCR[3]).

Memory access control and Arbiter

The arbiter is responsible for the Message RAM access authority between the CPU and the interface of Tx/Rx Handler to an external 32-bit Message RAM.

Message RAM (SRAM)

Stores message objects and Identifier masks. The maximum Message RAM size is 1536 words.

Tx/Rx Message Handler

State Machine that controls the data transfer between the Rx/Tx Shift Register of the CAN Core and the Message RAM as well as the generation of interrupts as programmed in the Control and Configuration registers.

AHB Interface

Connect the CAN FD controller to the AHB bus.

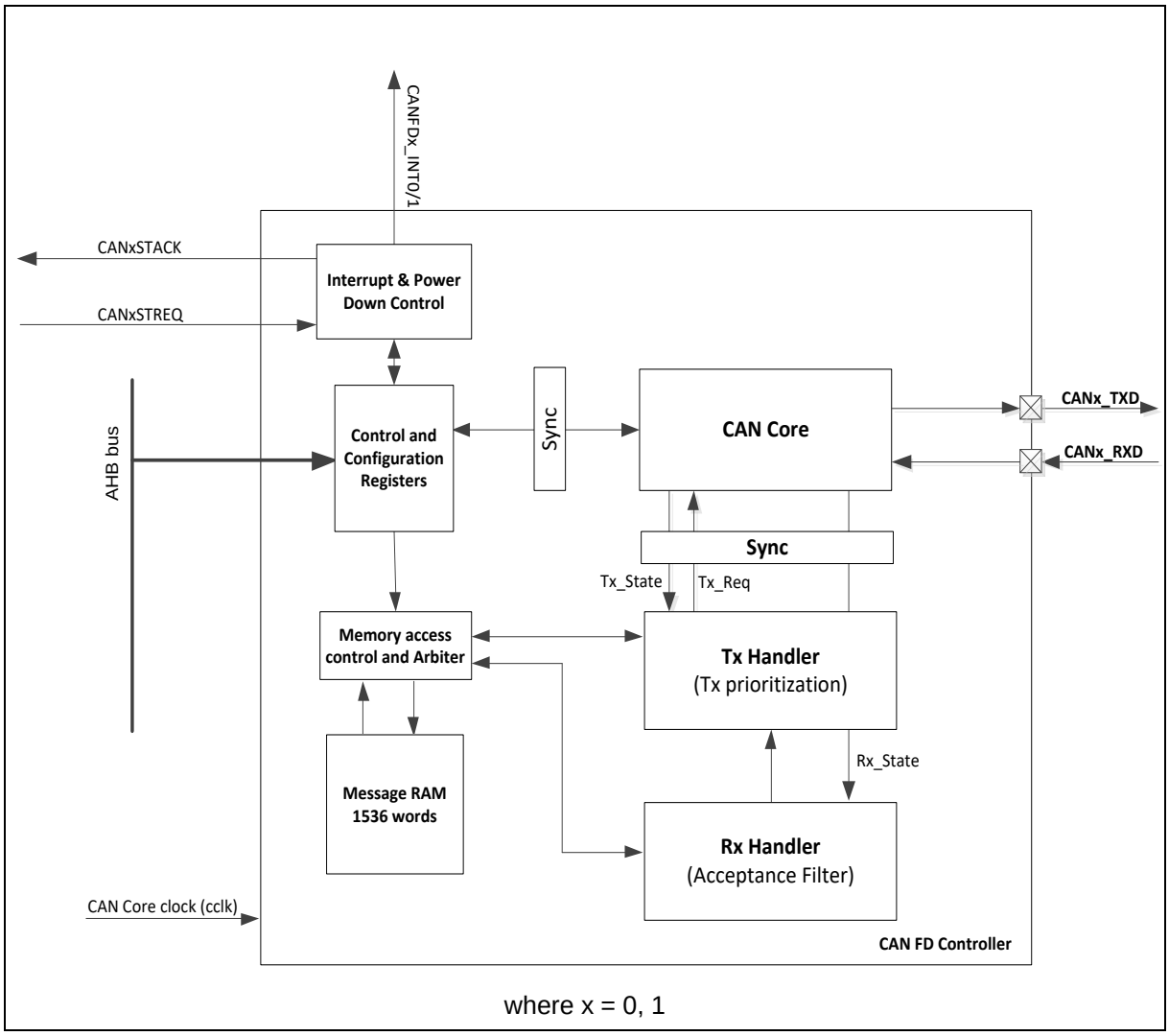


Figure 6.36-1 CAN FD Controller Block Diagram

6.36.4 Basic Configuration

6.36.4.1 CAN Basic Configuration

- Clock source Configuration
  - Enable CANFD0 controller clock (CANFD0CKEN (CLK_AHBCLK1[20])).
  - Enable CANFD1 controller clock (CANFD1CKEN (CLK_AHBCLK1[21])).
  -
- Reset Configuration
  - Reset CANFD0 controller (CANFD0RST (SYS_IPRST0[20])).
  - Reset CANFD1 controller (CANFD1RST (SYS_IPRST0[21])).

6.36.5 Functional Description

6.36.5.1 Operation Mode

Software Initialization

Software initialization is started by setting bit INIT (CANFD_CCCR[0]) bit, either by software or by a hardware reset, when an uncorrected bit error was detected in the Message RAM, or by going Bus_Off. While INIT (CANFD_CCCR[0]) bit is set, message transfer from and to the CAN bus is stopped, the status of the CAN bus output CANx_TXD pin is recessive (HIGH). The counters of the Error Management Logic EML are unchanged. Setting INIT (CANFD_CCCR[0]) bit does not change any configuration register. Resetting INIT (CANFD_CCCR[0]) bit finishes the software initialization. Afterwards the Bit Stream Processor (BSP) synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (=Bus_Idle) before it can take part in bus activities and start the message transfer.

Access to the CAN FD configuration registers is only enabled when both bits INIT (CANFD_CCCR[0]) and CCE (CANFD_CCCR[1]) are set (protected write). CCE (CANFD_CCCR[1]) can only be set/reset while INIT (CANFD_CCCR[0]) = 1. CCE (CANFD_CCCR[1]) is automatically reset when INIT (CANFD_CCCR[0]) is reset.

The following registers are reset when CCE (CANFD_CCCR[1]) is set:

- HPMS - High Priority Message Status
- RXF0S - Rx FIFO 0 Status
- RXF1S - Rx FIFO 1 Status
- TXFQS - Tx FIFO/Queue Status
- TXBRP - Tx Buffer Request Pending
- TXBTO - Tx Buffer Transmission Occurred
- TXBCF - Tx Buffer Cancellation Finished
- TXEFS - Tx Event FIFO Status

The Timeout Counter value TOC (CANFD_TOCV[15:0]) is preset to the value configured by TOP (CANFD_TOCC[31:16]) when CCE (CANFD_CCCR[1]) is set.

In addition the state machines of the Tx Handler and Rx Handler are held in idle state while CCE (CANFD_CCCR[1]) bit is set.

The following registers are only writeable while CCE (CANFD_CCCR[1]) bit is reset.

- TXBAR - Tx Buffer Add Request
- TXBCR - Tx Buffer Cancellation Request

TEST (CANFD_CCCR[7]) bit and MON (CANFD_CCCR[5]) bit can only be set by the software while both of INIT (CANFD_CCCR[0]) bit and CCE (CANFD_CCCR[1]) bit are set. Both bits may be reset at any time. DAR (CANFD_CCCR[6]) can only be set/reset while both INIT (CANFD_CCCR[0]) and CCE (CANFD_CCCR[1]) are set.

Normal Operation

Once the CAN FD controller is initialized and INIT (CANFD_CCCR[0]) is reset to zero, the CAN FD controller synchronizes itself to the CAN bus and is ready for communication.

After passing the acceptance filtering, received messages including Message ID and DLC are stored into a dedicated Rx Buffer or into Rx FIFO 0 or Rx FIFO 1.

For messages to be transmitted dedicated Tx Buffers and/or a Tx FIFO or a Tx queue can be initialized or updated. Automated transmission on reception of remote frames is not implemented.

CAN FD Operation

There are two variants in the CAN FD frame transmission:

(1). The CAN FD frame without bit rate switching.

(2). The CAN FD frame where control field, data field, and CRC field are transmitted with a higher bit rate than the beginning and the end of the frame.

The previously reserved bit in CAN frames with 11-bit identifiers and the first previously reserved bit in CAN frames with 29-bit identifiers will now be decoded as FDF bit.

If FDF is recessive, it signifies a CAN FD frame.

If FDF is dominant, it signifies a Classic CAN frame.

In a CAN FD frame, the two bits following FDF, res and BRS, decide whether the bit rate inside of this CAN FD frame is switched. A CAN FD bit rate switch is signified by res = dominant and BRS = recessive. The coding of res = recessive is reserved for future expansion of the protocol.

In case the CAN FD controller receives a frame with FDF = recessive and res = recessive, it will signal a Protocol Exception Event by setting bit PXE (CANFD_PSR[14]) bit. When Protocol Exception Handling is enabled (PXHD (CANFD_CCCR[12]) is reset), this causes the operation state to change from Receiver (ACT (CANFD_PSR[4:3]) = 2'b10) to Integrating (ACT (CANFD_PSR[4:3]) = 00) at the next sample point.

In case Protocol Exception Handling is disabled (PXHD (CANFD_CCCR[12]) is set), the CAN FD controller will treat a recessive res bit as a form error and will respond with an error frame.

CAN FD operation is enabled by programming FDOE (CANFD_CCCR[8]). In case FDOE (CANFD_CCCR[8]) = 1, transmission and reception of CAN FD frames is enabled. Transmission and reception of Classic CAN frames is always possible. Whether a CAN FD frame or a Classic CAN frame is transmitted can be configured via bit FDF in the respective Tx Buffer element. With FDOE (CANFD_CCCR[8]) = 0, received frames are interpreted as Classic CAN frames, which leads to the transmission of an error frame when receiving a CAN FD frame. When CAN FD operation is disabled, no CAN FD frames are transmitted even if bit FDF of a Tx Buffer element is set. FDOE (CANFD_CCCR[8]) and BRSE (CANFD_CCCR[9]) can only be changed while INIT (CANFD_CCCR[0]) and CCE (CANFD_CCCR[1]) are both set.

With FDOE (CANFD_CCCR[8]) = 0, the setting of bits FDF and BRS is ignored and frames are transmitted in Classic CAN format. If FDOE (CANFD_CCCR[8]) is set and BRSE (CANFD_CCCR[9]) is clear, only bit FDF of a Tx Buffer element is evaluated. If both FDOE (CANFD_CCCR[8]) and BRSE (CANFD_CCCR[9]) are set, transmission of CAN FD frames with bit rate switching is enabled. All Tx Buffer elements with bits FDF and BRS set are transmitted in CAN FD format with bit rate switching.

A mode change during CAN operation is only recommended under the following conditions:

- The failure rate in the CAN FD data phase is significant higher than in the CAN FD arbitration phase. In this case, disable the CAN FD bit rate switching option for transmissions.
- During system startup all nodes are transmitting Classic CAN messages until it is verified that they are able to communicate in CAN FD format. If this is true, all nodes switch to CAN FD operation.
- Wake-up messages in CAN Partial Networking have to be transmitted in Classic CAN format.
- End-of-line programming in case not all nodes are CAN FD capable. Non CAN FD nodes are held in Silent mode until programming has completed. Then all nodes switch back to Classic CAN communication.

In the CAN FD format, the coding of the DLC differs from the standard CAN format. The DLC codes 0 to 8 have the same coding as in standard CAN, the codes 9 to 15, in standard CAN all code a data field of 8 bytes, which are coded according to Table 6.36-1 below.

DLC	9	10	11	12	13	14	15
Number of Data Bytes	12	16	20	24	32	48	64

Table 6.36-1 DLC coding in CAN FD format

In CAN FD frames, the bit timing will be switched inside the frame, after the BRS (Bit Rate Switch) bit, if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by the Nominal Bit Timing & Prescaler Register NBTP. In the following CAN FD data phase, the data phase bit timing is used as defined by the Data Bit Timing & Prescaler Register DBTP. The bit timing is switched back from the data phase timing at the CRC delimiter or when an error is detected, whichever occurs first.

The maximum configurable bit rate in the CAN FD data phase depends on the CAN Core clock frequency (cclk). Example: with a CAN FD controller clock frequency of 20 MHz and the shortest configurable bit time of 4 time quanta (tq), the bit rate in the data phase is 5 Mbit/s.

In both data frame formats, CAN FD and CAN FD with bit rate switching, the value of the bit ESI (Error Status Indicator) is determined by the transmitter’s error state at the start of the transmission. If the transmitter is error passive, ESI is transmitted recessive, else it is transmitted dominant.

Transmitter Delay Compensation

During the data phase of a CAN FD transmission only one node is transmitting, all others are receivers. The length of the bus line has no impact. When transmitting via pin CANx_TXD the CAN FD Controller receives the transmitted data from its local CAN transceiver via pin CANx_RXD. The received data is delayed by the transceiver loop delay. In case this delay is greater than TSEG1 (time segment before sample point), a bit error is detected. In order to enable a data phase bit time that is even shorter than the transceiver loop delay, the delay compensation is introduced. Without transceiver delay compensation, the bit rate in the data phase of a CAN FD frame is limited by the transceiver loop delay.

The CAN FD Controller implements a delay compensation mechanism to compensate the CAN transceiver loop delay, thereby enabling transmission with higher bit rates during the CAN FD data phase independent of the delay of a specific CAN transceiver.

To check for bit errors during the data phase of transmitting nodes, the delayed transmit data is compared against the received data at the Secondary Sample Point (SSP). If a bit error is detected, the transmitter will react on this bit error at the next following regular sample point. During arbitration phase the delay compensation is always disabled.

The transmitter delay compensation enables configurations where the data bit time is shorter than the transmitter delay, it is described in detail in ISO 11898-1:2015. It is enabled by setting bit TDC (CANFD_DBTP[23]).

The received bit is compared against the transmitted bit at the SSP. The SSP position is defined as the sum of the measured delay from the CAN FD controllers transmit output CANx_TXD pin through the transceiver to the receive input CANx_RXD pin plus the transmitter delay compensation offset as configured by TDCO (CANFD_TDCR[[14:8]). The transmitter delay compensation offset is used to adjust the position of the SSP inside the received bit (e.g. half of the bit time in the data phase). The position of the secondary sample point is rounded down to the next integer number of mtq (minimum time quantum).

TDCV (CANFD_PSR[22:16]) shows the actual transmitter delay compensation value. TDCV (CANFD_PSR[22:16]) is cleared when INIT (CANFD_CCCR[0]) is set and is updated at each transmission of an FD frame while TDC (CANFD_DBTP[23]) is set.

The following boundary conditions have to be considered for the transmitter delay compensation implemented in the CAN FD controller:

- The sum of the measured delay from CANx_TXD to CANx_RXD and the configured transmitter delay compensation offset TDCO (CANFD_TDCR[[14:8]) has to be less than 6 bit times in the data phase.

- The sum of the measured delay from CANx_TXD to CANx_RXD and the configured transmitter delay compensation offset TDCO (CANFD_TDCR[[14:8]) has to be less than or equal to 127 mtq (minimum time quantum). In case this sum exceeds 127 mtq, the maximum value of 127 mtq is used f or transmitter delay compensation.
- The data phase ends at the sample point of the CRC delimiter, that stops checking of receive bits at the SSPs.

If transmitter delay compensation is enabled by programming TDC (CANFD_DBTP[23]) is set, the measurement is started within each transmitted CAN FD frame at the falling edge of bit FDF to bit res. The measurement is stopped when this edge is seen at the receive input CANx_RXD pin of the transmitter. The resolution of this measurement is one mtq (minimum time quantum).

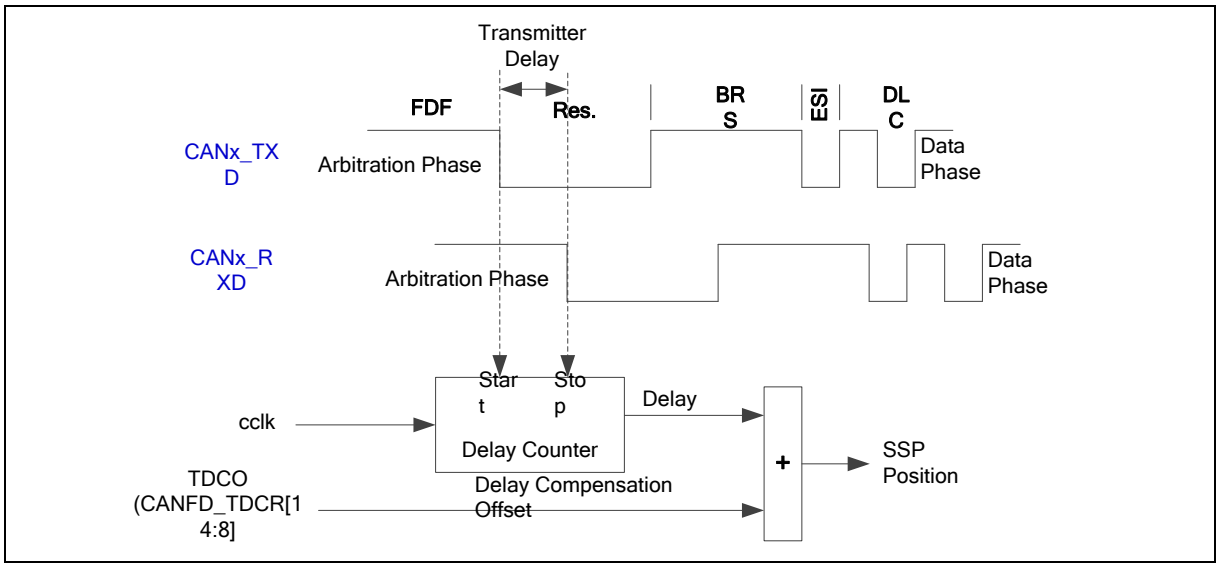


Figure 6.36-2 Transceiver Delay Measurement

To avoid that a dominant glitch inside the received FDF bit ends the delay compensation measurement before the falling edge of the received res bit, resulting in a too early SSP position, the use of a transmitter delay compensation filter window can be enabled by programming TDCF (CANFD_TDCR[6:0]). This defines a minimum value for the SSP position. Dominant edges on CANx_RXD, that would result in an earlier SSP position are ignored for transmitter delay measurement. The measurement is stopped when the SSP position is at least TDCF (CANFD_TDCR[6:0]) AND CANx_RXD is low.

Restricted Operation Mode

In Restricted Operation Mode the node is able to receive data and remote frames and to give acknowledge to valid frames, but it does not send data frames, remote frames, active error frames, or overload frames. In case of an error condition or overload condition, it does not send dominant bits, instead it waits for the occurrence of bus idle condition to resynchronize itself to the CAN communication. The error counters (REC (CANFD_ECR[14:8]), TEC (CANFD_ECR[7:0])) are frozen while Error Logging (CEL (CANFD_ECR[23:16])) is active. The software can set the CAN FD controller into Restricted Operation mode by setting bit ASM (CANFD_CCCR[2]). The bit can only be set by the software when both CCE (CANFD_CCCR[1]) and INIT (CANFD_CCCR[0]) are set to 1. The bit can be reset by the software at any time.

Restricted Operation Mode is automatically entered when the Tx Handler was not able to read data from the Message RAM in time. To leave Restricted Operation Mode, the software has to reset ASM (CANFD_CCCR[2]).

The Restricted Operation Mode can be used in applications that adapt themselves to different CAN bit rates. In this case the application tests different bit rates and leaves the Restricted Operation Mode after

it has received a valid frame.

Note: The Restricted Operation Mode must not be combined with the Loop Back Mode (internal or external).

Bus Monitor Mode

The CAN FD controller is set in Bus Monitoring Mode by programming MON (CANFD_CCCR[5]) to one. In Bus Monitoring Mode (refer to ISO 11898-1:2015, 10.14 Bus monitoring), the CAN FD controller is able to receive valid data frames and valid remote frames, but cannot start a transmission. In this mode, it sends only recessive bits on the CAN bus. If the controller is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the controller monitors this dominant bit, although the CAN bus may remain in recessive state. In Bus Monitoring Mode register CANFD_TXBRP is held in reset state.

The Bus Monitoring Mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits. Figure 6.36-3 shows the connection of signals CANx_TXD pin and CANx_RXD pin to the CAN FD controller in Bus Monitoring Mode.

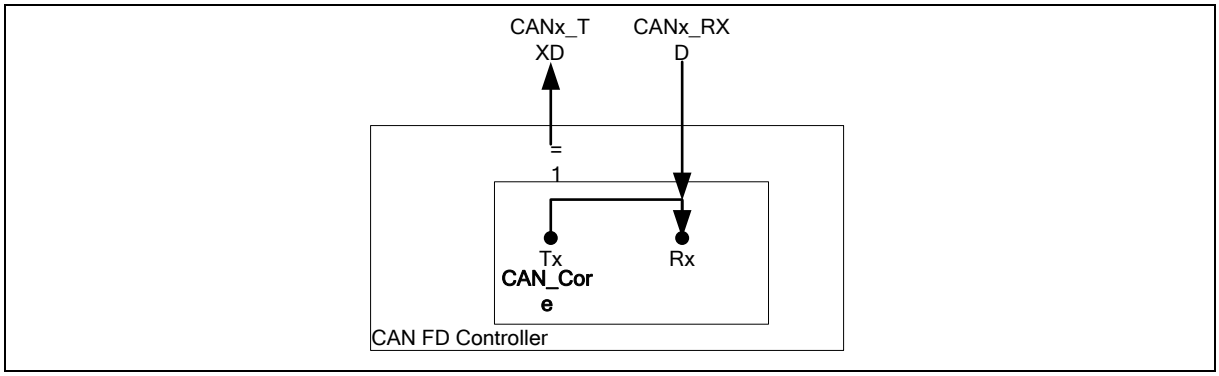


Figure 6.36-3 Pin Controller in Bus Monitor Mode

Disable Automatic Retransmission (DAR) mode

According to the CAN Specification (refer to ISO 11898-1:2015, 8.3.4 Recovery Management), the CAN FD controller provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. By default, automatic retransmission is enabled. To support time-triggered communication as described in ISO 11898-1:2015, chapter 9.2, the automatic retransmission may be disabled via DAR (CANFD_CCCR[6]).

Frame Transmission in DAR Mode

In DAR mode all transmissions are automatically cancelled after they started on the CAN bus. A Tx Buffer’s Tx Request Pending bit TRPn (CANFD_TXBRP[31:0]) is reset after successful transmission, when a transmission has not yet been started at the point of cancellation, has been aborted due to lost arbitration, or when an error occurred during frame transmission.

- Successful transmission:
  - Corresponding Tx Buffer Transmission Occurred bit TOn (CANFD_TXBTO[31:0]) set
  - Corresponding Tx Buffer Cancellation Finished bit CFn (CANFD_TXBCF[31:0]) not set
- Successful transmission in spite of cancellation:
  - Corresponding Tx Buffer Transmission Occurred bit TOn (CANFD_TXBTO[31:0]) set
  - Corresponding Tx Buffer Cancellation Finished bit CFn (CANFD_TXBCF[31:0]) set

- Arbitration lost or frame transmission disturbed:
  - Corresponding Tx Buffer Transmission Occurred bit TOn (CANFD_TXBTO[31:0]) not set
  - Corresponding Tx Buffer Cancellation Finished bit CFn (CANFD_TXBCF[31:0]) set

In case of a successful frame transmission, and if storage of Tx events is enabled, a Tx Event FIFO element is written with Event Type ET = 10 (transmission in spite of cancellation).

Power Mode

The CAN FD controller can be set into Power-down mode controlled by input signal power_down_req or via CSR (CANFD_CCCR[4]). As long as the clock stop request signal power_down_req is active, bit CSR (CANFD_CCCR[4]) is read as one.

When all pending transmission requests have completed, the CAN FD controller waits until bus idle state is detected. Then the CAN FD controller sets then INIT (CANFD_CCCR[0]) to one to prevent any further CAN transfers. Now the CAN FD controller acknowledges that it is ready for power down by setting output signal power_down_ack to one and CSA(CANFD_CCCR[3]) to one. Before the clocks are switched off, further register accesses can be made except for INIT (CANFD_CCCR[0]) which is held at one.

Note: In case of a heavily disturbed CAN bus, it may happen that idle state is never reached and INIT (CANFD_CCCR[0]) is therefore not set by the controller. This situation can be detected by polling ACT (CANFD_PSR[4:3]). In case the controller does not enter idle state, the software can write INIT (CANFD_CCCR[0]) to 1 which stops CAN communication of the CAN FD controller immediately, regardless whether there is a transmission/reception ongoing or not.

To leave Power-down mode, the application has to turn on the module clocks before resetting control register flag CSR (CANFD_CCCR[4]). The controller will acknowledge this by resetting CSA (CANFD_CCCR[3]). Afterwards, the application can restart CAN communication by resetting bit INIT (CANFD_CCCR[0]).

Test Mode

To enable write access to register TEST, bit TEST (CANFD_CCCR[7]) has to be set to 1. This allows the configuration of the test modes and test functions.

Four output functions are available for the CAN transmit pin CANx_TXD by programming TX (CANFD_TEST[6:5]). Additionally to its default function – the serial data output – it can drive the CAN Sample Point signal to monitor the CAN FD controller bit timing and it can drive constant dominant or recessive values. The actual value at pin CANx_RXD can be read from RX (CANFD_TEST[7]). Both functions can be used to check the CAN bus physical layer.

Due to the synchronization mechanism between CAN Core clock and AHB clock domain, there may be a delay of several AHB clock periods between writing to TX (CANFD_TEST6:5]) until the new configuration is visible at output pin CANx_TXD. This applies also when reading input pin CANx_RXD via RX (CANFD_TEST[7]).

Note: Test modes should be used for production tests or self-test only. The software control for pin CANx_TXD interferes with all CAN protocol functions. It is not recommended to use test modes for application.

- External Loop Back Mode

  The CAN FD controller can be set in External Loop Back Mode by programming LBCK (CANFD_TEST[4]) to one. In Loop Back Mode, the controller treats its own transmitted messages as received messages and stores them (if they pass acceptance filtering) into an Rx Buffer or an Rx FIFO. Figure 6.36-4 shows the connection of signals CANx_TXD pin and CANx_RXD pin to the CAN FD controller in External Loop Back Mode.

  This mode is provided for hardware self-test. To be independent from external stimulation,

the controller ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop Back Mode. In this mode the controller performs an internal feedback from its Tx output to its Rx input. The actual value of the CANx_RXD input pin is disregarded by the controller. The transmitted messages can be monitored at the CANx_TXD pin.

- Internal Loop Back Mode

  Internal Loop Back Mode is entered by programming bits LBCK (CANFD_TEST[4]) and MON (CANFD_CCCR[5]) to one. This mode can be used for a “Hot Selftest”, meaning the controller can be tested without affecting a running CAN system connected to the pins CANx_TXD and CANx_RXD. In this mode pin CANx_RXD is disconnected from the controller and pin CANx_TXD is held recessive. Figure 6.36-4 shows the connection of CANx_TXD and CANx_RXD to the controller in case of Internal Loop Back Mode.

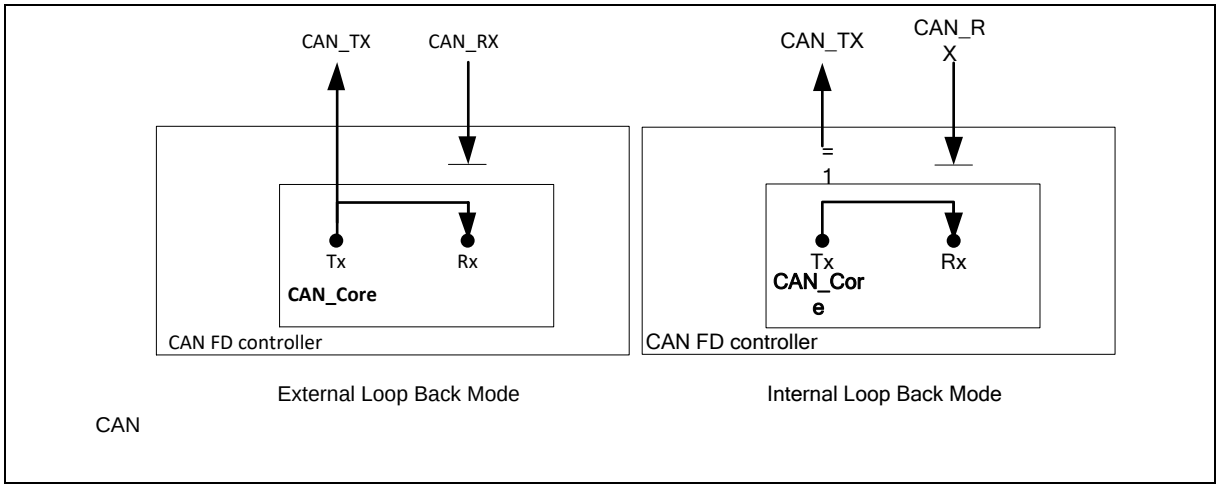


Figure 6.36-4 Pin Control in Loop Back Modes

6.36.5.2 Timestamp Generation

For timestamp generation the CAN FD controller supplies a 16-bit wrap-around counter. A pre-scaler TCP (CANFD_TSCC[19:16]) can be configured to clock the counter in multiples of CAN bit times (1…16). The counter is readable via TSC (CANFD_TSCV[15:0]). A write access to register CANFD_TSCV resets the counter to 0. When the timestamp counter wraps around interrupt flag TSW (CANFD_IR[16]) is set.

On start of frame reception / transmission (SOF) the counter value is captured and stored into the **Timestamp Generation** section of an Rx Buffer / Rx FIFO (RXTS(RX1A[15:0])) or Tx Event FIFO (TXTS(E1A[15:0])) element.

By programming bit TSS (CANFD_TSCC[[1:0]), a 16-bit timestamp can be used.

6.36.5.3 Timeout Counter

To signal timeout conditions for Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO the CANFD controller supplies a 16-bit timeout counter. It operates as down-counter and uses the same pre-scaler controlled by TCP (CANFD_TSCC[19:16]) as the timestamp counter. The timeout counter is configured via register CANFD_TOCC. The actual counter value can be read from TOC (CANFD_TOCV[15:0]). The timeout counter can only be started while INIT (CANFD_CCCR[0]) = 0. It is stopped when INIT (CANFD_CCCR[0]) = 1, e.g. when the CAN FD Controller enters Bus_Off state.

The operation mode is selected by TOS (CANFD_TOCC[2:1]). When operating in Continuous Mode, the counter starts when INIT (CANFD_CCCR[0]) is reset. A write to TOCV presets the counter to the value configured by TOP (CANFD_TOCC[31:16]) and continues down-counting.

When the timeout counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the

value configured by TOP (CANFD_TOCC[31:16]). Down-counting is started when the first FIFO element is stored. Writing to TOCV has no effect.

When the counter reaches zero, interrupt flag TOO (CANFD_IR[18]) is set. In Continuous Mode, the counter is immediately restarted at TOP (CANFD_TOCC[31:16]).

Note: The clock signal for the timeout counter is derived from the CAN Core's sample point signal. Therefore, the point in time where the timeout counter is decremented may vary due to the synchronization / re-synchronization mechanism of the CAN Core. If the bit rate switch feature in CAN FD controller is used, the timeout counter is clocked differently in arbitration and data field.

6.36.5.4 Rx Handling

The Rx Handler controls the acceptance filtering, the transfer of received messages to the Rx Buffers or to one of the two Rx FIFOs, as well as the Rx FIFO's put and get Indices.

Acceptance Filtering

The CAN FD controller offers the possibility to configure two sets of acceptance filters, one for standard identifiers and one for extended identifiers. These filters can be assigned to an Rx Buffer or to Rx FIFO 0,1. For acceptance filtering each list of filters is executed from element #0 until the first matching element. Acceptance filtering stops at the first matching element. The following filter elements are not evaluated for this message.

The main features are:

- Each filter element can be configured as
  - range filter (from 0 to 128 elements for the 11-bit filter and from 0 to 64 for the 29-bit filter)
  - filter for one or two dedicated IDs
  - classic bit mask filter
- Each filter element is configurable for acceptance or rejection filtering
- Each filter element can be enabled / disabled individually
- Filters are checked sequentially, execution stops with the first matching filter element

Related configuration registers are:

- Global Filter Configuration CANFD_GFC
- Standard ID Filter Configuration CANFD_SIDFC
- Extended ID Filter Configuration CANFD_XIDFC
- Extended ID AND Mask CANFD_XIDAM

Depending on the configuration of the filter element (SFEC/EFEC) a match triggers one of the following actions:

- Store received frame in FIFO 0 or FIFO 1
- Store received frame in Rx Buffer
- Store received frame in Rx Buffer and generate pulse at filter event pin
- Reject received frame
- Set High Priority Message interrupt flag HPM (CANFD_IR[8])
- Set High Priority Message interrupt flag HPM (CANFD_IR[8]) and store received frame in FIFO 0 or FIFO 1

Acceptance filtering is started after the complete identifier has been received. After acceptance filtering has completed, and if a matching Rx Buffer or Rx FIFO has been found, the Message Handler starts writing the received message data in portions of 32 bit to the matching Rx Buffer or Rx FIFO. If the CAN

protocol controller has detected an error condition (e.g. CRC error), this message is discarded with the following impact on the affected Rx Buffer or Rx FIFO:

Rx Buffer

New Data flag of matching Rx Buffer is not set, but Rx Buffer (partly) overwritten with received data. For error type refer to LEC (CANFD_PSR[2:0]) respectively DLEC (CANFD_PSR[10:8]).

Rx FIFO

Put index of matching Rx FIFO is not updated, but related Rx FIFO element (partly) overwritten with received data. For error type refer to LEC (CANFD_PSR[2:0]) respectively DLEC (CANFD_PSR[10:8]). In case the matching Rx FIFO is operated in overwrite mode, the boundary conditions described in **Rx FIFO Overwrite Mode** section have to be considered.

Note: When an accepted message is written to one of the two Rx FIFOs, or into an Rx Buffer, the unmodified received identifier is stored independent of the filter(s) used. The result of the acceptance filter process is strongly depending on the sequence of configured filter elements.

- Range Filter

  The filter matches for all received frames with Message IDs in the range defined by SFID1/SFID2 resp. EFID1/EFID2.

  There are two possibilities when range filtering is used together with extended frames:

  EFT = 2'b00: The Message ID of received frames is ANDed with the Extended ID AND Mask (CANFD_XIDAM) before the range filter is applied.

  EFT = 2'b11: The Extended ID AND Mask (CANFD_XIDAM) is not used for range filtering.

- Filter for specific IDs

  A filter element can be configured to filter for one or two specific Message IDs. To filter for one specific Message ID, the filter element has to be configured with SFID1 = SFID2 resp. EFID1 = EFID2.

- Classic Bit Mask Filter

  Classic bit mask filtering is intended to filter groups of Message IDs by masking single bits of a received Message ID. With classic bit mask filtering SFID1/EFID1 is used as Message ID filter, while SFID2/EFID2 is used as filter mask.

  A zero bit at the filter mask will mask out the corresponding bit position of the configured ID filter, e.g. the value of the received Message ID at that bit position is not relevant for acceptance filtering.

  Only those bits of the received Message ID where the corresponding mask bits are one are relevant for acceptance filtering.

  In case all mask bits are one, a match occurs only when the received Message ID and the Message ID filter are identical. If all mask bits are zero, all Message IDs match.

- Standard Message ID Filtering

  Figure 6.36-6 below shows the flow for standard Message ID (11-bit Identifier) filtering. The Standard Message ID Filter element is described in **Message RAM** section.

  Controlled by the global filter configuration GFC and the Standard ID filter configuration (CANFD_SIDFC) Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

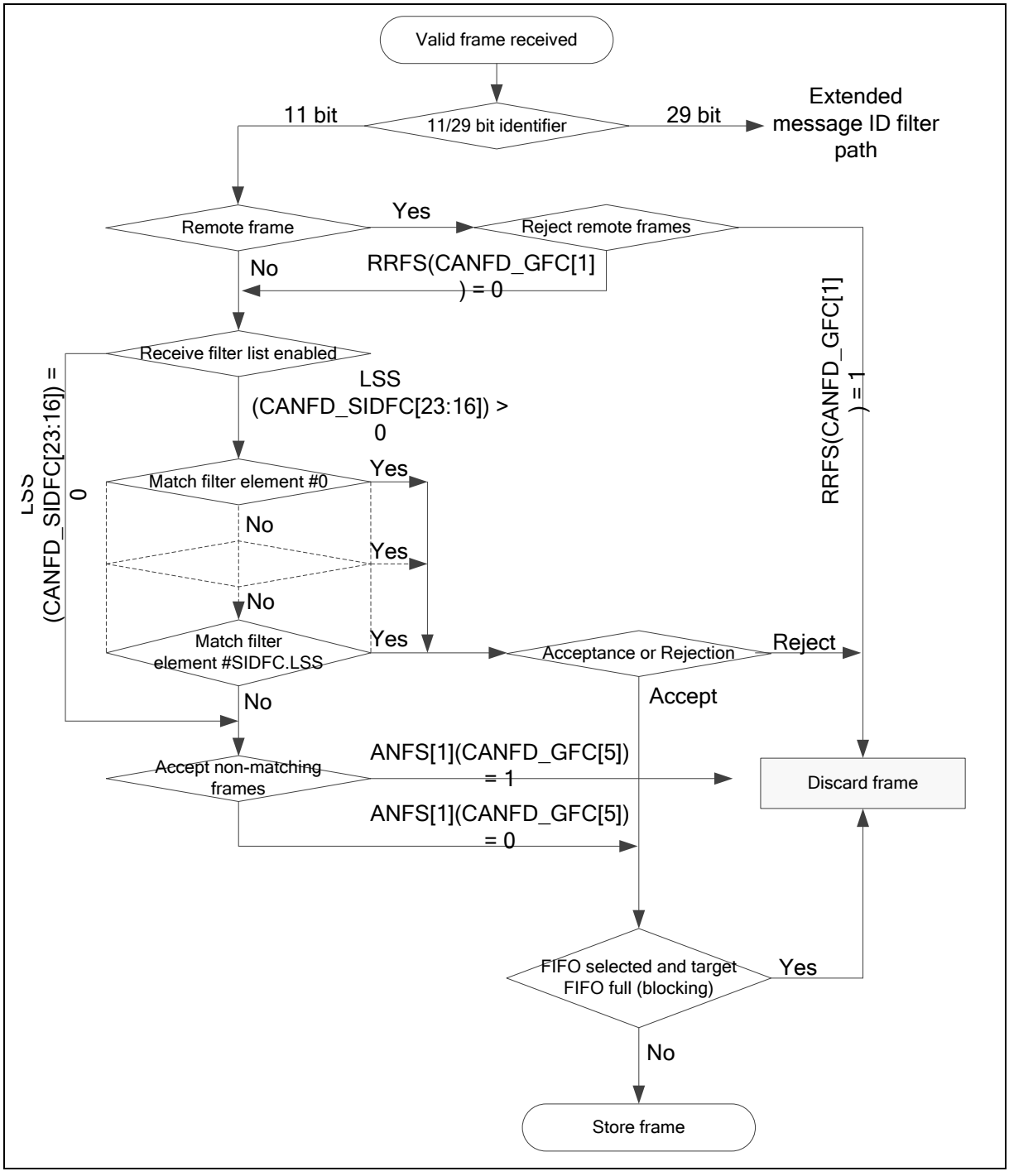


Figure 6.36-5 Standard Message ID Filter Path

- Extended Message ID Filtering

  Figure 6.36-6 shows the flow for extended Message ID (29-bit Identifier) filtering. The Extended Message ID Filter element is described in **Message RAM** Section.

  Controlled by the global filter configuration (CANFD_GFC) and the Extended ID filter configuration (CANFD_XIDFC) Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

The Extended ID AND Mask (CANFD_XIDAM) is AND-ed with the received identifier before the filter list is executed.

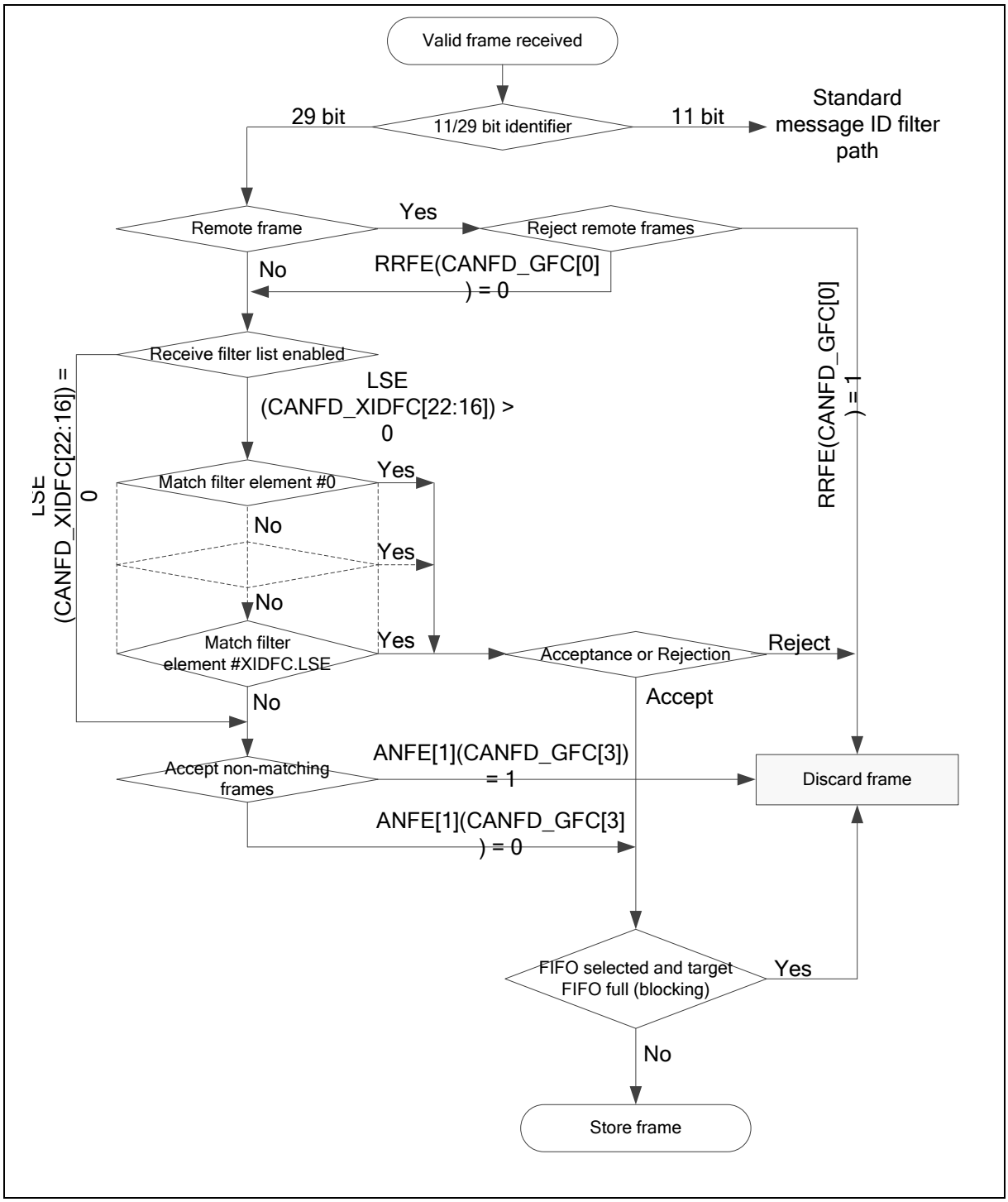


Figure 6.36-6 Extended Message ID Filter Path

Rx FIFOs

Rx FIFO 0 and Rx FIFO 1 can be configured to hold up to 64 elements each. Configuration of the two Rx FIFOs is done via registers CANFD_RXF0C and CANFD_RXF1C.

Received messages that passed acceptance filtering are transferred to the Rx FIFO as configured by

the matching filter element. For a description of the filter mechanisms available for Rx FIFO 0 and Rx FIFO 1 refer to **Acceptance Filtering** section. The Rx FIFO element is described in Rx Buffer and **FIFO Element** section.

To avoid an Rx FIFO overflow, the Rx FIFO watermark can be used. When the Rx FIFO fill level reaches the Rx FIFO watermark configured by FnWM (CANFD_RXFnC[30:24]), interrupt flag RFnW (CANFD_IR) is set. When the Rx FIFO put Index reaches the Rx FIFO get Index an Rx FIFO Full condition is signalled by FnF (CANFD_RXFnS). In addition, interrupt flag RFnF (CANFD_IR) is set.

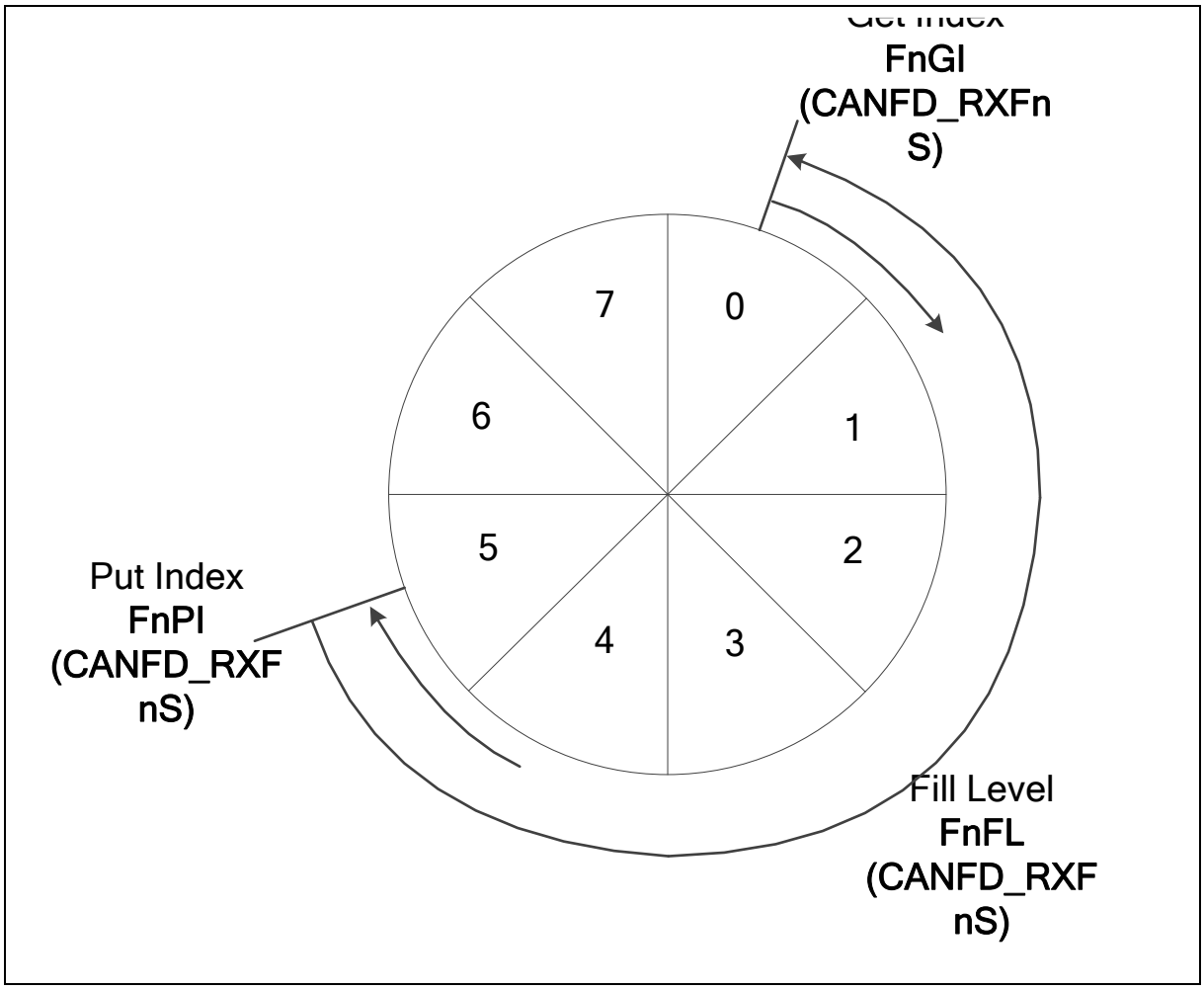


Figure 6.36-7 Rx FIFO Status

When reading from an Rx FIFO, Rx FIFO Get Index FnGI (CANFD_RXFnS) + FIFO element size has to be added to the corresponding Rx FIFO start address FnSA (CANFD_RXFnC).

RBDS(CANFD_RXESC[10:8]) FnDS(CANFD_RXESC[2:0])	Data Field [Bytes]	FIFO Element Size [RAM Words]
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8

101	32	10
110	48	14
111	64	18

Table 6.36-2 Rx Buffer / FIFO Element Size

- Rx FIFO Blocking Mode

  The Rx FIFO blocking mode is configured by FnOM (CANFD_RXFnC[31]) = 0. This is the default operation mode for the Rx FIFOs.

  When an Rx FIFO full condition is reached (FnPI (CANFD_RXFnS[21:16]) = FnGI (CANFD_RXFnS[13:8])), no further messages are written to the corresponding Rx FIFO until at least one message has been read out and the Rx FIFO Get Index has been incremented. An Rx FIFO full condition is signalled by FnF (CANFD_RXFnS[24]) = 1. In addition interrupt flag RFnF (CANFD_IR) is set.

  In case a message is received while the corresponding Rx FIFO is full, this message is discarded and the message lost condition is signalled by RFnL (CANFD_RXFnS[25]) = 1. In addition interrupt flag RFnL (CANFD_IR) is set.

- Rx FIFO Overwrite Mode

  The Rx FIFO overwrite mode is configured by FnOM (CANFD_RXFnC[31]) = 1.

  When an Rx FIFO full condition (FnPI (CANFD_RXFnS[21:16]) = FnGI (CANFD_RXFnS[13:8])) is signalled by FnF (CANFD_RXFnS[24]) = 1, the next message accepted for the FIFO will overwrite the oldest FIFO message. Put and get index are both incremented by one.

  When an Rx FIFO is operated in overwrite mode and an Rx FIFO full condition is signalled, reading of the Rx FIFO elements should start at least at get index + 1. The reason for that is, that it might happen, that a received message is written to the Message RAM (put index) while the CPU is reading from the Message RAM (get index). In this case inconsistent data may be read from the respective Rx FIFO element. Adding an offset to the get index when reading from the Rx FIFO avoids this problem. The offset depends on how fast the CPU accesses the Rx FIFO. Figure 6.36-8 shows an offset of two with respect to the get index when reading the Rx FIFO. In this case the two messages stored in element 1 and 2 are lost.

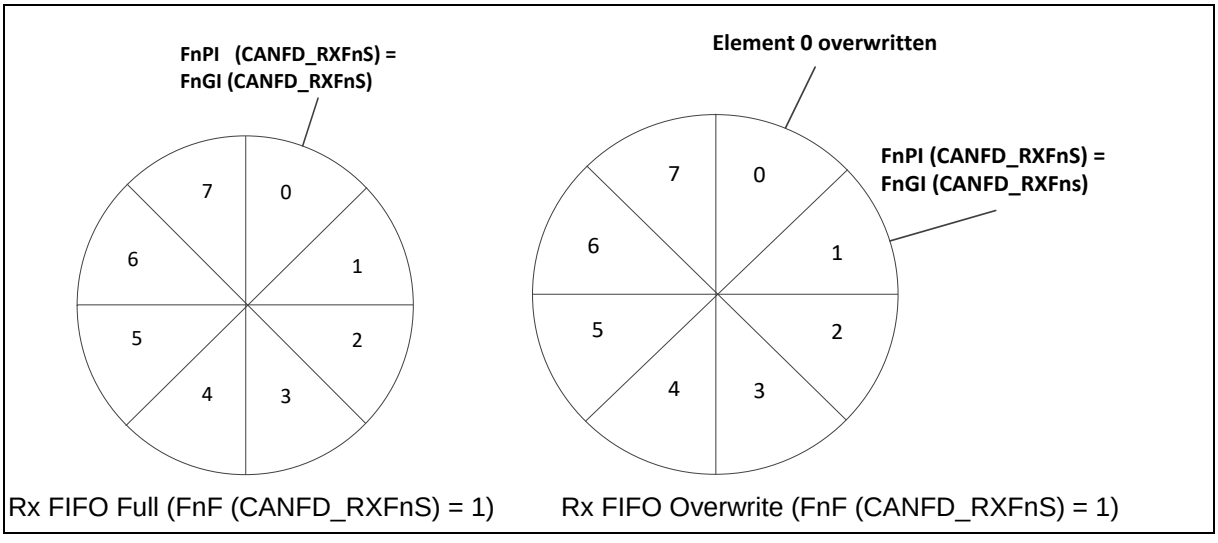


Figure 6.36-8 Rx FIFO Overflow Handling

After reading from the Rx FIFO, the number of the last element read has to be written to the Rx FIFO Acknowledge Index FnA (CANFD_RXFnA[5:0]). This increments the get index to that element number. In case the put index has not been incremented to this Rx FIFO element, the Rx FIFO full condition is reset (FnF (CANFD_RXFnS[24]) = '0').

Dedicated Rx Buffers

The CAN FD controller supports up to 64 dedicated Rx Buffers. The start address of the dedicated Rx Buffer section is configured via RBSA (CANFD_RXBC[15:2]).

For each Rx Buffer a Standard or Extended Message ID Filter Element with SFEC / EFEC = 111 and SFID2 / EFID2[10:9] = 00 has to be configured (refer to **Standard Message ID Filter Element** and **Extended Message ID Filter Element**).

After a received message has been accepted by a filter element, the message is stored into the Rx Buffer in the Message RAM referenced by the filter element. The format is the same as for an Rx FIFO element. In addition the flag DRX(CANFD_IR[19]) (Message stored in Dedicated Rx Buffer) in the interrupt register is set.

Filter Element	SFID1[10:0] EFID1[28:0]	SFID2[10:9] EFID2[10:9]	SFID2[5:0] EFID2[5:0]
0	ID message 1	00	00 0000
1	ID message 2	00	00 0001
2	ID message 3	00	00 0010

Table 6.36-3 Example Filter Configuration for Rx Buffers

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in register NDAT1,2 is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the software by writing a '1' to the respective bit position.

While an Rx buffer new data flag is set, a message ID filter element referencing this specific Rx Buffer will not match, causing the acceptance filtering to continue. The following message ID filter elements may cause the received message to be stored into another Rx Buffer, or into an Rx FIFO, or the message may be rejected, depending on filter configuration.

- Rx Buffer Handling
  - Reset interrupt flag DRX(CANFD_IR[19])
  - Read New Data registers
  - Read messages from Message RAM
  - Reset New Data flags of processed messages

6.36.5.5 Tx Handling

The Tx Handler handles transmission requests for the dedicated Tx Buffers, the Tx FIFO, and the Tx Queue. It controls the transfer of transmit messages to the CAN Core, the Put and Get Indices, and the Tx Event FIFO. Up to 32 Tx Buffers can be set up for message transmission. The CAN mode for transmission (Classic CAN or CAN FD) can be configured separately for each Tx Buffer element. The Tx Buffer element is described in **Tx Buffer Element** section. Table 6.36-4 describes the possible configurations for frame transmission.

CANFD_CCCR		Tx Buffer Element		Frame Transmission
BRSE	FDOE	FDF	BRS	
ignored	0	ignored	ignored	Classic CAN

0	1	0	ignored	Classic CAN
0	1	1	ignored	FD without bit rate switching
1	1	0	ignored	Classic CAN
1	1	1	0	FD without bit rate switching
1	1	1	1	FD with bit rate switching

Table 6.36-4 Possible Configuration for Frame Transmission

Note: AUTOSAR requires at least three Tx Queue Buffers and support of transmit cancellation

The Tx Handler starts a Tx scan of the Message RAM to check for the highest priority pending Tx request (Tx buffer with lowest message ID) when the Tx buffer request pending register CANFD_TXBRP is updated, or when a transmission has been started. CANFD_TXBRP is updated when a transmission finishes, or after the software has requested a transmission by writing to CANFD_TXBAR, or when the Host has cancelled a pending transmission by writing to CANFD_TXBCR.

Transmit Pause

The transmit pause feature is intended for use in CAN systems where the CAN message identifiers are (permanently) specified to specific values and cannot easily be changed. These message identifiers may have a higher CAN arbitration priority than other defined messages, while in a specific application their relative arbitration priority should be inverse. This may lead to a case where one electronic control unit (ECU) sends a burst of CAN messages that cause another ECU CAN messages to be delayed because that other messages have a lower CAN arbitration priority.

If e.g. CAN ECU-1 has the transmit pause feature enabled and is requested by its application software to transmit four messages, it will, after the first successful message transmission, wait for two CAN bit times of bus idle before it is allowed to start the next requested message. If there are other ECUs with pending messages, those messages are started in the idle time, they would not need to arbitrate with the next message of ECU-1. After having received a message, ECU-1 is allowed to start its next transmission as soon as the received message releases the CAN bus.

The transmit pause feature is controlled by bit TXP (CANFD_CCCR[14]). If the bit is set, the controller will, each time it has successfully transmitted a message, pause for two CAN bit times before starting the next transmission. This enables other CAN nodes in the network to transmit messages even if their messages have lower prior identifiers. Default is transmit pause disabled (TXP (CANFD_CCCR[14]) = 0).

This feature looses up burst transmissions coming from a single node and it protects against "babbling idiot" scenarios where the application program erroneously requests too many transmissions.

Dedicated Tx Buffers

Dedicated Tx Buffers are intended for message transmission under complete control of the CPU. Each Dedicated Tx Buffer is configured with a specific Message ID. In case that multiple Tx Buffers are configured with the same Message ID, the Tx Buffer with the lowest buffer number is transmitted first.

If the data section has been updated, a transmission is requested by an Add Request via ARn (CANFD_TXBAR[31:0]). The requested messages arbitrate internally with messages from an optional Tx FIFO or Tx queue and externally with messages on the CAN bus, and are sent out according to their Message ID.

A dedicated Tx buffer allocates four 32-bit words in the Message RAM (refer to Table 6.36-5). Therefore the start address of a dedicated Tx Buffer in the Message RAM is calculated by adding transmit buffer index (0…31) to the Tx buffer start address in TBSA (CANFD_TXBC[15:2]).

TBDS (CANFD_TXESC[2:0])	Data Field [Bytes]	Element Size [RAM Words]

000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

Table 6.36-5 Tx Buffer / FIFO / Queue Element Size

Tx FIFO

Tx FIFO operation is configured by programming TFQM (CANFD_TXBC[30]) to 0. Messages stored in the Tx FIFO are transmitted starting with the message referenced by the get index TFGI (CANFD_TXFQS[12:8]). After each transmission the Get index is incremented cyclically until the Tx FIFO is empty. The Tx FIFO enables transmission of messages with the same Message ID from different Tx Buffers in the order these messages have been written to the Tx FIFO. The controller calculates the Tx FIFO Free Level TFFL (CANFD_TXFQS[5:0]) as difference between get and put index. It indicates the number of available (free) Tx FIFO elements.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the put index TFQPI(CANFD_TXFQS[20:16]). An add request increments the Put Index to the next free Tx FIFO element. When the put index reaches the get index, Tx FIFO Full (TFQF (CANFD_TXFQS[21]) = 1) is signaled. In this case no further messages should be written to the Tx FIFO until the next message has been transmitted and the Get Index has been incremented.

When a single message is added to the Tx FIFO, the transmission is requested by writing a 1 to the TXBAR bit related to the Tx Buffer referenced by the Tx FIFO put index.

When multiple (n) messages are added to the Tx FIFO, they are written to n consecutive Tx Buffers starting with the Put Index. The transmissions are then requested via CANFD_TXBAR. The put index is then cyclically incremented by n. The number of requested Tx buffers should not exceed the number of free Tx Buffers as indicated by the Tx FIFO Free Level.

When a transmission request for the Tx Buffer referenced by the get Index is cancelled, the Get index is incremented to the next Tx Buffer with pending transmission request and the Tx FIFO free level is recalculated. When transmission cancellation is applied to any other Tx Buffer, the get index and the FIFO free level remain unchanged.

A Tx FIFO element allocates element size 32-bit words in the Message RAM (refer to Table 6.36-5). Therefore the start address of the next available (free) Tx FIFO Buffer is calculated by adding the put index TFQPI (CANFD_TXFQS[20:16]) * element size to the Tx buffer start address TBSA (CANFD_TXBC[15:2]).

Tx Queue

Tx Queue operation is configured by programming TFQM (CANFD_TXBC[30]) to 1. Messages stored in the Tx Queue are transmitted starting with the message with the lowest Message ID (highest priority). In case that multiple Queue Buffers are configured with the same Message ID, the Queue Buffer with the lowest buffer number is transmitted first.

New messages have to be written to the Tx Buffer referenced by the Put Index TFQPI (CANFD_TXFQS[20:16]). An Add Request cyclically increments the Put Index to the next free Tx Buffer. In case that the Tx Queue is full (TFQF (CANFD_TXFQS[21]) = 1), the Put Index is not valid and no further message should be written to the Tx Queue until at least one of the requested messages has been sent out or a pending transmission request has been cancelled.

The application may use register CANFD_TXBRP instead of the put index and may place messages to any Tx Buffer without pending transmission request.

A Tx Queue Buffer allocates element size 32-bit words in the Message RAM (refer to Table 6.36-5). Therefore the start address of the next available (free) Tx Queue Buffer is calculated by adding Tx FIFO/Queue put index TFQPI (CANFD_TXFQS[20:16]) * element size to the Tx buffer start address TBSA (CANFD_TXBC[15:2]).

Mixed Dedicated Tx Buffers / Tx FIFO

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx FIFO. The number of dedicated Tx Buffers is configured by NDTB (CANFD_TXBC[21:16]). The number of Tx Buffers assigned to the Tx FIFO is configured by TFQS (CANFD_TXBC[29:24]). In case TFQS (CANFD_TXBC[29:24]) is programmed to zero, only Dedicated Tx Buffers are used.

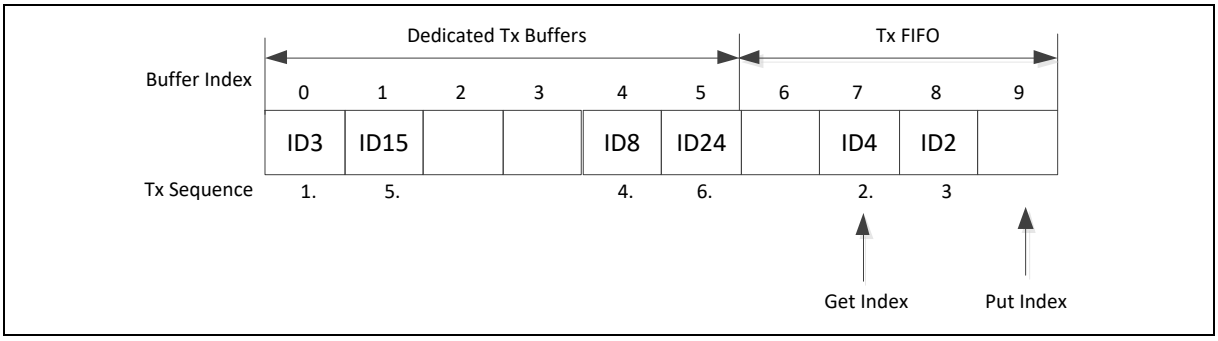


Figure 6.36-9 Example of Mixed Configuration Dedicated Tx Buffers / Tx FIFO

Tx prioritization:

- Scan Dedicated Tx Buffers and oldest pending Tx FIFO Buffer (referenced by TFGI (CANFD_TXFQS[12:8]))
- Buffer with lowest Message ID gets highest priority and is transmitted next.

Mixed Dedicated Tx Buffers / Tx Queue

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx Queue. The number of dedicated Tx Buffers is configured by NDTB (CANFD_TXBC[21:16]). The number of Tx Queue Buffers is configured by TFQS (CANFD_TXBC[29:24]). In case TFQS (CANFD_TXBC[29:24]) is programmed to zero, only dedicated Tx Buffers are used.

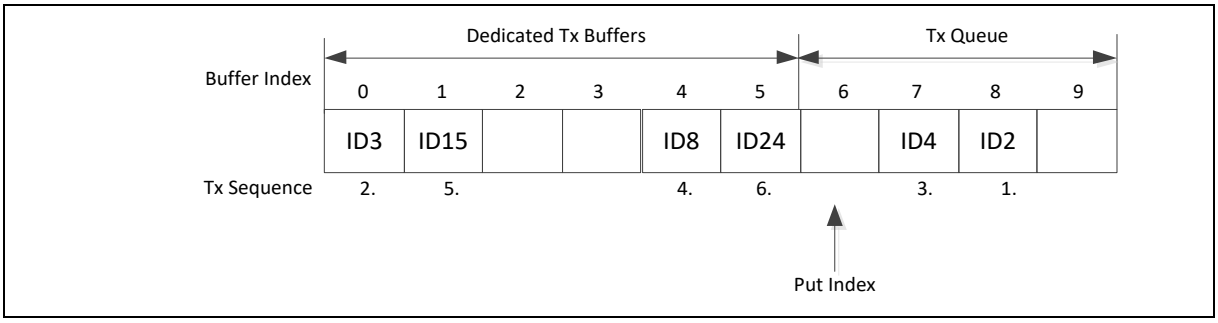


Figure 6.36-10 Example of mixed Configuration Dedicated Tx Buffers / Tx Queue

Tx prioritization:

- Scan all Tx Buffers with activated transmission request
- Tx Buffer with lowest Message ID gets highest priority and is transmitted next

Transmit Cancellation

The CAN FD controller supports transmit cancellation. This feature is especially intended for gateway applications and AUTOSAR based applications. To cancel a requested transmission from a dedicated Tx Buffer or a Tx Queue Buffer the software has to write a 1 to the corresponding bit position (=number of Tx Buffer) of register CANFD_TXBCR. Transmit cancellation is not intended for Tx FIFO operation.

Successful cancellation is signaled by setting the corresponding bit of register CANFD_TXBCF to 1. In case a transmit cancellation is requested while a transmission from a Tx Buffer is already ongoing, the corresponding CANFD_TXBRP bit remains set as long as the transmission is in progress. If the transmission was successful, the corresponding CANFD_TXBTO and CANFD_TXBCF bits are set. If the transmission was not successful, it is not repeated and only the corresponding CANFD_TXBCF bit is set.

Note: In case a pending transmission is cancelled immediately before this transmission could have been started, there follows a short time window where no transmission is started even if another message is also pending in this node. This may enable another node to transmit a message which may have a lower priority than the second message in this node.

Tx Event Handling

To support Tx event handling the controller has implemented a Tx Event FIFO. After the controller has transmitted a message on the CAN bus, message ID and timestamp are stored in a Tx Event FIFO element. To link a Tx event to a Tx Event FIFO element, the message marker from the transmitted Tx Buffer is copied into the Tx Event FIFO element.

The Tx Event FIFO can be configured to a maximum of 32 elements. The Tx Event FIFO element is described in **Tx Event FIFO Element** section.

The purpose of the Tx Event FIFO is to decouple handling transmit status information from transmit message handling i.e. a Tx Buffer holds only the message to be transmitted, while the transmit status is stored separately in the Tx Event FIFO. This has the advantage, especially when operating a dynamically managed transmit queue, that a Tx Buffer can be used for a new message immediately after successful transmission. There is no need to save transmit status information from a Tx Buffer before overwriting that Tx Buffer.

When a Tx Event FIFO full condition is signaled by TEFF (CANFD_IR[14]), no further elements are written to the Tx Event FIFO until at least one element has been read out and the Tx Event FIFO get Index has been incremented. In case a Tx event occurs while the Tx Event FIFO is full, this event is discarded and interrupt flag TEFL (CANFD_IR[15]) is set.

To avoid a Tx Event FIFO overflow, the Tx Event FIFO watermark can be used. When the Tx Event FIFO fill level reaches the Tx Event FIFO watermark configured by EFWM (CANFD_TXEFC[29:24]), interrupt flag TEFW (CANFD_IR[13]) is set.

When reading from the Tx Event FIFO, two times the Tx Event FIFO get Index EFGI (CANFD_TXEFS[12:8]) has to be added to the Tx Event FIFO start address EFSA (CANFD_TXEFC[15:2]).

6.36.5.6 FIFO Acknowledge Handling

The get indices of Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index (refer to **Rx FIFO 0 Acknowledge**, **Rx FIFO 1 Acknowledge**, and **Tx Event FIFO Acknowledge**). Writing to the FIFO Acknowledge Index will set the FIFO Get Index to the FIFO Acknowledge Index plus one and thereby updates the FIFO Fill Level. There are two use cases:

- When only a single element has been read from the FIFO (the one being pointed to by the get index), this get index value is written to the FIFO Acknowledge Index.
- When a sequence of elements has been read from the FIFO, it is sufficient to write the FIFO Acknowledge Index only once at the end of that read sequence (value: Index of the last element read), to update the FIFO’ s get index.

Due to the fact that the CPU has free access to the controller Message RAM, special care has to be taken when reading FIFO elements in an arbitrary order (get index not considered). This might be useful when reading a High Priority Message from one of the two Rx FIFOs. In this case the FIFO's Acknowledge Index should not be written because this would set the get index to a wrong position and also alters the FIFO's Fill Level. In this case some of the older FIFO elements would be lost.

Note: The application has to ensure that a valid value is written to the FIFO Acknowledge Index. The controller does not check for erroneous values.

6.36.5.7 Message RAM

For storage of Rx/Tx messages and for storage of the filter configuration a single-ported Message RAM has to be connected to the CAN Core module.

Message RAM Configuration

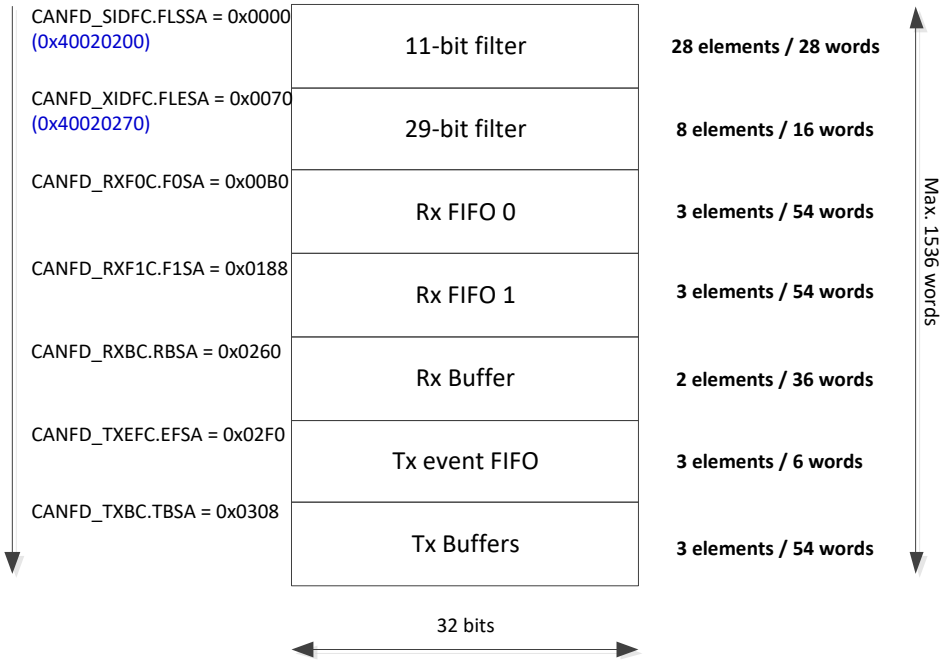
The address of the Message RAM and the address of the register are accessed through the same AHB interface. The maximum address of register used by the CAN FD controller is within 0x200. Therefore, the message SRAM used by the CAN controller must be used after 0x200. When reading or writing the SRAM content, the CAN controller will automatically subtract the offset of 0x200.

For example:

When the CPU wants to access the location of SRAM address is 0x100, its AHB address should be set as CANFD Base address + 0x200+0x100.

The Message RAM has a width of 32 bits. The CAN FD controller can be configured to allocate up to 1536 words in the Message RAM. It is not necessary to configure each of the sections listed in Figure 6.36-11, nor is there any restriction with respect to the sequence of the sections.

When operated in CAN FD mode the required Message RAM size strongly depends on the element size configured for Rx FIFO0, Rx FIFO1, Rx Buffers, and Tx Buffers via F0DS (CANFD_RXESC[2:0], F1DS (CANFD_RXESC[6:4]), RBDS (CANFD_RXESC[10:8]), and TBDS (CANFD_TXESC[2:0]).



Note: The base address of 1st CANFD = 0x40020200.

Figure 6.36-11 Message RAM configuration

When the controller addresses the Message RAM it addresses 32-bit words, not single bytes. The configurable start addresses are 32-bit word addresses i.e. only bits 15 to 2 are evaluated, the two least significant bits are ignored.

Rx Buffer and FIFO Element

Two Rx FIFOs and up to 64 Rx Buffers can be configured in the Message RAM. Each Rx FIFO section can be configured to store up to 64 received messages. The structure of a Rx Buffer / FIFO element is shown in below. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register CANFD_RXESC.

Bit	31			24	23			16	15	8	7	0
R0	ESI	XTD	RTR	ID[28:0]								
R1	ANMF	FIDX[6:0]			-	-	FDF	BRS	DLC[3:0]	RXTS[15:0]		
R2	DB3[7:0]				DB2[7:0]				DB1[7:0]		DB0[7:0]	
R3	DB7[7:0]				DB6[7:0]				DB5[7:0]		DB4[7:0]	

Rn	DBm[7:0]				DBm-1[7:0]				DBm-2[7:0]		DBm-3[7:0]	

Table 6.36-6 Rx Buffer and FIFO Element

Bit Field	Description
R0 Bit 31 ESI	Error State Indicator 0 = Transmitting node is error active 1 = Transmitting node is error passive
R0 Bit 30 XTD	Extended Identifier Signals to the MCU whether the received frame has a standard or extended identifier. 0 = 11-bit standard identifier 1 = 29-bit extended identifier
R0 Bit 29 RTR	Remote Transmission Request Signals to the Host whether the received frame is a data frame or a remote frame. 0 = Received frame is a data frame 1 = Received frame is a remote frame **Note:** There are no remote frames in CAN FD format. In CAN FD frames (FDF = 1'), the dominant RRS (Remote Request Substitution) bit replaces bit RTR (Remote Transmission Request)
R0 Bits 28:0 ID[28:0]	Identifier Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
R1 Bit 31 ANMF	Accepted Non-matching Frame Acceptance of non-matching frames may be enabled via ANFS (CANFD_GFC[5:4]) and ANFE (CANFD_GFC[3:2]). 0 = Received frame matching filter index FIDX 1 = Received frame did not match any Rx filter element

R1 Bits 30:24 FIDX[6:0]	Filter Index 0-127=Index of matching Rx acceptance filter element (invalid if ANMF = '1'). **Note:** Range is 0 to LSS (CANFD_SIDFC[23:16]) – 1 for standard and 0 to LSE (CANFD_XIDFC[22:16]) – 1 for extended.
R1 Bit 21 FDF	FD Format 0 = Classical CAN frame format 1 = CAN FD frame format (new DLC-coding and CRC)
R1 Bit 20 BRS	Bit Rate Switch 0 = Frame received without bit rate switching 1 = Frame received with bit rate switching
R1 Bits 19:16 DLC[3:0]	Data Length Code 0-8 = CAN + CAN FD: received frame has 0-8 data bytes 9-15 = CAN: received frame has 8 data bytes 9-15 = CAN FD: received frame has 12/16/20/24/32/48/64 data bytes
R1 Bits 15:0 RXTS[15:0]	Rx Timestamp Timestamp Counter value captured on start of frame reception. Resolution depending on configuration of the Timestamp Counter Prescaler TCP (CANFD_TSCC[19:16]).
R2 Bits 31:24 DB3[7:0]	Data Byte 3
R2 Bits 23:16 DB2[7:0]	Data Byte 2
R2 Bits 15:8 DB1[7:0]	Data Byte 1
R2 Bits 7:0 DB0[7:0]	Data Byte 0
R3 Bits 31:24 DB7[7:0]	Data Byte 7
R3 Bits 23:16 DB6[7:0]	Data Byte 6
R3 Bits 15:8 DB5[7:0]	Data Byte 5
R3 Bits 7:0 DB4[7:0]	Data Byte 4
…	…
Rn Bits 31:24 DBm[7:0]	Data Byte mm
Rn Bits 23:16 DBm-1[7:0]	Data Byte m-1
Rn Bits 15:8 DBm-2[7:0]	Data Byte m-2
Rn Bits 7:0 DBm-3[7:0]	Data Byte m-3 **Note:** Depending on the configuration of the element size (RXESC), between two and sixteen 32-bit words (Rn = 2 ..17) are used for storage of a CAN message's data field

Table 6.36-7 Rx Buffer and FIFO Element Description

Tx Buffer Element

The Tx Buffers section can be configured to hold dedicated Tx Buffers as well as a Tx FIFO / Tx Queue. In case that the Tx Buffers section is shared by dedicated Tx buffers and a Tx FIFO / Tx Queue, the dedicated Tx Buffers start at the beginning of the Tx Buffers section followed by the buffers assigned to the Tx FIFO or Tx Queue. The Tx Handler distinguishes between dedicated Tx Buffers and Tx FIFO / Tx Queue by evaluating the Tx Buffer configuration TFQS (CANFD_TXBC[29:24]) and NDTB (CANFD_TXBC[21:16]). The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register CANFD_TXESC.

Bit	31	30	29	28:0	24	23	22	21	20	19:16	15:8	7:0
T0	ESI	XTD	RTR	ID[28:0]								
T1	MM[7:0]					EFC	-	FDF	BRS	DLC[3:0]	MM[15:8]	-
T2	DB3[7:0]					DB2[7:0]					DB1[7:0]	DB0[7:0]
T3	DB7[7:0]					DB6[7:0]					DB5[7:0]	DB4[7:0]

Tn	DBm[7:0]					DBm-1[7:0]					DBm-2[7:0]	DBm-3[7:0]

Table 6.36-8 Tx Buffer and FIFO Element

Bit Field	Description
T0 Bit 31 ESI	Error State Indicator 0 = ESI bit in CAN FD format depends only on error passive flag 1 = ESI bit in CAN FD format transmitted recessive **Note:** The ESI bit of the transmit buffer is OR-ed with the error passive flag to decide the value of the ESI bit in the transmitted FD frame. As required by the CAN FD protocol specification, an error active node may optionally transmit the ESI bit recessive, but an error passive node will always transmit the ESI bit recessive.
T0 Bit 30 XTD	Extended Identifier 0 = 11-bit standard identifier 1 = 29-bit extended identifier
T0 Bit 29 RTR	Remote Transmission Request 0 = Transmit data frame 1 = Transmit remote frame **Note:** When RTR = 1, the controller transmits a remote frame according to ISO 11898-1:2015, even if FDOE (CANFD_CCCR[8]) enables the transmission in CAN FD format.
T0 Bits 28:0 ID[28:0]	Identifier Standard or extended identifier depending on bit XTD. A standard identifier has to be written to ID[28:18].
T1 Bits 31:24 MM[7:0]	Message Marker Written by CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status.
T1 Bit 23 EFC	Event FIFO Control 0 = Don't store Tx events 1 = Store Tx events
T1 Bit 21	FD Format

FDF	0 = Frame transmitted in Classic CAN format 1 = Frame transmitted in CAN FD format
T1 Bit 20 BRS	Bit Rate Switch 0 = CAN FD frames transmitted without bit rate switching 1 = CAN FD frames transmitted with bit rate switching **Note:** Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled FDOE (CANFD_CCCR[8]) = 1. Bit BRS is only evaluated when in addition BRSE (CANFD_CCCR[9]) = 1.
T1 Bits 19:16 DLC[3:0]	Data Length Code 0-8 = CAN + CAN FD: transmit frame has 0-8 data bytes 9-15 = CAN: transmit frame has 8 data bytes 9-15 = CAN FD: transmit frame has 12/16/20/24/32/48/64 data bytes
T2 Bits 31:24 DB3[7:0]	Data Byte 3
T2 Bits 23:16 DB2[7:0]	Data Byte 2
T2 Bits 15:8 DB1[7:0]	Data Byte 1
T2 Bits 7:0 DB0[7:0]	Data Byte 0
T3 Bits 31:24 DB7[7:0]	Data Byte 7
T3 Bits 23:16 DB6[7:0]:	Data Byte 6
T3 Bits 15:8 DB5[7:0]	Data Byte 5
T3 Bits 7:0 DB4[7:0]	Data Byte 4
... ...	... ...
Tn Bits 31:24 DBm[7:0]	Data Byte m
Tn Bits 23:16 DBm-1[7:0]	Data Byte m-1
Tn Bits 15:8 DBm-2[7:0]	Data Byte m-2
Tn Bits 7:0 DBm-3[7:0]	Data Byte m-3 **Note:** Depending on the configuration of the element size (TXESC), between two and sixteen 32-bit words (Tn = 2 ..17) are used for storage of a CAN message's data field.

Table 6.36-9 Tx Buffer Element Description

Tx Event FIFO Element

Each element stores information about transmitted messages. By reading the Tx Event FIFO the MCU gets this information in the order the messages were transmitted. Status information about the Tx Event

FIFO can be obtained from register CANFD_TXEFS.

Bit	31			24	23			16	15	8	7	0
E0	ESI	XTD	RTR	ID[28:0]								
E1	MM[7:0]				ET [1:0]	FDF	BRS	DLC[3:0]	TXTS[15:0]			

Table 6.36-10 Tx Event FIFO Element

Bit Field	Description
E0 Bit 31 ESI	Error State Indicator 0 = Transmitting node is error active 1 = Transmitting node is error passive
E0 Bit 30 XTD	Extended Identifier 0 = 11-bit standard identifier 1 = 29-bit extended identifier
E0 Bit 29 RTR	Remote Transmission Request 0 = Data frame transmitted 1 = Remote frame transmitted
E0 Bits 28:0 ID[28:0]	Identifier Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
E1 Bits 31:24 MM[7:0]	Message Marker Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status.
E1 Bit 23:22 ET[1:0]	Event Type 00 = Reserved 01 = Tx event 10 = Transmission in spite of cancellation (always set for transmissions in DAR mode) 11 = Reserved
E1 Bit 21 FDF	FD Format 0 = Classical CAN frame format 1 = CAN FD frame format (new DLC-coding and CRC)
E1 Bit 20 BRS	Bit Rate Switch 0 = Frame transmitted without bit rate switching 1 = Frame transmitted with bit rate switching
E1 Bits 19:16 DLC[3:0]	Data Length Code 0-8 = CAN + CAN FD: frame with 0-8 data bytes transmitted 9-15 = CAN: frame with 8 data bytes transmitted 9-15 = CAN FD: frame with 12/16/20/24/32/48/64 data bytes transmitted
E1 Bits 15:0 TXTS[15:0]	Tx Timestamp Timestamp Counter value captured on start of frame transmission. Resolution depending on configuration of the Timestamp Counter Prescaler TCP (CANFD_TSCC[19:16]).

Table 6.36-11 Tx Event FIFO Element Description

Standard Message ID Filter Element

Up to 128 filter elements can be configured for 11-bit IDs. When accessing a Standard Message ID Filter element, its address is the Filter List Standard Start Address FLSSA (CANFD_SIDFC[15:2]) plus the index of the filter element (0…127).

Bit	31		24	23	16	15	8	7	0
S0	SFT [1:0]	SFEC [2:0]	SFID1[10:0]			-	SFID2[10:0]		

Table 6.36-12 Standard Message ID Filter Element

Bit Field		Description
Bits 31:30 SFT[1:0]		Standard Filter Type 00 = Range filter from SFID1 to SFID2 (SFID2 ≥ SFID1) 01 = Dual ID filter for SFID1 or SFID2 10 = Classic filter: SFID1 = filter, SFID2 = mask 11 = Filter element disabled **Note:** With SFT = 11 the filter element is disabled and the acceptance filtering continues (same behavior as with SFEC = 000)
Bit 29:27 SFEC[2:0]		Standard Filter Element Configuration All enabled filter elements are used for acceptance filtering of 11-bit ID frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If SFEC = 100, 101, or 110 a match sets interrupt flag HPM (CANFD_IR[8]) and, if enabled, an interrupt is generated. In this case register HPMS is updated with the status of the priority match. 000 = Disable filter element 001 = Store in Rx FIFO 0 if filter matches 010 = Store in Rx FIFO 1 if filter matches 011 = Reject ID if filter matches, not intended to be used with Sync messages 100 = Set priority if filter matches, not intended to be used with Sync messages, no storage 101 = Set priority and store in FIFO 0 if filter matches 110 = Set priority and store in FIFO 1 if filter matches 111 = Store into Rx Buffer, configuration of SFT[1:0] ignored
Bits 26:16 SFID1[10:0]		Standard Filter ID 1 First ID of standard ID filter element. When filtering for Rx Buffers, Sync messages, this field defines the ID of the message to be stored. The received identifiers must match exactly, no masking mechanism is used.
Bits 15:0	SFID2[10:0]	Standard filter ID 2 This bit field has a different meaning depending on the configuration of SFEC: SFEC = 001 ... 110 Second ID of standard ID filter element SFEC = 111 Filter for Rx buffers
	SFID2[10:9]	Decides whether the received message is stored into an Rx buffer . 00: Store message into an Rx buffer
	SFID2[8:6]	The bits are used to control the filter event pins at the Extension Interface. A '1' at the respective bit position enables generation of a pulse at the related filter event pin with the duration of one system clock period in case the filter matches. SFID2[8] is used by the calibration unit.
	SFID2[5:0]	Defines the offset to the Rx buffer start address RBSA (CANFD_RXBC[15:2]) for storage of a matching message.

Table 6.36-13 Standard Message ID Filter Element Description

Extended Message ID Filter Element

Up to 64 filter elements can be configured for 29-bit IDs. When accessing an Extended Message ID Filter element, its address is the Filter List Extended Start Address FLESA (CANFD_XIDFC[15:2]) plus two times the index of the filter element (0…63).

Bit	31		24	23	16	15	8	7	0
F0	EFECV [2:0]		EFID1[28:0]						
F0	EFT [1:0]	-	EFID2[28:0]						

Table 6.36-14 Extended Message ID Filter Element

Field		Description
F0 Bit 31:29 EFEC[2:0]		Extended Filter Element Configuration All enabled filter elements are used for acceptance filtering of 29-bit ID frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If EFEC = 100, 101, or 110 a match sets interrupt flag HPM (CANFD_IR[8]) and, if enabled, an interrupt is generated. In this case register HPMS is updated with the status of the priority match. 000 = Disable filter element 001 = Store in Rx FIFO 0 if filter matches 010 = Store in Rx FIFO 1 if filter matches 011 = Reject ID if filter matches 100 = Set priority if filter matches 101 = Set priority and store in FIFO 0 if filter matches 110 = Set priority and store in FIFO 1 if filter matches 111 = Store into Rx Buffer, configuration of EFT[1:0] ignored
F0 Bits 28:0 EFID1[28:0]		Extended Filter ID 1 First ID of extended ID filter element. When filtering for Rx Buffers, Sync messages, this field defines the ID of an extended message to be stored. The received identifiers must match exactly, only CANFD_XIDAM masking mechanism (refer to Extended Message ID Filtering) is used.
F1 Bits 31:30 EFT[1:0]		Extended Filter Type 00 = Range filter from EFID1 to EFID2 (EFID2 > EFID1 or EFID2 = EFID1) 01 = Dual ID filter for EFID1 or EFID2 10 = Classic filter: EFID1 = filter, EFID2 = mask 11 = Range filter from EFID1 to EFID2 (EFID2 >= EFID1), CANFD_XIDAM mask not applied.
F1 Bits 28:0	EFID2[10:0]	Extended filter ID 2 This bit field has a different meaning depending on the configuration of EFEC: – SFEC = 001 ... 110 Second ID of extended ID filter element – SFEC = 111 Filter for Rx buffers
	EFID2[10:9]	Decides whether the received message is stored into an Rx Buffer. 00 = Store message into an Rx Buffer
	EFID2[8:6]	Is used to control the filter event pins at the Extension Interface. A one at the respective bit position enables generation of a pulse at the related filter event pin with the duration of one system clock period in case the filter matches.
	EFID2[5:0]	Defines the offset to the Rx Buffer Start Address RBSA (CANFD_RXBC[15:2]) for storage of a matching message.

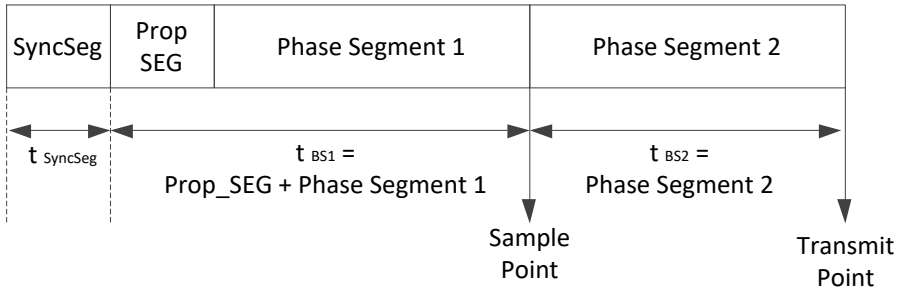
Table 6.36-15 Extended Message ID Filter Element Description

6.36.5.8 Bit Timing

The bit timing logic monitors the serial bus-line and performs sampling and adjustment of the sample point by synchronizing on the start-bit edge and resynchronizing on the following edges.

As shown in Figure 6.36-12, its operation may be explained simply by splitting the bit time in three segments, as follows:

- Synchronization segment (SYNC_SEG): a bit change is expected to occur within this time segment, having a fixed length of one time quantum (1 x tq).
- Bit segment 1 (BS1): defines the location of the sample point. It includes the PROP_SEG and PHASE_SEG1 of the CAN standard. Its duration is programmable between 1 and 16 time quanta, but may be automatically lengthened to compensate for positive phase drifts due to differences in the frequency of various nodes of the network.
- Bit segment 2 (BS2): defines the location of the transmit point. It represents the PHASE_SEG2 of the CAN standard, its duration is programmable between one and eight time quanta, but may also be automatically shortened to compensate for negative phase drifts.



Note: BS = Bit segment

Figure 6.36-12 Bit Timing

The baud rate is the inverse of bit time (baud rate = 1 / bit time), which, in turn, is the sum of three components. Figure 6.36-12 indicates that bit time = $t_{SyncSeg} + t_{BS1} + t_{BS2}$, where:

for the nominal bit time

– tq = (NBRP (CANFD_NBTP[24:16])+ 1) * $t_{canfd_tq_clk}$

– $t_{SyncSeg}$ = 1 tq

– t_{BS1} = tq * (NTSEG1(CANFD_NBTP[15:8]) + 1)

– t_{BS2} = tq * (NTSEG2(CANFD_NBTP[6:0]) + 1)

for the data bit time

– tq = (DBRP(CANFD_DBTP[20:16]) + 1) * $t_{canfd_tq_clk}$

– $t_{SyncSeg}$ = 1 tq

– t_{BS1} = tq * (DTSEG1(CANFD_DBTP[12:8]) + 1)

– t_{BS2} = tq * (DTSEG2(CANFD_DBTP[7:4]) + 1)

The (Re)Synchronization Jump Width (SJW) defines an upper bound for the amount of lengthening or shortening of the bit segments. It is programmable between one and four time quanta.

A valid edge is defined as the first transition in a bit time from dominant to recessive bus level, provided the controller itself does not send a recessive bit.

If a valid edge is detected in BS1 instead of SYNC_SEG. BS1 is extended by up to SJW so that the sample point is delayed.

Conversely, if a valid edge is detected in BS2 instead of SYNC_SEG. BS2 is shortened by up to SJW so that the transmit point is moved earlier.

As a safeguard against programming errors, the configuration of the Bit Timing register is only possible while the device is in Standby mode. Registers CANFD_DBTP and CANFD_NBTP (dedicated, respectively, to data and nominal bit timing) are only accessible when CCE (CANFD_CCCR[1]) and INIT (CANFD_CCCR[0]) are set.

Note: For a detailed description of the CAN bit timing and resynchronization mechanism, refer to the ISO 11898-1 standard.

6.36.6 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
CANFD Base Address: **CANFDn_BA = 0x4002_0000+ (0x4000 *n)** **n=0, 1**				
CANFD_DBTP	CANFDn_BA+0x00C	R	Data Bit Timing & Prescaler Register (P*)	0x0000_0A33
CANFD_TEST	CANFDn_BA+0x010	R	Test Register (P*)	0x0000_0000
CANFD_RWD	CANFDn_BA+0x014	R	RAM Watchdog (P*)	0x0000_0000
CANFD_CCCR	CANFDn_BA+0x018	R/W	CC Control Register (Pp*)	0x0000_0001
CANFD_NBTP	CANFDn_BA+0x01C	R	Nominal Bit Timing & Prescaler Register (P*)	0x0600_0A03
CANFD_TSCC	CANFDn_BA+0x020	R	Timestamp Counter Configuration (P*)	0x0000_0000
CANFD_TSCV	CANFDn_BA+0x024	R	Timestamp Counter Value (C*)	0x0000_0000
CANFD_TOCC	CANFDn_BA+0x028	R	Timeout Counter Configuration (P*)	0xFFFF_0000
CANFD_TOCV	CANFDn_BA+0x02C	R	Timeout Counter Value (C*)	0x0000_FFFF
CANFD_ECR	CANFDn_BA+0x040	R	Error Counter Register (X*)	0x0000_0000
CANFD_PSR	CANFDn_BA+0x044	R	Protocol Status Register (XS*)	0x0000_0707
CANFD_TDCR	CANFDn_BA+0x048	R	Transmitter Delay Compensation Register (P*)	0x0000_0000
CANFD_IR	CANFDn_BA+0x050	R/W	Interrupt Register	0x0000_0000
CANFD_IE	CANFDn_BA+0x054	R/W	Interrupt Enable	0x0000_0000
CANFD_ILS	CANFDn_BA+0x058	R/W	Interrupt Line Select	0x0000_0000
CANFD_ILE	CANFDn_BA+0x05C	R/W	Interrupt Line Enable	0x0000_0000
CANFD_GFC	CANFDn_BA+0x080	R	Global Filter Configuration (P*)	0x0000_0000
CANFD_SIDFC	CANFDn_BA+0x084	R	Standard ID Filter Configuration (P*)	0x0000_0000
CANFD_XIDFC	CANFDn_BA+0x088	R	Extended ID Filter Configuration (P*)	0x0000_0000
CANFD_XIDAM	CANFDn_BA+0x090	R	Extended ID AND Mask (P*)	0x1FFF_FFFF
CANFD_HPMS	CANFDn_BA+0x094	R	High Priority Message Status	0x0000_0000
CANFD_NDAT1	CANFDn_BA+0x098	R/W	New Data 1	0x0000_0000
CANFD_NDAT2	CANFDn_BA+0x09C	R/W	New Data 2	0x0000_0000
CANFD_RXF0C	CANFDn_BA+0x0A0	R	Rx FIFO 0 Configuration (P*)	0x0000_0000
CANFD_RXF0S	CANFDn_BA+0x0A4	R	Rx FIFO 0 Status	0x0000_0000
CANFD_RXF0A	CANFDn_BA+0x0A8	R/W	Rx FIFO 0 Acknowledge	0x0000_0000

CANFD_RXBC	CANFDn_BA+0x0AC	R	Rx Buffer Configuration (P*)	0x0000_0000
CANFD_RXF1C	CANFDn_BA+0x0B0	R	Rx FIFO 1 Configuration (P*)	0x0000_0000
CANFD_RXF1S	CANFDn_BA+0x0B4	R	Rx FIFO 1 Status	0x0000_0000
CANFD_RXF1A	CANFDn_BA+0x0B8	R/W	Rx FIFO 1 Acknowledge	0x0000_0000
CANFD_RXESC	CANFDn_BA+0x0BC	R	Rx Buffer / FIFO Element Size Configuration (P*)	0x0000_0000
CANFD_TXBC	CANFDn_BA+0x0C0	R	Tx Buffer Configuration (P*)	0x0000_0000
CANFD_TXFQS	CANFDn_BA+0x0C4	R	Tx FIFO/Queue Status	0x0000_0000
CANFD_TXESC	CANFDn_BA+0x0C8	R	Tx Buffer Element Size Configuration (P*)	0x0000_0000
CANFD_TXBRP	CANFDn_BA+0x0CC	R	Tx Buffer Request Pending	0x0000_0000
CANFD_TXBAR	CANFDn_BA+0x0D0	R/W	Tx Buffer Add Request	0x0000_0000
CANFD_TXBCR	CANFDn_BA+0x0D4	R/W	Tx Buffer Cancellation Request	0x0000_0000
CANFD_TXBTO	CANFDn_BA+0x0D8	R	Tx Buffer Transmission Occurred	0x0000_0000
CANFD_TXBCF	CANFDn_BA+0x0DC	R	Tx Buffer Cancellation Finished	0x0000_0000
CANFD_TXBTIE	CANFDn_BA+0x0E0	R/W	Tx Buffer Transmission Interrupt Enable	0x0000_0000
CANFD_TXBCIE	CANFDn_BA+0x0E4	R/W	Tx Buffer Cancellation Finished Interrupt Enable	0x0000_0000
CANFD_TXEFC	CANFDn_BA+0x0F0	R	Tx Event FIFO Configuration (P*)	0x0000_0000
CANFD_TXEFS	CANFDn_BA+0x0F4	R	Tx Event FIFO Status	0x0000_0000
CANFD_TXEFA	CANFDn_BA+0x0F8	R/W	Tx Event FIFO Acknowledge	0x0000_0000

Note:

R = Read, S = Set on read, X = Reset on read, W = Write, P = Protected write, p = Protected set, C = Clear/preset on write, r = release, d = date

6.36.7 Register Description

Data Bit Timing & Prescaler Register (CANFD_DBTP)

Register	Offset	R/W	Description	Reset Value
CANFD_DBTP	CANFDn_BA+0x00C	R	Data Bit Timing & Prescaler Register (P*)	0x0000_0A33

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TDC	Reserved		DBRP				
15	14	13	12	11	10	9	8
Reserved			DTSEG1				
7	6	5	4	3	2	1	0
DTSEG2				DSJW			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**TDC**	**Transmitter Delay Compensation** *0*=Transmitter Delay Compensation disabled. *1*=Transmitter Delay Compensation enabled.
[22:21]	**Reserved**	Reserved.
[20:16]	**DBRP**	**Data Bit Rate Prescaler** The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Bit Rate Prescaler are 0 to 31. When **TDC** = '1', the range is limited to 0,1. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
[15:13]	**Reserved**	Reserved.
[12:8]	**DTSEG1**	**Data time segment before sample point** Valid values are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. t_{BS1} = (DTSEG1 + 1) x tq.
[7:4]	**DTSEG2**	**Data time segment after sample point** Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. t_{BS2} = (DTSEG2 + 1) x tq.
[3:0]	**DSJW**	**Data Re-Synchronization Jump Width** Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. t_{SJW} = (DSJW + 1) x tq.

TEST Register (CANFD_TEST)

Register	Offset	R/W	Description	Reset Value
CANFD_TEST	CANFDn_BA+0x010	R	Test Register (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
RX	TX		LBCK	Reserved			

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**RX**	**Receive Pin** Monitors the actual value of pin CANx_RXD 0 = The CAN bus is dominant (CANx_RXD = 0). 1 = The CAN bus is recessive (CANx_RXD = 1).
[6:5]	**TX**	**Control of Transmit Pin** 00 = Reset value, CANx_TXD controlled by the CAN Core, updated at the end of the CAN bit time. 01 = Sample Point can be monitored at pin CANx_TXD. 10 = Dominant (‘0’) level at pin CANx_TXD. 11 = Recessive (‘1’) level at pin CANx_TXD.
[4]	**LBCK**	**Loop Back Mode** 0 = Reset value, Loop Back Mode is disabled. 1 = Loop Back Mode is enabled (refer to 6.36.5.1 TEST Mode).
[3:0]	**Reserved**	Reserved.

RAM Watchdog Register (CANFD_RWD)

The RAM watchdog monitors the READY output of the message RAM. A message RAM access starts the message RAM watchdog counter with the value configured by the WDC (CANFD_RWD[7:0]) bits. The counter is reloaded with WDC bits when the message RAM signals successful completion by activating its READY output. In case there is no response from the message RAM until the counter has counted down to zero, the counter stops and interrupt flag WDI (CANFD_IR[26]) bit is set.

Register	Offset	R/W	Description	Reset Value
CANFD_RWD	CANFDn_BA+0x014	R	RAM Watchdog (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
WDV							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WDC							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**WDV**	**Watchdog Value** Actual Message RAM Watchdog Counter Value.
[7:0]	**WDC**	**Watchdog Configuration** Start value of the Message RAM Watchdog Counter. With the reset value of 00 the counter is disabled.

CC Control Register (CANFD_CCCR)

Register	Offset	R/W	Description	Reset Value
CANFD_CCCR	CANFDn_BA+0x018	R/W	CC Control Register (Pp*)	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
NISO	TXP	EFBI	PXHD	Reserved		BRSE	FDOE
7	6	5	4	3	2	1	0
TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**NISO**	**Non ISO Operation** If this bit is set, the CAN FD controller uses the CAN FD frame format as specified by the Bosch CAN FD Specification V1.0. 0 = CAN FD frame format according to ISO 11898-1:2015. 1 = CAN FD frame format according to Bosch CAN FD Specification V1.0.
[14]	**TXP**	**Transmit Pause** If this bit is set, the CAN FD controller pauses for two CAN bit times before starting the next transmission after itself has successfully transmitted a frame (refer to 6.36.5.5). 0 = Transmit pause disabled. 1 = Transmit pause enabled.
[13]	**EFBI**	**Edge Filtering during Bus Integration** 0 = Edge filtering disabled. 1 = Two consecutive dominant tq required to detect an edge f or hard synchronization.
[12]	**PXHD**	**Protocol Exception Handling Disable** 0 = Protocol exception handling enabled. 1 = Protocol exception handling disabled. **Note:** When protocol exception handling is disabled, the controller will transmit an error frame when it detects a protocol exception condition.
[11:10]	**Reserved**	Reserved.
[9]	**BRSE**	**Bit Rate Switch Enable** 0 = Bit rate switching for transmissions disabled. 1 = Bit rate switching for transmissions enabled. **Note:** When CAN FD operation is disabled FDOE = 0, BRSE is not evaluated.
[8]	**FDOE**	**FD Operation Enable** 0 = FD operation disabled. 1 = FD operation enabled.

[7]	**TEST**	**Test Mode Enable** 0 = Normal operation, register TEST holds reset values. 1 = Test Mode, write access to register TEST enabled.
[6]	**DAR**	**Disable Automatic Retransmission** 0 = Automatic retransmission of messages not transmitted successfully enabled. 1 = Automatic retransmission disabled.
[5]	**MON**	**Bus Monitoring Mode** Bit MON can only be set by the Host when both CCE and INIT are set to 1. The bit can be reset by the Host at any time. 0 = Bus Monitoring Mode is disabled. 1 = Bus Monitoring Mode is enabled.
[4]	**CSR**	**Clock Stop Request** 0 = No clock stop is requested. 1 = Clock stop requested. When clock stop is requested, first INIT and then CSA will be set after all pending transfer requests have been completed and the CAN bus reached idle.
[3]	**CSA**	**Clock Stop Acknowledge** 0 = No clock stop acknowledged. 1 = The Controller may be set in power down by stopping AHB clock and CAN Core clock.
[2]	**ASM**	**Restricted Operation Mode** Bit ASM can only be set by the Host when both CCE and INIT are set to 1. The bit can be reset by the software at any time. This bit will be set automatically set to 1 when the Tx handler was not able to read data from the message RAM in time. For a description of the Restricted Operation Mode refer to **Restricted Operation Mode**. 0 = Normal CAN operation. 1 = Restricted Operation Mode active.
[1]	**CCE**	**Configuration Change Enable** 0 = The CPU has no write access to the protected configuration registers. 1 = The CPU has write access to the protected configuration registers (while INIT (CANFD_CCCR[0]) = 1).
[0]	**INIT**	**Initialization** 0 = Normal Operation. 1 = Initialization is started. **Note:** Due to the synchronization mechanism between the two clock domains, there may be a delay until the value written to INIT can be read back. Therefore the programmer has to assure that the previous value written to INIT has been accepted by reading INIT before setting INIT to a new value.

Nominal Bit Timing & Prescaler Register (CANFD_NBTP)

Register	Offset	R/W	Description	Reset Value
CANFD_NBTP	CANFDn_BA+0x01C	R	Nominal Bit Timing & Prescaler Register (P*)	0x0600_0A03

31	30	29	28	27	26	25	24
NSJW							NBRP
23	22	21	20	19	18	17	16
NBRP							
15	14	13	12	11	10	9	8
NTSEG1							
7	6	5	4	3	2	1	0
Reserved	NTSEG2						

Bits	Description	
[31:25]	**NSJW**	**Nominal Re-Synchronization Jump Width** Valid values are 0 to 127. Should be smaller than NTSEG2. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. t_{SJW} = (NSJW + 1) x tq.
[24:16]	**NBRP**	**Nominal Bit Rate Prescaler** The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Bit Rate Prescaler are 0 to 511. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used
[15:8]	**NTSEG1**	**Nominal Time segment before sample point** Valid values are 1 to 255. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. t_{BS1} = (NTSEG1 + 1) x tq.
[7]	**Reserved**	Reserved.
[6:0]	**NTSEG2**	**Nominal Time segment after sample point** 0x01-0x7F Valid values are 1 to 127. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. t_{BS2} = (NTSEG2 + 1) x tq. **Note:** With a CAN Core clock (cclk) of 8 MHz, the reset value of 0x06000A03 configures the controller for a bit rate of 500 kBit/s.

Timestamp Counter Configuration Register (CANFD_TSCC)

Register	Offset	R/W	Description	Reset Value
CANFD_TSCC	CANFDn_BA+0x020	R	Timestamp Counter Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				TCP			
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TSS	

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**TCP**	**Timestamp Counter Prescaler** Configures the timestamp and timeout counters time unit in multiples of CAN bit times [1…16]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
[15:2]	**Reserved**	Reserved.
[1:0]	**TSS**	**Timestamp Select** 00 = Timestamp counter value always 0x0000. 01 = Timestamp counter value incremented according to TCP. 10 = Reserved 11 = Same as “00”.

Timestamp Counter Register (CANFD_TSCV)

Register	Offset	R/W	Description	Reset Value
CANFD_TSCV	CANFDn_BA+0x024	R	Timestamp Counter Value (C*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TSC							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TSC							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TSC**	**Timestamp Counter** The internal Timestamp Counter value is captured on start of frame (both Rx and Tx). When TSS (CANFD_TSCC[[1:0]) = 2'b01, the Timestamp Counter is incremented in multiples of CAN bit times [1...16] depending on the configuration of TCP (CANFD_TSCC[19:16]). A wrap around sets interrupt flag TSW (CANFD_IR[16]). Write access resets the counter to zero. **Note:** A "wrap around" is a change of the Timestamp Counter value from non-zero to zero not caused by write access to CANFD_TSCV.

Timeout Counter Configuration Register (CANFD_TOCC)

Register	Offset	R/W	Description	Reset Value
CANFD_TOCC	CANFDn_BA+0x028	R	Timeout Counter Configuration (P*)	0xFFFF_0000

31	30	29	28	27	26	25	24
TOP							
23	22	21	20	19	18	17	16
TOP							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					TOS		ETOC

Bits	Description	
[31:16]	**TOP**	**Timeout Period** Start value of the Timeout Counter (down-counter)**.** Configures the Timeout Period.
[15:3]	**Reserved**	Reserved.
[2:1]	**TOS**	**Timeout Select** When operating in Continuous mode, a write to CANFD_TOCV presets the counter to the value configured by TOP (CANFD_TOCC[31:16]) and continues down-counting. When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by TOP (CANFD_TOCC[31:16]). Down-counting is started when the first FIFO element is stored. 00 = Continuous operation. 01 = Timeout controlled by Tx Event FIFO. 10 = Timeout controlled by Rx FIFO 0. 11 = Timeout controlled by Rx FIFO 1.
[0]	**ETOC**	**Enable Timeout Counter** 0 = Timeout Counter disabled. 1 = Timeout Counter enabled. **Note:** For use of timeout function with CAN FD refer to 6.36.5.3.

Timeout Counter Value Register (CANFD_TOCV)

Register	Offset	R/W	Description	Reset Value
CANFD_TOCV	CANFDn_BA+0x02C	R	Timeout Counter Value (C*)	0x0000_FFFF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TOC							
7	6	5	4	3	2	1	0
TOC							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TOC**	**Timeout Counter** The filed is decremented in multiples of CAN bit times [1...16] depending on the configuration of TCP (CANFD_TSCC[19:16]). When decremented to zero, interrupt flag TOO (CANFD_IR[18]) is set and the timeout counter is stopped. Start and reset/restart conditions are configured via TOS (CANFD_TOCC[1:0]).

Error Counter Register (CANFD_ECR)

Register	Offset	R/W	Description	Reset Value
CANFD_ECR	CANFDn_BA+0x040	R	Error Counter Register (X*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CEL							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RP	**REC**						
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TEC							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**CEL**	**CAN Error Logging** The counter is incremented each time when a CAN protocol error causes the 8-bit Transmit Error Counter TEC or the 7-bit Receive Error Counter REC to be incremented. The counter is also incremented when the Bus_Off limit is reached. It is not incremented when only RP is set without changing REC. The increment of CEL follows after the increment of REC or TEC. The counter is reset by read access to CEL. The counter stops at 0xFF; the next increment of TEC or REC sets interrupt flag ELO (CANFD_IR[22]).
[15]	**RP**	**Receive Error Passive** 0 = The Receive Error Counter is below the error passive level of 128. 1 = The Receive Error Counter has reached the error passive level of 128.
[14:8]	**REC**	**Receive Error Counter** Actual state of the Receive Error Counter, values between 0 and 127.
[7:0]	**TEC**	**Transmit Error Counter** Actual state of the Transmit Error Counter, values between 0 and 255. **Note:** When ASM (CANFD_CCCR[2]) is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented.

Protocol Status Register (CANFD_PSR)

Register	Offset	R/W	Description	Reset Value
CANFD_PSR	CANFDn_BA+0x044	R	Protocol Status Register (XS*)	0x0000_0707

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	TDCV						
15	14	13	12	11	10	9	8
Reserved	PXE	RFDF	RBRS	RESI	DLEC		
7	6	5	4	3	2	1	0
BO	EW	EP	ACT		LEC		

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22:16]	**TDCV**	**Transmitter Delay Compensation Value** Position of the secondary sample point, defined by the sum of the measured delay from CANx_TXD to CANx_RXD and TDCO (CANFD_TDCR[[14:8]). The SSP position is, in the data phase, the number of minimum time quata (mtq) between the start of the transmitted bit and the secondary sample point. Valid values are 0 to 127 mtq.
[15]	**Reserved**	Reserved.
[14]	**PXE**	**Protocol Exception Event** 0 = No protocol exception event occurred since last read access. 1 = Protocol exception event occurred.
[13]	**RFDF**	**Received a CAN FD Message** This bit is set independent of acceptance filtering. 0 = Since this bit was reset by the CPU, no CAN FD message has been received. 1 = Message in CAN FD format with FDF flag set has been received. **Note:** Byte access: Reading byte 0 will reset RFDF, reading bytes 3/2/1 has no impact.
[12]	**RBRS**	**BRS flag of last received CAN FD Message** This bit is set together with RFDF, independent of acceptance filtering. 0 = Last received CAN FD message did not have its BRS flag set. 1 = Last received CAN FD message had its BRS flag set. **Note:** Byte access: Reading byte 0 will reset RBRS, reading bytes 3/2/1 has no impact.
[11]	**RESI**	**ESI flag of last received CAN FD Message** This bit is set together with RFDF, independent of acceptance filtering. 0 = Last received CAN FD message did not have its ESI flag set. 1 = Last received CAN FD message had its ESI flag set.
[10:8]	**DLEC**	**Data Phase Last Error Code** Type of last error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. This field will be cleared to zero when a CAN FD format frame with its BRS flag set has been transferred (reception or transmission) without error.

[7]	**BO**	**Bus_Off Status** 0 = The CAN FD controller is not Bus_Off. 1 = The CAN FD controller is in Bus_Off state.
[6]	**EW**	**Warning Status** 0 = Both error counters are below the Error_Warning limit of 96. 1 = At least one of error counter has reached the Error_Warning limit of 96.
[5]	**EP**	**Error Passive** 0 = The CAN FD controller is in the Error_Active state. It normally takes part in bus communication and sends an active error flag when an error has been detected. 1 = The CAN FD controller is in the Error_Passive state.
[4:3]	**ACT**	**Activity** Monitors the module’s CAN communication state. 00 = Synchronizing - node is synchronizing on CAN communication. 01 = Idle - node is neither receiver nor transmitter. 10 = Receiver - node is operating as receiver. 11 = Transmitter - node is operating as transmitter.
[2:0]	**LEC**	**Last Error Code** The LEC indicates the type of the last error to occur on the CAN bus. This field will be cleared to 0 when a message has been transferred (reception or transmission) without error. 000 = No Error: No error occurred since LEC has been reset by successful reception or transmission. 001 = Stuff Error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed. 010 = Form Error: A fixed format part of a received frame has the wrong format. 011 = AckError: The message transmitted by the CANFD CONTROLLER was not acknowledged by another node. 100 = Bit1Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value 1), but the monitored bus value was dominant. 101 = Bit0Error : During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (data or identifier bit logical value 0), but the monitored bus value was recessive. During Bus_Off recovery this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the Bus_Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed). 110 = CRCError: The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match with the CRC calculated from the received data. 111 = NoChange: Any read access to the Protocol Status Register re-initializes the LEC to 7.When the LEC shows the value 7, no CAN bus event was detected since the last CPU read access to the Protocol Status Register.

Note: When a frame in CAN FD format has reached the data phase with BRS flag set, the next CAN event (error or valid frame) will be shown in DLEC instead of LEC. An error in a fixed stuff bit of a CAN FD CRC sequence will be shown as a Form Error, not Stuff Error.

Note: The Bus_Off recovery sequence (refer to ISO 11898-1:2015) cannot be shortened by setting or resetting INIT (CANFD_CCCR[0]). If the device goes Bus_Off, it will set INIT (CANFD_CCCR[0]) of its own accord, stopping all bus activities. Once INIT (CANFD_CCCR[0]) has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129 * 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus_Off recovery sequence, the Error Management Counters will be reset. During the waiting time after the resetting of INIT (CANFD_CCCR[0]), each time a sequence of 11 recessive bits has been monitored, a Bit0Error code is written to LEC (CANFD_PSR[2:0]), enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the Bus_Off recovery sequence. REC(CANFD_ECR[14:8]) is used to count these sequences.

Transmitter Delay Compensation Register (CANFD_TDCR)

Register	Offset	R/W	Description	Reset Value
CANFD_TDCR	CANFDn_BA+0x048	R	Transmitter Delay Compensation Register (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	TDCO						
7	6	5	4	3	2	1	0
Reserved	TDCF						

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:8]	**TDCO**	**Transmitter Delay Compensation SSP Offset** Offset value defining the distance between the measured delay from CANx_TXD to CANx_RXD and the secondary sample point. Valid values are 0 to 127 mtq.
[7]	**Reserved**	Reserved.
[6:0]	**TDCF**	**Transmitter Delay Compensation Filter Window Length** Defines the minimum value for the SSP position, dominant edges on CANx_RXD that would result in an earlier SSP position are ignored for transmitter delay measurement. The feature is enabled when TDCF is configured to a value greater than TDCO. Valid values are 0 to 127 mtq.

Interrupt Register (CANFD_IR)

The flags are set when one of the listed conditions is detected (edge-sensitive). A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect. A hard reset will clear the register.

Register	Offset	R/W	Description	Reset Value
CANFD_IR	CANFDn_BA+0x050	R/W	Interrupt Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		ARA	PED	PEA	WDI	BO	EW
23	22	21	20	19	18	17	16
EP	ELO	BEU	BEC	DRX	TOO	MRAF	TSW
15	14	13	12	11	10	9	8
TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM
7	6	5	4	3	2	1	0
RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**ARA**	**Access to Reserved Address** 0 = No access to reserved address occurred. 1 = Access to reserved address occurred.
[28]	**PED**	**Protocol Error in Data Phase** 0 = No protocol error in data phase. 1= Protocol error in data phase detected (DLEC (CANFD_PSR[10:8]) no equal 0 or 7). **Note:** Data bit time is used
[27]	**PEA**	**Protocol Error in Arbitration Phase** 0 = No protocol error in arbitration phase. 1 = Protocol error in arbitration phase detected (LEC (CANFD_PSR[2:0]) no equal 0 or 7). **Note:** Nominal bit time is used
[26]	**WDI**	**Watchdog Interrupt** 0 = No Message RAM Watchdog event occurred. 1 = Message RAM Watchdog event due to missing READY.
[25]	**BO**	**Bus_Off Status** 0 = Bus_Off status unchanged. 1 = Bus_Off status changed.
[24]	**EW**	**Warning Status** 0 = Error_Warning status unchanged. 1 = Error_Warning status changed.
[23]	**EP**	**Error Passive** 0 = Error_Passive status unchanged. 1 = Error_Passive status changed.

[22]	**ELO**	**Error Logging Overflow** 0 = CAN Error Logging Counter did not overflow. 1 = Overflow of CAN Error Logging Counter occurred.
[21]	**BEU**	**Bit Error Uncorrected** Message RAM bit error detected, uncorrected. Controlled by input signal m_can_aeim_berr[1] generated by an optional external parity / ECC logic attached to the Message RAM. An uncorrected Message RAM bit error sets CCCR.INIT to '1'. This is done to avoid transmission of corrupted data. 0 = No bit error detected when reading from Message RAM. 1 = Bit error detected, uncorrected (e.g. parity logic).
[20]	**BEC**	**Bit Error Corrected** Message RAM bit error detected and corrected. Controlled by input signal m_can_aeim_berr[0] generated by an optional external parity / ECC logic attached to the Message RAM. 0 = No bit error detected when reading from Message RAM. 1 = Bit error detected and corrected (e.g. ECC).
[19]	**DRX**	**Message stored to Dedicated Rx Buffer** The flag is set whenever a received message has been stored into a dedicated Rx Buffer. 0 = No Rx Buffer updated. 1 = At least one received message stored into an Rx Buffer.
[18]	**TOO**	**Timeout Occurred** 0 = No timeout. 1 = Timeout reached.
[17]	**MRAF**	**Message RAM Access Failure** The flag is set, when the Rx Handler • Has not completed acceptance filtering or storage of an accepted message until the arbitration field of the following message has been received. In this case acceptance filtering or message storage is aborted and the Rx Handler starts processing of the following message. • Was not able to write a message to the Message RAM. In this case message storage is aborted. In both cases the FIFO put index is not updated resp. the New Data flag for a dedicated Rx Buffer is not set, a partly stored message is overwritten when the next message is stored to this location. The flag is also set when the Tx Handler was not able to read a message from the Message RAM in time. In this case message transmission is aborted. In case of a Tx Handler access failure the CAN FD controller is switched into Restricted Operation Mode (refer to **Restricted Operation Mode**). To leave Restricted Operation Mode, the Host CPU has to reset ASM (CANFD_CCCR[2]). 0 = No Message RAM access failure occurred. 1 = Message RAM access failure occurred.
[16]	**TSW**	**Timestamp Wraparound** 0 = No timestamp counter wrap-around. 1 = Timestamp counter wrapped around.
[15]	**TEFL**	**Tx Event FIFO Element Lost** 0 = No Tx Event FIFO element lost. 1 = Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero.
[14]	**TEFF**	**Tx Event FIFO Full** 0 = Tx Event FIFO is not full. 1 = Tx Event FIFO is full.
[13]	**TEFW**	**Tx Event FIFO Watermark Reached** 0 = Tx Event FIFO fill level below watermark. 1 = Tx Event FIFO fill level reached watermark.

[12]	**TEFN**	**Tx Event FIFO New Entry** 0 = Tx Event FIFO unchanged. 1 = Tx Handler wrote Tx Event FIFO element.
[11]	**TFE**	**Tx FIFO Empty** 0 = Tx FIFO non-empty. 1 = Tx FIFO empty.
[10]	**TCF**	**Transmission Cancellation Finished** 0 = No transmission cancellation finished. 1 = Transmission cancellation finished.
[9]	**TC**	**Transmission Completed** 0 = No transmission completed. 1 = Transmission completed.
[8]	**HPM**	**High Priority Message** 0 = No high priority message received. 1 = High priority message received.
[7]	**RF1L**	**Rx FIFO 1 Message Lost** 0 = No Rx FIFO 1 message lost. 1 = Rx FIFO 1 message lost, also set after write attempt to Rx FIFO 1 of size zero.
[6]	**RF1F**	**Rx FIFO 1 Full** 0 = Rx FIFO 1 is not full. 1 = Rx FIFO 1 is full.
[5]	**RF1W**	**Rx FIFO 1 Watermark Reached** 0 = Rx FIFO 1 fill level below watermark. 1 = Rx FIFO 1 fill level reached watermark.
[4]	**RF1N**	**Rx FIFO 1 New Message** 0 = No new message written to Rx FIFO 1. 1 = New message written to Rx FIFO 1.
[3]	**RF0L**	**Rx FIFO 0 Message Lost** 0 = No Rx FIFO 0 message lost. 1 = Rx FIFO 0 message lost, also set after write attempt to Rx FIFO 0 of size zero.
[2]	**RF0F**	**Rx FIFO 0 Full** 0 = Rx FIFO 0 is not full. 1 = Rx FIFO 0 is full.
[1]	**RF0W**	**Rx FIFO 0 Watermark Reached** 0 = Rx FIFO 0 fill level below watermark. 1 = Rx FIFO 0 fill level reached watermark.
[0]	**RF0N**	**Rx FIFO 0 New Message** 0 = No new message written to Rx FIFO 0. 1 = New message written to Rx FIFO 0.

Interrupt Enable Register (CANFD_IE)

Register	Offset	R/W	Description	Reset Value
CANFD_IE	CANFDn_BA+0x054	R/W	Interrupt Enable	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		ARAE	PEDE	PEAE	WDIE	BOE	EWE
23	22	21	20	19	18	17	16
EPE	ELOE	BEUE	BECE	DRXE	TOOE	MRAFE	TSWE
15	14	13	12	11	10	9	8
TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME
7	6	5	4	3	2	1	0
RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**ARAE**	**Access to Reserved Address Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[28]	**PEDE**	**Protocol Error in Data Phase Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[27]	**PEAE**	**Protocol Error in Arbitration Phase Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[26]	**WDIE**	**Watchdog Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[25]	**BOE**	**Bus_Off Status Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[24]	**EWE**	**Warning Status Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[23]	**EPE**	**Error Passive Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[22]	**ELOE**	**Error Logging Overflow Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[21]	**BEUE**	**Bit Error Uncorrected Interrupt Enable**

		0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[20]	**BECE**	**Bit Error Corrected Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[19]	**DRXE**	**Message stored to Dedicated Rx Buffer Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[18]	**TOOE**	**Timeout Occurred Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[17]	**MRAFE**	**Message RAM Access Failure Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[16]	**TSWE**	**Timestamp Wraparound Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[15]	**TEFLE**	**Tx Event FIFO Event Lost Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[14]	**TEFFE**	**Tx Event FIFO Full Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[13]	**TEFWE**	**Tx Event FIFO Watermark Reached Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[12]	**TEFNE**	**Tx Event FIFO New Entry Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[11]	**TFEE**	**Tx FIFO Empty Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[10]	**TCFE**	**Transmission Cancellation Finished Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[9]	**TCE**	**Transmission Completed Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[8]	**HPME**	**High Priority Message Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[7]	**RF1LE**	**Rx FIFO 1 Message Lost Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.

[6]	**RF1FE**	**Rx FIFO 1 Full Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[5]	**RF1WE**	**Rx FIFO 1 Watermark Reached Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[4]	**RF1NE**	**Rx FIFO 1 New Message Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[3]	**RF0LE**	**Rx FIFO 0 Message Lost Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[2]	**RF0FE**	**Rx FIFO 0 Full Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[1]	**RF0WE**	**Rx FIFO 0 Watermark Reached Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.
[0]	**RF0NE**	**Rx FIFO 0 New Message Interrupt Enable** 0 = Interrupt is Disabled. 1 = Interrupt is Enabled.

Interrupt Line Select Register (CANFD_ILS)

This register assigns an interrupt generated by a specific interrupt flag from the CANFD_IR register to one of the two module interrupt lines.

Register	Offset	R/W	Description	Reset Value
CANFD_ILS	CANFDn_BA+0x058	R/W	Interrupt Line Select	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		ARAL	PEDL	PEAL	WDIL	BOL	EWL
23	22	21	20	19	18	17	16
EPL	ELOL	BEUL	BECL	DRXL	TOOL	MRAFL	TSWL
15	14	13	12	11	10	9	8
TEFLL	TEFFL	TEFWL	TEFNL	TFEL	TCFL	TCL	HPML
7	6	5	4	3	2	1	0
RF1LL	RF1FL	RF1WL	RF1NL	RF0LL	RF0FL	RF0WL	RF0NL

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**ARAL**	**Access to Reserved Address Line**
[28]	**PEDL**	**Protocol Error in Data Phase Line**
[27]	**PEAL**	**Protocol Error in Arbitration Phase Line**
[26]	**WDIL**	**Watchdog Interrupt Line**
[25]	**BOL**	**Bus_Off Status Interrupt Line**
[24]	**EWL**	**Warning Status Interrupt Line**
[23]	**EPL**	**Error Passive Interrupt Line**
[22]	**ELOL**	**Error Logging Overflow Interrupt Line**
[21]	**BEUL**	**Bit Error Uncorrected Interrupt Line** .
[20]	**BECL**	**Bit Error Corrected Interrupt Line** .
[19]	**DRXL**	**Message stored to Dedicated Rx Buffer Interrupt Line**
[18]	**TOOL**	**Timeout Occurred Interrupt Line**
[17]	**MRAFL**	**Message RAM Access Failure Interrupt Line**
[16]	**TSWL**	**Timestamp Wraparound Interrupt Line**
[15]	**TEFLL**	**Tx Event FIFO Event Lost Interrupt Line**
[14]	**TEFFL**	**Tx Event FIFO Full Interrupt Line**
[13]	**TEFWL**	**Tx Event FIFO Watermark Reached Interrupt Line**

[12]	**TEFNL**	**Tx Event FIFO New Entry Interrupt Line**
[11]	**TFEL**	**Tx FIFO Empty Interrupt Line**
[10]	**TCFL**	**Transmission Cancellation Finished Interrupt Line**
[9]	**TCL**	**Transmission Completed Interrupt Line**
[8]	**HPML**	**High Priority Message Interrupt Line**
[7]	**RF1LL**	**Rx FIFO 1 Message Lost Interrupt Line**
[6]	**RF1FL**	**Rx FIFO 1 Full Interrupt Line**
[5]	**RF1WL**	**Rx FIFO 1 Watermark Reached Interrupt Line**
[4]	**RF1NL**	**Rx FIFO 1 New Message Interrupt Line**
[3]	**RF0LL**	**Rx FIFO 0 Message Lost Interrupt Line**
[2]	**RF0FL**	**Rx FIFO 0 Full Interrupt Line**
[1]	**RF0WL**	**Rx FIFO 0 Watermark Reached Interrupt Line**
[0]	**RF0NL**	**Rx FIFO 0 New Message Interrupt Line** 0 = Interrupt assigned to CAN interrupt line 0. 1 = Interrupt assigned to CAN interrupt line 1.

Interrupt Line Register (CANFD_ILE)

Register	Offset	R/W	Description	Reset Value
CANFD_ILE	CANFDn_BA+0x05C	R/W	Interrupt Line Enable	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**EINT1**	**EINT0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**EINT1**	**Enable Interrupt Line 1** 0 = Interrupt line m_can_int1 disabled. 1 = Interrupt line m_can_int1 enabled.
[0]	**EINT0**	**Enable Interrupt Line 0** 0 = Interrupt line m_can_int0 disabled. 1 = Interrupt line m_can_int0 enabled.

Global Filter Configuration Register (CANFD_GFC)

Register	Offset	R/W	Description	Reset Value
CANFD_GFC	CANFDn_BA+0x080	R	Global Filter Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		ANFS		ANFE		RRFS	RRFE

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:4]	**ANFS**	**Accept Non-matching Frames Standard** Defines how received messages with 11-bit IDs that do not match any element of the filter list are treated. 00 = Accept in Rx FIFO 0. 01 = Accept in Rx FIFO 1. 10 = Reject. 11 = Reject.
[3:2]	**ANFE**	**Accept Non-matching Frames Extended** Defines how received messages with 29-bit IDs that do not match any element of the filter list are treated. 00 = Accept in Rx FIFO 0. 01 = Accept in Rx FIFO 1. 10 = Reject. 11 = Reject.
[1]	**RRFS**	**Reject Remote Frames Standard** 0= Filter remote frames with 11-bit standard IDs. 1= Reject all remote frames with 11-bit standard IDs.
[0]	**RRFE**	**Reject Remote Frames Extended** 0= Filter remote frames with 29-bit extended IDs. 1= Reject all remote frames with 29-bit extended IDs.

Standard ID Filter Configuration Register (CANFD_SIDFC)

Register	Offset	R/W	Description	Reset Value
CANFD_SIDFC	CANFDn_BA+0x084	R	Standard ID Filter Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
LSS							
15	14	13	12	11	10	9	8
FLSSA							
7	6	5	4	3	2	1	0
FLSSA						Reserved	

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**LSS**	**List Size Standard** 0= No standard Message ID filter. 1-128 = Number of standard Message ID filter elements. >128= Values greater than 128 are interpreted as 128.
[15:2]	**FLSSA**	**Filter List Standard Start Address** Start address of standard Message ID filter list (32-bit word address, refer to Figure 6.36-11).
[1:0]	**Reserved**	Reserved.

Extended ID Filter Configuration Register (CANFD_XIDFC)

Register	Offset	R/W	Description	Reset Value
CANFD_XIDFC	CANFDn_BA+0x088	R	Extended ID Filter Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	LSE						
15	14	13	12	11	10	9	8
FLESA							
7	6	5	4	3	2	1	0
FLESA						Reserved	

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22:16]	**LSE**	**List Size Extended** 0= No extended Message ID filter. 1-64= Number of extended Message ID filter elements. >64= Values greater than 64 are interpreted as 64.
[15:2]	**FLESA**	**Filter List Extended Start Address** Start address of extended Message ID filter list (32-bit word address, refer to Figure 6.36-11)
[1:0]	**Reserved**	Reserved.

Extended ID AND Mask Register (CANFD_XIDAM)

Register	Offset	R/W	Description	Reset Value
CANFD_XIDAM	CANFDn_BA+0x090	R	Extended ID AND Mask (P*)	0x1FFF_FFFF

31	30	29	28	27	26	25	24
Reserved			EIDM				
23	22	21	20	19	18	17	16
EIDM							
15	14	13	12	11	10	9	8
EIDM							
7	6	5	4	3	2	1	0
EIDM							

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:0]	**EIDM**	**Extended ID Mask** For acceptance filtering of extended frames the Extended ID AND Mask is ANDed with the Message ID of a received frame. Intended for masking of 29-bit IDs in SAE J1939. With the reset value of all bits set to one the mask is not active.

High Priority Message Status Register (CANFD_HPMS)

Register	Offset	R/W	Description	Reset Value
CANFD_HPMS	CANFDn_BA+0x094	R	High Priority Message Status	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
FLST	FIDX						
7	6	5	4	3	2	1	0
MSI		BIDX					

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**FLST**	**Filter List** Indicates the filter list of the matching filter element. 0 = Standard Filter List. 1 = Extended Filter List.
[14:8]	**FIDX**	**Filter Index** Index of matching filter element. Range is 0 to LSS (CANFD_SIDFC[23:16]) - 1 or LSE (CANFD_XIDFC[22:16]) – 1
[7:6]	**MSI**	**Message Storage Indicator** 00 = No FIFO selected. 01 = FIFO message lost. 10 = Message stored in FIFO 0. 11 = Message stored in FIFO 1.
[5:0]	**BIDX**	**Buffer Index** Index of Rx FIFO element to which the message was stored. Only valid when MSI[1] = 1.

New Data 1 Register (CANFD_NDAT1)

Register	Offset	R/W	Description	Reset Value
CANFD_NDAT1	CANFDn_BA+0x098	R/W	New Data 1	0x0000_0000

31	30	29	28	27	26	25	24
NDn							
23	22	21	20	19	18	17	16
NDn							
15	14	13	12	11	10	9	8
NDn							
7	6	5	4	3	2	1	0
NDn							

Bits	Description	
[31:0] n=0,1..31	NDn	**New Data** The register holds the New Data flags of Rx Buffers 0 to 31. The flags are set when the respective Rx Buffer has been updated from a received frame. The flags remain set until the Host clears them. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect. A hard reset will clear the register. 0 = Rx Buffer not updated. 1 = Rx Buffer updated from new message.

New Data 2 Register (CANFD_NDAT2)

Register	Offset	R/W	Description	Reset Value
CANFD_NDAT2	CANFDn_BA+0x09C	R/W	New Data 2	0x0000_0000

31	30	29	28	27	26	25	24
NDn							
23	22	21	20	19	18	17	16
NDn							
15	14	13	12	11	10	9	8
NDn							
7	6	5	4	3	2	1	0
NDn							

Bits	Description	
[31:0] n=32,33..63	NDn	**New Data** The register holds the New Data flags of Rx Buffers 32 to 63. The flags are set when the respective Rx Buffer has been updated from a received frame. The flags remain set until the Host clears them. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect. A hard reset will clear the register. 0 = Rx Buffer not updated. 1 = Rx Buffer updated from new message.

Rx FIFO 0 Configuration Register (CANFD_RXF0C)

Register	Offset	R/W	Description	Reset Value
CANFD_RXF0C	CANFDn_BA+0x0A0	R	Rx FIFO 0 Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
F0OM	F0WM						
23	22	21	20	19	18	17	16
Reserved	F0S						
15	14	13	12	11	10	9	8
F0SA							
7	6	5	4	3	2	1	0
F0SA						Reserved	

Bits	Description	
[31]	**F0OM**	**FIFO 0 Operation Mode** FIFO 0 can be operated in blocking or in overwrite mode (refer to **Rx FIFOs**). 0 = FIFO 0 blocking mode. 1 = FIFO 0 overwrite mode.
[30:24]	**F0WM**	**Rx FIFO 0 Watermark** 0= Watermark interrupt disabled. 1-64 = Level for Rx FIFO 0 watermark interrupt (RF0W(CANFD_IR[1]). >64 = Watermark interrupt disabled.
[23]	**Reserved**	Reserved.
[22:16]	**F0S**	**Rx FIFO 0 Size** 0= No Rx FIFO 0. 1-64= Number of Rx FIFO 0 elements. >64= Values greater than 64 are interpreted as 64. The Rx FIFO 0 elements are indexed from 0 to F0S-1
[15:2]	**F0SA**	**Rx FIFO 0 Start Address** Start address of Rx FIFO 0 in Message RAM (32-bit word address).
[1:0]	**Reserved**	Reserved.

Rx FIFO 0 Status Register (CANFD_RXF0S)

Register	Offset	R/W	Description	Reset Value
CANFD_RXF0S	CANFDn_BA+0x0A4	R	Rx FIFO 0 Status	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						RF0L	F0F
23	22	21	20	19	18	17	16
Reserved		F0PI					
15	14	13	12	11	10	9	8
Reserved		F0GI					
7	6	5	4	3	2	1	0
Reserved	F0FL						

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**RF0L**	**Rx FIFO 0 Message Lost** This bit is a copy of interrupt flagRF0L (CANFD_IR[3]). When CANFD_IR.RF0L is reset, this bit is also reset. 0 = No Rx FIFO 0 message lost. 1 = Rx FIFO 0 message lost, also set after write attempt to Rx FIFO 0 of size zero. **Note:** Overwriting the oldest message when F0OM (CANFD_RXF0C[31]) = 1 will not set this flag.
[24]	**F0F**	**Rx FIFO 0 Full** 0= Rx FIFO 0 is not full. 1= Rx FIFO 0 is full.
[23:22]	**Reserved**	Reserved.
[21:16]	**F0PI**	**Rx FIFO 0 Put Index** Rx FIFO 0 write index pointer, range 0 to 63.
[15:14]	**Reserved**	Reserved.
[13:8]	**F0GI**	**Rx FIFO 0 Get Index** Rx FIFO 0 read index pointer, range 0 to 63.
[7]	**Reserved**	Reserved.
[6:0]	**F0FL**	**Rx FIFO 0 Fill Level** Number of elements stored in Rx FIFO 0, range 0 to 64

Rx FIFO 0 Acknowledge Register (CANFD_RXF0A)

Register	Offset	R/W	Description	Reset Value
CANFD_RXF0A	CANFDn_BA+0x0A8	R/W	Rx FIFO 0 Acknowledge	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**F0A**					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**F0A**	**Rx FIFO 0 Acknowledge Index** After the Host has read a message or a sequence of messages from Rx FIFO 0 it has to write the buffer index of the last element read from Rx FIFO 0 to F0AI. This will set the Rx FIFO 0 Get Index F0GI (CANFD_RXF0S[13:8]) to F0AI (CANFD_RXF0A[5:0]) + 1 and update the FIFO 0 Fill Level F0FL(CANFD_RXF0S[6:0]).

Rx Buffer Configuration Register (CANFD_RXBC)

Register	Offset	R/W	Description	Reset Value
CANFD_RXBC	CANFDn_BA+0x0AC	R	Rx Buffer Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RBSA							
7	6	5	4	3	2	1	0
RBSA						Reserved	

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:2]	**RBSA**	**Rx Buffer Start Address** Configures the start address of the Rx Buffers section in the Message RAM (32-bit word address).
[1:0]	**Reserved**	Reserved.

Rx FIFO 1 Configuration Register (CANFD_RXF1C)

Register	Offset	R/W	Description	Reset Value
CANFD_RXF1C	CANFDn_BA+0x0B0	R	Rx FIFO 1 Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
F1OM	F1WM						
23	22	21	20	19	18	17	16
Reserved	F1S						
15	14	13	12	11	10	9	8
F1SA							
7	6	5	4	3	2	1	0
F1SA						Reserved	

Bits	Description	
[31]	**F1OM**	**FIFO 1 Operation Mode** FIFO 1 can be operated in blocking or in overwrite mode (refer to **Rx FIFOs**). 0= FIFO 1 blocking mode. 1= FIFO 1 overwrite mode.
[30:24]	**F1WM**	**Rx FIFO 1 Watermark** 0= Watermark interrupt disabled. 1-64 = Level for Rx FIFO 1 watermark interrupt (RF1W(CANFD_IR[5])). >64 = Watermark interrupt disabled.
[23]	**Reserved**	Reserved.
[22:16]	**F1S**	**Rx FIFO 1 Size** 0= No Rx FIFO 1. 1-64 = Number of Rx FIFO 1 elements. >64 = Values greater than 64 are interpreted as 64. The Rx FIFO 1 elements are indexed from 0 to F1S - 1
[15:2]	**F1SA**	**Rx FIFO 1 Start Address** Start address of Rx FIFO 1 in Message RAM (32-bit word address, refer to Figure 6.36-11).
[1:0]	**Reserved**	Reserved.

Rx FIFO 1 Status Register (CANFD_RXF1S)

Register	Offset	R/W	Description	Reset Value
CANFD_RXF1S	CANFDn_BA+0x0B4	R	Rx FIFO 1 Status	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						RF1L	F1F
23	22	21	20	19	18	17	16
Reserved		F1PI					
15	14	13	12	11	10	9	8
Reserved		F1GI					
7	6	5	4	3	2	1	0
Reserved	F1FL						

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**RF1L**	**Rx FIFO 1 Message Lost** This bit is a copy of interrupt flag RF1L(CANFD_IR[7]). When RF1L(CANFD_IR[7]) is reset, this bit is also reset. 0= No Rx FIFO 1 message lost. 1= Rx FIFO 1 message lost, also set after write attempt to Rx FIFO 1 of size zero. **Note:** Overwriting the oldest message when F1OM (CANFD_RXF1C[31]) = 1 will not set this flag.
[24]	**F1F**	**Rx FIFO 1 Full** 0 = Rx FIFO 1 is not full. 1 = Rx FIFO 1 is full.
[23:22]	**Reserved**	**Rx FIFO 1 Get Index** Rx FIFO 1 read index pointer, range 0 to 63.
[21:16]	**F1PI**	**Rx FIFO 1 Fill Level** Number of elements stored in Rx FIFO 1, range 0 to 64.
[15:14]	**Reserved**	Reserved.
[13:8]	**F1GI**	**Rx FIFO 1 Get Index** Rx FIFO 1 read index pointer, range 0 to 63.
[7]	**Reserved**	Reserved.
[6:0]	**F1FL**	**Rx FIFO 1 Fill Level** Number of elements stored in Rx FIFO 1, range 0 to 64

Rx FIFO 1 Acknowledge Register (CANFD_RXF1A)

Register	Offset	R/W	Description	Reset Value
CANFD_RXF1A	CANFDn_BA+0x0B8	R/W	Rx FIFO 1 Acknowledge	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		F1AI					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**F1AI**	**Rx FIFO 1 Acknowledge Index** After the Host has read a message or a sequence of messages from Rx FIFO 1 it has to write the buffer index of the last element read from Rx FIFO 1 to F1AI. This will set the Rx FIFO 1 Get Index F1GI (CANFD_RXF1S[13:8]) to F1AI (CANFD_RXF1A[5:0]) + 1 and update the FIFO 1 Fill Level F1FL (CANFD_RXF1S[6:0]).

Rx Buffer / FIFO Element Size Configuration Register (CANFD_RXESC)

Register	Offset	R/W	Description	Reset Value
CANFD_RXESC	CANFDn_BA+0x0BC	R	Rx Buffer / FIFO Element Size Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					RBDS		
7	6	5	4	3	2	1	0
Reserved	F1DS			Reserved	F0DS		

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**RBDS**	**Rx Buffer Data Field Size** 000 = 8 byte data field. 001 = 12 byte data field. 010 = 16 byte data field. 011 = 20 byte data field. 100 = 24 byte data field. 101 = 32 byte data field. 110 = 48 byte data field. 111 = 64 byte data field.
[7]	**Reserved**	Reserved.
[6:4]	**F1DS**	**Rx FIFO 1 Data Field Size** 000 = 8 byte data field. 001 = 12 byte data field. 010 = 16 byte data field. 011 = 20 byte data field. 100 = 24 byte data field. 101 = 32 byte data field. 110 = 48 byte data field. 111 = 64 byte data field.
[3]	**Reserved**	Reserved.
[2:0]	**F0DS**	**Rx FIFO 0 Data Field Size** 000 = 8 byte data field. 001 = 12 byte data field. 010 = 16 byte data field. 011 = 20 byte data field. 100 = 24 byte data field.

		101 = 32 byte data field. 110 = 48 byte data field. 111 = 64 byte data field. **Note:** In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Rx Buffer or Rx FIFO, only the number of bytes as configured by CANFD_RXESC are stored to the Rx Buffer resp. Rx FIFO element. The rest of the frame data field is ignored.

Tx Buffer Configuration Register (CANFD_TXBC)

Register	Offset	R/W	Description	Reset Value
CANFD_TXBC	CANFDn_BA+0x0C0	R	Tx Buffer Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	TFQM	TFQS					
23	22	21	20	19	18	17	16
Reserved		NDTB					
15	14	13	12	11	10	9	8
TBSA							
7	6	5	4	3	2	1	0
TBSA						Reserved	

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**TFQM**	**Tx FIFO/Queue Mode** 0= Tx FIFO operation. 1= Tx Queue operation.
[29:24]	**TFQS**	**Transmit FIFO/Queue Size** 0= No Tx FIFO/Queue. 1-32= Number of Tx Buffers used for Tx FIFO/Queue. >32= Values greater than 32 are interpreted as 32.
[23:22]	**Reserved**	Reserved.
[21:16]	**NDTB**	**Number of Dedicated Transmit Buffers** 0= No Dedicated Tx Buffers. 1-32= Number of Dedicated Tx Buffers. >32= Values greater than 32 are interpreted as 32.
[15:2]	**TBSA**	**Tx Buffers Start Address** Start address of Tx Buffers section in Message RAM (32-bit word address, refer to Figure 6.36-11). **Note:** The sum of TFQS and NDTB may be not greater than 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers
[1:0]	**Reserved**	Reserved.

Tx FIFO / Queue Status Register (CANFD_TXFQS)

Register	Offset	R/W	Description	Reset Value
CANFD_TXFQS	CANFDn_BA+0x0C4	R	Tx FIFO/Queue Status	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		TFQF	TFQPI				
15	14	13	12	11	10	9	8
Reserved			TFGI				
7	6	5	4	3	2	1	0
Reserved		TFFL					

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21]	**TFQF**	**Tx FIFO/Queue Full** 0= Tx FIFO/Queue is not full. 1= Tx FIFO/Queue is full.
[20:16]	**TFQPI**	**Tx FIFO/Queue Put Index** Tx FIFO/Queue write index pointer, range 0 to 31.
[15:13]	**Reserved**	Reserved.
[12:8]	**TFGI**	**Tx FIFO Get Index** Tx FIFO read index pointer, range 0 to 31. Read as zero when Tx Queue operation is configured (TFQM (CANFD_TXBC[30]) = 1).
[7:6]	**Reserved**	Reserved.
[5:0]	**TFFL**	**Tx FIFO Free Level** Number of consecutive free Tx FIFO elements starting from TFGI, range 0 to 32. Read as zero when Tx Queue operation is configured (TFQM (CANFD_TXBC[30]) = 1)**.** **Note:** In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers. Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO.

Tx Buffer Element Size Configuration Register (CANFD_TXESC)

Register	Offset	R/W	Description	Reset Value
CANFD_TXESC	CANFDn_BA+0x0C8	R	Tx Buffer Element Size Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**TBDS**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**TBDS**	**Tx Buffer Data Field Size** 000 = 8 byte data field. 001 = 12 byte data field. 010 = 16 byte data field. 011 = 20 byte data field. 100 = 24 byte data field. 101 = 32 byte data field. 110 = 48 byte data field. 111 = 64 byte data field. **Note:** In case the data length code DLC of a Tx Buffer element is configured to a value higher than the Tx Buffer data field size TBDS(CANFD_TXESC[2:0]), the bytes not defined by the Tx Buffer are transmitted as 0xCC (padding bytes).

Tx Buffer Request Pending Register (CANFD_TXBRP)

Register	Offset	R/W	Description	Reset Value
CANFD_TXBRP	CANFDn_BA+0x0CC	R	Tx Buffer Request Pending	0x0000_0000

31	30	29	28	27	26	25	24
TRPn							
23	22	21	20	19	18	17	16
TRPn							
15	14	13	12	11	10	9	8
TRPn							
7	6	5	4	3	2	1	0
TRPn							

Bits	Description	
[31:0]	TRPn	**Transmission Request Pending** **Each Tx Buffer has its own Transmission Request Pending bit The bits are set via register CANFD_TXBAR The bits are reset after a requested transmission has completed or has been cancelled via register CANFD_TXBCR** CANFD_TXBRP bits are set only for those Tx Buffers configured via CANFD_TXBC. After a CANFD_TXBRP bit has been set, a Tx scan (refer to 6.36.5.5, Tx Handling) is started to check for the pending Tx request with the highest priority (Tx Buffer with lowest Message ID). A cancellation request resets the corresponding transmission request pending bit of register CANFD_TXBRP. In case a transmission has already been started when a cancellation is requested, this is done at the end of the transmission, regardless whether the transmission was successful or not. The cancellation request bits are reset directly after the corresponding CANFD_TXBRP bit has been reset. After a cancellation has been requested, a finished cancellation is signaled via CANFD_TXBCF • after successful transmission together with the corresponding CANFD_TXBTO bit • when the transmission has not yet been started at the point of cancellation • when the transmission has been aborted due to lost arbitration • when an error occurred during frame transmission In DAR mode all transmissions are automatically cancelled if they are not successful. The corresponding CANFD_TXBCF bit is set for all unsuccessful transmissions. 0 = No transmission request pending. 1 = Transmission request pending. **Note:** CANFD_TXBRP bits which are set while a Tx scan is in progress are not considered during this particular Tx scan. In case a cancellation is requested for such a Tx Buffer, this Add Request is cancelled immediately, the corresponding CANFD_TXBRP bit is reset.

Tx Buffer Add Request Register (CANFD_TXBAR)

Register	Offset	R/W	Description	Reset Value
CANFD_TXBAR	CANFDn_BA+0x0D0	R/W	Tx Buffer Add Request	0x0000_0000

31	30	29	28	27	26	25	24
ARn							
23	22	21	20	19	18	17	16
ARn							
15	14	13	12	11	10	9	8
ARn							
7	6	5	4	3	2	1	0
ARn							

Bits	Description	
[31:0]	ARn	**Add Request** **Each Tx Buffer has its own Add Request bit Writing a 1 will set the corresponding Add Request bit; writing a 0 has no impact This enables the Host to set transmission requests for multiple Tx Buffers with one write to CANFD_TXBAR CANFD_TXBAR bits are set only for those Tx Buffers configured via CANFD_TXBC When no Tx scan is running, the bits are reset immediately, else the bits remain set until the Tx scan process has completed** 0 = No transmission request added. 1 = Transmission requested added. **Note:** If an add request is applied for a Tx Buffer with pending transmission request (corresponding CANFD_TXBRP bit already set), this add request is ignored.

Tx Buffer Cancellation Request Register (CANFD_TXBCR)

Register	Offset	R/W	Description	Reset Value
CANFD_TXBCR	CANFDn_BA+0x0D4	R/W	Tx Buffer Cancellation Request	0x0000_0000

31	30	29	28	27	26	25	24
CRn							
23	22	21	20	19	18	17	16
CRn							
15	14	13	12	11	10	9	8
CRn							
7	6	5	4	3	2	1	0
CRn							

Bits	Description	
[31:0]	CRn	**Cancellation Request** Each Tx Buffer has its own Cancellation Request bit. Writing a 1 will set the corresponding Cancellation Request bit; writing a 0 has no impact. This enables the Host to set cancellation requests for multiple Tx Buffers with one write to CANFD_TXBCR. CANFD_TXBCR bits are set only for those Tx Buffers configured via CANFD_TXBC. The bits remain set until the corresponding bit of CANFD_TXBRP is reset. 0 = No cancellation pending. 1 = Cancellation pending.

Tx Buffer Transmission Occurred Register (CANFD_TXBTO)

Register	Offset	R/W	Description	Reset Value
CANFD_TXBTO	CANFDn_BA+0x0D8	R	Tx Buffer Transmission Occurred	0x0000_0000

31	30	29	28	27	26	25	24
TOn							
23	22	21	20	19	18	17	16
TOn							
15	14	13	12	11	10	9	8
TOn							
7	6	5	4	3	2	1	0
TOn							

Bits	Description	
[31:0]	TOn	**Transmission Occurred** Each Tx Buffer has its own Transmission Occurred bit. The bits are set when the corresponding CANFD_TXBRP bit is cleared after a successful transmission. The bits are reset when a new transmission is requested by writing a 1 to the corresponding bit of register CANFD_TXBAR. 0 = No transmission occurred. 1 = Transmission occurred.

Tx Buffer Cancellation Finished Register (CANFD_TXBCF)

Register	Offset	R/W	Description	Reset Value
CANFD_TXBCF	CANFDn_BA+0x0DC	R	Tx Buffer Cancellation Finished	0x0000_0000

31	30	29	28	27	26	25	24
CFn							
23	22	21	20	19	18	17	16
CFn							
15	14	13	12	11	10	9	8
CFn							
7	6	5	4	3	2	1	0
CFn							

Bits	Description	
[31:0]	CFn	**Cancellation Finished** Each Tx Buffer has its own Cancellation Finished bit. The bits are set when the corresponding CANFD_TXBRP bit is cleared after a cancellation was requested via CANFD_TXBCR. In case the corresponding CANFD_TXBRP bit was not set at the point of cancellation, CF is set immediately. The bits are reset when a new transmission is requested by writing a 1 to the corresponding bit of register CANFD_TXBAR. 0 = No transmit buffer cancellation. 1 = Transmit buffer cancellation finished.

Tx Buffer Transmission Interrupt Enable Register (CANFD_TXBTIE)

Register	Offset	R/W	Description	Reset Value
CANFD_TXBTIE	CANFDn_BA+0x0E0	R/W	Tx Buffer Transmission Interrupt Enable	0x0000_0000

31	30	29	28	27	26	25	24
TIEn							
23	22	21	20	19	18	17	16
TIEn							
15	14	13	12	11	10	9	8
TIEn							
7	6	5	4	3	2	1	0
TIEn							

Bits	Description	
[31:0]	TIEn	**Transmission Interrupt Enable** Each Tx Buffer has its own Transmission Interrupt enable bit. 0 = Transmission interrupt disabled. 1 = Transmission interrupt enable.

Tx Buffer Cancellation Finished Interrupt Enable Register (CANFD_TXBCIE)

Register	Offset	R/W	Description	Reset Value
CANFD_TXBCIE	CANFDn_BA+0x0E4	R/W	Tx Buffer Cancellation Finished Interrupt Enable	0x0000_0000

31	30	29	28	27	26	25	24
CFIEn							
23	22	21	20	19	18	17	16
CFIEn							
15	14	13	12	11	10	9	8
CFIEn							
7	6	5	4	3	2	1	0
CFIEn							

Bits	Description	
[31:0]	**CFIEn**	**Cancellation Finished Interrupt Enable** Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 = Cancellation finished interrupt disabled. 1 = Cancellation finished interrupt enabled.

Tx Event FIFO Configuration Register (CANFD_TXEFC)

Register	Offset	R/W	Description	Reset Value
CANFD_TXEFC	CANFDn_BA+0x0F0	R	Tx Event FIFO Configuration (P*)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		EFWM					
23	22	21	20	19	18	17	16
Reserved		EFS					
15	14	13	12	11	10	9	8
EFSA							
7	6	5	4	3	2	1	0
EFSA						Reserved	

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**EFWM**	**Event FIFO Watermark** 0 = Watermark interrupt disabled. 1-32= Level for Tx Event FIFO watermark interrupt (TEFW (CANFD_IR[13])). >32= Watermark interrupt disabled.
[23:22]	**Reserved**	Reserved.
[21:16]	**EFS**	**Event FIFO Size** 0= Tx Event FIFO disabled. 1-32= Number of Tx Event FIFO elements. >32= Values greater than 32 are interpreted as 32. The Tx Event FIFO elements are indexed from 0 to EFS - 1
[15:2]	**EFSA**	**Event FIFO Start Address** Start address of Tx Event FIFO in Message RAM (32-bit word address, refer to Figure 6.36-11).
[1:0]	**Reserved**	Reserved.

Tx Event FIFO Status Register (CANFD_TXEFS)

Register	Offset	R/W	Description	Reset Value
CANFD_TXEFS	CANFDn_BA+0x0F4	R	Tx Event FIFO Status	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						TEFL	EFF
23	22	21	20	19	18	17	16
Reserved			EFPI				
15	14	13	12	11	10	9	8
Reserved			EFGI				
7	6	5	4	3	2	1	0
Reserved		EFFL					

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**TEFL**	**Tx Event FIFO Element Lost** This bit is a copy of interrupt flag TEFL (CANFD_IR[15]). When TEFL is reset, this bit is also reset. 0= No Tx Event FIFO element lost. 1= Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero.
[24]	**EFF**	**Event FIFO Full** 0= Tx Event FIFO is not full. 1= Tx Event FIFO is full.
[23:21]	**Reserved**	Reserved.
[20:16]	**EFPI**	**Event FIFO Put Index** Tx Event FIFO write index pointer, range 0 to 31
[15:13]	**Reserved**	Reserved.
[12:8]	**EFGI**	**Event FIFO Get Index** Tx Event FIFO read index pointer, range 0 to 31
[7:6]	**Reserved**	Reserved.
[5:0]	**EFFL**	**Event FIFO Fill Level** Number of elements stored in Tx Event FIFO, range 0 to 32

Tx Event FIFO Acknowledge Register (CANFD_TXEFA)

Register	Offset	R/W	Description	Reset Value
CANFD_TXEFA	CANFDn_BA+0x0F8	R/W	Tx Event FIFO Acknowledge	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**EFAI**				

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:0]	**EFAI**	**Event FIFO Acknowledge Index** After the Host has read an element or a sequence of elements from the Tx Event FIFO it has to write the index of the last element read from Tx Event FIFO to EFAI. This will set the Tx Event FIFO Get Index EFGI (CANFD_TXEFS[12:8]) to EFAI + 1 and update the Event FIFO Fill Level EFFL (CANFD_TXEFS[5:0])

6.37 Key Store (KS)

6.37.1 Overview

The Key Store (KS) is a key management device for key storage. The Key Store is capable of providing a crypto engine to access or store the key while encryption, decryption and generation. The Key Store supports revoke key operation if key is unused. The Key Store is able to protect the key by integrity checking and auto verifying.

6.37.2 Features

- Supports programming interface for key management
- Supports key size 256-bit required for Cryptography engine
- Supports 2 OTP keys, maximum key size is 256-bit
- Supports crypto engine access Key Store directly
- Supports revoke operation
- Supports revoke key in OTP while tamper detected
- Supports integrity checking
- Supports auto verify function
- Supports lock function

6.37.3 Block Diagram

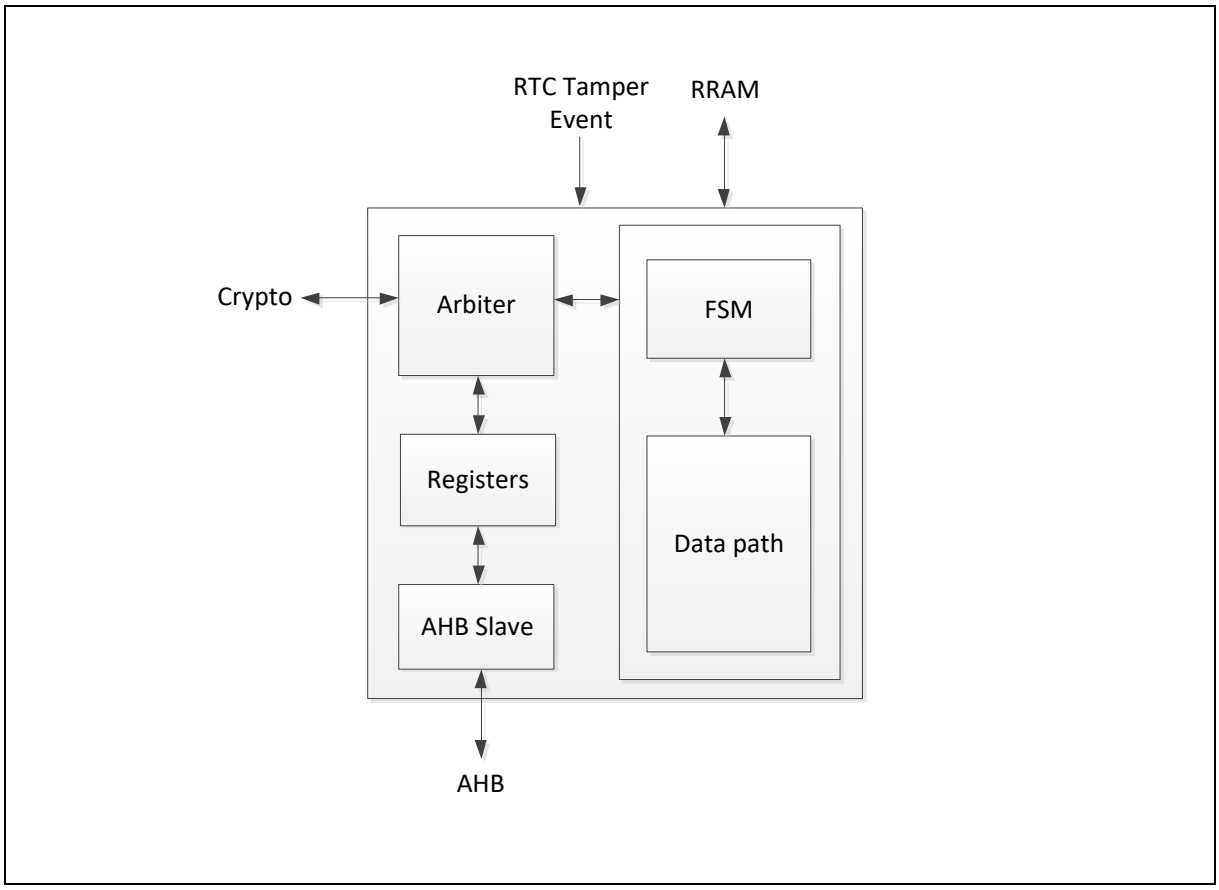


Figure 6.37-1 Key Store Block Diagram

6.37.4 Basic Configuration

- Clock Source Configuration
  - Enable KS peripheral clock in KSCKEN (CLK_AHBCLK0[13]).

6.37.5 Functional Description

6.37.5.1 Initialization

When chip is powered up, Key Store has no information about the key at OTP. The Key Store needs to read the metadata and check status, so the Key Store should be initialized by setting INIT(KS_CTL[8]) and START(KS_CTL[0]). When the Key Store is initialized, INITDONE(KS_STS[7]) will be set. The Key Store will check INITDONE before executing the operation except the initialization. If INITDONE is 0, the current operation will fail and EIF(KS_STS[1]) will be set. User should clear EIF by writing it to 0 and re-initialize Key Store.

Before initialization, KS_OTPSTS register keep the reset value and will update the current status after initialization.

The Key Store needs to be initialized when chip power up, wake-up from Power-down mode, and reset. If INITDONE is set, the Key Store has not to do initialization again.

Initial flow:

1. Check BUSY(KS_STS[2]) is 0 and INITDONE(KS_STS[7]) is 0.
2. Set INIT(KS_CTL[8]) to 1 and START(KS_CTL[0]) to 1
3. Wait BUSY(KS_STS[2]) to 0
4. Check INITDONE(KS_STS[7]) is 1

6.37.5.2 Basic Operation

There is a basic operation about create key, read key, erase key and revoke key in Key Store. User can select the location, fill in the metadata, select the operation and enable start bit, and then Key Store will start to operating. When the operation is finished, Key Store will be set the interrupt flag. If key creation is finished, the Key Store will return the key number for access this key in the future.

Each key has its metadata and metadata contains the information shown as follows.

1. Owner – OWNER(KS_METADATA[18:16])
   - Key can be set to which owner can use. Only owner can access the key.
2. Size – SIZE(KS_METADATA[12:8])
   - Set key size.
3. Readable – READABLE(KS_METADATA[2])
   - Key can be read by CPU or not.
4. Booting State – BS(KS_METADATA[4])
   - Key can be used at booting state (BL1 state) or not. If BS is set, the key can be used at booting state only.

Among the metadata mentioned above, only revoke attribute can be changed to 1 by revoke key operation and others metadata is not allowed to be changed.

User can create keys to OTP, and then Key Store programs the key and its metadata to OTP. When EIF(KS_STS[1]) is set, the EIF should be cleared by writing 1 before next operation start.

If the key is placed in OTP, this key can be erased by erase one key operation and the Key Store will replace current space with spare space to add new key. Each key storage block has three spare space, so each OTP key has three time to erase.

Table 6.37-1 shows the detailed information for OTP and Table 6.37-2 shows the information of each OTP key storage block.

	OTP
Storage Size	328 bytes
Store maximum key size	256 bits
Store maximum key count	2 keys
Store RSA key	-
Store RSA middle data	-
Store ECDH or ECDSA random key generated from PRNG	-
Read operation	V
Create operation	V
Erase one key operation	V (up to 3 times before locked)
Erase all keys operation	-
Revoke key operation	V
Data Remanence	-
Lock key operation	V
Integrity checking	V
Auto verify	V

Table 6.37-1 Detailed Information for OTP

OTP Key Number	Key Naming	Key Storage Space	Erase Time
0	Secure Boot ECC public key	256 bits	3
1	Secure Boot ECC public key	256 bits	3

Table 6.37-2 Information of OTP Key Storage

The user manual for read, create, erase and revoke operation is shown as follows.

Read Operation Programming Flow:

Use read operation can read out keys which are readable or key owner are CPU. If a key is unreadable, EIF(KS_STS[1]) will be set to 1.

1. Check BUSY(KS_STS[2]) is 0 and EIF(KS_STS[1]) is 0.
2. Set DST(KS_METADATA[31:30]) to 10
3. Configure NUMBER(KS_METADATA[25:20])
4. Set OPMODE(KS_CTL[3:1]) to 000
5. Set START(KS_CTL[0]) to 1
6. Wait BUSY(KS_STS[2]) cleared.
7. Read key from KS_KEY0~KS_KEY7 registers if EIF(KS_STS[1]) is 0.

Create Operation Programming Flow:

Use create operation to create a key to OTP with specified key number. After the create operation is finished, KS will return key number according written the specified number.

1. Check BUSY(KS_STS[2]) is 0 and EIF(KS_STS[1]) is 0.
2. Set DST(KS_METADATA[31:30]) to 10
3. Configure NUMBER(KS_METADATA[25:20]), OWNER(KS_METADATA[18:16]), SIZE(KS_METADATA[12:8]),BS(KS_METADATA[4]), READABLE(KS_METADATA[2]) according key's metadata.
4. Set OPMODE(KS_CTL[3:1]) to 001
5. Write key to KS_KEY0~KEY7 registers
6. Set START(KS_CTL[0]) to 1
7. Wait BUSY(KS_STS[2]) cleared.
8. Check EIF(KS_STS[1]) is 0.
9. Read key number from NUMBER(KS_METADATA [25:20]) and KS_OTPSTS register to double check

Erase One Key Operation Programming Flow:

Use erase one key operation can erase the specified key if key is unused. After the operation is finished, Key Store will replace used space with spare space. Each OTP key can use this operation three times.

1. Check BUSY(KS_STS[2]) is 0 and EIF(KS_STS[1]) is 0.
2. Set DST(KS_METADATA[31:30]) to 10
3. Configure NUMBER(KS_METADATA[25:20])
4. Set OPMODE(KS_CTL[3:1]) to 010 and START(KS_CTL[0]) to 1
5. Wait BUSY(KS_STS[2]) cleared.
6. Check EIF(KS_STS[1]) is 0.

Revoke Key Operation Programming Flow:

Use revoke key operation can revoke the specified key if key is unused. After the operation is finished, the key is unused but the space which key is used will not be released.

1. Check BUSY(KS_STS[2]) is 0 and EIF(KS_STS[1]) is 0.
2. Set DST(KS_METADATA[31:30]) to 10
3. Configure NUMBER(KS_METADATA[25:20])

4. Set OPMODE(KS_CTL[3:1]) to 100 and START(KS_CTL[0]) to 1
5. Wait BUSY(KS_STS[2]) cleared.
6. Check EIF(KS_STS[1]) is 0.

Lock Key Operation Programming Flow

Use lock operation to lock key if key does not need to be updated. When the key is locked, this key only can be read out using read operation but cannot be erased or re-created using erase operation or create operation.

1. Check BUSY(KS_STS[2]) is 0 and EIF(KS_STS[1]) is 0.
2. Set DST(KS_METADATA[31:30]) to 10
3. Configure NUMBER(KS_METADATA[25:20])
4. Set OPMODE(KS_CTL[3:1]) to 111 and START(KS_CTL[0]) to 1
5. Wait BUSY(KS_STS[2]) to 0.
6. Check EIF(KS_STS[1]) is 0.

6.37.5.3 Access/Store keys by Crypto Engine

The Key Store supports ECC to access key for secure boot. Table 6.37-3 shows all situations of CPU and Crypto access key.

	CPU Read Key	Crypto Engine Access Key
Key created by CPU	Controlled by READABLE (KS_METADATA[2]) while create key operation	Controlled by OWNER (KS_METADATA[18:16]) while create key operation

Table 6.37-3 CPU and Crypto Engine Access Key for All Situations

6.37.5.4 Integrity Checking

The integrity checking function is always enabled to check key integrity. The integrity checking can detect if the key has been damaged.

When the create key operation is executed, the Key Store calculates the number of '0' and '1' and records the results. When the read key operation is executed, the Key Store will read out key, re-calculate the number of '0' and '1', and compare the result. If comparison is correct, the key can be read out from KS_KEY0~KS_KEY7. If comparison is failed, the key will not be read out and EIF will rise.

6.37.5.5 Tamper Event

Key Store supports revoke key in OTP while tamper detected. When receiving the tamper event and TCLRSF(KS_STS[6]) is 1, KeyStore will stop current operation and execute the process of revoke key. Even if system is reset, Key Store will prioritize the process of revoke key. If TCLREN(KS_CTL[14]) is set to 1, TCLRSF will be set and cannot be clear.

6.37.5.6 Auto Verify

While creating key operation, Key Store will check whether key is written completely. When create key operation is enabled, Key Store will read back and compare with KS_KEY0~KS_KET7 registers after program. If key read back is not the same as KS_KEY0~KS_KET7 registers, key will be revoke and EIF will be set. Moreover, the space will decrease and key count will be increase.

6.37.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
KS Base Address: **KS_BA = 0x4003_5000**				
KS_CTL	KS_BA+0x000	R/W	Key Store Control Register	0x0000_0000
KS_METADATA	KS_BA+0x004	R/W	Key Store Metadata Register	0x0000_0000
KS_STS	KS_BA+0x008	R/W	Key Store Status Register	0x0000_00XX
KS_KEY0	KS_BA+0x020	R/W	Key Store Entry Key Word 0 Register	0x0000_0000
KS_KEY1	KS_BA+0x024	R/W	Key Store Entry Key Word 1 Register	0x0000_0000
KS_KEY2	KS_BA+0x028	R/W	Key Store Entry Key Word 2 Register	0x0000_0000
KS_KEY3	KS_BA+0x02C	R/W	Key Store Entry Key Word 3 Register	0x0000_0000
KS_KEY4	KS_BA+0x030	R/W	Key Store Entry Key Word 4 Register	0x0000_0000
KS_KEY5	KS_BA+0x034	R/W	Key Store Entry Key Word 5 Register	0x0000_0000
KS_KEY6	KS_BA+0x038	R/W	Key Store Entry Key Word 6 Register	0x0000_0000
KS_KEY7	KS_BA+0x03C	R/W	Key Store Entry Key Word 7 Register	0x0000_0000
KS_OTPSTS	KS_BA+0x040	R	Key Store OTP Keys Status Register	0x0000_0000

6.37.7 Register Description

Key Store Control Register (KS_CTL)

Register	Offset	R/W	Description	Reset Value
KS_CTL	KS_BA+0x000	R/W	Key Store Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
IEN	TCLREN	Reserved					INIT
7	6	5	4	3	2	1	0
Reserved				OPMODE			START

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**IEN**	**Key Store Interrupt Enable Bit** 0 = Key Store interrupt Disabled. 1 = Key Store interrupt Enabled.
[14]	**TCLREN**	**Tamper Event Clear Enable Bit** 0 = Key Store does not revoke all OTP keys when tamper event occurs. 1 = Key Store revokes all OTP keys when tamper event occurs.
[13:9]	**Reserved**	Reserved.
[8]	**INIT**	**Key Store Initialization** User should check BUSY(KS_STS[2]) is 0, and then write 1 to this bit and START(KS_CTL[0]), Key Store will start initialization. After the Key Store is initialized, INIT will be cleared.
[7:4]	**Reserved**	Reserved.
[3:1]	**OPMODE**	**Key Store Operation Mode** 000 = Read operation. 001 = Create operation. 010 = Erase one key operation. 100 = Revoke key operation. 111 = Lock operation. Others = Reserved.
[0]	**START**	**Key Store Start Control Bit (Write Only)** 0 = No operation. 1 = Start the operation.

Key Store Metadata Register (KS_METADATA)

Register	Offset	R/W	Description	Reset Value
KS_METADATA	KS_BA+0x004	R/W	Key Store Metadata Register	0x0000_0000

31	30	29	28	27	26	25	24
DST		Reserved				NUMBER	
23	22	21	20	19	18	17	16
NUMBER				Reserved	OWNER		
15	14	13	12	11	10	9	8
Reserved			SIZE				
7	6	5	4	3	2	1	0
Reserved			BS	Reserved	READABLE	Reserved	

Bits	Description	
[31:30]	**DST**	**Key Location Selection Bits** 10 = Key is in OTP. Others = Reserved.
[29:26]	**Reserved**	Reserved.
[25:20]	**NUMBER**	**Key Number** Before read or erase one key operation is started, user should write the key number to be operated. When create operation is finished, user can read these bits to get its key number.
[19]	**Reserved**	Reserved.
[18:16]	**OWNER**	**Key Owner Selection Bits** 100 = ECC. Others = Reserved.
[15:13]	**Reserved**	Reserved.
[12:8]	**SIZE**	**Key Size Selection Bits** 00000 = 128 bits. 00001 = 163 bits. 00010 = 192 bits. 00011 = 224 bits. 00100 = 233 bits. 00101 = 255 bits. 00110 = 256 bits. Others = Reserved.
[7:5]	**Reserved**	Reserved.
[4]	**BS**	**Booting State Selection Bit** 0 = Set key used at all state. 1 = Set key used at boot loader state 1 (BL1 state).
[3]	**Reserved**	Reserved.

[2]	**READABLE**	**Key Readable Control Bit** 0 = key is un-readable. 1 = key is readable.
[1:0]	**Reserved**	Reserved.

Key Store Status Register (KS_STS)

Register	Offset	R/W	Description	Reset Value
KS_STS	KS_BA+0x008	R/W	Key Store Status Register	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						KRVKF	Reserved
7	6	5	4	3	2	1	0
INITDONE	TCLRSF	Reserved			BUSY	EIF	IF

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**KRVKF**	**Key Store Key Revoked Flag (Read Only)** If KSPLOCK(CONFIG2[15:8]) is locked and mass erase occurs, Key Store will revoke OTP keys at next initialization. When initialization is finished, KRVKF will be set forever. 0 = All Keys have not been erased/revoked. 1 = All Keys have been erased/revoked.
[8]	**Reserved**	Reserved.
[7]	**INITDONE**	**Key Store Initialization Done Status (Read Only)** 0 = Key Store is un-initialized. 1 = Key Store is initialized.
[6]	**TCLRSF**	**Tamper Event Clear Set Flag (Read Only)** 0 = TCLREN(KS_CTL[14]) has been set. 1 = TCLREN(KS_CTL[14]) has not been set.
[5:3]	**Reserved**	Reserved.
[2]	**BUSY**	**Key Store Busy Flag (Read Only)** 0 = Key Store is idle or finished. 1 = Key Store is busy.
[1]	**EIF**	**Key Store Error Flag** 0 = No Key Store error. 1 = Key Store error interrupt. **Note:** This bit is cleared by writing 1 and it has no effect by writing 0.
[0]	**IF**	**Key Store Finish Interrupt Flag** 0 = No Key Store interrupt. 1 = Key Store operation done interrupt. **Note:** This bit is cleared by writing 1 and it has no effect by writing 0.

Key Store Entry Key Register (KS_KEY)

Register	Offset	R/W	Description	Reset Value
KS_KEY0	KS_BA+0x020	R/W	Key Store Entry Key Word 0 Register	0x0000_0000
KS_KEY1	KS_BA+0x024	R/W	Key Store Entry Key Word 1 Register	0x0000_0000
KS_KEY2	KS_BA+0x028	R/W	Key Store Entry Key Word 2 Register	0x0000_0000
KS_KEY3	KS_BA+0x02C	R/W	Key Store Entry Key Word 3 Register	0x0000_0000
KS_KEY4	KS_BA+0x030	R/W	Key Store Entry Key Word 4 Register	0x0000_0000
KS_KEY5	KS_BA+0x034	R/W	Key Store Entry Key Word 5 Register	0x0000_0000
KS_KEY6	KS_BA+0x038	R/W	Key Store Entry Key Word 6 Register	0x0000_0000
KS_KEY7	KS_BA+0x03C	R/W	Key Store Entry Key Word 7 Register	0x0000_0000

31	30	29	28	27	26	25	24
KEY							
23	22	21	20	19	18	17	16
KEY							
15	14	13	12	11	10	9	8
KEY							
7	6	5	4	3	2	1	0
KEY							

Bits	Description	
[31:0]	**KEY**	**Key Data (Read/Write, Read to Clear)** These registers will be cleared if Key Store executes the write operation or CPU completes the reading key.

Key Store OTP Keys Status Register (KS_OTPSTS)

Register	Offset	R/W	Description	Reset Value
KS_OTPSTS	KS_BA+0x040	R	Key Store OTP Keys Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						KEY1	KEY0

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**KEY1**	**OTP Key 1 Used Status** 0 = OTP key 1 is unused. 1 = OTP key 1 is used. **Note:** When chip is in RMA stage, this bit will be set to 1 and key is revoked after initialization if key is existed.
[0]	**KEY0**	**OTP Key 0 Used Status** 0 = OTP key 0 is unused. 1 = OTP key 0 is used. **Note:** When chip is in RMA stage, this bit will be set to 1 and key is revoked after initialization if key is existed.

6.38 Cryptographic Accelerator (CRYPTO)

6.38.1 Overview

The Crypto (Cryptographic Accelerator) includes a secure pseudo random number generator (PRNG) core and supports AES, algorithms.

The PRNG core supports 128, 163, 192, 224, 233, 255, and 256 bits random number generation.

The AES accelerator is an implementation fully compliant with the AES (Advance Encryption Standard) encryption and decryption algorithm. The AES accelerator supports ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, and CBC-CS3 mode.

Section	Sub-section	M2L31xxDAE	M2L31xx4AE
Cryptographic Accelerator	6.38.5.1 PRNG (Pseudo Random Number Generator)	●	-

Table 6.38-1 Cryptographic Features Comparison Table at Different Chip

6.38.2 Features

- PRNG
  - Supports 128, 163, 192, 224, 233, 255, and 256 bits random number generation
  - Able to take the true random number seed from TRNG
- AES
  - Supports FIPS NIST 197
  - Supports SP800-38A and addendum
  - Supports 128, 192, and 256 bits key
  - Supports both encryption and decryption
  - Supports ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2 and CBC-CS3 modes
  - Supports key expander for key expansion in FIPS NIST 197

6.38.3 Block Diagram

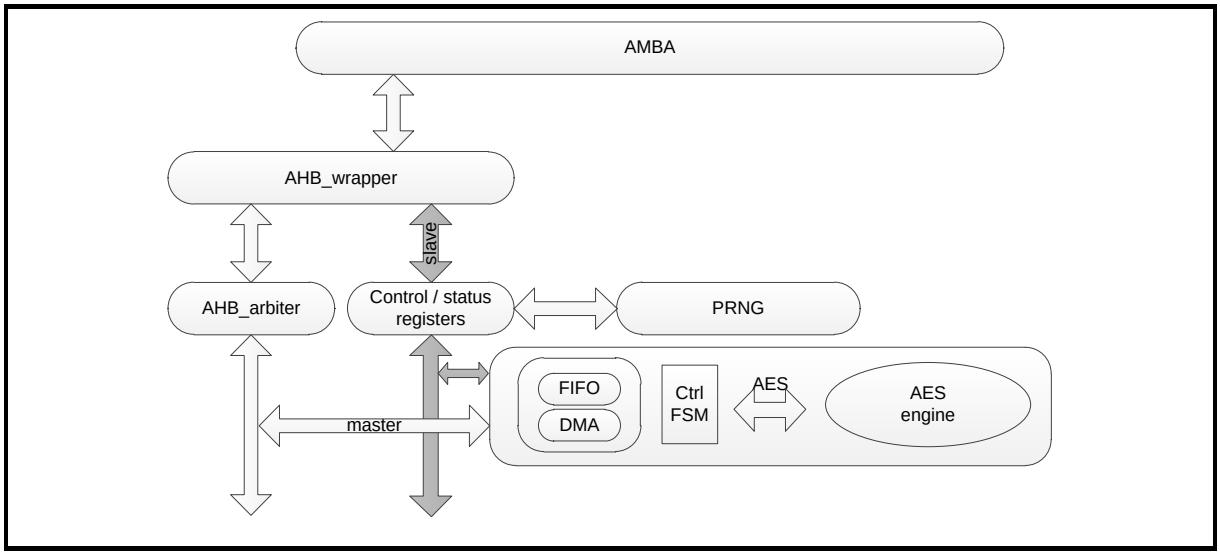


Figure 6.38-1 Cryptographic Accelerator Block Diagram

6.38.4 Basic Configuration

- Clock Source Configuration
  - Enable CRYPTO peripheral clock in CRPTCKEN(CLK_AHBCLK0[12])
- Reset Configuration
  - Reset CRYPTO controller in CRPTRST(SYS_IPRST0[12])

6.38.5 Functional Description

The cryptographic accelerator includes a secure pseudo random number generator (PRNG) core and supports AES, algorithms. The accelerator can be used in different data security applications, such as secure communications that need cryptographic protection and integrity.

1. The PRNG core supports 128, 163, 192, 224, 233, 255, and 256 bits random number generation configured by KEYSZ(CRYPTO_PRNG_CTL[4:2]).
2. The AES accelerator is a fully compliant implementation of the AES (Advance Encryption Standard) encryption and decryption algorithm. The AES accelerator supports ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, and CBC-CS3 mode. The AES accelerator provides the DMA function to reduce the CPU intervention, and supports three burst lengths, sixteen-words, eight-words, and four-words.

Software can control the data flow by enabling the CRYPTO_INTEN, and monitor the accelerator status by checking the CRYPTO_INTSTS status register. When any engine happened operation error or buffer error, the corresponding error flag will be set to 1 and inform to CPU if error interrupt enable bit is set to 1. If users want to detail error condition, software can check status flag register of each engine.

Engine	Error Interrupt Enable Bit	Error Interrupt Flag	Error Conditions
PRNG	PRNGEIEN (CRYPTO_INTEN[17])	PRNGEIF (CRYPTO_INTSTS[17])	TRNGERR
AES	AESEIEN (CRYPTO_INTEN[1])	AESEIF (CRYPTO_INTSTS[1])	INBUFERR/OUTBUFERR/BUS ERR

Table 6.38-2 Each Engine Error Conditions and Error Flag

The cryptographic accelerator supports the following features to enhance the performance.

DMA Mode

Once DMA source address register, destination address register, and byte count register are configured by CPU, moving data from and to accelerator is done by DMA logic totally. This mode can off-load the loading from the CPU. The cryptographic accelerator embeds one hardware DMA channel for AES engine.

Engine	DMA Enable Bit
AES	DMAEN (CRYPTO_AES_CTL[7])

Table 6.38-3 DMA Enable Bit Table

DMA Cascade Mode

In the case that the data SRAM resource is tight, or another peripheral is scheduled to switch, the data source or sink needs an update, while the setting for the accelerator operation is planned to be kept. In this mode, software can update DMA source address register, destination address register, and byte count register during a cascade operation, without finishing the accelerator operation. Please note that the last two blocks of AES-CBC-CS1/2/3 must be in the last cascade operation.

Engine	DMA Cascade Bit
AES	DMACSCAD (CRYPTO_AES_CTL[6])

Table 6.38-4 DMA Cascade Bit Table

Non-DMA Mode

In the case that the input data is small in size, DMA mode is not preferred. This mode can reduce the processing time for the accelerator, since no DMA related register needs a configuration, and no latency in DMA logic is introduced. Input data was feeding to cryptographic engine via writing to data input register.

Channel Expansion Mode

In this mode, several virtual channels are feasible in AES mode. The intermediate data (or called feedback information) from feedback registers (CRYPTO_AES_FDBCKx) should be stored temporarily in data SRAM and switched to another configuration setting of accelerator operation that includes operational mode, encryption/decryption, key, key size, initial vector (IV), and other parameters. Once switching back, the intermediate data from feedback registers should be written to initial vectors (CRYPTO_AES_IVx) for the accelerator to continue the operation with the original configuration setting. Finally, user must configure AES control register (CRYPTO_AES_CTL) to complete accelerator operation. Note that, in ECB mode, there is no need to move the intermediate data from feedback registers to IV(CRYPTO_AES_IVx).

6.38.5.1 PRNG (Pseudo Random Number Generator)

The PRNG block diagram is depicted in Figure 6.38-2. The core supports 128, 163, 192, 224, 233, 255 and 256 bits random number generation configured by KEYSZ(CRYPTO_PRNG_CTL[4:2]).

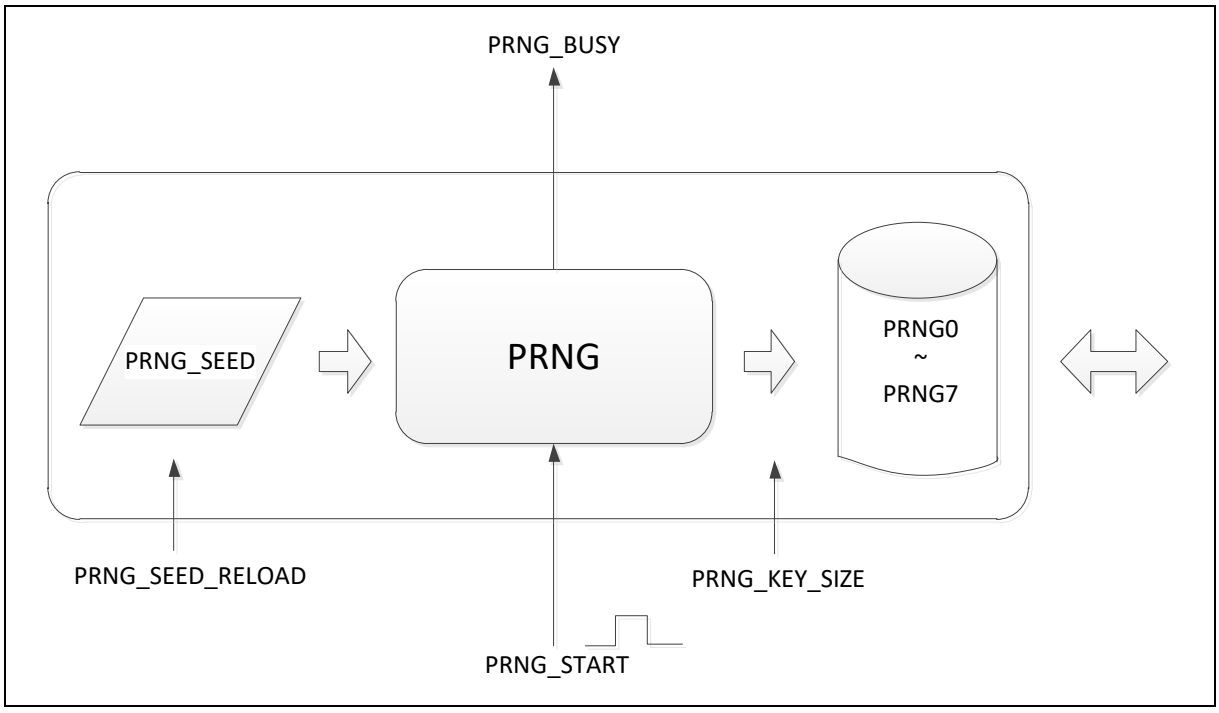


Figure 6.38-2 PRNG Function Diagram

Programming steps to get the **pseudo** random number are depicted below.

1. Check the BUSY(CRYPTO_PRNG_CTL[8]) until it comes to 0.
2. Write a random seed to CRYPTO_PRNG_SEED. Note that CRYPTO_PRNG_SEED should be initialized since it is not initialized as the chip powers up.
3. Configure PRNG control register CRYPTO_PRNG_CTL for seed source

(CRYPTO_PRNG_CTL[16]), key size(CRYPTO_PRNG_CTL[4:2]), seed reload(CRYPTO_PRNG_CTL[1]), and PRNG start(CRYPTO_PRNG_CTL[0]).

4. Software checks BUSY(CRYPTO_PRNG_CTL[8]) until it comes to 0, or waits for the PRNGIF(CRYPTO_INTSTS[16]) (must enable PRNGIEN(CRYPTO_INTEN[16])). Then software can read the output random numbers from CRYPTO_PRNG_KEY0 ~ CRYPTO_PRNG_KEY7.

The programming steps of PRNG to get the random number with the true random number seed from TRNG are depicted below.

7. Enable LDOEN (TRNG_CTL[0]) bit.
8. Wait LDORDY(TRNG_STS[0]) bit to be set to 1.
9. Set NRST (TRNG_CTL[1]) bit to 1 and enable TRNGEN (TRNG_CTL[2]) bit.
10. Check the BUSY(CRYPTO_PRNG_CTL[8]) until it comes to 0.
11. Configure PRNG control register CRYPTO_PRNG_CTL for seed source (CRYPTO_PRNG_CTL[16]), key size(CRYPTO_PRNG_CTL[4:2]), seed reload(CRYPTO_PRNG_CTL[1]), and PRNG start(CRYPTO_PRNG_CTL[0]).
12. Software checks BUSY(CRYPTO_PRNG_CTL[8]) until it comes to 0, or waits for the PRNGIF(CRYPTO_INTSTS[16]) (must enable PRNGIEN(CRYPTO_INTEN[16])). Then software can read the output random numbers from CRYPTO_PRNG_KEY0 ~ CRYPTO_PRNG_KEY7.

6.38.5.2 AES (Advanced Encryption Standard)

The AES accelerator supports AES algorithm from U.S NIST. This accelerator provides 8 block cipher modes includes ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, and CBC-CS3 mode. The detail of above block cipher modes are described in the following:

Electronic Codebook Mode

The Electronic Codebook (ECB) mode is a confidentiality mode that features the assignment of a fixed ciphertext block to each plaintext block, for a given key. It's analogous to the assignment of code words in a codebook.

In ECB encryption, each block of the plaintext is applied to the forward cipher function $CIPH_k$ directly and independently. The resulting sequence of output blocks is the ciphertext. In ECB decryption, each block of the ciphertext is applied to the inverse cipher function $CIPH^{-1}_k$ directly and independently. The resulting sequence of output blocks is the plaintext.

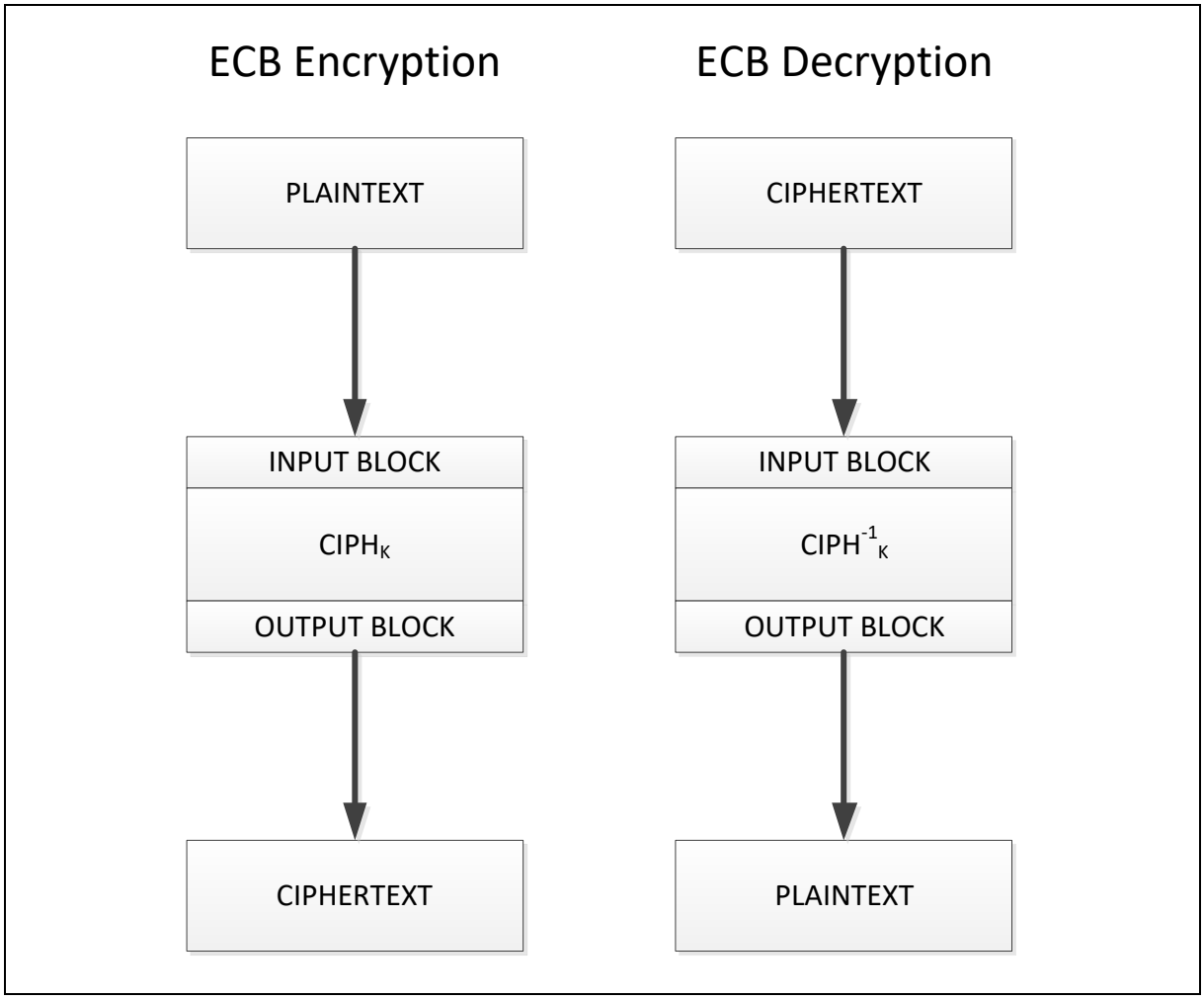


Figure 6.38-3 Electronic Codebook Mode

In ECB mode, any given plaintext block always gets encrypted to the same ciphertext block under a given key. If this property is undesirable in a particular application, the ECB mode should not be used.

Cipher Block Chaining Mode

The Cipher Block Chaining (CBC) mode is a confidentiality mode whose encryption process features the combining chaining of the plaintext blocks with the previous ciphertext blocks. The CBC mode requires an initialization vector (IV) to combine with the first plaintext block. The IV does not need to be secret, but it must be unpredictable.

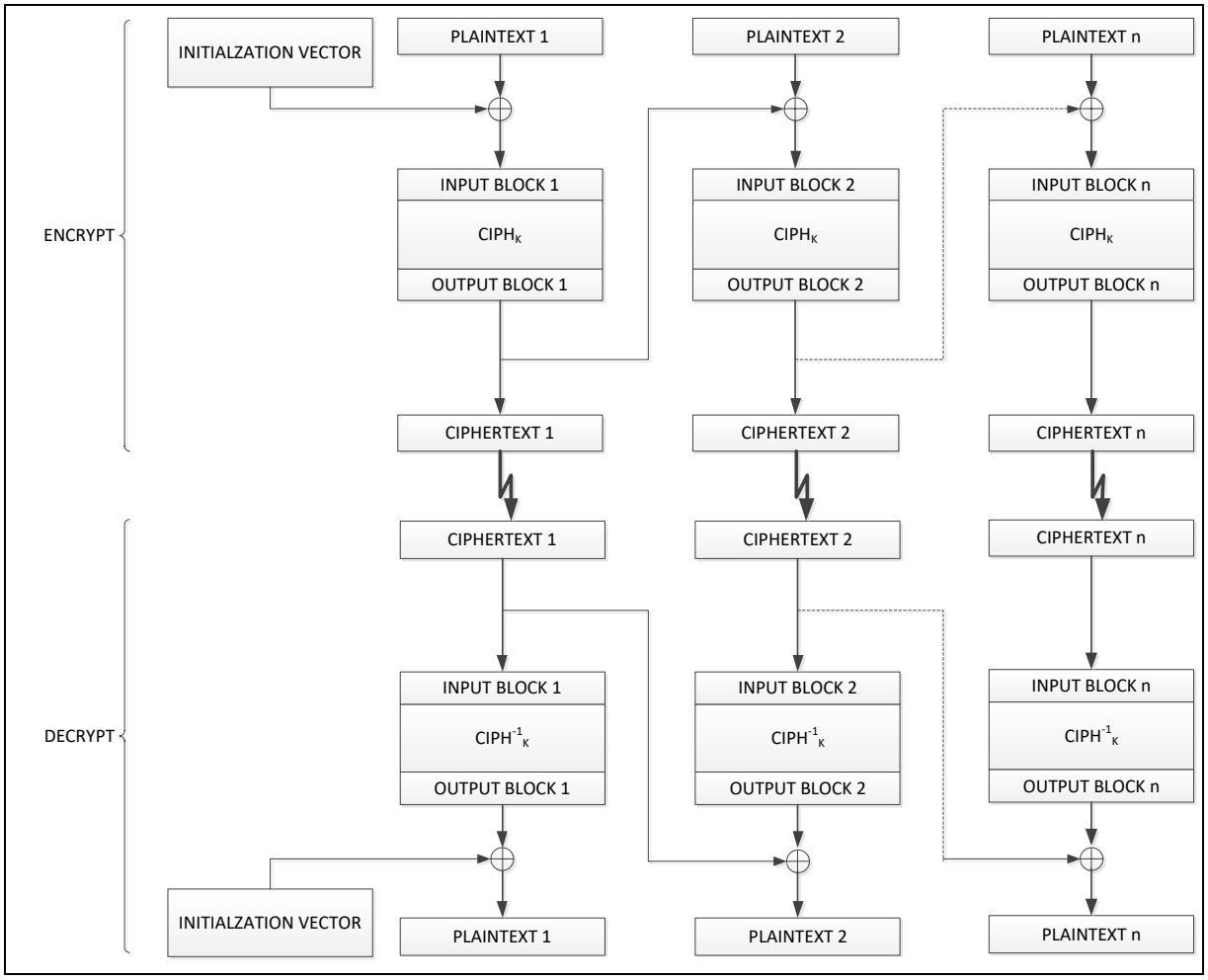


Figure 6.38-4 Cipher Block Chaining Mode

Cipher Feedback Mode (CFB)

The Cipher Feedback (CFB) mode is a confidentiality mode that features the feedback of successive ciphertext segments into the input blocks of the forward cipher to generate output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. The CFB mode requires an IV as the initial input block. The IV need not be secret, but it must be unpredictable. The AES only supports 128-bit segment length CFB mode.

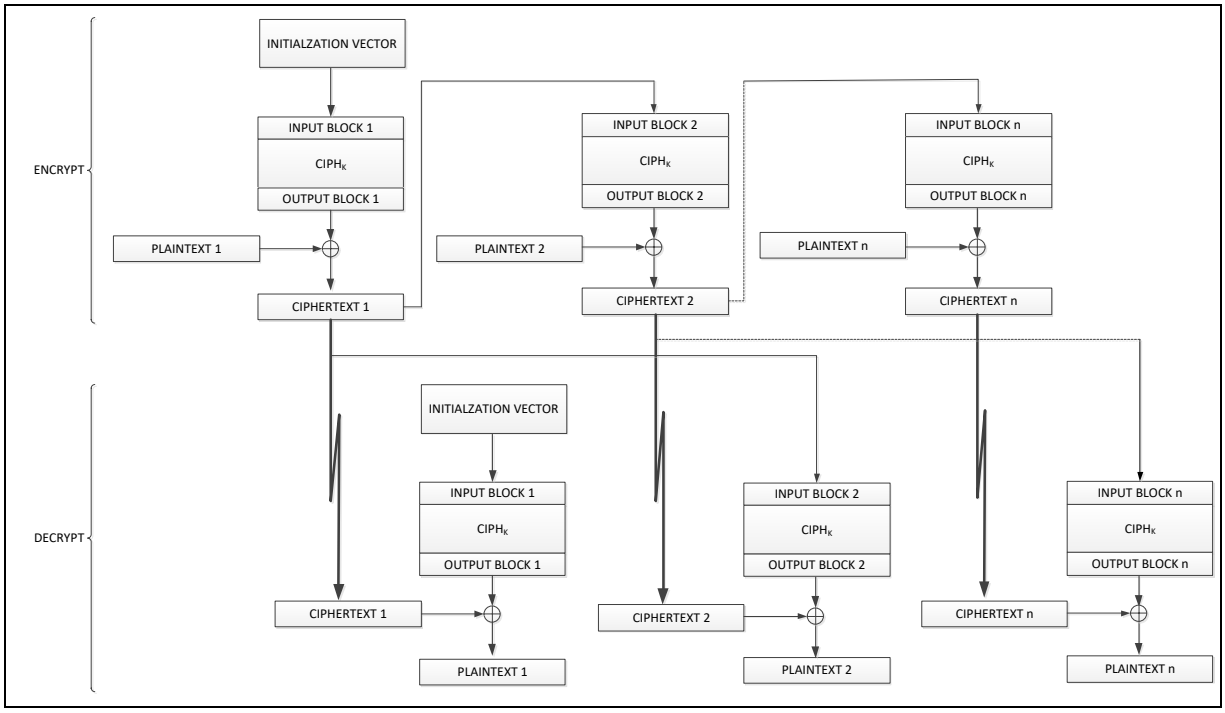


Figure 6.38-5 Cipher Feedback Mode

Output Feedback Mode

The Output Feedback (OFB) mode is a confidentiality mode that features the iteration of the forward cipher on an IV to generate a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. The OFB mode requires that the IV is a nonce, i.e., the IV must be unique for each execution of the mode under the given key.

The OFB mode requires a unique IV for every message that is ever encrypted under the given key. If, contrary to this requirement, the same IV is used for the encryption of more than one message, then the confidentiality of those messages may be compromised. Confidentiality may be similarly compromised if any of the input blocks to the forward cipher function for the encryption of a message is designated as the IV for the encryption of another message under the given key.

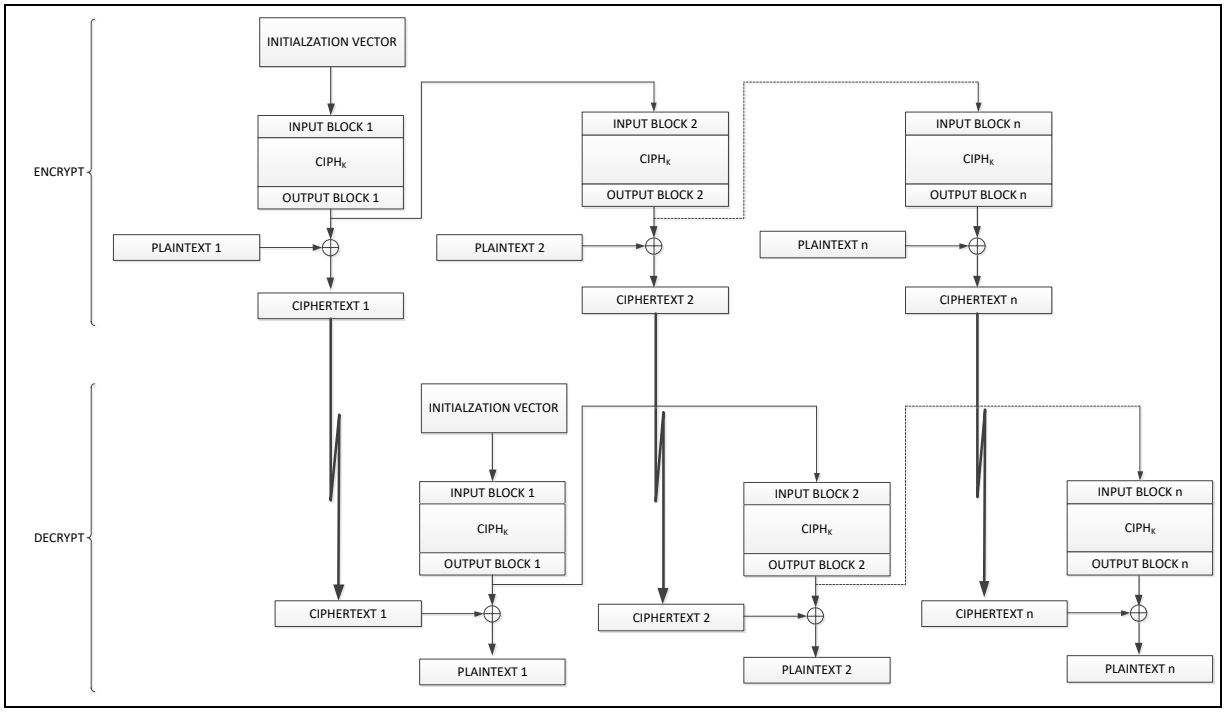


Figure 6.38-6 Output Feedback Mode

Counter Mode (CTR)

The Counter (CTR) mode is a confidentiality mode that features the application of the forward cipher to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. The sequence of counters must have the property that each block in the sequence is different from every other block. This condition is not restricted to a single message: across all of the messages that are encrypted under the given key, all of the counters must be distinct.

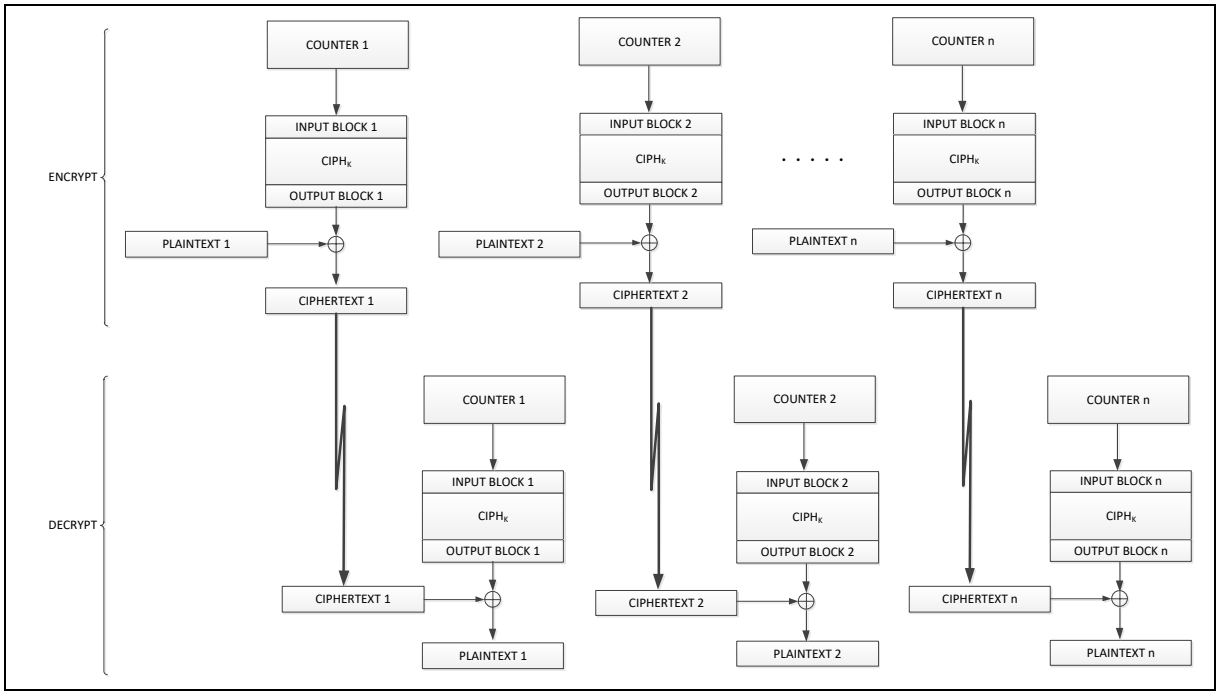


Figure 6.38-7 Counter Mode

CBC Ciphertext-Stealing 1 Mode (CBC-CS1)

Figure 6.38-8 illustrates the CBC-CS1-Encrypt algorithm for the case that P_n^* is a partial block. The cryptographic accelerator would append P_n^* with '0' to form a complete block P_n.

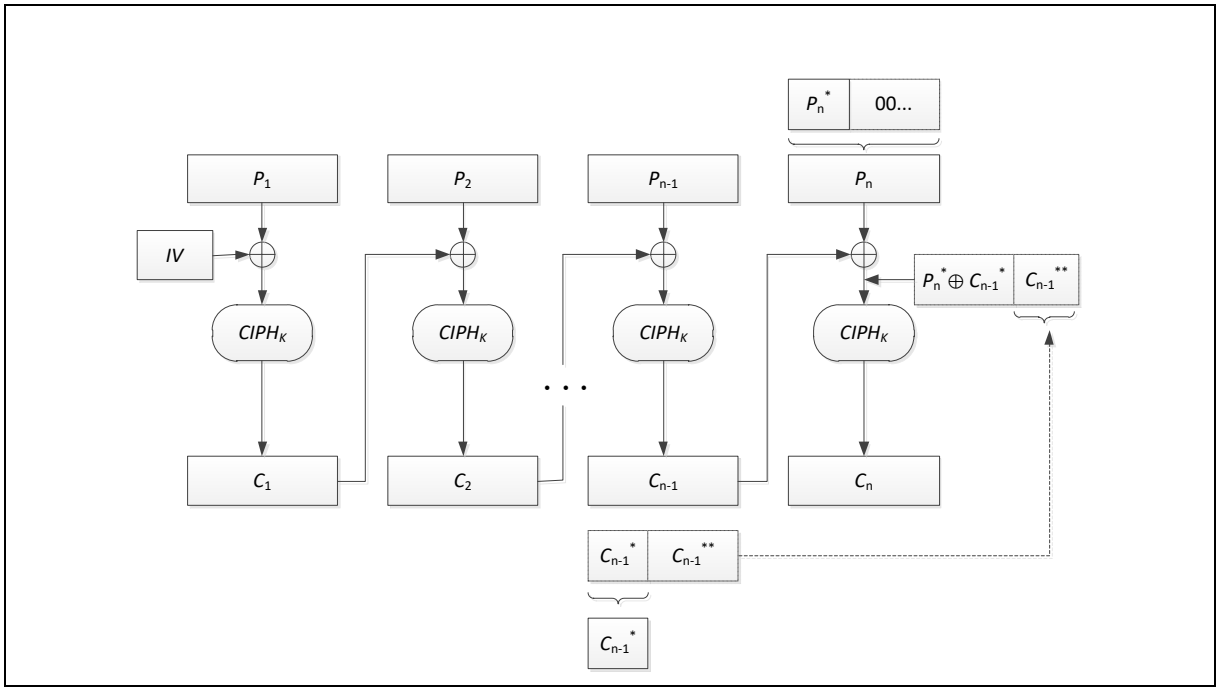


Figure 6.38-8 CBC-CS1 Encryption

Figure 6.38-9 illustrates the CBC-CS1-Decrypt algorithm for the case that C_{n-1}^* is a partial block.

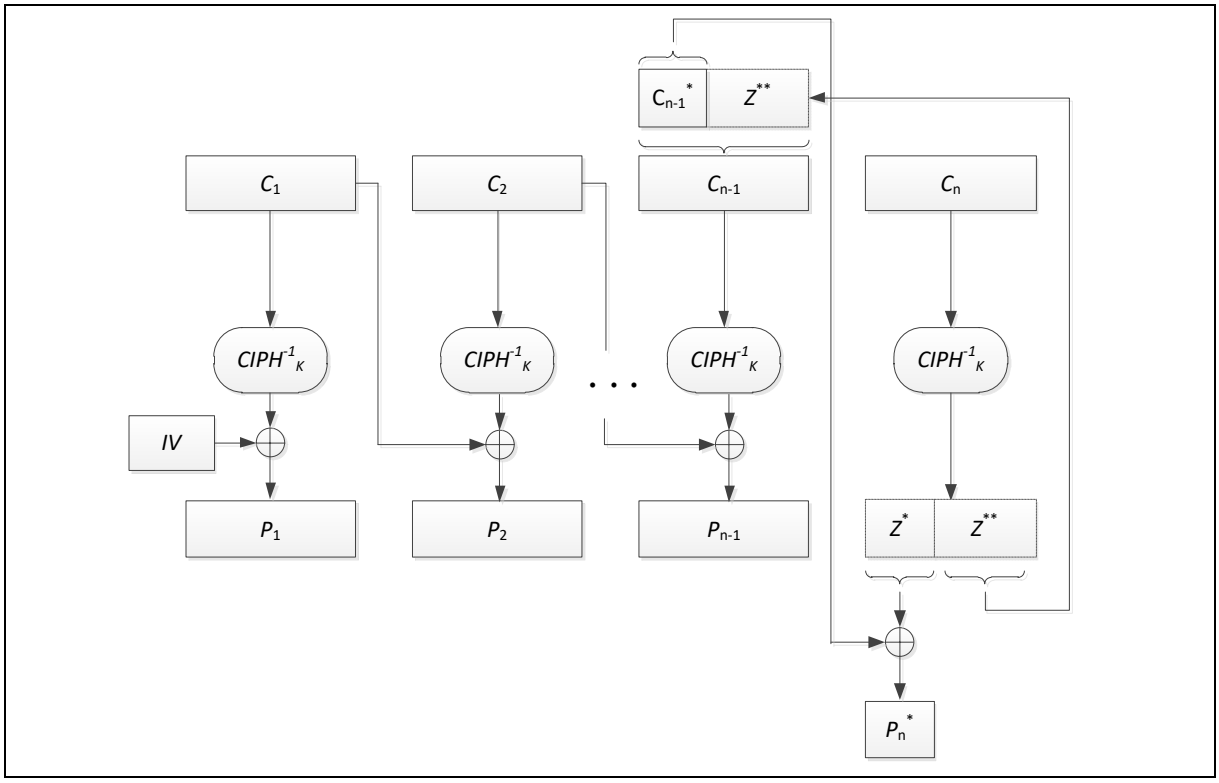


Figure 6.38-9 CBC-CS1 Decryption

CBC Ciphertext-Stealing 2 Mode (CBC-CS2)

When P_n^* is a partial block, then CBC-CS2-Encrypt and CBC-CS1-Encrypt differ only in the ordering of C_{n-1}^* and C_n.

CBC Ciphertext-Stealing 3 Mode (CBC-CS3)

C_{n-1}^* and C_n are unconditionally swapped, i.e., even when C_{n-1}^* is a complete block; therefore, CBC-CS3 is not strictly an extension of CBC mode. In the other case, i.e., when C_{n-1}^* is a nonempty partial block, CBC-CS3-Encrypt is equivalent to CBC-CS2-Encrypt.

Refer to the following programming steps for how to program the AES related registers.

AES DMA Mode Programming Flow

1. Write 1 to AESIEN (CRYPTO_INTEN[0]) to enable AES interrupt if needed.
2. Program AES key to registers CRYPTO_AES_KEY0 ~ CRYPTO_AES_KEY7.
3. Program initial vectors to registers CRYPTO_AES_IV0 ~ CRYPTO_AES_IV3.
4. Program DMA source address to register CRYPTO_AES_SADDR.
5. Program DMA destination address to register CRYPTO_AES_DADDR.
6. Program DMA byte count to register CRYPTO_AES_CNT.
7. Configure AES control register CRYPTO_AES_CTL for key protection (KEYPRT (CRYPTO_AES_CTL[31])), key and initial vector input/output swap(KINSWAP (CRYPTO_AES_CTL[25])/KOUTSWAP (CRYPTO_AES_CTL[24])), DMA input/output swap(INSWAP (CRYPTO_AES_CTL[23])/OUTSWAP (CRYPTO_AES_CTL[22])), encryption/decryption(ENCRYPTO (CRYPTO_AES_CTL[16])), operational mode(OPMODE (CRYPTO_AES_CTL[15:8])), DMA mode enable (DMAEN (CRYPTO_AES_CTL[7])), DMA cascade control(DMACSCAD (CRYPTO_AES_CTL[6])) and key size(KEYSZ (CRYPTO_AES_CTL[3:2])). Write 1 to DMALAST (CRYPTO_AES_CTL[5]) if enabled

DMACSCAD (CRYPTO_AES_CTL[6]) and current operation is last cascade operation.

8. Write input data to DMA source address with selected DMA byte count.
9. Write 1 to START(CRYPTO_AES_CTL[0]) to start AES encryption/decryption.
10. Wait for the AES interrupt flag AESIF (CRYPTO_INTSTS[0]) to be set or AES busy flag (CRYPTO_AES_STS[0]) to be set to zero.
11. Read output data from DMA destination address with selected DMA byte count.
12. Repeat step 4 to step 11 until all data processed if enabled DMACSCAD (CRYPTO_AES_CTL[6]).

AES Non-DMA Mode Programming Flow

1. Write 1 to AESIEN (CRYPTO_INTEN[0]) to enable AES interrupt if needed.
2. Program AES key to register CRYPTO_AES_KEY0 ~ CRYPTO_AES_KEY7.
3. Program initial vectors to register CRYPTO_AES_IV0 ~ CRYPTO_AES_IV3.
4. Configure AES control register CRYPTO_AES_CTL for key protection (KEYPRT (CRYPTO_AES_CTL[31])), key and initial vector input/output swap(KINSWAP (CRYPTO_AES_CTL[25])/KOUTSWAP (CRYPTO_AES_CTL[24])), input/output swap (INSWAP (CRYPTO_AES_CTL[23])/OUTSWAP (CRYPTO_AES_CTL[22])), encryption/decryption (ENCRYPTO (CRYPTO_AES_CTL[16])), operational mode (OPMODE (CRYPTO_AES_CTL[15:8])), DMA cascade control and key size(KEYSZ (CRYPTO_AES_CTL[3:2])).
5. Write 1 to START(CRYPTO_AES_CTL[0]) to start AES encryption/decryption.
6. Polling INBUFFULL(CRYPTO_AES_STS[9]) and OUTBUFEMPTY(CRYPTO_AES_STS[16]). If INBUFFULL(CRYPTO_AES_STS[9]) is 0, write 32 bits input data to CRYPTO_AES_DATIN. If OUTBUFEMPTY(CRYPTO_AES_STS[16]) is 0, read 32 bits data from CRYPTO_AES_DATOUT.
7. Repeat step 6 until 128 bits data (16 bytes) are written to and read from AES engine.
8. Write 1 to DMALAST(CRYPTO_AES_CTL[5]) if current operation is last operation
9. Write data byte count of last operation to register CRYPTO_AES_CNT if current operation is last operation.
10. Repeat Steps 5 to 9 until all data processed.

6.38.6 Register Map

R: read only, **W:** write only, **R/W:** both read and write

Register	Offset	R/W	Description	Reset Value
CRYPTO Base Address: **CRYPTO_BA = 0x4003_2000**				
CRYPTO_INTEN	CRYPTO_BA+0x000	R/W	Crypto Interrupt Enable Control Register	0x0000_0000
CRYPTO_INTSTS	CRYPTO_BA+0x004	R/W	Crypto Interrupt Flag	0x0000_0000
CRYPTO_PRNG_CTL	CRYPTO_BA+0x008	R/W	PRNG Control Register	0x0000_0000
CRYPTO_PRNG_SEED	CRYPTO_BA+0x00C	W	Seed for PRNG	0xXXXX_XXXX
CRYPTO_PRNG_KEY0	CRYPTO_BA+0x010	R	PRNG Generated Key0	0xXXXX_XXXX
CRYPTO_PRNG_KEY1	CRYPTO_BA+0x014	R	PRNG Generated Key1	0xXXXX_XXXX
CRYPTO_PRNG_KEY2	CRYPTO_BA+0x018	R	PRNG Generated Key2	0xXXXX_XXXX
CRYPTO_PRNG_KEY3	CRYPTO_BA+0x01C	R	PRNG Generated Key3	0xXXXX_XXXX
CRYPTO_PRNG_KEY4	CRYPTO_BA+0x020	R	PRNG Generated Key4	0xXXXX_XXXX
CRYPTO_PRNG_KEY5	CRYPTO_BA+0x024	R	PRNG Generated Key5	0xXXXX_XXXX
CRYPTO_PRNG_KEY6	CRYPTO_BA+0x028	R	PRNG Generated Key6	0xXXXX_XXXX
CRYPTO_PRNG_KEY7	CRYPTO_BA+0x02C	R	PRNG Generated Key7	0xXXXX_XXXX
CRYPTO_PRNG_STS	CRYPTO_BA+0x030	R	PRNG Status Register	0x0000_0000
CRYPTO_AES_FDBCK0	CRYPTO_BA+0x050	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRYPTO_AES_FDBCK1	CRYPTO_BA+0x054	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRYPTO_AES_FDBCK2	CRYPTO_BA+0x058	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRYPTO_AES_FDBCK3	CRYPTO_BA+0x05C	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRYPTO_AES_CTL	CRYPTO_BA+0x100	R/W	AES Control Register	0x0000_0000
CRYPTO_AES_STS	CRYPTO_BA+0x104	R	AES Engine Flag	0x0001_0100
CRYPTO_AES_DATIN	CRYPTO_BA+0x108	R/W	AES Engine Data Input Port Register	0x0000_0000
CRYPTO_AES_DATOUT	CRYPTO_BA+0x10C	R	AES Engine Data Output Port Register	0x0000_0000
CRYPTO_AES_KEY0	CRYPTO_BA+0x110	R/W	AES Key Word 0 Register	0x0000_0000
CRYPTO_AES_KEY1	CRYPTO_BA+0x114	R/W	AES Key Word 1 Register	0x0000_0000
CRYPTO_AES_KEY2	CRYPTO_BA+0x118	R/W	AES Key Word 2 Register	0x0000_0000
CRYPTO_AES_KEY3	CRYPTO_BA+0x11C	R/W	AES Key Word 3 Register	0x0000_0000
CRYPTO_AES_KEY4	CRYPTO_BA+0x120	R/W	AES Key Word 4 Register	0x0000_0000
CRYPTO_AES_KEY5	CRYPTO_BA+0x124	R/W	AES Key Word 5 Register	0x0000_0000
CRYPTO_AES_KEY6	CRYPTO_BA+0x128	R/W	AES Key Word 6 Register	0x0000_0000
CRYPTO_AES_KEY7	CRYPTO_BA+0x12C	R/W	AES Key Word 7 Register	0x0000_0000

CRYPTO_AES_IV0	CRYPTO_BA+0x130	R/W	AES Initial Vector Word 0 Register	0x0000_0000
CRYPTO_AES_IV1	CRYPTO_BA+0x134	R/W	AES Initial Vector Word 1 Register	0x0000_0000
CRYPTO_AES_IV2	CRYPTO_BA+0x138	R/W	AES Initial Vector Word 2 Register	0x0000_0000
CRYPTO_AES_IV3	CRYPTO_BA+0x13C	R/W	AES Initial Vector Word 3 Register	0x0000_0000
CRYPTO_AES_SADDR	CRYPTO_BA+0x140	R/W	AES DMA Source Address Register	0x0000_0000
CRYPTO_AES_DADDR	CRYPTO_BA+0x144	R/W	AES DMA Destination Address Register	0x0000_0000
CRYPTO_AES_CNT	CRYPTO_BA+0x148	R/W	AES Byte Count Register	0x0000_0000

6.38.7 Register Description

6.38.7.1 Crypto Register

CRYPTO Interrupt Enable Control Register (CRYPTO_INTEN)

Register	Offset	R/W	Description	Reset Value
CRYPTO_INTEN	CRYPTO_BA+0x000	R/W	Crypto Interrupt Enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						PRNGEIEN	PRNGIEN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						AESEIEN	AESIEN

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**PRNGEIEN**	**PRNG Error Flag Enable Bit** 0 = PRNG error interrupt flag Disabled. 1 = PRNG error interrupt flag Enabled.
[16]	**PRNGIEN**	**PRNG Interrupt Enable Bit** 0 = PRNG interrupt Disabled. 1 = PRNG interrupt Enabled.
[15:2]	**Reserved**	Reserved.
[1]	**AESEIEN**	**AES Error Flag Enable Bit** 0 = AES error interrupt flag Disabled. 1 = AES error interrupt flag Enabled.
[0]	**AESIEN**	**AES Interrupt Enable Bit** 0 = AES interrupt Disabled. 1 = AES interrupt Enabled. **Note:** In DMA mode, an interrupt will be triggered when an amount of data set in AES_DMA_CNT is fed into the AES engine. In Non-DMA mode, an interrupt will be triggered when the AES engine finishes the operation.

CRYPTO Interrupt Flag Register (CRYPTO_INTSTS)

Register	Offset	R/W	Description	Reset Value
CRYPTO_INTSTS	CRYPTO_BA+0x004	R/W	Crypto Interrupt Flag	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						PRNGEIF	PRNGIF
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						AESEIF	AESIF

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**PRNGEIF**	**PRNG Error Flag** This register includes operating and setting error. The detail flag is shown in CRYPTO_PRNG_STS register. 0 = No PRNG error. 1 = PRNG error interrupt. **Note:** This bit is cleared by writing 1, and it has no effect by writing 0.
[16]	**PRNGIF**	**PRNG Finish Interrupt Flag** 0 = No PRNG interrupt. 1 = PRNG done interrupt. **Note:** This bit is cleared by writing 1, and it has no effect by writing 0.
[15:2]	**Reserved**	Reserved.
[1]	**AESEIF**	**AES Error Flag** This register includes operating and setting error. The detail flag is shown in CRYPTO_AES_STS register. 0 = No AES error. 1 = AES error interrupt. **Note:** This bit is cleared by writing 1, and it has no effect by writing 0.
[0]	**AESIF**	**AES Finish Interrupt Flag** 0 = No AES interrupt. 1 = AES done interrupt. **Note:** This bit is cleared by writing 1, and it has no effect by writing 0.

6.38.7.2 PRNG Register

PRNG Control Register (CRYPTO_PRNG_CTL)

Register	Offset	R/W	Description	Reset Value
CRYPTO_PRNG_CTL	CRYPTO_BA+0x008	R/W	PRNG Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							SEEDSRC
15	14	13	12	11	10	9	8
Reserved							BUSY
7	6	5	4	3	2	1	0
Reserved			KEYSZ			SEEDRLD	START

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**SEEDSRC**	**Seed Source** 0 = Seed is from TRNG. 1 = Seed is from PRNG seed register. **Note:** When SEEDRLD is set to 0, this bit (SEEDSRC) is meaningless.
[15:9]	**Reserved**	Reserved.
[8]	**BUSY**	**PRNG Busy (Read Only)** 0 = PRNG engine is idle. 1 = PRNG engine is generating CRYPTO_PRNG_KEYx. **Note:** This bit is equal to the busy bit of CRYPTO_PRNG_STS[0].
[7:5]	**Reserved**	Reserved.
[4:2]	**KEYSZ**	**PRNG Generate Key Size** 000 = 128 bits. 001 = 163 bits. 010 = 192 bits. 011 = 224 bits. 100 = 233 bits. 101 = 255 bits. 110 = 256 bits. 111 = Reserved.
[1]	**SEEDRLD**	**Reload New Seed for PRNG Engine** 0 = Generating key based on the current seed. 1 = Reload new seed.

[0]	**START**	**Start PRNG Engine** 0 = Stop PRNG engine. 1 = Generate new key and store the new key to register CRYPTO_PRNG_KEYx, which will be cleared when the new key is generated.

PRNG Seed Register (CRYPTO_PRNG_SEED)

Register	Offset	R/W	Description	Reset Value
CRYPTO_PRNG_SEED	CRYPTO_BA+0x00C	W	Seed for PRNG	0xXXXX_XXXX

31	30	29	28	27	26	25	24
SEED							
23	22	21	20	19	18	17	16
SEED							
15	14	13	12	11	10	9	8
SEED							
7	6	5	4	3	2	1	0
SEED							

Bits	Description	
[31:0]	SEED	**Seed for PRNG (Write Only)** The bits store the seed for PRNG engine. **Note:** In TRNG+PRNG mode, the seed is from TRNG engine, and it will not be stored in this register.

PRNG Key x Register (CRYPTO_PRNG_KEYx)

Register	Offset	R/W	Description	Reset Value
CRYPTO_PRNG_KEY0	CRYPTO_BA+0x010	R	PRNG Generated Key0	0xXXXX_XXXX
CRYPTO_PRNG_KEY1	CRYPTO_BA+0x014	R	PRNG Generated Key1	0xXXXX_XXXX
CRYPTO_PRNG_KEY2	CRYPTO_BA+0x018	R	PRNG Generated Key2	0xXXXX_XXXX
CRYPTO_PRNG_KEY3	CRYPTO_BA+0x01C	R	PRNG Generated Key3	0xXXXX_XXXX
CRYPTO_PRNG_KEY4	CRYPTO_BA+0x020	R	PRNG Generated Key4	0xXXXX_XXXX
CRYPTO_PRNG_KEY5	CRYPTO_BA+0x024	R	PRNG Generated Key5	0xXXXX_XXXX
CRYPTO_PRNG_KEY6	CRYPTO_BA+0x028	R	PRNG Generated Key6	0xXXXX_XXXX
CRYPTO_PRNG_KEY7	CRYPTO_BA+0x02C	R	PRNG Generated Key7	0xXXXX_XXXX

31	30	29	28	27	26	25	24
KEY							
23	22	21	20	19	18	17	16
KEY							
15	14	13	12	11	10	9	8
KEY							
7	6	5	4	3	2	1	0
KEY							

Bits	Description	
[31:0]	**KEY**	**Store PRNG Generated Key (Read Only)** The bits store the key that is generated by PRNG.

PRNG Status Register (CRYPTO_PRNG_STS)

Register	Offset	R/W	Description	Reset Value
CRYPTO_PRNG_STS	CRYPTO_BA+0x030	R	PRNG Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					TRNGERR	Reserved	
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							BUSY

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**TRNGERR**	**True Random Number Generator Error Flag** 0 = No error. 1 = Getting random number or seed failed.
[17:1]	**Reserved**	Reserved.
[0]	**BUSY**	**PRNG Busy Flag** 0 = PRNG engine is idle. 1 = PRNG engine is generating CRYPTO_PRNG_KEYx. **Note:** This bit is equal to the busy bit of CRYPTO_PRNG_CTL[8].

6.38.7.3 AES Register

AES Feedback x Register (CRYPTO_AES_FDBCKx)

Register	Offset	R/W	Description	Reset Value
CRYPTO_AES_FDBCK0	CRYPTO_BA+0x050	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRYPTO_AES_FDBCK1	CRYPTO_BA+0x054	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRYPTO_AES_FDBCK2	CRYPTO_BA+0x058	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRYPTO_AES_FDBCK3	CRYPTO_BA+0x05C	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000

31	30	29	28	27	26	25	24
FDBCK							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
FDBCK							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FDBCK							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FDBCK							

Bits	Description	
[31:0]	**FDBCK**	**AES Feedback Information** The feedback value is 128 bits in size. The AES engine uses the data from CRYPTO_AES_FDBCKx as the data inputted to CRYPTO_AES_IVx for the next block in DMA cascade mode. The AES engine outputs feedback information for IV in the next block's operation. Software can use this feedback information to implement more than four DMA channels. Software can store that feedback value temporarily. After switching back, fill the stored feedback value to CRYPTO_AES_IVx in the same channel operation, and then continue the operation with the original setting.

AES Control Register (CRYPTO_AES_CTL)

Register	Offset	R/W	Description	Reset Value
CRYPTO_AES_CTL	CRYPTO_BA+0x100	R/W	AES Control Register	0x0000_0000

31	30	29	28	27	26	25	24
KEYPRT	KEYUNPRT					KINSWAP	KOUTSWAP
23	22	21	20	19	18	17	16
INSWAP	OUTSWAP	Reserved					ENCRYPTO
15	14	13	12	11	10	9	8
OPMODE							
7	6	5	4	3	2	1	0
DMAEN	DMACSCAD	DMALAST	Reserved	KEYSZ		STOP	START

Bits	Description	
[31]	**KEYPRT**	**Protect Key** Read as a flag to reflect KEYPRT. 0 = No effect. 1 = Protect the content of the AES key from reading. The return value for reading CRYPTO_AES_KEYx is not the content of the registers CRYPTO_AES_KEYx. Once it is set, it can be cleared by asserting KEYUNPRT. The key content would be cleared as well.
[30:26]	**KEYUNPRT**	**Unprotect Key** Writing 0 to CRYPTO_AES_CTL[31] and “10110” to CRYPTO_AES_CTL[30:26] is to unprotect the AES key. The KEYUNPRT can be read and written. When it is written as the AES engine is operating, BUSY flag is 1, there would be no effect on KEYUNPRT.
[25]	**KINSWAP**	**AES Engine Input Key and Initial Vector Swap** 0 = Keep the original order. 1 = The order that CPU feeds key and initial vector to the accelerator will be changed from {byte3, byte2, byte1, byte0} to {byte0, byte1, byte2, byte3}.
[24]	**KOUTSWAP**	**AES Engine Output Key, Initial Vector and Feedback Swap** 0 = Keep the original order. 1 = The order that CPU reads key, initial vector and feedback from the accelerator will be changed from {byte3, byte2, byte1, byte0} to {byte0, byte1, byte2, byte3}.
[23]	**INSWAP**	**AES Engine Input Data Swap** 0 = Keep the original order. 1 = The order that CPU feeds data to the accelerator will be changed from {byte3, byte2, byte1, byte0} to {byte0, byte1, byte2, byte3}.
[22]	**OUTSWAP**	**AES Engine Output Data Swap** 0 = Keep the original order. 1 = The order that CPU reads data from the accelerator will be changed from {byte3, byte2, byte1, byte0} to {byte0, byte1, byte2, byte3}.
[21:17]	**Reserved**	Reserved.

[16]	**ENCRYPTO**	**AES Encryption/Decryption** 0 = AES engine executes decryption operation. 1 = AES engine executes encryption operation.
[15:8]	**OPMODE**	**AES Engine Operation Modes** 0x00 = ECB (Electronic Codebook Mode) 0x01 = CBC (Cipher Block Chaining Mode). 0x02 = CFB (Cipher Feedback Mode). 0x03 = OFB (Output Feedback Mode). 0x04 = CTR (Counter Mode). 0x10 = CBC-CS1 (CBC Ciphertext-Stealing 1 Mode). 0x11 = CBC-CS2 (CBC Ciphertext-Stealing 2 Mode). 0x12 = CBC-CS3 (CBC Ciphertext-Stealing 3 Mode).
[7]	**DMAEN**	**AES Engine DMA Enable Bit** 0 = AES DMA engine Disabled. The AES engine operates in Non-DMA mode. The data need to be written in CRYPTO_AES_DATIN. 1 = AES_DMA engine Enabled. The AES engine operates in DMA mode, and data movement from/to the engine is done by DMA logic.
[6]	**DMACSCAD**	**AES Engine DMA with Cascade Mode** 0 = DMA cascade function Disabled. 1 = In DMA cascade mode, software can update DMA source address register, destination address register, and byte count register during a cascade operation, without finishing the accelerator operation. **Note:** The last two blocks of AES-CBC-CS1/2/3 must be in the last cascade operation.
[5]	**DMALAST**	**AES Last Block** In DMA mode, this bit must be set as beginning the last DMA cascade round. In Non-DMA mode, this bit must be set when feeding in the last block of data in ECB, CBC, CTR, OFB, and CFB mode, and feeding in the (last-1) block of data at CBC-CS1, CBC-CS2, and CBC-CS3 mode. This bit is always 0 when it is read back, and must be written again once START is triggered.
[4]	**Reserved**	Reserved.
[3:2]	**KEYSZ**	**AES Key Size** This bit defines three different key size for AES operation. 2'b00 = 128 bits key. 2'b01 = 192 bits key. 2'b10 = 256 bits key. 2'b11 = Reserved. If the AES accelerator is operating and the corresponding flag BUSY is 1, updating this register has no effect.
[1]	**STOP**	**AES Engine Stop** 0 = No effect. 1 = Stop AES engine. **Note:** This bit is always 0 when it is read back.
[0]	**START**	**AES Engine Start** 0 = No effect. 1 = Start AES engine. BUSY flag will be set. **Note:** This bit is always 0 when it is read back.

AES Status Flag Register (CRYPTO_AES_STS)

Register	Offset	R/W	Description	Reset Value
CRYPTO_AES_STS	CRYPTO_BA+0x104	R	AES Engine Flag	0x0001_0100

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			BUSERR	Reserved	OUTBUFERR	OUTBUFFULL	OUTBUFEMPTY
15	14	13	12	11	10	9	8
Reserved			CNTERR	Reserved	INBUFERR	INBUFFULL	INBUFEMPTY
7	6	5	4	3	2	1	0
Reserved							BUSY

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**BUSERR**	**AES DMA Access Bus Error Flag** 0 = No error. 1 = Bus error will stop DMA operation and AES engine.
[19]	**Reserved**	Reserved.
[18]	**OUTBUFERR**	**AES Out Buffer Error Flag** 0 = No error. 1 = Error happened during getting the result from AES engine.
[17]	**OUTBUFFULL**	**AES Out Buffer Full Flag** 0 = AES output buffer is not full. 1 = AES output buffer is full, and software needs to get data from CRYPTO_AES_DATOUT. Otherwise, the AES engine will be pending since the output buffer is full.
[16]	**OUTBUFEMPTY**	**AES Out Buffer Empty** 0 = AES output buffer is not empty. There are some valid data kept in output buffer. 1 = AES output buffer is empty. Software cannot get data from CRYPTO_AES_DATOUT. Otherwise, the flag OUTBUFERR will be set to 1 since the output buffer is empty.
[15:13]	**Reserved**	Reserved.
[12]	**CNTERR**	**CRYPTO_AES_CNT Setting Error** 0 = No error in CRYPTO_AES_CNT setting. 1 = CRYPTO_AES_CNT is 0 if DMAEN (CRYPTO_AES_CTL[7]) is enabled.
[11]	**Reserved**	Reserved.
[10]	**INBUFERR**	**AES Input Buffer Error Flag** 0 = No error. 1 = Error happened during feeding data to the AES engine.

[9]	**INBUFFULL**	**AES Input Buffer Full Flag** 0 = AES input buffer is not full. Software can feed the data into the AES engine. 1 = AES input buffer is full. Software cannot feed data to the AES engine. Otherwise, the flag INBUFERR will be set to 1.
[8]	**INBUFEMPTY**	**AES Input Buffer Empty** 0 = There are some data in input buffer waiting for the AES engine to process. 1 = AES input buffer is empty. Software needs to feed data to the AES engine. Otherwise, the AES engine will be pending to wait for input data.
[7:1]	**Reserved**	Reserved.
[0]	**BUSY**	**AES Engine Busy** 0 = The AES engine is idle or finished. 1 = The AES engine is under processing.

AES Data Input Port Register (CRYPTO_AES_DATIN)

Register	Offset	R/W	Description	Reset Value
CRYPTO_AES_DATIN	CRYPTO_BA+0x108	R/W	AES Engine Data Input Port Register	0x0000_0000

31	30	29	28	27	26	25	24
DATIN							
23	22	21	20	19	18	17	16
DATIN							
15	14	13	12	11	10	9	8
DATIN							
7	6	5	4	3	2	1	0
DATIN							

Bits	Description	
[31:0]	DATIN	**AES Engine Input Port** CPU feeds data to AES engine through this port by checking CRYPTO_AES_STS. Feed data as INBUFFULL is 0.

AES Data Output Port Register (CRYPTO_AES_DATOUT)

Register	Offset	R/W	Description	Reset Value
CRYPTO_AES_DATOUT	CRYPTO_BA+0x10C	R	AES Engine Data Output Port Register	0x0000_0000

31	30	29	28	27	26	25	24
DATOUT							
23	22	21	20	19	18	17	16
DATOUT							
15	14	13	12	11	10	9	8
DATOUT							
7	6	5	4	3	2	1	0
DATOUT							

Bits	Description	
[31:0]	DATOUT	**AES Engine Output Port** CPU gets results from the AES engine through this port by checking CRYPTO_AES_STS. Get data as OUTBUFEMPTY is 0.

AES Key Word x Register (CRYPTO_AES_KEYx)

Register	Offset	R/W	Description	Reset Value
CRYPTO_AES_KEY0	CRYPTO_BA+0x110	R/W	AES Key Word 0 Register	0x0000_0000
CRYPTO_AES_KEY1	CRYPTO_BA+0x114	R/W	AES Key Word 1 Register	0x0000_0000
CRYPTO_AES_KEY2	CRYPTO_BA+0x118	R/W	AES Key Word 2 Register	0x0000_0000
CRYPTO_AES_KEY3	CRYPTO_BA+0x11C	R/W	AES Key Word 3 Register	0x0000_0000
CRYPTO_AES_KEY4	CRYPTO_BA+0x120	R/W	AES Key Word 4 Register	0x0000_0000
CRYPTO_AES_KEY5	CRYPTO_BA+0x124	R/W	AES Key Word 5 Register	0x0000_0000
CRYPTO_AES_KEY6	CRYPTO_BA+0x128	R/W	AES Key Word 6 Register	0x0000_0000
CRYPTO_AES_KEY7	CRYPTO_BA+0x12C	R/W	AES Key Word 7 Register	0x0000_0000

31	30	29	28	27	26	25	24
KEY							
23	22	21	20	19	18	17	16
KEY							
15	14	13	12	11	10	9	8
KEY							
7	6	5	4	3	2	1	0
KEY							

Bits	Description	
[31:0]	**KEY**	**CRYPTO_AES_KEYx** The KEY keeps the security key for AES operation. x = 0, 1..7. The security key for AES accelerator can be 128, 192, or 256 bits and four, six, or eight 32-bit registers are to store each security key. {CRYPTO_AES_KEY3, CRYPTO_AES_KEY2, CRYPTO_AES_KEY1, CRYPTO_AES_KEY0} stores the 128-bit security key for AES operation. {CRYPTO_AES_KEY5, CRYPTO_AES_KEY4, CRYPTO_AES_KEY3, CRYPTO_AES_KEY2, CRYPTO_AES_KEY1, CRYPTO_AES_KEY0} stores the 192-bit security key for AES operation. {CRYPTO_AES_KEY7, CRYPTO_AES_KEY6, CRYPTO_AES_KEY5, CRYPTO_AES_KEY4, CRYPTO_AES_KEY3, CRYPTO_AES_KEY2, CRYPTO_AES_KEY1, CRYPTO_AES_KEY0} stores the 256-bit security key for AES operation.

AES Initial Vector Word x Register (CRYPTO_AES_IVx)

Register	Offset	R/W	Description	Reset Value
CRYPTO_AES_IV0	CRYPTO_BA+0x130	R/W	AES Initial Vector Word 0 Register	0x0000_0000
CRYPTO_AES_IV1	CRYPTO_BA+0x134	R/W	AES Initial Vector Word 1 Register	0x0000_0000
CRYPTO_AES_IV2	CRYPTO_BA+0x138	R/W	AES Initial Vector Word 2 Register	0x0000_0000
CRYPTO_AES_IV3	CRYPTO_BA+0x13C	R/W	AES Initial Vector Word 3 Register	0x0000_0000

31	30	29	28	27	26	25	24
IV							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
IV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IV							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IV							

Bits		Description
[31:0]	**IV**	**AES Initial Vectors** x = 0, 1..3. Four initial vectors (CRYPTO_AES_IV0, CRYPTO_AES_IV1, CRYPTO_AES_IV2, and CRYPTO_AES_IV3) are for AES operating in CBC, CFB, and OFB mode. Four registers (CRYPTO_AES_IV0, CRYPTO_AES_IV1, CRYPTO_AES_IV2, and CRYPTO_AES_IV3) act as Nonce counter when the AES engine is operating in CTR mode.

AES DMA Source Address Register (CRYPTO_AES_SADDR)

Register	Offset	R/W	Description	Reset Value
CRYPTO_AES_SADDR	CRYPTO_BA+0x140	R/W	AES DMA Source Address Register	0x0000_0000

31	30	29	28	27	26	25	24
SADDR							
23	22	21	20	19	18	17	16
SADDR							
15	14	13	12	11	10	9	8
SADDR							
7	6	5	4	3	2	1	0
SADDR							

Bits	Description	
[31:0]	**SADDR**	**AES DMA Source Address** The AES accelerator supports DMA function to transfer the plain text between SRAM memory space and embedded FIFO. The SADDR keeps the source address of the data buffer where the source text is stored. Based on the source address, the AES accelerator can read the plain text (encryption) / cipher text (decryption) from SRAM memory space and do AES operation. The start of source address should be located at word boundary. In other words, bit 1 and 0 of SADDR are ignored. SADDR can be read and written. Writing to SADDR while the AES accelerator is operating doesn't affect the current AES operation. But the value of SADDR will be updated later on. Consequently, software can prepare the DMA source address for the next AES operation. In DMA mode, software can update the next CRYPTO_AES_SADDR before triggering START. The value of CRYPTO_AES_SADDR and CRYPTO_AES_DADDR can be the same.

AES DMA Destination Address Register (CRYPTO_AES_DADDR)

Register	Offset	R/W	Description	Reset Value
CRYPTO_AES_DADDR	CRYPTO_BA+0x144	R/W	AES DMA Destination Address Register	0x0000_0000

31	30	29	28	27	26	25	24
DADDR							
23	22	21	20	19	18	17	16
DADDR							
15	14	13	12	11	10	9	8
DADDR							
7	6	5	4	3	2	1	0
DADDR							

Bits	Description	
[31:0]	DADDR	**AES DMA Destination Address** The AES accelerator supports DMA function to transfer the cipher text between SRAM memory space and embedded FIFO. The DADDR keeps the destination address of the data buffer where the engine output's text will be stored. Based on the destination address, the AES accelerator can write the cipher text (encryption) / plain text (decryption) back to SRAM memory space after the AES operation is finished. The start of destination address should be located at word boundary. In other words, bit 1 and 0 of DADDR are ignored. DADDR can be read and written. Writing to DADDR while the AES accelerator is operating doesn't affect the current AES operation. But the value of DADDR will be updated later on. Consequently, software can prepare the destination address for the next AES operation. In DMA mode, software can update the next CRYPTO_AES_DADDR before triggering START. The value of CRYPTO_AES_SADDR and CRYPTO_AES_DADDR can be the same.

AES Byte Count Register (CRYPTO_AES_CNT)

Register	Offset	R/W	Description	Reset Value
CRYPTO_AES_CNT	CRYPTO_BA+0x148	R/W	AES Byte Count Register	0x0000_0000

31	30	29	28	27	26	25	24
CNT							
23	22	21	20	19	18	17	16
CNT							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

Bits	Description	
[31:0]	CNT	**AES Byte Count** The CRYPTO_AES_CNT keeps the byte count of source text that is for the AES engine operating in DMA mode. The CRYPTO_AES_CNT is 32-bit and the maximum of byte count is 4G bytes. CRYPTO_AES_CNT can be read and written. Writing to CRYPTO_AES_CNT while the AES accelerator is operating doesn't affect the current AES operation. But the value of CRYPTO_AES_CNT will be updated later on. Consequently, software can prepare the byte count of data for the next AES operation. According to CBC-CS1, CBC-CS2, and CBC-CS3 standard, the count of operation data must be more than 16 bytes. Operations that are qual to or less than one block will output unexpected result. In Non-DMA ECB, CBC, CFB, OFB, and CTR mode, CRYPTO_AES_CNT must be set as byte count for the last block of data before feeding in the last block of data. In Non-DMA CBC-CS1, CBC-CS2, and CBC-CS3 mode, CRYPTO_AES_CNT must be set as byte count for the last two blocks of data before feeding in the last two blocks of data.

6.39 Touch Key (TK)

6.39.1 Overview

The capacitive touch key sensing controller supports several programmable sensitivity levels for different applications to detect the finger touched or near the electrode covered by dielectric. It supports up to 16 touch keys with single-scan or programmable periodic key-scans, and system can be woken up by any key for low power applications.

6.39.2 Features

- Supports up to 16 touch keys
- Supports flexible reference pad and shielding electrode channel setting; at least 1 reference pad channel and 1 shielding electrode channel are needed
- Programmable sensitivity levels for each channel
- Programmable scanning speed for different applications
- Supports any touch key wake-up for low-power applications
- Supports single key-scan and programmable periodic key-scan
- Programmable interrupt options for key-scan complete with or without threshold control
- Supports independent reference capacitor bank(CB) registers for each channels
- Supports Timer0~3 time-out interrupts signal(TIF) to trigger touch key scan
- Supports RTC tick event to trigger touch key scan

6.39.3 Block Diagram

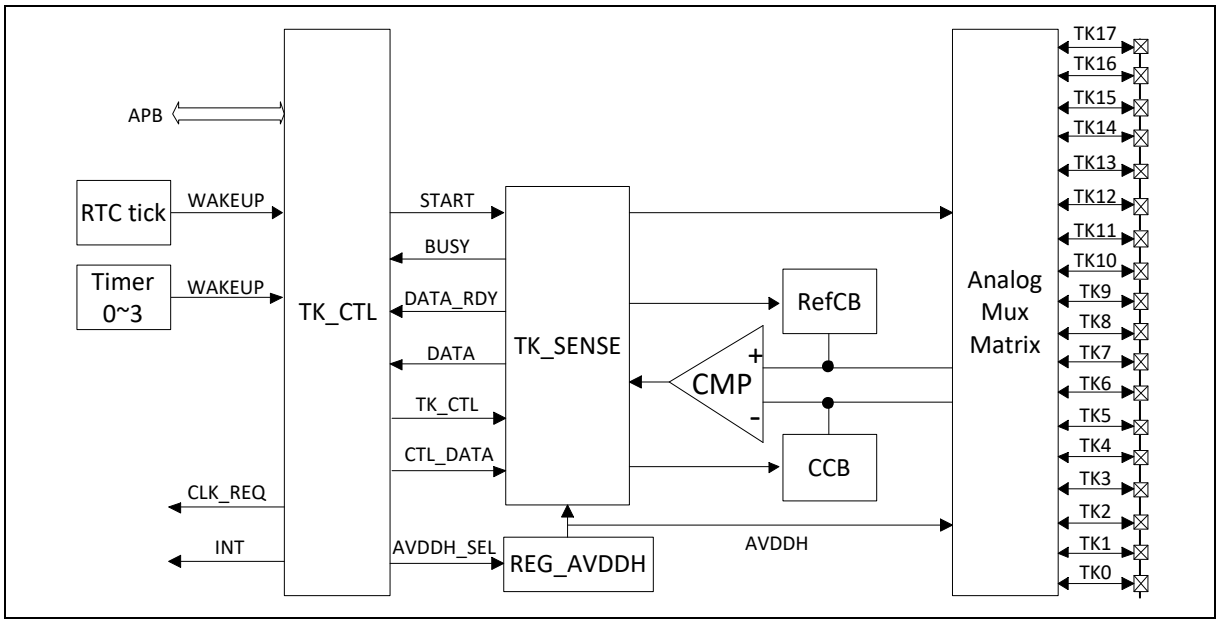


Figure 6.39-1 Touch Key Block Diagram

6.39.4 Basic Configuration

- The Touch Key clock is enabled in TKCKEN (CLK_APBCLK1[24]).
- The Touch Key function pins are configured in GPA_MFPL, GPB_MFPH, GPC_MFPL, GPC_MFPH, GPD_MFPL, GPD_MFPH, and GPG_MFPH Multiple Function Register.

6.39.5 Functional Description

When a finger touches on a key-pad, the capacitance of the key-pad sensed by Touch Key controller will be greater than it is not touched. The capacitance is measured by Touch Key controller analog front-end circuit, and user can read the sensed capacitance value to distinguish if a finger-touch event occurs or not.

There is a high sensing threshold control for each channel. It supports keys scanning automatically, until any threshold setting condition is met. Thus processor can keep its work or sleep state to save power-consumption without interrupt.

The sensitivity level of Touch Key controller is programmable. The key scanning time is programmable for higher sensitivity or scanning speed.

Any key wake-up function and proximity detection are also supported.

In this chapter, the letter x within register name denotes channel 0 to 17.

Capacitive Touch Key Fundamentals

The capacitance of the touch key without a finger touch is called “parasitic capacitance”, C_P. Parasitic capacitance results from the electric field between the touch key (including the sensor pad, traces, and vias) and other conductors in the system such as the ground planes, traces, any metal in the product’s chassis or enclosure, etc. The touch key pin and internal capacitances also contribute to the parasitic capacitance.

C_F is known as the finger capacitance. When a finger touches the touch key, the parasitic capacitance C_P and finger capacitance C_F are parallel to each other because both represent the capacitance between the touch key pin and ground. Therefore, the total capacitance C_T of the sensor is the sum of C_P and C_F.

$$C_T = C_P + C_F$$

The Touch Key controller converts the capacitance C_T into equivalent digital counts called raw data. Because a finger touch increases the total capacitance of the touch key, an increase in raw data indicates a finger touch.

Capacitive Touch Key Sensing Method

The touch key sensing method, as shown in Figure 6.39-2, implements two switching capacitor banks for injecting charges to C_P (or C_T) and C_R which is the parasitic capacitance of reference channel.

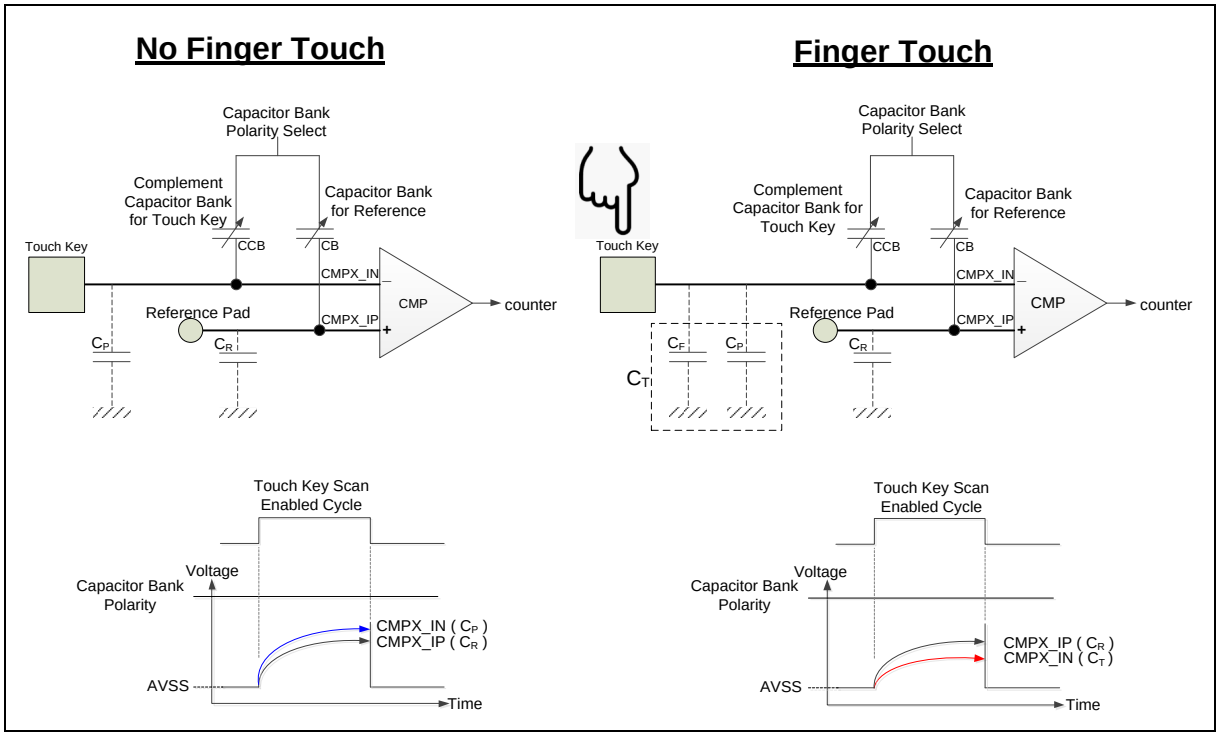


Figure 6.39-2 Touch Key Sensing Method

After touch key calibration, C_P and C_R are balanced with C_B and C_{CB}. A finger touches presents on sensing touch key results in C_T ($C_T = C_P + C_F$), makes negative input terminal voltage of the comparator lower than positive side and comparator output is “high”. This means C_T and C_R are not balanced, and the Touch Key controller will increase C_{CB} to C_{CB}’ to make C_T and C_R be balanced again. A finger touch can be detected by checking the difference of C_{CB} and C_{CB}’, as shown in Figure 6.39-3 .

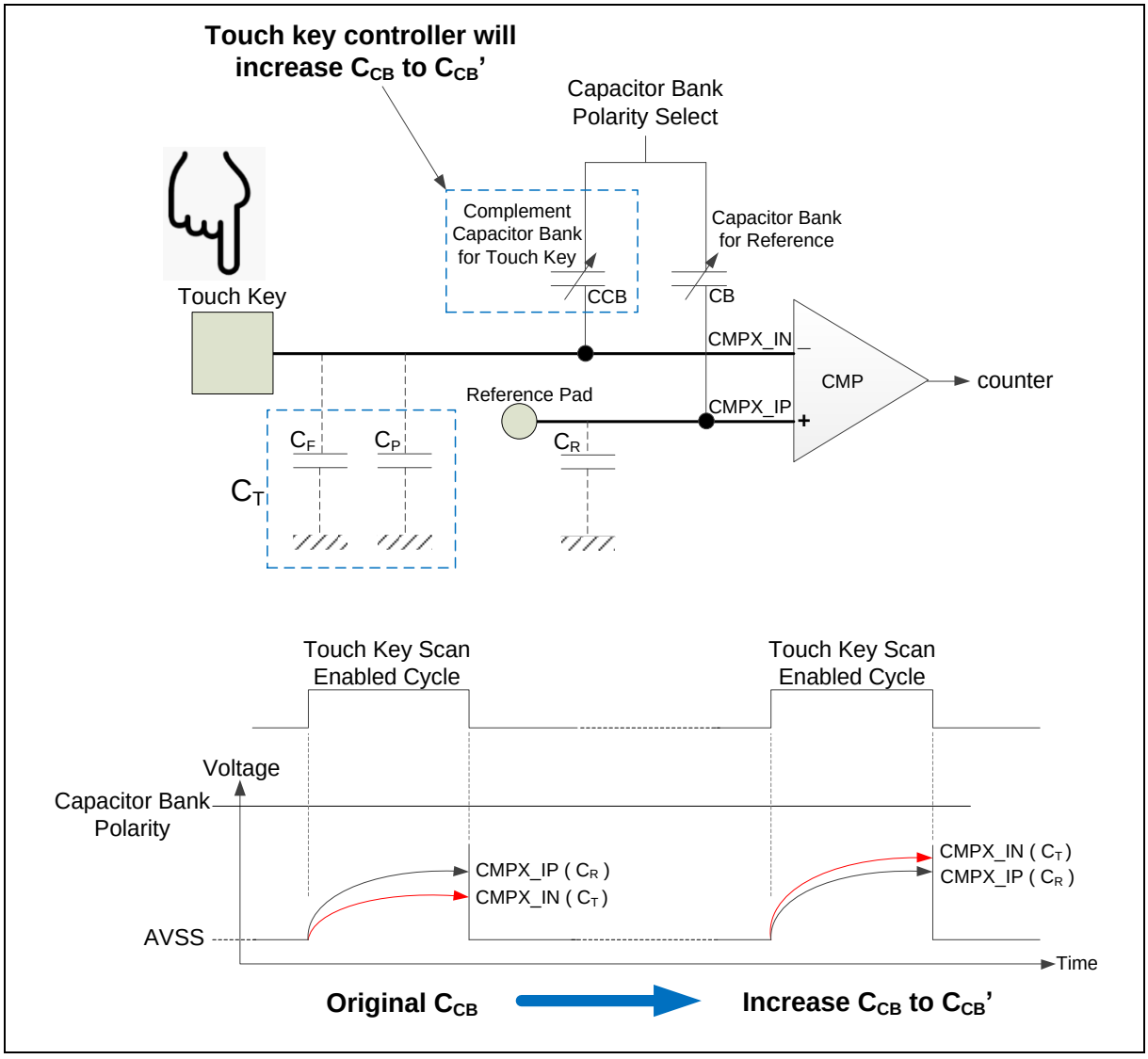


Figure 6.39-3 Finger Touch Detection Method

6.39.5.1 Touch Key Single Scan Mode

In this mode, user must set TK_EN (TK_SCANC[31]) and SCAN (TK_SCANC[24]) to high and other bits of TK_SCANC/TK_SCANC1~TK_POLC/TK_POLC1 must be set according to application requirements for sensitivity setting. Those channels enabled by corresponding bits of TKxSEN (TK_SCANC[16:0]/TK_SCANC1[0]) register will be scanned successively when scan is initiated. Once channel scanning starts, BUSY (TK_STA[0]) will be set until the scanning is complete. The SCIF (TK_STA[1]) register will be set when channel scanning is completed and the sensed data in TK_DATn registers is valid for reading, according to those channels enabled in TK_SCANC/TK_SCANC1, where n denotes number 0 to 5. SCIF (TK_STA[1]) can generate interrupt by setting SCIE (TK_INTEN[1]) register to high.

6.39.5.2 Touch Key Periodic Scan Mode

This mode supports automatic scanning periodically. User can set TK_EN (TK_SCANC[31]) to high and TRG_EN (TK_SCANC[28:25]). In addition to setting other bits of TK_SCANC/TK_SCANC1~TK_POLC/TK_POLC1 according to application requirements, user must set Timer0~3 or RTC tick properly for determining period between scanning. Besides, user can apply wake-up function to key scanning in a low power system. By setting high threshold control properly, periodic

scanning keeps even system is in sleep state, and until the system woken up by interrupt which means whether a finger-touch meets the threshold. Key scanning stops until the interrupt flags reset.

6.39.5.3 Reference Channel for Touch Key Scanning

There must be one channel at least as reference channel which PCB layout has special concern. TK16 is assigned as default reference channel automatically if there is no reference channel assigned by user. The Touch Key controller malfunctions without assigning physical reference channel(s) to pin(s) in the application.

6.39.5.4 Idle State and Polarity Control

The keys-pads are not being sensed always keep in Idle-State, and their output level can be pre-determined in TK_IDLSC/ TK_IDLSC1. For another sensing configuration, their output level is separated to two states: Idle-state and Polarity-State if their polarity control is activated in TK_POLC/ TK_POLC1. Their output level is pre-determined in TK_POLSEL/TK_POLSEL1.

6.39.5.5 Sensing Time

Sensing Time is the time needed for each key-pad sensing, which is composed of PULSET (TK_REFC[30:28]) and SENSET (TK_REFC[26:24]). The shorter Sensing Time comes with poor sensitivity and less power-consumption, vice versa.

Sensing Time = SENSET x PULSET

6.39.5.6 Sensitivity Configuration

Sensitivity can be adjusted by setting Sensing Time properly. Besides, selecting capacitor bank polarity source properly which is pre-determined in POL_CAP (TK_POLC[5:4]) also affects sensitivity. User can choose AVDDH as capacitor bank polarity source to have more sensitivity choice by programming AVDDH to proper level, which is pre-determined in AVDDH_S (TK_SCANC[23:20]).

6.39.5.7 Scan Complete without Threshold Control Interrupt

SCIF (TK_STA[1]) is always set when key-scan is complete. An interrupt is generated if SCIE TK_INTEN[1]) is set.

6.39.5.8 Scan Complete with Threshold Control Interrupt

Different to SCIF (TK_STA[1]), TKIFx (TK_STA[24:8] and TK_STA1[0]) is set if and only if the corresponding scan result meets its threshold control requirement. Besides, set SCTHIE (TK_INTEN[0]) rather than SCIE (TK_INTEN[1]) to generate interrupt when a potential key touch/release is detected. The threshold control for each channel can be pre-determined in TK_THCmn registers, where m,n denotes two adjacent channels' number, for example, TK_THC01.

6.39.5.9 Level Trigger Mode

In this alternative mode, TKIFx is set if TKDATx is greater than HTHx. TKDATx is greater than HTHx means a potential key touch occurs, as shown in Figure 6.39-4.

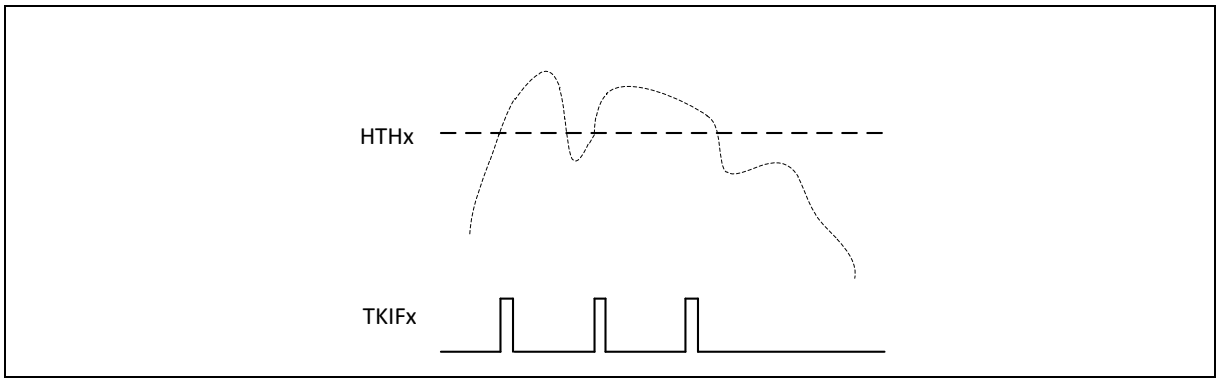


Figure 6.39-4 Touch Key Controller Interrupt Modes for Threshold Control

6.39.5.10 Low Power Consumption Solution

Non-stop key scanning in a low power system can be easily achieved. User can use Timer0 to wake up Touch Key controller for key scanning periodically. The Touch Key controller requests **MIRC** for key scanning only when it is woken up, and keeps system in power-down state. Interrupts are generated when a potential key scanning event occurs; otherwise, the Touch Key controller will terminate key scanning without interrupt and makes itself in sleep state.

6.39.5.11 Wake-up by Key Touch/Release

To enable threshold control for interrupt generation, system keeps in low power until any potential key touch/release is detected.

6.39.5.12 Wake-up by Any Key Touch

To save more power consumption for system, user can use Any Key Wake-up function by setting SCAN_ALL (TK_REFC[23]). All channels enabled but not assigned as reference channel are scanned, and scanning data is valid in TKDAT_ALL (TK_DAT4[15:8]). The CCBD_ALL (TK_CCBD4[15:8]) may be different from normal and need to be calibrated individually. Proximity detection is also achievable by using this mode.

6.39.5.13 Key Sensitivity Adjustment

The touch key capacitor bank data and reference capacitor bank data are 9 bits. There are three ways to generate these 9 bits. User can use EXT_CBSEL (TK_EXTCBC[1:0]) to decide which method to generate touch key capacitor bank data and reference capacitor bank data.

First of all, CCBDx (TK_CCBDx[8n+7:8n]) and CBDx (TK_REFCBDx[8n+7:8n]) are valid, but EXT_CCBDx (TK_EXTCCBDx[0]) and EXT_CBDx (TK_REFCBDx[0]) are invalid. The capacitor touch key sensing controller adds 1'b0 to the MSB of capacitor bank data and reference capacitor bank data, and the controller finally outputs 9 bits such as {1'b0, CCBDx} or {1'b0, CBDx}.

Secondly, CCBDx (TK_CCBDx[8n+7:8n]) and CBDx (TK_REFCBDx[8n+7:8n]) are valid, but EXT_CCBDx (TK_EXTCCBDx[0]) and EXT_CBDx (TK_REFCBDx[0]) are invalid. The capacitor touch key sensing controller shifts the capacitor bank data and reference capacitor bank data one bit to the left, and the controller finally output 9 bits such as {CCBDx, 1'b0} or {CBDx, 1'b0}.

Thirdly, CCBDx (TK_CCBDx[8n+7:8n]), CBDx (TK_REFCBDx[8n+7:8n]), EXT_CCBDx (TK_EXTCCBDx[0]) and EXT_CBDx (TK_REFCBDx[0]) are valid. The capacitor touch key sensing controller outputs 9 bits such as {EXT_CCBDx, CCBDx} or {EXT_CBDx, CBDx}.

According to different options, user needs to care whether EXT_CCBDx (TK_EXTCCBDx[0]) and EXT_CBDx (TK_REFCBDx[0]) need to be filled in.

6.39.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
TK Base Address: **TK_BA = 0x400C_4000**				
TK_SCANC	TK_BA+0x00	R/W	Touch Key Scan Control Register	0x0070_0000
TK_REFC	TK_BA+0x04	R/W	Touch Key Reference Control Register	0x1101_0000
TK_CCBD0	TK_BA+0x08	R/W	Touch Key Complement Capacitor Bank Data Register 0	0x0000_0000
TK_CCBD1	TK_BA+0x0C	R/W	Touch Key Complement Capacitor Bank Data Register 1	0x0000_0000
TK_CCBD2	TK_BA+0x10	R/W	Touch Key Complement Capacitor Bank Data Register 2	0x0000_0000
TK_CCBD3	TK_BA+0x14	R/W	Touch Key Complement Capacitor Bank Data Register 3	0x0000_0000
TK_CCBD4	TK_BA+0x18	R/W	Touch Key Complement Capacitor Bank Data Register 4	0x0000_0000
TK_IDLSC	TK_BA+0x1C	R/W	Touch Key Idle State Control Register	0xFFFF_FFFF
TK_POLSEL	TK_BA+0x20	R/W	Touch Key Polarity Select Register	0xFFFF_FFFF
TK_POLC	TK_BA+0x24	R/W	Touch Key Polarity Control Register	0x0000_001F
TK_STA	TK_BA+0x28	R/W	Touch Key Status Register	0x0000_0000
TK_DAT0	TK_BA+0x2C	R	Touch Key Data Register 0	0x0000_0000
TK_DAT1	TK_BA+0x30	R	Touch Key Data Register 1	0x0000_0000
TK_DAT2	TK_BA+0x34	R	Touch Key Data Register 2	0x0000_0000
TK_DAT3	TK_BA+0x38	R	Touch Key Data Register 3	0x0000_0000
TK_DAT4	TK_BA+0x3C	R	Touch Key Data Register 4	0x0000_0000
TK_INTEN	TK_BA+0x40	R/W	Touch Key Interrupt Enable Register	0x0000_0000
TK_THC01	TK_BA+0x44	R/W	Touch Key TK0/TK1 Threshold Control Register	0xFF00_FF00
TK_THC23	TK_BA+0x48	R/W	Touch Key TK2/TK3 Threshold Control Register	0xFF00_FF00
TK_THC45	TK_BA+0x4C	R/W	Touch Key TK4/TK5 Threshold Control Register	0xFF00_FF00
TK_THC67	TK_BA+0x50	R/W	Touch Key TK6/TK7 Threshold Control Register	0xFF00_FF00
TK_THC89	TK_BA+0x54	R/W	Touch Key TK8/TK9 Threshold Control Register	0xFF00_FF00
TK_THC1011	TK_BA+0x58	R/W	Touch Key TK10/TK11 Threshold Control Register	0xFF00_FF00
TK_THC1213	TK_BA+0x5C	R/W	Touch Key TK12/TK13 Threshold Control Register	0xFF00_FF00
TK_THC1415	TK_BA+0x60	R/W	Touch Key TK14/TK15 Threshold Control Register	0xFF00_FF00
TK_THC16	TK_BA+0x64	R/W	Touch Key TK16 Threshold Control Register	0xFF00_FF00
TK_REFCBD0	TK_BA+0x68	R/W	Touch Key Reference Capacitor Bank Data Register 0	0x0000_0000

TK_REFCBD1	TK_BA+0x6C	R/W	Touch Key Reference Capacitor Bank Data Register 1	0x0000_0000
TK_REFCBD2	TK_BA+0x70	R/W	Touch Key Reference Capacitor Bank Data Register 2	0x0000_0000
TK_REFCBD3	TK_BA+0x74	R/W	Touch Key Reference Capacitor Bank Data Register 3	0x0000_0000
TK_REFCBD4	TK_BA+0x78	R/W	Touch Key Reference Capacitor Bank Data Register 4	0x0000_0000
TK_SCANC1	TK_BA+0x80	R/W	Touch Key Scan Control Register 1	0x0000_0000
TK_REFC1	TK_BA+0x84	R/W	Touch Key Reference Control Register 1	0x0000_0000
TK_CCBD5	TK_BA+0x88	R/W	Touch Key Complement Capacitor Bank Data Register 5	0x0000_0000
TK_IDLSC1	TK_BA+0x9C	R/W	Touch Key Idle State Control Register 1	0x0000_0003
TK_POLSEL1	TK_BA+0xA0	R/W	Touch Key Polarity Select Register 1	0x0000_0003
TK_POLC1	TK_BA+0xA4	R/W	Touch Key Polarity Control Register 1	0x0000_0000
TK_STA1	TK_BA+0xA8	R/W	Touch Key Status Register 1	0x0000_0000
TK_DAT5	TK_BA+0xAC	R	Touch Key Data Register 5	0x0000_0000
TK_THC17	TK_BA+0xC4	R/W	Touch Key TK17 Threshold Control Register	0x0000_FF00
TK_REFCBD5	TK_BA+0xE8	R/W	Touch Key Reference Capacitor Bank Data Register 5	0x0000_0000
TK_EXTCBC	TK_BA+0x100	R/W	Touch Key Extend Capacitor Bank Control Register	0x0000_0000
TK_EXTCCBD0	TK_BA+0x104	R/W	Touch Key Extend Complement Capacitor Bank Data Register 0	0x0000_0000
TK_EXTCCBD1	TK_BA+0x108	R/W	Touch Key Extend Complement Capacitor Bank Data Register 1	0x0000_0000
TK_EXTCCBD2	TK_BA+0x10C	R/W	Touch Key Extend Complement Capacitor Bank Data Register 2	0x0000_0000
TK_EXTCCBD3	TK_BA+0x110	R/W	Touch Key Extend Complement Capacitor Bank Data Register 3	0x0000_0000
TK_EXTCCBD4	TK_BA+0x114	R/W	Touch Key Extend Complement Capacitor Bank Data Register 4	0x0000_0000
TK_EXTCCBD5	TK_BA+0x118	R/W	Touch Key Extend Complement Capacitor Bank Data Register 5	0x0000_0000
TK_EXTREFCBD0	TK_BA+0x130	R/W	Touch Key Extend Reference Capacitor Bank Data Register 0	0x0000_0000
TK_EXTREFCBD1	TK_BA+0x134	R/W	Touch Key Extend Reference Capacitor Bank Data Register 1	0x0000_0000
TK_EXTREFCBD2	TK_BA+0x138	R/W	Touch Key Extend Reference Capacitor Bank Data Register 2	0x0000_0000
TK_EXTREFCBD3	TK_BA+0x13C	R/W	Touch Key Extend Reference Capacitor Bank Data Register 3	0x0000_0000
TK_EXTREFCBD4	TK_BA+0x140	R/W	Touch Key Extend Reference Capacitor Bank Data Register 4	0x0000_0000
TK_EXTREFCBD5	TK_BA+0x144	R/W	Touch Key Extend Reference Capacitor Bank Data Register 5	0x0000_0000

6.39.7 Register Description

Touch Key Scan Control Register (TK_SCANC)

Register	Offset	R/W	Description	Reset Value
TK_SCANC	TK_BA+0x00	R/W	Touch Key Scan Control Register	0x0070_0000

31	30	29	28	27	26	25	24
TK_EN	Reserved		TRG_EN				SCAN
23	22	21	20	19	18	17	16
AVDDH_S				Reserved			TK16SEN
15	14	13	12	11	10	9	8
TK15SEN	TK14SEN	TK13SEN	TK12SEN	TK11SEN	TK10SEN	TK9SEN	TK8SEN
7	6	5	4	3	2	1	0
TK7SEN	TK6SEN	TK5SEN	TK4SEN	TK3SEN	TK2SEN	TK1SEN	TK0SEN

Bits	Description	
[31]	**TK_EN**	**Touch Key Scan Enable Bit** 0 = Touch Key function Disabled. 1 = Touch Key function Enabled.
[30:29]	**Reserved**	Reserved.
[28:25]	**TRG_EN**	**Trigger Enable Bit** Timer0~3 and RTC tick trigger key scan periodically Enabled. Key scan will be initiated by timer or RTC periodically. 0000 = Trigger Disabled. 0001 = Timer 0 trigger Enabled. 0010 = Timer 1 trigger Enabled. 0011 = Timer 2 trigger Enabled. 0100 = Timer 3 trigger Enabled. 0101 = RTC tick event trigger Enabled. 0110 = Reserved. 0111 = Reserved. 1000 = Reserved. 1001 = Reserved. 1010 = Reserved. 1011 = Reserved. 1100 = Reserved. 1101 = Reserved. 1110 = Reserved. 1111 = Reserved.
[24]	**SCAN**	**Scan** Writing “1” to this bit will immediately initiate key scan on all channels which are enabled. This bit will be self-cleared after key scan is started.

[23:20]	**AVDDH_S**	**AVDDH Voltage Select** 0000 = 1/16 V_{DD}. 0001 = 1/8 V_{DD}. 0010 = 3/16 V_{DD}. 0011 = 1/4 V_{DD}. 0100 = 5/16 V_{DD}. 0101 = 3/8 V_{DD}. 0110 = 7/16 V_{DD}. 0111 = 1/2 V_{DD}. 1000 = 1/32 V_{DD}. 1001 = 1/16 V_{DD}. 1010 = 3/32 V_{DD}. 1011 = 1/8 V_{DD}. 1100 = 5/32 V_{DD}. 1101 = 3/16 V_{DD}. 1110 = 7/32 V_{DD}. 1111 = 1/4 V_{DD}.
[19:17]	**Reserved**	Reserved.
[16]	**TK16SEN**	**TK16 Scan Enable Bit** This bit is ignored if TK16REN (TK_REFC[16]) is “1”. 0 = TKDAT16 (TK_DAT4[7:0]) is invalid. 1 = TK16 is always enabled for Touch Key scan. TKDAT16 (TK_DAT4[7:0]) is valid.
[15]	**TK15SEN**	**TK15 Scan Enable Bit** This bit is ignored if TK15REN (TK_REFC[15]) is “1”. 0 = TKDAT15 (TK_DAT3[31:24]) is invalid. 1 = TK15 is always enabled for Touch Key scan. TKDAT15 (TK_DAT3[31:24]) is valid.
[14]	**TK14SEN**	**TK14 Scan Enable Bit** This bit is ignored if TK14REN (TK_REFC[14]) is “1”. 0 = TKDAT14 (TK_DAT3[23:16]) is invalid. 1 = TK14 is always enabled for Touch Key scan. TKDAT14 (TK_DAT3[23:16]) is valid.
[13]	**TK13SEN**	**TK13 Scan Enable Bit** This bit is ignored if TK13REN (TK_REFC[13]) is “1”. 0 = TKDAT13 (TK_DAT3[15:8]) is invalid. 1 = TK13 is always enabled for Touch Key scan. TKDAT13 (TK_DAT3[15:8]) is valid.
[12]	**TK12SEN**	**TK12 Scan Enable Bit** This bit is ignored if TK12REN (TK_REFC[12]) is “1”. 0 = TKDAT12 (TK_DAT3[7:0]) is invalid. 1 = TK12 is always enabled for Touch Key scan. TKDAT12 (TK_DAT3[7:0]) is valid.
[11]	**TK11SEN**	**TK11 Scan Enable Bit** This bit is ignored if TK11REN (TK_REFC[11]) is “1”. 0 = TKDAT11 (TK_DAT2[31:24]) is invalid. 1 = TK11 is always enabled for Touch Key scan. TKDAT11 (TK_DAT2[31:24]) is valid.
[10]	**TK10SEN**	**TK10 Scan Enable Bit** This bit is ignored if TK10REN (TK_REFC[10]) is “1”. 0 = TKDAT10 (TK_DAT2[23:16]) is invalid. 1 = TK10 is always enabled for Touch Key scan. TKDAT10 (TK_DAT2[23:16]) is valid.

[9]	**TK9SEN**	**TK9 Scan Enable Bit** This bit is ignored if TK9REN (TK_REFC[9]) is “1”. 0 = TKDAT9 (TK_DAT2[15:8]) is invalid. 1 = TK9 is always enabled for Touch Key scan. TKDAT9 (TK_DAT2[15:8]) is valid.
[8]	**TK8SEN**	**TK8 Scan Enable Bit** This bit is ignored if TK8REN (TK_REFC[8]) is “1”. 0 = TKDAT8 (TK_DAT2[7:0]) is invalid. 1 = TK8 is always enabled for Touch Key scan. TKDAT8 (TK_DAT2[7:0]) is valid.
[7]	**TK7SEN**	**TK7 Scan Enable Bit** This bit is ignored if TK7REN (TK_REFC[7]) is “1”. 0 = TKDAT7 (TK_DAT1[31:24]) is invalid. 1 = TK7 is always enabled for Touch Key scan. TKDAT7 (TK_DAT1[31:24]) is valid.
[6]	**TK6SEN**	**TK6 Scan Enable Bit** This bit is ignored if TK6REN (TK_REFC[6]) is “1”. 0 = TKDAT6 (TK_DAT1[23:16]) is invalid. 1 = TK6 is always enabled for Touch Key scan. TKDAT6 (TK_DAT1[23:16]) is valid.
[5]	**TK5SEN**	**TK5 Scan Enable Bit** This bit is ignored if TK5REN (TK_REFC[5]) is “1”. 0 = TKDAT5 (TK_DAT1[15:8]) is invalid. 1 = TK5 is always enabled for Touch Key scan. TKDAT5 (TK_DAT1[15:8]) is valid.
[4]	**TK4SEN**	**TK4 Scan Enable Bit** This bit is ignored if TK4REN (TK_REFC[4]) is “1”. 0 = TKDAT4 (TK_DAT1[7:0]) is invalid. 1 = TK4 is always enabled for Touch Key scan. TKDAT4 (TK_DAT1[7:0]) is valid.
[3]	**TK3SEN**	**TK3 Scan Enable Bit** 0 = TKDAT3 (TK_DAT0[31:24]) is invalid. 1 = TK3 is always enabled for Touch Key scan. TKDAT3 (TK_DAT0[31:24]) is valid. This bit is ignored if TK3REN (TK_REFC[3]) is “1”.
[2]	**TK2SEN**	**TK2 Scan Enable Bit** This bit is ignored if TK2REN (TK_REFC[2]) is “1”. 0 = TKDAT2 (TK_DAT0[23:16]) is invalid. 1 = TK2 is always enabled for Touch Key scan. TKDAT2 (TK_DAT0[23:16]) is valid.
[1]	**TK1SEN**	**TK1 Scan Enable Bit** This bit is ignored if TK1REN (TK_REFC[1]) is “1”. 0 = TKDAT1 (TK_DAT0[15:8]) is invalid. 1 = TK1 is always enabled for Touch Key scan. TKDAT1 (TK_DAT0[15:8]) is valid.
[0]	**TK0SEN**	**TK0 Scan Enable Bit** This bit is ignored if TK0REN (TK_REFC[0]) is “1” except SCAN_ALL (TK_REFC[23]) is “1”. 0 = TKDAT0 (TK_DAT0[7:0]) is invalid. 1 = TK0 is always enabled for Touch Key scan. TKDAT0 (TK_DAT0[7:0]) is valid.

Touch Key Reference Control Register (TK_REFC)

Register	Offset	R/W	Description	Reset Value
TK_REFC	TK_BA+0x04	R/W	Touch Key Reference Control Register	0x1101_0000

31	30	29	28	27	26	25	24
Reserved	PULSET			Reserved	SENSET		
23	22	21	20	19	18	17	16
SCAN_ALL	Reserved						TK16REN
15	14	13	12	11	10	9	8
TK15REN	TK14REN	TK13REN	TK12REN	TK11REN	TK10REN	TK9REN	TK8REN
7	6	5	4	3	2	1	0
TK7REN	TK6REN	TK5REN	TK4REN	TK3REN	TK2REN	TK1REN	TK0REN

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**PULSET**	**Touch Key Sensing Pulse Width Time Control** 000 = 1us. 001 = 2us. 010 = 4us. 011 = 8us. 100 = Reserved. 101 = Reserved. 110 = Reserved. 111 = 500ns.
[27]	**Reserved**	Reserved.
[26:24]	**SENSET**	**Touch Key Sensing Time Control** 000 = 128 x PULSET. 001 = 255 x PULSET. 010 = 511 x PULSET. 011 = 1023 x PULSET. 100 = 8 x PULSET. 101 = 16 x PULSET. 110 = 32 x PULSET. 111 = 64 x PULSET.
[23]	**SCAN_ALL**	**All Keys Scan Enable Bit** This function is used for low power key scanning operation. TKDAT_ALL (TK_DAT4[15:8]) is the only one valid data when key scan is complete. 0 = All Keys Scan function Disabled. 1 = All Keys Scan function Enabled.
[22:17]	**Reserved**	Reserved.
[16]	**TK16REN**	**TK16 Reference Enable Bit**

		0 = TK16 is not reference. 1 = TK16 is set as reference, and TKDAT16 (TK_DAT4[7:0]) is invalid. **Note:** This bit is forced to “1” automatically if none is set as reference.
[15]	**TK15REN**	**TK15 Reference Enable Bit** 0 = TK15 is not reference. 1 = TK15 is set as reference, and TKDAT15 (TK_DAT3[31:24]) is invalid.
[14]	**TK14REN**	**TK14 Reference Enable Bit** 0 = TK14 is not reference. 1 = TK14 is set as reference, and TKDAT14 (TK_DAT3[23:16]) is invalid.
[13]	**TK13REN**	**TK13 Reference Enable Bit** 0 = TK13 is not reference. 1 = TK13 is set as reference, and TKDAT13 (TK_DAT3[15:8]) is invalid.
[12]	**TK12REN**	**TK12 Reference Enable Bit** 0 = TK12 is not reference. 1 = TK12 is set as reference, and TKDAT12 (TK_DAT3[7:0]) is invalid.
[11]	**TK11REN**	**TK11 Reference Enable Bit** 0 = TK11 is not reference. 1 = TK11 is set as reference, and TKDAT11 (TK_DAT2[31:24]) is invalid.
[10]	**TK10REN**	**TK10 Reference Enable Bit** 0 = TK10 is not reference. 1 = TK10 is set as reference, and TKDAT10 (TK_DAT2[23:16]) is invalid.
[9]	**TK9REN**	**TK9 Reference Enable Bit** 0 = TK9 is not reference. 1 = TK9 is set as reference, and TKDAT9 (TK_DAT2[15:8]) is invalid.
[8]	**TK8REN**	**TK8 Reference Enable Bit** 0 = TK8 is not reference. 1 = TK8 is set as reference, and TKDAT8 (TK_DAT2[7:0]) is invalid.
[7]	**TK7REN**	**TK7 Reference Enable Bit** 0 = TK7 is not reference. 1 = TK7 is set as reference, and TKDAT7 (TK_DAT1[31:24]) is invalid.
[6]	**TK6REN**	**TK6 Reference Enable Bit** 0 = TK6 is not reference. 1 = TK6 is set as reference, and TKDAT6 (TK_DAT1[23:16]) is invalid.
[5]	**TK5REN**	**TK5 Reference Enable Bit** 0 = TK5 is not reference. 1 = TK5 is set as reference, and TKDAT5 (TK_DAT1[15:8]) is invalid.
[4]	**TK4REN**	**TK4 Reference Enable Bit** 0 = TK4 is not reference. 1 = TK4 is set as reference, and TKDAT4 (TK_DAT1[7:0]) is invalid.
[3]	**TK3REN**	**TK3 Reference Enable Bit** 0 = TK3 is not reference. 1 = TK3 is set as reference, and TKDAT3 (TK_DAT0[31:24]) is invalid.
[2]	**TK2REN**	**TK2 Reference Enable Bit** 0 = TK2 is not reference.

		1 = TK2 is set as reference, and TKDAT2 (TK_DAT0[23:16]) is invalid.
[1]	**TK1REN**	**TK1 Reference Enable Bit** 0 = TK1 is not reference. 1 = TK1 is set as reference, and TKDAT1 (TK_DAT0[15:8]) is invalid.
[0]	**TK0REN**	**TK0 Reference Enable Bit** 0 = TK0 is not reference. 1 = TK0 is set as reference, and TKDAT0 (TK_DAT0[7:0]) is invalid.

Touch Key Complement Capacitor Bank Data Register 0 (TK_CCBD0)

Register	Offset	R/W	Description	Reset Value
TK_CCBD0	TK_BA+0x08	R/W	Touch Key Complement Capacitor Bank Data Register 0	0x0000_0000

31	30	29	28	27	26	25	24
CCBD3							
23	22	21	20	19	18	17	16
CCBD2							
15	14	13	12	11	10	9	8
CCBD1							
7	6	5	4	3	2	1	0
CCBD0							

Bits	Description	
[31:24]	CCBD3	**TK3 Complement CB Data** These bits are used for TK3 sensitivity adjustment.
[23:16]	CCBD2	**TK2 Complement CB Data** These bits are used for TK2 sensitivity adjustment.
[15:8]	CCBD1	**TK1 Complement CB Data** These bits are used for TK1 sensitivity adjustment.
[7:0]	CCBD0	**TK0 Complement CB Data** These bits are used for TK0 sensitivity adjustment.

Touch Key Complement Capacitor Bank Data Register 1 (TK_CCBD1)

Register	Offset	R/W	Description	Reset Value
TK_CCBD1	TK_BA+0x0C	R/W	Touch Key Complement Capacitor Bank Data Register 1	0x0000_0000

31	30	29	28	27	26	25	24
CCBD7							
23	22	21	20	19	18	17	16
CCBD6							
15	14	13	12	11	10	9	8
CCBD5							
7	6	5	4	3	2	1	0
CCBD4							

Bits	Description	
[31:24]	CCBD7	**TK7 Complement CB Data** These bits are used for TK7 sensitivity adjustment.
[23:16]	CCBD6	**TK6 Complement CB Data** These bits are used for TK6 sensitivity adjustment.
[15:8]	CCBD5	**TK5 Complement CB Data** These bits are used for TK5 sensitivity adjustment.
[7:0]	CCBD4	**TK4 Complement CB Data** These bits are used for TK4 sensitivity adjustment.

Touch Key Complement Capacitor Bank Data Register 2 (TK_CCBD2)

Register	Offset	R/W	Description	Reset Value
TK_CCBD2	TK_BA+0x10	R/W	Touch Key Complement Capacitor Bank Data Register 2	0x0000_0000

31	30	29	28	27	26	25	24
CCBD11							
23	22	21	20	19	18	17	16
CCBD10							
15	14	13	12	11	10	9	8
CCBD9							
7	6	5	4	3	2	1	0
CCBD8							

Bits	Description	
[31:24]	CCBD11	**TK11 Complement CB Data** These bits are used for TK11 sensitivity adjustment.
[23:16]	CCBD10	**TK10 Complement CB Data** These bits are used for TK10 sensitivity adjustment.
[15:8]	CCBD9	**TK9 Complement CB Data** These bits are used for TK9 sensitivity adjustment.
[7:0]	CCBD8	**TK8 Complement CB Data** These bits are used for TK8 sensitivity adjustment.

Touch Key Complement Capacitor Bank Data Register 3 (TK_CCBD3)

Register	Offset	R/W	Description	Reset Value
TK_CCBD3	TK_BA+0x14	R/W	Touch Key Complement Capacitor Bank Data Register 3	0x0000_0000

31	30	29	28	27	26	25	24
CCBD15							
23	22	21	20	19	18	17	16
CCBD14							
15	14	13	12	11	10	9	8
CCBD13							
7	6	5	4	3	2	1	0
CCBD12							

Bits	Description	
[31:24]	**CCBD15**	**TK15 Complement CB Data** These bits are used for TK15 sensitivity adjustment.
[23:16]	**CCBD14**	**TK14 Complement CB Data** These bits are used for TK14 sensitivity adjustment.
[15:8]	**CCBD13**	**TK13 Complement CB Data** These bits are used for TK13 sensitivity adjustment.
[7:0]	**CCBD12**	**TK12 Complement CB Data** These bits are used for TK12 sensitivity adjustment.

Touch Key Complement Capacitor Bank Data Register 4 (TK_CCBD4)

Register	Offset	R/W	Description	Reset Value
TK_CCBD4	TK_BA+0x18	R/W	Touch Key Complement Capacitor Bank Data Register 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CCBD_ALL							
7	6	5	4	3	2	1	0
CCBD16							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**Reserved**	Reserved.
[15:8]	**CCBD_ALL**	**All Keys Scan Complement CB Data** These bits are used for All Key Scan sensitivity adjustment.
[7:0]	**CCBD16**	**TK16 Complement CB Data** These bits are used for TK16 sensitivity adjustment.

Touch Key Idle State Control Register (TK_IDLSC)

Register	Offset	R/W	Description	Reset Value
TK_IDLSC	TK_BA+0x1C	R/W	Touch Key Idle State Control Register	0xFFFF_FFFF

31	30	29	28	27	26	25	24
IDLS15		IDLS14		IDLS13		IDLS12	
23	22	21	20	19	18	17	16
IDLS11		IDLS10		IDLS9		IDLS8	
15	14	13	12	11	10	9	8
IDLS7		IDLS6		IDLS5		IDLS4	
7	6	5	4	3	2	1	0
IDLS3		IDLS2		IDLS1		IDLS0	

Bits	Description	
[31:30]	**IDLS15**	**TK15 Idle State Control** These bits are ignored if both TK15SEN (TK_SCANC[15]) and POLEN15 (TK_POLC[23]) are “0” or TK15REN (TK_REFC[15]) is “1”. 00 = TK15 connected to Gnd. 01 = TK15 connected to AVDDH. 10 = TK15 connected to V_{DD}. 11 = TK15 connected to V_{DD}.
[29:28]	**IDLS14**	**TK14 Idle State Control** These bits are ignored if both TK14SEN (TK_SCANC[14]) and POLEN14 (TK_POLC[22]) are “0” or TK14REN (TK_REFC[14]) is “1”. 00 = TK14 connected to Gnd. 01 = TK14 connected to AVDDH. 10 = TK14 connected to V_{DD}. 11 = TK14 connected to V_{DD}.
[27:26]	**IDLS13**	**TK13 Idle State Control** These bits are ignored if both TK13SEN (TK_SCANC[13]) and POLEN13 (TK_POLC[21]) are “0” or TK13REN (TK_REFC[13]) is “1”. 00 = TK13 connected to Gnd. 01 = TK13 connected to AVDDH. 10 = TK13 connected to V_{DD}. 11 = TK13 connected to V_{DD}.
[25:24]	**IDLS12**	**TK12 Idle State Control** These bits are ignored if both TK12SEN (TK_SCANC[12]) and POLEN12 (TK_POLC[20]) are “0” or TK12REN (TK_REFC[12]) is “1”. 00 = TK12 connected to Gnd. 01 = TK12 connected to AVDDH. 10 = TK12 connected to V_{DD}. 11 = TK12 connected to V_{DD}.

[23:22]	**IDLS11**	**TK11 Idle State Control** These bits are ignored if both TK11SEN (TK_SCANC[11]) and POLEN11 (TK_POLC[19]) are “0” or TK11REN (TK_REFC[11]) is “1”. 00 = TK11 connected to Gnd. 01 = TK11 connected to AVDDH. 10 = TK11 connected to V_{DD}. 11 = TK11 connected to V_{DD}.
[21:20]	**IDLS10**	**TK10 Idle State Control** These bits are ignored if both TK10SEN (TK_SCANC[10]) and POLEN10 (TK_POLC[18]) are “0” or TK10REN (TK_REFC[10]) is “1”. 00 = TK10 connected to Gnd. 01 = TK10 connected to AVDDH. 10 = TK10 connected to V_{DD}. 11 = TK10 connected to V_{DD}.
[19:18]	**IDLS9**	**TK9 Idle State Control** These bits are ignored if both TK9SEN (TK_SCANC[9]) and POLEN9 (TK_POLC[17]) are “0” or TK9REN (TK_REFC[9]) is “1”. 00 = TK9 connected to Gnd. 01 = TK9 connected to AVDDH. 10 = TK9 connected to V_{DD}. 11 = TK9 connected to V_{DD}.
[17:16]	**IDLS8**	**TK8 Idle State Control** These bits are ignored if both TK8SEN (TK_SCANC[8]) and POLEN8 (TK_POLC[16]) are “0” or TK8REN (TK_REFC[8]) is “1”. 00 = TK8 connected to Gnd. 01 = TK8 connected to AVDDH. 10 = TK8 connected to V_{DD}. 11 = TK8 connected to V_{DD}.
[15:14]	**IDLS7**	**TK7 Idle State Control** These bits are ignored if both TK7SEN (TK_SCANC[7]) and POLEN7 (TK_POLC[15]) are “0” or TK7REN (TK_REFC[7]) is “1”. 00 = TK7 connected to Gnd. 01 = TK7 connected to AVDDH. 10 = TK7 connected to V_{DD}. 11 = TK7 connected to V_{DD}.
[13:12]	**IDLS6**	**TK6 Idle State Control** These bits are ignored if both TK6SEN (TK_SCANC[6]) and POLEN6 (TK_POLC[14]) are “0” or TK6REN (TK_REFC[6]) is “1”. 00 = TK6 connected to Gnd. 01 = TK6 connected to AVDDH. 10 = TK6 connected to V_{DD}. 11 = TK6 connected to V_{DD}.
[11:10]	**IDLS5**	**TK5 Idle State Control** These bits are ignored if both TK5SEN (TK_SCANC[5]) and POLEN5 (TK_POLC[13]) are “0” or TK5REN (TK_REFC[5]) is “1”. 00 = TK5 connected to Gnd. 01 = TK5 connected to AVDDH. 10 = TK5 connected to V_{DD}.

		11 = TK5 connected to V_{DD}.
[9:8]	**IDLS4**	**TK4 Idle State Control** These bits are ignored if both TK4SEN (TK_SCANC[4]) and POLEN4 (TK_POLC[12]) are “0” or TK4REN (TK_REFC[4]) is “1”. 00 = TK4 connected to Gnd. 01 = TK4 connected to AVDDH. 10 = TK4 connected to V_{DD}. 11 = TK4 connected to V_{DD}.
[7:6]	**IDLS3**	**TK3 Idle State Control** These bits are ignored if both TK3SEN (TK_SCANC[3]) and POLEN3 (TK_POLC[11]) are “0” or TK3REN (TK_REFC[3]) is “1”. 00 = TK3 connected to Gnd. 01 = TK3 connected to AVDDH. 10 = TK3 connected to V_{DD}. 11 = TK3 connected to V_{DD}.
[5:4]	**IDLS2**	**TK2 Idle State Control** These bits are ignored if both TK2SEN (TK_SCANC[2]) and POLEN2 (TK_POLC[10]) are “0” or TK2REN (TK_REFC[2]) is “1”. 00 = TK2 connected to Gnd. 01 = TK2 connected to AVDDH. 10 = TK2 connected to V_{DD}. 11 = TK2 connected to V_{DD}.
[3:2]	**IDLS1**	**TK1 Idle State Control** These bits are ignored if both TK1SEN (TK_SCANC[1]) and POLEN1 (TK_POLC[9]) are “0” or TK1REN (TK_REFC[1]) is “1”. 00 = TK1 connected to Gnd. 01 = TK1 connected to AVDDH. 10 = TK1 connected to V_{DD}. 11 = TK1 connected to V_{DD}.
[1:0]	**IDLS0**	**TK0 Idle State Control** These bits are ignored if both TK0SEN (TK_SCANC[0]) and POLEN0 (TK_POLC[8]) are “0” or TK0REN (TK_REFC[0]) is “1”. 00 = TK0 connected to Gnd. 01 = TK0 connected to AVDDH. 10 = TK0 connected to V_{DD}. 11 = TK0 connected to V_{DD}.

Touch Key Polarity Select Register (TK_POLSEL)

Register	Offset	R/W	Description	Reset Value
TK_POLSEL	TK_BA+0x20	R/W	Touch Key Polarity Select Register	0xFFFF_FFFF

31	30	29	28	27	26	25	24
POL15		POL14		POL13		POL12	
23	22	21	20	19	18	17	16
POL11		POL10		POL9		POL8	
15	14	13	12	11	10	9	8
POL7		POL6		POL5		POL4	
7	6	5	4	3	2	1	0
POL3		POL2		POL1		POL0	

Bits	Description	
[31:30]	**POL15**	**TK15 Polarity Select** These bits are ignored if POLEN15 (TK_POLC[23]) is “0” or TK15REN (TK_REFC[15]) is “1”. 00 = TK15 connected to Gnd. 01 = TK15 connected to AVDDH. 10 = TK15 connected to V_{DD}. 11 = TK15 connected to V_{DD}.
[29:28]	**POL14**	**TK14 Polarity Select** These bits are ignored if POLEN14 (TK_POLC[22]) is “0” or TK14REN (TK_REFC[14]) is “1”. 00 = TK14 connected to Gnd. 01 = TK14 connected to AVDDH. 10 = TK14 connected to V_{DD}. 11 = TK14 connected to V_{DD}.
[27:26]	**POL13**	**TK13 Polarity Select** These bits are ignored if POLEN13 (TK_POLC[21]) is “0” or TK13REN (TK_REFC[13]) is “1”. 00 = TK13 connected to Gnd. 01 = TK13 connected to AVDDH. 10 = TK13 connected to V_{DD}. 11 = TK13 connected to V_{DD}.
[25:24]	**POL12**	**TK12 Polarity Select** These bits are ignored if POLEN12 (TK_POLC[20]) is “0” or TK12REN (TK_REFC[12]) is “1”. 00 = TK12 connected to Gnd. 01 = TK12 connected to AVDDH. 10 = TK12 connected to V_{DD}. 11 = TK12 connected to V_{DD}.
[23:22]	**POL11**	**TK11 Polarity Select** These bits are ignored if POLEN11 (TK_POLC[19]) is “0” or TK11REN (TK_REFC[11]) is “1”. 00 = TK11 connected to Gnd.

		01 = TK11 connected to AVDDH. 10 = TK11 connected to V_{DD}. 11 = TK11 connected to V_{DD}.
[21:20]	**POL10**	**TK10 Polarity Select** These bits are ignored if POLEN10 (TK_POLC[18]) is “0” or TK10REN (TK_REFC[10]) is “1”. 00 = TK10 connected to Gnd. 01 = TK10 connected to AVDDH. 10 = TK10 connected to V_{DD}. 11 = TK10 connected to V_{DD}.
[19:18]	**POL9**	**TK9 Polarity Select** These bits are ignored if POLEN9 (TK_POLC[17]) is “0” or TK9REN (TK_REFC[9]) is “1”. 00 = TK9 connected to Gnd. 01 = TK9 connected to AVDDH. 10 = TK9 connected to V_{DD}. 11 = TK9 connected to V_{DD}.
[17:16]	**POL8**	**TK8 Polarity Select** These bits are ignored if POLEN8 (TK_POLC[16]) is “0” or TK8REN (TK_REFC[8]) is “1”. 00 = TK8 connected to Gnd. 01 = TK8 connected to AVDDH. 10 = TK8 connected to V_{DD}. 11 = TK8 connected to V_{DD}.
[15:14]	**POL7**	**TK7 Polarity Select** These bits are ignored if POLEN7 (TK_POLC[15]) is “0” or TK7REN (TK_REFC[7]) is “1”. 00 = TK7 connected to Gnd. 01 = TK7 connected to AVDDH. 10 = TK7 connected to V_{DD}. 11 = TK7 connected to V_{DD}.
[13:12]	**POL6**	**TK6 Polarity Select** These bits are ignored if POLEN6 (TK_POLC[14]) is “0” or TK6REN (TK_REFC[6]) is “1”. 00 = TK6 connected to Gnd. 01 = TK6 connected to AVDDH. 10 = TK6 connected to V_{DD}. 11 = TK6 connected to V_{DD}.
[11:10]	**POL5**	**TK5 Polarity Select** These bits are ignored if POLEN5 (TK_POLC[13]) is “0” or TK5REN (TK_REFC[5]) is “1”. 00 = TK5 connected to Gnd. 01 = TK5 connected to AVDDH. 10 = TK5 connected to V_{DD}. 11 = TK5 connected to V_{DD}.
[9:8]	**POL4**	**TK4 Polarity Select** These bits are ignored if POLEN4 (TK_POLC[12]) is “0” or TK4REN (TK_REFC[4]) is “1”. 00 = TK4 connected to Gnd. 01 = TK4 connected to AVDDH. 10 = TK4 connected to V_{DD}. 11 = TK4 connected to V_{DD}.
[7:6]	**POL3**	**TK3 Polarity Select**

		These bits are ignored if POLEN3 (TK_POLC[11]) is “0” or TK3REN (TK_REFC[3]) is “1”. 00 = TK3 connected to Gnd. 01 = TK3 connected to AVDDH. 10 = TK3 connected to V_{DD}. 11 = TK3 connected to V_{DD}.
[5:4]	**POL2**	**TK2 Polarity Select** These bits are ignored if POLEN2 (TK_POLC[10]) is “0” or TK2REN (TK_REFC[2]) is “1”. 00 = TK2 connected to Gnd. 01 = TK2 connected to AVDDH. 10 = TK2 connected to V_{DD}. 11 = TK2 connected to V_{DD}.
[3:2]	**POL1**	**TK1 Polarity Select** These bits are ignored if POLEN1 (TK_POLC[9]) is “0” or TK1REN (TK_REFC[1]) is “1”. 00 = TK1 connected to Gnd. 01 = TK1 connected to AVDDH. 10 = TK1 connected to V_{DD}. 11 = TK1 connected to V_{DD}.
[1:0]	**POL0**	**TK0 Polarity Select** These bits are ignored if POLEN0 (TK_POLC[8]) is “0” or TK0REN (TK_REFC[0]) is “1”. 00 = TK0 connected to Gnd. 01 = TK0 connected to AVDDH. 10 = TK0 connected to V_{DD}. 11 = TK0 connected to V_{DD}.

Touch Key Polarity Control Register (TK_POLC)

Register	Offset	R/W	Description	Reset Value
TK_POLC	TK_BA+0x24	R/W	Touch Key Polarity Control Register	0x0000_001F

31	30	29	28	27	26	25	24
POL_INIT	Reserved						POLEN16
23	22	21	20	19	18	17	16
POLEN15	POLEN14	POLEN13	POLEN12	POLEN11	POLEN10	POLEN9	POLEN8
15	14	13	12	11	10	9	8
POLEN7	POLEN6	POLEN5	POLEN4	POLEN3	POLEN2	POLEN1	POLEN0
7	6	5	4	3	2	1	0
Reserved		POL_CAP		POL16		IDLS16	

Bits	Description	
[31]	**POL_INIT**	**Touch Key Sensing Initial Potential Control** 0 = Key pad is connected to Gnd before sensing. 1 = Key pad is connected to AVDDH before sensing.
[30:25]	**Reserved**	Reserved.
[24]	**POLEN16**	**TK16 Polarity Function Enable Bit** 0 = TK16 polarity function Disabled. 1 = TK16 polarity function Enabled.
[23]	**POLEN15**	**TK15 Polarity Function Enable Bit** 0 = TK15 polarity function Disabled. 1 = TK15 polarity function Enabled.
[22]	**POLEN14**	**TK14 Polarity Function Enable Bit** 0 = TK14 polarity function Disabled. 1 = TK14 polarity function Enabled.
[21]	**POLEN13**	**TK13 Polarity Function Enable Bit** 0 = TK13 polarity function Disabled. 1 = TK13 polarity function Enabled.
[20]	**POLEN12**	**TK12 Polarity Function Enable Bit** 0 = TK12 polarity function Disabled. 1 = TK12 polarity function Enabled.
[19]	**POLEN11**	**TK11 Polarity Function Enable Bit** 0 = TK11 polarity function Disabled. 1 = TK11 polarity function Enabled.
[18]	**POLEN10**	**TK10 Polarity Function Enable Bit** 0 = TK10 polarity function Disabled. 1 = TK10 polarity function Enabled.

[17]	**POLEN9**	**TK9 Polarity Function Enable Bit** 0 = TK9 polarity function Disabled. 1 = TK9 polarity function Enabled.
[16]	**POLEN8**	**TK8 Polarity Function Enable Bit** 0 = TK8 polarity function Disabled. 1 = TK8 polarity function Enabled.
[15]	**POLEN7**	**TK7 Polarity Function Enable Bit** 0 = TK7 polarity function Disabled. 1 = TK7 polarity function Enabled.
[14]	**POLEN6**	**TK6 Polarity Function Enable Bit** 0 = TK6 polarity function Disabled. 1 = TK6 polarity function Enabled.
[13]	**POLEN5**	**TK5 Polarity Function Enable Bit** 0 = TK5 polarity function Disabled. 1 = TK5 polarity function Enabled.
[12]	**POLEN4**	**TK4 Polarity Function Enable Bit** 0 = TK4 polarity function Disabled. 1 = TK4 polarity function Enabled.
[11]	**POLEN3**	**TK3 Polarity Function Enable Bit** 0 = TK3 polarity function Disabled. 1 = TK3 polarity function Enabled.
[10]	**POLEN2**	**TK2 Polarity Function Enable Bit** 0 = TK2 polarity function Disabled. 1 = TK2 polarity function Enabled.
[9]	**POLEN1**	**TK1 Polarity Function Enable Bit** 0 = TK1 polarity function Disabled. 1 = TK1 polarity function Enabled.
[8]	**POLEN0**	**TK0 Polarity Function Enable Bit** 0 = TK0 polarity function Disabled. 1 = TK0 polarity function Enabled.
[7:6]	**Reserved**	Reserved.
[5:4]	**POL_CAP**	**Capacitor Bank Polarity Select** 00 = Gnd. 01 = AVDDH. 10 = V_{DD}. 11 = V_{DD}.
[3:2]	**POL16**	**TK16 Polarity Control** These bits are ignored if POLEN16 (TK_POLC[24]) is “0” or TK16REN (TK_REFC[16]) is “1”. 00 = TK16 connected to Gnd. 01 = TK16 connected to AVDDH. 10 = TK16 connected to V_{DD}. 11 = TK16 connected to V_{DD}.
[1:0]	**IDLS16**	**TK16 Idle State Control** These bits are ignored if both TK16SEN (TK_SCANC[16]) and POLEN16 (TK_POLC[24]) are “0” or

		TK16REN (TK_REFC[16]) is “1”. 00 = TK16 connected to Gnd. 01 = TK16 connected to AVDDH. 10 = TK16 connected to V_{DD}. 11 = TK16 connected to V_{DD}.

Touch Key Status Register (TK_STA)

Register	Offset	R/W	Description	Reset Value
TK_STA	TK_BA+0x28	R/W	Touch Key Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							TKIF16
23	22	21	20	19	18	17	16
TKIF15	TKIF14	TKIF13	TKIF12	TKIF11	TKIF10	TKIF9	TKIF8
15	14	13	12	11	10	9	8
TKIF7	TKIF6	TKIF5	TKIF4	TKIF3	TKIF2	TKIF1	TKIF0
7	6	5	4	3	2	1	0
TKIF_ALL	TKIF	Reserved				SCIF	BUSY

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**TKIF16**	**TK16 Interrupt Flag** 0 = No threshold control event with TK16. 1 = Threshold control event occurs with TK16. **Note:** This bit will be cleared by writing a “1” to this bit.
[23]	**TKIF15**	**TK15 Interrupt Flag** 0 = No threshold control event with TK15. 1 = Threshold control event occurs with TK15. **Note:** This bit will be cleared by writing a “1” to this bit.
[22]	**TKIF14**	**TK14 Interrupt Flag** 0 = No threshold control event with TK14. 1 = Threshold control event occurs with TK14. **Note:** This bit will be cleared by writing a “1” to this bit.
[21]	**TKIF13**	**TK13 Interrupt Flag** 0 = No threshold control event with TK13. 1 = Threshold control event occurs with TK13. **Note:** This bit will be cleared by writing a “1” to this bit.
[20]	**TKIF12**	**TK12 Interrupt Flag** 0 = No threshold control event with TK12. 1 = Threshold control event occurs with TK12. **Note:** This bit will be cleared by writing a “1” to this bit.
[19]	**TKIF11**	**TK11 Interrupt Flag** 0 = No threshold control event with TK11. 1 = Threshold control event occurs with TK11. **Note:** This bit will be cleared by writing a “1” to this bit.
[18]	**TKIF10**	**TK10 Interrupt Flag**

		0 = No threshold control event with TK10. 1 = Threshold control event occurs with TK10. **Note:** This bit will be cleared by writing a “1” to this bit.
[17]	**TKIF9**	**TK9 Interrupt Flag** 0 = No threshold control event with TK9. 1 = Threshold control event occurs with TK9. **Note:** This bit will be cleared by writing a “1” to this bit.
[16]	**TKIF8**	**TK8 Interrupt Flag** 0 = No threshold control event with TK8. 1 = Threshold control event occurs with TK8. **Note:** This bit will be cleared by writing a “1” to this bit.
[15]	**TKIF7**	**TK7 Interrupt Flag** This bit will be cleared by writing a “1” to this bit. 0 = No threshold control event with TK7. 1 = Threshold control event occurs with TK7. **Note:** This bit will be cleared by writing a “1” to this bit.
[14]	**TKIF6**	**TK6 Interrupt Flag** This bit will be cleared by writing a “1” to this bit. 0 = No threshold control event with TK6. 1 = Threshold control event occurs with TK6.
[13]	**TKIF5**	**TK5 Interrupt Flag** This bit will be cleared by writing a “1” to this bit. 0 = No threshold control event with TK5. 1 = Threshold control event occurs with TK5.
[12]	**TKIF4**	**TK4 Interrupt Flag** This bit will be cleared by writing a “1” to this bit. 0 = No threshold control event with TK4. 1 = Threshold control event occurs with TK4.
[11]	**TKIF3**	**TK3 Interrupt Flag** This bit will be cleared by writing a “1” to this bit. 0 = No threshold control event with TK3. 1 = Threshold control event occurs with TK3.
[10]	**TKIF2**	**TK2 Interrupt Flag** This bit will be cleared by writing a “1” to this bit. 0 = No threshold control event with TK2. 1 = Threshold control event occurs with TK2.
[9]	**TKIF1**	**TK1 Interrupt Flag** 0 = No threshold control event with TK1. 1 = Threshold control event occurs with TK1. **Note:** This bit will be cleared by writing a “1” to this bit.
[8]	**TKIF0**	**TK0 Interrupt Flag** 0 = No threshold control event with TK0. 1 = Threshold control event occurs with TK0. **Note:** This bit will be cleared by writing a “1” to this bit.
[7]	**TKIF_ALL**	**All Keys Scan Interrupt Flag**

		0 = No threshold control event with All Keys Scan. 1 = Threshold control event occurs with All Keys Scan. **Note:** This bit will be cleared by writing a “1” to this bit.
[6]	**TKIF**	**Key Scan Interrupt Flag (Read Only)** 0 = No threshold control event with each Keys Scan. 1 = Threshold control event occurs with each Keys Scan. **Note:** This bit is 1 while any one of TKIF0~TKIF17 is 1.
[5:2]	**Reserved**	Reserved.
[1]	**SCIF**	**Touch Key Scan Complete Interrupt Flag** This bit will be cleared by writing a “1” to this bit. 0 = Key scan is proceeding and data is not ready for read. 1 = Key scan is complete and data is ready for read in TKDATx registers. **Note 1:** The Touch Key interrupt asserts if SCIE bit of TK_INTEN register is set. **Note 2:** The Touch Key interrupt also asserts if STHIE bit of TK_INTEN register is set and any channel data value is greater/less than its threshold setting. **Note 3:** This bit will be cleared by writing a “1” to this bit.
[0]	**BUSY**	**Touch Key Busy (Read Only)** 0 = Key scan is complete or stopped. 1 = Key scan is proceeding.

Touch Key Data Register 0 (TK_DAT0)

Register	Offset	R/W	Description	Reset Value
TK_DAT0	TK_BA+0x2C	R	Touch Key Data Register 0	0x0000_0000

31	30	29	28	27	26	25	24
TKDAT3							
23	22	21	20	19	18	17	16
TKDAT2							
15	14	13	12	11	10	9	8
TKDAT1							
7	6	5	4	3	2	1	0
TKDAT0							

Bits	Description	
[31:24]	TKDAT3	**TK3 Sensing Result Data (Read Only)** This data is invalid if TK3SEN (TK_SCANC[3]) is “0” or TK3REN (TK_REFC[3]) is “1”.
[23:16]	TKDAT2	**TK2 Sensing Result Data (Read Only)** This data is invalid if TK2SEN (TK_SCANC[2]) is “0” or TK2REN (TK_REFC[2]) is “1”.
[15:8]	TKDAT1	**TK1 Sensing Result Data (Read Only)** This data is invalid if TK1SEN (TK_SCANC[1]) is “0” or TK1REN (TK_REFC[1]) is “1”.
[7:0]	TKDAT0	**TK0 Sensing Result Data (Read Only)** This data is invalid if TK0SEN (TK_SCANC[0]) is “0” or TK0REN (TK_REFC[0]) is “1” except SCAN_ALL (TK_REFC[23]) is “1”.

Touch Key Data Register 1 (TK_DAT1)

Register	Offset	R/W	Description	Reset Value
TK_DAT1	TK_BA+0x30	R	Touch Key Data Register 1	0x0000_0000

31	30	29	28	27	26	25	24
TKDAT7							
23	22	21	20	19	18	17	16
TKDAT6							
15	14	13	12	11	10	9	8
TKDAT5							
7	6	5	4	3	2	1	0
TKDAT4							

Bits	Description	
[31:24]	TKDAT7	**TK7 Sensing Result Data (Read Only)** This data is invalid if TK7SEN (TK_SCANC[7]) is “0” or TK7REN (TK_REFC[7]) is “1”.
[23:16]	TKDAT6	**TK6 Sensing Result Data (Read Only)** This data is invalid if TK6SEN (TK_SCANC[6]) is “0” or TK6REN (TK_REFC[6]) is “1”.
[15:8]	TKDAT5	**TK5 Sensing Result Data (Read Only)** This data is invalid if TK5SEN (TK_SCANC[5]) is “0” or TK5REN (TK_REFC[5]) is “1”.
[7:0]	TKDAT4	**TK0 Sensing Result Data (Read Only)** This data is invalid if TK4SEN (TK_SCANC[4]) is “0” or TK4REN (TK_REFC[4]) is “1”.

Touch Key Data Register 2 (TK_DAT2)

Register	Offset	R/W	Description	Reset Value
TK_DAT2	TK_BA+0x34	R	Touch Key Data Register 2	0x0000_0000

31	30	29	28	27	26	25	24
TKDAT11							
23	22	21	20	19	18	17	16
TKDAT10							
15	14	13	12	11	10	9	8
TKDAT9							
7	6	5	4	3	2	1	0
TKDAT8							

Bits	Description	
[31:24]	**TKDAT11**	**TK11 Sensing Result Data (Read Only)** This data is invalid if TK11SEN (TK_SCANC[11]) is “0” or TK11REN (TK_REFC[11]) is “1”.
[23:16]	**TKDAT10**	**TK10 Sensing Result Data (Read Only)** This data is invalid if TK10SEN (TK_SCANC[10]) is “0” or TK10REN (TK_REFC[10]) is “1”.
[15:8]	**TKDAT9**	**TK9 Sensing Result Data (Read Only)** This data is invalid if TK9SEN (TK_SCANC[9]) is “0” or TK9REN (TK_REFC[9]) is “1”.
[7:0]	**TKDAT8**	**TK8 Sensing Result Data (Read Only)** This data is invalid if TK8SEN (TK_SCANC[8]) is “0” or TK8REN (TK_REFC[8]) is “1”.

Touch Key Data Register 3 (TK_DAT3)

Register	Offset	R/W	Description	Reset Value
TK_DAT3	TK_BA+0x38	R	Touch Key Data Register 3	0x0000_0000

31	30	29	28	27	26	25	24
TKDAT15							
23	22	21	20	19	18	17	16
TKDAT14							
15	14	13	12	11	10	9	8
TKDAT13							
7	6	5	4	3	2	1	0
TKDAT12							

Bits	Description	
[31:24]	TKDAT15	**TK15 Sensing Result Data (Read Only)** This data is invalid if TK15SEN (TK_SCANC[15]) is “0” or TK15REN (TK_REFC[15]) is “1”.
[23:16]	TKDAT14	**TK14 Sensing Result Data (Read Only)** This data is invalid if TK14SEN (TK_SCANC[14]) is “0” or TK14REN (TK_REFC[14]) is “1”.
[15:8]	TKDAT13	**TK13 Sensing Result Data (Read Only)** This data is invalid if TK13SEN (TK_SCANC[13]) is “0” or TK13REN (TK_REFC[13]) is “1”.
[7:0]	TKDAT12	**TK12 Sensing Result Data (Read Only)** This data is invalid if TK12SEN (TK_SCANC[12]) is “0” or TK12REN (TK_REFC[12]) is “1”.

Touch Key Data Register 4 (TK_DAT4)

Register	Offset	R/W	Description	Reset Value
TK_DAT4	TK_BA+0x3C	R	Touch Key Data Register 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TKDAT_ALL							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TKDAT16							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**TKDAT_ALL**	**All Keys Scan Sensing Result Data (Read Only)** This data is invalid if SCAN_ALL (TK_REFC[23]) is “0”.
[7:0]	**TKDAT16**	**TK16 Sensing Result Data (Read Only)** This data is invalid if TK16SEN (TK_SCANC[16]) is “0” or TK16REN (TK_REFC[16]) is “1”.

Touch Key Interrupt Enable Register (TK_INTEN)

Register	Offset	R/W	Description	Reset Value
TK_INTEN	TK_BA+0x40	R/W	Touch Key Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**SCIE**	**SCTHIE**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**SCIE**	**Touch Key Scan Complete Interrupt Enable Bit** 0 = Key scan complete without threshold control interrupt Disabled. 1 = Key scan complete without threshold control interrupt Enabled.
[0]	**SCTHIE**	**Touch Key Scan Complete with High Threshold Control Interrupt Enable Bit** 0 = Key scan complete with threshold control interrupt Disabled. 1 = Key scan complete with threshold control interrupt Enabled.

Touch Key TK0/TK1 Threshold Control Register (TK_THC01)

Register	Offset	R/W	Description	Reset Value
TK_THC01	TK_BA+0x44	R/W	Touch Key TK0/TK1 Threshold Control Register	0xFF00_FF00

31	30	29	28	27	26	25	24
HTH1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HTH0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**HTH1**	**High Threshold of TK1** High level for TK1 threshold control.
[23:16]	**Reserved**	Reserved.
[15:8]	**HTH0**	**High Threshold of TK0** High level for TK0 threshold control.
[7:0]	**Reserved**	Reserved.

Touch Key TK2/TK3 Threshold Control Register (TK_THC23)

Register	Offset	R/W	Description	Reset Value
TK_THC23	TK_BA+0x48	R/W	Touch Key TK2/TK3 Threshold Control Register	0xFF00_FF00

31	30	29	28	27	26	25	24
HTH3							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HTH2							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**HTH3**	**High Threshold of TK3** High level for TK3 threshold control.
[23:16]	**Reserved**	Reserved.
[15:8]	**HTH2**	**High Threshold of TK2** High level for TK2 threshold control.
[7:0]	**Reserved**	Reserved.

Touch Key TK4/TK5 Threshold Control Register (TK_THC45)

Register	Offset	R/W	Description	Reset Value
TK_THC45	TK_BA+0x4C	R/W	Touch Key TK4/TK5 Threshold Control Register	0xFF00_FF00

31	30	29	28	27	26	25	24
HTH5							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HTH4							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**HTH5**	**High Threshold of TK5** High level for TK5 threshold control.
[23:16]	**Reserved**	Reserved.
[15:8]	**HTH4**	**High Threshold of TK4** High level for TK4 threshold control.
[7:0]	**Reserved**	Reserved.

Touch Key TK6/TK7 Threshold Control Register (TK_THC67)

Register	Offset	R/W	Description	Reset Value
TK_THC67	TK_BA+0x50	R/W	Touch Key TK6/TK7 Threshold Control Register	0xFF00_FF00

31	30	29	28	27	26	25	24
HTH7							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HTH6							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**HTH7**	**High Threshold of TK7** High level for TK7 threshold control.
[23:16]	**Reserved**	Reserved.
[15:8]	**HTH6**	**High Threshold of TK6** High level for TK6 threshold control.
[7:0]	**Reserved**	Reserved.

Touch Key TK8/TK9 Threshold Control Register (TK_THC89)

Register	Offset	R/W	Description	Reset Value
TK_THC89	TK_BA+0x54	R/W	Touch Key TK8/TK9 Threshold Control Register	0xFF00_FF00

31	30	29	28	27	26	25	24
HTH9							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HTH8							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**HTH9**	**High Threshold of TK9** High level for TK9 threshold control.
[23:16]	**Reserved**	Reserved.
[15:8]	**HTH8**	**High Threshold of TK8** High level for TK8 threshold control.
[7:0]	**Reserved**	Reserved.

Touch Key TK10/TK11 Threshold Control Register (TK_THC1011)

Register	Offset	R/W	Description	Reset Value
TK_THC1011	TK_BA+0x58	R/W	Touch Key TK10/TK11 Threshold Control Register	0xFF00_FF00

31	30	29	28	27	26	25	24
HTH11							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HTH10							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**HTH11**	**High Threshold of TK11** High level for TK11 threshold control.
[23:16]	**Reserved**	Reserved.
[15:8]	**HTH10**	**High Threshold of TK10** High level for TK10 threshold control.
[7:0]	**Reserved**	Reserved.

Touch Key TK12/TK13 Threshold Control Register (TK_THC1213)

Register	Offset	R/W	Description	Reset Value
TK_THC1213	TK_BA+0x5C	R/W	Touch Key TK12/TK13 Threshold Control Register	0xFF00_FF00

31	30	29	28	27	26	25	24
HTH13							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HTH12							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**HTH13**	**High Threshold of TK13** High level for TK13 threshold control.
[23:16]	**Reserved**	Reserved.
[15:8]	**HTH12**	**High Threshold of TK12** High level for TK12 threshold control.
[7:0]	**Reserved**	Reserved.

Touch Key TK14/TK15 Threshold Control Register (TK_THC1415)

Register	Offset	R/W	Description	Reset Value
TK_THC1415	TK_BA+0x60	R/W	Touch Key TK14/TK15 Threshold Control Register	0xFF00_FF00

31	30	29	28	27	26	25	24
HTH15							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HTH14							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**HTH15**	**High Threshold of TK15** High level for TK15 threshold control.
[23:16]	**Reserved**	Reserved.
[15:8]	**HTH14**	**High Threshold of TK14** High level for TK14 threshold control.
[7:0]	**Reserved**	Reserved.

Touch Key TK16Threshold Control Register (TK_THC16)

Register	Offset	R/W	Description	Reset Value
TK_THC16	TK_BA+0x64	R/W	Touch Key TK16 Threshold Control Register	0xFF00_FF00

31	30	29	28	27	26	25	24
HTH_ALL							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HTH16							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**HTH_ALL**	**High Threshold of ALL Keys Scan** High level for All Keys Scan threshold control.
[23:16]	**Reserved**	Reserved.
[15:8]	**HTH16**	**High Threshold of TK16** High level for TK16 threshold control.
[7:0]	**Reserved**	Reserved.

Touch Key Reference Capacitor Bank Data Register 0 (TK_REFCBD0)

Register	Offset	R/W	Description	Reset Value
TK_REFCBD0	TK_BA+0x68	R/W	Touch Key Reference Capacitor Bank Data Register 0	0x0000_0000

31	30	29	28	27	26	25	24
CBD3							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CBD2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CBD1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CBD0							

Bits	Description	
[31:24]	**CBD3**	**TK3 Capacitor Bank Data** These bits are used for TK3 sensitivity adjustment.
[23:16]	**CBD2**	**TK2 Capacitor Bank Data** These bits are used for TK2 sensitivity adjustment.
[15:8]	**CBD1**	**TK1 Capacitor Bank Data** These bits are used for TK1 sensitivity adjustment.
[7:0]	**CBD0**	**TK0 Capacitor Bank Data** These bits are used for TK0 sensitivity adjustment.

Touch Key Reference Capacitor Bank Data Register 1 (TK_REFCBD1)

Register	Offset	R/W	Description	Reset Value
TK_REFCBD1	TK_BA+0x6C	R/W	Touch Key Reference Capacitor Bank Data Register 1	0x0000_0000

31	30	29	28	27	26	25	24
CBD7							
23	22	21	20	19	18	17	16
CBD6							
15	14	13	12	11	10	9	8
CBD5							
7	6	5	4	3	2	1	0
CBD4							

Bits	Description	
[31:24]	CBD7	**TK7 Capacitor Bank Data** These bits are used for TK7 sensitivity adjustment.
[23:16]	CBD6	**TK6 Capacitor Bank Data** These bits are used for TK6 sensitivity adjustment.
[15:8]	CBD5	**TK5 Capacitor Bank Data** These bits are used for TK5 sensitivity adjustment.
[7:0]	CBD4	**TK4 Capacitor Bank Data** These bits are used for TK4 sensitivity adjustment.

Touch Key Reference Capacitor Bank Data Register 2 (TK_REFCBD2)

Register	Offset	R/W	Description	Reset Value
TK_REFCBD2	TK_BA+0x70	R/W	Touch Key Reference Capacitor Bank Data Register 2	0x0000_0000

31	30	29	28	27	26	25	24
CBD11							
23	22	21	20	19	18	17	16
CBD10							
15	14	13	12	11	10	9	8
CBD9							
7	6	5	4	3	2	1	0
CBD8							

Bits	Description	
[31:24]	**CBD11**	**TK11 Capacitor Bank Data** These bits are used for TK11 sensitivity adjustment.
[23:16]	**CBD10**	**TK10 Capacitor Bank Data** These bits are used for TK10 sensitivity adjustment.
[15:8]	**CBD9**	**TK9 Capacitor Bank Data** These bits are used for TK9 sensitivity adjustment.
[7:0]	**CBD8**	**TK8 Capacitor Bank Data** These bits are used for TK8 sensitivity adjustment.

Touch Key Reference Capacitor Bank Data Register 3 (TK_REFCBD3)

Register	Offset	R/W	Description	Reset Value
TK_REFCBD3	TK_BA+0x74	R/W	Touch Key Reference Capacitor Bank Data Register 3	0x0000_0000

31	30	29	28	27	26	25	24
CBD15							
23	22	21	20	19	18	17	16
CBD14							
15	14	13	12	11	10	9	8
CBD13							
7	6	5	4	3	2	1	0
CBD12							

Bits	Description	
[31:24]	**CBD15**	**TK15 Capacitor Bank Data** These bits are used for TK15 sensitivity adjustment.
[23:16]	**CBD14**	**TK14 Capacitor Bank Data** These bits are used for TK14 sensitivity adjustment.
[15:8]	**CBD13**	**TK13 Capacitor Bank Data** These bits are used for TK13 sensitivity adjustment.
[7:0]	**CBD12**	**TK12 Capacitor Bank Data** These bits are used for TK12 sensitivity adjustment.

Touch Key Reference Capacitor Bank Data Register 4 (TK_REFCBD4)

Register	Offset	R/W	Description	Reset Value
TK_REFCBD4	TK_BA+0x78	R/W	Touch Key Reference Capacitor Bank Data Register 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CBD_ALL							
7	6	5	4	3	2	1	0
CBD16							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**Reserved**	Reserved.
[15:8]	**CBD_ALL**	**All Keys Scan Capacitor Bank Data** These bits are used for All Keys Scan sensitivity adjustment.
[7:0]	**CBD16**	**TK16 Capacitor Bank Data** These bits are used for TK16 sensitivity adjustment.

Touch Key Scan Control Register 1 (TK_SCANC1)

Register	Offset	R/W	Description	Reset Value
TK_SCANC1	TK_BA+0x80	R/W	Touch Key Scan Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**TK17SEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**TK17SEN**	**TK17 Scan Enable Bit** This bit is ignored if TK17REN (TK_REFC[17]) is “1”. 0 = TKDAT17 (TK_DAT5[7:0]) is invalid. 1 = TK17 is always enabled for key scan. TKDAT17 (TK_DAT5[7:0]) is valid.

Touch Key Reference Control Register 1 (TK_REFC1)

Register	Offset	R/W	Description	Reset Value
TK_REFC1	TK_BA+0x84	R/W	Touch Key Reference Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**TK17REN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**TK17REN**	**TK17 Reference Enable Bit** 0 = TK17 is not reference. 1 = TK17 is set as reference, and TKDAT17 (TK_DAT5[7:0]) is invalid.

Touch Key Complement Capacitor Bank Data Register 5 (TK_CCBD5)

Register	Offset	R/W	Description	Reset Value
TK_CCBD5	TK_BA+0x88	R/W	Touch Key Complement Capacitor Bank Data Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCBD17							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**CCBD17**	**TK17 Complement CB Data** These bits are used for TK17 sensitivity adjustment.

Touch Key Idle State Control Register 1 (TK_IDLSC1)

Register	Offset	R/W	Description	Reset Value
TK_IDLSC1	TK_BA+0x9C	R/W	Touch Key Idle State Control Register 1	0x0000_0003

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						IDLS17	

Bits	Description	
[31:2]	Reserved	Reserved.
[1:0]	IDLS17	**TK17 Idle State Control** These bits are ignored if both TK17SEN (TK_SCANC1[0]) and POLEN17 (TK_POLC1[0]) are “0” or TK17REN (TK_REFC1[0]) is “1”. 00 = TK17 connected to Gnd. 01 = TK17 connected to AVDDH. 10 = TK17 connected to V_{DD}. 11 = TK17 connected to V_{DD}.

Touch Key Polarity Select Register 1 (TK_POLSEL1)

Register	Offset	R/W	Description	Reset Value
TK_POLSEL1	TK_BA+0xA0	R/W	Touch Key Polarity Select Register 1	0x0000_0003

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**POL17**	

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1:0]	**POL17**	**TK17 Polarity Select** These bits are ignored if POLEN17 (TK_POLC1[0]) is “0” or TK17REN (TK_REFC1[0]) is “1”. 00 = TK17 connected to Gnd. 01 = TK17 connected to AVDDH. 10 = TK17 connected to V_{DD}. 11 = TK17 connected to V_{DD}.

Touch Key Polarity Control Register 1 (TK_POLC1)

Register	Offset	R/W	Description	Reset Value
TK_POLC1	TK_BA+0xA4	R/W	Touch Key Polarity Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**POLEN17**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**POLEN17**	**TK17 Polarity Function Enable Bit** 0 = TK17 polarity function Disabled. 1 = TK17 polarity function Enabled.

Touch Key Status Register 1 (TK_STA1)

Register	Offset	R/W	Description	Reset Value
TK_STA1	TK_BA+0xA8	R/W	Touch Key Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**TKIF17**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**TKIF17**	**TK17 Interrupt Flag** This bit will be cleared by writing a “1” to this bit. 0 = No threshold control event with TK17. 1 = Threshold control event occurs with TK17.

Touch Key Data Register 5 (TK_DAT5)

Register	Offset	R/W	Description	Reset Value
TK_DAT5	TK_BA+0xAC	R	Touch Key Data Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TKDAT17							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**TKDAT17**	**TK17 Sensing Result Data (Read Only)** This data is invalid if TK17SEN (TK_SCANC1[0]) is “0” or TK17REN (TK_REFC1[0]) is “1”.

Touch Key TK17 Threshold Control Register (TK_THC17)

Register	Offset	R/W	Description	Reset Value
TK_THC17	TK_BA+0xC4	R/W	Touch Key TK17 Threshold Control Register	0x0000_FF00

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HTH17							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**HTH17**	**High Threshold of TK17** High level for TK17 threshold control.
[7:0]	**Reserved**	Reserved.

Touch Key Reference Capacitor Bank Data Register 5 (TK_REFCBD5)

Register	Offset	R/W	Description	Reset Value
TK_REFCBD5	TK_BA+0xE8	R/W	Touch Key Reference Capacitor Bank Data Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CBD17							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**CBD17**	**TK17 Capacitor Bank Data** These bits are used for TK17 sensitivity adjustment.

Touch Key Extend Capacitor Bank Control Register (TK_EXTCBC)

Register	Offset	R/W	Description	Reset Value
TK_EXTCBC	TK_BA+0x100	R/W	Touch Key Extend Capacitor Bank Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						EXT_CBSEL	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[1:0]	**EXT_CBSEL**	**TK Extend Capacitor Bank Select** TK capacitor bank data and reference capacitor bank data are 9 bits. These bits are used to select how the 9 bits are generated. 00 = CCBDx (TK_CCBDx[8n+7:8n]) and CBDx (TK_REFCBDx[8n+7:8n]) are valid, but EXT_CCBDx (TK_EXTCCBDx[0]) and EXT_CBDx (TK_REFCBDx[0]) are invalid. The capacitor touch key sensing controller adds 1'b0 to the MSB of capacitor bank data and reference capacitor bank data, and the controller finally outputs 9 bits such as {1'b0, CCBDx} or {1'b0, CBDx}. 01 = CCBDx (TK_CCBDx[8n+7:8n]) and CBDx (TK_REFCBDx[8n+7:8n]) are valid, but EXT_CCBDx (TK_EXTCCBDx[0]) and EXT_CBDx (TK_REFCBDx[0]) are invalid. The capacitor touch key sensing controller shifts the capacitor bank data and reference capacitor bank data one bit to the left, and the controller finally outputs 9 bits such as {CCBDx, 1'b0} or {CBDx, 1'b0}. 10 = CCBDx (TK_CCBDx[8n+7:8n]), CBDx (TK_REFCBDx[8n+7:8n]), EXT_CCBDx (TK_EXTCCBDx[0]) and EXT_CBDx (TK_REFCBDx[0]) are valid. The capacitor touch key sensing controller outputs 9 bits such as {EXT_CCBDx, CCBDx} or {EXT_CBDx, CBDx}. 11 = Reserved. **Note 1:** The above parameter x is 0 to 17. **Note 2:** The above parameter n is 0 to 3.

Touch Key Extend Complement Capacitor Bank Data Register 0 (TK_EXTCCBD0)

Register	Offset	R/W	Description	Reset Value
TK_EXTCCBD0	TK_BA+0x104	R/W	Touch Key Extend Complement Capacitor Bank Data Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							EXT_CCBD3
23	22	21	20	19	18	17	16
Reserved							EXT_CCBD2
15	14	13	12	11	10	9	8
Reserved							EXT_CCBD1
7	6	5	4	3	2	1	0
Reserved							EXT_CCBD0

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**CCBD3**	**TK3 Extend Complement CB Data** These bits are used for TK3 sensitivity adjustment.
[23:17]	**Reserved**	Reserved.
[16]	**CCBD2**	**TK2 Extend Complement CB Data** These bits are used for TK2 sensitivity adjustment.
[15:9]	**Reserved**	Reserved.
[8]	**CCBD1**	**TK1 Extend Complement CB Data** These bits are used for TK1 sensitivity adjustment.
[7:1]	**Reserved**	Reserved.
[0]	**CCBD0**	**TK0 Extend Complement CB Data** These bits are used for TK0 sensitivity adjustment.

Touch Key Extend Complement Capacitor Bank Data Register 1 (TK_EXTCCBD1)

Register	Offset	R/W	Description	Reset Value
TK_EXTCCBD1	TK_BA+0x108	R/W	Touch Key Extend Complement Capacitor Bank Data Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							EXT_CCBD7
23	22	21	20	19	18	17	16
Reserved							EXT_CCBD6
15	14	13	12	11	10	9	8
Reserved							EXT_CCBD5
7	6	5	4	3	2	1	0
Reserved							EXT_CCBD4

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**CCBD7**	**TK7 Extend Complement CB Data** These bits are used for TK7 sensitivity adjustment.
[23:17]	**Reserved**	Reserved.
[16]	**CCBD6**	**TK6 Extend Complement CB Data** These bits are used for TK6 sensitivity adjustment.
[15:9]	**Reserved**	Reserved.
[8]	**CCBD5**	**TK5 Extend Complement CB Data** These bits are used for TK5 sensitivity adjustment.
[7:1]	**Reserved**	Reserved.
[0]	**CCBD4**	**TK4 Extend Complement CB Data** These bits are used for TK4 sensitivity adjustment.

Touch Key Extend Complement Capacitor Bank Data Register 2 (TK_EXTCCBD2)

Register	Offset	R/W	Description	Reset Value
TK_EXTCCBD2	TK_BA+0x10C	R/W	Touch Key Extend Complement Capacitor Bank Data Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							EXT_CCBD11
23	22	21	20	19	18	17	16
Reserved							EXT_CCBD10
15	14	13	12	11	10	9	8
Reserved							EXT_CCBD9
7	6	5	4	3	2	1	0
Reserved							EXT_CCBD8

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**CCBD11**	**TK11 Extend Complement CB Data** These bits are used for TK11 sensitivity adjustment.
[23:17]	**Reserved**	Reserved.
[16]	**CCBD10**	**TK10 Extend Complement CB Data** These bits are used for TK10 sensitivity adjustment.
[15:9]	**Reserved**	Reserved.
[8]	**CCBD9**	**TK9 Extend Complement CB Data** These bits are used for TK9 sensitivity adjustment.
[7:1]	**Reserved**	Reserved.
[0]	**CCBD8**	**TK8 Extend Complement CB Data** These bits are used for TK8 sensitivity adjustment.

Touch Key Extend Complement Capacitor Bank Data Register 3 (TK_EXTCCBD3)

Register	Offset	R/W	Description	Reset Value
TK_EXTCCBD3	TK_BA+0x110	R/W	Touch Key Extend Complement Capacitor Bank Data Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							EXT_CCBD15
23	22	21	20	19	18	17	16
Reserved							EXT_CCBD14
15	14	13	12	11	10	9	8
Reserved							EXT_CCBD13
7	6	5	4	3	2	1	0
Reserved							EXT_CCBD12

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**CCBD15**	**TK15 Extend Complement CB Data** These bits are used for TK15 sensitivity adjustment.
[23:17]	**Reserved**	Reserved.
[16]	**CCBD14**	**TK14 Extend Complement CB Data** These bits are used for TK14 sensitivity adjustment.
[15:9]	**Reserved**	Reserved.
[8]	**CCBD13**	**TK13 Extend Complement CB Data** These bits are used for TK13 sensitivity adjustment.
[7:1]	**Reserved**	Reserved.
[0]	**CCBD12**	**TK12 Extend Complement CB Data** These bits are used for TK12 sensitivity adjustment.

Touch Key Extend Complement Capacitor Bank Data Register 4 (TK_EXTCCBD4)

Register	Offset	R/W	Description	Reset Value
TK_EXTCCBD4	TK_BA+0x114	R/W	Touch Key Extend Complement Capacitor Bank Data Register 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							EXT_CCBDALL
7	6	5	4	3	2	1	0
Reserved							EXT_CCBD16

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CCBDALL**	**All Keys Extend Complement CB Data** These bits are used for All Key Scan sensitivity adjustment.
[7:1]	**Reserved**	Reserved.
[0]	**CCBD16**	**TK16 Extend Complement CB Data** These bits are used for TK16 sensitivity adjustment.

Touch Key Extend Complement Capacitor Bank Data Register 5 (TK_EXTCCBD5)

Register	Offset	R/W	Description	Reset Value
TK_EXTCCBD5	TK_BA+0x118	R/W	Touch Key Extend Complement Capacitor Bank Data Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							EXT_CCBD17

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CCBD17**	**TK17 Extend Complement CB Data** These bits are used for TK17 sensitivity adjustment.

Touch Key Extend Reference Capacitor Bank Data Register 0 (TK_EXTREFCBD0)

Register	Offset	R/W	Description	Reset Value
TK_EXTREFCBD0	TK_BA+0x130	R/W	Touch Key Extend Reference Capacitor Bank Data Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							EXT_CBD3
23	22	21	20	19	18	17	16
Reserved							EXT_CBD2
15	14	13	12	11	10	9	8
Reserved							EXT_CBD1
7	6	5	4	3	2	1	0
Reserved							EXT_CBD0

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**EXT_CBD3**	**TK3 Extend Capacitor Bank Data** These bits are used for TK3 sensitivity adjustment.
[23:17]	**Reserved**	Reserved.
[16]	**EXT_CBD2**	**TK2 Extend Capacitor Bank Data** These bits are used for TK2 sensitivity adjustment.
[15:9]	**Reserved**	Reserved.
[8]	**EXT_CBD1**	**TK1 Extend Capacitor Bank Data** These bits are used for TK1 sensitivity adjustment.
[7:1]	**Reserved**	Reserved.
[0]	**EXT_CBD0**	**TK0 Extend Capacitor Bank Data** These bits are used for TK0 sensitivity adjustment.

Touch Key Extend Reference Capacitor Bank Data Register 1 (TK_EXTREFCBD1)

Register	Offset	R/W	Description	Reset Value
TK_EXTREFCBD1	TK_BA+0x134	R/W	Touch Key Extend Reference Capacitor Bank Data Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							EXT_CBD7
23	22	21	20	19	18	17	16
Reserved							EXT_CBD6
15	14	13	12	11	10	9	8
Reserved							EXT_CBD5
7	6	5	4	3	2	1	0
Reserved							EXT_CBD4

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**EXT_CBD7**	**TK7 Extend Capacitor Bank Data** These bits are used for TK7 sensitivity adjustment.
[23:17]	**Reserved**	Reserved.
[16]	**EXT_CBD6**	**TK6 Extend Capacitor Bank Data** These bits are used for TK6 sensitivity adjustment.
[15:9]	**Reserved**	Reserved.
[8]	**EXT_CBD5**	**TK5 Extend Capacitor Bank Data** These bits are used for TK5 sensitivity adjustment.
[7:1]	**Reserved**	Reserved.
[0]	**EXT_CBD4**	**TK4 Extend Capacitor Bank Data** These bits are used for TK4 sensitivity adjustment.

Touch Key Extend Reference Capacitor Bank Data Register 2 (TK_EXTREFCBD2)

Register	Offset	R/W	Description	Reset Value
TK_EXTREFCBD2	TK_BA+0x138	R/W	Touch Key Extend Reference Capacitor Bank Data Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							EXT_CBD11
23	22	21	20	19	18	17	16
Reserved							EXT_CBD10
15	14	13	12	11	10	9	8
Reserved							EXT_CBD9
7	6	5	4	3	2	1	0
Reserved							EXT_CBD8

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**EXT_CBD11**	**TK11 Extend Capacitor Bank Data** These bits are used for TK11 sensitivity adjustment.
[23:17]	**Reserved**	Reserved.
[16]	**EXT_CBD10**	**TK10 Extend Capacitor Bank Data** These bits are used for TK10 sensitivity adjustment.
[15:9]	**Reserved**	Reserved.
[8]	**EXT_CBD9**	**TK9 Extend Capacitor Bank Data** These bits are used for TK9 sensitivity adjustment.
[7:1]	**Reserved**	Reserved.
[0]	**EXT_CBD8**	**TK8 Extend Capacitor Bank Data** These bits are used for TK8 sensitivity adjustment.

Touch Key Extend Reference Capacitor Bank Data Register 3 (TK_EXTREFCBD3)

Register	Offset	R/W	Description	Reset Value
TK_EXTREFCBD3	TK_BA+0x13C	R/W	Touch Key Extend Reference Capacitor Bank Data Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							EXT_CBD15
23	22	21	20	19	18	17	16
Reserved							EXT_CBD14
15	14	13	12	11	10	9	8
Reserved							EXT_CBD13
7	6	5	4	3	2	1	0
Reserved							EXT_CBD12

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**EXT_CBD15**	**TK15 Extend Capacitor Bank Data** These bits are used for TK15 sensitivity adjustment.
[23:17]	**Reserved**	Reserved.
[16]	**EXT_CBD14**	**TK14 Extend Capacitor Bank Data** These bits are used for TK14 sensitivity adjustment.
[15:9]	**Reserved**	Reserved.
[8]	**EXT_CBD13**	**TK13 Extend Capacitor Bank Data** These bits are used for TK13 sensitivity adjustment.
[7:1]	**Reserved**	Reserved.
[0]	**EXT_CBD12**	**TK12 Extend Capacitor Bank Data** These bits are used for TK12 sensitivity adjustment.

Touch Key Extend Reference Capacitor Bank Data Register 4 (TK_EXTREFCBD4)

Register	Offset	R/W	Description	Reset Value
TK_EXTREFCBD4	TK_BA+0x140	R/W	Touch Key Extend Reference Capacitor Bank Data Register 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							EXT_CBDALL
7	6	5	4	3	2	1	0
Reserved							EXT_CBD16

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**EXT_CBDALL**	**All Keys Extend Capacitor Bank Data** These bits are used for All Keys Scan sensitivity adjustment.
[7:1]	**Reserved**	Reserved.
[0]	**EXT_CBD16**	**TK16 Extend Capacitor Bank Data** These bits are used for TK16 sensitivity adjustment.

Touch Key Extend Reference Capacitor Bank Data Register 5 (TK_EXTREFCBD5)

Register	Offset	R/W	Description	Reset Value
TK_EXTREFCBD5	TK_BA+0x144	R/W	Touch Key Extend Reference Capacitor Bank Data Register 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							EXT_CBD17

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**EXT_CBD17**	**TK17 Extend Capacitor Bank Data** These bits are used for TK17 sensitivity adjustment.

6.40 CRC Controller (CRC)

6.40.1 Overview

The Cyclic Redundancy Check (CRC) generator can perform CRC calculation with three common polynomials CRC-8, CRC-16, CRC-32, and 32-bit programmable polynomials settings.

6.40.2 Features

- Supports three common polynomials CRC-8, CRC-16, and CRC-32
- Programmable seed value
- Supports programmable order reverse setting for input data and CRC checksum
- Supports programmable 1's complement setting for input data and CRC checksum
- Supports 8/16/32-bit of data width
  - 8-bit write mode: 1-AHB clock cycle operation
  - 16-bit write mode: 2-AHB clock cycle operation
  - 32-bit write mode: 4-AHB clock cycle operation
- Supports Master mode

6.40.3 Block Diagram

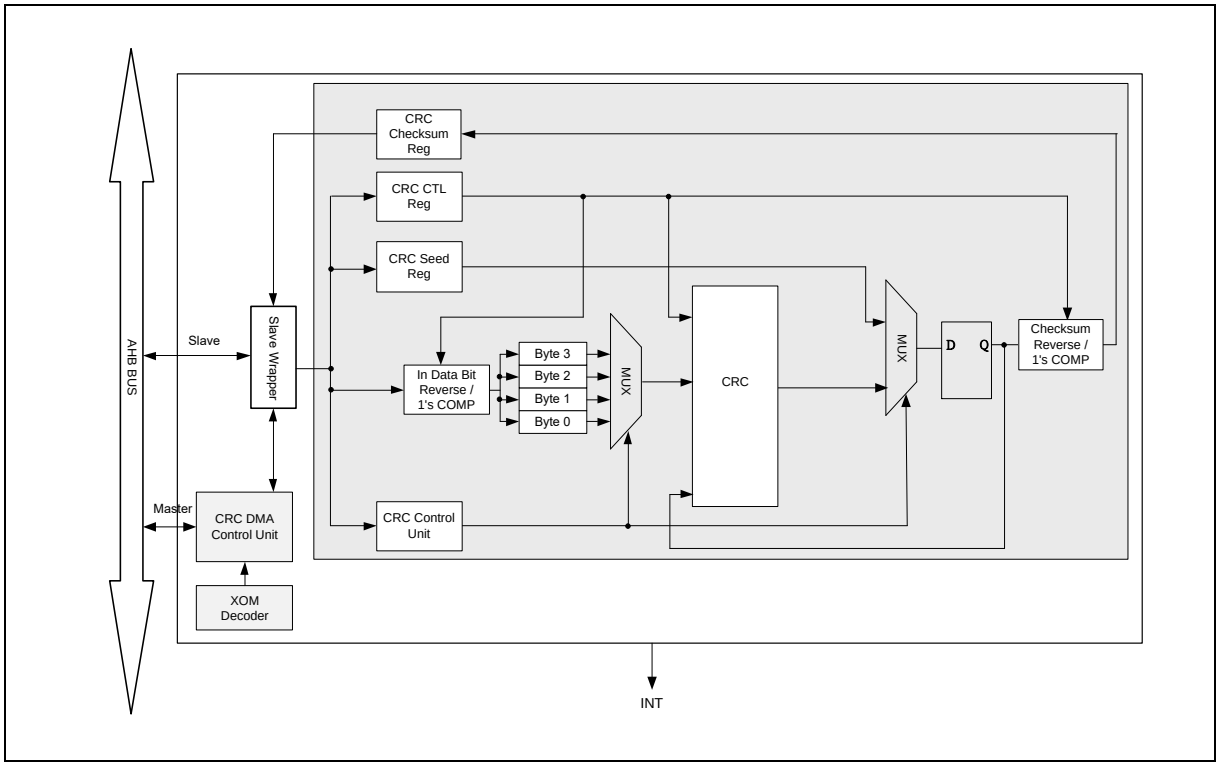


Figure 6.40-1 CRC Generator Block Diagram

6.40.4 Basic Configuration

- Clock Source Configuration
  - Enable CRC peripheral clock in CRCCKEN (CLK_AHBCLK[7]).

- Reset Configuration
  - Reset CRC controller in CRCRST (SYS_IPRST0[7]).

6.40.5 Functional Description

The CRC generator can perform CRC calculation with 8-bit, 16-bit and 32-bit programmable polynomial settings. User can choose the length of CRC polynomial by setting CRCMODE[1:0] (CRC_CTL[31:30] CRC Polynomial Mode) and program the value of CRC polynomial by setting POLYNOMIAL[31:0] (CRC_POLYNOMIAL[31:0] CRC Polynomial value). For example, you must write 0x07 to register CRC_POLYNOMIA when the polynomial is ($X^8 + X^2 + X + 1$). If the polynomial is ($X^{16} + X^{15} + X^2 + 1$), the value of register CRC_POLYNOMIA must be 0x8005. You must write 0x04C11DB7 to register CRC_POLYNOMIA when the polynomial is ($X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$).

The following is a programming sequence example:

1. Enable CRC generator by setting CRCEN (CRC_CTL[0] CRC Channel Enable Bit).
2. Initial setting for CRC calculation.
   - Configure 1's complement for CRC checksum by setting CHKSFMT (CRC_CTL[27] Checksum 1's Complement).
   - Configure bit order reverse for CRC checksum by setting CHKSREV (CRC_CTL[25] Checksum Bit Order Reverse). The functional block is also shown in Figure 6.40-2. Configure 1's complement for CRC write data by setting DATFMT (CRC_CTL[26] Write Data 1's Complement).
   - Configure bit order reverse for CRC write data per byte by setting DATREV (CRC_CTL[24] Write Data Bit Order Reverse). The functional block is also shown in Figure 6.40-3.
3. Perform CHKSINIT (CRC_CTL[1] Checksum Initialization) to load the initial checksum value from CRC_SEED register value.
4. Write data to CRC_DAT register to calculate CRC checksum.
5. Get the CRC checksum result by reading CRC_CHECKSUM register.

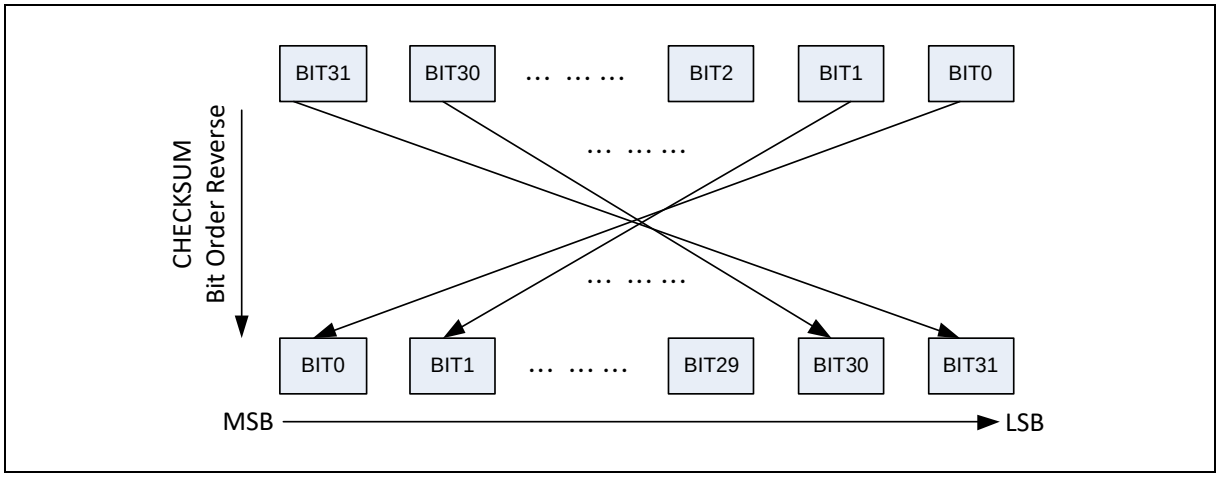


Figure 6.40-2 CHECKSUM Bit Order Reverse Functional Block

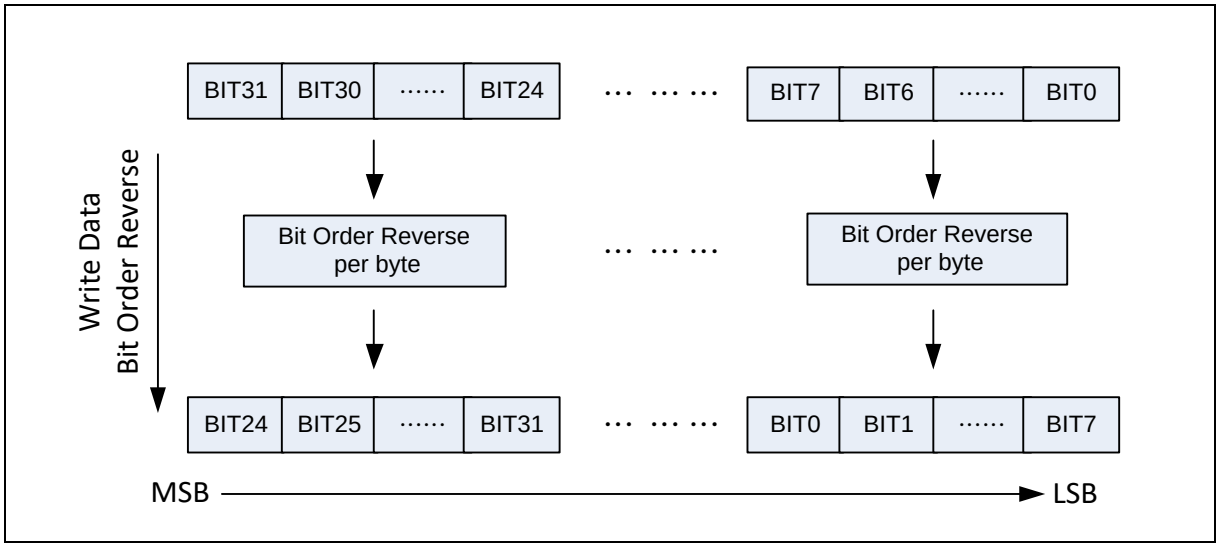


Figure 6.40-3 Write Data Bit Order Reverse Functional Block

6.40.5.2 Master mode

Equipped with a DMA, the CRC controller has a Master mode and is able to access data actively.

The CRC master mode uses a state machine to control the flow:

- Idle state
- Run state

When CRC DMA is not operating, the CRC DMA is at idle state, at which user can set CRC setting through registers.

When CRC DMA is operating, the CRC DMA is at run state, at which user cannot set CRC setting until the finish of DMA operation.

When the CRC DMA is operating, indicated by the START (CRC_DMACTL[0]) register, the CRC_CTL, CRC_SEED, CRC_POLYNOMIAL, CRC_SADDR and CRC_DMACNT do not accept any write access until the operation is finished. User can only control the CRC DMA through ABORT (CRC_DMACTL[4]), PAUSE (CRC_DMACTL[1]) bits.

Besides, CRC_DAT register becomes inaccessible (read-as-all-zero and write ignore, RAZWI) and CRC_CHECKSUM register is unreadable (RAZ) until the operation is finished.

The sample flow of using CRC master mode is as follows:

1. Configure the CRC mode, such as Polynomial mode, Data Length, 1's Complements, and Checksum Bit Order Reverse.
2. Configure the region for DMA operating through CRC_SADDR, CRC_DMACNT registers.
3. Set INTEN (CRC_DMACTL[8]) register to enable CRC interrupt, if required.
4. Initialize CRC by setting CHKSINIT (CRC_DMACTL[0]).
5. Set START (CRC_DMACTL[0]) register to trigger the CRC Master operation.
6. Check if CRC is still operating through reading CRC_DMASTS or reading the START bit in CRC_DMACTL or wait for CRC interrupt.
7. After the CRC Master operation finished, the START bit in CRC_DMACTL register will be cleared. The FINISH bit in CRC_DMASTS register will be set if the operation finished successfully.

The base address for CRC DMA to check is word-aligned and the granularity of size for CRC DMA access is 1 word (i.e. 4 bytes).

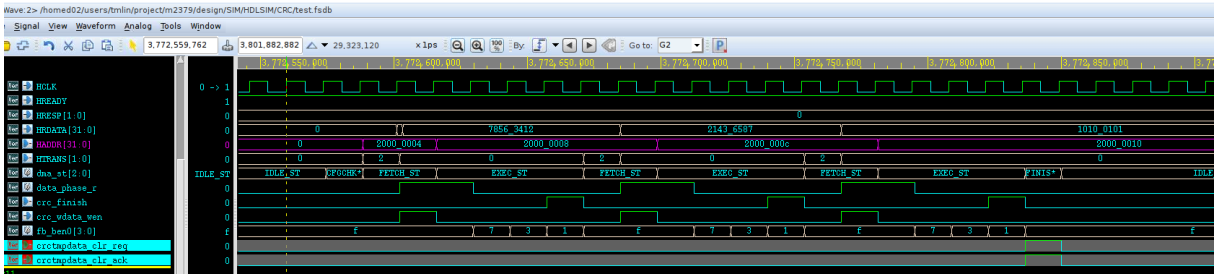
CRC will assert the interrupt to CPU when CRC_DMACTL.INTEN is set and one of CRC_DMASTS.CFGERR, CRC_DMASTS.ABORTED, CRC_DMASTS.FINISH flag is set. Besides, if any of the flag is not cleared to zero, the CRC_DMA cannot be set to run.

Condition of setting each flag:

- CFGERR Flag:
  - For Non Flash XOM mode:
    - CRCDMA Count = 0
    - Region for CRCDMA access cross 256 Mbytes boundary
  - For Flash XOM mode:
    - DMA Count = 0
    - Region for CRCDMA access cross 256 Mbytes boundary
    - Limitation of CRC on Flash XOM (listed below)
- ACCERR.Flag: When accessing CRC DMA gets bus error response
- ABORTED Flag: When operation of CRC DMA is finished incorrectly
- FINISH Flag: When operation of CRC DMA is finished correctly

Note that CRC Master mode is able to calculate CRC of Flash XOM region with some limitations:

- The Flash XOM region used to calculate CRC cannot cross the region boundary.
- The polynomial should not be zero.
- The minimum size of the Flash XOM region is 16 words, and the address should be aligned to 16 words. That is, the SADDR[3:0] bits in CRC_SADDR register and the DMACNT[3:0] bits in CRC_DMACNT register are reserved.



6.40.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
CRC Base Address: **CRC_BA = 0x4020_B000** **CRC non-secure base address is CRC_BA+0x1000_0000**				
CRC_CTL	CRC_BA+0x00	R/W	CRC Control Register	0x6000_0000
CRC_DAT	CRC_BA+0x04	R/W	CRC Write Data Register	0x0000_0000
CRC_SEED	CRC_BA+0x08	R/W	CRC Seed Register	0xFFFF_FFFF
CRC_CHECKSUM	CRC_BA+0x0C	R	CRC Checksum Register	0xFFFF_FFFF
CRC_POLYNOMIAL	CRC_BA+0x10	R/W	CRC Polynomial Register	0x0000_0000
CRC_DMACTL	CRC_BA+0x40	R/W	CRC DMA Mode Control Register	0x0000_0000
CRC_DMASTS	CRC_BA+0x44	R/W	CRC DMA Mode Status Register	0x0000_0000
CRC_SADDR	CRC_BA+0x48	R/W	CRC DMA Source Address Register	0x0000_0000
CRC_DMACNT	CRC_BA+0x4C	R/W	CRC DMA Word Count Register	0x0000_0000

6.40.7 Register Description

CRC Control Register (CRC_CTL)

Register	Offset	R/W	Description	Reset Value
CRC_CTL	CRC_BA+0x00	R/W	CRC Control Register	0x6000_0000

31	30	29	28	27	26	25	24
CRCMODE		DATLEN		CHKSFMT	DATFMT	CHKSREV	DATREV
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CHKSINIT	CRCEN

Bits	Description	
[31:30]	**CRCMODE**	**CRC Polynomial Mode** This field indicates the CRC operation polynomial mode. 01 = CRC-8 Polynomial mode. x0 = CRC-16 Polynomial mode. 11 = CRC-32 Polynomial mode.
[29:28]	**DATLEN**	**CPU Write Data Length** This field indicates the write data length. 00 = Data length is 8-bit mode. 01 = Data length is 16-bit mode. 1x = Data length is 32-bit mode. **Note:** When the write data length is 8-bit mode, the valid data in CRC_DAT register is only DATA[7:0] bits; if the write data length is 16-bit mode, the valid data in CRC_DAT register is only DATA[15:0].
[27]	**CHKSFMT**	**Checksum 1's Complement** This bit is used to enable the 1's complement function for checksum result in CRC_CHECKSUM register. 0 = 1's complement for CRC checksum Disabled. 1 = 1's complement for CRC checksum Enabled.
[26]	**DATFMT**	**Write Data 1's Complement** This bit is used to enable the 1's complement function for write data value in CRC_DAT register. 0 = 1's complement for CRC writes data in Disabled. 1 = 1's complement for CRC writes data in Enabled.
[25]	**CHKSREV**	**Checksum Bit Order Reverse** This bit is used to enable the bit order reverse function for checksum result in CRC_CHECKSUM register. 0 = Bit order reverse for CRC checksum Disabled. 1 = Bit order reverse for CRC checksum Enabled. **Note:** If the checksum result is 0xDD7B0F2E, the bit order reverse for CRC checksum is 0x74F0DEBB.

[24]	**DATREV**	**Write Data Bit Order Reverse** This bit is used to enable the bit order reverse function per byte for write data value in CRC_DAT register. 0 = Bit order reversed for CRC write data in Disabled. 1 = Bit order reversed for CRC write data in Enabled (per byte). **Note:** If the write data is 0xAABBCCDD, the bit order reverse for CRC write data in is 0x55DD33BB.
[23:2]	**Reserved**	Reserved.
[1]	**CHKSINIT**	**Checksum Initialization** 0 = No effect. 1 = Initial checksum value by auto reload CRC_SEED register value to CRC_CHECKSUM register value. **Note:** This bit will be cleared automatically.
[0]	**CRCEN**	**CRC Channel Enable Bit** 0 = No effect. 1 = CRC operation Enabled.

CRC Write Data Register (CRC_DAT)

Register	Offset	R/W	Description	Reset Value
CRC_DAT	CRC_BA+0x04	R/W	CRC Write Data Register	0x0000_0000

31	30	29	28	27	26	25	24
DATA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DATA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DATA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATA							

Bits	Description	
[31:0]	**DATA**	**CRC Write Data Bits** User can write data directly by CPU to write data to this field to perform CRC operation. **Note:** When the write data length is 8-bit mode, the valid data in CRC_DAT register is only DATA[7:0] bits; if the write data length is 16-bit mode, the valid data in CRC_DAT register is only DATA[15:0].

CRC Seed Register (CRC_SEED)

Register	Offset	R/W	Description	Reset Value
CRC_SEED	CRC_BA+0x08	R/W	CRC Seed Register	0xFFFF_FFFF

31	30	29	28	27	26	25	24
SEED							
23	22	21	20	19	18	17	16
SEED							
15	14	13	12	11	10	9	8
SEED							
7	6	5	4	3	2	1	0
SEED							

Bits	Description	
[31:0]	SEED	**CRC Seed Value** This field indicates the CRC seed value. **Note:** This field will be reloaded as checksum initial value (CRC_CHECKSUM register) after perform CHKSINIT (CRC_CTL[1]).

CRC Checksum Register (CRC_CHECKSUM)

Register	Offset	R/W	Description	Reset Value
CRC_CHECKSUM	CRC_BA+0x0C	R	CRC Checksum Register	0xFFFF_FFFF

31	30	29	28	27	26	25	24
CHECKSUM							
23	22	21	20	19	18	17	16
CHECKSUM							
15	14	13	12	11	10	9	8
CHECKSUM							
7	6	5	4	3	2	1	0
CHECKSUM							

Bits	Description	
[31:0]	CHECKSUM	**CRC Checksum Results** This field indicates the CRC checksum result. **Note:** Data in CRC_CHECKSUM register has different length when user chooses different operation polynomial modes. For example: If final checksum result is 0x12 in CRC-8 polynomial mode, the CHECKSUM[31:0] value will be read as 0x12121212, only CHECKSUM[7:0] is valid in this mode. If final checksum result is 0x1234 in CRC-CCITT or CRC-16 mode, the CHECKSUM[31:0] value will be read as 0x12341234, only CHECKSUM[15:0] is valid in this mode. And the CHECKSUM[31:0] is valid for CRC-32 mode.

CRC Polynomial Register (CRC_POLYNOMIAL)

Register	Offset	R/W	Description	Reset Value
CRC_POLYNOMIAL	CRC_BA+0x10	R/W	CRC Polynomial Register	0x0000_0000

31	30	29	28	27	26	25	24
POLYNOMIAL							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
POLYNOMIAL							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
POLYNOMIAL							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POLYNOMIAL							

Bits	Description	
[31:0]	**POLYNOMIAL**	**CRC Polynomial Value Results** This field indicates the value of CRC polynomial.

CRC DMA Mode Control Register (CRC_DMACTL)

Register	Offset	R/W	Description	Reset Value
CRC_DMACTL	CRC_BA+0x40	R/W	CRC DMA Mode Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							INTEN
7	6	5	4	3	2	1	0
Reserved			ABORT	Reserved		PAUSE	START

Bits		Description
[31:9]	**Reserved**	Reserved.
[8]	**INTEN**	**CRC DMA Mode Interrupt Enable Bit** Enable CRC DMA mode interrupt.
[7:5]	**Reserved**	Reserved.
[4]	**ABORT**	**CRC DMA Operation Abort Bit** Write 1 to abort the operation of CRC DMA.
[3:2]	**Reserved**	Reserved.
[1]	**PAUSE**	**CRC DMA Pause Bit** Bit to set CRC DMA to pause the operation. When write 0 = Continue the CRCDMA operation when CRCDMA is under operation 1 = Pause the CRCDMA operation. **Note:** This bit is read-as-1 when the CRC DMA is paused.
[0]	**START**	**CRC DMA Start Bit** Set CRC DMA to start operation. This bit is read-as-1 when the CRC DMA is operating. This bit can be set to 1 when all DMASTS flags are cleared to zero.

CRC DMA Mode Status Register (CRC_DMASTS)

Register	Offset	R/W	Description	Reset Value
CRC_DMASTS	CRC_BA+0x44	R/W	CRC DMA Mode Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				ACCERR	CFGERR	ABORTED	FINISH

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**ACCERR**	**DMA Access Error Flag (Write 1 to Clear)** Indicates the operation of DMA mode is aborted due to access error.
[2]	**CFGERR**	**Configuration Error Flag (Write 1 to Clear)** Indicates the configuration to run DMA operation is invalid and the operation of DMA mode is canceled.
[1]	**ABORTED**	**DMA Operation Aborted Flag (Write 1 to Clear)** Indicates the operation of DMA mode is aborted due to user setting CRC_DMACTL.ABORT bit.
[0]	**FINISH**	**DMA Operation Finish Flag (Write 1 to Clear)** Indicates the operation of DMA mode finished successfully.

CRC DMA Source Address Register (CRC_SADDR)

Register	Offset	R/W	Description	Reset Value
CRC_SADDR	CRC_BA+0x48	R/W	CRC DMA Source Address Register	0x0000_0000

31	30	29	28	27	26	25	24
SADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SADDR						**Reserved**	

Bits	Description	
[31:2]	**SADDR**	**CRC DMA Source Address** DMA source address This value should be less than 0xE000_0000
[1:0]	**Reserved**	Reserved.

CRC DMA Word Count Register (CRC_DMACNT)

Register	Offset	R/W	Description	Reset Value
CRC_DMACNT	CRC_BA+0x4C	R/W	CRC DMA Word Count Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**DMACNT**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DMACNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DMACNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMACNT						**Reserved**	

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:2]	**DMACNT**	**CRC DMA Operation Word Count** Indicates words for DMA to read.
[1:0]	**Reserved**	Reserved.

6.41 Analog Comparator Controller (ACMP)

6.41.1 Overview

The chip provides three comparators. The comparator output is logic 1 when positive input is greater than negative input; otherwise, the output is 0. Each comparator can be configured to generate an interrupt when the comparator output value changes.

6.41.2 Features

- Analog input voltage range: 0 ~ AV_{DD} (voltage of AV_{DD} pin)
- Up to three rail-to-rail analog comparators
- Supports hysteresis function
  - Supports programmable hysteresis window: 0mV,-, 20mV,-, 40mV,
- Supports wake-up function
- Selectable input sources of positive input and negative input
- ACMP0 supports:
  - 7 multiplexed I/O pins at positive sources:
    - ACMP0_P0, ACMP0_P1, ACMP0_P2, ACMP0_P3, OPA0_int_OUT, OPA1_int_OUT or OPA2_int_OUT
  - 5 negative sources:
    - ACMP0_N
    - Supports Comparator Reference Voltage (CRV0)
    - Internal band-gap voltage (V_{BG})
    - DAC0 output (DAC0_OUT)
    - DAC1 output (DAC1_OUT)
- ACMP1 supports:
  - 7 multiplexed I/O pins at positive sources:
    - ACMP1_P0, ACMP1_P1, ACMP1_P2, ACMP1_P3, OPA0_int_OUT, OPA1_int_OUT or OPA2_int_OUT
  - 5 negative sources:
    - ACMP1_N
    - Supports Comparator Reference Voltage (CRV1)
    - Internal band-gap voltage (V_{BG})
    - DAC0 output (DAC0_OUT)
    - DAC1 output (DAC1_OUT)
- ACMP2 supports:
  - 7 multiplexed I/O pins at positive sources:
    - ACMP2_P0, ACMP2_P1, ACMP2_P2, ACMP2_P3, OPA0_int_OUT, OPA1_int_OUT or OPA2_int_OUT
  - 5 negative sources:
    - ACMP2_N

- Supports Comparator Reference Voltage (CRV2)
- Internal band-gap voltage (V_{BG})
- DAC0 output (DAC0_OUT)
- DAC1 output (DAC1_OUT)

- ACMP0 and ACMP1 share one ACMP interrupt vector, and ACMP2 use one ACMP interrupt vector
- Interrupts generated when compare results change (Interrupt event condition is programmable)
- Supports triggers for break events and cycle-by-cycle control for EPWM
- Supports window compare mode and window latch mode
- Supports offset calibration
- Supports ACMP0 event, ACMP1 event, ACMP2 event be as PDMA request source
- Supports ACMP0 event, ACMP1 event, ACMP2 event be as EADC trigger source
- Supports ACMP0 output, ACMP1 output, ACMP2 output be as Timer, LPTimer, ECAP,LPADC, PWM and EPWM trigger source
- Supports ACMP0 interrupt flag, ACMP1 interrupt flag, ACMP2 interrupt flag be as UTCPD trigger source

6.41.3 Block Diagram

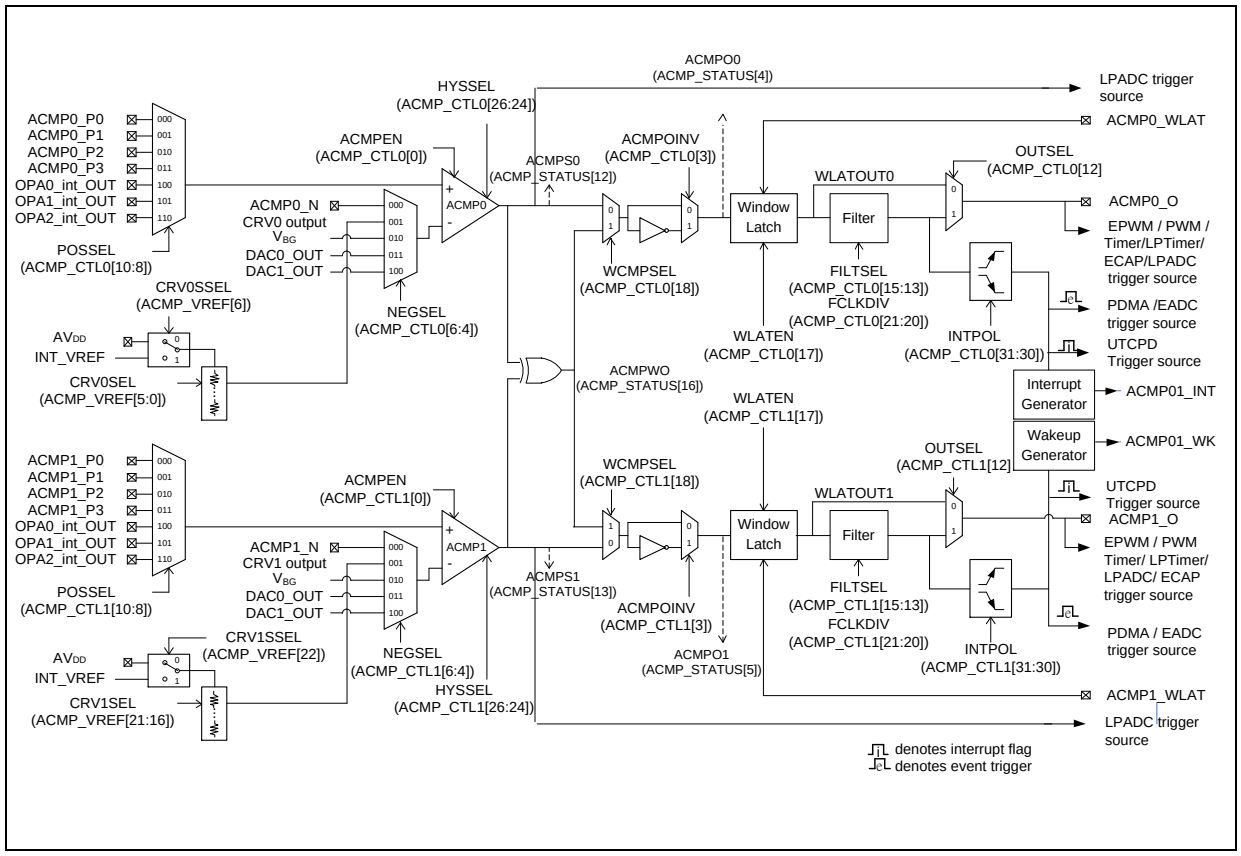


Figure 6.41-1 Analog Comparator 0/1 Block Diagram

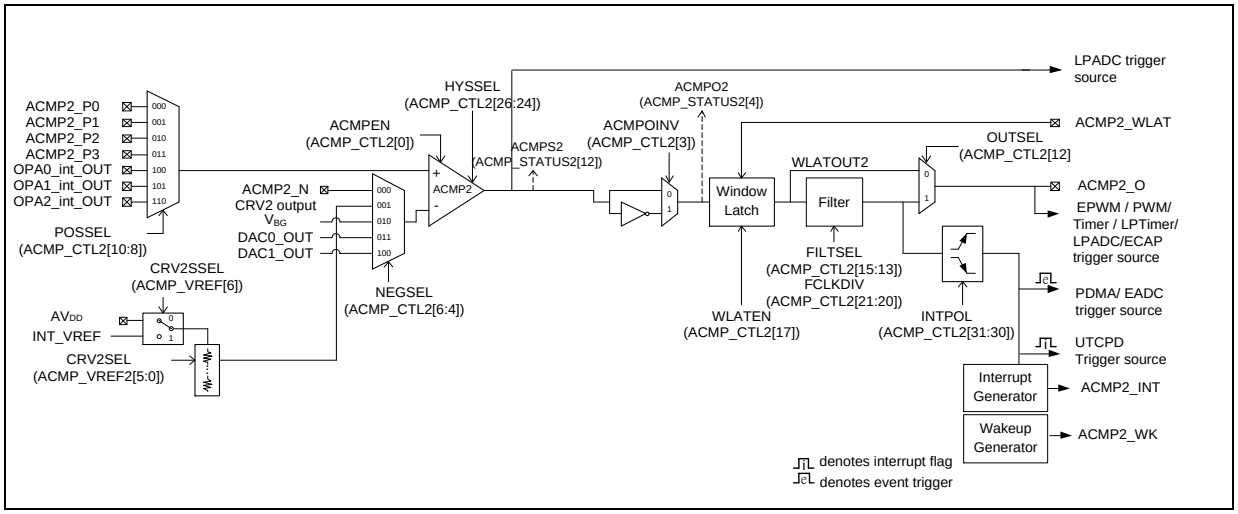


Figure 6.41-2 Analog Comparator 2 Block Diagram

6.41.4 Basic Configuration

6.41.4.1 ACMP0 Basic Configuration

- Clock Source Configuration
  - Enable ACMP0 peripheral clock in ACMP01CKEN (CLK_APBCLK0[7]).

- Reset Configuration
  - Reset ACMP0 controller in ACMP01RST (SYS_IPRST1[7]).

6.41.4.2 ACMP1 Basic Configuration

- Clock Source Configuration
  - Enable ACMP1 peripheral clock in ACMP01CKEN (CLK_APBCLK0[7]).
- Reset Configuration
  - Reset ACMP1 controller in ACMP01RST (SYS_IPRST1[7]).

6.41.4.3 ACMP2 Basic Configuration

- Clock Source Configuration
  - Enable ACMP2 peripheral clock in ACMP2CKEN (CLK_APBCLK2[7]).
- Reset Configuration
  - Reset ACMP2 controller in ACMP23RST (SYS_IPRST3[7]).

6.41.5 Functional Description

6.41.5.1 Hysteresis Function

The analog comparator provides the hysteresis function to make the comparator to have a stable output transition and it can refer to Figure 6.41-3. If comparator output is 0, it will not be changed to 1 until the positive input voltage exceeds the negative input voltage by a high threshold voltage. Similarly, if comparator output is 1, it will not be changed to 0 until the positive input voltage drops below the negative input voltage by a low threshold voltage.

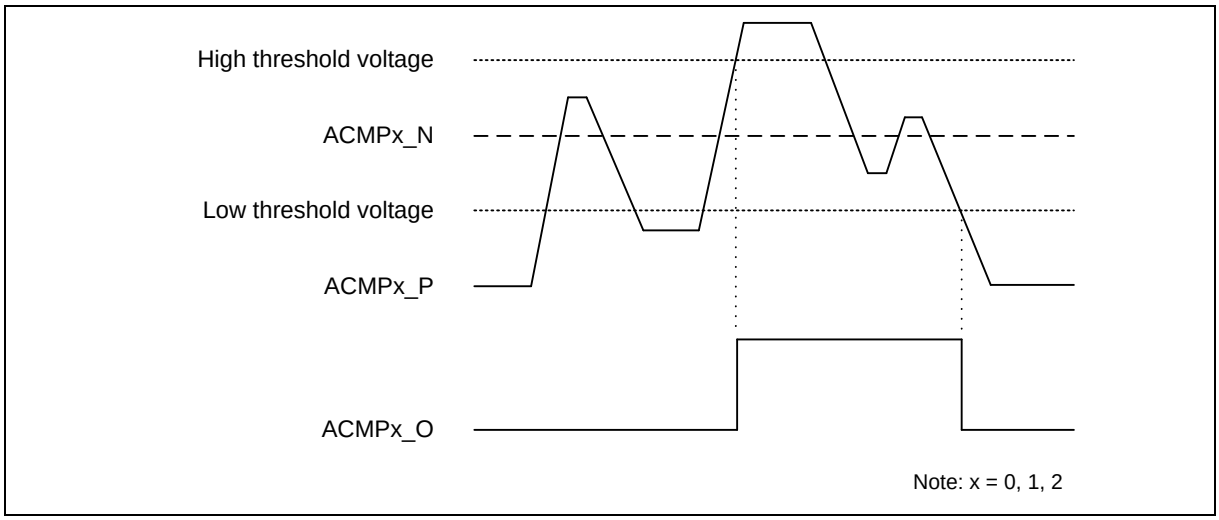


Figure 6.41-3 Comparator Hysteresis Function

6.41.5.2 Window Latch Mode

Figure 6.41-4 shows the example of comparator operation in window latch mode. Window latch mode can be enabled by setting WLATEN (ACMP_CTL0/1/2[17]) to 1. When window latch function enabled, ACMP0/1/2_WLAT pin is used to control the output WLATOUT0/1/2. When ACMP0/1/2_WLAT pin is high, ACMPO0/1/2 passes through to WLATOUT0/1/2. When ACMP0/1/2_WLAT pin is low, WLATOUT0/1/2 will keep last state of WLATOUT0/1/2.

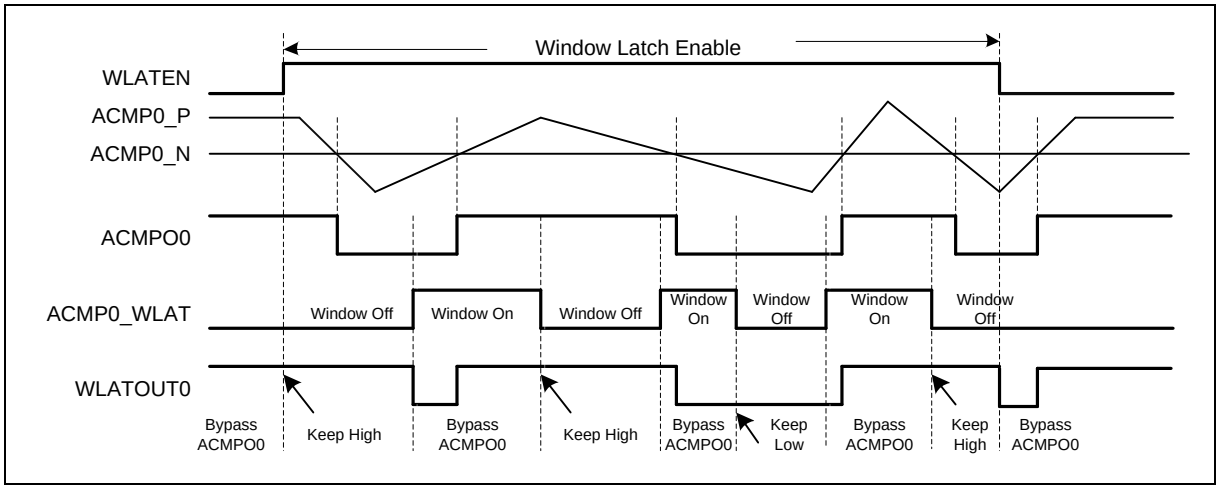


Figure 6.41-4 Window Latch Mode

6.41.5.3 Filter Function

The analog comparator provides filter function to avoid the un-stable state of comparator output.

By setting FILTSEL (ACMP_CTL0/1/2[15:13]), the comparator output would be sampled by consecutive PCLKs. With longer sample clocks, the comparator output would be more stable. But the sensitivity of comparator output would be reduced. Under the condition of setting FILTSEL to 3'h7 to get the comparator output is sampled 64 consecutive PCLKs, filter sample clock can be divided by 2 or 4 by setting FCLKDIV(ACMP_CTL0/1/2[21:20]) to get sampled 128, 256 consecutive PCLKs.

Figure 6.41-5 shows an example of filter function of ACMP0 with FILTSEL = 3 (4 PCLK). In this example, the comparing result is sampled by PCLK. All result must keep for 4 PCLK clocks before it can be output to ACMPO0. If the comparing result is shorter than 4 PCLK, it will be filtered.

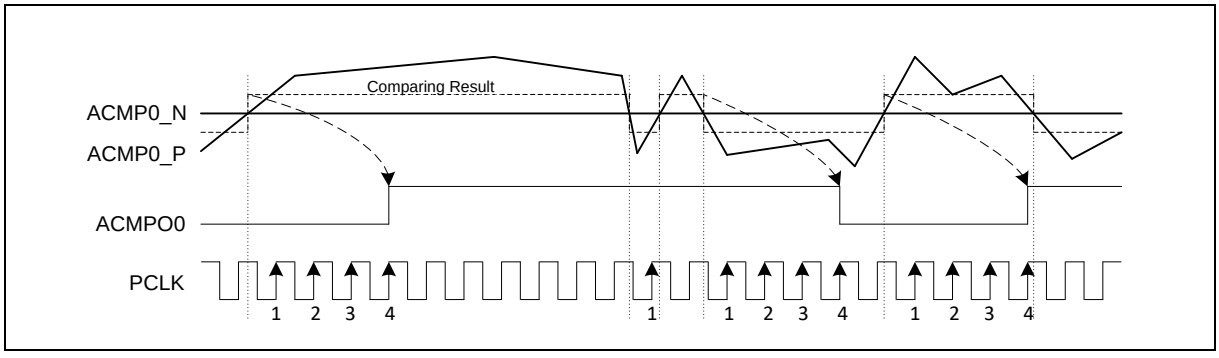


Figure 6.41-5 Example of Filter Function

6.41.5.4 Interrupt and Wake-up Sources

The outputs of ACMP0 and ACMP1 are reflected at ACMPO0 (ACMP_STATUS[4]) and ACMPO1 (ACMP_STATUS[5]) respectively.

Then they are processed by window latch and filter functions. Finally, the output signal could be utilized to assert interrupts. Refer to Figure 6.41-6, if ACMPIE of ACMP_CTL0/1 register is set to 1, the interrupt will be enabled. If the output state ACMPO0/1/2 is changed as the setting of INTPOL (ACMP_CTL0/1[31:30]), the comparator interrupt will be asserted and the corresponding flag, ACMPIF0 (ACMP_STATUS[0]) , ACMPIF1 (ACMP_STATUS[1]), The interrupt flag can be cleared to 0 by writing 1.

Figure 6.41-7 shows WKIF0/1(ACMP_STATUS[8]. ACMP_STATUS[9]) and WKIF2(ACMP_STATUS2[8]) will be set according to the setting of INTPOL (ACMP_CTL0/1/2[31:30]) if ACMP wakeup function is enabled. These three flags also cause interrupt rising that makes system wake-up from power down by ACMP. Figure 6.41-6 shows the interrupt sources of ACMP.

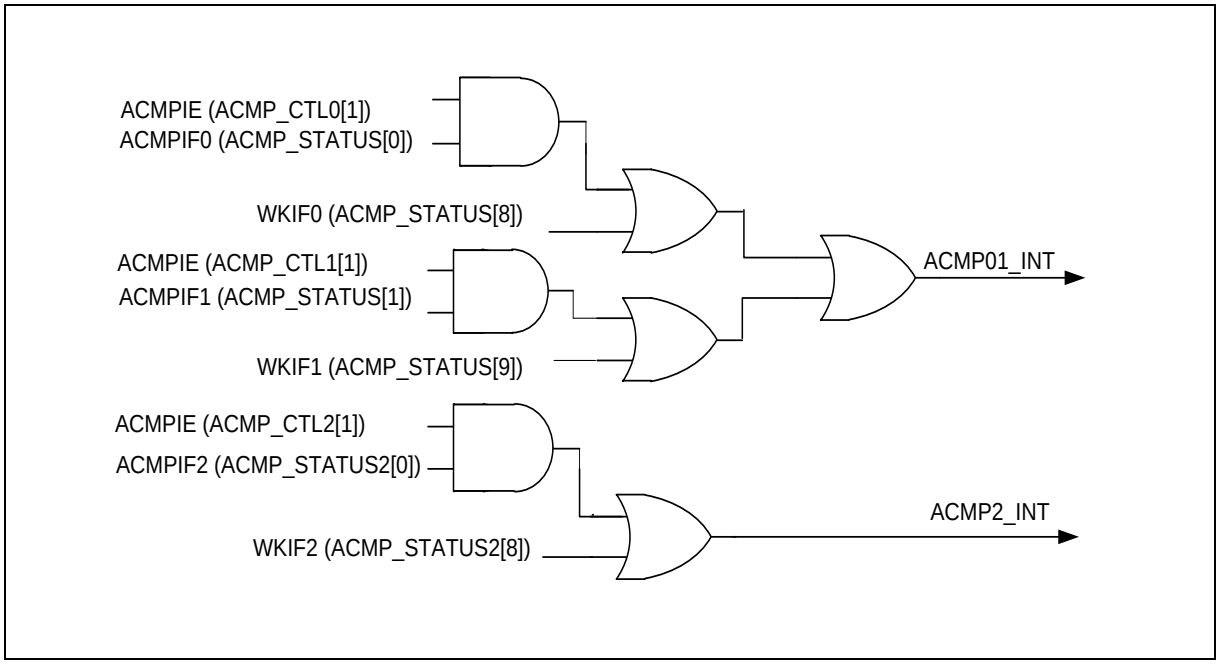


Figure 6.41-6 Comparator Controller Interrupt

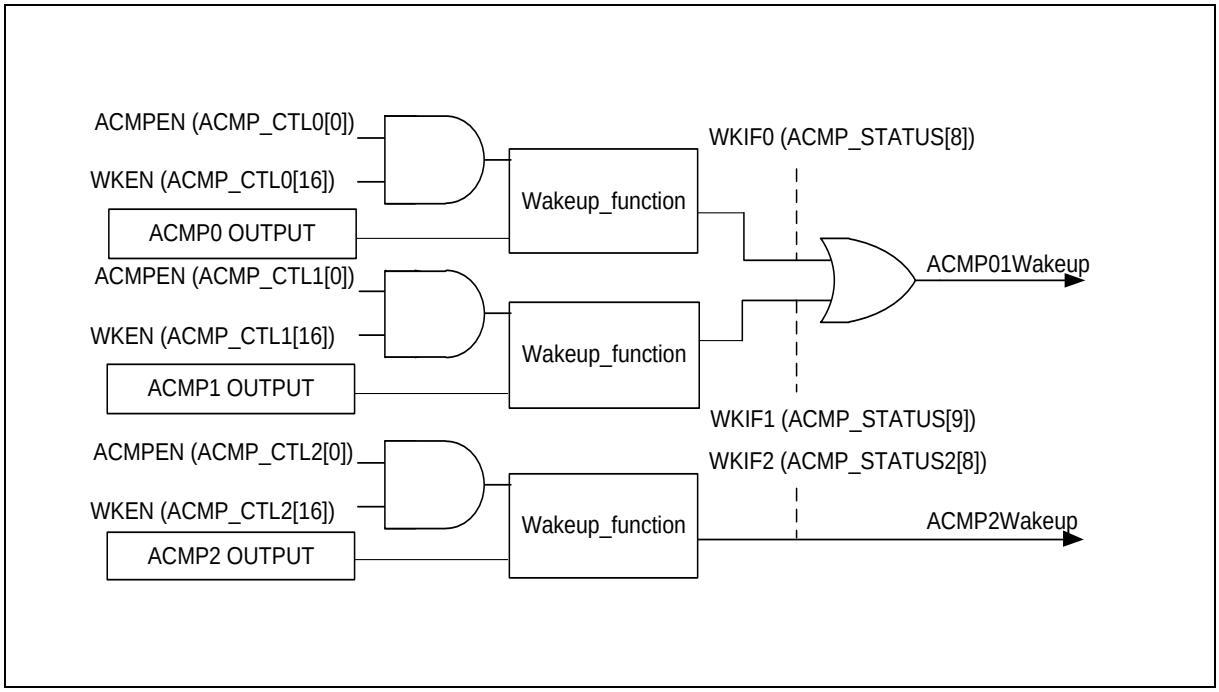


Figure 6.41-7 Comparator Controller Wakeup

CPU wakeup condition	Enable Bit	Wake-up Flag		
		NPD0/NPD1/NPD2/NPD3/NPD4/NPD5	SPD0/SPD1/SPD2	DPD/0

				DPD1
ACMP compare output edge condition match INTPOL (ACMP_CTLx[31:30]) – Rising, falling or both edge event	WKEN	WKIFx (ACMP_STATUS/ ACMP_STATUS2)	ACMPWKx (CLK_PMUSTS)	-

Table 6.41-1 Wake-up Support Table

Note that the filter and window latch function cannot work at any Power-down mode. User should disable these functions before chip enters Power-down mode.

6.41.5.5 Comparator Reference Voltage (CRV)

The comparator reference voltage (CRV) module is responsible for generating reference voltage for comparators. The CRV module consists of resistor ladder and analog switch. User can set the CRV output voltage by setting CRV0SEL (ACMP_VREF[5:0]), CRV1SEL (ACMP_VREF[21:16]), The CRV output voltage can be selected as the negative input of comparator by setting NEGSEL (ACMP_CTL0[6:4], ACMP_CTL1[6:4]). Figure 6.41-8 shows the block diagram of Comparator Reference Voltage. It is noted that the generation of CRV2/3 output is the same as CRV0/1

The resistor ladder will be disabled by hardware to reduce power consumption when NEGSEL (ACMP_CTL0/1[6:4]) is not selected to CRV module. The reference voltage of resistor ladder can be the voltage of AV_{DD} pin or the INT_VREF voltage which is controlled by SYS_VREFCTL register. Figure 6.41-8 shows comparator0 CRV0 output. For example, if user wants to get comparator1 CRV1 output, CRV1SSEL (ACMP_VREF[22]), CRV1SEL (ACMP_VREF[21:16]) must be set.

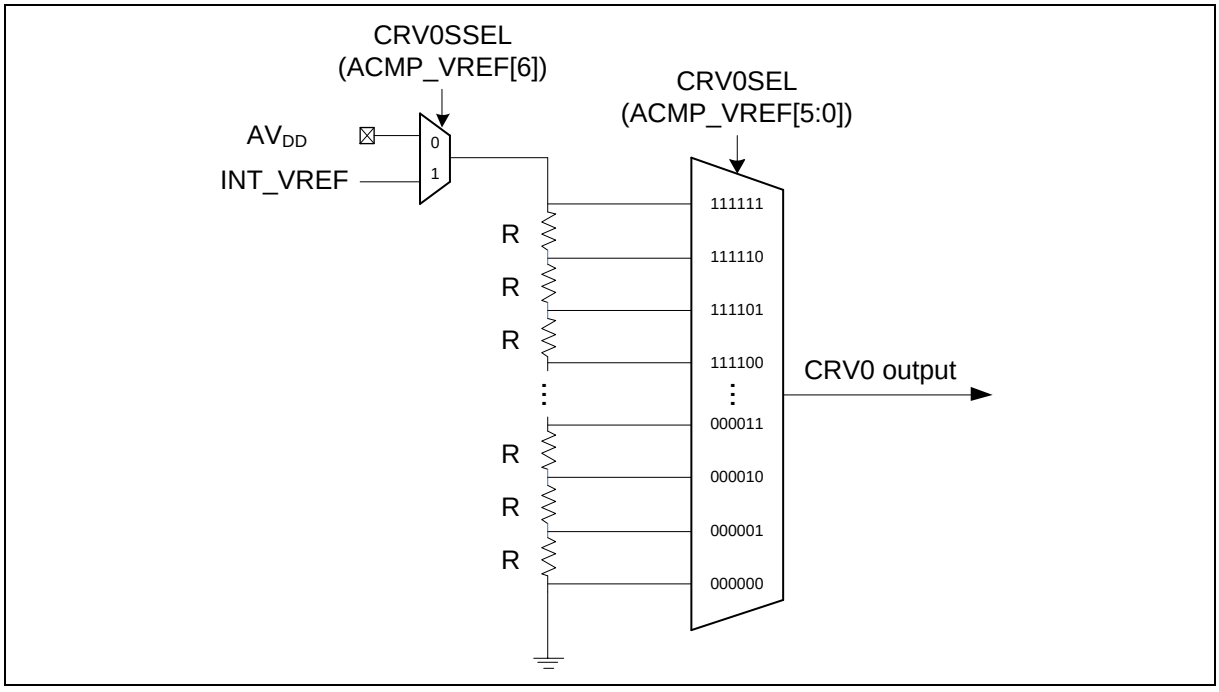


Figure 6.41-8 Comparator0 Reference Voltage Block Diagram

6.41.5.6 Window Compare Mode

The chip provide window compare mode. When window compare mode is enabled by setting WCMPSEL (ACMP_CTL0/1[18]) to 1, user can monitor a specific analog voltage source with a designated range. User can connect the specific analog voltage source to either the positive inputs of both comparators or the negative inputs of both comparators. The upper bound and lower bound of the

designated range are determined by the voltages applied to the other inputs of both comparators. If the output of a comparator is low and the other comparator outputs high, which means two comparators implies the upper and lower bound. User can directly monitor a specific analog voltage source via ACMPWO (ACMP_STATUS[16]). If ACMPWO is high, it implies a specific analog voltage source is in the range of upper and lower bound, which are called as the analog voltage is in the window.

Figure 6.41-9 illustrates an example of window compare mode. In this example, once window compare mode is selected, user can choose one of those positive input sources of each comparator and connect these two inputs together outside the chip.

If ACMPS0 outputs high and ACMPS1 outputs low, it means the voltage source is in the range of lower bound and upper bound, which are called as the voltage source in the window. Otherwise, the voltage source is outside the window.

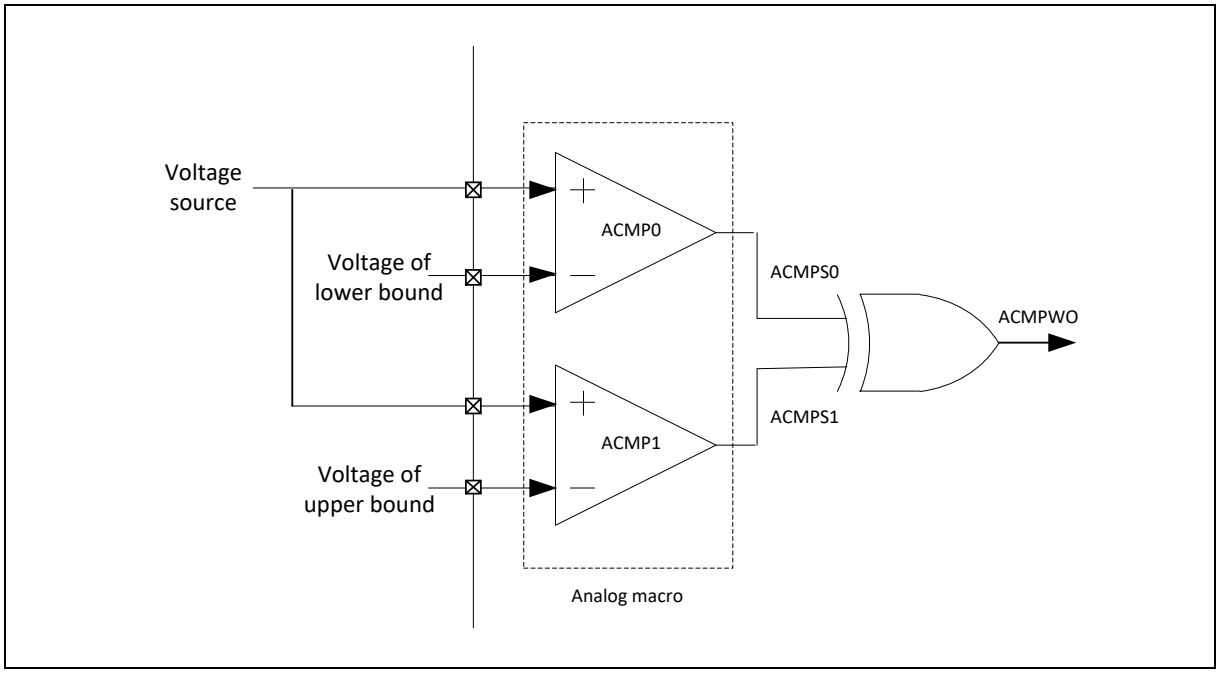


Figure 6.41-9 Example of Window Compare Mode

The example of comparator window output (ACMPWO) can be shown in ACMP_STATUS[16] and the truth table of window compare logic is shown in Table 6.41-2.

ACMPS0	ACMPS1	ACMPWO
0	0	0
0	1	1
1	1	0
1	0	1

Table 6.41-2 Truth Table of Window Compare Logic

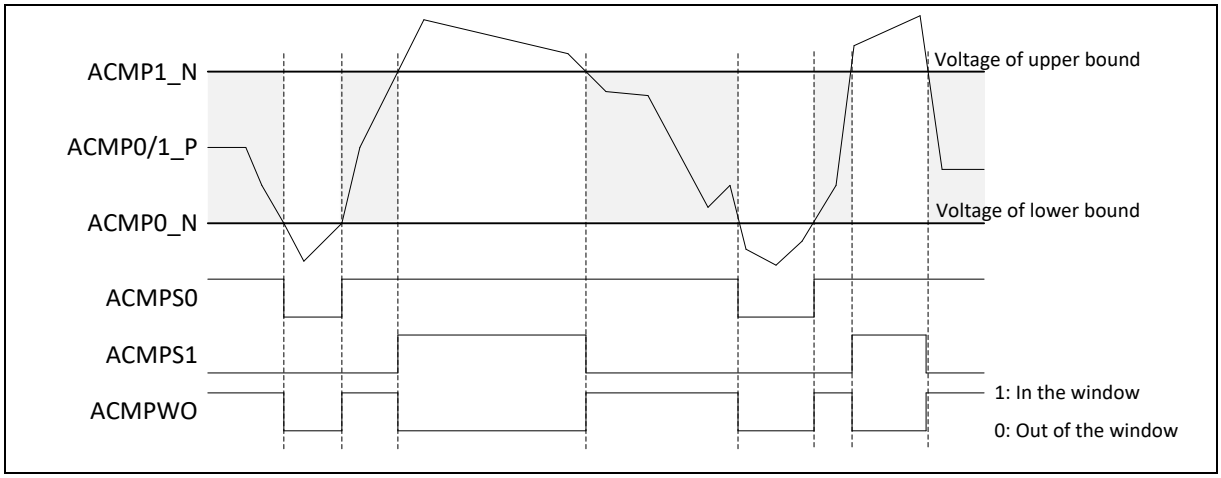


Figure 6.41-10 Example of Window Compare Mode

As shown in Figure 6.41-10, if ACMPWO equals 1, it means positive input voltage is inside the window. Otherwise, the positive input voltage is outside the window. Therefore, ACMPWO can be used to monitor voltage transition of external analog pin. Furthermore, ACMPWO still can be applied to window latch, filter functions and interrupt of ACMP.

Note that negative inputs must choose different source. Otherwise, the function will be meaningless.

6.41.5.7 Calibration Function

The three comparators have their own trim bits that can be used to calibrate the offset voltage. The offset voltage comes from both mismatch of NMOS-type differential and PMOS-type differential input stages. Calibration can be started by setting CALTRG0(ACMP_CALCTL[0]), CALTRG1 (ACMP_CALCTL[1]), CALTRG2(ACMP_CALCTL2[0]).

The common mode input range is achieved by using an NMOS and a PMOS differential pairs connected in parallel. After calibration, reading the ACMP_CALSR register can monitor calibration done status. Take a brief calibration flow for example: Once ACMPEN (ACMP_CTLx[0], x=0,1,2) is enabled or change MODESEL(ACMP_CTLx[29:28], x=0,1,2), hardware will automatically load default calibrated trim values to compensate offset voltage. Therefore, user can directly use comparator without doing additional calibration action. If user wants to start calibration function again, the newer calibrated trim value will be updated after calibration done. Note that every time ACMPEN is set, user must set CALTRG0 (ACMP_CALCTL[0]), CALTRG1 (ACMP_CALCTL[1]), and CALTRG2 (ACMP_CALCTL2[0]) and NEGSEL (ACMP_CTLx[6:4], x= 0,1,2) equal to 2'b01 to start calibration function to get newer offset trim values, or comparator will operate with previous offset trim values.

6.41.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
ACMP Base Address: **ACMP01_BA = 0x4004_5000** **ACMP2_BA = 0x400C_9000**				
ACMP_CTL0	ACMP01_BA+0x00	R/W	Analog Comparator 0 Control Register	0x1000_0000
ACMP_CTL1	ACMP01_BA+0x04	R/W	Analog Comparator 1 Control Register	0x1000_0000
ACMP_STATUS	ACMP01_BA+0x08	R/W	Analog Comparator 01 Status Register	0x0000_0000
ACMP_VREF	ACMP01_BA+0x0C	R/W	Analog Comparator 01 Reference Voltage Control Register	0x0000_0000
ACMP_CALCTL	ACMP01_BA+0x10	R/W	Analog Comparator 01 Calibration Control Register	0x0000_0050
ACMP_CALSR	ACMP01_BA+0x14	R	Analog Comparator 01 Calibration Status Register	0x0000_0000
ACMP_CTL2	ACMP2_BA+0x00	R/W	Analog Comparator 2 Control Register	0x1000_0000
ACMP_STATUS2	ACMP2_BA+0x08	R/W	Analog Comparator 2 Status Register	0x0000_0000
ACMP_VREF2	ACMP2_BA+0x0C	R/W	Analog Comparator 2 Reference Voltage Control Register	0x0000_0000
ACMP_CALCTL2	ACMP2_BA+0x10	R/W	Analog Comparator 2 Calibration Control Register	0x0000_0050
ACMP_CALSR2	ACMP2_BA+0x14	R	Analog Comparator 2 Calibration Status Register	0x0000_0000

6.41.7 Register Description

Analog Comparator 0 Control Register (ACMP_CTL0)

Register	Offset	R/W	Description	Reset Value
ACMP_CTL0	ACMP01_BA+0x00	R/W	Analog Comparator 0 Control Register	0x1000_0000

31	30	29	28	27	26	25	24
INTPOL		MODESEL		Reserved	HYSSEL		
23	22	21	20	19	18	17	16
Reserved		FCLKDIV		Reserved	WCMPSEL	WLATEN	WKEN
15	14	13	12	11	10	9	8
FILTSEL			OUTSEL	Reserved	POSSEL		
7	6	5	4	3	2	1	0
Reserved	NEGSEL			ACMPOINV	Reserved	ACMPIE	ACMPEN

Bits	Description	
[31:30]	**INTPOL**	**Interrupt Condition Polarity Selection** ACMPIF0 will be set to 1 when ACMP output edge condition is detected. 00 = Rising edge or falling edge. 01 = Rising edge. 10 = Falling edge. 11 = Reserved.
[29:28]	**MODESEL**	**ACMP Operating Mode Selection** The bit affects analog characteristics such as operating current, input offset voltage, setup time, and propagation delay. 00 = ACMP0 operating mode0. 01 = ACMP0 operating mode1. 10 = ACMP0 operating mode2. 11 = ACMP0 operating mode3. **Note:** Please refer to Analog Comparator Controller (ACMP) of Analog Characteristics in the Datasheet.
[27]	**Reserved**	Reserved.
[26:24]	**HYSSEL**	**Hysteresis Mode Selection** 000 = Hysteresis is 0mV. 010 = Hysteresis is 20mV for MODESEL =2'b00, 2'b01. 011 = Hysteresis is 20mV for MODESEL = 2'b10, 2'b11. 100 = Hysteresis is 40mV for MODESEL = 2'b00, 2'b01. 101 = Hysteresis is 40mV for MODESEL = 2'b10, 2'b11. Others = Reserved.
[23:22]	**Reserved**	Reserved.
[21:20]	**FCLKDIV**	**ACMP Output Filter Clock Divider** 00 = ACMP0 output filter clock = PCLK. 01 = ACMP0 output filter clock = PCLK/2.

		10 = ACMP0 output filter clock = PCLK/4. 11 = Reserved. **Note:** Use FCLKDIV under the condition of FILTSEL = 3'h7, then set FCLKDIV to get the effect of filtering 128,256 consecutive PCLKs.
[19]	**Reserved**	Reserved.
[18]	**WCMPSEL**	**Window Compare Mode Selection** 0 = Window Compare Mode Disabled. 1 = Window Compare Mode Selected.
[17]	**WLATEN**	**Window Latch Mode Enable Bit** 0 = Window Latch Mode Disabled. 1 = Window Latch Mode Enabled.
[16]	**WKEN**	**Power-down Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.
[15:13]	**FILTSEL**	**ACMP Output Filter Count Selection** 000 = Filter function is Disabled. 001 = ACMP0 output is sampled 1 consecutive PCLK. 010 = ACMP0 output is sampled 2 consecutive PCLKs. 011 = ACMP0 output is sampled 4 consecutive PCLKs. 100 = ACMP0 output is sampled 8 consecutive PCLKs. 101 = ACMP0 output is sampled 16 consecutive PCLKs. 110 = ACMP0 output is sampled 32 consecutive PCLKs. 111 = ACMP0 output is sampled 64 consecutive PCLKs.
[12]	**OUTSEL**	**ACMP Output Select** 0 = ACMP0 output to ACMP0_O pin is unfiltered ACMP output. 1 = ACMP0 output to ACMP0_O pin is from filter output.
[11]	**Reserved**	Reserved.
[10:8]	**POSSEL**	**ACMP Positive Input Selection** 000 = ACMP0_P0 pin. 001 = ACMP0_P1 pin. 010 = ACMP0_P2 pin. 011 = ACMP0_P3 pin. 100 = OPA0_int_OUT. 101 = OPA1_int_OUT. 110 = OPA2_int_OUT.
[7]	**Reserved**	Reserved.
[6:4]	**NEGSEL**	**ACMP Negative Input Selection** 000 = ACMP0_N pin. 001 = Internal comparator reference voltage (CRV0). 010 = Band-gap voltage. 011 = DAC0 output. 100 = DAC1 output. **Note:** NEGSEL must select 0x1 in calibration mode.
[3]	**ACMPOINV**	**ACMP Output Inverse** 0 = ACMP0 output inverse Disabled.

		1 = ACMP0 output inverse Enabled.
[2]	**Reserved**	Reserved.
[1]	**ACMPIE**	**ACMP Interrupt Enable Bit** 0 = ACMP0 interrupt Disabled. 1 = ACMP0 interrupt Enabled. **Note:** If WKEN (ACMP_CTL0[16]) is set to 1, the wake-up interrupt function will be enabled as well.
[0]	**ACMPEN**	**ACMP Enable Bit** 0 = ACMP0 Disabled. 1 = ACMP0 Enabled.

Analog Comparator 1 Control Register (ACMP_CTL1)

Register	Offset	R/W	Description	Reset Value
ACMP_CTL1	ACMP01_BA+0x04	R/W	Analog Comparator 1 Control Register	0x1000_0000

31	30	29	28	27	26	25	24
INTPOL		MODESEL		Reserved	HYSSEL		
23	22	21	20	19	18	17	16
Reserved		FCLKDIV		Reserved	WCMPSEL	WLATEN	WKEN
15	14	13	12	11	10	9	8
FILTSEL			OUTSEL	Reserved	POSSEL		
7	6	5	4	3	2	1	0
Reserved	NEGSEL			ACMPOINV	Reserved	ACMPIE	ACMPEN

Bits	Description	
[31:30]	**INTPOL**	**Interrupt Condition Polarity Selection** ACMPIF1 will be set to 1 when ACMP output edge condition is detected. 00 = Rising edge or falling edge. 01 = Rising edge. 10 = Falling edge. 11 = Reserved.
[29:28]	**MODESEL**	**ACMP Operating Mode Selection** The bit affects analog characteristics such as operating current, input offset voltage, setup time, and propagation delay. 00 = ACMP1 operating mode0. 01 = ACMP1 operating mode1. 10 = ACMP1 operating mode2. 11 = ACMP1 operating mode3. **Note:** Please refer to Analog Comparator Controller (ACMP) of Analog Characteristics in the Datasheet.
[27]	**Reserved**	Reserved.
[26:24]	**HYSSEL**	**Hysteresis Mode Selection** 000 = Hysteresis is 0mV. 010 = Hysteresis is 20mV for MODESEL =2'b00, 2'b01. 011 = Hysteresis is 20mV for MODESEL = 2'b10, 2'b11. 100 = Hysteresis is 40mV for MODESEL = 2'b00, 2'b01. 101 = Hysteresis is 40mV for MODESEL = 2'b10, 2'b11. Others = Reserved.
[23:22]	**Reserved**	Reserved.
[21:20]	**FCLKDIV**	**ACMP Output Filter Clock Divider** 00 = ACMP1 output filter clock = PCLK. 01 = ACMP1 output filter clock = PCLK/2. 10 = ACMP1 output filter clock = PCLK/4. 11 = Reserved.

Bits	Description	
[19]	**Reserved**	Reserved.
[18]	**WCMPSEL**	**Window Compare Mode Selection** 0 = Window Compare Mode Disabled. 1 = Window Compare Mode is Selected.
[17]	**WLATEN**	**Window Latch Mode Enable Bit** 0 = Window Latch Mode Disabled. 1 = Window Latch Mode Enabled.
[16]	**WKEN**	**Power-down Wakeup Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.
[15:13]	**FILTSEL**	**ACMP Output Filter Count Selection** 000 = Filter function is Disabled. 001 = ACMP1 output is sampled 1 consecutive PCLK. 010 = ACMP1 output is sampled 2 consecutive PCLKs. 011 = ACMP1 output is sampled 4 consecutive PCLKs. 100 = ACMP1 output is sampled 8 consecutive PCLKs. 101 = ACMP1 output is sampled 16 consecutive PCLKs. 110 = ACMP1 output is sampled 32 consecutive PCLKs. 111 = ACMP1 output is sampled 64 consecutive PCLKs.
[12]	**OUTSEL**	**ACMP Output Select** 0 = ACMP1 output to ACMP1_O pin is unfiltered ACMP output. 1 = ACMP1 output to ACMP1_O pin is from filter output.
[11]	**Reserved**	Reserved.
[10:8]	**POSSEL**	**ACMP Positive Input Selection** 000 = ACMP1_P0 pin. 001 = ACMP1_P1 pin. 010 = ACMP1_P2 pin. 011 = ACMP1_P3 pin. 100 = OPA0_int_OUT. 101 = OPA1_int_OUT. 110 = OPA2_int_OUT.
[7]	**Reserved**	Reserved.
[6:4]	**NEGSEL**	**ACMP Negative Input Selection** 000 = ACMP1_N pin. 001 = Internal comparator reference voltage (CRV1). 010 = Band-gap voltage. 011 = DAC0 output. 100 = DAC1 output. **Note:** NEGSEL must select 0x1 in calibration mode.
[3]	**ACMPOINV**	**ACMP Output Inverse Control** 0 = ACMP1 output inverse Disabled. 1 = ACMP1 output inverse Enabled.

Bits	Description	
[2]	**Reserved**	Reserved.
[1]	**ACMPIE**	**ACMP Interrupt Enable Bit** 0 = ACMP1 interrupt Disabled. 1 = ACMP1 interrupt Enabled. **Note:** If WKEN (ACMP_CTL1[16]) is set to 1, the wake-up interrupt function will be enabled as well.
[0]	**ACMPEN**	**ACMP Enable Bit** 0 = ACMP1 Disabled. 1 = ACMP1 Enabled.

Analog Comparator 01 Status Register (ACMP_STATUS)

Register	Offset	R/W	Description	Reset Value
ACMP_STATUS	ACMP01_BA+0x08	R/W	Analog Comparator 01 Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							ACMPWO
15	14	13	12	11	10	9	8
Reserved		ACMPS1	ACMPS0	Reserved		WKIF1	WKIF0
7	6	5	4	3	2	1	0
Reserved		ACMPO1	ACMPO0	Reserved		ACMPIF1	ACMPIF0

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**ACMPWO**	**ACMP Window Output** This bit shows the output status of window compare mode 0 = The positive input voltage is outside the window. 1 = The positive input voltage is in the window.
[15:14]	**Reserved**	Reserved.
[13]	**ACMPS1**	**ACMP Status 1** Synchronized to the PCLK to allow reading by software. Cleared when the ACMP1 is disabled, i.e. ACMPEN (ACMP_CTL1[0]) is cleared to 0.
[12]	**ACMPS0**	**ACMP Status 0** Synchronized to the PCLK to allow reading by software. Cleared when the ACMP0 is disabled, i.e. ACMPEN (ACMP_CTL0[0]) is cleared to 0.
[11:10]	**Reserved**	Reserved.
[9]	**WKIF1**	**ACMP Power-down Wake-up Interrupt Flag 1** This bit will be set to 1 when ACMP1 wake-up interrupt event occurs. 0 = No power-down wake-up occurred. 1 = Power-down wake-up occurred. **Note:** Write 1 to clear this bit to 0.
[8]	**WKIF0**	**ACMP Power-down Wake-up Interrupt Flag 0** This bit will be set to 1 when ACMP0 wake-up interrupt event occurs. 0 = No power-down wake-up occurred. 1 = Power-down wake-up occurred. **Note:** Write 1 to clear this bit to 0.
[7:6]	**Reserved**	Reserved.
[5]	**ACMPO1**	**ACMP Output 1** Synchronized to the PCLK to allow reading by software. Cleared when the ACMP1 is disabled, i.e. ACMPEN (ACMP_CTL1[0]) is cleared to 0.

Bits	Description	
[4]	**ACMPO0**	**ACMP Output 0** Synchronized to the PCLK to allow reading by software. Cleared when the ACMP0 is disabled, i.e. ACMPEN (ACMP_CTL0[0]) is cleared to 0.
[3:2]	**Reserved**	Reserved.
[1]	**ACMPIF1**	**ACMP Interrupt Flag 1** This bit is set by hardware when the edge condition defined by INTPOL (ACMP_CTL1[31:30]) is detected on ACMP1 output. This will cause an interrupt if ACMPIE (ACMP_CTL1[1]) is set to 1. **Note:** Write 1 to clear this bit to 0.
[0]	**ACMPIF0**	**ACMP Interrupt Flag 0** This bit is set by hardware when the edge condition defined by INTPOL (ACMP_CTL0[31:30]) is detected on ACMP0 output. This will generate an interrupt if ACMPIE (ACMP_CTL0[1]) is set to 1. **Note:** Write 1 to clear this bit to 0.

Analog Comparator 01 Reference Voltage Control Register (ACMP_VREF)

Register	Offset	R/W	Description	Reset Value
ACMP_VREF	ACMP01_BA+0x0C	R/W	Analog Comparator 01 Reference Voltage Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							CRV1EN
23	22	21	20	19	18	17	16
Reserved	CRV1SSEL	CRV1SEL					
15	14	13	12	11	10	9	8
Reserved							CRV0EN
7	6	5	4	3	2	1	0
Reserved	CRV0SSEL	CRV0SEL					

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**CRV1EN**	**CRV1 Enable Bit** 0 = CRV1 Disabled. 1 = CRV1 Enabled.
[23]	**Reserved**	Reserved.
[22]	**CRV1SSEL**	**CRV1 Source Voltage Selection** 0 = AV_{DD} is selected as CRV1 source voltage. 1 = The reference voltage defined by VREFCTL (SYS_VREFCTL[4:0]) register is selected as CRV1 source voltage.
[21:16]	**CRV1SEL**	**ACMP Reference Voltage Setting 1** Set CRV output voltage for ACMP1 negtive input. CRV1 = CRV1 source voltage * (CRV1SEL)/ 63.
[15:9]	**Reserved**	Reserved.
[8]	**CRV0EN**	**CRV0 Enable Bit** 0 = CRV0 Disabled. 1 = CRV0 Enabled.
[7]	**Reserved**	Reserved.
[6]	**CRV0SSEL**	**CRV0 Source Voltage Selection** 0 = AV_{DD} is selected as CRV0 source voltage. 1 = The reference voltage defined by VREFCTL (SYS_VREFCTL[4:0]) register is selected as CRV0 source voltage.
[5:0]	**CRV0SEL**	**ACMP Reference Voltage Setting 0** Set CRV output voltage for ACMP0 negtive input. CRV0 = CRV0 source voltage * (CRV0SEL)/ 63.

Analog Comparator 01 Calibration Control Register (ACMP_CALCTL)

Register	Offset	R/W	Description	Reset Value
ACMP_CALCTL	ACMP01_BA+0x10	R/W	Analog Comparator 01 Calibration Control Register	0x0000_0050

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CALTRG1	CALTRG0

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CALTRG1**	**ACMP Calibration Trigger 1** Set ACMP1 calibration trigger. 0 = Calibration is stopped. 1 = Calibration is triggered. **Note 1:** Before this bit is enabled, ACMPEN(ACMP_CTL1[0]) should be set and the internal high speed RC oscillator (HIRC) should be enabled in advance. **Note 2:** Hardware will auto clear this bit when the next calibration is triggered by software. **Note 3:** If user must trigger calibration twice or more times, the second trigger has to wait at least 300us after the previous calibration is done.
[0]	**CALTRG0**	**ACMP Calibration Trigger 0** Set ACMP0 calibration trigger. 0 = Calibration is stopped. 1 = Calibration is triggered. **Note 1:** Before this bit is enabled, ACMPEN(ACMP_CTL0[0]) should be set and the internal high speed RC oscillator (HIRC) should be enabled in advance. **Note 2:** Hardware will auto clear this bit when the next calibration is triggered by software. **Note 3:** If user must trigger calibration twice or more times, the second trigger has to wait at least 300us after the previous calibration is done.

Analog Comparator 01 Calibration Status Register (ACMP_CALSR)

Register	Offset	R/W	Description	Reset Value
ACMP_CALSR	ACMP01_BA+0x14	R	Analog Comparator 01 Calibration Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**DONE1**	**Reserved**			**DONE0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**DONE1**	**ACMP1 Calibration Done Status** This bit will be set when ACMP1 calibration is done. 0 = Calibrating. 1 = Calibration done. **Note:** This bit is is cleared by writing 1 into it.
[3:1]	**Reserved**	Reserved.
[0]	**DONE0**	**ACMP0 Calibration Done Status** This bit will be set when ACMP0 calibration is done. 0 = Calibrating. 1 = Calibration done. **Note:** This bit is is cleared by writing 1 into it.

Analog Comparator 2 Control Register (ACMP_CTL2)

Register	Offset	R/W	Description	Reset Value
ACMP_CTL2	ACMP2_BA+0x00	R/W	Analog Comparator 2 Control Register	0x1000_0000

31	30	29	28	27	26	25	24
INTPOL		MODESEL		Reserved	HYSSEL		
23	22	21	20	19	18	17	16
Reserved		FCLKDIV		Reserved		WLATEN	WKEN
15	14	13	12	11	10	9	8
FILTSEL			OUTSEL	Reserved	POSSEL		
7	6	5	4	3	2	1	0
Reserved	NEGSEL			ACMPOINV	Reserved	ACMPIE	ACMPEN

Bits	Description	
[31:30]	**INTPOL**	**Interrupt Condition Polarity Selection** ACMPIF2 will be set to 1 when ACMP output edge condition is detected. 00 = Rising edge or falling edge. 01 = Rising edge. 10 = Falling edge. 11 = Reserved.
[29:28]	**MODESEL**	**ACMP Operating Mode Selection** The bit affects analog characteristics such as operating current, input offset voltage, setup time, and propagation delay. 00 = ACMP2 operating mode0. 01 = ACMP2 operating mode1. 10 = ACMP2 operating mode2. 11 = ACMP2 operating mode3. **Note:** Please refer to Analog Comparator Controller (ACMP) of Analog Characteristics in the Datasheet.
[27]	**Reserved**	Reserved.
[26:24]	**HYSSEL**	**Hysteresis Mode Selection** 000 = Hysteresis is 0mV. 010 = Hysteresis is 20mV for MODESEL =2'b00, 2'b01. 011 = Hysteresis is 20mV for MODESEL = 2'b10, 2'b11. 100 = Hysteresis is 40mV for MODESEL = 2'b00, 2'b01. 101 = Hysteresis is 40mV for MODESEL = 2'b10, 2'b11. Others = Reserved.
[23:22]	**Reserved**	Reserved.
[21:20]	**FCLKDIV**	**ACMP Output Filter Clock Divider** 00 = ACMP2 output filter clock = PCLK. 01 = ACMP2 output filter clock = PCLK/2. 10 = ACMP2 output filter clock = PCLK/4.

		11 = Reserved. **Note:** Use FCLKDIV under the condition of FILTSEL = 3'h7, then set FCLKDIV to get the effect of filtering 128,256 consecutive PCLKs.
[19:18]	**Reserved**	Reserved.
[17]	**WLATEN**	**Window Latch Mode Enable Bit** 0 = Window Latch Mode Disabled. 1 = Window Latch Mode Enabled.
[16]	**WKEN**	**Power-down Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.
[15:13]	**FILTSEL**	**ACMP Output Filter Count Selection** 000 = Filter function is Disabled. 001 = ACMP2 output is sampled 1 consecutive PCLK. 010 = ACMP2 output is sampled 2 consecutive PCLKs. 011 = ACMP2 output is sampled 4 consecutive PCLKs. 100 = ACMP2 output is sampled 8 consecutive PCLKs. 101 = ACMP2 output is sampled 16 consecutive PCLKs. 110 = ACMP2 output is sampled 32 consecutive PCLKs. 111 = ACMP2 output is sampled 64 consecutive PCLKs.
[12]	**OUTSEL**	**ACMP Output Select** 0 = ACMP2 output to ACMP2_O pin is unfiltered ACMP output. 1 = ACMP2 output to ACMP2_O pin is from filter output.
[11]	**Reserved**	Reserved.
[10:8]	**POSSEL**	**ACMP Positive Input Selection** 000 = ACMP2_P0 pin. 001 = ACMP2_P1 pin. 010 = ACMP2_P2 pin. 011 = ACMP2_P3 pin. 100 = OPA0_int_OUT. 101 = OPA1_int_OUT. 110 = OPA2_int_OUT.
[7]	**Reserved**	Reserved.
[6:4]	**NEGSEL**	**ACMP Negative Input Selection** 000 = ACMP2_N pin. 001 = Internal comparator reference voltage (CRV2). 010 = Band-gap voltage. 011 = DAC0 output. 100 = DAC1 output. **Note:** NEGSEL must select 0x1 in calibration mode.
[3]	**ACMPOINV**	**ACMP Output Inverse** 0 = ACMP2 output inverse Disabled. 1 = ACMP2 output inverse Enabled.
[2]	**Reserved**	Reserved.
[1]	**ACMPIE**	**ACMP Interrupt Enable Bit** 0 = ACMP2 interrupt Disabled.

		1 = ACMP2 interrupt Enabled. **Note:** If WKEN (ACMP_CTL2[16]) is set to 1, the wake-up interrupt function will be enabled as well.
[0]	**ACMPEN**	**ACMP Enable Bit** 0 = ACMP2 Disabled. 1 = ACMP2 Enabled.

Analog Comparator 2 Status Register (ACMP_STATUS2)

Register	Offset	R/W	Description	Reset Value
ACMP_STATUS2	ACMP2_BA+0x08	R/W	Analog Comparator 2 Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			ACMPS2	Reserved			WKIF2
7	6	5	4	3	2	1	0
Reserved			ACMPO2	Reserved			ACMPIF2

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**ACMPS2**	**ACMP Status 0** Synchronized to the PCLK to allow reading by software. Cleared when the ACMP2 is disabled, i.e. ACMPEN (ACMP_CTL2[0]) is cleared to 0.
[11:9]	**Reserved**	Reserved.
[8]	**WKIF2**	**ACMP Power-down Wake-up Interrupt Flag 0** This bit will be set to 1 when ACMP2 wake-up interrupt event occurs. 0 = No power-down wake-up occurred. 1 = Power-down wake-up occurred. **Note:** Write 1 to clear this bit to 0.
[7:5]	**Reserved**	Reserved.
[4]	**ACMPO2**	**ACMP Output 0** Synchronized to the PCLK to allow reading by software. Cleared when the ACMP2 is disabled, i.e. ACMPEN (ACMP_CTL2[0]) is cleared to 0.
[3:1]	**Reserved**	Reserved.
[0]	**ACMPIF2**	**ACMP Interrupt Flag 0** This bit is set by hardware when the edge condition defined by INTPOL (ACMP_CTL2[31:30]) is detected on ACMP2 output. This will generate an interrupt if ACMPIE (ACMP_CTL2[1]) is set to 1. **Note:** Write 1 to clear this bit to 0.

Analog Comparator 2 Reference Voltage Control Register (ACMP_VREF2)

Register	Offset	R/W	Description	Reset Value
ACMP_VREF2	ACMP2_BA+0x0C	R/W	Analog Comparator 2 Reference Voltage Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							CRV2EN
7	6	5	4	3	2	1	0
Reserved	CRV2SSEL	CRV2SEL					

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CRV2EN**	**CRV2 Enable Bit** 0 = CRV2 Disabled. 1 = CRV2 Enabled.
[7]	**Reserved**	Reserved.
[6]	**CRV2SSEL**	**CRV2 Source Voltage Selection** 0 = AV_{DD} is selected as CRV2 source voltage. 1 = The reference voltage defined by VREFCTL (SYS_VREFCTL[4:0]) register is selected as CRV2 source voltage.
[5:0]	**CRV2SEL**	**ACMP Reference Voltage Setting 0** CRV2 = CRV2 source voltage * (CRV2SEL)/ 63.

Analog Comparator 2 Calibration Control Register (ACMP_CALCTL2)

Register	Offset	R/W	Description	Reset Value
ACMP_CALCTL2	ACMP2_BA+0x10	R/W	Analog Comparator 2 Calibration Control Register	0x0000_0050

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CALTRG2

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CALTRG2**	**ACMP Calibration Trigger 0** Set ACMP2 calibration trigger. 0 = Calibration is stopped. 1 = Calibration is triggered. **Note 1:** Before this bit is enabled, ACMPEN(ACMP_CTL2[0]) should be set and the internal high speed RC oscillator (HIRC) should be enabled in advance. **Note 2:** Hardware will auto clear this bit when the next calibration is triggered by software. **Note 3:** If user must trigger calibration twice or more times, the second trigger has to wait at least 300us after the previous calibration is done.

Analog Comparator 2 Calibration Status Register (ACMP_CALSR2)

Register	Offset	R/W	Description	Reset Value
ACMP_CALSR2	ACMP2_BA+0x14	R	Analog Comparator 2 Calibration Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							DONE2

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**DONE2**	**ACMP Calibration Done Status 0** This bit will be set when ACMP2 calibration is done. 0 = Calibrating. 1 = Calibration done. **Note:** This bit is is cleared by writing 1 into it.

6.42 Enhanced 12-bit Analog-to-Digital Converter (EADC)

6.42.1 Overview

The chip contains one 12-bit successive approximation analog-to-digital converter (SAR EADC converter) with 24 external input channels and 7 internal channels. The EADC converter can be started by software trigger, EPWM0/1 triggers, PWM0/1 triggers, timer0~3 overflow pulse triggers, ADINT0 interrupt EOC (End of conversion) and ADINT1 interrupt EOC pulse trigger, external pin (EADC0_ST) input signal, and ACMP0~2 triggers.

6.42.2 Features

- Analog input voltage range: 0~ V_{REF} (Max to 3.6V)
- Reference voltage from V_{REF} pin
- 12-bit resolution and 10-bit accuracy is guaranteed
- Up to 24 single-end analog external input channels or 8 pair differential analog input channels
- Up to 7 internal channels, they are AN_OPA0_OUT, AN_OPA1_OUT, AN_OPA2_OUT, $AV_{DD}/4$, band-gap voltage (V_{BG}), temperature sensor (V_{TEMP}), and battery power/4 ($V_{BAT}/4$).
- Four EADC interrupts (ADINT0~3) with individual interrupt vector addresses for EADC
- Maximum EADC clock frequency is 72 MHz for EADC
- Up to 3.42 MSPS conversion rate for EADC
- Supports calibration function and calibration interrupt
- Supports internal reference voltage V_{REF}: 1.6V, 2V, 2.5V and 3V.
- Supports power-down mode
- Up to 31 sample modules
  - Sample module 0~26 is configurable for EADC converter channel (EADC_CH0~23, AN_OPA0_OUT, AN_OPA1_OUT, AN_OPA2_OUT) and trigger source for each EADC.
  - Sample module 27~30 are fixed for input sources $AV_{DD}/4$, band-gap voltage, temperature sensor, and battery power/4 ($V_{BAT}/4$), respectively.
  - Double buffer for sample control logic module 0~3.
  - Configurable sampling time for each sample module.
  - Conversion results are held in 31 data registers with valid and overrun indicators.
- Averaging (2^n times, n=0~8) to support up to 12-bit result and over-sampling, or called Accumulation, (2^n times, n=0~8) to support up to 16-bit result
- Supports conversion results left-alignment
- Any EADC conversion of each EADC can be started by:
  - Write 1 to SWTRGn (EADC_SWTRG[n], n = 0~30)
  - External pin EADC0_ST trigger
  - Timer0~3 overflow pulse triggers
  - ADINT0 and ADINT1 interrupt EOC (End of conversion) pulse triggers
  - EPWM/PWM triggers

- ADINT0 and ADINT1 interrupt SOC (Start of conversion) pulse trigger with delay counter
- ACMP0, ACMP1, and ACMP2 interrupt pulse triggers

- Supports PDMA transfer
- Conversion Result Monitor by Compare Mode

6.42.3 Block Diagram

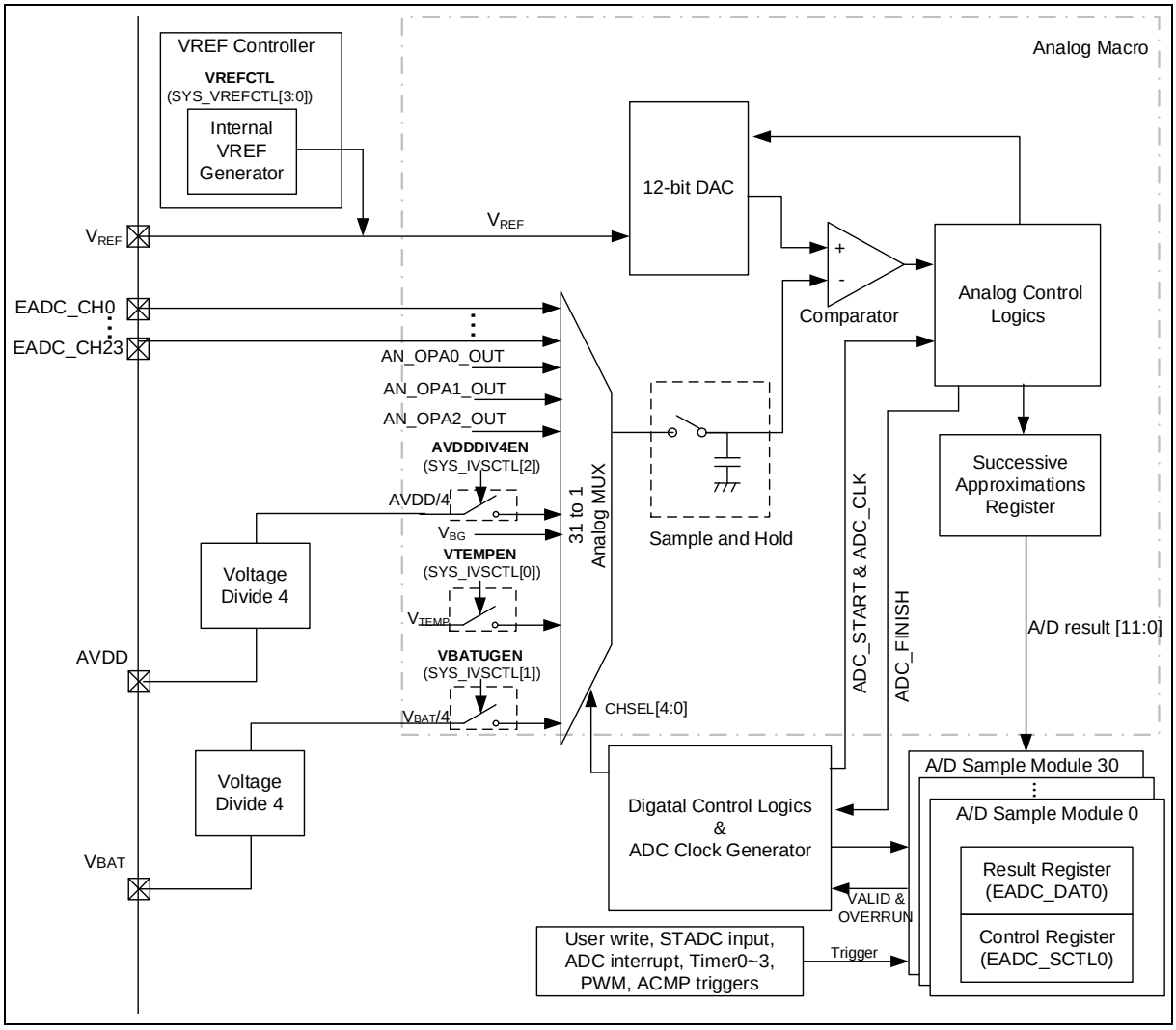


Figure 6.42-1 EADC Converter Block Diagram

6.42.4 Basic Configuration

- EADC0 Clock Source Configuration
  - Select the clock divider number on EADCDIV (CLK_CLKDIV0[23:16])
  - Enable EADC peripheral clock in EADC0CKEN (CLK_APBCLK0[28]).
- Reset Configuration
  - Reset EADC controller in EADCRST (SYS_IPRST1[28]).
- Control Select Setting

- Set ADCCSEL (SYS_IVSCTL[7]) to 0 to make EADC0 can controll the ADC macro.

6.42.5 Functional Description

The EADC controller consists of a 31 channel analog switch, 31 sample modules and a 12-bit successive approximation analog-to-digital converter. The EADC operation is based on sample module 0~30, and sample module 0~26 has its configuration to decide which trigger source to start the conversion, which channel to convert. Sample module 0~26 can be configured to EADC0_CH0~23 channel, AN_OPA0_OUT, AN_OPA1_OUT, AN_OPA2_OUT and different trigger source. It provides user a flexible means to get the over-sampling results. The sample module 0~3 and sample module 4~26 are shown as follows. Note that sample module 27~30 are fixed for internal channel AV_{DD}/4, band-gap voltage, (V_{BG}), temperature sensor, and battery power/4 (V_{BAT}/4), respectively.

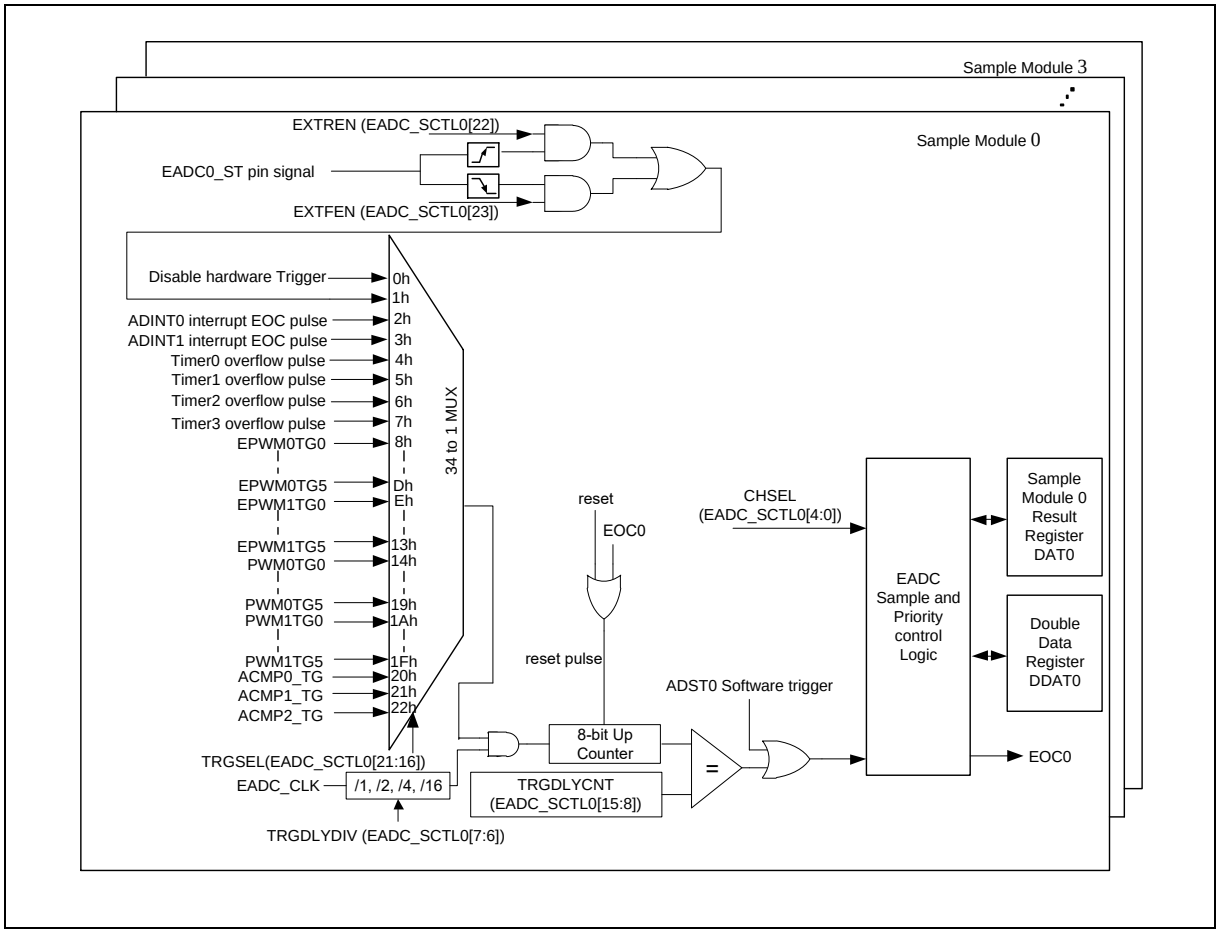


Figure 6.42-2 Sample Module 0~3 Block Diagram

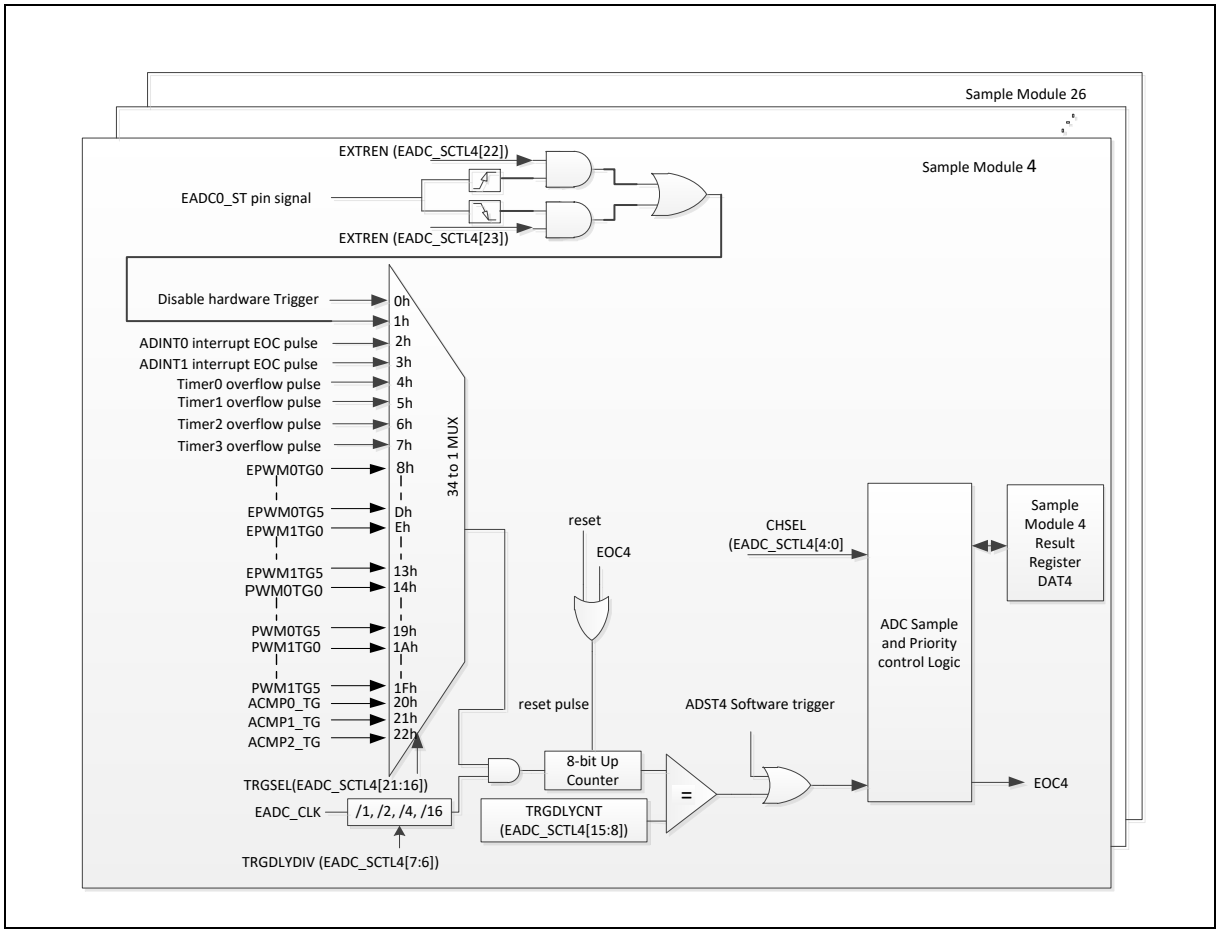


Figure 6.42-3 Sample Module 4~26 Block Diagram

Sample module 27~30 can convert internal channel ($AV_{DD}/4$, V_{BG}, V_{TEMP}, $V_{BAT}/4$) and can be triggered by user write SWTRGn (EADC_SWTRG[n], n = 27~30). Figure 6.42-4 shows the sample module 27~30.

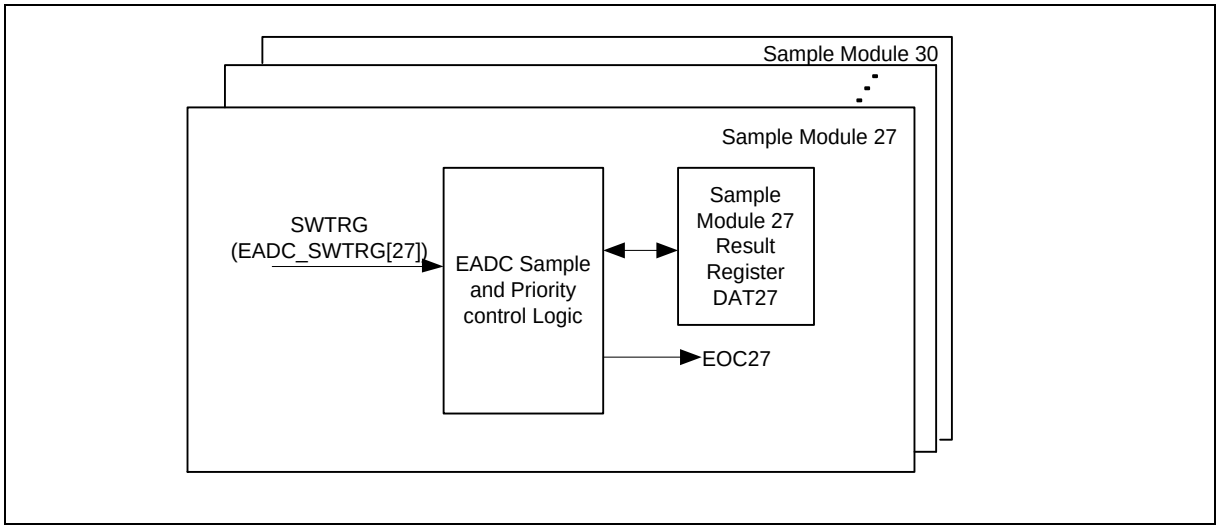


Figure 6.42-4 Sample Module 27~30 Block Diagram

The EADC conversion trigger sources in sample module 0~26 are listed below:

- Write 1 to SWTRGn (EADC_SWTRG[n], n = 0~26)
- External pin EADC0_ST respectively
- Timer0~3 overflow pulse triggers
- ADINT0, ADINT1 EADC interrupt EOC (End of conversion) pulse triggers
- EPWM/PWM triggers
- ACMP triggers

The ADINT0 or ADINT1 interrupt pulses are generated whenever the specific sample module EADC EOC (End of conversion) pulse is generated. ADINT0 or ADINT1 interrupt pulse triggers can be fed back to trigger another EADC conversion, and is useful if a continuous scan conversion is needed.

6.42.5.1 EADC Clock Generator

The maximum EADC clock frequency is up to 72 MHz and the maximum sampling rate is up to 3.42 MSPS.

The clock control of EADC is shown as Figure 6.42-5. The EADC has three clock sources selected by EADC0SEL(CLK_CLKSEL0[11:10]). The EADC clock frequency is divided by an 8-bit pre-scalar with the following formula:

EADC0 clock frequency = (EADC0 clock source frequency) / (EADCDIV (CKL_CLKDIV0[23:16])+1)

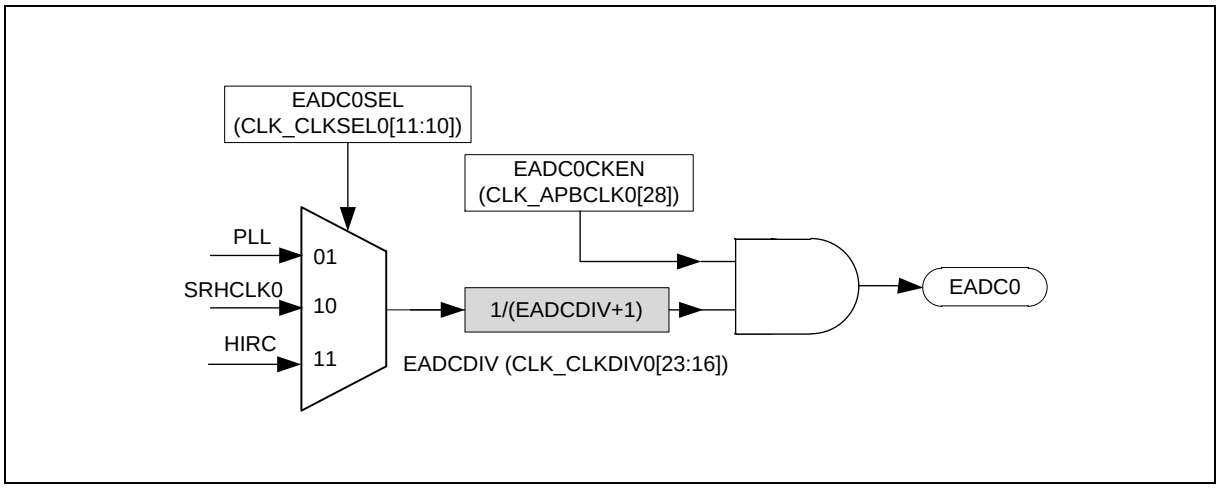


Figure 6.42-5 EADC Clock Control

6.42.5.2 EADC Software Trigger Mode

When a EADC conversion is performed on the sample module specified single channel, the operations are as follows:

1. EADC conversion is started when the SWTRGn (EADC_SWTRG[n], n=0~30) is set to 1 by user or other trigger inputs.
2. When EADC conversion is finished, the 12-bit result is stored in the EADC data register EADC_DATn (n=0~30) corresponding to the sample module.
3. On completion of conversion, the ADIFn (EADC_STATUS2[3:0], n=0~3) is set to 1 and EADC interrupt (ADINTn, n=0~3) is requested if the ADCIENn (EADC_CTL[5:2], n=0~3) bit is set to 1.
4. When EADC conversion ends, the SWTRGn (n=0~30) bit is automatically cleared to 0 and the EADC converter will do another pending conversion.

5. Take EADC0 for example. The timing diagram of a conversion cycle is shown in Figure 6.42-6.

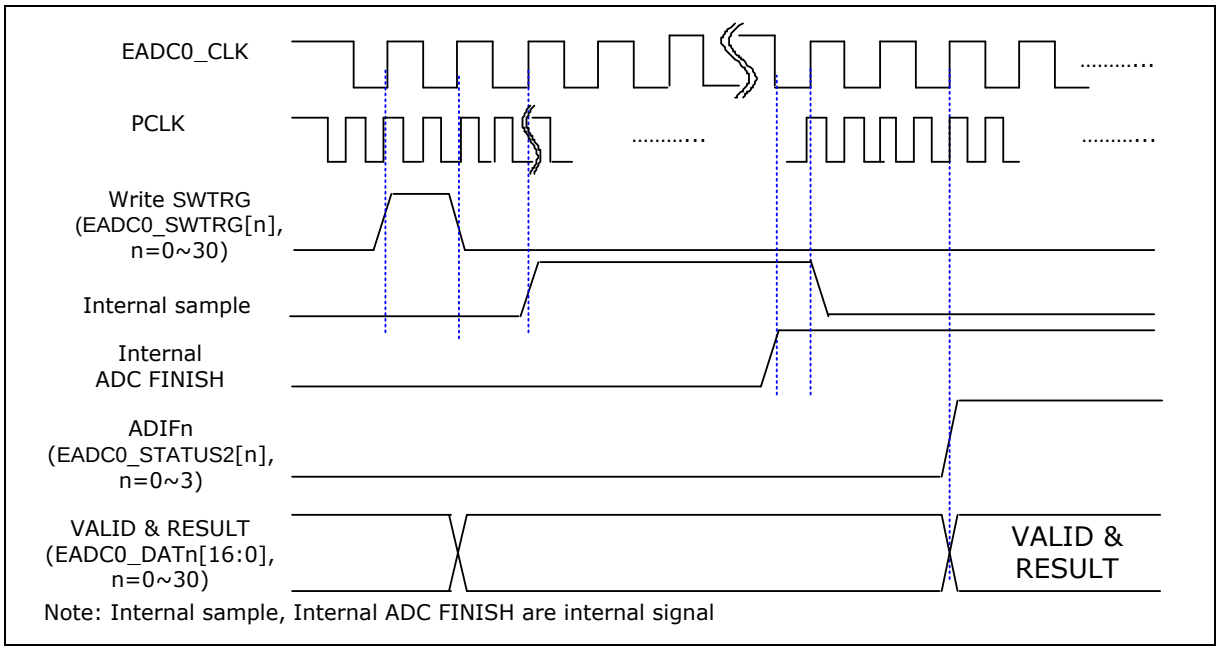


Figure 6.42-6 Example EADC Conversion Timing Diagram, n=0~30

If more than one sample module is enabled to convert analog signal, the sample module specified channel with highest priority is firstly converted and other enabled sample module will be pended. The lower number sample module has higher priority. The sample module 0 is highest priority and the sample module 30 is lowest priority.

6.42.5.3 EADC Conversion Priority

There is a priority group converter for determining the conversion order when multiple sample module trigger flags are set at the same time. Sample module with lower number has higher priority than the higher number sample module. The priority of sample module is shown as Figure 6.42-7. When more than one Sample Module are triggered at the same time, the Sample Module with lower number will start to convert first. The other Sample Module will be in the queue and the corresponding pending flag STPF(EADC_PENDSTS[n], n=0~30) are set to 1 by HW. After the Sample Module finish the conversion, STPF(EADC_PENDSTS[n], n=0~30) will be set to 0 automatically. If the Sample Module which is in the queue is triggered once more, the corresponding Overrun Flag SPOVF(EADC_OVSTS[n], n=0~30) will be set to 1 by HW.

For example, the Sample Module 0, 2, 3, 5 are triggered simultaneously. The input channel of Sample Module 0 will be converted first. Sample Module 2, 3, 5 will be suspended and STPF (EADC_PENDSTS[2], EADC_PENDSTS [3], EADC_PENDSTS [5]) will be set to 1. If Sample Module 5 is trigger once more in the same time, SPOVF(EADC_OVSTS[5]) will be set to 1.

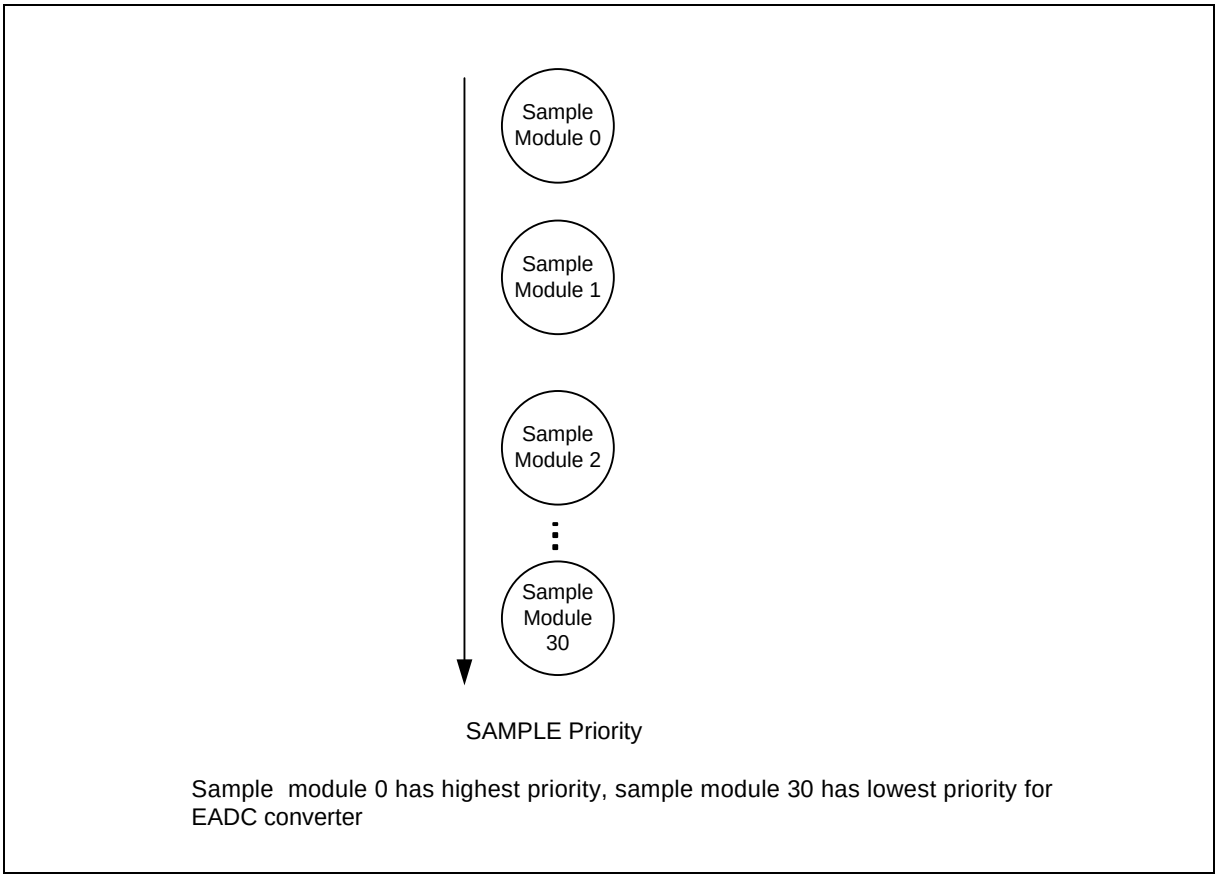


Figure 6.42-7 Sample Module Conversion Priority Arbitrator Diagram

6.42.5.4 Maximum Sampling Frequency Conversion by Software Trigger

If user needs to get maximum sampling frequency, the conversion needs to be executed by the condition as: multiple sample modules, triggered by software, and triggered repeatedly during the last conversion. An example of continuous scan is as follows:

1. Use Module 0~15 to carry out successive conversion. Set CHSEL (EADC_SCTL0~15[4:0]) and Set EXTSMPT (EADC_SCTL0~15[31:24]) and TRGDLYCNT (EADC_SCTL0~15[15:8]) as 0x00 to minimize the sampling time.
2. Set SWTRG (EADC_SWTRG[30:0]) as 0x3fffff to trigger Module 0~15.
3. Wait CURSPL (EADC_STATUS3[4:0]) changes to 0xf which means Module 0~14 have been executed and Module 15 is in the process. Set SWTRG (EADC_SWTRG[21:0]) as 0x7fff to trigger Module 0~15 again for next round.
4. Wait CURSPL (EADC_STATUS3[4:0]) changes to 0x1, set SWTRG (EADC_SWTRG[21:0]) as 0x8000 to trigger Module 15.
5. Repeat Step 3~4 to continue the conversion.

6.42.5.5 EADC Sample Module End of Conversion Interrupt Operation

There are 4 EADC interrupts ADINT0~3, and each of these interrupts has its own interrupt vector address and can be configured to set multiple sample module EOC pulse (sample module 0~30 End of conversion pulses) as its interrupt trigger source. Figure 6.42-8 shows the control logic of interrupts. Take ADINT0 as an example, when ADCIEN0 (EADC_CTL[2]) = 1 and SPLIEn (EADC_INTSRC0[n]) = 1 (n=0~30), the specific module EOC (End of conversion) pulses will set flag ADIF0 (EADC_STATUS2[0]) as 1 and interrupt (ADINT0) will be asserted either.

The interrupt pulses (ADINT0/1) are generated whenever the specific sample module EADC EOC pulse is generated. It also can be the sample module conversion trigger sources, and user can use it to do the EADC continuous scan conversion.

The example of continuous scan triggered by interrupt is as follows:

1. If EADC sample module 2 EOC2 pulse is selected as ADINT0 interrupt trigger SPLIE2 (EADC_INTSRC0[2]) = 1 and ADINT0 is selected as sample module 0, 1, 2 hardware conversion trigger.
2. Set software trigger SWTRG2 (EADC_SWTRG[2]) to 1 to start a sample module 2 EADC conversion, after the conversion completes, it generates an EOC2 pulse signal and ADINT0 interrupt pulse at end of sample module 2 EADC conversion, ADINT0 interrupt pulse will trigger the sample module 0, 1, 2 to start the EADC conversions.
3. ADINT0 interrupt pulse repeats to trigger sample module 0, 1, 2 EADC conversions automatically.
4. Clear TRGSEL (EADC_SCTL2[21:16]) to 0 to disable sample module 2 ADINT0 interrupt pulse hardware trigger, if needs to stop the continuous scan.

Note: The average conversion cycles of continuous scan triggered by interrupt is 20 EADC_CLK.

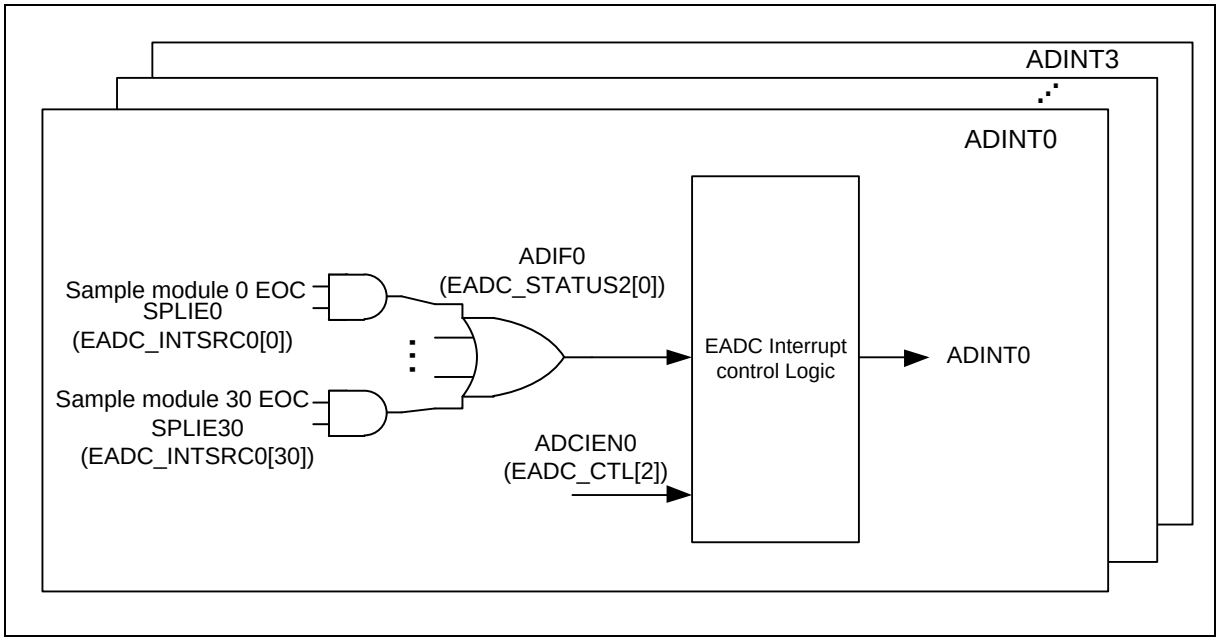


Figure 6.42-8 Specific Sample Module EADC EOC Signal for ADINT0~3 Interrupt

6.42.5.6 EADC Trigger by Timer Trigger and External Pin

There are 4 Timer trigger sources and external pin which can configure sample module 0~26 to trigger EADC start when Timer overflow occurs.

6.42.5.7 EADC Start Synchronous with EPWM/PWM Trigger

Besides user start, ADINT0/1 interrupt pulse, external pin EADC0_ST, EADC1_ST, EADC2_ST and Timer0~3 overflow pulse to start EADC conversion, this device has new feature to allow EPWM/BPWM channels to trigger the EADC start. User may configure EPWM/BPWM trigger types: rising, falling EPWM/BPWM edge or center point of EPWM/BPWM (center-aligned mode only) to trigger EADC start. The device also allows user to configure the amount of delay period to EADC start after hardware detected the external trigger. User can configure the trigger delay time by setting TRGDLYCNT (EADC_SCTLn[15:8], n=0~26) and TRGDLYDIV (EADC_SCTLn[7:6], n=0~26). Figure 6.42-9 shows the programmable delay time for EPWM/PWM-triggered EADC start conversion.

Figure 6.42-10 shows the programmable delay time for other trigger source.

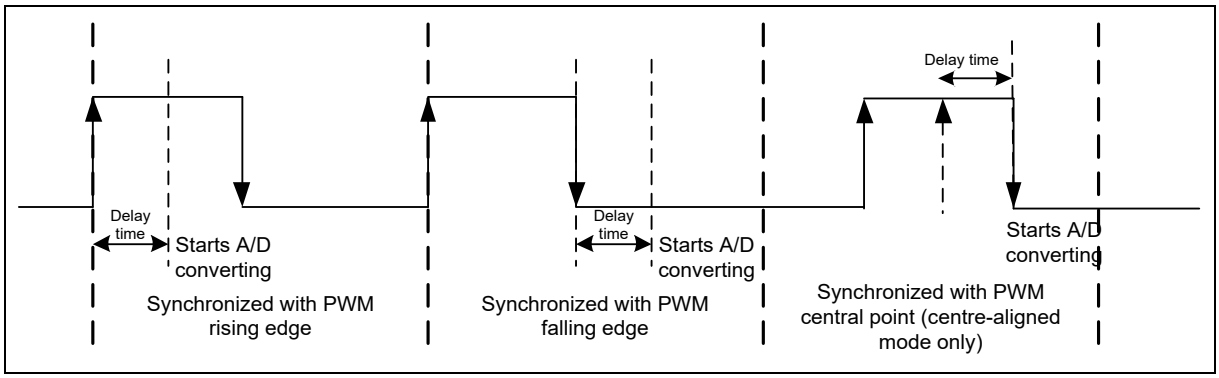


Figure 6.42-9 EPWM-triggered EADC Start Conversion

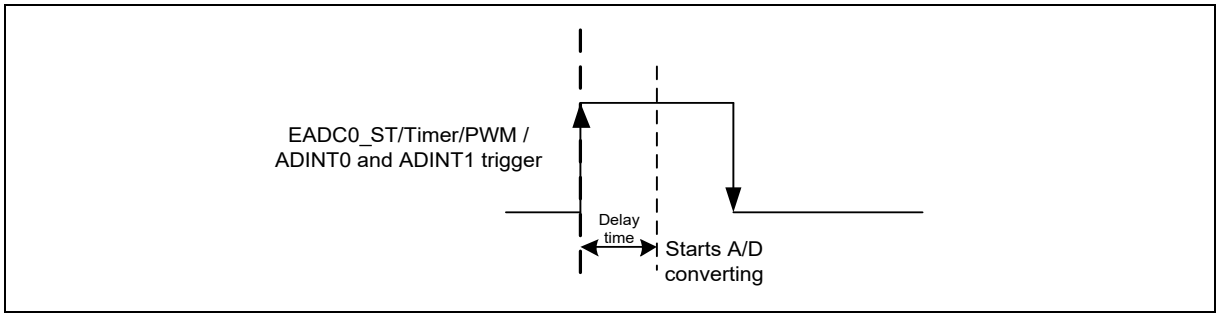


Figure 6.42-10 External triggered EADC Start Conversion

6.42.5.8 EADC Conversion Time and External Trigger

The EADC converter samples the analog input when EADC conversion start delay time (Td) has passed after SWTRGn (EADC_SWTRG[n], n=0~30) is set to 1, then start conversion. The maximum delay time from user write SWTRGn to EADC start sampling analog input time is four EADC clock cycles that operate for synchronization and control pre-processing. The start delay time is shown in Figure 6.42-11.

For example, EADC conversion can be triggered by external pin EADC0_ST request for EADC0. Setting the TRGSEL (EADC_SCTLn[21:16], n=0~26) to 0x01 is to select external trigger input from the EADC0_ST pin. User can set EXTFEN (EADC_SCTLn[23], n=0~26) and EXTREN (EADC_SCTLn[22], n=0~26) to enable pin EADC0_ST trigger condition is falling or rising edge. There is a de-bounce circuit to detect falling or rising edge. If rising edge trigger condition is selected, the low state must be kept at least 2 PCLK cycles and the following high state must be kept at least 3 PCLK cycles. If falling edge trigger condition is selected, the high state must be kept at least 2 PCLK cycles and the following low state must be kept at least 3 PCLK cycles. Pulse that is shorter than this specification will be ignored. The external trigger timing is shown in Figure 6.42-12.

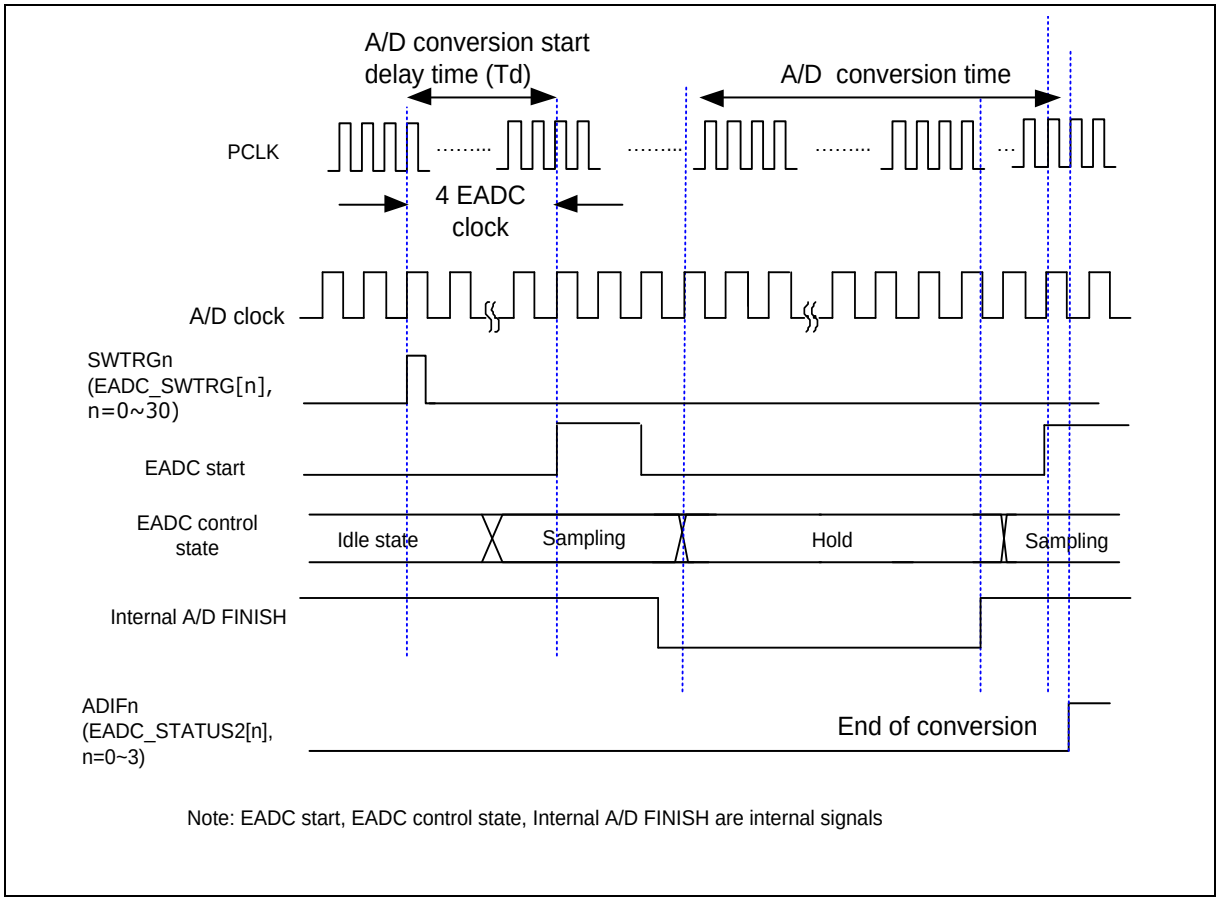


Figure 6.42-11 Conversion Start Delay Timing Diagram

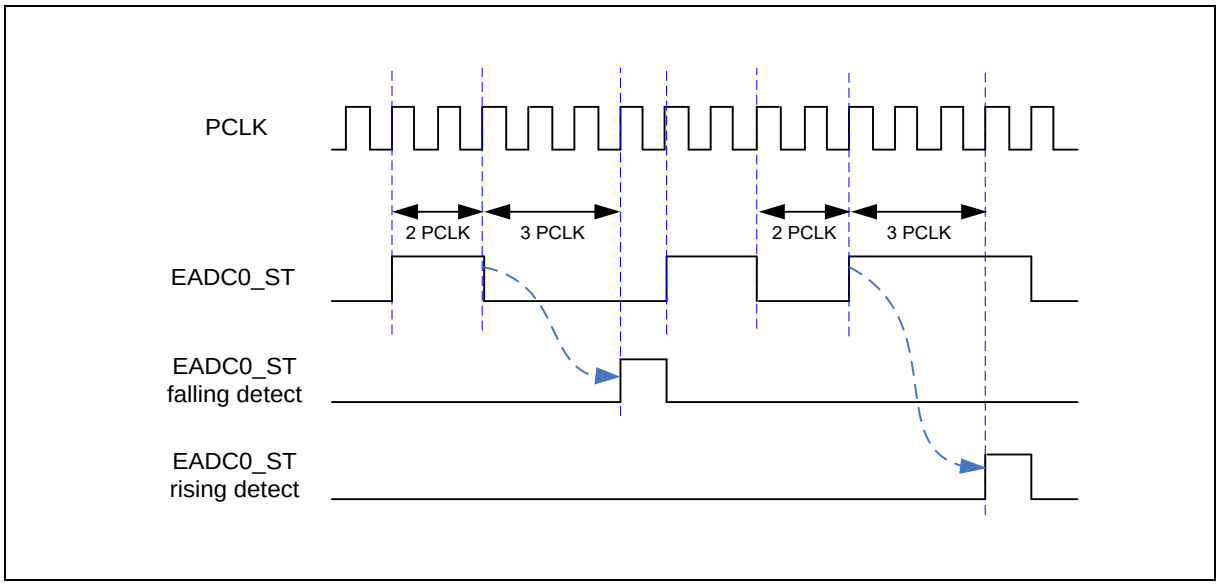


Figure 6.42-12 EADC0_ST De-bounce Timing Diagram

6.42.5.9 EADC Extend Sampling Time

When EADC operates at high EADC clock rate, the sampling time of analog input voltage may not be enough if the analog channel has heavy loading to cause fully charge time is longer. User can set extend

sampling time by writing EXTSMPT (EADC_SCTLn[31:24], n=0~30) for each sample module. The EADC extend sampling time is present between EADC controller judge which channel to be converting and EADC start to conversion. The range of setting extend sampling time is from 0~255 EADC clock. For example, EADC sample module2 is configured with extend sampling time, as shown in Figure 6.42-13.

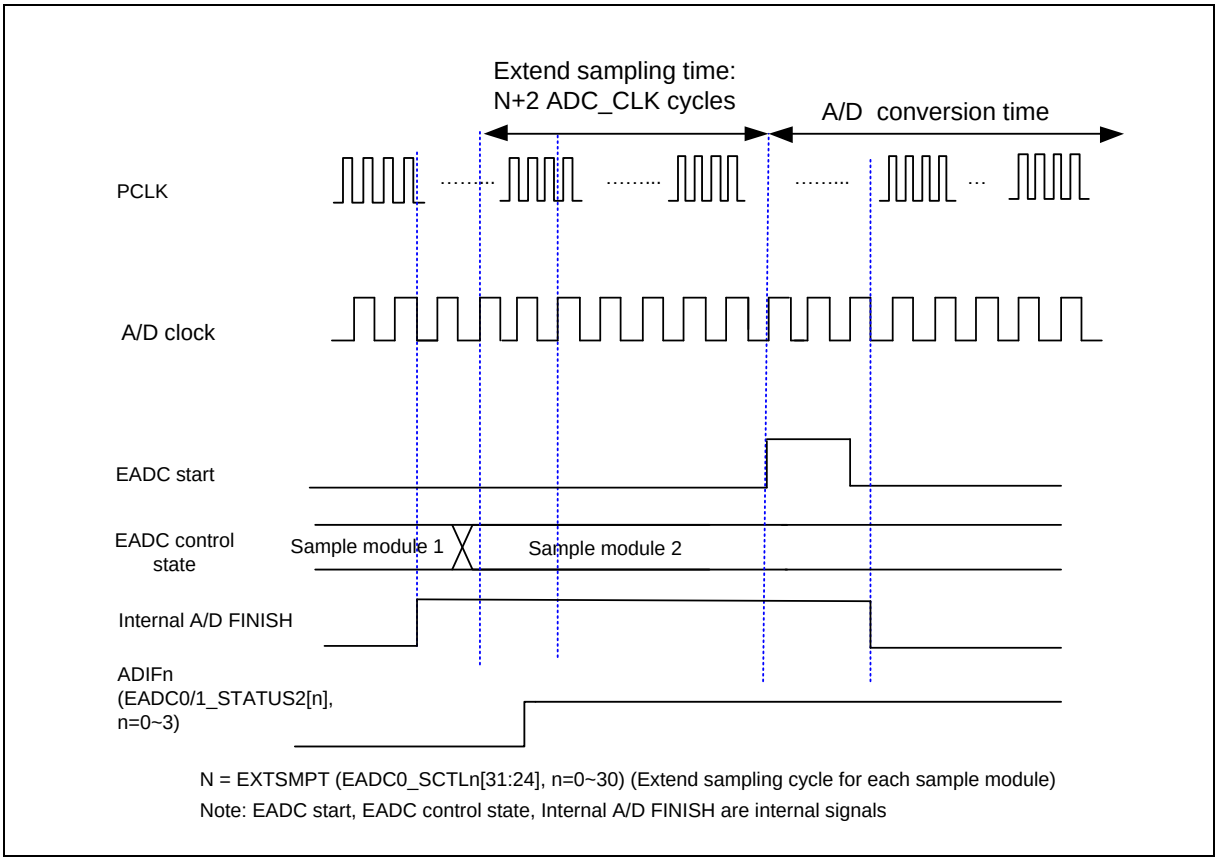


Figure 6.42-13 EADC Extend Sampling Timing Diagram

6.42.5.10 Oversampling Mode

The function provides multiple consecutive conversion results to be accumulated to a final conversion result. Setting ACU (EADC_MnCTL1[7:4], n=0~26) determine whcich sample module and the number of samples to be accumulated. It is noted that when more than 16 samples to be accumulated, the final result after accumulating will exceed the 16-bit RESULT(EADC_DATn [0:15], n=0~26) register size. In this case, the result is right shifted automatically to fit within the appropriate register size. It is noted that accumulation function is only effective without data left alignment.

ACU (EADC_MnCTL1 [7:4]), n=0~26	Number of Samples To Be Accumulated	Right Shift Division Factor	Final Result Precision
0x0	1	1	12 bits
0x1	2	1	13 bits
0x2	4	1	14 bits
0x3	8	1	15 bits
0x4	16	1	16 bits

0x5	32	2	16 bits
0x6	64	4	16 bits
0x7	128	8	16 bits
0x8	256	16	16 bits

Table 6.42-1 Setting of Accumulation and Conversion Result Precision

6.42.5.11 Averaging Mode

The function increasing the sample accuracy with reduced sampling rate and the feature is suitable in noisy operating environment for getting more stable average conversion results. According to the accumulated final result in 6.42.5.10, setting AVG (EADC_MnCTL1[1]) can enable averaging function as shown in Table 6.42-2. The automatic division factor is decided by the number of accumulated samples and the final result will get 12 bits precision. It is noted that averaging is only effective when accumulating two more samples without data left alignment.

AVG EADC_MnCTL1 [1]	ACU EADC_MnCTL1 [7:4]	Number of Samples To Be Accumulated	Automatic Division Factor	Final Result Precision
0x1	0x0	1	1	12 bits
0x1	0x1	2	2	12 bits
0x1	0x2	4	4	12 bits
0x1	0x3	8	8	12 bits
0x1	0x4	16	16	12 bits
0x1	0x5	32	32	12 bits
0x1	0x6	64	64	12 bits
0x1	0x7	128	128	12 bits
0x1	0x8	256	256	12 bits

Table 6.42-2 Setting of Averaging and Conversion Result Precision

6.42.5.12 Conversion Result Monitor by Compare Mode

The EADC controller provides four sets of compare registers EADC_CMP0 ~ EADC_CMP3 to monitor a maximum of four specified sample module 0~30 conversion results from EADC conversion module, as shown in Figure 6.42-14. User can select which sample module result to be monitored by set CMPSPL (EADC_CMPn[7:3], n =0~3) and CMPCOND (EADC_CMPn[2], where n =0~3) is used to check conversion result is less than specify value or greater than (equal to) value specified in CMPDAT (EADC_CMPn[27:16], where n =0~3). When the conversion of the sample module specified by CMPSPL is completed, the comparing action will be triggered one time automatically. When the compare result meets the compare condition, the internal compare match counter will increase 1. If the compare result does not meet the condition, the compare match counter will reset to 0. When counter value reach the setting of (CMPMCNT (EADC_CMPn[11:8])+1, where n =0~3) then ADCMPFn (EADC_STATUS2[7:4], where n =0~3) bit will be set to 1, if ADCMPIE (EADC_CMPn[1] , n =0~3) is set then an ADINT3 interrupt request is generated. User can use it to monitor the external analog input pin voltage transition. Detailed logics diagram is shown in Figure 6.42-14.

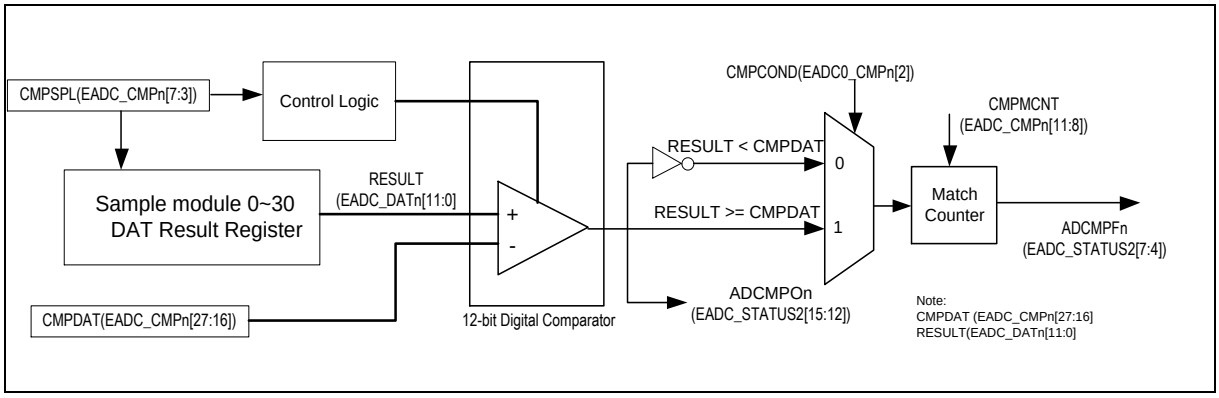


Figure 6.42-14 EADC Conversion Result Monitor Logics Diagram

The EADC controller supports a window compare mode. User can set CMPWEN (EADC_CMP0[15]/ EADC_CMP2[15]) to enable this function. If user enables this function, EADCMPF0 (EADC_STATUS2[4]) will be set when both EADC_CMP0 and EADC_CMP1 compared condition matched. EADCMPF2 (EADC_STATUS2[6]) will be set when both EADC_CMP2 and EADC_CMP3 compared condition matched.

6.42.5.13 Differential Mode

The EADC controller supports analog fully-differential mode. A fully-differential SAR converts the differential voltage across its inputs; however, in this case both inputs change dynamically and are complementary or inverted from each other. In this case, the VINP and VINM pins must be driven 180° out of phase with respect to each other, centered on a fixed common mode voltage, for example $V_{REF}/2$. Fully-differential EADCs have an input voltage common-mode (VCM) range specification. VCM is defined as the average voltage between the inputs: VCM = (VINP + VINM) / 2. Most fully-differential input SAR EADCs prohibit the input common-mode voltage from varying more than approximately 10 percent beyond the mid-scale input (V_{REF} /2).

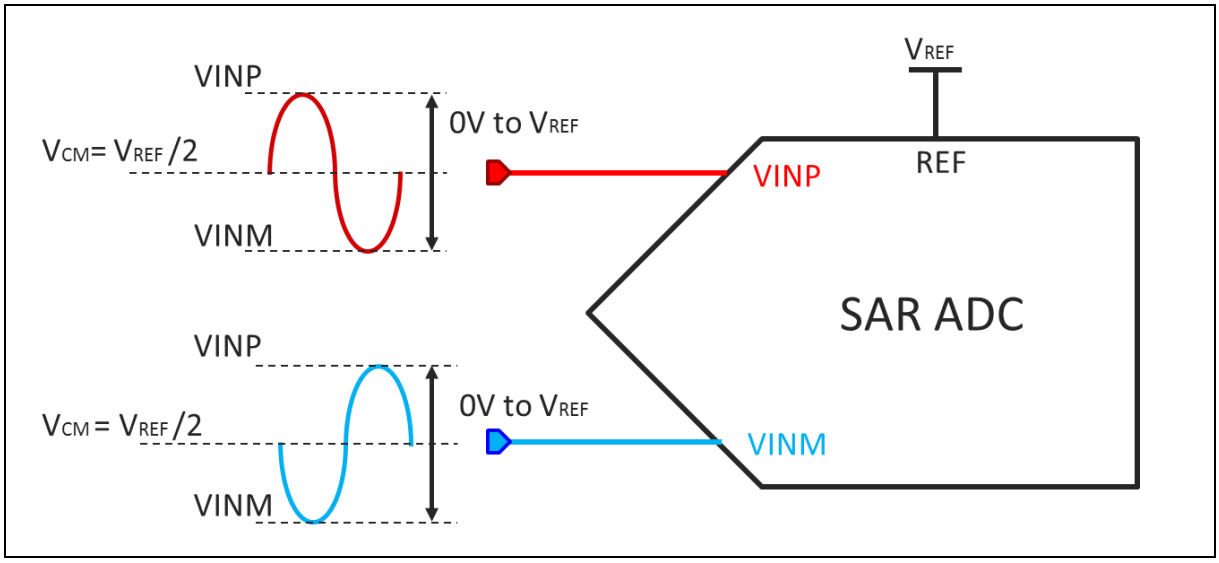


Figure 6.42-15 Input of Fully-Differential EADC

If user enables DIFFEN (EADC_CTL[8]), the differential mode will enable. The pair of analog input channel is as Table 6.42-3

Differential Analog Input Paired	ADC Analog Input

Channel	V_{plus}	V_{minus}
0	EADC_CH0	EADC_CH1
1	EADC_CH2	EADC_CH3
2	EADC_CH4	EADC_CH5
3	EADC_CH6	EADC_CH7
4	EADC_CH8	EADC_CH9
5	EADC_CH10	EADC_CH11
6	EADC_CH12	EADC_CH13
7	EADC_CH14	EADC_CH15

Table 6.42-3 EADC Differential Model Channel Selection

In differential analog input mode, only the even number of the two corresponding channels needs to be enabled in CHSEL (EADC_SCTLn[4:0]). The conversion result will be placed to the corresponding data register of the enabled channel.

6.42.5.14 Double Buffer Mode

The EADC controller supports a double buffer mode in sample module 0~3. If user enable DBMEN (EADC_MnCTL[20], n=0~3), the double buffer mode will enable. In double buffer mode, after first time EADC convert finish, the VALID (EADC_DATn[17], n=0~3) will set to high, but VALID (EADC_DDATn[17], n=0~3) will keep low. And the second time EADC converts finish, VALID (EADC_DDATn[17], n=0~3) will set to high either. Then, user can get the EADC results from EADC_DATn and EADC_DDATn register.

6.42.5.15 PDMA Request

The EADC controller supports PDMA. PDMA could service each channel when corresponding channel PDMA transfer enable bit, PDMATEN (EADC_PDMACTL[30:0]), is activated. For example, user can enable PDMATEN for specific channels and configure PDMA channel's source address as CURDAT(EADC_CURDAT[30:0]). After enabling PDMATEN and PDMA channel, if any VALID (EADC_DATn[17],n=0~30) is high, EADC controller will send request to PDMA and PDMA will read EADC_CURDAT to get result. The EADC_CURDAT register is a shadow register of highest priority EADC_DAT register. The lower number sample module is higher priority. After PDMA read EADC_CURDAT register, the VAILD of the EADC_DAT register will be automatically cleared.

6.42.5.16 Interrupt Sources

The EADC converter generates ADIFn (EADC_STATUS2[3:0], n=0~3) at the start of conversion or the end of conversion decide by INTPOS (EADC_SCTLn[5], n=0~26). If EADCIENn (EADC_CTL[5:2], n=0~3) is set then conversion end interrupt request ADINTn (n=0~3) is generated. The controller of interrupts is shown as Figure 6.42-16.

It is noted that calibration interrupt CALINT is also shown in Figure 6.42-16.

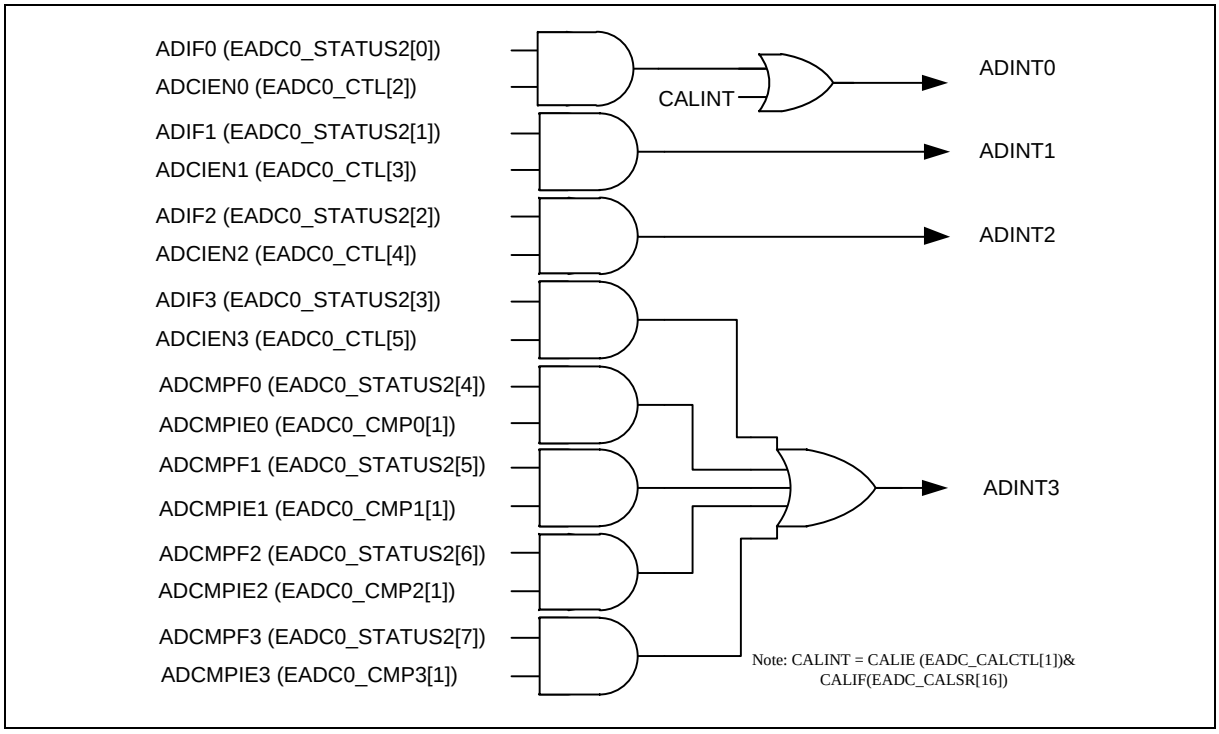


Figure 6.42-16 EADC Controller Interrupts

6.42.5.17 Calibration Mode

To decrease the effect of electrical random noise, the calibration mode performs an offset and mismatch measurement cycles. Afterwards, in normal operation mode, the calibration engine applies to the capacitor array, so that the offset and mismatch are removed. If chip power off, calibration function should be executed again.

In Calibration Mode:

1. Set CAL(EADC_CALCTL[0]) bit to 1.
2. After 3 ADC_CLK when CAL is set to 1, calibration is started.
3. After calibration is finished, the flag CALIF(EADC_CALSR[16]) will be set to 1. If CALIE (EADC_CALCTL[1]) is set, the calibration interrupt will be generated.
4. After calibration is finished, the mode will become Normal Mode and CAL is cleared by hardware automatically.

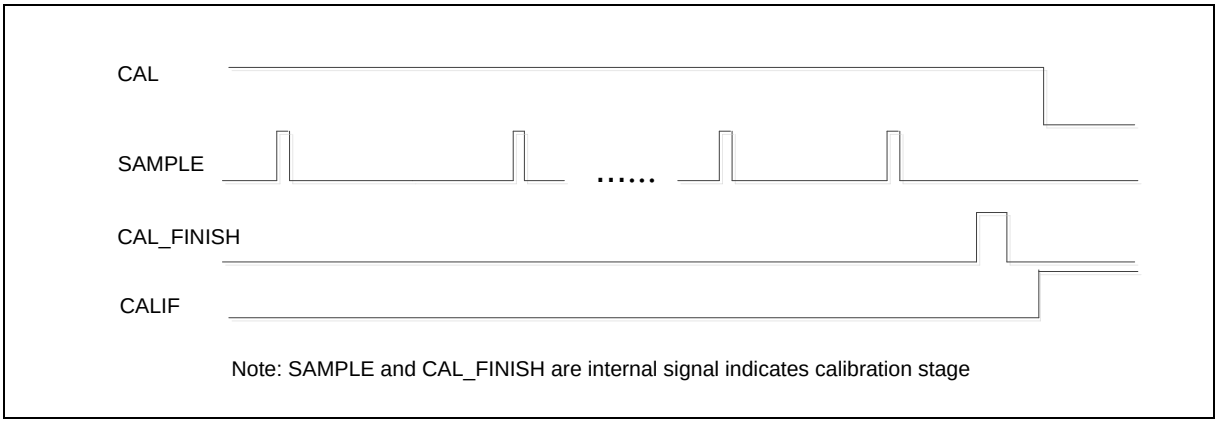


Figure 6.42-17 Calibration Mode with 16 Times Average

6.42.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
EADC Base Address: **EADC_BA = 0x4004_3000**				
EADC_DAT0	EADC_BA+0x00	R	EADC Data Register 0 for Sample Module 0	0x0000_0000
EADC_DAT1	EADC_BA+0x04	R	EADC Data Register 1 for Sample Module 1	0x0000_0000
EADC_DAT2	EADC_BA+0x08	R	EADC Data Register 2 for Sample Module 2	0x0000_0000
EADC_DAT3	EADC_BA+0x0C	R	EADC Data Register 3 for Sample Module 3	0x0000_0000
EADC_DAT4	EADC_BA+0x10	R	EADC Data Register 4 for Sample Module 4	0x0000_0000
EADC_DAT5	EADC_BA+0x14	R	EADC Data Register 5 for Sample Module 5	0x0000_0000
EADC_DAT6	EADC_BA+0x18	R	EADC Data Register 6 for Sample Module 6	0x0000_0000
EADC_DAT7	EADC_BA+0x1C	R	EADC Data Register 7 for Sample Module 7	0x0000_0000
EADC_DAT8	EADC_BA+0x20	R	EADC Data Register 8 for Sample Module 8	0x0000_0000
EADC_DAT9	EADC_BA+0x24	R	EADC Data Register 9 for Sample Module 9	0x0000_0000
EADC_DAT10	EADC_BA+0x28	R	EADC Data Register 10 for Sample Module 10	0x0000_0000
EADC_DAT11	EADC_BA+0x2C	R	EADC Data Register 11 for Sample Module 11	0x0000_0000
EADC_DAT12	EADC_BA+0x30	R	EADC Data Register 12 for Sample Module 12	0x0000_0000
EADC_DAT13	EADC_BA+0x34	R	EADC Data Register 13 for Sample Module 13	0x0000_0000
EADC_DAT14	EADC_BA+0x38	R	EADC Data Register 14 for Sample Module 14	0x0000_0000
EADC_DAT15	EADC_BA+0x3C	R	EADC Data Register 15 for Sample Module 15	0x0000_0000
EADC_DAT16	EADC_BA+0x40	R	EADC Data Register 16 for Sample Module 16	0x0000_0000
EADC_DAT17	EADC_BA+0x44	R	EADC Data Register 17 for Sample Module 17	0x0000_0000
EADC_DAT18	EADC_BA+0x48	R	EADC Data Register 18 for Sample Module 18	0x0000_0000
EADC_CURDAT	EADC_BA+0x4C	R	EADC PDMA Current Transfer Data Register	0x0000_0000
EADC_CTL	EADC_BA+0x50	R/W	EADC Control Register	0x0000_0000
EADC_SWTRG	EADC_BA+0x54	W	EADC Sample Module Software Start Register	0x0000_0000
EADC_PENDSTS	EADC_BA+0x58	R/W	EADC Start of Conversion Pending Flag Register	0x0000_0000
EADC_OVSTS	EADC_BA+0x5C	R/W	EADC Sample Module Start of Conversion Overrun Flag Register	0x0000_0000
EADC_CTL1	EADC_BA+0x60	R/W	EADC Control1 Register	0x0000_0000
EADC_SCTL0	EADC_BA+0x80	R/W	EADC Sample Module 0 Control Register	0x0000_0014
EADC_SCTL1	EADC_BA+0x84	R/W	EADC Sample Module 1 Control Register	0x0000_0014

Register	Offset	R/W	Description	Reset Value
EADC_SCTL2	EADC_BA+0x88	R/W	EADC Sample Module 2 Control Register	0x0000_0014
EADC_SCTL3	EADC_BA+0x8C	R/W	EADC Sample Module 3 Control Register	0x0000_0014
EADC_SCTL4	EADC_BA+0x90	R/W	EADC Sample Module 4 Control Register	0x0000_0014
EADC_SCTL5	EADC_BA+0x94	R/W	EADC Sample Module 5 Control Register	0x0000_0014
EADC_SCTL6	EADC_BA+0x98	R/W	EADC Sample Module 6 Control Register	0x0000_0014
EADC_SCTL7	EADC_BA+0x9C	R/W	EADC Sample Module 7 Control Register	0x0000_0014
EADC_SCTL8	EADC_BA+0xA0	R/W	EADC Sample Module 8 Control Register	0x0000_0014
EADC_SCTL9	EADC_BA+0xA4	R/W	EADC Sample Module 9 Control Register	0x0000_0014
EADC_SCTL10	EADC_BA+0xA8	R/W	EADC Sample Module 10 Control Register	0x0000_0014
EADC_SCTL11	EADC_BA+0xAC	R/W	EADC Sample Module 11 Control Register	0x0000_0014
EADC_SCTL12	EADC_BA+0xB0	R/W	EADC Sample Module 12 Control Register	0x0000_0014
EADC_SCTL13	EADC_BA+0xB4	R/W	EADC Sample Module 13 Control Register	0x0000_0014
EADC_SCTL14	EADC_BA+0xB8	R/W	EADC Sample Module 14 Control Register	0x0000_0014
EADC_SCTL15	EADC_BA+0xBC	R/W	EADC Sample Module 15 Control Register	0x0000_0014
EADC_SCTL16	EADC_BA+0xC0	R/W	EADC Sample Module 16 Control Register	0x0000_0014
EADC_SCTL17	EADC_BA+0xC4	R/W	EADC Sample Module 17 Control Register	0x0000_0014
EADC_SCTL18	EADC_BA+0xC8	R/W	EADC Sample Module 18 Control Register	0x0000_0014
EADC_INTSRC0	EADC_BA+0xD0	R/W	EADC Interrupt 0 Source Enable Control Register.	0x0000_0000
EADC_INTSRC1	EADC_BA+0xD4	R/W	EADC Interrupt 1 Source Enable Control Register.	0x0000_0000
EADC_INTSRC2	EADC_BA+0xD8	R/W	EADC Interrupt 2 Source Enable Control Register.	0x0000_0000
EADC_INTSRC3	EADC_BA+0xDC	R/W	EADC Interrupt 3 Source Enable Control Register.	0x0000_0000
EADC_CMP0	EADC_BA+0xE0	R/W	EADC Result Compare Register 0	0x0000_0000
EADC_CMP1	EADC_BA+0xE4	R/W	EADC Result Compare Register 1	0x0000_0000
EADC_CMP2	EADC_BA+0xE8	R/W	EADC Result Compare Register 2	0x0000_0000
EADC_CMP3	EADC_BA+0xEC	R/W	EADC Result Compare Register 3	0x0000_0000
EADC_STATUS0	EADC_BA+0xF0	R	EADC Status Register 0	0x0000_0000
EADC_STATUS1	EADC_BA+0xF4	R	EADC Status Register 1	0x0000_0000
EADC_STATUS2	EADC_BA+0xF8	R/W	EADC Status Register 2	0x001B_0000
EADC_STATUS3	EADC_BA+0xFC	R	EADC Status Register 3	0x0000_001F
EADC_DDAT0	EADC_BA+0x100	R	EADC Double Data Register 0 for Sample Module 0	0x0000_0000

Register	Offset	R/W	Description	Reset Value
EADC_DDAT1	EADC_BA+0x104	R	EADC Double Data Register 1 for Sample Module 1	0x0000_0000
EADC_DDAT2	EADC_BA+0x108	R	EADC Double Data Register 2 for Sample Module 2	0x0000_0000
EADC_DDAT3	EADC_BA+0x10C	R	EADC Double Data Register 3 for Sample Module 3	0x0000_0000
EADC_CALCTL	EADC_BA+0x114	R/W	EADC Calibration Control Register	0x0000_0020
EADC_CALSR	EADC_BA+0x118	R/W	EADC Calibration Status Register	0x0000_0000
EADC_PDMACTL	EADC_BA+0x130	R/W	EADC PDMA Control Register	0x0000_0000
EADC_M0CTL1	EADC_BA+0x140	R/W	EADC Sample Module 0 Control Register 1	0x0000_0000
EADC_M1CTL1	EADC_BA+0x144	R/W	EADC Sample Module 1 Control Register 1	0x0000_0000
EADC_M2CTL1	EADC_BA+0x148	R/W	EADC Sample Module 2 Control Register 1	0x0000_0000
EADC_M3CTL1	EADC_BA+0x14C	R/W	EADC Sample Module 3 Control Register 1	0x0000_0000
EADC_M4CTL1	EADC_BA+0x150	R/W	EADC Sample Module 4 Control Register 1	0x0000_0000
EADC_M5CTL1	EADC_BA+0x154	R/W	EADC Sample Module 5 Control Register 1	0x0000_0000
EADC_M6CTL1	EADC_BA+0x158	R/W	EADC Sample Module 6 Control Register 1	0x0000_0000
EADC_M7CTL1	EADC_BA+0x15C	R/W	EADC Sample Module 7 Control Register 1	0x0000_0000
EADC_M8CTL1	EADC_BA+0x160	R/W	EADC Sample Module 8 Control Register 1	0x0000_0000
EADC_M9CTL1	EADC_BA+0x164	R/W	EADC Sample Module 9 Control Register 1	0x0000_0000
EADC_M10CTL1	EADC_BA+0x168	R/W	EADC Sample Module 10 Control Register 1	0x0000_0000
EADC_M11CTL1	EADC_BA+0x16C	R/W	EADC Sample Module 11 Control Register 1	0x0000_0000
EADC_M12CTL1	EADC_BA+0x170	R/W	EADC Sample Module 12 Control Register 1	0x0000_0000
EADC_M13CTL1	EADC_BA+0x174	R/W	EADC Sample Module 13 Control Register 1	0x0000_0000
EADC_M14CTL1	EADC_BA+0x178	R/W	EADC Sample Module 14 Control Register 1	0x0000_0000
EADC_M15CTL1	EADC_BA+0x17C	R/W	EADC Sample Module 15 Control Register 1	0x0000_0000
EADC_M16CTL1	EADC_BA+0x180	R/W	EADC Sample Module 16 Control Register 1	0x0000_0000
EADC_M17CTL1	EADC_BA+0x184	R/W	EADC Sample Module 17 Control Register 1	0x0000_0000
EADC_M18CTL1	EADC_BA+0x188	R/W	EADC Sample Module 18 Control Register 1	0x0000_0000
EADC_DAT19	EADC_BA+0x200	R	EADC Data Register 19 for Sample Module 19	0x0000_0000
EADC_DAT20	EADC_BA+0x204	R	EADC Data Register 20 for Sample Module 20	0x0000_0000
EADC_DAT21	EADC_BA+0x208	R	EADC Data Register 21 for Sample Module 21	0x0000_0000
EADC_DAT22	EADC_BA+0x20C	R	EADC Data Register 22 for Sample Module 22	0x0000_0000
EADC_DAT23	EADC_BA+0x210	R	EADC Data Register 23 for Sample Module 23	0x0000_0000

Register	Offset	R/W	Description	Reset Value
EADC_DAT24	EADC_BA+0x214	R	EADC Data Register 24 for Sample Module 24	0x0000_0000
EADC_DAT25	EADC_BA+0x218	R	EADC Data Register 25 for Sample Module 25	0x0000_0000
EADC_DAT26	EADC_BA+0x21C	R	EADC Data Register 26 for Sample Module 26	0x0000_0000
EADC_DAT27	EADC_BA+0x220	R	EADC Data Register 27 for Sample Module 27	0x0000_0000
EADC_DAT28	EADC_BA+0x224	R	EADC Data Register 28 for Sample Module 28	0x0000_0000
EADC_DAT29	EADC_BA+0x228	R	EADC Data Register 29 for Sample Module 29	0x0000_0000
EADC_DAT30	EADC_BA+0x22C	R	EADC Data Register 30 for Sample Module 30	0x0000_0000
EADC_SCTL19	EADC_BA+0x230	R/W	EADC Sample Module 19 Control Register	0x0000_0014
EADC_SCTL20	EADC_BA+0x234	R/W	EADC Sample Module 20 Control Register	0x0000_0014
EADC_SCTL21	EADC_BA+0x238	R/W	EADC Sample Module 21 Control Register	0x0000_0014
EADC_SCTL22	EADC_BA+0x23C	R/W	EADC Sample Module 22 Control Register	0x0000_0014
EADC_SCTL23	EADC_BA+0x240	R/W	EADC Sample Module 23 Control Register	0x0000_0014
EADC_SCTL24	EADC_BA+0x244	R/W	EADC Sample Module 24 Control Register	0x0000_0014
EADC_SCTL25	EADC_BA+0x248	R/W	EADC Sample Module 25 Control Register	0x0000_0014
EADC_SCTL26	EADC_BA+0x24C	R/W	EADC Sample Module 26 Control Register	0x0000_0014
EADC_SCTL27	EADC_BA+0x250	R/W	EADC Sample Module 27 Control Register	0x0000_0000
EADC_SCTL28	EADC_BA+0x254	R/W	EADC Sample Module 28 Control Register	0x0000_0000
EADC_SCTL29	EADC_BA+0x258	R/W	EADC Sample Module 29 Control Register	0x0000_0000
EADC_SCTL30	EADC_BA+0x25C	R/W	EADC Sample Module 30 Control Register	0x0000_0000
EADC_M19CTL1	EADC_BA+0x260	R/W	EADC Sample Module 19 Control Register 1	0x0000_0000
EADC_M20CTL1	EADC_BA+0x264	R/W	EADC Sample Module 20 Control Register 1	0x0000_0000
EADC_M21CTL1	EADC_BA+0x268	R/W	EADC Sample Module 21 Control Register 1	0x0000_0000
EADC_M22CTL1	EADC_BA+0x26C	R/W	EADC Sample Module 22 Control Register 1	0x0000_0000
EADC_M23CTL1	EADC_BA+0x270	R/W	EADC Sample Module 23 Control Register 1	0x0000_0000
EADC_M24CTL1	EADC_BA+0x274	R/W	EADC Sample Module 24 Control Register 1	0x0000_0000
EADC_M25CTL1	EADC_BA+0x278	R/W	EADC Sample Module 25 Control Register 1	0x0000_0000
EADC_M26CTL1	EADC_BA+0x27C	R/W	EADC Sample Module 26 Control Register 1	0x0000_0000
EADC_M27CTL1	EADC_BA+0x280	R/W	EADC Sample Module 27 Control Register 1	0x0000_0000
EADC_M28CTL1	EADC_BA+0x284	R/W	EADC Sample Module 28 Control Register 1	0x0000_0000
EADC_M29CTL1	EADC_BA+0x288	R/W	EADC Sample Module 29 Control Register 1	0x0000_0000

Register	Offset	R/W	Description	Reset Value
EADC_M30CTL1	EADC_BA+0x28C	R/W	EADC Sample Module 30 Control Register 1	0x0000_0000
EADC_TEST	EADC_BA+0xFF0	R/W	EADC Test Mode Control Register	0x0000_0000

6.42.7 Register Description

EADC Data Registers (EADC_DAT0~ EADC_DAT18)

Register	Offset	R/W	Description	Reset Value
EADC_DAT0	EADC_BA+0x00	R	EADC Data Register 0 for Sample Module 0	0x0000_0000
EADC_DAT1	EADC_BA+0x04	R	EADC Data Register 1 for Sample Module 1	0x0000_0000
EADC_DAT2	EADC_BA+0x08	R	EADC Data Register 2 for Sample Module 2	0x0000_0000
EADC_DAT3	EADC_BA+0x0C	R	EADC Data Register 3 for Sample Module 3	0x0000_0000
EADC_DAT4	EADC_BA+0x10	R	EADC Data Register 4 for Sample Module 4	0x0000_0000
EADC_DAT5	EADC_BA+0x14	R	EADC Data Register 5 for Sample Module 5	0x0000_0000
EADC_DAT6	EADC_BA+0x18	R	EADC Data Register 6 for Sample Module 6	0x0000_0000
EADC_DAT7	EADC_BA+0x1C	R	EADC Data Register 7 for Sample Module 7	0x0000_0000
EADC_DAT8	EADC_BA+0x20	R	EADC Data Register 8 for Sample Module 8	0x0000_0000
EADC_DAT9	EADC_BA+0x24	R	EADC Data Register 9 for Sample Module 9	0x0000_0000
EADC_DAT10	EADC_BA+0x28	R	EADC Data Register 10 for Sample Module 10	0x0000_0000
EADC_DAT11	EADC_BA+0x2C	R	EADC Data Register 11 for Sample Module 11	0x0000_0000
EADC_DAT12	EADC_BA+0x30	R	EADC Data Register 12 for Sample Module 12	0x0000_0000
EADC_DAT13	EADC_BA+0x34	R	EADC Data Register 13 for Sample Module 13	0x0000_0000
EADC_DAT14	EADC_BA+0x38	R	EADC Data Register 14 for Sample Module 14	0x0000_0000
EADC_DAT15	EADC_BA+0x3C	R	EADC Data Register 15 for Sample Module 15	0x0000_0000
EADC_DAT16	EADC_BA+0x40	R	EADC Data Register 16 for Sample Module 16	0x0000_0000
EADC_DAT17	EADC_BA+0x44	R	EADC Data Register 17 for Sample Module 17	0x0000_0000
EADC_DAT18	EADC_BA+0x48	R	EADC Data Register 18 for Sample Module 18	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						VALID	OV
15	14	13	12	11	10	9	8
RESULT							
7	6	5	4	3	2	1	0
RESULT							

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**Valid Flag** This bit is set to 1 when corresponding sample module channel analog input conversion is completed and cleared by hardware after EADC_DAT register is read. 0 = Data in RESULT[11:0] bits is not valid. 1 = Data in RESULT[11:0] bits is valid.
[16]	**OV**	**Overrun Flag** If converted data in RESULT[11:0] has not been read before new conversion result is loaded to this register, OV is set to 1. 0 = Data in RESULT[11:0] is recent conversion result. 1 = Data in RESULT[11:0] is overwrite. **Note:** It is cleared by hardware after EADC_DAT register is read.
[15:0]	**RESULT**	**EADC Conversion Result** This field contains 12 bits conversion result. The 12-bit EADC conversion result with unsigned format will be filled in RESULT[11:0] and zero will be filled in RESULT[15:12]. **Note:** When operating in oversampling mode, RESULT[15:0] can represent oversampling results.

EADC PDMA Current Transfer Data Register (EADC_CURDAT)

Register	Offset	R/W	Description	Reset Value
EADC_CURDAT	EADC_BA+0x4C	R	EADC PDMA Current Transfer Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**CURDAT**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CURDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CURDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CURDAT							

Bits	Description	
[31]	**Reserved**	Reserved.
[30:0]	**CURDAT**	**EADC PDMA Current Transfer Data (Read Only)** This register is a shadow register of EADC_DATn (n=0~30) for PDMA support.

EADC Control Register (EADC_CTL)

Register	Offset	R/W	Description	Reset Value
EADC_CTL	EADC_BA+0x50	R/W	EADC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
INTDEL3				INTDEL2			
23	22	21	20	19	18	17	16
INTDEL1				INTDEL0			
15	14	13	12	11	10	9	8
Reserved						DMOF	DIFFEN
7	6	5	4	3	2	1	0
Reserved		ADCIEN3	ADCIEN2	ADCIEN1	ADCIEN0	ADCRST	ADCEN

Bits	Description	
[31:28]	**INTDEL3**	**ADC Start Of Conversion ADINT3 Delay Cycle Selection** Start of conversion interrupt ADINT3 will delay INTDEL3 PCLK cycles to generate interrupt. The function support delay 1 PCLK to 15 PCLK cycles, user can select one of the options according to the relationship of PCLK and ADC _CLK which user selected. 4'h0 = No delay cycle. 4'h1 = Start of conversion interrupt ADINT3 delay 1 PCLK cycle. 4'h2 = Start of conversion interrupt ADINT3 delay 2 PCLK cycles. 4'h3 = Start of conversion interrupt ADINT3 delay 3 PCLK cycles. 4'h4 = Start of conversion interrupt ADINT3 delay 4 PCLK cycles. 4'h5 = Start of conversion interrupt ADINT3 delay 5 PCLK cycles. 4'h6 = Start of conversion interrupt ADINT3 delay 6 PCLK cycles. 4'h7 = Start of conversion interrupt ADINT3 delay 7 PCLK cycles. 4'h8 = Start of conversion interrupt ADINT3 delay 8 PCLK cycles. 4'h9 = Start of conversion interrupt ADINT3 delay 9 PCLK cycles. 4'ha = Start of conversion interrupt ADINT3 delay 10 PCLK cycles. 4'hb = Start of conversion interrupt ADINT3 delay 11 PCLK cycles. 4'hc = Start of conversion interrupt ADINT3 delay 12 PCLK cycles. 4'hd = Start of conversion interrupt ADINT3 delay 13 PCLK cycles. 4'he = Start of conversion interrupt ADINT3 delay 14 PCLK cycles. 4'hf = Start of conversion interrupt ADINT3 delay 15 PCLK cycles. **Note:** This function is workable only when any one of INTPOS (EADC_SCTLx[5]), x=0~15 is set. **Note:** It is noted that the delayed interrupt ADINT3 must occurs before the next ADINT3 generated when use the same sample module to control EADC conversion.

Bits	Description	
[27:24]	**INTDEL2**	**ADC Start Of Conversion ADINT2 Delay Cycle Selection** Start of conversion interrupt ADINT2 will delay INTDEL2 PCLK cycles to generate interrupt. The function support delay 1 PCLK to 15 PCLK cycles, user can select one of the options according to the relationship of PCLK and ADC _CLK which user which user selected. 4'h0 = No delay cycle. 4'h1 = Start of conversion interrupt ADINT2 delay 1 PCLK cycle. 4'h2 = Start of conversion interrupt ADINT2 delay 2 PCLK cycles. 4'h3 = Start of conversion interrupt ADINT2 delay 3 PCLK cycles. 4'h4 = Start of conversion interrupt ADINT2 delay 4 PCLK cycles. 4'h5 = Start of conversion interrupt ADINT2 delay 5 PCLK cycles. 4'h6 = Start of conversion interrupt ADINT2 delay 6 PCLK cycles. 4'h7 = Start of conversion interrupt ADINT2 delay 7 PCLK cycles. 4'h8 = Start of conversion interrupt ADINT2 delay 8 PCLK cycles. 4'h9 = Start of conversion interrupt ADINT2 delay 9 PCLK cycles. 4'ha = Start of conversion interrupt ADINT2 delay 10 PCLK cycles. 4'hb = Start of conversion interrupt ADINT2 delay 11 PCLK cycles. 4'hc = Start of conversion interrupt ADINT2 delay 12 PCLK cycles. 4'hd = Start of conversion interrupt ADINT2 delay 13 PCLK cycles. 4'he = Start of conversion interrupt ADINT2 delay 14 PCLK cycles. 4'hf = Start of conversion interrupt ADINT2 delay 15 PCLK cycles. **Note:** This function is workable only when any one of INTPOS (EADC_SCTLx[5]), x=0~15 is set. **Note:** It is noted that the delayed interrupt ADINT2 must occurs before the next ADINT2 generated when use the same sample module to control EADC conversion.
[23:20]	**INTDEL1**	**ADC Start Of Conversion ADINT1 Delay Cycle Selection** Start of conversion interrupt ADINT1 will delay INTDEL1 PCLK cycles to generate interrupt. The function support delay 1 PCLK to 15 PCLK cycles, user can select one of the options according to the relationship of PCLK and ADC _CLK which user selected. 4'h0 = No delay cycle. 4'h1 = Start of conversion interrupt ADINT1 delay 1 PCLK cycle. 4'h2 = Start of conversion interrupt ADINT1 delay 2 PCLK cycles. 4'h3 = Start of conversion interrupt ADINT1 delay 3 PCLK cycles. 4'h4 = Start of conversion interrupt ADINT1 delay 4 PCLK cycles. 4'h5 = Start of conversion interrupt ADINT1 delay 5 PCLK cycles. 4'h6 = Start of conversion interrupt ADINT1 delay 6 PCLK cycles. 4'h7 = Start of conversion interrupt ADINT1 delay 7 PCLK cycles. 4'h8 = Start of conversion interrupt ADINT1 delay 8 PCLK cycles. 4'h9 = Start of conversion interrupt ADINT1 delay 9 PCLK cycles. 4'ha = Start of conversion interrupt ADINT1 delay 10 PCLK cycles. 4'hb = Start of conversion interrupt ADINT1 delay 11 PCLK cycles. 4'hc = Start of conversion interrupt ADINT1 delay 12 PCLK cycles. 4'hd = Start of conversion interrupt ADINT1 delay 13 PCLK cycles. 4'he = Start of conversion interrupt ADINT1 delay 14 PCLK cycles. 4'hf = Start of conversion interrupt ADINT1 delay 15 PCLK cycles. **Note:** This function is workable only when any one of INTPOS (EADC_SCTLx[5]), x=0~15 is set. **Note:** It is noted that the delayed interrupt ADINT1 must occurs before the next ADINT1 generated when use the same sample module to control EADC conversion.

Bits	Description	
[19:16]	**INTDEL0**	**ADC Start Of Conversion ADINT0 Delay Cycle Selection** Start of conversion interrupt ADINT0 will delay INTDEL0 PCLK cycles to generate interrupt. The function support delay 1 PCLK to 15 PCLK cycles, user can select one of the options according to the relationship of PCLK and ADC _CLK which user selected. 4'h0 = No delay cycle. 4'h1 = Start of conversion interrupt ADINT0 delay 1 PCLK cycle. 4'h2 = Start of conversion interrupt ADINT0 delay 2 PCLK cycles. 4'h3 = Start of conversion interrupt ADINT0 delay 3 PCLK cycles. 4'h4 = Start of conversion interrupt ADINT0 delay 4 PCLK cycles. 4'h5 = Start of conversion interrupt ADINT0 delay 5 PCLK cycles. 4'h6 = Start of conversion interrupt ADINT0 delay 6 PCLK cycles. 4'h7 = Start of conversion interrupt ADINT0 delay 7 PCLK cycles. 4'h8 = Start of conversion interrupt ADINT0 delay 8 PCLK cycles. 4'h9 = Start of conversion interrupt ADINT0 delay 9 PCLK cycles. 4'ha = Start of conversion interrupt ADINT0 delay 10 PCLK cycles. 4'hb = Start of conversion interrupt ADINT0 delay 11 PCLK cycles. 4'hc = Start of conversion interrupt ADINT0 delay 12 PCLK cycles. 4'hd = Start of conversion interrupt ADINT0 delay 13 PCLK cycles. 4'he = Start of conversion interrupt ADINT0 delay 14 PCLK cycles. 4'hf = Start of conversion interrupt ADINT0 delay 15 PCLK cycles. **Note:** This function is workable only when any one of INTPOS (EADC_SCTLx[5]), x=0~15 is set. **Note:** It is noted that the delayed interrupt ADINT0 must occurs before the next ADINT0 generated when use the same sample module to control EADC conversion.
[15:10]	**Reserved**	Reserved.
[9]	**DMOF**	**ADC Differential Input Mode Output Format** 0 = ADC conversion result will be filled in RESULT (EADC_DATn[15:0], where n= 0 ~18) with unsigned format. 1 = ADC conversion result will be filled in RESULT (EADC_DATn[15:0], where n= 0 ~18) with 2'complement format.
[8]	**DIFFEN**	**Differential Analog Input Mode Enable Bit** 0 = Single-end analog input mode. 1 = Differential analog input mode. **Note:** In the differential mode, the input channel pair must be configured to EADC_CHx, EADC_CHx+1 , x=0,2,4,6,8,10,12,14.
[7:6]	**Reserved**	Reserved.
[5]	**ADCIEN3**	**Specific Sample Module EADC ADINT3 Interrupt Enable Bit** The EADC converter generates a conversion end ADIF3 (EADC_STATUS2[3]) upon the end of specific sample module EADC conversion. If EADCIEN3 bit is set then conversion end interrupt request ADINT3 is generated. 0 = Specific sample module EADC ADINT3 interrupt function Disabled. 1 = Specific sample module EADC ADINT3 interrupt function Enabled.
[4]	**ADCIEN2**	**Specific Sample Module EADC ADINT2 Interrupt Enable Bit** The EADC converter generates a conversion end ADIF2 (EADC_STATUS2[2]) upon the end of specific sample module EADC conversion. If EADCIEN2 bit is set then conversion end interrupt request ADINT2 is generated. 0 = Specific sample module EADC ADINT2 interrupt function Disabled. 1 = Specific sample module EADC ADINT2 interrupt function Enabled.

Bits	Description	
[3]	**ADCIEN1**	**Specific Sample Module EADC ADINT1 Interrupt Enable Bit** The EADC converter generates a conversion end ADIF1 (EADC_STATUS2[1]) upon the end of specific sample module EADC conversion. If EADCIEN1 bit is set then conversion end interrupt request ADINT1 is generated. 0 = Specific sample module EADC ADINT1 interrupt function Disabled. 1 = Specific sample module EADC ADINT1 interrupt function Enabled.
[2]	**ADCIEN0**	**Specific Sample Module EADC ADINT0 Interrupt Enable Bit** The EADC converter generates a conversion end ADIF0 (EADC_STATUS2[0]) upon the end of specific sample module EADC conversion. If ADCIEN0 bit is set then conversion end interrupt request ADINT0 is generated. 0 = Specific sample module EADC ADINT0 interrupt function Disabled. 1 = Specific sample module EADC ADINT0 interrupt function Enabled.
[1]	**ADCRST**	**EADC Converter Control Circuits Reset** 0 = No effect. 1 = Cause EADC control circuits reset to initial state, but not change the EADC registers value. **Note: EADC**RST bit remains 1 during EADC reset, when EADC reset end, the EADCRST bit is automatically cleared to 0.
[0]	**ADCEN**	**EADC Converter Enable Bit** 0 = EADC Disabled. 1 = EADC Enabled. **Note:** Before starting EADC conversion function, this bit should be set to 1. Clear it to 0 to disable EADC converter analog circuit power consumption.

EADC Sample Module Software Start Register (EADC_SWTRG)

Register	Offset	R/W	Description	Reset Value
EADC_SWTRG	EADC_BA+0x54	W	EADC Sample Module Software Start Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	SWTRG						
23	22	21	20	19	18	17	16
SWTRG							
15	14	13	12	11	10	9	8
SWTRG							
6	6	5	4	3	2	1	0
SWTRG							

Bits	Description	
[31]	**Reserved**	Reserved.
[30:0]	**SWTRG**	**EADC Sample Module 0~30 Software Force to Start EADC Conversion** 0 = No effect. 1 = Cause an EADC conversion when the priority is given to sample module. **Note:** After writing this register to start EADC conversion, the EADC_PENDSTS register will show which sample module will conversion. If user want to disable the conversion of the sample module, user can write EADC_PENDSTS register to clear it.

EADC Sample Module Start of Conversion Pending Flag Register (EADC_PENDSTS)

Register	Offset	R/W	Description	Reset Value
EADC_PENDSTS	EADC_BA+0x58	R/W	EADC Start of Conversion Pending Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**STPF**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
STPF							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
STPF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STPF							

Bits	Description	
[31]	**Reserved**	Reserved.
[30:0]	**STPF**	**EADC Sample Module 0~30 Start of Conversion Pending Flag** Read Operation: 0 = There is no pending conversion for sample module. 1 = Sample module EADC start of conversion is pending. Write Operation: 1 = Clear pending flag & cancel the conversion for sample module. **Note:** This bit remains 1 during pending state, when the respective EADC conversion is end, the STPFn (n=0~30) bit is automatically cleared to 0.

EADC Sample Module Overrun Flag Register (EADC_OVSTS)

Register	Offset	R/W	Description	Reset Value
EADC_OVSTS	EADC_BA+0x5C	R/W	EADC Sample Module Start of Conversion Overrun Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**SPOVF**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SPOVF							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SPOVF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPOVF							

Bits	Description	
[31]	**Reserved**	Reserved.
[30:0]	**SPOVF**	**EADC SAMPLE0~30 Overrun Flag** 0 = No sample module event overrun. 1 = Indicates a new sample module event is generated while an old one event is pending. **Note:** This bit is cleared by writing 1 to it.

EADC Control1 Register (EADC_CTL1)

Register	Offset	R/W	Description	Reset Value
EADC_CTL1	EADC_BA+0x60	R/W	EADC Control1 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CMP3TRG	CMP2TRG	CMP1TRG	CMP0TRG	Reserved			
15	14	13	12	11	10	9	8
Reserved							FDETCHEN
7	6	5	4	3	2	1	0
Reserved		RESSEL		Reserved		DISCHEN	PRECHEN

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**CMP3TRG**	**ADC Comparator 3 Trigger EPWM Brake Enable Bit** 0 = Comparator 3 trigger EPWM brake is disabled. 1 = Comparator 3 trigger EPWM brake is enabled.
[22]	**CMP2TRG**	**ADC Comparator 2 Trigger EPWM Brake Enable Bit** 0 = Comparator 2 trigger EPWM brake is disabled. 1 = Comparator 2 trigger EPWM brake is enabled.
[21]	**CMP1TRG**	**ADC Comparator 1 Trigger EPWM Brake Enable Bit** 0 = Comparator 1 trigger EPWM brake is disabled. 1 = Comparator 1 trigger EPWM brake is enabled.
[20]	**CMP0TRG**	**ADC Comparator 0 Trigger EPWM Brake Enable Bit** 0 = Comparator 0 trigger EPWM brake is disabled. 1 = Comparator 0 trigger EPWM brake is enabled.
[19:9]	**Reserved**	Reserved.
[8]	**FDETCHEN**	**Floating Detect Channel Enable Bit** 0 = Floating Detect Channel Disabled. 1 = Floating Detect Channel Enabled.
[7:6]	**Reserved**	Reserved.
[5:4]	**RESSEL**	**Resolution Select Bits** 00 = ADC resolution 12 bits. 01 = ADC resolution 10 bits. 10 = ADC resolution 8 bits. 11 = Reserved.
[3:2]	**Reserved**	Reserved.
[1]	**DISCHEN**	**Discharge Enable Bit** 0 = Channel discharge Disabled.

		1 = Channel discharge Enabled. **Note:** Analog input voltage is 1/2 V_{REF} when PRECHEN and DISCHEN are all enable.
[0]	**PRECHEN**	**Precharge Enable Bit** 0 = Channel precharge Disabled. 1 = Channel precharge Enabled. **Note:** Analog input voltage is 1/2 V_{REF} when PRECHEN and DISCHEN are all enable.

EADC Sample Module 0~18 Control Registers (EADC_SCTL0~EADC_SCTL18)

Register	Offset	R/W	Description	Reset Value
EADC_SCTL0	EADC_BA+0x80	R/W	EADC Sample Module 0 Control Register	0x0000_0014
EADC_SCTL1	EADC_BA+0x84	R/W	EADC Sample Module 1 Control Register	0x0000_0014
EADC_SCTL2	EADC_BA+0x88	R/W	EADC Sample Module 2 Control Register	0x0000_0014
EADC_SCTL3	EADC_BA+0x8C	R/W	EADC Sample Module 3 Control Register	0x0000_0014
EADC_SCTL4	EADC_BA+0x90	R/W	EADC Sample Module 4 Control Register	0x0000_0014
EADC_SCTL5	EADC_BA+0x94	R/W	EADC Sample Module 5 Control Register	0x0000_0014
EADC_SCTL6	EADC_BA+0x98	R/W	EADC Sample Module 6 Control Register	0x0000_0014
EADC_SCTL7	EADC_BA+0x9C	R/W	EADC Sample Module 7 Control Register	0x0000_0014
EADC_SCTL8	EADC_BA+0xA0	R/W	EADC Sample Module 8 Control Register	0x0000_0014
EADC_SCTL9	EADC_BA+0xA4	R/W	EADC Sample Module 9 Control Register	0x0000_0014
EADC_SCTL10	EADC_BA+0xA8	R/W	EADC Sample Module 10 Control Register	0x0000_0014
EADC_SCTL11	EADC_BA+0xAC	R/W	EADC Sample Module 11 Control Register	0x0000_0014
EADC_SCTL12	EADC_BA+0xB0	R/W	EADC Sample Module 12 Control Register	0x0000_0014
EADC_SCTL13	EADC_BA+0xB4	R/W	EADC Sample Module 13 Control Register	0x0000_0014
EADC_SCTL14	EADC_BA+0xB8	R/W	EADC Sample Module 14 Control Register	0x0000_0014
EADC_SCTL15	EADC_BA+0xBC	R/W	EADC Sample Module 15 Control Register	0x0000_0014
EADC_SCTL16	EADC_BA+0xC0	R/W	EADC Sample Module 16 Control Register	0x0000_0014
EADC_SCTL17	EADC_BA+0xC4	R/W	EADC Sample Module 17 Control Register	0x0000_0014
EADC_SCTL18	EADC_BA+0xC8	R/W	EADC Sample Module 18 Control Register	0x0000_0014

31	30	29	28	27	26	25	24
EXTSMPT							
23	22	21	20	19	18	17	16
EXTFEN	EXTREN	TRGSEL					
15	14	13	12	11	10	9	8
TRGDLCNT							
7	6	5	4	3	2	1	0
TRGDLDIV		INTPOS	CHSEL				

Bits	Description	
[31:24]	**EXTSMPT**	**EADC Sampling Time Extend** When EADC converting at high conversion rate, the sampling time of analog input voltage may not be enough if input channel loading is heavy, and software can extend EADC sampling time after trigger source is coming to get enough sampling time. Extended Sampling Time = (EXTSMPT x EADC_CLK period x n)+2 [n=1,2,4,16 from EXTSTDIV setting].
[23]	**EXTFEN**	**EADC External Trigger Falling Edge Enable Bit** 0 = Falling edge Disabled when EADC selects EADC0_ST as trigger source. 1 = Falling edge Enabled when EADC selects EADC0_ST as trigger source.
[22]	**EXTREN**	**EADC External Trigger Rising Edge Enable Bit** 0 = Rising edge Disabled when EADC selects EADC0_ST as trigger source. 1 = Rising edge Enabled when EADC selects EADC0_ST as trigger source.

Bits	Description	
[21:16]	**TRGSEL**	**EADC Sample Module Start of Conversion Trigger Source Selection** 0H = Disable trigger. 1H = External trigger from EADC0_ST pin input. 2H = EADC ADINT0 interrupt EOC pulse trigger. 3H = EADC ADINT1 interrupt EOC pulse trigger. 4H = Timer0 overflow pulse trigger. 5H = Timer1 overflow pulse trigger. 6H = Timer2 overflow pulse trigger. 7H = Timer3 overflow pulse trigger. 8H = EPWM0TG0. 9H = EPWM0TG1. AH = EPWM0TG2. BH = EPWM0TG3. CH = EPWM0TG4. DH = EPWM0TG5. EH = EPWM1TG0. FH = EPWM1TG1. 10H = EPWM1TG2. 11H = EPWM1TG3. 12H = EPWM1TG4. 13H = EPWM1TG5. 14H = PWM0TG0. 15H = PWM0TG1. 16H = PWM0TG2. 17H = PWM0TG3. 18H = PWM0TG4. 19H = PWM0TG5. 1AH = PWM1TG0. 1BH = PWM1TG1. 1CH = PWM1TG2. 1DH = PWM1TG3. 1EH = PWM1TG4. 1FH = PWM1TG5. 20H = ACMP0_INT. 21H = ACMP1_INT. 22H = ACMP2_INT. other = Reserved.
[15:8]	**TRGDLCNT**	**EADC Sample Module Start of Conversion Trigger Delay Time** Trigger delay time = TRGDLCNT x EADC_CLK period x n (n=1,2,4,16 from TRGDLDIV setting).
[7:6]	**TRGDLDIV**	**EADC Sample Module Start of Conversion Trigger Delay Clock Divider Selection** Trigger delay clock frequency: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.

Bits	Description	
[5]	**INTPOS**	**Interrupt Flag Position Select** 0 = Set ADIFn (EADC_STATUS2[n], n=0~3) at EADC end of conversion. 1 = Set ADIFn (EADC_STATUS2[n], n=0~3) at EADC start of conversion.
[4:0]	**CHSEL**	**EADC Sample Module Channel Selection** 00H = EADC_CH0. 01H = EADC_CH1. 02H = EADC_CH2. 03H = EADC_CH3. 04H = EADC_CH4. 05H = EADC_CH5. 06H = EADC_CH6. 07H = EADC_CH7. 08H = EADC_CH8. 09H = EADC_CH9. 0AH = EADC_CH10. 0BH = EADC_CH11. 0CH = EADC_CH12. 0DH = EADC_CH13. 0EH = EADC_CH14. 0FH = EADC_CH15. 10H = EADC_CH16. 11H = EADC_CH17. 12H = EADC_CH18. 13H = EADC_CH19. 14H = EADC_CH20. 15H = EADC_CH21. 16H = EADC_CH22. 17H = EADC_CH23. 18H = EADC_CH24. 19H = EADC_CH25. 1AH = EADC_CH26. Others = Reserved.

EADC Interrupt Source Enable Control Registers (EADC_INTSRC0~EADC_INTSRC3)

Register	Offset	R/W	Description	Reset Value
EADC_INTSRC0	EADC_BA+0xD0	R/W	EADC Interrupt 0 Source Enable Control Register.	0x0000_0000
EADC_INTSRC1	EADC_BA+0xD4	R/W	EADC Interrupt 1 Source Enable Control Register.	0x0000_0000
EADC_INTSRC2	EADC_BA+0xD8	R/W	EADC Interrupt 2 Source Enable Control Register.	0x0000_0000
EADC_INTSRC3	EADC_BA+0xDC	R/W	EADC Interrupt 3 Source Enable Control Register.	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	SPLIE30	SPLIE29	SPLIE28	SPLIE27	SPLIE26	SPLIE25	SPLIE24
23	22	21	20	19	18	17	16
SPLIE23	SPLIE22	SPLIE21	SPLIE20	SPLIE19	SPLIE18	SPLIE17	SPLIE16
15	14	13	12	11	10	9	8
SPLIE15	SPLIE14	SPLIE13	SPLIE12	SPLIE11	SPLIE10	SPLIE9	SPLIE8
7	6	5	4	3	2	1	0
SPLIE7	SPLIE6	SPLIE5	SPLIE4	SPLIE3	SPLIE2	SPLIE1	SPLIE0

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**SPLIE30**	**Sample Module 30 Interrupt Enable Bit** 0 = Sample Module 30 interrupt Disabled. 1 = Sample Module 30 interrupt Enabled.
[29]	**SPLIE29**	**Sample Module 29 Interrupt Enable Bit** 0 = Sample Module 29 interrupt Disabled. 1 = Sample Module 29 interrupt Enabled.
[28]	**SPLIE28**	**Sample Module 28 Interrupt Enable Bit** 0 = Sample Module 28 interrupt Disabled. 1 = Sample Module 28 interrupt Enabled.
[27]	**SPLIE27**	**Sample Module 27 Interrupt Enable Bit** 0 = Sample Module 27 interrupt Disabled. 1 = Sample Module 27 interrupt Enabled.
[26]	**SPLIE26**	**Sample Module 26 Interrupt Enable Bit** 0 = Sample Module 26 interrupt Disabled. 1 = Sample Module 26 interrupt Enabled.
[25]	**SPLIE25**	**Sample Module 25 Interrupt Enable Bit** 0 = Sample Module 25 interrupt Disabled. 1 = Sample Module 25 interrupt Enabled.
[24]	**SPLIE24**	**Sample Module 24 Interrupt Enable Bit** 0 = Sample Module 24 interrupt Disabled. 1 = Sample Module 24 interrupt Enabled.

Bits	Description	
[23]	**SPLIE23**	**Sample Module 23 Interrupt Enable Bit** 0 = Sample Module 23 interrupt Disabled. 1 = Sample Module 23 interrupt Enabled.
[22]	**SPLIE22**	**Sample Module 22 Interrupt Enable Bit** 0 = Sample Module 22 interrupt Disabled. 1 = Sample Module 22 interrupt Enabled.
[21]	**SPLIE21**	**Sample Module 21 Interrupt Enable Bit** 0 = Sample Module 21 interrupt Disabled. 1 = Sample Module 21 interrupt Enabled.
[20]	**SPLIE20**	**Sample Module 20 Interrupt Enable Bit** 0 = Sample Module 20 interrupt Disabled. 1 = Sample Module 20 interrupt Enabled.
[19]	**SPLIE19**	**Sample Module 19 Interrupt Enable Bit** 0 = Sample Module 19 interrupt Disabled. 1 = Sample Module 19 interrupt Enabled.
[18]	**SPLIE18**	**Sample Module 18 Interrupt Enable Bit** 0 = Sample Module 18 interrupt Disabled. 1 = Sample Module 18 interrupt Enabled.
[17]	**SPLIE17**	**Sample Module 17 Interrupt Enable Bit** 0 = Sample Module 17 interrupt Disabled. 1 = Sample Module 17 interrupt Enabled.
[16]	**SPLIE16**	**Sample Module 16 Interrupt Enable Bit** 0 = Sample Module 16 interrupt Disabled. 1 = Sample Module 16 interrupt Enabled.
[15]	**SPLIE15**	**Sample Module 15 Interrupt Enable Bit** 0 = Sample Module 15 interrupt Disabled. 1 = Sample Module 15 interrupt Enabled.
[14]	**SPLIE14**	**Sample Module 14 Interrupt Enable Bit** 0 = Sample Module 14 interrupt Disabled. 1 = Sample Module 14 interrupt Enabled.
[13]	**SPLIE13**	**Sample Module 13 Interrupt Enable Bit** 0 = Sample Module 13 interrupt Disabled. 1 = Sample Module 13 interrupt Enabled.
[12]	**SPLIE12**	**Sample Module 12 Interrupt Enable Bit** 0 = Sample Module 12 interrupt Disabled. 1 = Sample Module 12 interrupt Enabled.
[11]	**SPLIE11**	**Sample Module 11 Interrupt Enable Bit** 0 = Sample Module 11 interrupt Disabled. 1 = Sample Module 11 interrupt Enabled.
[10]	**SPLIE10**	**Sample Module 10 Interrupt Enable Bit** 0 = Sample Module 10 interrupt Disabled. 1 = Sample Module 10 interrupt Enabled.

Bits	Description	
[9]	**SPLIE9**	**Sample Module 9 Interrupt Enable Bit** 0 = Sample Module 9 interrupt Disabled. 1 = Sample Module 9 interrupt Enabled.
[8]	**SPLIE8**	**Sample Module 8 Interrupt Enable Bit** 0 = Sample Module 8 interrupt Disabled. 1 = Sample Module 8 interrupt Enabled.
[7]	**SPLIE7**	**Sample Module 7 Interrupt Enable Bit** 0 = Sample Module 7 interrupt Disabled. 1 = Sample Module 7 interrupt Enabled.
[6]	**SPLIE6**	**Sample Module 6 Interrupt Enable Bit** 0 = Sample Module 6 interrupt Disabled. 1 = Sample Module 6 interrupt Enabled.
[5]	**SPLIE5**	**Sample Module 5 Interrupt Enable Bit** 0 = Sample Module 5 interrupt Disabled. 1 = Sample Module 5 interrupt Enabled.
[4]	**SPLIE4**	**Sample Module 4 Interrupt Enable Bit** 0 = Sample Module 4 interrupt Disabled. 1 = Sample Module 4 interrupt Enabled.
[3]	**SPLIE3**	**Sample Module 3 Interrupt Enable Bit** 0 = Sample Module 3 interrupt Disabled. 1 = Sample Module 3 interrupt Enabled.
[2]	**SPLIE2**	**Sample Module 2 Interrupt Enable Bit** 0 = Sample Module 2 interrupt Disabled. 1 = Sample Module 2 interrupt Enabled.
[1]	**SPLIE1**	**Sample Module 1 Interrupt Enable Bit** 0 = Sample Module 1 interrupt Disabled. 1 = Sample Module 1 interrupt Enabled.
[0]	**SPLIE0**	**Sample Module 0 Interrupt Enable Bit** 0 = Sample Module 0 interrupt Disabled. 1 = Sample Module 0 interrupt Enabled.

EADC Result Compare Register 0/1/2/3 (EADC_CMP0/1/2/3)

Register	Offset	R/W	Description	Reset Value
EADC_CMP0	EADC_BA+0xE0	R/W	EADC Result Compare Register 0	0x0000_0000
EADC_CMP1	EADC_BA+0xE4	R/W	EADC Result Compare Register 1	0x0000_0000
EADC_CMP2	EADC_BA+0xE8	R/W	EADC Result Compare Register 2	0x0000_0000
EADC_CMP3	EADC_BA+0xEC	R/W	EADC Result Compare Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				CMPDAT			
23	22	21	20	19	18	17	16
CMPDAT							
15	14	13	12	11	10	9	8
CMPWEN	Reserved			CMPMCNT			
7	6	5	4	3	2	1	0
CMPSPL					CMPCOND	ADCMPIE	ADCMPEN

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**CMPDAT**	**Comparison Data** The 12 bits data is used to compare with conversion result of specified sample module. User can use it to monitor the external analog input pin voltage transition without imposing a load on software.
[15]	**CMPWEN**	**Compare Window Mode Enable Bit** 0 = EADCMPF0 (EADC_STATUS2[4]) will be set when EADC_CMP0 compared condition matched. EADCMPF2 (EADC_STATUS2[6]) will be set when EADC_CMP2 compared condition matched 1 = EADCMPF0 (EADC_STATUS2[4]) will be set when both EADC_CMP0 and EADC_CMP1 compared condition matched. EADCMPF2 (EADC_STATUS2[6]) will be set when both EADC_CMP2 and EADC_CMP3 compared condition matched. **Note:** This bit is only present in EADC_CMP0 and EADC_CMP2 register. **Note:** When in compare window mode, the CMPCNT setting only follow EADC_CMP0, EADC_CMP2 registers
[14:12]	**Reserved**	Reserved.
[11:8]	**CMPMCNT**	**Compare Match Count** When the specified EADC sample module analog conversion result matches the compare condition defined by CMPCOND (EADC_CMPn[2], n=0~3), the internal match counter will increase 1. If the compare result does not meet the compare condition, the internal compare match counter will reset to 0. When the internal counter reaches the value to (CMPMCNT +1), the ADCMPFn (EADC_STATUS2[7:4], n=0~3) will be set.

Bits	Description	
[7:3]	**CMPSPL**	**Compare Sample Module Selection** 00000 = Sample Module 0 conversion result EADC_DAT0 is selected to be compared. 00001 = Sample Module 1 conversion result EADC_DAT1 is selected to be compared. 00010 = Sample Module 2 conversion result EADC_DAT2 is selected to be compared. 00011 = Sample Module 3 conversion result EADC_DAT3 is selected to be compared. 00100 = Sample Module 4 conversion result EADC_DAT4 is selected to be compared. 00101 = Sample Module 5 conversion result EADC_DAT5 is selected to be compared. 00110 = Sample Module 6 conversion result EADC_DAT6 is selected to be compared. 00111 = Sample Module 7 conversion result EADC_DAT7 is selected to be compared. 01000 = Sample Module 8 conversion result EADC_DAT8 is selected to be compared. 01001 = Sample Module 9 conversion result EADC_DAT9 is selected to be compared. 01010 = Sample Module 10 conversion result EADC_DAT10 is selected to be compared. 01011 = Sample Module 11 conversion result EADC_DAT11 is selected to be compared. 01100 = Sample Module 12 conversion result EADC_DAT12 is selected to be compared. 01101 = Sample Module 13 conversion result EADC_DAT13 is selected to be compared. 01110 = Sample Module 14 conversion result EADC_DAT14 is selected to be compared. 01111 = Sample Module 15 conversion result EADC_DAT15 is selected to be compared. 10000 = Sample Module 16 conversion result EADC_DAT16 is selected to be compared. 10001 = Sample Module 17 conversion result EADC_DAT17 is selected to be compared. 10010 = Sample Module 18 conversion result EADC_DAT18 is selected to be compared. 10011 = Sample Module 19 conversion result EADC_DAT19 is selected to be compared. 10100 = Sample Module 20 conversion result EADC_DAT20 is selected to be compared. 10101 = Sample Module 21 conversion result EADC_DAT21 is selected to be compared. 10110 = Sample Module 22 conversion result EADC_DAT22 is selected to be compared. 10111 = Sample Module 23 conversion result EADC_DAT23 is selected to be compared. 11000 = Sample Module 24 conversion result EADC_DAT24 is selected to be compared. 11001 = Sample Module 25 conversion result EADC_DAT25 is selected to be compared. 11010 = Sample Module 26 conversion result EADC_DAT26 is selected to be compared. 11011 = Sample Module 27 conversion result EADC_DAT27 is selected to be compared. 11100 = Sample Module 28 conversion result EADC_DAT28 is selected to be compared. 11101 = Sample Module 29 conversion result EADC_DAT29 is selected to be compared. 11110 = Sample Module 30 conversion result EADC_DAT30 is selected to be compared. Others = reserved.
[2]	**CMPCOND**	**Compare Condition** 0= Set the compare condition as that when a 12-bit EADC conversion result is less than the 12-bit CMPDAT (EADC_CMPn [27:16]), the internal match counter will increase one. 1= Set the compare condition as that when a 12-bit EADC conversion result is greater or equal to the 12-bit CMPDAT (EADC_CMPn [27:16]), the internal match counter will increase one. **Note:** When the internal counter reaches the value to (CMPMCNT (EADC_CMPn[11:8], n=0~3) +1), the ADCMPF bit will be set.
[1]	**ADCMPIE**	**EADC Result Compare Interrupt Enable Bit** 0 = Compare function interrupt Disabled. 1 = Compare function interrupt Enabled. If the compare function is enabled and the compare condition matches the setting of CMPCOND (EADC_CMPn[2], n=0~3) and CMPMCNT (EADC_CMPn[11:8], n=0~3), ADCMPFn (EADC_STATUS2[7:4], n=0~3) will be asserted, in the meanwhile, if ADCMPIE is set to 1, a compare interrupt request is generated.

Bits	Description	
[0]	**ADCMPEN**	**EADC Result Compare Enable Bit** 0 = Compare Disabled. 1 = Compare Enabled. Set this bit to 1 to enable compare CMPDAT (EADC_CMPn[27:16], n=0~3) with specified sample module conversion result when converted data is loaded into EADC_DAT register.

EADC Status Register 0 (EADC_STATUS0)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS0	EADC_BA+0xF0	R	EADC Status Register 0	0x0000_0000

31	30	29	28	27	26	25	24
OV							
23	22	21	20	19	18	17	16
OV							
15	14	13	12	11	10	9	8
VALID							
7	6	5	4	3	2	1	0
VALID							

Bits	Description	
[31:16]	**OV**	**EADC_DAT0~15 Overrun Flag** It is a mirror to OV bit in sample module EADC result data register EADC_DATn. (n=0~15).
[15:0]	**VALID**	**EADC_DAT0~15 Data Valid Flag** It is a mirror of VALID bit in sample module EADC result data register EADC_DATn. (n=0~15).

EADC Status Register 1 (EADC_STATUS1)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS1	EADC_BA+0xF4	R	EADC Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	OV						
23	22	21	20	19	18	17	16
OV							
15	14	13	12	11	10	9	8
Reserved	VALID						
7	6	5	4	3	2	1	0
VALID							

Bits	Description	
[31]	**Reserved**	Reserved.
[30:16]	**OV**	**EADC_DAT16~30 Overrun Flag** It is a mirror to OV bit in sample module EADC result data register EADC_DATn. (n=16~30).
[15]	**Reserved**	Reserved.
[14:0]	**VALID**	**EADC_DAT16~30 Data Valid Flag** It is a mirror of VALID bit in sample module EADC result data register EADC_DATn. (n=16~30).

EADC Status Register 2 (EADC_STATUS2)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS2	EADC_BA+0xF8	R/W	EADC Status Register 2	0x001B_0000

31	30	29	28	27	26	25	24
Reserved				AOV	AVALID	STOVF	ADOVIF
23	22	21	20	19	18	17	16
BUSY	Reserved		CHANNEL				
15	14	13	12	11	10	9	8
ADCMPO3	ADCMPO2	ADCMPO1	ADCMPO0	ADOVIF3	ADOVIF2	ADOVIF1	ADOVIF0
7	6	5	4	3	2	1	0
ADCMPF3	ADCMPF2	ADCMPF1	ADCMPF0	ADIF3	ADIF2	ADIF1	ADIF0

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**AOV**	**for All Sample Module EADC Result Data Register Overrun Flags Check (Read Only)** n=0~30. 0 = None of sample module data register overrun flag OVn (EADC_DATn[16]) is set to 1. 1 = Any one of sample module data register overrun flag OVn (EADC_DATn[16]) is set to 1. **Note:** This bit will keep 1 when any OVn Flag is equal to 1.
[26]	**AVALID**	**for All Sample Module EADC Result Data Register EADC_DAT Data Valid Flag Check (Read Only)** n=0~30. 0 = None of sample module data register valid flag VALIDn (EADC_DATn[17]) is set to 1. 1 = Any one of sample module data register valid flag VALIDn (EADC_DATn[17]) is set to 1. **Note:** This bit will keep 1 when any VALIDn Flag is equal to 1.
[25]	**STOVF**	**for All EADC Sample Module Start of Conversion Overrun Flags Check (Read Only)** n=0~30. 0 = None of sample module event overrun flag SPOVFn (EADC_OVSTS[n]) is set to 1. 1 = Any one of sample module event overrun flag SPOVFn (EADC_OVSTS[n]) is set to 1. **Note:** This bit will keep 1 when any SPOVFn Flag is equal to 1.
[24]	**ADOVIF**	**All EADC Interrupt Flag Overrun Bits Check (Read Only)** n=0~3. 0 = None of ADINT interrupt flag ADOVIFn, n=0~3 is overwritten to 1. 1 = Any one of ADINT interrupt flag ADOVIFn, n=0~3 is overwritten to 1. **Note:** This bit will keep 1 when any ADOVIFn Flag is equal to 1.
[23]	**BUSY**	**Busy/Idle (Read Only)** 0 = EADC is in idle state. 1 = EADC is busy at conversion. **Note:** This flag will be high after 4*EADC_CLK cycles, when the trigger source is coming.
[22:21]	**Reserved**	Reserved.

Bits	Description	
[20:16]	**CHANNEL**	**Current Conversion Channel (Read Only)** This filed reflects EADC current conversion channel when BUSY=1. 00H = EADC_CH0. 01H = EADC_CH1. 02H = EADC_CH2. 03H = EADC_CH3. 04H = EADC_CH4. 05H = EADC_CH5. 06H = EADC_CH6. 07H = EADC_CH7. 08H = EADC_CH8. 09H = EADC_CH9. 0AH = EADC_CH10. 0BH = EADC_CH11. 0CH = EADC_CH12. 0DH = EADC_CH13. 0EH = EADC_CH14. 0FH = EADC_CH15. 10H = EADC_CH16. 11H = EADC_CH17. 12H = EADC_CH18. 13H = EADC_CH19. 14H = EADC_CH20. 15H = EADC_CH21. 16H = EADC_CH22. 17H = EADC_CH23. 18H = EADC_CH24. 19H = EADC_CH25. 1AH = EADC_CH26. 1BH = AV_{DD}/4. 1CH = VBG. 1DH = VTEMP . 1EH = V_{BAT}/4.
[15]	**ADCMPO3**	**EADC Compare 3 Output Status (Read Only)** The 12 bits compare3 data CMPDAT3 (EADC_CMP3[27:16]) is used to compare with conversion result of specified sample module. User can use it to monitor the external analog input pin voltage status. 0 = Conversion result in EADC_DAT less than CMPDAT3 setting. 1 = Conversion result in EADC_DAT great than or equal to CMPDAT3 setting.
[14]	**ADCMPO2**	**EADC Compare 2 Output Status (Read Only)** The 12 bits compare2 data CMPDAT2 (EADC_CMP2[27:16]) is used to compare with conversion result of specified sample module. User can use it to monitor the external analog input pin voltage status. 0 = Conversion result in EADC_DAT less than CMPDAT2 setting. 1 = Conversion result in EADC_DAT great than or equal to CMPDAT2 setting.

Bits	Description	
[13]	**ADCMPO1**	**EADC Compare 1 Output Status (Read Only)** The 12 bits compare1 data CMPDAT1 (EADC_CMP1[27:16]) is used to compare with conversion result of specified sample module. User can use it to monitor the external analog input pin voltage status. 0 = Conversion result in EADC_DAT less than CMPDAT1 setting. 1 = Conversion result in EADC_DAT great than or equal to CMPDAT1 setting.
[12]	**ADCMPO0**	**EADC Compare 0 Output Status (Read Only)** The 12 bits compare0 data CMPDAT0 (EADC_CMP0[27:16]) is used to compare with conversion result of specified sample module. User can use it to monitor the external analog input pin voltage status. 0 = Conversion result in EADC_DAT less than CMPDAT0 setting. 1 = Conversion result in EADC_DAT great than or equal CMPDAT0 setting.
[11]	**ADOVIF3**	**EADC ADINT3 Interrupt Flag Overrun** 0 = ADINT3 interrupt flag is not overwritten to 1. 1 = ADINT3 interrupt flag is overwritten to 1. **Note:** This bit is cleared by writing 1 to it.
[10]	**ADOVIF2**	**EADC ADINT2 Interrupt Flag Overrun** 0 = ADINT2 interrupt flag is not overwritten to 1. 1 = ADINT2 interrupt flag is s overwritten to 1. **Note:** This bit is cleared by writing 1 to it.
[9]	**ADOVIF1**	**EADC ADINT1 Interrupt Flag Overrun** 0 = ADINT1 interrupt flag is not overwritten to 1. 1 = ADINT1 interrupt flag is overwritten to 1. **Note:** This bit is cleared by writing 1 to it.
[8]	**ADOVIF0**	**EADC ADINT0 Interrupt Flag Overrun** 0 = ADINT0 interrupt flag is not overwritten to 1. 1 = ADINT0 interrupt flag is overwritten to 1. **Note:** This bit is cleared by writing 1 to it.
[7]	**ADCMPF3**	**EADC Compare 3 Flag** When the specific sample module EADC conversion result meets setting condition in EADC_CMP3 then this bit is set to 1. 0 = Conversion result in EADC_DAT does not meet EADC_CMP3 register setting. 1 = Conversion result in EADC_DAT meets EADC_CMP3 register setting. **Note:** This bit is cleared by writing 1 to it.
[6]	**ADCMPF2**	**EADC Compare 2 Flag** When the specific sample module EADC conversion result meets setting condition in EADC_CMP2 then this bit is set to 1. 0 = Conversion result in EADC_DAT does not meet EADC_CMP2 register setting. 1 = Conversion result in EADC_DAT meets EADC_CMP2 register setting. **Note:** This bit is cleared by writing 1 to it.
[5]	**ADCMPF1**	**EADC Compare 1 Flag** When the specific sample module EADC conversion result meets setting condition in EADC_CMP1 then this bit is set to 1. 0 = Conversion result in EADC_DAT does not meet EADC_CMP1 register setting. 1 = Conversion result in EADC_DAT meets EADC_CMP1 register setting. **Note:** This bit is cleared by writing 1 to it.

Bits	Description	
[4]	**ADCMPF0**	**EADC Compare 0 Flag** When the specific sample module EADC conversion result meets setting condition in EADC_CMP0 then this bit is set to 1. 0 = Conversion result in EADC_DAT does not meet EADC_CMP0 register setting. 1 = Conversion result in EADC_DAT meets EADC_CMP0 register setting. **Note:** This bit is cleared by writing 1 to it.
[3]	**ADIF3**	**EADC ADINT3 Interrupt Flag** 0 = No ADINT3 interrupt pulse received. 1 = ADINT3 interrupt pulse has been received. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** This bit indicates whether an EADC conversion of specific sample module has been completed
[2]	**ADIF2**	**EADC ADINT2 Interrupt Flag** 0 = No ADINT2 interrupt pulse received. 1 = ADINT2 interrupt pulse has been received. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** This bit indicates whether an EADC conversion of specific sample module has been completed
[1]	**ADIF1**	**EADC ADINT1 Interrupt Flag** 0 = No ADINT1 interrupt pulse received. 1 = ADINT1 interrupt pulse has been received. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** This bit indicates whether an EADC conversion of specific sample module has been completed
[0]	**ADIF0**	**EADC ADINT0 Interrupt Flag** 0 = No ADINT0 interrupt pulse received. 1 = ADINT0 interrupt pulse has been received. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** This bit indicates whether an EADC conversion of specific sample module has been completed

EADC Status Register 3 (EADC_STATUS3)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS3	EADC_BA+0xFC	R	EADC Status Register 3	0x0000_001F

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			CURSPL				

Bits	Description	
[31:5]	Reserved	Reserved.
[4:0]	CURSPL	**EADC Current Sample Module (Read Only)** This register shows the current EADC is controlled by which sample module control logic modules. If the EADC is Idle, the bit filed will set to 0x1F.

EADC Double Data Register n for Sample Module n (EADC_DDAT0~3)

Register	Offset	R/W	Description	Reset Value
EADC_DDAT0	EADC_BA+0x100	R	EADC Double Data Register 0 for Sample Module 0	0x0000_0000
EADC_DDAT1	EADC_BA+0x104	R	EADC Double Data Register 1 for Sample Module 1	0x0000_0000
EADC_DDAT2	EADC_BA+0x108	R	EADC Double Data Register 2 for Sample Module 2	0x0000_0000
EADC_DDAT3	EADC_BA+0x10C	R	EADC Double Data Register 3 for Sample Module 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						VALID	OV
15	14	13	12	11	10	9	8
RESULT							
7	6	5	4	3	2	1	0
RESULT							

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**Valid Flag** 0 = Double data in RESULT (EADC_DDATn[15:0]) is not valid. 1 = Double data in RESULT (EADC_DDATn[15:0]) is valid. This bit is set to 1 when corresponding sample module channel analog input conversion is completed and cleared by hardware after EADC_DDATn register is read. (n=0~3).
[16]	**OV**	**Overrun Flag** 0 = Double Data in RESULT (EADC_DDATn[15:0], n=0~3) is recent conversion result. 1 = Double Data in RESULT (EADC_DDATn[15:0], n=0~3) is overwrite. If converted data in RESULT[15:0] has not been read before new conversion result is loaded to this register, OV is set to 1. It is cleared by hardware after EADC_DDAT register is read.
[15:0]	**RESULT**	**EADC Conversion Results** This field contains 12 bits conversion results. The 12-bit EADC conversion result with unsigned format will be filled in RESULT [11:0] and zero will be filled in RESULT [15:12].

EADC Calibration Control Register (EADC_CALCTL)

Register	Offset	R/W	Description	Reset Value
EADC_CALCTL	EADC_BA+0x114	R/W	EADC Calibration Control Register	0x0000_0020

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**CALIE**	**CAL**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CALIE**	**Calibration Interrupt Enable Bit** 0 = Calibration interrupt Disabled. 1 = Calibration interrupt Enabled.
[0]	**CAL**	**Calibration Enable Bit** 0 = Calibration Disabled. 1 = Calibration Enabled. **Note:** This bit is hardware auto cleared when calibration is done

EADC Calibration Status Register (EADC_CALSR)

Register	Offset	R/W	Description	Reset Value
EADC_CALSR	EADC_BA+0x118	R/W	EADC Calibration Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**CALIF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**CALIF**	**Calibration Finish Interrupt Flag** If calibration is finished, this flag will be set to 1. It is cleared by writing 1 to it.
[15:0]	**Reserved**	Reserved.

EADC PDMA Control Register (EADC_PDMACTL)

Register	Offset	R/W	Description	Reset Value
EADC_PDMACTL	EADC_BA+0x130	R/W	EADC PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**PDMATEN**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PDMATEN							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PDMATEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PDMATEN							

Bits	Description	
[31]	**Reserved**	Reserved.
[30:0]	**PDMATEN**	**PDMA Transfer Enable Bit** When EADC conversion is completed, the converted data is loaded into EADC_DATn (n: 0 ~ 30) register, user can enable this bit to generate a PDMA data transfer request. 0 = PDMA data transfer Disabled. 1 = PDMA data transfer Enabled. **Note:** When setting this bit field to 1, user must set ADCIENn (EADC_CTL[5:2], n=0~3) = 0 to disable interrupt.

EADC Sample Module 0~3 Control Registers1 (EADC_M0CTL1~EADC_M3CTL1)

Register	Offset	R/W	Description	Reset Value
EADC_M0CTL1	EADC_BA+0x140	R/W	EADC Sample Module 0 Control Register 1	0x0000_0000
EADC_M1CTL1	EADC_BA+0x144	R/W	EADC Sample Module 1 Control Register 1	0x0000_0000
EADC_M2CTL1	EADC_BA+0x148	R/W	EADC Sample Module 2 Control Register 1	0x0000_0000
EADC_M3CTL1	EADC_BA+0x14C	R/W	EADC Sample Module 3 Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			DBMEN	Reserved		EXTSTDIV	
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ACU				Reserved		AVG	ALIGN

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**DBMEN**	**Double Buffer Mode Enable Bit** 0 = Sample has one sample result register (default). 1 = Sample has two sample result registers.
[19:18]	**Reserved**	Reserved.
[17:16]	**EXTSTDIV**	**EADC Extended Sampling Time Clock Divider Selection** Clock frequency for extending sampling time: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.
[15:8]	**Reserved**	Reserved.

Bits	Description	
[7:4]	**ACU**	**Number of Accumulated Conversion Results Selection** 0000 = 1 conversion result will be accumulated. 0001 = 2 conversion result will be accumulated. 0010 = 4 conversion result will be accumulated. 0011 = 8 conversion result will be accumulated. 0100 = 16 conversion result will be accumulated. 0101 = 32 conversion result will be accumulated. 0110 = 64 conversion result will be accumulated. 0111 = 128 conversion result will be accumulated. 1000 = 256 conversion result will be accumulated. Others = Reserved.
[3:2]	**Reserved**	Reserved.
[1]	**AVG**	**Average Mode Selection** 0 = Conversion results will be stored in data register without averaging. 1 = Conversion results in data register will be averaged. **Note:** This bit needs to work with ACU (EADC_MnCTL1[7:4], n=0~26).
[0]	**ALIGN**	**Alignment Selection** 0 = The conversion result will be right aligned in data register. 1 = The conversion result will be left aligned in data register.

EADC Sample Module 4~18 Control Registers1 (EADC_M4CTL1~EADC_M18CTL1)

Register	Offset	R/W	Description	Reset Value
EADC_M4CTL1	EADC_BA+0x150	R/W	EADC Sample Module 4 Control Register 1	0x0000_0000
EADC_M5CTL1	EADC_BA+0x154	R/W	EADC Sample Module 5 Control Register 1	0x0000_0000
EADC_M6CTL1	EADC_BA+0x158	R/W	EADC Sample Module 6 Control Register 1	0x0000_0000
EADC_M7CTL1	EADC_BA+0x15C	R/W	EADC Sample Module 7 Control Register 1	0x0000_0000
EADC_M8CTL1	EADC_BA+0x160	R/W	EADC Sample Module 8 Control Register 1	0x0000_0000
EADC_M9CTL1	EADC_BA+0x164	R/W	EADC Sample Module 9 Control Register 1	0x0000_0000
EADC_M10CTL1	EADC_BA+0x168	R/W	EADC Sample Module 10 Control Register 1	0x0000_0000
EADC_M11CTL1	EADC_BA+0x16C	R/W	EADC Sample Module 11 Control Register 1	0x0000_0000
EADC_M12CTL1	EADC_BA+0x170	R/W	EADC Sample Module 12 Control Register 1	0x0000_0000
EADC_M13CTL1	EADC_BA+0x174	R/W	EADC Sample Module 13 Control Register 1	0x0000_0000
EADC_M14CTL1	EADC_BA+0x178	R/W	EADC Sample Module 14 Control Register 1	0x0000_0000
EADC_M15CTL1	EADC_BA+0x17C	R/W	EADC Sample Module 15 Control Register 1	0x0000_0000
EADC_M16CTL1	EADC_BA+0x180	R/W	EADC Sample Module 16 Control Register 1	0x0000_0000
EADC_M17CTL1	EADC_BA+0x184	R/W	EADC Sample Module 17 Control Register 1	0x0000_0000
EADC_M18CTL1	EADC_BA+0x188	R/W	EADC Sample Module 18 Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**EXTSTDIV**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACU				**Reserved**		**AVG**	**ALIGN**

Bits	Description	
[31:18]	**Reserved**	Reserved.

Bits	Description	
[17:16]	**EXTSTDIV**	**EADC Extended Sampling Time Clock Divider Selection** Clock frequency for extending sampling time: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.
[15:8]	**Reserved**	Reserved.
[7:4]	**ACU**	**Number of Accumulated Conversion Results Selection** 0000 = 1 conversion result will be accumulated. 0001 = 2 conversion result will be accumulated. 0010 = 4 conversion result will be accumulated. 0011 = 8 conversion result will be accumulated. 0100 = 16 conversion result will be accumulated. 0101 = 32 conversion result will be accumulated. 0110 = 64 conversion result will be accumulated. 0111 = 128 conversion result will be accumulated. 1000 = 256 conversion result will be accumulated. Others = Reserved.
[3:2]	**Reserved**	Reserved.
[1]	**AVG**	**Average Mode Selection** 0 = Conversion results will be stored in data register without averaging. 1 = Conversion results in data register will be averaged. **Note:** This bit needs to work with ACU (EADC_MnCTL1[7:4], n=0~26).
[0]	**ALIGN**	**Alignment Selection** 0 = The conversion result will be right aligned in data register. 1 = The conversion result will be left aligned in data register.

EADC Data Registers (EADC_DAT19~ EADC_DAT30)

Register	Offset	R/W	Description	Reset Value
EADC_DAT19	EADC_BA+0x200	R	EADC Data Register 19 for Sample Module 19	0x0000_0000
EADC_DAT20	EADC_BA+0x204	R	EADC Data Register 20 for Sample Module 20	0x0000_0000
EADC_DAT21	EADC_BA+0x208	R	EADC Data Register 21 for Sample Module 21	0x0000_0000
EADC_DAT22	EADC_BA+0x20C	R	EADC Data Register 22 for Sample Module 22	0x0000_0000
EADC_DAT23	EADC_BA+0x210	R	EADC Data Register 23 for Sample Module 23	0x0000_0000
EADC_DAT24	EADC_BA+0x214	R	EADC Data Register 24 for Sample Module 24	0x0000_0000
EADC_DAT25	EADC_BA+0x218	R	EADC Data Register 25 for Sample Module 25	0x0000_0000
EADC_DAT26	EADC_BA+0x21C	R	EADC Data Register 26 for Sample Module 26	0x0000_0000
EADC_DAT27	EADC_BA+0x220	R	EADC Data Register 27 for Sample Module 27	0x0000_0000
EADC_DAT28	EADC_BA+0x224	R	EADC Data Register 28 for Sample Module 28	0x0000_0000
EADC_DAT29	EADC_BA+0x228	R	EADC Data Register 29 for Sample Module 29	0x0000_0000
EADC_DAT30	EADC_BA+0x22C	R	EADC Data Register 30 for Sample Module 30	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**VALID**	**OV**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RESULT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RESULT							

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**Valid Flag** This bit is set to 1 when corresponding sample module channel analog input conversion is completed and cleared by hardware after EADC_DAT register is read. 0 = Data in RESULT[11:0] bits is not valid. 1 = Data in RESULT[11:0] bits is valid.

[16]	**OV**	**Overrun Flag** If converted data in RESULT[11:0] has not been read before new conversion result is loaded to this register, OV is set to 1. 0 = Data in RESULT[11:0] is recent conversion result. 1 = Data in RESULT[11:0] is overwrite. **Note:** It is cleared by hardware after EADC_DAT register is read.
[15:0]	**RESULT**	**EADC Conversion Result** This field contains 12 bits conversion result. The 12-bit EADC conversion result with unsigned format will be filled in RESULT[11:0] and zero will be filled in RESULT[15:12]. **Note:** When operating in oversampling mode, RESULT[15:0] can represent oversampling results.

EADC Sample Module 19~26 Control Registers (EADC_SCTL19~EADC_SCTL26)

Register	Offset	R/W	Description	Reset Value
EADC_SCTL19	EADC_BA+0x230	R/W	EADC Sample Module 19 Control Register	0x0000_0014
EADC_SCTL20	EADC_BA+0x234	R/W	EADC Sample Module 20 Control Register	0x0000_0014
EADC_SCTL21	EADC_BA+0x238	R/W	EADC Sample Module 21 Control Register	0x0000_0014
EADC_SCTL22	EADC_BA+0x23C	R/W	EADC Sample Module 22 Control Register	0x0000_0014
EADC_SCTL23	EADC_BA+0x240	R/W	EADC Sample Module 23 Control Register	0x0000_0014
EADC_SCTL24	EADC_BA+0x244	R/W	EADC Sample Module 24 Control Register	0x0000_0014
EADC_SCTL25	EADC_BA+0x248	R/W	EADC Sample Module 25 Control Register	0x0000_0014
EADC_SCTL26	EADC_BA+0x24C	R/W	EADC Sample Module 26 Control Register	0x0000_0014

31	30	29	28	27	26	25	24
EXTSMPT							
23	22	21	20	19	18	17	16
EXTFEN	EXTREN	TRGSEL					
15	14	13	12	11	10	9	8
TRGDLCNT							
7	6	5	4	3	2	1	0
TRGDLDIV		INTPOS	CHSEL				

Bits	Description	
[31:24]	**EXTSMPT**	**EADC Sampling Time Extend** When EADC converting at high conversion rate, the sampling time of analog input voltage may not be enough if input channel loading is heavy, and software can extend EADC sampling time after trigger source is coming to get enough sampling time. Extended Sampling Time = (EXTSMPT x EADC_CLK period x n)+2 [n=1,2,4,16 from EXTSTDIV setting].
[23]	**EXTFEN**	**EADC External Trigger Falling Edge Enable Bit** 0 = Falling edge Disabled when EADC selects EADC0_ST as trigger source. 1 = Falling edge Enabled when EADC selects EADC0_ST as trigger source.
[22]	**EXTREN**	**EADC External Trigger Rising Edge Enable Bit** 0 = Rising edge Disabled when EADC selects EADC0_ST as trigger source. 1 = Rising edge Enabled when EADC selects EADC0_ST as trigger source.

Bits	Description	
[21:16]	**TRGSEL**	**EADC Sample Module Start of Conversion Trigger Source Selection** 0H = Disable trigger. 1H = External trigger from EADC0_ST pin input. 2H = EADC ADINT0 interrupt EOC pulse trigger. 3H = EADC ADINT1 interrupt EOC pulse trigger. 4H = Timer0 overflow pulse trigger. 5H = Timer1 overflow pulse trigger. 6H = Timer2 overflow pulse trigger. 7H = Timer3 overflow pulse trigger. 8H = EPWM0TG0. 9H = EPWM0TG1. AH = EPWM0TG2. BH = EPWM0TG3. CH = EPWM0TG4. DH = EPWM0TG5. EH = EPWM1TG0. FH = EPWM1TG1. 10H = EPWM1TG2. 11H = EPWM1TG3. 12H = EPWM1TG4. 13H = EPWM1TG5. 14H = PWM0TG0. 15H = PWM0TG1. 16H = PWM0TG2. 17H = PWM0TG3. 18H = PWM0TG4. 19H = PWM0TG5. 1AH = PWM1TG0. 1BH = PWM1TG1. 1CH = PWM1TG2. 1DH = PWM1TG3. 1EH = PWM1TG4. 1FH = PWM1TG5. 20H = ACMP0_INT. 21H = ACMP1_INT. 22H = ACMP2_INT. other = Reserved.
[15:8]	**TRGDLCNT**	**EADC Sample Module Start of Conversion Trigger Delay Time** Trigger delay time = TRGDLCNT x EADC_CLK period x n (n=1,2,4,16 from TRGDLDIV setting).
[7:6]	**TRGDLDIV**	**EADC Sample Module Start of Conversion Trigger Delay Clock Divider Selection** Trigger delay clock frequency: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.

Bits	Description	
[5]	**INTPOS**	**Interrupt Flag Position Select** 0 = Set ADIFn (EADC_STATUS2[n], n=0~3) at EADC end of conversion. 1 = Set ADIFn (EADC_STATUS2[n], n=0~3) at EADC start of conversion.
[4:0]	**CHSEL**	**EADC Sample Module Channel Selection** 00H = EADC_CH0. 01H = EADC_CH1. 02H = EADC_CH2. 03H = EADC_CH3. 04H = EADC_CH4. 05H = EADC_CH5. 06H = EADC_CH6. 07H = EADC_CH7. 08H = EADC_CH8. 09H = EADC_CH9. 0AH = EADC_CH10. 0BH = EADC_CH11. 0CH = EADC_CH12. 0DH = EADC_CH13. 0EH = EADC_CH14. 0FH = EADC_CH15. 10H = EADC_CH16. 11H = EADC_CH17. 12H = EADC_CH18. 13H = EADC_CH19. 14H = EADC_CH20. 15H = EADC_CH21. 16H = EADC_CH22. 17H = EADC_CH23. 18H = EADC_CH24. 19H = EADC_CH25. 1AH = EADC_CH26. Others = Reserved.

EADC Sample Module 27~30 Control Registers (EADC_SCTL27~EADC_SCTL30)

Register	Offset	R/W	Description	Reset Value
EADC_SCTL27	EADC_BA+0x250	R/W	EADC Sample Module 27 Control Register	0x0000_0000
EADC_SCTL28	EADC_BA+0x254	R/W	EADC Sample Module 28 Control Register	0x0000_0000
EADC_SCTL29	EADC_BA+0x258	R/W	EADC Sample Module 29 Control Register	0x0000_0000
EADC_SCTL30	EADC_BA+0x25C	R/W	EADC Sample Module 30 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**EXTSMPT**	**ADC Sampling Time Extend** When EADC converting at high conversion rate, the sampling time of analog input voltage may not be enough if input channel loading is heavy, and software can extend EADC sampling time after trigger source is coming to get enough sampling time. Extended Sampling Time = (EXTSMPT x EADC_CLK period x n)+2 [n=1,2,4,16 from EXTSTDIV setting].
[23:0]	**Reserved**	Reserved.

EADC Sample Module 19~26 Control Registers1 (EADC_M19CTL1~EADC_M26CTL1)

Register	Offset	R/W	Description	Reset Value
EADC_M19CTL1	EADC_BA+0x260	R/W	EADC Sample Module 19 Control Register 1	0x0000_0000
EADC_M20CTL1	EADC_BA+0x264	R/W	EADC Sample Module 20 Control Register 1	0x0000_0000
EADC_M21CTL1	EADC_BA+0x268	R/W	EADC Sample Module 21 Control Register 1	0x0000_0000
EADC_M22CTL1	EADC_BA+0x26C	R/W	EADC Sample Module 22 Control Register 1	0x0000_0000
EADC_M23CTL1	EADC_BA+0x270	R/W	EADC Sample Module 23 Control Register 1	0x0000_0000
EADC_M24CTL1	EADC_BA+0x274	R/W	EADC Sample Module 24 Control Register 1	0x0000_0000
EADC_M25CTL1	EADC_BA+0x278	R/W	EADC Sample Module 25 Control Register 1	0x0000_0000
EADC_M26CTL1	EADC_BA+0x27C	R/W	EADC Sample Module 26 Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						EXTSTDIV	
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ACU				Reserved		AVG	ALIGN

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**EXTSTDIV**	**EADC Extended Sampling Time Clock Divider Selection** Clock frequency for extending sampling time: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.
[15:8]	**Reserved**	Reserved.

Bits	Description	
[7:4]	**ACU**	**Number of Accumulated Conversion Results Selection** 0000 = 1 conversion result will be accumulated. 0001 = 2 conversion result will be accumulated. 0010 = 4 conversion result will be accumulated. 0011 = 8 conversion result will be accumulated. 0100 = 16 conversion result will be accumulated. 0101 = 32 conversion result will be accumulated. 0110 = 64 conversion result will be accumulated. 0111 = 128 conversion result will be accumulated. 1000 = 256 conversion result will be accumulated. Others = Reserved.
[3:2]	**Reserved**	Reserved.
[1]	**AVG**	**Average Mode Selection** 0 = Conversion results will be stored in data register without averaging. 1 = Conversion results in data register will be averaged. **Note:** This bit needs to work with ACU (EADC_MnCTL1[7:4], n=0~26).
[0]	**ALIGN**	**Alignment Selection** 0 = The conversion result will be right aligned in data register. 1 = The conversion result will be left aligned in data register.

EADC Sample Module 27~30 Control Registers1 (EADC_M27CTL1~EADC_M30CTL1)

Register	Offset	R/W	Description	Reset Value
EADC_M27CTL1	EADC_BA+0x280	R/W	EADC Sample Module 27 Control Register 1	0x0000_0000
EADC_M28CTL1	EADC_BA+0x284	R/W	EADC Sample Module 28 Control Register 1	0x0000_0000
EADC_M29CTL1	EADC_BA+0x288	R/W	EADC Sample Module 29 Control Register 1	0x0000_0000
EADC_M30CTL1	EADC_BA+0x28C	R/W	EADC Sample Module 30 Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						EXTSTDIV	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**EXTSTDIV**	**EADC Extended Sampling Time Clock Divider Selection** Clock frequency for extending sampling time: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.
[15:0]	**Reserved**	Reserved.

EADC Test Mode Control Register (EADC_TEST) for Test, Not Release to TRM

Register	Offset	R/W	Description	Reset Value
EADC_TEST	EADC_BA+0xFF0	R/W	EADC Test Mode Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**DECADD**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**DECADD**	**EADC Decode Add one cycle** 0 = Decode leep original cycle. 1 = Decode ADD one cycle.
[7:0]	**Reserved**	Reserved.

6.43 Low Power Analog-to-Digital Converter (LPADC)

6.43.1 Overview

LPADC contains one 12-bit successive approximation analog-to-digital converter (SAR A/D converter) with 24 input channels. The A/D converter supports four operation modes: Single, Single-cycle Scan and Continuous Scan mode. The A/D converter can be started by software, external pin (LPADC0_ST), Timer 0~3 overflow pulse trigger, EPWM trigger, PWM trigger and ACMP0_O/ACMP1_O/ACMP2_O edge event. The A/D converter can also be triggered to start by software write 1 to SWTRIG (LPADC_AUTOSTRG[0]), Low power timer 0/1, Tick timer, Wake-up I/O or ACMP at auto-operation mode

6.43.2 Features

- Operating voltage: 1.8V~3.6V
- Analog input voltage: 0 ~ AV_{DD}
- Supports external reference voltage from V_{REF} pin.
- 12-bit resolution and 10-bit accuracy guaranteed
- Up to 24 single-end analog input channels or 12 differential analog input channels
- Maximum LPADC peripheral clock (LPADC_CLK) frequency is 12 MHz
- Up to 571 KSPS sampling rate
- Scan on enabled channels
- Threshold voltage detection
- Three operation modes:
  - Single mode: A/D conversion is performed one time on a specified channel.
  - Single-cycle Scan mode: A/D conversion is performed only one cycle on all specified channels with the sequence from the smallest numbered channel to the largest numbered channel.
  - Continuous Scan mode: A/D converter continuously performs Single-cycle Scan mode until software stops A/D conversion.
- An A/D conversion can be started by:
  - Software Write 1 to ADST bit
  - External pin (LPADC0_ST)
  - Timer 0~3 overflow pulse trigger
  - EPWM trigger
  - PWM trigger
  - ACMP0_O/ ACMP1_O/ ACMP2_O edge event
- Supports auto-operation mode with A/D conversion that can be started by:
  - Software Write 1 to SWTRIG bit
  - Low power Timer 0/1
  - Tick Timer 0/1
  - ACMP analog ouput 0/1(ACMPS0/ ACMPS1) edge event
- Each conversion result held in data register of each channel with valid and overrun indicators

- Conversion result can be compared with specified value and user can select whether to generate an interrupt when conversion result matches the compare register setting.
- Supports extend sample time function (0~16383 LPADC clock)
- Seven internal channels from band-gap voltage (V_{BG}), temperature sensor (V_{TEMP}), V_{BAT} voltage divided by 4, AV_{DD} voltage divided by 4 and three OPAs.
- Supports LPPDMA transfer mode
- Supports Calibration mode
- Supports Floating Detect Function

Note 1: LPADC sampling rate = (LPADC peripheral clock frequency) / (total LPADC conversion cycle)

Note 2: If the internal channel for band-gap voltage is active, the maximum sampling rate will be 100 KSPS.

Note 3: LPADC peripheral clock frequency must be slower than or equal to PCLK2.

6.43.3 Block Diagram

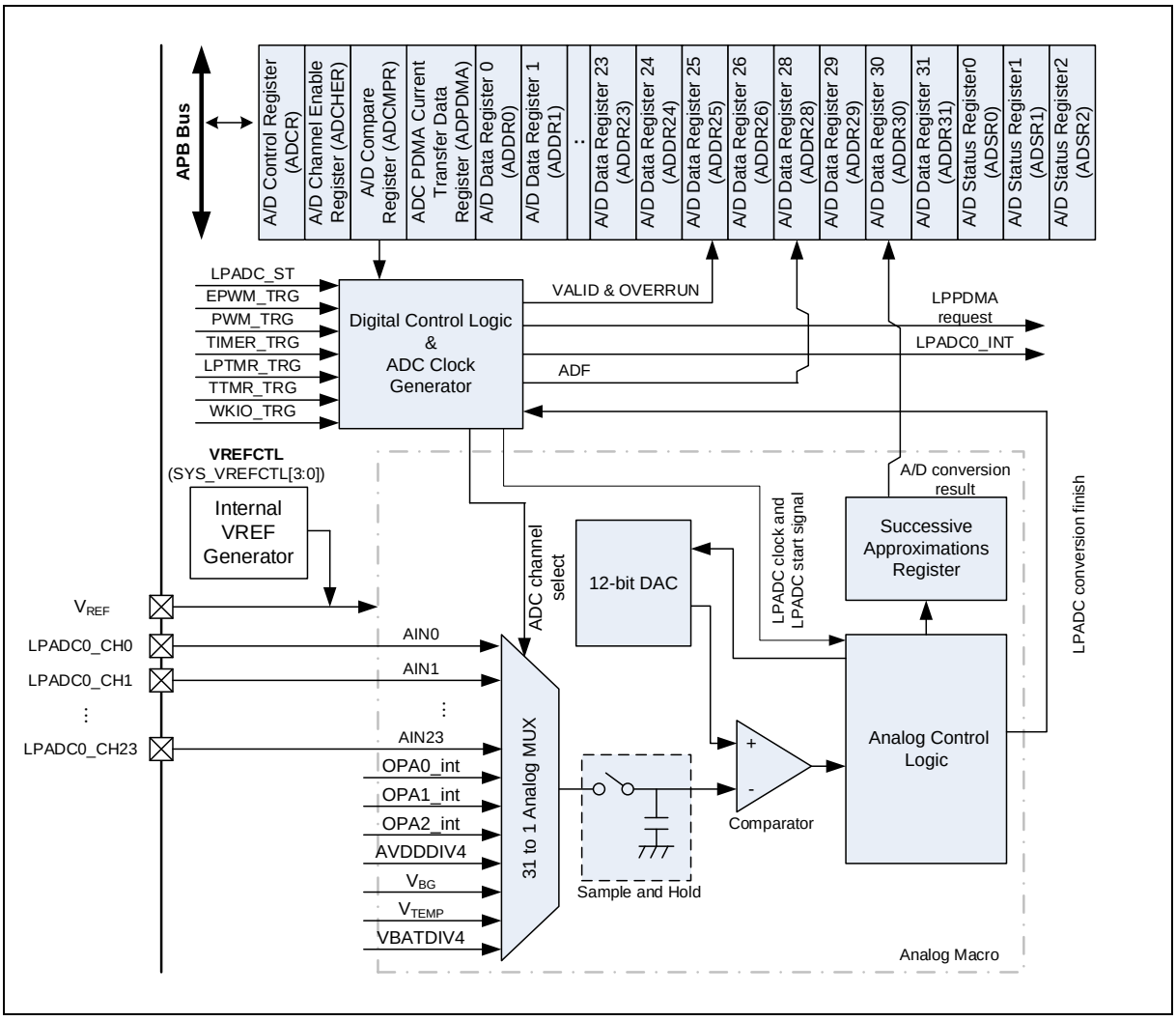


Figure 6.43-1 LPADC Controller Block Diagram

6.43.4 Basic Configuration

- Clock Source Configuration
  - Select the source of LPADC0 peripheral clock on LPADC0SEL(LPSCC_CLKSEL0 [17:16]).
  - Select the clock divider number of LPADC peripheral clock on LPADC0DIV (LPSCC_CLKDIV0[19:16]).
  - Enable LPADC0 peripheral clock in LPADC0CKEN (LPSCC_CLKEN0[24]).
- Reset Configuration
  - Reset LPADC in LPADC0RST (LPSCC_IPRST0[24]).
- Control Select Setting
  - Set ADCCSEL (SYS_IVSCTL[7]) to 1 to make LPADC0 can control the ADC analog macro.

6.43.5 Functional Description

The A/D converter operates by successive approximation with 12-bit resolution. LPADC has four operation modes: Single, Single-cycle Scan mode and Continuous Scan mode. When user wants to change the operation mode or analog input channel, in order to prevent incorrect operation, software must clear ADST(LPADC_ADCR[11]) bit to 0 in advance.

6.43.5.1 LPADC peripheral Clock Generator

The maximum sampling rate is up to 571 KSPS. LPADC has four clock sources selected by LPADC0SEL(LPSCC_CLKSEL0 [17:16]), the LPADC peripheral clock frequency is divided by an 4-bit prescaler with the following formula:

LPADC peripheral clock frequency = (LPADC peripheral clock source frequency) / (LPADC0DIV+1); where the 4-bit LPADC0DIV is located in register LPSCC_CLKDIV0[19:16].

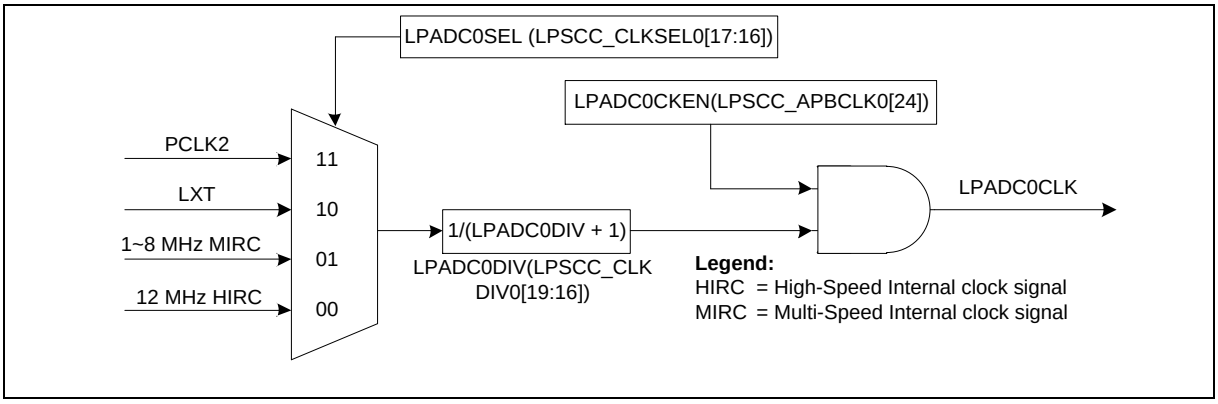


Figure 6.43-2 LPADC Peripheral Clock Control

6.43.5.2 Single Mode

In Single mode, A/D conversion is performed only once on the specified single channel. The operations are as follows:

1. A/D conversion will be started when the ADST bit of LPADC_ADCR register is set to 1 by software or external trigger input.
2. When A/D conversion is finished, the result is stored in the LPADC data register corresponding to the channel.
3. After A/D conversion is finished, the ADF bit of ADSR0 register will be set to 1. If the ADIE bit of LPADC_ADCR register is set to 1, the LPADC interrupt will be asserted.

4. The ADST bit remains 1 during A/D conversion. When A/D conversion ends, the ADST bit is automatically cleared to 0 and the A/D converter enters idle state.

Note 1: If software enables more than one channel in Single mode, only the channel with the smallest number will be selected and the other enabled channels will be ignored.

Note 2: If ADST bit is cleared to 0 before LPADC conversion done, LPADC cannot finish the current conversion, and the BUSY bit will be cleared to 0 immediately and A/D converter enters idle state directly.

An example timing diagram for Single mode is shown in Figure 6.43-3.

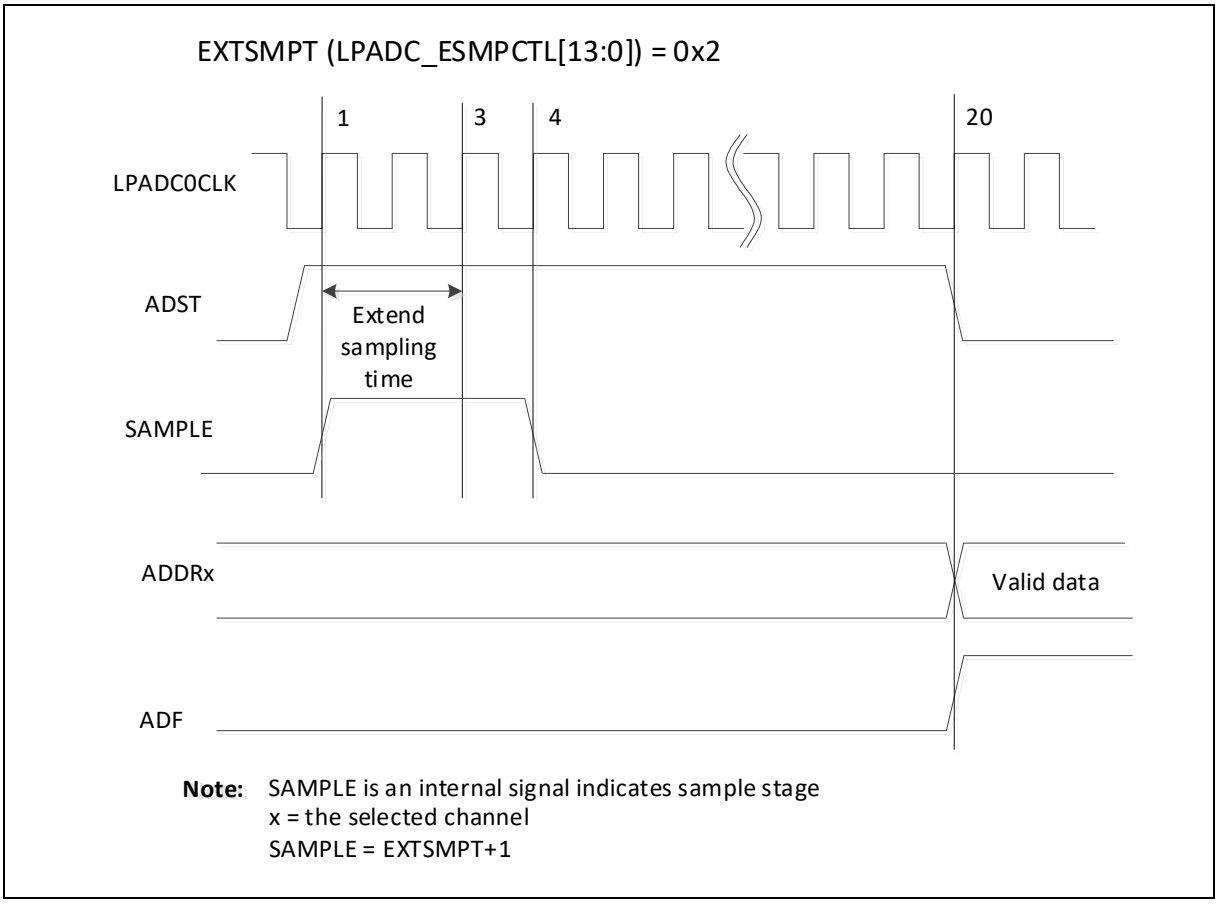


Figure 6.43-3 Single Mode Conversion Timing Diagram

6.43.5.3 Single-Cycle Scan Mode

In Single-cycle Scan mode, A/D converter samples and converts all of the specified channels once in the sequence from the smallest number enabled channel to the largest number enabled channel. Operations are as follows:

1. When the ADST bit in LPADC_ADCR register is set to 1 by software or external trigger input, A/D conversion is started on the enabled channel with the smallest number.
2. When A/D conversion for each enabled channel is completed, the result is sequentially transferred to the LPADC data register corresponding to each channel.
3. When the conversions of all the enabled channels are completed, the ADF bit in ADSR0 register is set to 1. If the LPADC interrupt function is enabled, the LPADC interrupt occurs.
4. After LPADC finishes one cycle conversion, the ADST bit is automatically cleared to 0 and the A/D converter enters idle state. If ADST bit is cleared to 0 before all enabled LPADC channels

conversion done, LPADC cannot finish the current conversion and A/D converter enters idle state directly.

An example timing diagram for Single-cycle Scan mode on enabled channels (0, 2, 3 and 7) is shown in Figure 6.43-4.

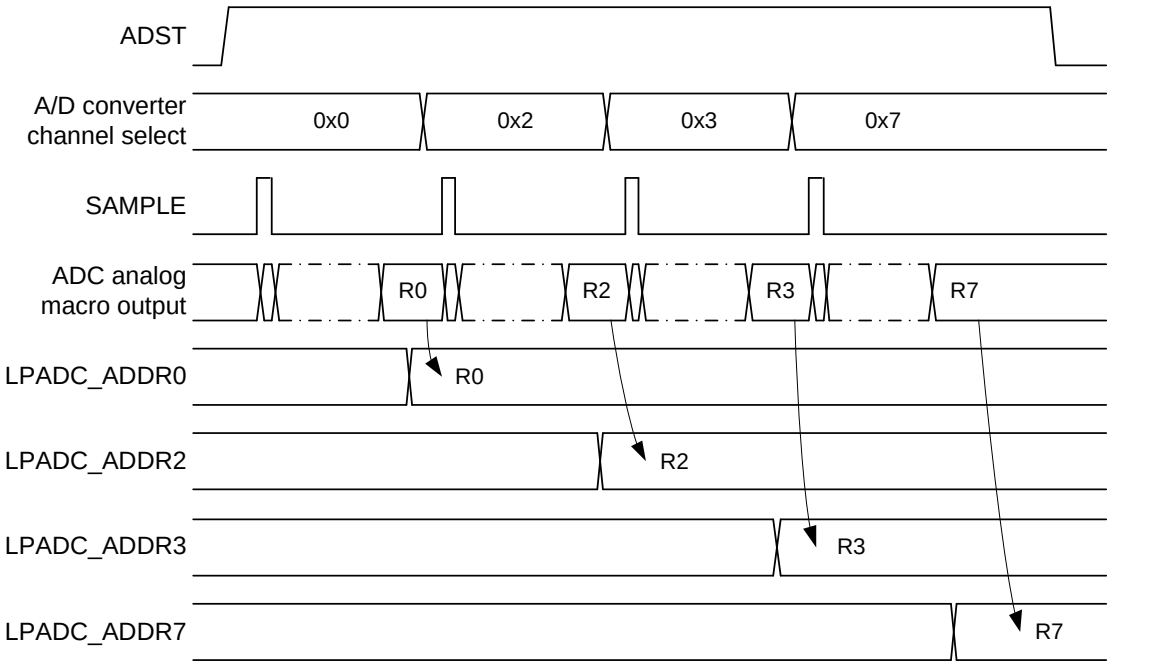


Figure 6.43-4 Single-Cycle Scan Mode on Enabled Channels Timing Diagram

6.43.5.4 A/D Extend Sampling Time

When A/D operates at high LPADC clock rate, the sampling time of analog input voltage may not be enough if the analog channel has heavy loading to cause fully charge time is longer. User can also set extend sampling time by writing EXTSMPT (LPADC_ESMPCTL[13:0]) for each sample module. The range of extend sampling time is from 0 ~16383 LPADC clock.

6.43.5.5 Continuous Scan Mode

In Continuous Scan mode, A/D conversion is performed sequentially on the specified channels that enabled by CHEN bits in LPADC_ADCHER register. The operations are as follows:

1. When the ADST bit in LPADC_ADCR register is set to 1 by software or external trigger input, A/D conversion is started on the enabled channel with the smallest number.
2. When A/D conversion for each enabled channel is completed, the result of each enabled channel is stored in the LPADC data register corresponding to each enabled channel.
3. When A/D converter completes the conversions of all enabled channels sequentially, the ADF bit in ADSR0 register will be set to 1. If the LPADC interrupt function is enabled, the LPADC interrupt occurs. The conversion of the enabled channel with the smallest number will start again if software does not clear the ADST bit.
4. As long as the ADST bit remains at 1, the step 2 ~ 3 will be repeated. When ADST is cleared to 0, LPADC cannot finish the current conversion and A/D converter enters idle state directly.

An example timing diagram for Continuous Scan mode on enabled channels (0, 2, 3 and 7) is shown in

Figure 6.43-5.

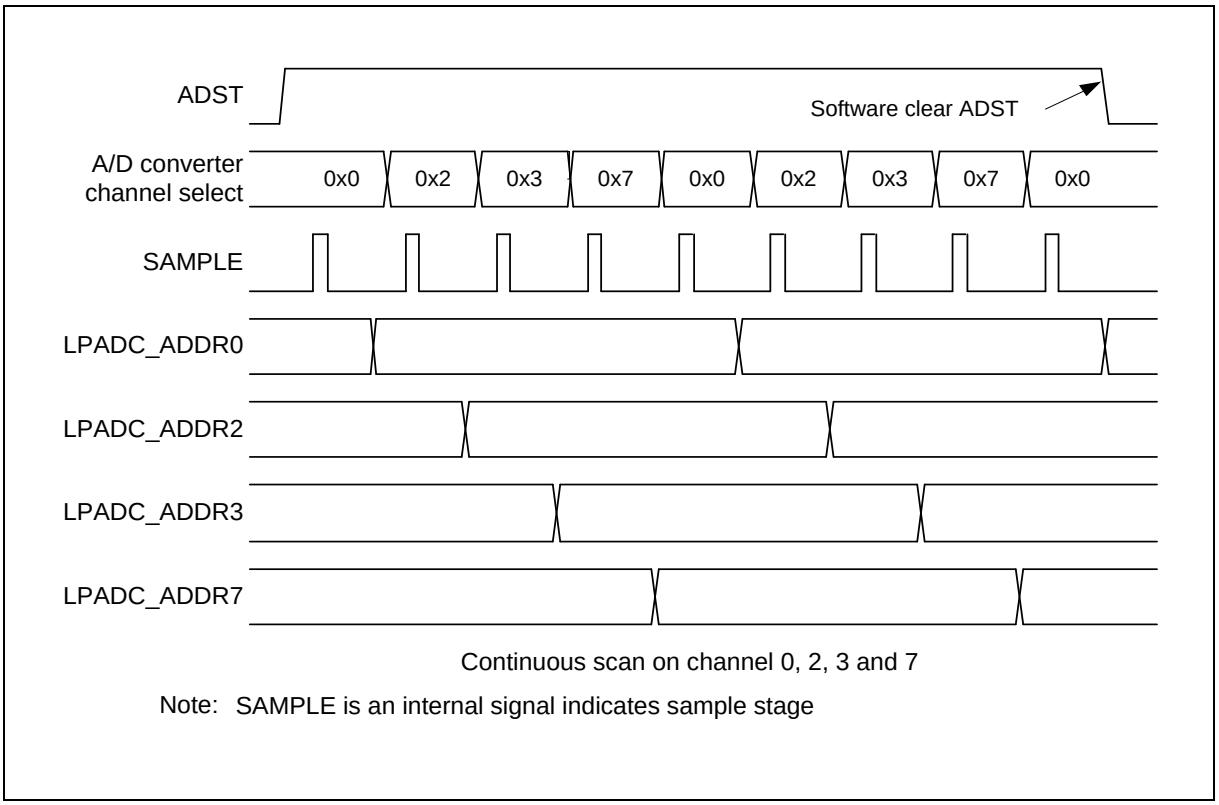


Figure 6.43-5 Continuous Scan Mode on Enabled Channels Timing Diagram

6.43.5.6 Calibration Mode

To decrease the effect of electrical random noise, the calibration mode performs an offset and mismatch measurement cycles. Afterwards, in normal operation mode, the calibration engine applies to the capacitor array, so that the offset and mismatch are removed. If chip power off, calibration function should be executed again.

In Calibration Mode:

1. Set CALEN(LPADC_ADCAL[0]) bit to 1.
2. When the ADST bit in LPADC_ADCR register is set to 1 by software, calibration is started.
3. After calibration is finished, the flag CALIF(LPADC_ADCALSTS[0]) will be set to 1.
4. After calibration is finished, the mode will become Normal Mode automatically.

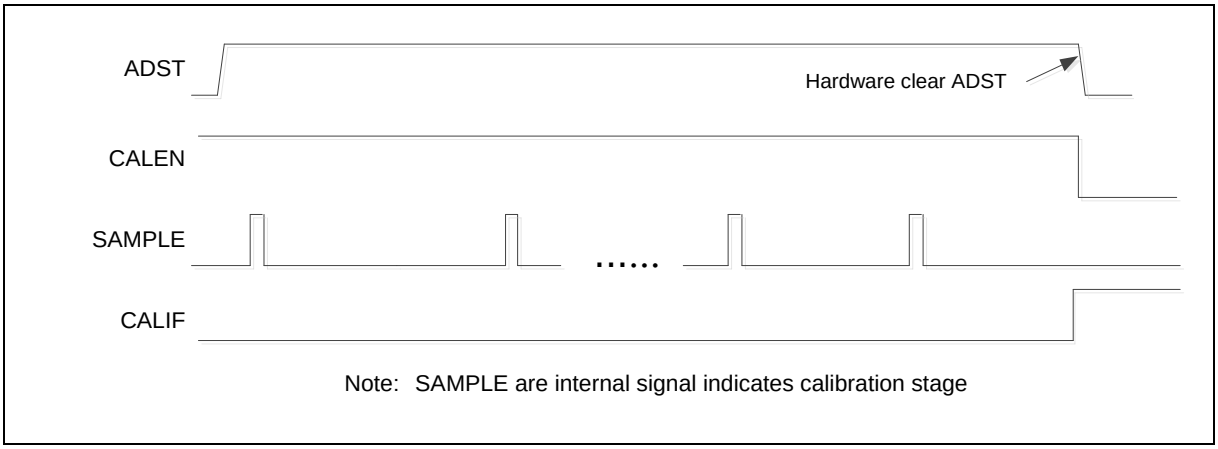


Figure 6.43-6 Calibration Mode

6.43.5.7 *External Trigger Input*

In Single-cycle Scan mode, A/D conversion can be triggered by external pin request. When the TRGEN bit of LPADC_ADCR register is set to 1 to enable LPADC external trigger function, setting the TRGS(LPADC_ADCR[19:16]) bits to 4'b0000 is to select external trigger input from the LPADC0_ST pin. Software can set TRGCOND(LPADC_ADCR[7:6]) to select trigger condition is falling/rising edge or low/high level. If level trigger condition is selected, the LPADC0_ST pin must be kept at specified state at least 8 PCLKs. The ADST bit will be set to 1 at the 9th PCLK and start to convert. Conversion will keep going if external trigger input is kept at active state in level trigger mode. It is stopped only when external condition trigger condition disappears. If edge trigger condition is selected, the high and low state must be kept at least 4 PCLKs. Pulse that is shorter than this specification will be ignored.

Note: User enables the external trigger function or enables LPADC must be at least 4 PCLKs after enabling LPADC peripheral clock.

6.43.5.8 *Internal Reference Voltage*

The band-gap voltage reference (V_{BG}) is an internal fixed reference regardless of the power supply variation. The V_{BG} output is internally connected to the LPADC channel multiplexer and analog comparator (ACMP) negative input side. Therefore, the voltage of AV_{DD} can be calculated by the A/D conversion result of V_{BG}.

V_{BG} can be used to calculate the AV_{DD} voltage when AV_{DD} is directly connected to the power supply. Compared to the voltage divider circuit, using V_{BG} can reduce the external components and power consumption.

The V_{BG} of each chip might be slightly different due to the process variation. Therefore, the calculated AV_{DD} might have some deviation. The chip has built-in V_{BG} A/D conversion value (when AV_{DD} = 3072 mV) when chips are shipped from the factory. With the built-in V_{BG} A/D conversion value and the current V_{BG} A/D conversion value, users can calculate AV_{DD} in a more accurately way. It is more accurate in comparison with using only the V_{BG} A/D conversion value.

The chip's built-in AV_{DD} divided by four circuit as LPADC internal channel for user to convert AV_{DD} voltage easily when V_{REF} source from external input voltage or internal reference voltage.

The chip's built-in temperature sensor output voltage is one source of LPADC internal channel. User can use LPADC conversion to know the chip internal temperature.

The chip's built-in V_{BAT} divided by four circuit as LPADC internal channel for user to convert V_{BAT} voltage easily. By conversion this internal channel, user can know V_{BAT} domain's voltage to check if battery is low voltage.

6.43.5.9 Timer Trigger

There are 4 Timer trigger sources for ADC. When the TRGEN bit of ADCR register is set to high to

enable ADC external hardware trigger function, setting the TRGS (ADCR[19:16]) bits to 4'b0001 is to select external hardware trigger input source from Timer trigger. The detail trigger conditions of Timer are described at TIMER0_CTL ~ TIMER3_CTL register.

6.43.5.10 EPWM Trigger

In all operation mode, A/D conversion can be triggered by EPWM request. When the TRGEN bit of ADCR register is set to high to enable ADC external hardware trigger function, setting the TRGS(ADCR[19:16]) bits to 4'b0100 is to select external hardware trigger input source from EPWM trigger.

6.43.5.11 PWM Trigger

In all operation mode, A/D conversion can be triggered by PWM request. When the TRGEN bit of ADCR register is set to high to enable ADC external hardware trigger function, setting the TRGS(ADCR[19:16]) bits to 4'b0011 is to select external hardware trigger input source from PWM trigger.

6.43.5.12 ACMP Trigger

In all ADC operation mode, A/D conversion can be triggered by ACMP. When the TRGEN bit of ADCR register is set to high to enable ADC external hardware trigger function, setting the TRGS(ADCR[19:16]) bits to 4'b1000 is to select external hardware trigger input source from ACMP0 trigger. Setting the TRGS(ADCR[19:16]) bits to 4'b1001 is to select external hardware trigger input source from ACMP1 trigger. Setting the TRGS(ADCR[19:16]) bits to 4'b1010 is to select external hardware trigger input source from ACMP2 trigger. Any ACMP0/1/2 trigger source can be set as both edge trigger by setting ACMPTES (ADCR[21:20]) bits to 00, rising edge trigger by setting ACMPTES (ADCR[21:20]) bits to 01 or falling edge trigger by setting ACMPTES (ADCR[21:20]) bits to 10.

6.43.5.13 Conversion Result Monitored by Compare Mode Function

LPADC provides two compare registers, ADCMPRx(x=0,1), to monitor maximum two specified channels. Software can select which channel to be monitored by setting CMPCH (ADCMPRx[7:3]). CMPCOND (ADCMPRx[2]) bit is used to determine the compare condition. If CMPCOND bit is cleared to 0, the internal match counter will increase one when the conversion result is less than the value specified in CMPD (ADCMPRx[27:16]); if CMPCOND bit is set to 1, the internal match counter will increase one when the conversion result is greater than or equal to the value specified in CMPD (ADCMPRx[27:16]). When the conversion of the channel specified by CMPCH(ADCMPRx[7:3]) is completed, the comparing action will be triggered one time automatically. When the compare result meets the setting, compare match counter will increase 1, otherwise, the compare match counter will be cleared to 0. When the match counter reaches the setting of (CMPMCNT+1) then CMPF0/1 (ADSR0[1]/[2]) bit will be set to 1, if CMPIE (ADCMPRx[1]) bit is set then an ADC interrupt request is generated. Software can use it to monitor the external analog input pin voltage transition in scan mode without imposing a load on software. The detailed logic diagram is shown in Figure 6.43-7.

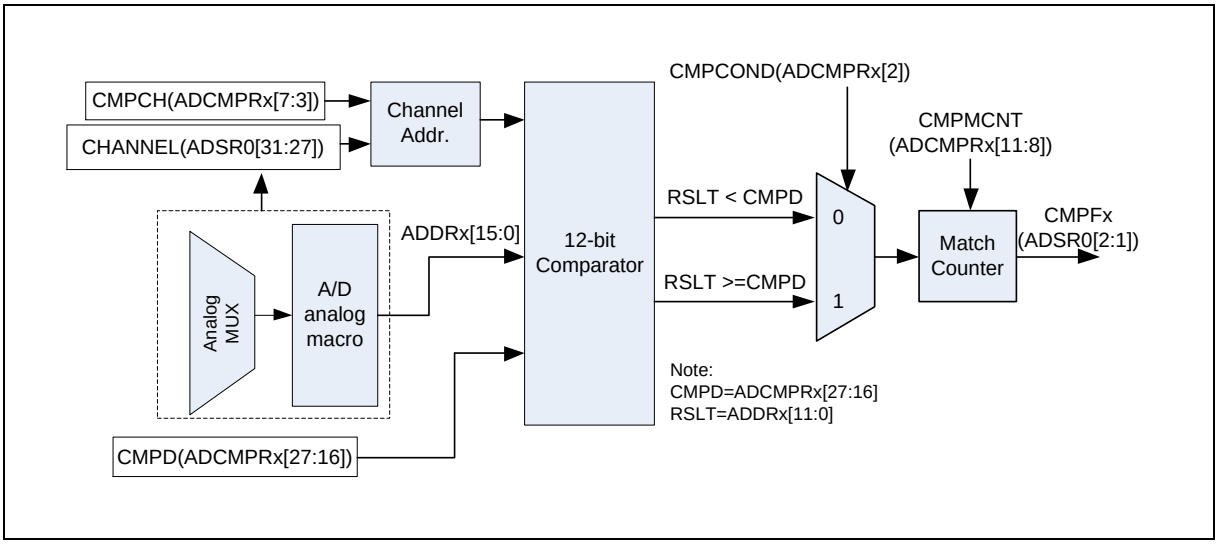


Figure 6.43-7 A/D Conversion Result Monitor Logic Diagram

6.43.5.14 Compare Window Mode

LPADC supports a compare window mode. User can set CMPWEN (LPADC_ADCMPR0[15]) to enable this function. If user enables this function, CMPF0 (ADSR0[1]) will be set only when compared conditions of two conversion result monitor logic are matched and CMPF1 (LPADC_ADSR0[2]) will always be zero. The range of compare window is between CMPD (LPADC_ADCMPR0[27:16]) and CMPD (LPADC_ADCMPR1[27:16]).

6.43.5.15 LPPDMA Transfer Mode

When an A/D conversion is finished, the conversion result will be loaded into ADDRx(x=0~26, 28~31) register and VALID(LPADC_ADDRx[17]) bit will be set to 1. If the PTEN(LPADC_ADCR[9]) bit is set, LPADC will generate a request to LPPDMA. User can use LPPDMA to transfer the conversion results to a user-specified memory space without CPU's intervention. The source address of LPPDMA operation is fixed at ADPDMA, no matter what channels was selected. When LPPDMA is transferring the conversion result, LPADC will continue converting the next selected channel if the operation mode of LPADC is burst mode, single scan mode or continuous scan mode. User can monitor current LPPDMA transfer data through reading ADPDMA register. If LPADC completes the conversion of a selected channel and the last conversion result of the same channel has not been transferred by LPPDMA, OVERRUN(LPADC_ADSR2[31:0]) bit of the corresponding channel will be set and the last LPADC conversion result will be overwritten by the new LPADC conversion result. LPPDMA will transfer the latest data of selected channels to the user-specified destination address.

6.43.5.16 Interrupt Sources

There are four interrupt sources of LPADC interrupt. When an LPADC operation mode finishes its conversion, the A/D conversion end flag, ADF(ADSR0[0]), will be set to 1. The CMPF0(ADSR0[1]) and CMPF1(ADSR0[2]) are the compare flags of compare function. When the conversion result meets the settings of LPADC_ADCMPR0/1 registers, the corresponding flag will be set to 1. The CALIF(LPADC_ADCALSTS[0]) is flag of calibration function. When calibration is finished, this flag will be set to 1. When one of the flags, ADF, CMPF0, CMPF1 and CALIF, is set to 1 and the corresponding interrupt enable bit, ADIE of LPADC_ADCR register, CMPIE of LPADC_ADCMPR0/1 registers and CALIE of LPADC_ADCAL register, is set to 1, the LPADC interrupt will be asserted. Software can clear these flags to revoke the interrupt request.

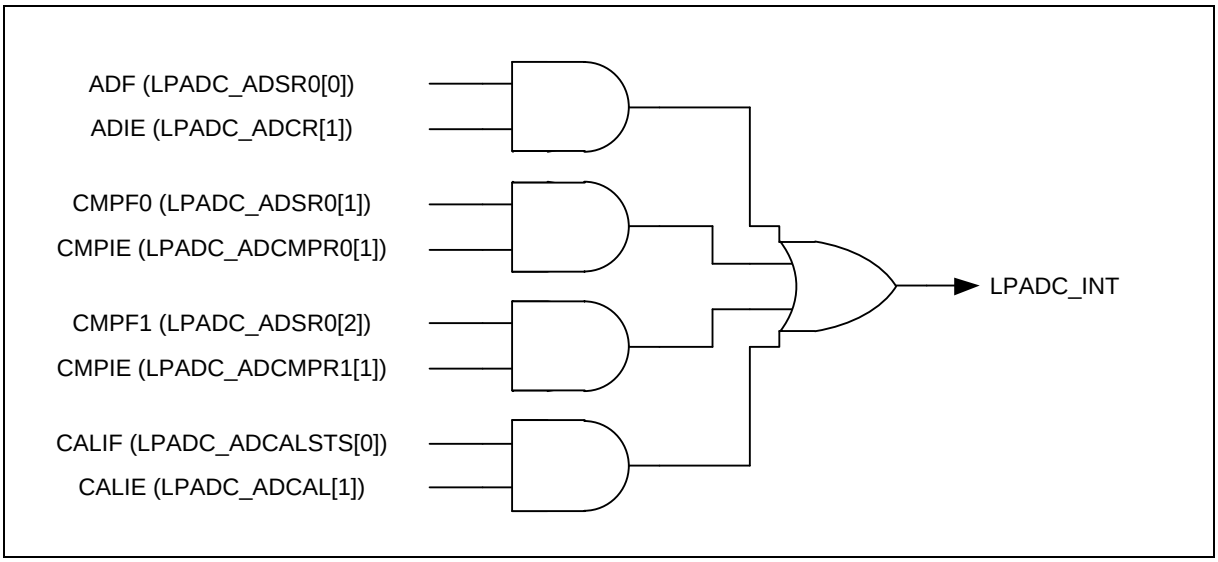


Figure 6.43-8 LPADC Interrupt Diagram

6.43.5.17 Floating Detect Function

Floating Detect Channel can be used to detect pin disconnection situations. SetFDETCHEN (LPADC_CFDCTL[8]) to 1 and select one of detected channels by LPADC_ADCHER. If the selected channel can be pull-up to V_{REF} or pull-down to AV_{SS}, this channel disconnection is detected. Assume V_{REF} is 3.0V, for example:

1. Enable one external channel and set input voltage is 1.2V.
2. Set FDETCHEN and PRECHEN (LPADC_CFDCTL[0]) to 1.
3. Wait for a time (by resistor value) and start conversion.
4. If the conversion result is close to V_{REF} result, this external channel disconnection is detected.

User can pull-up input voltage to V_{REF} when PRECHEN (LPADC_CFDCTL[0]) is 1 or pull-down to ground when DISCHEN (LPADC_CFDCTL[1]) is 1. When PRECHEN and DISCHEN are both set to 1, input voltage is 1/2 V_{REF}.

6.43.5.18 Differential Mode

LPADC supports analog fully-differential mode. A fully-differential SAR converts the differential voltage across its inputs; however, in this case both inputs change dynamically and are complementary or inverted from each other. In this case, the VINP and VINM pins must be driven 180° out of phase with respect to each other, centered on a fixed common mode voltage, for example V_{REF}/2. Fully-differential LPADCs have an input voltage common-mode (VCM) range specification. VCM is defined as the average voltage between the inputs : VCM = (VINP + VINM) / 2. Most fully-differential input SAR ADCs prohibit the input common-mode voltage from varying more than approximately 10 percent beyond the mid-scale input (V_{REF} /2).

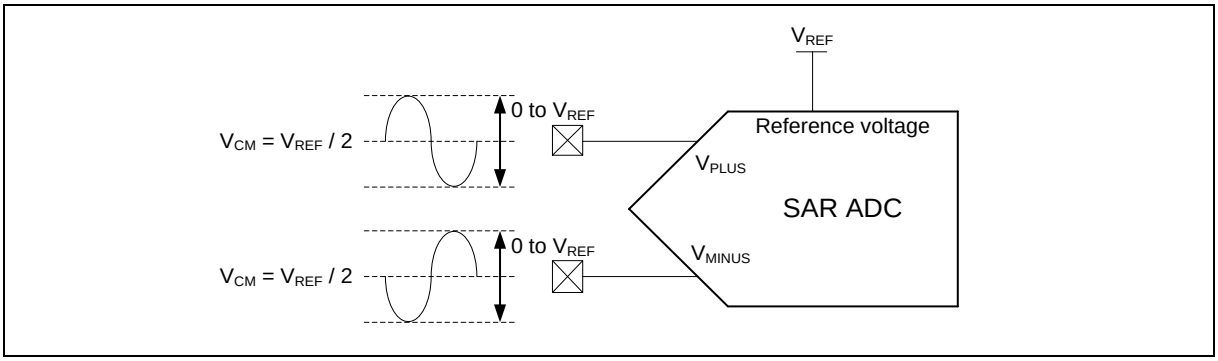


Figure 6.43-9 Input of Fully-Differential LPADC

If user enables DIFFEN (LPADC_ADCR[10]), the differential mode will enable. The pair of analog input channel is as Table 6.42-3.

Differential Analog Input Paired Channel	LPADC Analog Input	
	V_{plus}	V_{minus}
0	LPADC0_CH0	LPADC0_CH1
1	LPADC0_CH2	LPADC0_CH3
2	LPADC0_CH4	LPADC0_CH5
3	LPADC0_CH6	LPADC0_CH7
4	LPADC0_CH8	LPADC0_CH9
5	LPADC0_CH10	LPADC0_CH11
6	LPADC0_CH12	LPADC0_CH13
7	LPADC0_CH14	LPADC0_CH15
8	LPADC0_CH16	LPADC0_CH17
9	LPADC0_CH18	LPADC0_CH19
10	LPADC0_CH20	LPADC0_CH21
11	LPADC0_CH22	LPADC0_CH23

Table 6.43-1 LPADC Differential Model Channel Selection

In differential analog input mode, only the even number of the two corresponding channels needs to be enabled in CHEN (LPADC_ADCHER[31:0]). The conversion result will be placed to the corresponding data register of the enabled channel. The conversion result will store with 2'complement format when DMOF (LPADC_ADCR [31]) = 1.

6.43.5.19 Auto-operation Mode

LPADC supports auto-operation mode to make LPADC can convert when chip enters NPD0/1/2/3/4 mode. The block diagram of LPADC auto-operation mode is shown in Figure 6.28-19 . Setting the AUTOEN (LPADC_AUTOCTL[31]) to 1 can enable LPADC auto-operation mode. In auto-operation mode, the LPADC can be triggered to start a conversion when TRIGEN (LPADC_AUTOCTL[4]) is set to 1, even chip is under NPD0/1/2/3/4 mode. LPADC auto-operation mode trigger sources include LPTMR0/1, TTMR0/1, WKIOA0, WKIOB0, WKIOC0, WKIOD0 and ACMP0/1/2. The trigger source can be selected by TRIGSEL (LPADC_AUTOCTL[3:0]). For the trigger source of ACMP0/1/2, user can select both edge, rising edge or falling edge by setting ACMPTES (ADCR[21:20]) and the source is analog comparator output signal. User can also trigger LPADC start conversion in auto-operation mode

by setting SWTRIG (LPADC_AUTOSTRG[0]) to 1. When LPADC conversion is done, the ADWKF (LPADC_AUTOSTS[0]) will be set to 1 to record a conversion finish event under auto-operation mode. When ADWKF is arised to 1, chip will wake up from NPD0/1/2/3/4 mode if ADWKEN (LPADC_AUTOCTL[8]) is set to 1.

LPADC also supports conversion result monitor function working under auto-operation mode. User can set ADCMPRx(x=0,1) register to define result monitor condition. After trigger source to make LPADC start working in NPD0/1/2/3/4 mode, the conversion result monitor also working to see if LPADC conversion result meets user's compare criteria. If the condition is matched, the CMP0WKF (LPADC_AUTOSTS[1]) and CMP1WKF (LPADC_AUTOSTS[2]) will be set to 1 to record the event. And the chip will be woken up if correlated wake-up enable control bit CMP0WKEN (LPADC_AUTOCTL[9]) and CMP1WKEN (LPADC_AUTOCTL[10]) is set to 1.

When ADF bit in ADSR0 register is set to 1, after LPADC conversion ends under auto-operation mode, LPADC can send a LPPDMA request if PTEN (LPADC_ADCR[9]) is set to 1 to let LPPDMA start transferring LPADC conversion result to LPSRAM. By this setting, LPADC can keep working in Power-down mode and record data without CPU handling the progress.

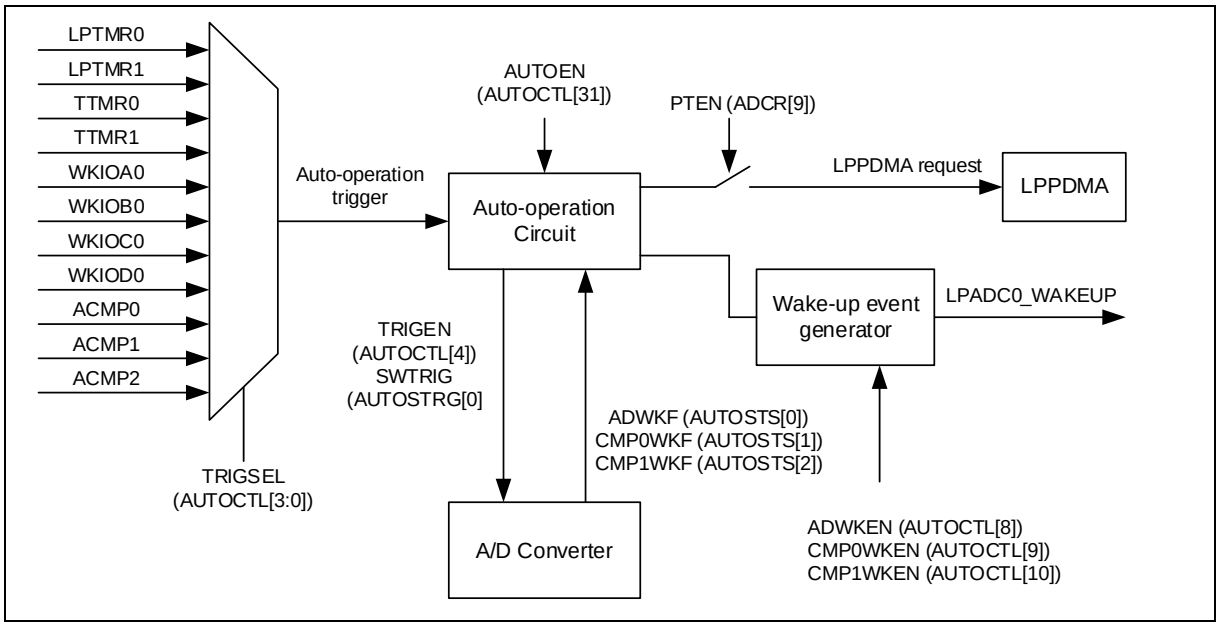


Figure 6.43-10 LPADC Auto-operation Mode Block Diagram

Note that, user should do calibration before enabling auto-operation mode because LPADC cannot do calibration when entering NPD0/1/2/3/4 mode. Also, the trigger source in auto-operation mode does not include LPADC0_ST, TIMER0~3, PWM0/1 and EPWM0/1 which are set by TRGS (LPADC_ADCR[19:16]). User should use TRIGSEL (LPADC_AUTOCTL[3:0]) to define trigger source in auto-operation mode.

User can also enable LPADC auto-operation mode and let LPADC start working when chip is in normal mode. When LPADC starts working in auto-operation mode, LPADC will keep working even chip enters NPD0/1/2/3/4 mode. When entering Power-down mode, the LPADC will enable HCLK1/PCLK2 and LPADC_CLK when LPADC is converting. After conversion is done, the HCLK1/PCLK2 and LPADC will stop and wait next conversion to start. The auto-operation clock monitor pins AOCM0 and AOCM1 can output HCLK1 or HIRC12M or MIRC1/2/4/8M clock enable or disable status to I/O for checking the clock behavior.

Data Collector

LPADC can be a sensor data collector in auto-operation mode: For a low power data collector through LPADC, chip can enter NPDx mode and set LPADC working in auto-operation mode. LPADC selects certain one or more analog input channel to convert periodically by setting auto-operation trigger source

from LPTMR or TTMR. Also enable PTEN (LPADC_ADCR[9]) to make conversion data transfer to LPSRAM through LPPDMA. When LPPDMA collect data count matches the transfer count the chip will be woken up from Power-down mode to do data processing and analysis.

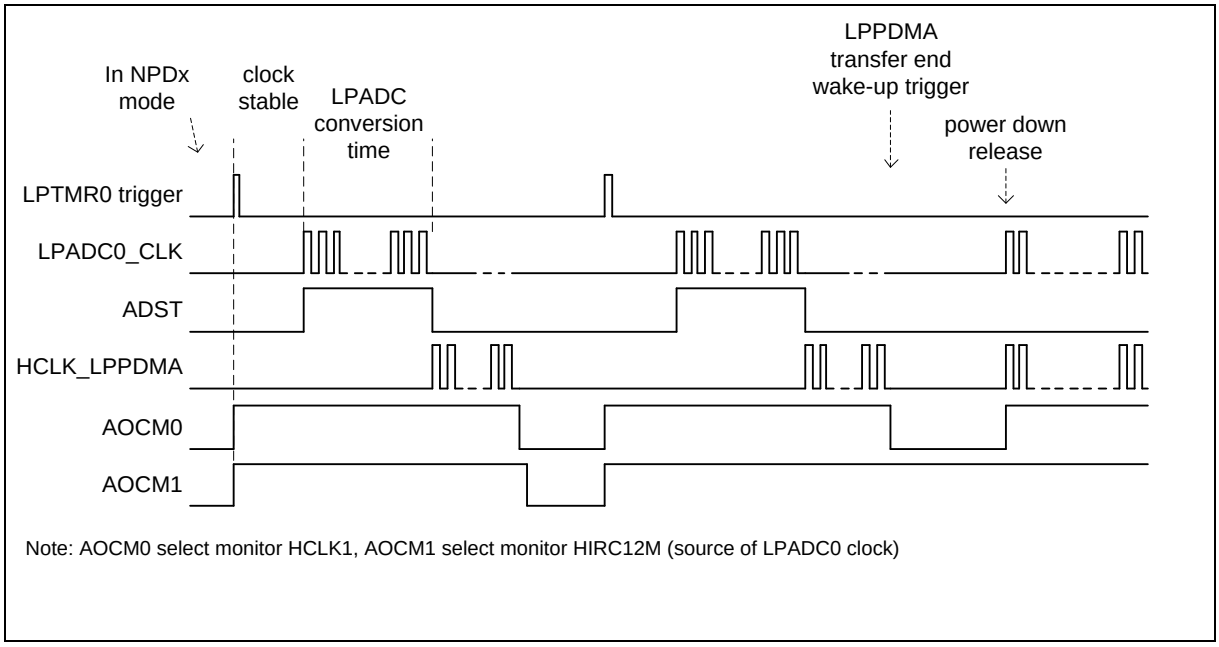


Figure 6.43-11 Clock Enable/Disable Control for Data Collector

Threshold Detector

LPADC can be a threshold detector: The conversion result monitor function can be enabled as a threshold detector when auto-operation mode is enabled. To realize a low power threshold detector, set LPADC working in auto-operation mode and enable conversion result monitor function when chip is under Power-down mode. User can select LPADC operated in continuous mode for fast scan conversion result or set LPTMR or TTIMR as periodic trigger source for slow scan conversion result. User can define target compared value and condition by setting ADCMPRx(x=0,1). If the conversion result matches LPADC conversion result monitor setting, LPADC can wake up chip and CPU can handle this event.

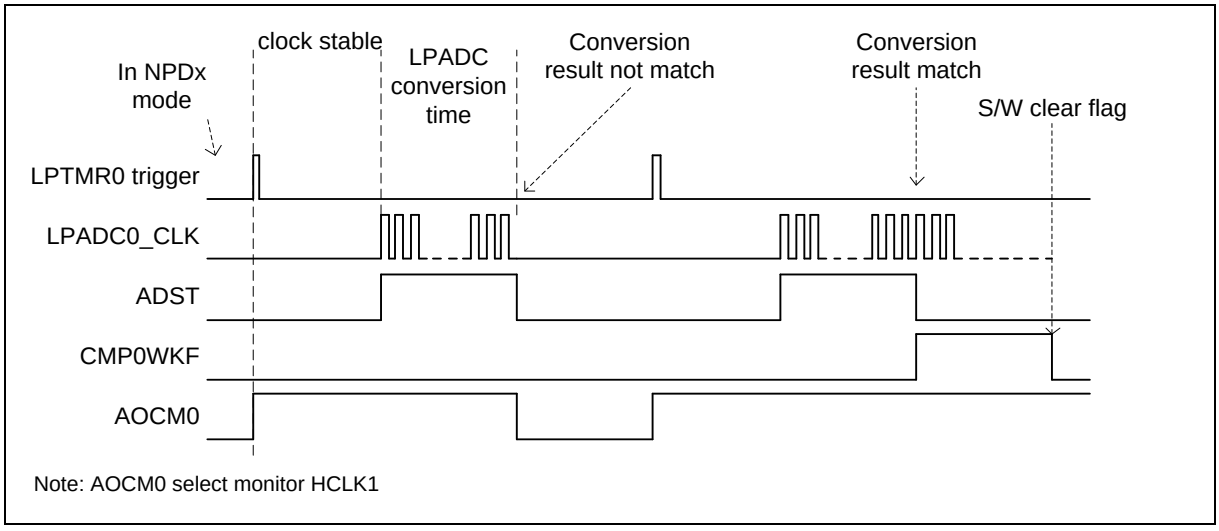


Figure 6.43-12 Clock Enable/Disable Control for Threshold Detector

6.43.5.20 Auto-operation Mode Wake-up Sources

When LPADC is working in auto-operation mode, chip can enter Power-down mode and wake up by LPADC. There are three wake-up sources of LPADC wake-up event. When an LPADC operation mode finishes its conversion in auto-operation mode, the A/D conversion end flag, ADWKF(AUTOSTS[0]), will be set to 1. The CMP0WKF(AUTOSTS[1]) and CMP1WKF(AUTOSTS[2]) are the compare match flags of conversion result monitor function. When the conversion result meets the settings of LPADC_ADCMPR0/1 registers in auto-operation mode, the corresponding flag will be set to 1. When one of the flags, ADWKF, CMP0WKF and CMP1WKF is set to 1 and the corresponding wake-up enable bit, ADWKEN, CMP0WKEN and CMP1WKEN of LPADC_AUTOCTL register is set to 1, the LPADC wake-up will be asserted. Software needs to clear these flags to revoke the chip wake-up request and enter the next chip power down successfully.

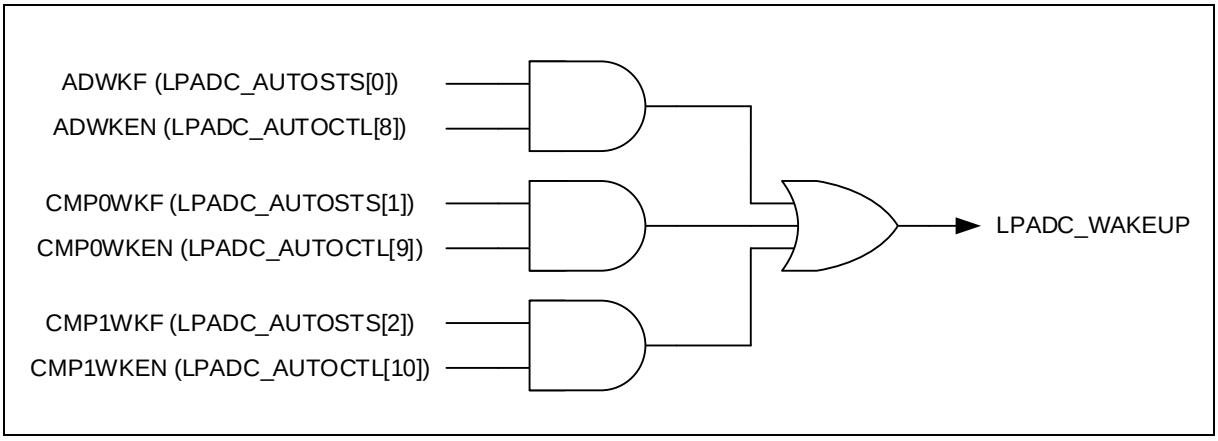


Figure 6.43-13 LPADC Wake-up Diagram

6.43.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write.

Register	Offset	R/W	Description	Reset Value
LPADC Base Address: **LPADC0_BA = 0x400E_3000**				
LPADC_ADDR0	LPADC0_BA+0x00	R	ADC Data Register 0	0x0000_0000
LPADC_ADDR1	LPADC0_BA+0x04	R	ADC Data Register 1	0x0000_0000
LPADC_ADDR2	LPADC0_BA+0x08	R	ADC Data Register 2	0x0000_0000
LPADC_ADDR3	LPADC0_BA+0x0C	R	ADC Data Register 3	0x0000_0000
LPADC_ADDR4	LPADC0_BA+0x10	R	ADC Data Register 4	0x0000_0000
LPADC_ADDR5	LPADC0_BA+0x14	R	ADC Data Register 5	0x0000_0000
LPADC_ADDR6	LPADC0_BA+0x18	R	ADC Data Register 6	0x0000_0000
LPADC_ADDR7	LPADC0_BA+0x1C	R	ADC Data Register 7	0x0000_0000
LPADC_ADDR8	LPADC0_BA+0x20	R	ADC Data Register 8	0x0000_0000
LPADC_ADDR9	LPADC0_BA+0x24	R	ADC Data Register 9	0x0000_0000
LPADC_ADDR10	LPADC0_BA+0x28	R	ADC Data Register 10	0x0000_0000
LPADC_ADDR11	LPADC0_BA+0x2C	R	ADC Data Register 11	0x0000_0000
LPADC_ADDR12	LPADC0_BA+0x30	R	ADC Data Register 12	0x0000_0000
LPADC_ADDR13	LPADC0_BA+0x34	R	ADC Data Register 13	0x0000_0000
LPADC_ADDR14	LPADC0_BA+0x38	R	ADC Data Register 14	0x0000_0000
LPADC_ADDR15	LPADC0_BA+0x3C	R	ADC Data Register 15	0x0000_0000
LPADC_ADDR16	LPADC0_BA+0x40	R	ADC Data Register 16	0x0000_0000
LPADC_ADDR17	LPADC0_BA+0x44	R	ADC Data Register 17	0x0000_0000
LPADC_ADDR18	LPADC0_BA+0x48	R	ADC Data Register 18	0x0000_0000
LPADC_ADDR19	LPADC0_BA+0x4C	R	ADC Data Register 19	0x0000_0000
LPADC_ADDR20	LPADC0_BA+0x50	R	ADC Data Register 20	0x0000_0000
LPADC_ADDR21	LPADC0_BA+0x54	R	ADC Data Register 21	0x0000_0000
LPADC_ADDR22	LPADC0_BA+0x58	R	ADC Data Register 22	0x0000_0000
LPADC_ADDR23	LPADC0_BA+0x5C	R	ADC Data Register 23	0x0000_0000
LPADC_ADDR24	LPADC0_BA+0x60	R	ADC Data Register 24	0x0000_0000
LPADC_ADDR25	LPADC0_BA+0x64	R	ADC Data Register 25	0x0000_0000
LPADC_ADDR26	LPADC0_BA+0x68	R	ADC Data Register 26	0x0000_0000

LPADC_ADDR28	LPADC0_BA+0x70	R	ADC Data Register 28	0x0000_0000
LPADC_ADDR29	LPADC0_BA+0x74	R	ADC Data Register 29	0x0000_0000
LPADC_ADDR30	LPADC0_BA+0x78	R	ADC Data Register 30	0x0000_0000
LPADC_ADDR31	LPADC0_BA+0x7C	R	ADC Data Register 31	0x0000_0000
LPADC_ADCR	LPADC0_BA+0x80	R/W	ADC Control Register	0x0000_0000
LPADC_ADCHER	LPADC0_BA+0x84	R/W	ADC Channel Enable Register	0x0000_0000
LPADC_ADCMPR0	LPADC0_BA+0x88	R/W	ADC Compare Register 0	0x0000_0000
LPADC_ADCMPR1	LPADC0_BA+0x8C	R/W	ADC Compare Register 1	0x0000_0000
LPADC_ADSR0	LPADC0_BA+0x90	R/W	ADC Status Register0	0x0000_0000
LPADC_ADSR1	LPADC0_BA+0x94	R	ADC Status Register1	0x0000_0000
LPADC_ADSR2	LPADC0_BA+0x98	R	ADC Status Register2	0x0000_0000
LPADC_ESMPCTL	LPADC0_BA+0xA0	R/W	ADC Extend Sample Time Control Register	0x0000_0000
LPADC_CFDCTL	LPADC0_BA+0xA4	R/W	ADC Channel Floating Detect Control Register	0x0000_0000
LPADC_ADPDMA	LPADC0_BA+0x100	R	ADC LPPDMA Current Transfer Data Register	0x0000_0000
LPADC_ADCAL	LPADC0_BA+0x180	R/W	ADC Calibration Mode Register	0x0000_005C
LPADC_ADCALSTS	LPADC0_BA+0x184	R/W	ADC Calibration Status Register	0x0000_0000
LPADC_AUTOCTL	LPADC0_BA+0x800	R/W	ADC Auto Operation Control Register	0x0000_0000
LPADC_AUTOSTRG	LPADC0_BA+0x804	W	ADC Auto Operation Software Trigger Register	0x0000_0000
LPADC_AUTOSTS	LPADC0_BA+0x810	R/W	ADC Auto Operation Status Register	0x0000_0000
LPADC_ALDOCTL	LPADC0_BA+0xFEC	R/W	ADC ALDO Control Register	0x0000_0000
LPADC_MTEST	LPADC0_BA+0xFF0	R/W	ADC Macro Test Mode Control Register	0x0000_0000

6.43.7 Register Description

ADC Data Registers (LPADC_ADDRx x = 0~23, 24, 25, 26, 28, 29, 30, 31)

Register	Offset	R/W	Description	Reset Value
LPADC_ADDR0	LPADC0_BA+0x00	R	ADC Data Register 0	0x0000_0000
LPADC_ADDR1	LPADC0_BA+0x04	R	ADC Data Register 1	0x0000_0000
LPADC_ADDR2	LPADC0_BA+0x08	R	ADC Data Register 2	0x0000_0000
LPADC_ADDR3	LPADC0_BA+0x0C	R	ADC Data Register 3	0x0000_0000
LPADC_ADDR4	LPADC0_BA+0x10	R	ADC Data Register 4	0x0000_0000
LPADC_ADDR5	LPADC0_BA+0x14	R	ADC Data Register 5	0x0000_0000
LPADC_ADDR6	LPADC0_BA+0x18	R	ADC Data Register 6	0x0000_0000
LPADC_ADDR7	LPADC0_BA+0x1C	R	ADC Data Register 7	0x0000_0000
LPADC_ADDR8	LPADC0_BA+0x20	R	ADC Data Register 8	0x0000_0000
LPADC_ADDR9	LPADC0_BA+0x24	R	ADC Data Register 9	0x0000_0000
LPADC_ADDR10	LPADC0_BA+0x28	R	ADC Data Register 10	0x0000_0000
LPADC_ADDR11	LPADC0_BA+0x2C	R	ADC Data Register 11	0x0000_0000
LPADC_ADDR12	LPADC0_BA+0x30	R	ADC Data Register 12	0x0000_0000
LPADC_ADDR13	LPADC0_BA+0x34	R	ADC Data Register 13	0x0000_0000
LPADC_ADDR14	LPADC0_BA+0x38	R	ADC Data Register 14	0x0000_0000
LPADC_ADDR15	LPADC0_BA+0x3C	R	ADC Data Register 15	0x0000_0000
LPADC_ADDR16	LPADC0_BA+0x40	R	ADC Data Register 16	0x0000_0000
LPADC_ADDR17	LPADC0_BA+0x44	R	ADC Data Register 17	0x0000_0000
LPADC_ADDR18	LPADC0_BA+0x48	R	ADC Data Register 18	0x0000_0000
LPADC_ADDR19	LPADC0_BA+0x4C	R	ADC Data Register 19	0x0000_0000
LPADC_ADDR20	LPADC0_BA+0x50	R	ADC Data Register 20	0x0000_0000
LPADC_ADDR21	LPADC0_BA+0x54	R	ADC Data Register 21	0x0000_0000
LPADC_ADDR22	LPADC0_BA+0x58	R	ADC Data Register 22	0x0000_0000
LPADC_ADDR23	LPADC0_BA+0x5C	R	ADC Data Register 23	0x0000_0000
LPADC_ADDR24	LPADC0_BA+0x60	R	ADC Data Register 24	0x0000_0000
LPADC_ADDR25	LPADC0_BA+0x64	R	ADC Data Register 25	0x0000_0000
LPADC_ADDR26	LPADC0_BA+0x68	R	ADC Data Register 26	0x0000_0000
LPADC_ADDR28	LPADC0_BA+0x70	R	ADC Data Register 28	0x0000_0000

LPADC_ADDR29	LPADC0_BA+0x74	R	ADC Data Register 29	0x0000_0000
LPADC_ADDR30	LPADC0_BA+0x78	R	ADC Data Register 30	0x0000_0000
LPADC_ADDR31	LPADC0_BA+0x7C	R	ADC Data Register 31	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**VALID**	**OVERRUN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RSLT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RSLT							

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**Valid Flag (Read Only)** This bit will be set to 1 when the conversion of the corresponding channel is completed. This bit will be cleared to 0 by hardware after ADDR register is read. 0 = Data in RSLT bits is not valid. 1 = Data in RSLT bits is valid.
[16]	**OVERRUN**	**Overrun Flag (Read Only)** If converted data in RSLT bits has not been read before new conversion result is loaded to this register, OVERRUN bit is set to 1. It is cleared by hardware after ADDR register is read. 0 = Data in RSLT bits is not overwritten. 1 = Data in RSLT bits is overwritten.
[15:0]	**RSLT**	**A/D Conversion Result (Read Only)** This field contains conversion result of ADC.

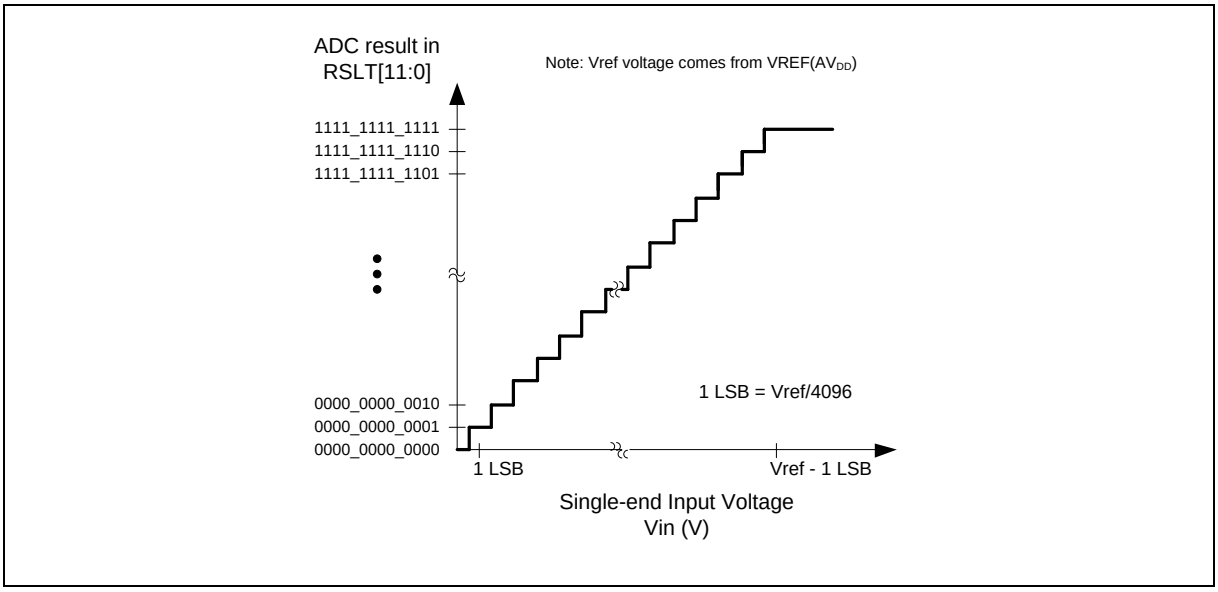


Figure 6.43-14 Conversion Result Mapping Diagram of ADC Single-end Input

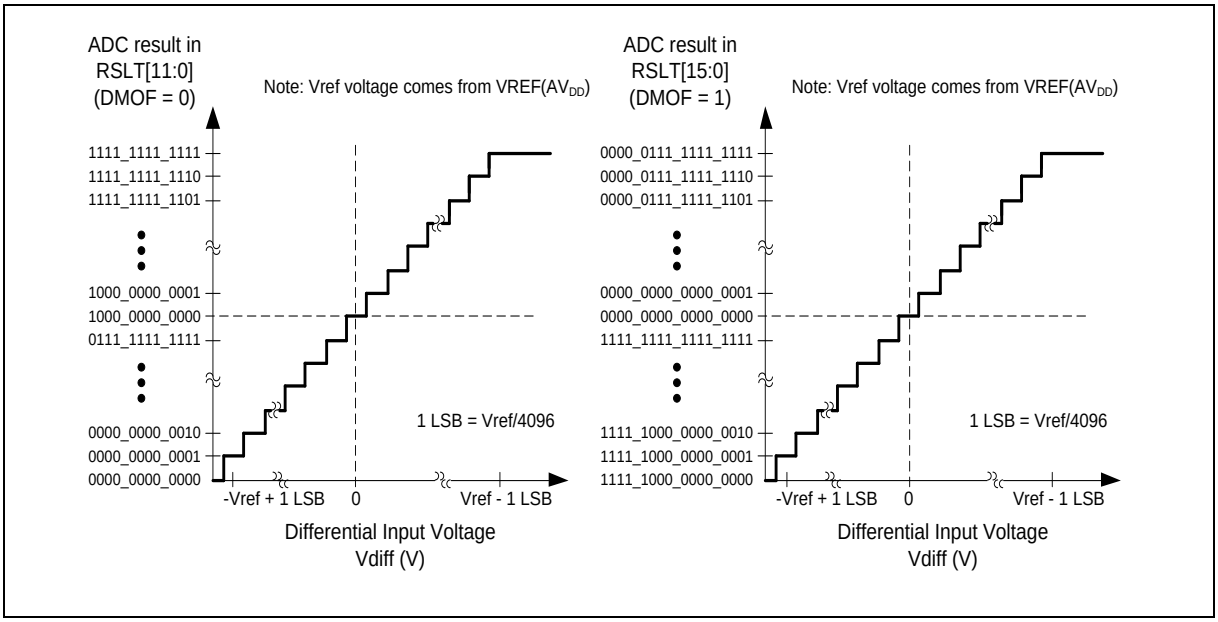


Figure 6.43-15 Conversion Result Mapping Diagram of ADC Differential Input

ADC Control Register (LPADC_ADCR)

Register	Offset	R/W	Description	Reset Value
LPADC_ADCR	LPADC0_BA+0x80	R/W	ADC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
DMOF	Reserved	RESSEL		Reserved			
23	22	21	20	19	18	17	16
Reserved		ACMPTES		TRGS			
15	14	13	12	11	10	9	8
Reserved			RESET	ADST	DIFFEN	PTEN	TRGEN
7	6	5	4	3	2	1	0
TRGCOND		Reserved		ADMD		ADIE	ADEN

Bits	Description	
[31]	**DMOF**	**Differential Input Mode Output Format** If user enables differential input mode, the conversion result can be expressed with binary straight format (unsigned format) or 2's complement format (signed format). 0 = A/D Conversion result will be filled in RSLT at ADDRx registers with unsigned format (straight binary format). 1 = A/D Conversion result will be filled in RSLT at ADDRx registers with 2's complement format.
[30]	**Reserved**	Reserved.
[29:28]	**RESSEL**	**Resolution Select Bits** 00 = ADC resolution 12 bits. 01 = ADC resolution 10 bits. 10 = ADC resolution 8 bits. 11 = Reserved.
[27:22]	**Reserved**	Reserved.
[21:20]	**ACMPTES**	**ACMP Trigger Event Selection** 00 = ACMP0/1/2 both edge event as trigger source. 01 = ACMP0/1/2 rising edge event as trigger source. 10 = ACMP0/1/2 falling edge event as trigger source. 11 = Reserved.
[19:16]	**TRGS**	**Hardware Trigger Source** 0000 = A/D conversion is started by external LPADC0_ST pin. 0001 = Timer0 ~ Timer3 overflow pulse trigger. 0011 = A/D conversion is started by PWM trigger. 0100 = A/D conversion is started by EPWM trigger. 1000 = ACMP0_O edge event. 1001 = ACMP1_O edge event. 1010 = ACMP2_O edge event. Others = Reserved.

		Note 1: Software should clear TRGEN bit and ADST bit to 0 before changing TRGS bits. **Note 2:** These trigger sources are only abaliable when chip is in run and idle mode.
[15:13]	**Reserved**	Reserved.
[12]	**RESET**	**ADC RESET (Write Protect)** If user writes this bit, the ADC analog macro will reset. Calibration data in macro will be deleted, but registers LPADC will keep. **Note:** This bit is cleared by hardware.
[11]	**ADST**	**A/D Conversion Start or Calibration Start** ADST bit can be set to 1 from five sources: software, external pin LPADC0_ST, PWM trigger, EPWM trigger, ACMPx_O edge event and Timer trigger. ADST bit will be cleared to 0 by hardware automatically at the ends of Single mode, Single-cycle Scan mode and Calibration mode. In Continuous Scan mode, A/D conversion is continuously performed until software writes 0 to this bit or chip is reset. 0 = Conversion stops and A/D converter enters idle state. 1 = Conversion starts or Calibration Start. **Note 1:** When ADST becomes from 1 to 0, ADC macro will reset to initial state. After macro reset to initial state, user should wait at most 2 ADC clock and set this bit to start next conversion. **Note 2:** Calibration Start only if CALEN (LPADC_ADCAL[0]) = 1.
[10]	**DIFFEN**	**Differential Input Mode Control** Differential input voltage (V_{diff}) = V_{plus} - V_{minus}. The relation between V_{plus} and V_{minus} is V_{plus} + V_{minus} = V_{ref}. The V_{plus} of differential input paired channel x is from LPADC0_CHy pin; V_{minus} is from LPADC0_CHz pin, x=0,1..11, y=2*x, z=y+1. 0 = Single-end analog input mode. 1 = Differential analog input mode. **Note:** In Differential Input mode, only the even number of the two corresponding channels needs to be enabled in ADCHER register. The conversion result will be placed to the corresponding data register of the enabled channel.
[9]	**PTEN**	**LPPDMA Transfer Enable Bit** When A/D conversion is completed, the converted data is loaded into ADDR0~26 and ADDR28~ADDR31. Software can enable this bit to generate a LPPDMA data transfer request. 0 = LPPDMA data transfer Disabled. 1 = LPPDMA data transfer in ADDR0~26 and ADDR28~ADDR31 Enabled. **Note:** When PTEN=1, software must set ADIE=0 to disable interrupt.
[8]	**TRGEN**	**External Trigger Enable Bit** Enable or disable triggering of A/D conversion by external LPADC0_ST pin, EPWM trigger, PWM trigger, ACMP interrupt event and Timer trigger. If external trigger is enabled, the ADST bit can be set to 1 by the selected hardware trigger source. 0 = External trigger Disabled. 1 = External trigger Enabled. **Note:** The ADC external trigger function is only supported in Single-cycle Scan mode.
[7:6]	**TRGCOND**	**External Trigger Condition** These two bits decide external pin LPADC0_ST trigger event is level or edge. The signal must be kept at stable state at least 8 PCLKs for level trigger and at least 4 PCLKs for edge trigger. 00 = Low level. 01 = High level. 10 = Falling edge.

		11 = Rising edge.
[5:4]	**Reserved**	Reserved.
[3:2]	**ADMD**	**A/D Converter Operation Mode Control** 00 = Single conversion. 10 = Single-cycle Scan. 11 = Continuous Scan. **Note 1:** When changing the operation mode, software should clear ADST bit first.
[1]	**ADIE**	**A/D Interrupt Enable Bit** A/D conversion end interrupt request is generated if ADIE bit is set to 1. 0 = A/D conversion end interrupt function Disabled. 1 = A/D conversion end interrupt function Enabled.
[0]	**ADEN**	**A/D Converter Enable Bit** 0 = A/D converter Disabled. 1 = A/D converter Enabled. **Note:** Before starting A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit to save power consumption.

ADC Channel Enable Register (LPADC_ADCHER)

Register	Offset	R/W	Description	Reset Value
LPADC_ADCHER	LPADC0_BA+0x84	R/W	ADC Channel Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
CHEN							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CHEN							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CHEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHEN							

Bits	Description	
[31:0]	**CHEN**	**Analog Input Channel Enable Control** Set ADCHER[23:0] bits to enable the corresponding analog external input channel 23 ~ 0. If DIFFEN bit is set to 1, only the even number channel needs to be enabled. Besides, setting the ADCHER[26:24] bits will enable internal OPA0/1/2 output voltage, setting the ADCHER[28] bit will enable internal channel for AV_{DD} divided by four, setting the ADCHER[29] bit will enable internal channel for band-gap voltage, setting the ADCHER[30] bit will enable internal channel for temperature sensor output voltage, setting the ADCHER[31] bit will enable internal channel for V_{BAT} divided by four. Other bits are reserved. 0 = Channel Disabled. 1 = Channel Enabled. **Note:** If the internal channel for AV_{DD} divide by four (CHEN[28]), band-gap voltage (CHEN[29]), temperature sensor output voltage (CHEN[30]) or V_{BAT} divided by four (CHEN[31]) is active, the maximum sampling rate will be 300k SPS.

ADC Compare Register 0/1 (LPADC_ADCMPR0/1)

Register	Offset	R/W	Description	Reset Value
LPADC_ADCMPR0	LPADC0_BA+0x88	R/W	ADC Compare Register 0	0x0000_0000
LPADC_ADCMPR1	LPADC0_BA+0x8C	R/W	ADC Compare Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				CMPD			
23	22	21	20	19	18	17	16
CMPD							
15	14	13	12	11	10	9	8
CMPWEN	Reserved			CMPMCNT			
7	6	5	4	3	2	1	0
CMPCH					CMPCOND	CMPIE	CMPEN

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**CMPD**	**Comparison Data** The 12-bit data is used to compare with conversion result of specified channel. **Note:** CMPD bits should be filled in unsigned format (straight binary format).
[15]	**CMPWEN**	**Compare Window Mode Enable Bit** 0 = Compare Window Mode Disabled. 1 = Compare Window Mode Enabled. **Note:** This bit is only presented in ADCMPR0 register.
[14:12]	**Reserved**	Reserved.
[11:8]	**CMPMCNT**	**Compare Match Count** When the specified A/D channel analog conversion result matches the compare condition defined by CMPCOND bit, the internal match counter will increase 1. When the internal counter reaches the value to (CMPMCNT +1), the CMPFx bit will be set.
[7:3]	**CMPCH**	**Compare Channel Selection** 00000 = Channel 0 conversion result is selected to be compared. 00001 = Channel 1 conversion result is selected to be compared. 00010 = Channel 2 conversion result is selected to be compared. 00011 = Channel 3 conversion result is selected to be compared. 00100 = Channel 4 conversion result is selected to be compared. 00101 = Channel 5 conversion result is selected to be compared. 00110 = Channel 6 conversion result is selected to be compared. 00111 = Channel 7 conversion result is selected to be compared. 01000 = Channel 8 conversion result is selected to be compared. 01001 = Channel 9 conversion result is selected to be compared. 01010 = Channel 10 conversion result is selected to be compared. 01011 = Channel 11 conversion result is selected to be compared.

		01100 = Channel 12 conversion result is selected to be compared. 01101 = Channel 13 conversion result is selected to be compared. 01110 = Channel 14 conversion result is selected to be compared. 01111 = Channel 15 conversion result is selected to be compared. 10000 = Channel 16 conversion result is selected to be compared. 10001 = Channel 17 conversion result is selected to be compared. 10010 = Channel 18 conversion result is selected to be compared. 10011 = Channel 19 conversion result is selected to be compared. 10100 = Channel 20 conversion result is selected to be compared. 10101 = Channel 21 conversion result is selected to be compared. 10110 = Channel 22 conversion result is selected to be compared. 10111 = Channel 23 conversion result is selected to be compared. 11000 = OPA0 internal output voltage conversion result is selected to be compared. 11001 = OPA1 internal output voltage conversion result is selected to be compared. 11010 = OPA2 internal output voltage conversion result is selected to be compared. 11100 = AV_{DD}/4 channel conversion result is selected to be compared. 11101 = Band-gap voltage conversion result is selected to be compared. 11110 = Temperature sensor voltage conversion result is selected to be compared. 11111 = V_{BAT}/4 channel conversion result is selected to be compared. Others = Reserved.
[2]	**CMPCOND**	**Compare Condition** 0 = Set the compare condition as that when a 12-bit A/D conversion result is less than the 12-bit CMPD bits, the internal match counter will increase one. 1 = Set the compare condition as that when a 12-bit A/D conversion result is greater than or equal to the 12-bit CMPD bits, the internal match counter will increase one. **Note:** When the internal counter reaches to (CMPMCNT +1), the CMPFx bit will be set.
[1]	**CMPIE**	**Compare Interrupt Enable Bit** If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMCNT, CMPFx bit will be asserted, in the meanwhile, if CMPIE bit is set to 1, a compare interrupt request is generated. 0 = Compare function interrupt Disabled. 1 = Compare function interrupt Enabled.
[0]	**CMPEN**	**Compare Enable Bit** Set this bit to 1 to enable LPADC to compare CMPD (ADCMPRx[27:16]) with specified channel conversion result when converted data is loaded into ADDR register. 0 = Compare function Disabled. 1 = Compare function Enabled.

ADC Status Register0 (LPADC_ADSR0)

Register	Offset	R/W	Description	Reset Value
LPADC_ADSR0	LPADC0_BA+0x90	R/W	ADC Status Register0	0x0000_0000

31	30	29	28	27	26	25	24
CHANNEL					Reserved		ADPRDY
23	22	21	20	19	18	17	16
Reserved							OVERRUNF
15	14	13	12	11	10	9	8
Reserved							VALIDF
7	6	5	4	3	2	1	0
BUSY	Reserved				CMPF1	CMPF0	ADF

Bits	Description	
[31:27]	**CHANNEL**	**Current Conversion Channel (Read Only)** When BUSY=1, this filed reflects current conversion channel. When BUSY=0, it shows the number of the next converted channel.
[26:25]	**Reserved**	Reserved.
[24]	**ADPRDY**	**ADC Power On Ready (Read Only)** After set ADEN (LPADC_ADCR[0]) to 1, the ADPRDY will set to 1 within one PCLK2 clock period. Then ADC macro can start convet. The chip also support ALDO when chip want to do A/D conversion under chip power level is PL2 and PL3. When ALDOEN (LPADC_ALDOCTL[8]) set to 1 to enable ALDO, the ALDO need a power stable time. The ADPRDY will keep low until ALDO power stable. The ALDO power ready time depends on STBSEL (LPADC_ALDOCTL[18:16]). **Note:** LPADC can start converting after ADPRDY is 1.
[23:17]	**Reserved**	Reserved.
[16]	**OVERRUNF**	**Overrun Flag (Read Only)** If any one of OVERRUN (ADDRx[16]) is set, this flag will be set to 1. **Note:** When ADC is in burst mode and the FIFO is overrun, this flag will be set to 1.
[15:9]	**Reserved**	Reserved.
[8]	**VALIDF**	**Data Valid Flag (Read Only)** If any one of VALID (ADDRx[17]) is set, this flag will be set to 1. **Note:** When ADC is in burst mode and any conversion result is valid, this flag will be set to 1.
[7]	**BUSY**	**BUSY/IDLE (Read Only)** This bit is a mirror of ADST bit in ADCR register. 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion.
[6:3]	**Reserved**	Reserved.
[2]	**CMPF1**	**Compare Flag 1**

		When the A/D conversion result of the selected channel meets setting condition in ADCMPR1 register, this bit is set to 1; it is cleared by writing 1 to it 0 = Conversion result in ADDR does not meet ADCMPR1 setting. 1 = Conversion result in ADDR meets ADCMPR1 setting.
[1]	**CMPF0**	**Compare Flag 0** When the A/D conversion result of the selected channel meets setting condition in ADCMPR0 register then this bit is set to 1. This bit is cleared by writing 1 to it. 0 = Conversion result in ADDR does not meet ADCMPR0 setting. 1 = Conversion result in ADDR meets ADCMPR0 setting.
[0]	**ADF**	**A/D Conversion End Flag** A status flag that indicates the end of A/D conversion. Software can write 1 to clear this bit. The ADF bit is set to 1 at the following three conditions: 1. When A/D conversion ends in Single mode. 2. When A/D conversion ends on all specified channels in Single-cycle Scan mode and Continuous Scan mode.

ADC Status Register1 (LPADC_ADSR1)

Register	Offset	R/W	Description	Reset Value
LPADC_ADSR1	LPADC0_BA+0x94	R	ADC Status Register1	0x0000_0000

31	30	29	28	27	26	25	24
VALID							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
VALID							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VALID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VALID							

Bits	Description	
[31:0]	**VALID**	**Data Valid Flag (Read Only)** VALID[31, 30, 29, 28, 26, 25, 24, 23:0] are the mirror of the VALID bits in ADDR31[17], ADDR30[17], ADDR29[17], ADDR28[17], ADDR26[17], ADDR25[17], ADDR24[17], ADDR23[17]~ ADDR0[17]. The other bits are reserved. **Note:** When ADC is in burst mode and any conversion result is valid, VALID[31, 30, 29, 28, 26, 25, 24, 23:0] will be set to 1.

ADC Status Register2 (LPADC_ADSR2)

Register	Offset	R/W	Description	Reset Value
LPADC_ADSR2	LPADC0_BA+0x98	R	ADC Status Register2	0x0000_0000

31	30	29	28	27	26	25	24
OVERRUN							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
OVERRUN							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
OVERRUN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OVERRUN							

Bits	Description	
[31:0]	**OVERRUN**	**Overrun Flag (Read Only)** OVERRUN[31, 30, 29, 28, 26, 25, 24, 23:0] are the mirror of the OVERRUN bit in ADDR31[16], ADDR30[16], ADDR29[16], ADDR28[16], ADDR26[16], ADDR25[16], ADDR24[16], ADDR23[16] ~ ADDR0[16]. The other bits are reserved. **Note:** When ADC is in burst mode and the FIFO is overrun, OVERRUN[31, 30, 29, 28, 26, 25, 24, 23:0] will be set to 1.

ADC Extend Sample Time Control Register (LPADC_ESMPCTL)

Register	Offset	R/W	Description	Reset Value
LPADC_ESMPCTL	LPADC0_BA+0xA0	R/W	ADC Extend Sample Time Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**23**	**22**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**EXTSMPT**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EXTSMPT							

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:0]	**EXTSMPT**	**ADC Sampling Time Extend** When A/D conversion at high conversion rate, the sampling time of analog input voltage may not enough if input channel loading is heavy, user can extend ADC sampling time after trigger source is coming to get enough sampling time. The range of start delay time is from 0~16383 ADC clock.

ADC Channel Floating Detect Control Register (LPADC_CFDCTL)

Register	Offset	R/W	Description	Reset Value
LPADC_CFDCTL	LPADC0_BA+0xA4	R/W	ADC Channel Floating Detect Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**23**	**22**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**15**	**14**	**11**	**10**	**9**	**8**
Reserved							**FDETCHEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**DISCHEN**	**PRECHEN**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**FDETCHEN**	**Floating Detect Channel Enable Bit** 0 = Floating Detect Channel Disabled. 1 = Floating Detect Channel Enabled.
[7:2]	**Reserved**	Reserved.
[1]	**DISCHEN**	**Discharge Enable Bit** 0 = Channel discharge Disabled. 1 = Channel discharge Enabled. **Note:** Analog input voltage is 1/2 V_{REF} when PRECHEN and DISCHEN are all enabled.
[0]	**PRECHEN**	**Precharge Enable Bit** 0 = Channel precharge Disabled. 1 = Channel precharge Enabled. **Note:** Analog input voltage is 1/2 V_{REF} when PRECHEN and DISCHEN are all enabled.

ADC LPPDMA Current Transfer Data Register (LPADC_ADPDMA)

Register	Offset	R/W	Description	Reset Value
LPADC_ADPDMA	LPADC0_BA+0x100	R	ADC LPPDMA Current Transfer Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**23**	**22**	**19**	**18**	**17**	**16**
Reserved						**CURDAT**	
15	**14**	**15**	**14**	**11**	**10**	**9**	**8**
CURDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CURDAT							

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:0]	**CURDAT**	**ADC LPPDMA Current Transfer Data Register (Read Only)** When LPPDMA transfers data, reading the register can monitor the current LPPDMA transfer data. Current LPPDMA transfer data could be the content of ADDR0 ~ ADDR23, ADDR24, ADDR25, ADDR26, ADDR28, ADDR29, ADDC30 and ADDR31 registers.

ADC Calibration Mode Register (LPADC_ADCAL)

Register	Offset	R/W	Description	Reset Value
LPADC_ADCAL	LPADC0_BA+0x180	R/W	ADC Calibration Mode Register	0x0000_005C

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**CALIE**	**CALEN**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CALIE**	**Calibration Interrupt Enable Bit** If calibration function is enabled and the calibration is finished, CALIF bit will be asserted, in the meanwhile, if CALIE bit is set to 1, a calibration interrupt request generated. 0 = Calibration function Interrupt Disabled. 1 = Calibration function Interrupt Enabled.
[0]	**CALEN**	**Calibration Function Enable Bit** 0 = Calibration function Disabled. 1.= Calibration function Enabled. **Note:** If chip is powered off, calibration function should be executed again.

ADC Calibration Status Register (LPADC_ADCALSTS)

Register	Offset	R/W	Description	Reset Value
LPADC_ADCALSTS	LPADC0_BA+0x184	R/W	ADC Calibration Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CALIF

Bits	Description	
[31:1]	Reserved	Reserved.
[0]	CALIF	**Calibration Finish Interrupt Flag** If calibration is finished, this flag will be set to 1. It is cleared by writing 1 to it.

ADC Auto Operation Control Register (LPADC_AUTOCTL)

Register	Offset	R/W	Description	Reset Value
LPADC_AUTOCTL	LPADC0_BA+0x800	R/W	ADC Auto Operation Control Register	0x0000_0000

31	30	29	28	27	26	25	24
AUTOEN	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					CMP1WKEN	CMP0WKEN	ADWKEN
7	6	5	4	3	2	1	0
Reserved			TRIGEN	TRIGSEL			

Bits	Description	
[31]	**AUTOEN**	**Automatic Operation Mode Enable Bit** 0 = LPADC automatic operation Disabled. 1 = LPADC automatic operation Enabled.
[30:11]	**Reserved**	Reserved.
[10]	**CMP1WKEN**	**Automatic Operation Mode Comparator 1 Wake-up Enable Bit** 0 = LPADC automatic operation comparator 1 wake-up Disabled. 1 = LPADC automatic operation comparator 1 wake-up Enabled.
[9]	**CMP0WKEN**	**Automatic Operation Mode Comparator 0 Wake-up Enable Bit** 0 = LPADC automatic operation comparator 0 wake-up Disabled. 1 = LPADC automatic operation comparator 0 wake-up Enabled.
[8]	**ADWKEN**	**Automatic Operation Mode Conversion End Wake-up Enable Bit** 0 = LPADC automatic operation conversion end wake-up Disabled. 1 = LPADC automatic operation conversion end wake-up Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**TRIGEN**	**Automatic Operation Trigger Enable Bit** When automatic operation mode is enabled, the LPADC will start working if LPADC is triggered by a event sent from the trigger source selected by TRIGSEL[3:0] after this bit is set to 1. 0 = LPADC Automatic Operation Trigger Disabled. 1 = LPADC Automatic Operation Trigger Enabled.
[3:0]	**TRIGSEL**	**Automatic Operation Trigger Source Select** 0000 = Auto-operation Trigger Source from LPTMR0. 0001 = Auto-operation Trigger Source from LPTMR1. 0010 = Auto-operation Trigger Source from TTMR0. 0011 = Auto-operation Trigger Source from TTMR1. 0100 = Auto-operation Trigger Source from WKIOA0. 0101 = Auto-operation Trigger Source from WKIOB0.

		0110 = Auto-operation Trigger Source from WKIOC0. 0111 = Auto-operation Trigger Source from WKIOD0. 1000 = Auto-operation Trigger Source from ACMP0 analog output event. 1001 = Auto-operation Trigger Source from ACMP1 analog output event. 1010 = Auto-operation Trigger Source from ACMP2 analog output event. Others = Reserved. **Note:** TRIGSEL cannot be changed when TRIGEN is 1.

ADC Auto Operation Software Triger Register (LPADC_AUTOSTRG)

Register	Offset	R/W	Description	Reset Value
LPADC_AUTOSTRG	LPADC0_BA+0x804	W	ADC Auto Operation Software Trigger Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SWTRIG**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SWTRIG**	**Automatic Operation Software Trigger Bit** Write 1 to this bit will trigger LPADC start automatic operation. **Note:** This bit will be auto cleared by hardware.

ADC Auto Operation Status Register (LPADC_AUTOSTS)

Register	Offset	R/W	Description	Reset Value
LPADC_AUTOSTS	LPADC0_BA+0x810	R/W	ADC Auto Operation Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**CMP1WKF**	**CMP0WKF**	**ADWKF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CMP1WKF**	**Automatic Operation Compare 1 Wake-up Flag Bit** When automatic operation mode is enabled, and the conversion result monitor comparator 0 is match setting criteria, this flag will be set. If CMP0WKEN is also set to 1, the chip will be woken up. **Note:** Software can write 1 to clear this bit.
[1]	**CMP0WKF**	**Automatic Operation Compare 0 Wake-up Flag Bit** When automatic operation mode is enabled, and the conversion result monitor comparator 1 matches setting criteria, this flag will be set. If CMP1WKEN is also set to 1, the chip will be woken up. **Note:** Software can write1 to clear this bit.
[0]	**ADWKF**	**Automatic Operation Conversion End Wake-up Flag Bit** When automatic operation mode is enabled, and the conversion is finished, this flag will be set. If WKEN is also set to 1, the chip will be woken up. **Note:** Software can write 1 to clear this bit.

ADC ALDO Control Register (LPADC_ALDOCTL)

Register	Offset	R/W	Description	Reset Value
LPADC_ALDOCTL	LPADC0_BA+0xFEC	R/W	ADC ALDO Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
IGENEN	Reserved				STBSEL		
15	14	13	12	11	10	9	8
Reserved							ALDOEN
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**IGENEN**	**ALDO Enable under IGEN Enable Bit** 0 = ALDO enable under IGEN Disabled. 1 = ALDO enable under IGEN Enabled. **Note:** When ALDO enable under IGEN enable, the ALDO stable time is shorter.
[22:19]	**Reserved**	Reserved.
[18:16]	**STBSEL**	**ALDO Stable Count Select Bits** If IGENEN (ALDOCTL[23]) is 0, the ALDO stable count is as follows. 010 = ALDO stable count is 408 PCLK2 (When PCLK is 12 MHz). 011 = ALDO stable count is 272 PCLK2 (When PCLK is 8 MHz). 100 = ALDO stable count is 136 PCLK2 (When PCLK is 4 MHz). Others = Reserved. If IGENEN (ALDOCTL[23]) is 1, the ALDO stable count is as follows. 010 = ALDO stable count is 48 PCLK2 (When PCLK is 12 MHz). 011 = ALDO stable count is 32 PCLK2 (When PCLK is 8 MHz). 100 = ALDO stable count is 16 PCLK2 (When PCLK is 4 MHz). Others = Reserved.
[15:9]	**Reserved**	Reserved.
[8]	**ALDOEN**	**ALDO Enable Bit** 0 = ALDO Disabled. 1 = ALDO Enabled. **Note 1:** After ADEN (LPADC_ADCR[0]) is enabled, the ALDOEN cannot be changed. **Note 2:** ALDO can be enabled only when the chip runs at power level 3 (PL3) or NPD2/4 or NPD0/1/3 with power level 3. When the chip runs at power level 1 (PL1) or power level 2 (PL2), ALDO should be disabled.
[7:0]	**Reserved**	Reserved.

ADC Macro Test Mode Control Register (LPADC_MTEST) for Test, Not Release to TRM

Register	Offset	R/W	Description	Reset Value
LPADC_MTEST	LPADC0_BA+0xFF0	R/W	ADC Macro Test Mode Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**DECADD**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**DECADD**	**EADC Decode Add one cycle** 0 = Decode keep original cycle. 1 = Decode ADD one cycle. **Note:** To ensure good ADC accuracy, this bit should be set to 1.
[7:0]	**Reserved**	Reserved.

6.44 Digital to Analog Converter (DAC)

6.44.1 Overview

The DAC module is a 12-bit, voltage output digital-to-analog converter. It can be configured to 12- or 8-bit output mode and can be used in conjunction with the PDMA controller. The DAC integrates a voltage output buffer that can be used to reduce output impendence and drive external loads directly without having to add an external operational amplifier.

6.44.2 Features

- Analog output voltage range: 0~AV_{DD}.
- Supports 12- or 8-bit output mode.
- Rail to rail settle time 8us.
- Supports up to two 12-bit 1 MSPS voltage type DAC.
- Reference voltage from internal reference voltage (INT_VREF), V_{REF} pin.
- DAC maximum conversion updating rate 1 MSPS.
- Supports voltage output buffer mode and bypass voltage output buffer mode.
- Supports software and hardware trigger, including Timer0~3, EPWM0, EPWM1, PWM0, PWM1 and external trigger pin to start DAC conversion.
- Supports PDMA mode.
- Supports group mode of synchronized update capability for two DACs.

6.44.3 Block Diagram

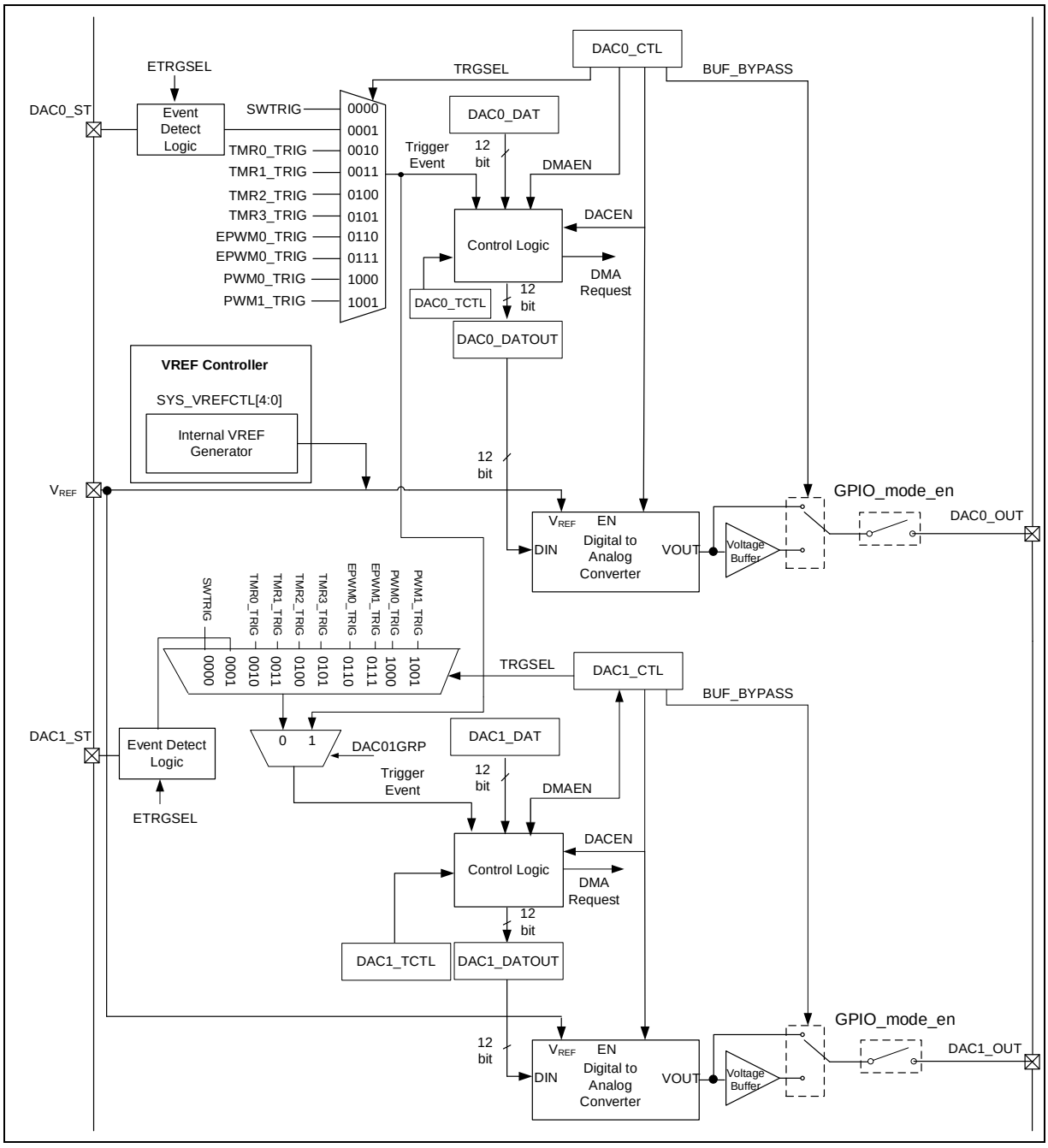


Figure 6.44-1 Digital-to-Analog Converter Block Diagram

6.44.4 Basic Configuration

6.44.4.1 DAC0 Basic Configuration

- Clock source Configuration
  - Enable DAC0 peripheral clock in DACCKEN (CLK_APBCLK1[12]).
- Reset Configuration
  - Reset DAC0 controller in DACRST (SYS_IPRST2[12]).

6.44.4.2 DAC1 Basic Configuration

- Clock source Configuration
  - Enable DAC1 peripheral clock in DACCKEN (CLK_APBCLK1[12]).
- Reset Configuration
  - Reset DAC1 controller in DACRST (SYS_IPRST2[12]).

6.44.5 Functional Description

6.44.5.1 DAC Output

The DAC is a 12-bit voltage output digital-to-analog converter and can be configured as 12- or 8-bit operation mode. The DAC integrates a voltage output buffer that can be used to reduce output impendence and drive external loads directly without having to add an external operational amplifier. The DAC channel output buffer can be enabled and disabled by BYPASS (DACn_CTL[18]), n=0, 1. The maximum DAC output voltage is limited to the selected reference voltage source.

6.44.5.2 DAC Reference Voltage

The DAC reference voltage is shared with EADC reference voltage and it is configured by VREFCTL (SYS_VREFCTL[4:0]) in system manager control registers. The reference voltage for the DAC can be configured from external reference voltage pin (V_{REF}) or internal reference voltage generator (INT_VREF).

6.44.5.3 DAC Data Format

The DAC supports conversion data left alignment or right alignment mode when DAC is working in 12-bit mode. Depending on the selected configuration mode, the data needs to be written into the specified register as follows:

- 12-bit left alignment: user has to load data into DACn_DAT[15:4] bits. DACn_DAT[31:16] and DACn_DAT[3:0] are ignored in DAC conversion.
- 12-bit right alignment: user has to load data into DACn_DAT[11:0] bits, DACn_DAT[31:12] are ignored in DAC conversion.

While DAC is working in 8-bit mode, alignment setting has no effect. To enable 8-bit mode, set BWSEL(DACn_CTL[15:14]) to 01. Otherwise, keep BWSEL as 00.

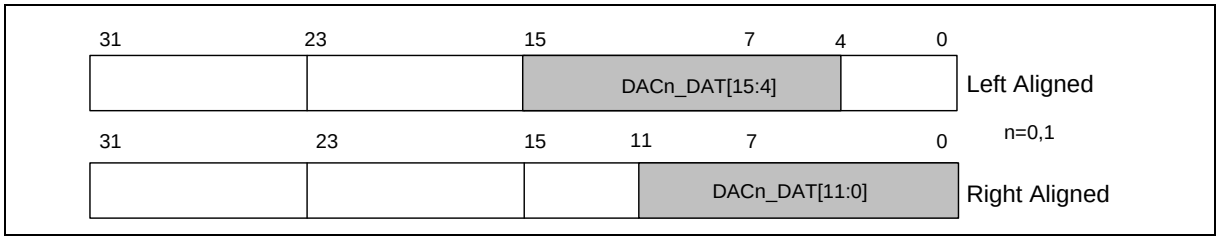


Figure 6.44-2 Data Holding Register Format

6.44.5.4 DAC Conversion

Any data transfer to the DAC channel is performed by loading the data into DACn_DAT register. Figure 6.44-3 shows the DAC conversion started by software write operation. When user writes the conversion data to data holding register DACn_DAT, the data is loaded into data output register DACn_DATOUT by hardware and DAC starts data conversion after one PCLK (APB clock) clock cycle. Figure 6.44-4 shows the DAC conversion started by hardware trigger (external pin DACn_ST, timer trigger event, PWM timer trigger event or EPWM timer trigger event). The data stored in the DACn_DAT register is automatically transferred to the data output buffer DACn_DATOUT after occurring one PCLK (APB clock) the event.

When DAC data output register DACn_DATOUT is loaded with the DACn_DAT contents, the analog output voltage becomes available after specified conversion settling time. The conversion settling time is 8us when 12- bit input code transition from lowest code (0x000) to highest code (0xFFF). Two adjacent codes conversion settling time is 1us. The DAC controller provides a 10-bit time counter for user to count the conversion time period. In continuous conversion operation, user needs to write appropriate value to SETTLET (DACn_TCTL[9:0]) to define DAC conversion time period. The value must be longer than DAC conversion settling time which is specified in DAC electric characteristic table. For example, when DAC controller APB clock speed is 72 MHz and DAC conversion settling time is 1us, the selected SETTLET value must be greater than 0x48. When the conversion is started, the conversion finish flag FINISH (DACn_STATUS[0]) is cleared to 0 by hardware and set to 1 after the time counter counts to SETTLET. Note that n=0,1.

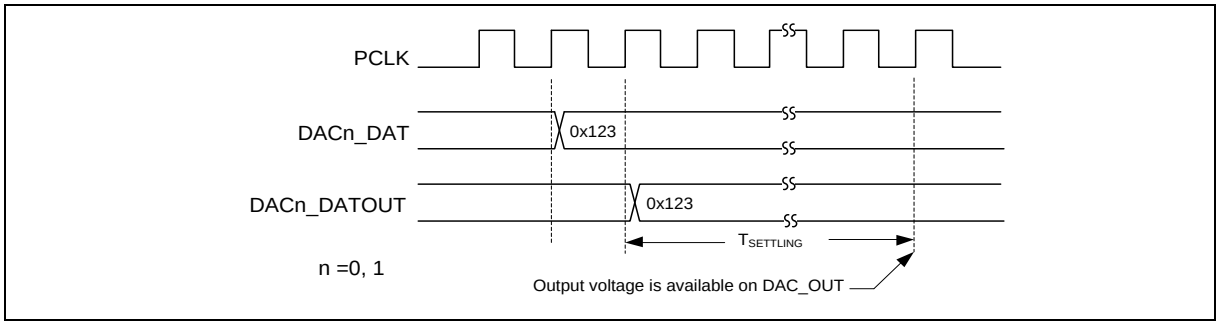


Figure 6.44-3 DAC Conversion Started by Software Write Trigger

6.44.5.5 DAC Output Voltage

Digital inputs are converted to output voltage on a linear conversion between 0 and reference voltage V_{REF}. The analog output voltage on DAC pin is determined by the following equation:

$$DACOUT = V_{REF} * \frac{DATnOUT[11:0]}{4096}, \quad n=0,1$$

6.44.5.6 DAC Trigger Selection

The DAC conversion can be started by writing DACn_DAT, software trigger or hardware trigger. When TRGEN (DACn_CTL[4]) is 0, the data conversion is started by writing DACn_DAT register. When TRGEN (DACn_CTL[4]) is 1, the data conversion is started by external DACn_ST pin, Timer event, PWM Timer event or EPWM Timer event. If the software trigger is selected, the conversion starts once the SWTRG (DACn_SWTRG[0]) is set to 1. The SWTRG is cleared to 0 by hardware automatically when DACn_DATOUT has been loaded with DACDAT content. The TRGSEL (DACn_CTL[8:5]) determines which one of eight events is selected to start the conversion.

When DAC detects a rising edge on the selected trigger event input, the last data stored in DACDAT is transferred into the DACn_DATOUT[11:0] and DAC starts converting after one PCLK (APB clock) clock cycle. Note that n=0,1.

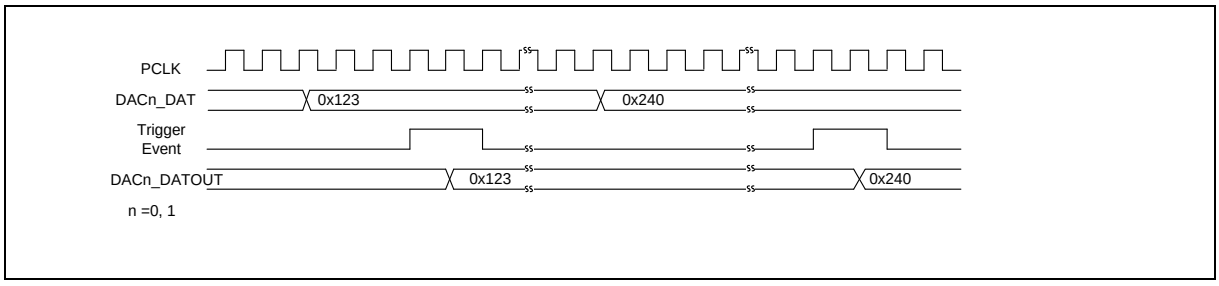


Figure 6.44-4 DAC Conversion Started by Hardware Trigger Event

6.44.5.7 DAC Group Mode

The DAC0 and DAC1 can be grouped together by setting GRPEN (DAC0_CTL[16]) to synchronize the update of each DAC output. Hardware ensures that these two DACs will be updated simultaneously in group mode. In group mode, DAC1_CTL and DAC1_TCTL has no effect. DAC1's behavior is controlled by DAC_GRPDAT, DAC0_CTL and DAC0_TCTL. In group mode, User should write DAC0, DAC1 conversion data into DAC_GRPDAT rather than DAC0_DAT and DAC1_DAT. It gives advantage that only one DMA channel in DMA operation instead of two DMA channel setting is required. Figure 6.44-5 shows an example of group mode with writing conversion data into DAC_GRPDAT and compared with normal mode.

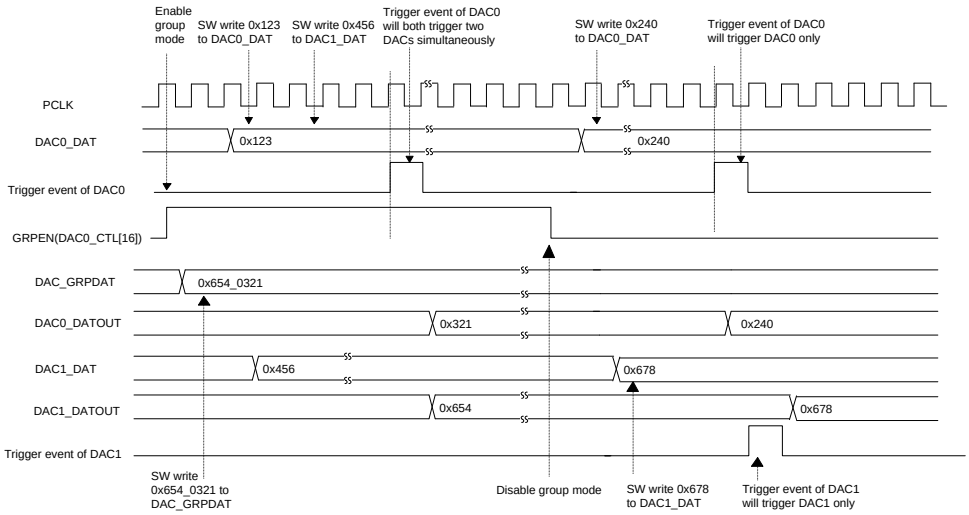


Figure 6.44-5 DAC0 and DAC1 Group and Ungroup Update Example

6.44.5.8 DMA Operation

A DAC DMA request is generated when a hardware trigger event occurs while DMAEN (DACn_CTL[2]) is set. The content of DACn_DAT is transferred to the DACn_DATOUT[11:0] and DAC starts data conversion after one PCLK (APB clock) clock cycle. The new transferred data by PDMA in DACn_DAT will be converted when next trigger event arrives. Figure 6.44-6 shows the DAC PDMA under-run condition, when the second DMA request trigger event arrives before the first conversion finish, then no new PDMA request is issued and DMA under-run flag DMAUDR (DACn_STATUS[1]) is set 1 to report the error condition. DMA data transfers are then disabled and no further DMA request is treated and DAC continues to convert last data. An interrupt is also generated if the corresponding DMAURIEN (DACn_CTL[3]) is enabled. User has to change the trigger event frequency in Timer, PWM timer or EPWM timer and then start DAC conversion again. Note that n=0,1.

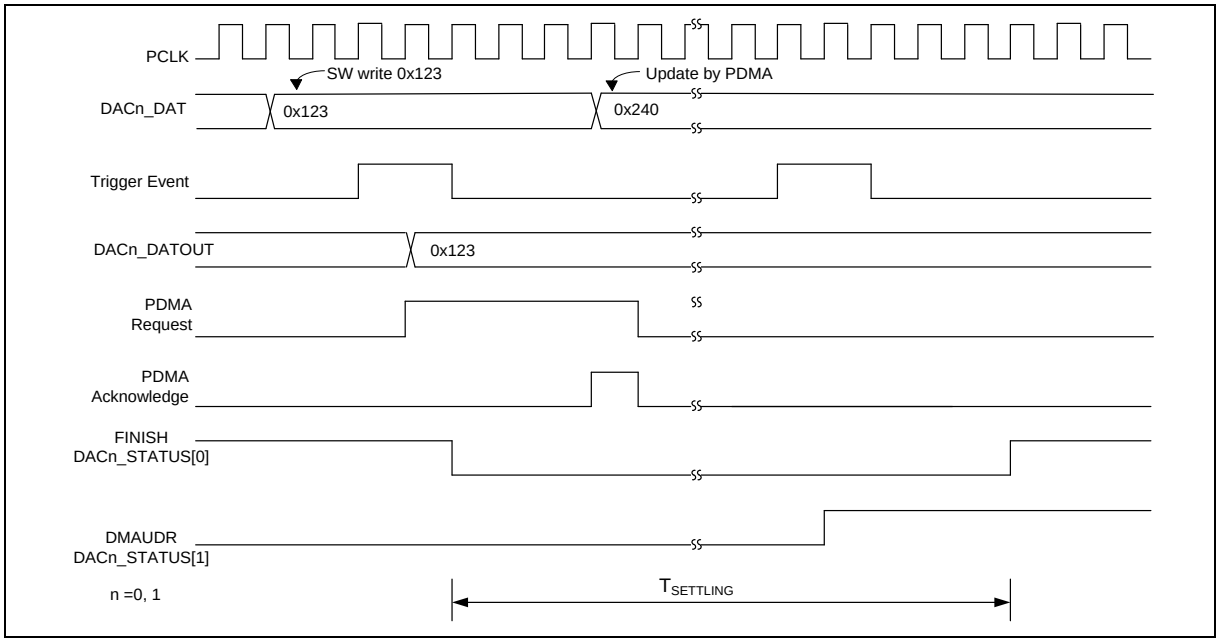


Figure 6.44-6 DAC PDMA Under-Run Condition Example

The DMA request can also be generated by software enable, user sets DMAEN (DACn_CTL[2]) to 1 and TRGEN (DACn_CTL[4]) to 0, DMA request is generated periodically according to the conversion time defined by SETTLET (DACn_TCTL[9:0]) value. DAC output is updated periodically. When user clears DMAEN (DACn_CTL[2]) to 0, DAC controller will stop issuing next new PDMA transfer request. Figure 6.44-7 provides an example of DAC continuous conversion with software PDMA mode. Note that n = 0,1.

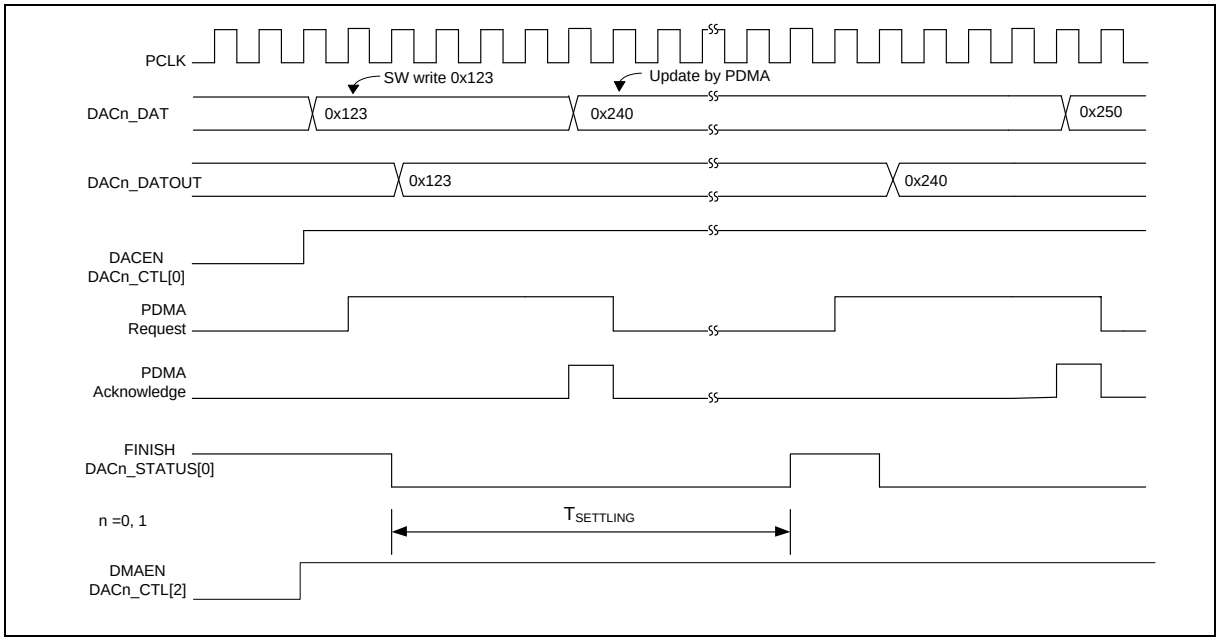


Figure 6.44-7 DAC Continuous Conversion with Software PDMA Mode

6.44.5.9 Interrupt Sources

There are two interrupt sources in the DAC controller, one is DAC data conversion finish interrupt and the other is DMA under-run interrupt as shown in Figure 6.44-8. When DAC conversion is finished, the

FINISH (DACn_STATUS[0]) is set to 1 and an interrupt occurs while DACIEN (DACn_CTL[1]) is enabled. If a new DMA trigger event occurs during DAC data conversion period, the DMA under-run flag DMAUDR (DACn_STATUS[1]) is generated and an interrupt occurs if DMAURIEN (DACn_CTL[3]) is enabled. Note that n = 0,1.

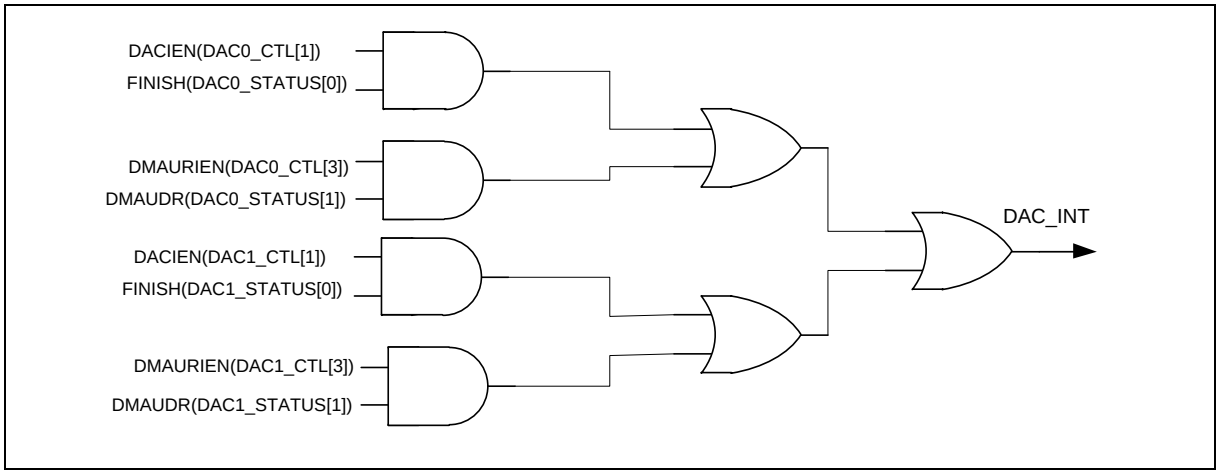


Figure 6.44-8 DAC Interrupt Source

6.44.5.10 GPIO Mode Enable

To make I/O more flexible, GPIO Mode Enable is added. When the GPIO Mode Enable (DACn_CTL[17]) is set to 0, it will execute according to the original behavior, that is, the output of the DAC can be seen from the DACn_OUT I/O. When the GPIO Mode Enable (DACn_CTL[17]) is set to 1, the DAC will still provide internal voltage, but the original DACn_OUT I/O can be used for other multiple function pin.

6.44.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write

Register	Offset	R/W	Description	Reset Value
DAC Base Address: **DAC_BA = 0x4004_7000**				
DAC0_CTL	DAC_BA+0x00	R/W	DAC0 Control Register	0x0000_0000
DAC0_SWTRG	DAC_BA+0x04	R/W	DAC0 Software Trigger Control Register	0x0000_0000
DAC0_DAT	DAC_BA+0x08	R/W	DAC0 Data Holding Register	0x0000_0000
DAC0_DATOUT	DAC_BA+0x0C	R	DAC0 Data Output Register	0x0000_0000
DAC0_STATUS	DAC_BA+0x10	R/W	DAC0 Status Register	0x0000_0000
DAC0_TCTL	DAC_BA+0x14	R/W	DAC0 Timing Control Register	0x0000_0000
DAC_GRPDAT	DAC_BA+0x30	R/W	DAC Group Mode Data Holding Register	0x0000_0000
DAC1_CTL	DAC_BA+0x40	R/W	DAC1 Control Register	0x0000_0000
DAC1_SWTRG	DAC_BA+0x44	R/W	DAC1 Software Trigger Control Register	0x0000_0000
DAC1_DAT	DAC_BA+0x48	R/W	DAC1 Data Holding Register	0x0000_0000
DAC1_DATOUT	DAC_BA+0x4C	R	DAC1 Data Output Register	0x0000_0000
DAC1_STATUS	DAC_BA+0x50	R/W	DAC1 Status Register	0x0000_0000
DAC1_TCTL	DAC_BA+0x54	R/W	DAC1 Timing Control Register	0x0000_0000

6.44.7 Register Description

DAC0 Control Register (DAC0_CTL)

Register	Offset	R/W	Description	Reset Value
DAC0_CTL	DAC_BA+0x00	R/W	DAC0 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					BYPASS	GPIOEN	GRPEN
15	14	13	12	11	10	9	8
BWSEL		ETRGSEL		Reserved	LALIGN	Reserved	TRGSEL
7	6	5	4	3	2	1	0
TRGSEL			TRGEN	DMAURIEN	DMAEN	DACIEN	DACEN

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**BYPASS**	**Bypass Buffer Mode** 0 = Output voltage buffer Enabled. 1 = Output voltage buffer Disabled.
[17]	**GPIOEN**	**GPIO Mode Enable Bit** 0 = GPIO pin can output DAC0_OUT voltage. 1 = GPIO pin can be used as other MFP.
[16]	**GRPEN**	**DAC Group Mode Enable Bit** 0 = DAC0 and DAC1 are not grouped. 1 = DAC0 and DAC1 are grouped.
[15:14]	**BWSEL**	**DAC Data Bit-width Selection** 00 = data is 12 bits. 01 = data is 8 bits. Others = reserved.
[13:12]	**ETRGSEL**	**External Pin Trigger Selection** 00 = Low level trigger. 01 = High level trigger. 10 = Falling edge trigger. 11 = Rising edge trigger.
[11]	**Reserved**	Reserved.
[10]	**LALIGN**	**DAC Data Left-aligned Enabled Bit** 0 = Right alignment. 1 = Left alignment.
[9]	**Reserved**	Reserved.

[8:5]	**TRGSEL**	**Trigger Source Selection** 0000 = Software trigger. 0001 = External pin DAC0_ST trigger. 0010 = Timer 0 trigger. 0011 = Timer 1 trigger. 0100 = Timer 2 trigger. 0101 = Timer 3 trigger. 0110 = EPWM0 trigger. 0111 = EPWM1 trigger. 1000 = PWM0 trigger. 1001 = PWM1 trigger.
[4]	**TRGEN**	**Trigger Mode Enable Bit** 0 = DAC event trigger mode Disabled. 1 = DAC event trigger mode Enabled.
[3]	**DMAURIEN**	**DMA Under-run Interrupt Enable Bit** 0 = DMA under-run interrupt Disabled. 1 = DMA under-run interrupt Enabled.
[2]	**DMAEN**	**DMA Mode Enable Bit** 0 = DMA mode Disabled. 1 = DMA mode Enabled.
[1]	**DACIEN**	**DAC Interrupt Enable Bit** 0 = DAC interrupt Disabled. 1 = DAC interrupt Enabled.
[0]	**DACEN**	**DAC Enable Bit** 0 = DAC Disabled. 1 = DAC Enabled.

DAC0 Software Trigger Control Register (DAC0_SWTRG)

Register	Offset	R/W	Description	Reset Value
DAC0_SWTRG	DAC_BA+0x04	R/W	DAC0 Software Trigger Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SWTRG**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SWTRG**	**Software Trigger** 0 = Software trigger Disabled. 1 = Software trigger Enabled. **Note:** User writes this bit to generate one shot pulse and it is cleared to 0 by hardware automatically; reading this bit will always get 0.

DAC0 Data Holding Register (DAC0_DAT)

Register	Offset	R/W	Description	Reset Value
DAC0_DAT	DAC_BA+0x08	R/W	DAC0 Data Holding Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DACDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DACDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DACDAT**	**DAC0 12-bit Holding Data** These bits are written by user software which specifies 12-bit conversion data for DAC output. The unused bits (DAC_DAT[3:0] in left-alignment mode and DAC_DAT[15:12] in right alignment mode) are ignored by DAC controller hardware. The DACDAT will be loaded into DAC when the trigger event occurs. 12 bit left alignment: user has to load data into DAC_DAT[15:4] bits. 12 bit right alignment: user has to load data into DAC_DAT[11:0] bits.

DAC0 Data Output Register (DAC0_DATOUT)

Register	Offset	R/W	Description	Reset Value
DAC0_DATOUT	DAC_BA+0x0C	R	DAC0 Data Output Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**DATOUT**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATOUT							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**DATOUT**	**DAC 12-bit Output Data** These bits are current digital data for DAC output conversion. It is loaded from DAC_DAT register and user cannot write it directly.

DAC0 Status Register (DAC0_STATUS)

Register	Offset	R/W	Description	Reset Value
DAC0_STATUS	DAC_BA+0x10	R/W	DAC0 Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BUSY
7	6	5	4	3	2	1	0
Reserved						DMAUDR	FINISH

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**BUSY**	**DAC Busy Flag (Read Only)** 0 = DAC is ready for next conversion. 1 = DAC is busy in conversion.
[7:2]	**Reserved**	Reserved.
[1]	**DMAUDR**	**DMA Under-run Interrupt Flag** 0 = No DMA under-run error condition occurred. 1 = DMA under-run error condition occurred. **Note:** User writes 1 to clear this bit.
[0]	**FINISH**	**DAC Conversion Complete Finish Flag** 0 = DAC is in conversion state. 1 = DAC conversion finish. **Note:** This bit is set to 1 when conversion time counter counts to SETTLET. It is cleared to 0 when DAC starts a new conversion. User writes 1 to clear this bit to 0.

DAC0 Timing Control Register (DAC0_TCTL)

Register	Offset	R/W	Description	Reset Value
DAC0_TCTL	DAC_BA+0x14	R/W	DAC0 Timing Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SETTLET**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETTLET							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**SETTLET**	**DAC Output Settling Time** User software needs to write appropriate value to these bits to meet DAC conversion settling time base on PCLK (APB clock) speed. For example, DAC controller clock speed is 72 MHz and DAC conversion settling time is 1 us, SETTLETvalue must be greater than 0x48. SELTTLET = DAC controller clock speed x settling time.

DAC Group Data Register (DAC_GRPDAT)

Register	Offset	R/W	Description	Reset Value
DAC_GRPDAT	DAC_BA+0x30	R/W	DAC Group Mode Data Holding Register	0x0000_0000

31	30	29	28	27	26	25	24
DAC1DAT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DAC1DAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DAC0DAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAC0DAT							

Bits	Description	
[31:16]	**DAC1DAT**	**DAC1 12-bit Holding Data** In group mode, user can write these bits for DAC1 12-bit conversion data. The unused bits (DAC_GRPDAT[3:0] in left-alignment mode and DAC_GRPDAT[15:12] in right alignment mode) are ignored by DAC controller hardware. 12 bit left alignment: user has to load data into DAC_GRPDAT[15:4] bits. 12 bit right alignment: user has to load data into DAC_GRPDAT[11:0] bits. **Note:** In group mode**,** the advantage of writing 12-bit conversion data in DAC_GRPDAT[31:16] is that can share one PDMA transfer mechanism. **Note:** After setting GRPEN(DAC0_CTL[16]), user can write 12-bit conversion data for DAC1 in DAC_GRPDAT[31:16]
[15:0]	**DAC0DAT**	**DAC0 12-bit Holding Data** These bits are written by user software which specifies 12-bit conversion data for DAC output. The unused bits (DAC_GRPDAT[3:0] in left-alignment mode and DAC_GRPDAT[15:12] in right alignment mode) are ignored by DAC controller hardware. 12 bit left alignment: user has to load data into DAC_GRPDAT[15:4] bits. 12 bit right alignment: user has to load data into DAC_GRPDAT[11:0] bits. **Note:** In group mode**,** the advantage of writing 12-bit conversion data in DAC_GRPDAT[15:0] is that can share one PDMA transfer mechanism. **Note:** After setting GRPEN(DAC0_CTL[16]), user can write 12-bit conversion data for DAC0 in DAC_GRPDAT[15:0].

DAC1 Control Register (DAC1_CTL)

Register	Offset	R/W	Description	Reset Value
DAC1_CTL	DAC_BA+0x40	R/W	DAC1 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					BYPASS	GPIOEN	Reserved
15	14	13	12	11	10	9	8
BWSEL		ETRGSEL		Reserved	LALIGN	Reserved	TRGSEL
7	6	5	4	3	2	1	0
TRGSEL			TRGEN	DMAURIEN	DMAEN	DACIEN	DACEN

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**BYPASS**	**Bypass Buffer Mode** 0 = Output voltage buffer Enabled. 1 = Output voltage buffer Disabled.
[17]	**GPIOEN**	**GPIO Mode Enable Bit** 0 = GPIO pin can output DAC0_OUT voltage. 1 = GPIO pin can be used as other MFP.
[16]	**Reserved**	Reserved.
[15:14]	**BWSEL**	**DAC Data Bit-width Selection** 00 = Data is 12 bits. 01 = Data is 8 bits. Others = reserved.
[13:12]	**ETRGSEL**	**External Pin Trigger Selection** 00 = Low level trigger. 01 = High level trigger. 10 = Falling edge trigger. 11 = Rising edge trigger.
[11]	**Reserved**	Reserved.
[10]	**LALIGN**	**DAC Data Left-aligned Enable Control** 0 = Right alignment. 1 = Left alignment.
[9]	**Reserved**	Reserved.
[8:5]	**TRGSEL**	**Trigger Source Selection** 0000 = Software trigger. 0001 = External pin DAC0_ST trigger.

		0010 = Timer 0 trigger. 0011 = Timer 1 trigger. 0100 = Timer 2 trigger. 0101 = Timer 3 trigger. 0110 = EPWM0 trigger. 0111 = EPWM1 trigger. 1000 = PWM0 trigger. 1001 = PWM1 trigger.
[4]	**TRGEN**	**Trigger Mode Enable Bit** 0 = DAC event trigger mode Disabled. 1 = DAC event trigger mode Enabled.
[3]	**DMAURIEN**	**DMA Under-run Interrupt Enable Bit** 0 = DMA under-run interrupt Disabled. 1 = DMA under-run interrupt Enabled.
[2]	**DMAEN**	**DMA Mode Enable Bit** 0 = DMA mode Disabled. 1 = DMA mode Enabled.
[1]	**DACIEN**	**DAC Interrupt Enable Bit** 0 = DAC interrupt Disabled. 1 = DAC interrupt Enabled.
[0]	**DACEN**	**DAC Enable Bit** 0 = DAC Disabled. 1 = DAC Enabled.

DAC1 Software Trigger Control Register (DAC1_SWTRG)

Register	Offset	R/W	Description	Reset Value
DAC1_SWTRG	DAC_BA+0x44	R/W	DAC1 Software Trigger Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SWTRG**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SWTRG**	**Software Trigger** 0 = Software trigger Disabled. 1 = Software trigger Enabled. **Note:** User writes this bit to generate one shot pulse and it is cleared to 0 by hardware automatically; Reading this bit will always get 0.

DAC1 Data Holding Register (DAC1_DAT)

Register	Offset	R/W	Description	Reset Value
DAC1_DAT	DAC_BA+0x48	R/W	DAC1 Data Holding Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DACDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DACDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DACDAT**	**DAC 12-bit Holding Data** These bits are written by user software which specifies 12-bit conversion data for DAC output. The unused bits (DAC_DAT[3:0] in left-alignment mode and DAC_DAT[15:12] in right alignment mode) are ignored by DAC controller hardware. The DACDAT will be loaded into DAC when the trigger event occurs. 12 bit left alignment: user has to load data into DAC_DAT[15:4] bits. 12 bit right alignment: user has to load data into DAC_DAT[11:0] bits.

DAC1 Data Output Register (DAC1_DATOUT)

Register	Offset	R/W	Description	Reset Value
DAC1_DATOUT	DAC_BA+0x4C	R	DAC1 Data Output Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**DATOUT**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATOUT							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**DATOUT**	**DAC 12-bit Output Data** These bits are current digital data for DAC output conversion. It is loaded from DAC_DAT register and user cannot write it directly.

DAC1 Status Register (DAC1_STATUS)

Register	Offset	R/W	Description	Reset Value
DAC1_STATUS	DAC_BA+0x50	R/W	DAC1 Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BUSY
7	6	5	4	3	2	1	0
Reserved						DMAUDR	FINISH

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**BUSY**	**DAC Busy Flag (Read Only)** 0 = DAC is ready for next conversion. 1 = DAC is busy in conversion.
[7:2]	**Reserved**	Reserved.
[1]	**DMAUDR**	**DMA Under-run Interrupt Flag** 0 = No DMA under-run error condition occurred. 1 = DMA under-run error condition occurred. **Note:** User writes 1 to clear this bit.
[0]	**FINISH**	**DAC Conversion Complete Finish Flag** 0 = DAC is in conversion state. 1 = DAC conversion finished. **Note:** This bit set to 1 when conversion time counter counts to SETTLET. It is cleared to 0 when DAC starts a new conversion. User writes 1 to clear this bit to 0.

DAC1 Timing Control Register (DAC1_TCTL)

Register	Offset	R/W	Description	Reset Value
DAC1_TCTL	DAC_BA+0x54	R/W	DAC1 Timing Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SETTLET**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETTLET							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**SETTLET**	**DAC Output Settling Time** User software needs to write appropriate value to these bits to meet DAC conversion settling time base on PCLK (APB clock) speed. For example, DAC controller clock speed is 72 MHz and DAC conversion settling time is 1 us, SETTLET value must be greater than 0x48. SELTTLET = DAC controller clock speed x setting time.

6.45 OP Amplifier (OPA)

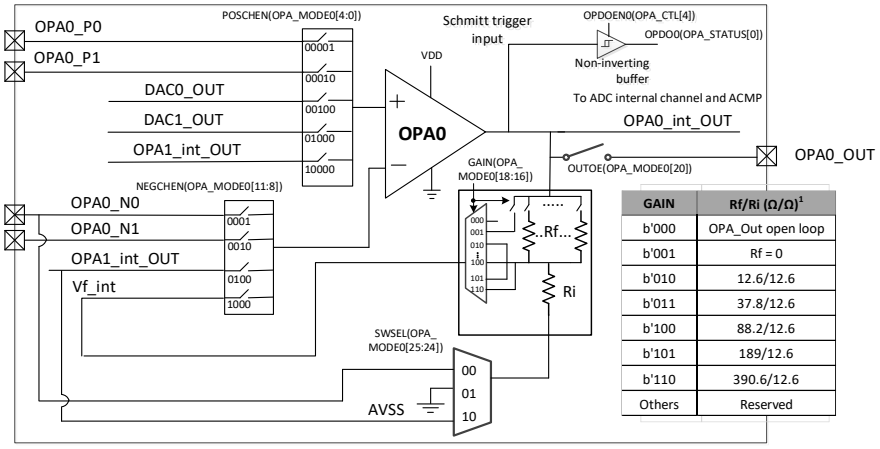
6.45.1 Overview

This device is equipped with three operational amplifiers. Users can enable each of them individually by their application purpose.These three OP amplifier circuits can also be used in the application of Programmable Gain Amplifier (PGA) or OPA mode, unit-gain mode.

6.45.2 Features

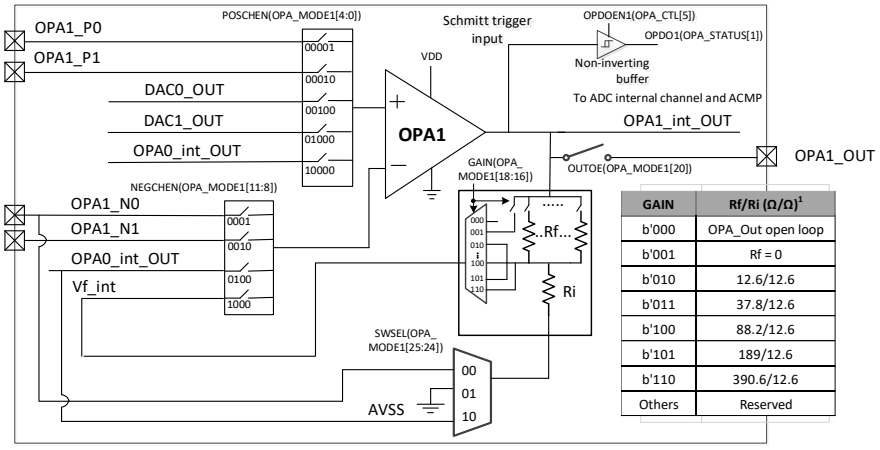
- Analog input voltage range: 0~V_{DD}.
- Supports up to 3 operational amplifiers
- Supports to use Schmitt trigger buffer output for simple comparator function
- Supports Schmitt trigger buffer output interrupts.
- Supports wake-up function
- PGA with non-inverting gain setting:
  - 2x, 4x, 8x, 16x, 32x
- PGA with inverting gain setting:
  - -1x, -3x, -7x, -15x, -31x
- 8 MHz gain bandwidth product

6.45.3 Block Diagram



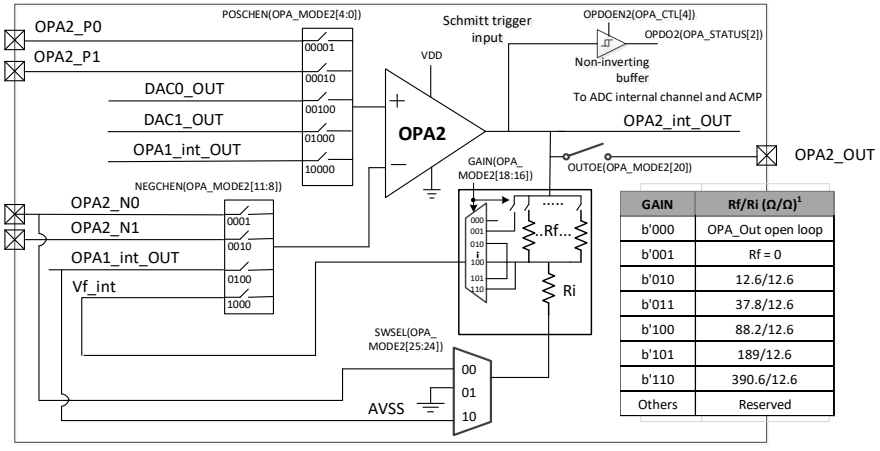
GAIN	Rf/Ri (Ω/Ω)[1]
b'000	OPA_Out open loop
b'001	Rf = 0
b'010	12.6/12.6
b'011	37.8/12.6
b'100	88.2/12.6
b'101	189/12.6
b'110	390.6/12.6
Others	Reserved

1. Typical value guaranteed by characteristic, not tested in production.



GAIN	Rf/Ri (Ω/Ω)[1]
b'000	OPA_Out open loop
b'001	Rf = 0
b'010	12.6/12.6
b'011	37.8/12.6
b'100	88.2/12.6
b'101	189/12.6
b'110	390.6/12.6
Others	Reserved

1. Typical value guaranteed by characteristic, not tested in production.



GAIN	Rf/Ri (Ω/Ω)[1]
b'000	OPA_Out open loop
b'001	Rf = 0
b'010	12.6/12.6
b'011	37.8/12.6
b'100	88.2/12.6
b'101	189/12.6
b'110	390.6/12.6
Others	Reserved

1. Typical value guaranteed by characteristic, not tested in production.

Figure 6.45-1 OP Amplifier Block Diagram

6.45.4 Basic Configuration

6.45.4.1 OPA0 Basic Configuration

- Clock source Configuration
  - Enable OPA0 peripheral clock in OPACKEN (LPSCC_CLKEN0[27]).
- Reset Configuration
  - Reset OPA0 controller in OPARST (LPSCC_IPRST0[27]).
-

6.45.4.2 OPA1 Basic Configuration

- Clock source Configuration
  - Enable OPA1 peripheral clock in OPACKEN (LPSCC_CLKEN0[27]).
- Reset Configuration
  - Reset OPA1 controller in OPARST (LPSCC_IPRST0[27]).
-

6.45.4.3 OPA2 Basic Configuration

- Clock source Configuration
  - Enable OPA2 peripheral clock in OPACKEN (LPSCC_CLKEN0[27]).
- Reset Configuration
  - Reset OPA2 controller in OPARST (LPSCC_IPRST0[27]).
-

6.45.5 Functional Description

6.45.5.1 OP Amplifier Function

The OP amplifier can be enabled by setting OPENn (OPA_CTL[2:0]) bits where n=0, 1, 2 for OPA0, OPA1 and OPA2 respectively. OP amplifier positive input channel must be enabled by selecting corresponding channel enable bit POSCHEN (OPA_MODEn[3:0]). Besides, negative input channel must be enabled by selecting corresponding channel enable bit NEGCHEN (OPA_MODEn[11:8]), n= 0,1,2. The OPA pin functions can refer to Figure 6.45-1 and OPA output is also internally connected to ACMP and ADC channel for measurement requirement. Schmitt trigger enable can be set by setting OPDOENn (OPA_CTL[6:4]) bits. OP amplifiers digital output OPDOn (OPA_STATUS[2:0]) are set only when the output state of OP amplifiers changes and Schmitt trigger enable bits are set. OP amplifier digital output is always 0 when the Schmitt trigger buffer is disabled.

6.45.5.2 PGA Mode Function

The OP amplifier circuit can be used in the application of Programmable Gain Amplifier (PGA) or unit-gain mode selected by GAIN (OPA_MODEn[18:16]), n= 0, 1, 2. When OP amplifier is configured as PGA mode, which can amplify signal by -1, -3, -5, -7, -15, -31 as inverting amplifier and amplify signal by 1, 2, 4, 8, 16, 32 as non-inverting gain setting.

6.45.5.3 Calibration Function

The circuit has five trim bits which can be used to calibrate the offset voltage. Offset voltage comes from both mismatch of NMOS-type differential and PMOS-type differential input stages. Calibration can be started by setting CALTRGn (OPA_CALCTL[n+2:n]) and setting CALCLKn (OPA_CALCTL[2n+7:2n+6]) to 2'b00, n=0, 1, 2. OPENn (OPA_CTL[2:0]) and OPDOENn (OPA_CTL[6:4]) should also be set in calibration function. After calibration, the input voltage offset can be reduced to be within ±1mV(typ).The rail-to-rail common mode input range is achieved by using an NMOS and a PMOS differential pairs connected in parallel. Reading the OPA_CALST register can monitor calibration status, which inculde

calibration done status, NMOS, and PMOS calibration result status. Take a brief calibration flow for example, once OPENn (OPA_CTL[2:0]) is enabled, hardware will automatically load default calibrated trim values to compensate offset voltage. Therefore, user can directly use OP amplifier without doing additional calibration action. If user wants to start calibration function again, the newer calibrated trim value will be updated after calibration done. Noted that every time OPEN is set, user must set CALTRGn (OPA_CALCTL[1:0]) to start calibration function getting newer offset trim values, or OP amplifier will operating with elder offset trim values. It is noted that n=0, 1, 2.

6.45.5.4 Interrupt and Wake-up Sources

If chip is not in Power-down mode, OPDOIF0 (OPA_STATUS[4]), OPDOIF1 (OPA_STATUS[5]) and OPDOIF2 (OPA_STATUS[6]) interrupt flag are set respectively by hardware whenever digital output interrupt enable OPDOIEN0 (OPA_CTL[8]), OPDOIEN1 (OPA_CTL[9]) and OPDOIEN2 (OPA_CTL[10]) are set and the OPA0, 1, 2 Schmitt trigger non-inverting buffer output change states. If chip is in Power-down mode, OPDOIF0 (OPA_STATUS[4]), OPDOIF1 (OPA_STATUS[5]) and OPDOIF2 (OPA_STATUS[6]) interrupt flag are set respectively by hardware whenever digital output wake-up enable OPDOWKE0 (OPA_CTL[12]), OPDOWKE1 (OPA_CTL[13]) and OPDOWKE2 (OPA_CTL[14]) are set and the OPA0, 1, 2 Schmitt trigger non-inverting buffer output change states. The flag bit is cleared by writing 1 to it. Schmitt trigger buffer outputs of OP amplifier0, 1, 2 can be as one of the OPA interrupt sources.

OPDOWKF0 (OPA_STATUS[8]), OPDOWKF1 (OPA_STATUS[9]) and OPDOWKF2 (OPA_STATUS[10]) wake-up flag are set respectively by hardware whenever digital output wake-up enable OPDOWKE0 (OPA_CTL[12]), OPDOWKE1 (OPA_CTL[13]) and OPDOWKE2 (OPA_CTL[14]) are set and the OPA0, 1, 2 Schmitt trigger non-inverting buffer digital output changes state in chip Power-down mode. The flag bit is cleared by writing 1 to it. Schmitt trigger buffer outputs of OP amplifier0, 1, 2 can be as one of the OPA wake-up sources.

Figure 6.45-1 shows the interrupt and wake-up sources of OPA.

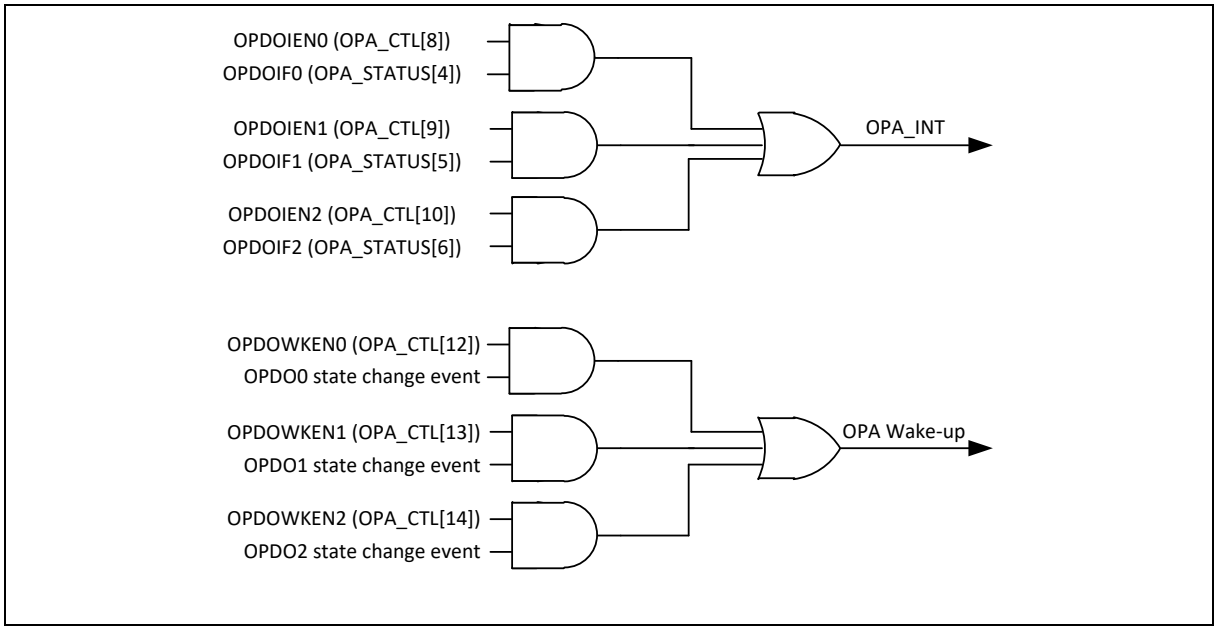


Figure 6.45-2 OP Amplifier Interrupt and Wake-up

6.45.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write, **C**: Only value 0 can be written

Register	Offset	R/W	Description	Reset Value
OPA Base Address: **OPA_BA = 0x4004_6000**				
OPA_CTL	OPA_BA+0x00	R/W	OP Amplifier Control Register	0x0000_0000
OPA_MODE0	OPA_BA+0x04	R/W	OP Amplifier Mode Control Register0	0x0000_0000
OPA_MODE1	OPA_BA+0x08	R/W	OP Amplifier Mode Control Register1	0x0000_0000
OPA_MODE2	OPA_BA+0x0C	R/W	OP Amplifier Mode Control Register2	0x0000_0000
OPA_STATUS	OPA_BA+0x20	R	OP Amplifier Status Register	0x0000_0000
OPA_CALCTL	OPA_BA+0x24	R/W	OP Amplifier Calibration Control Register	0x0000_03F0
OPA_CALST	OPA_BA+0x28	R	OP Amplifier Calibration Status Register	0x0000_0000

6.45.7 Register Description

OPA Control Register (OPA_CTL)

Register	Offset	R/W	Description	Reset Value
OPA_CTL	OPA_BA+0x00	R/W	OP Amplifier Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**OPDOWKE2**	**OPDOWKE1**	**OPDOWKE0**	**Reserved**	**OPDOIEN2**	**OPDOIEN1**	**OPDOIEN0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**OPDOEN2**	**OPDOEN1**	**OPDOEN0**	**Reserved**	**OPEN2**	**OPEN1**	**OPEN0**

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14]	**OPDOWKE2**	**OP Amplifier 2 Schmitt Trigger Digital Output Wake-up Enable Bit** 0 = OP amplifier 2 digital output wake-up function Disabled. 1 = OP amplifier 2 digital output wake-up function Enabled. **Note:** If OPDOWKE2 is set to 1 and the OP amplifier 2 Schmitt trigger non-inverting buffer digital output changes state in Power-down mode, an OPA wake-up is generated.
[13]	**OPDOWKE1**	**OP Amplifier 1 Schmitt Trigger Digital Output Wake-up Enable Bit** 0 = OP amplifier 1 digital output wake-up function Disabled. 1 = OP amplifier 1 digital output wake-up function Enabled. **Note:** If OPDOWKE1 is set to 1 and the OP amplifier 1 Schmitt trigger non-inverting buffer digital output changes state in Power-down mode, an OPA wake-up is generated.
[12]	**OPDOWKE0**	**OP Amplifier 0 Schmitt Trigger Digital Output Wake-up Enable Bit** 0 = OP amplifier 0 digital output wake-up function Disabled. 1 = OP amplifier 0 digital output wake-up function Enabled. **Note:** If OPDOWKE0 is set to 1 and the OP amplifier 0 Schmitt trigger non-inverting buffer digital output changes state in Power-down mode, an OPA wake-up is generated.
[11]	**Reserved**	Reserved.
[10]	**OPDOIEN2**	**OP Amplifier 2 Schmitt Trigger Digital Output Interrupt Enable Bit** 0 = OP amplifier 2 digital output interrupt function Disabled. 1 = OP amplifier 2 digital output interrupt function Enabled. **Note:** The OPDOIF2 interrupt flag is set by hardware whenever the OP amplifier 2 Schmitt trigger non-inverting buffer digital output changes state, in the meanwhile, if OPDOIEN2 is set to 1, an OPA interrupt request is generated.

Bits	Description	
[9]	**OPDOIEN1**	**OP Amplifier 1 Schmitt Trigger Digital Output Interrupt Enable Bit** 0 = OP amplifier 1 digital output interrupt function Disabled. 1 = OP amplifier 1 digital output interrupt function Enabled. **Note:** The OPDOIF1 interrupt flag is set by hardware whenever the OP amplifier 1 Schmitt trigger non-inverting buffer digital output changes state, in the meanwhile, if OPDOIEN1 is set to 1, an OPA interrupt request is generated.
[8]	**OPDOIEN0**	**OP Amplifier 0 Schmitt Trigger Digital Output Interrupt Enable Bit** 0 = OP amplifier 0 digital output interrupt function Disabled. 1 = OP amplifier 0 digital output interrupt function Enabled. **Note:** The OPDOIF0 interrupt flag is set by hardware whenever the OP amplifier 0 Schmitt trigger non-inverting buffer digital output changes state, in the meanwhile, if OPDOIEN0 is set to 1, an OPA interrupt request is generated.
[7]	**Reserved**	Reserved.
[6]	**OPDOEN2**	**OP Amplifier 2 Schmitt Trigger Non-inverting Buffer Enable Bit** 0 = OP amplifier 2 schmitt trigger non-invert buffer Disabled. 1 = OP amplifier 2 schmitt trigger non-invert buffer Enabled.
[5]	**OPDOEN1**	**OP Amplifier 1 Schmitt Trigger Non-inverting Buffer Enable Bit** 0 = OP amplifier 1 schmitt trigger non-invert buffer Disabled. 1 = OP amplifier 1 schmitt trigger non-invert buffer Enabled.
[4]	**OPDOEN0**	**OP Amplifier 0 Schmitt Trigger Non-inverting Buffer Enable Bit** 0 = OP amplifier 0 schmitt trigger non-invert buffer Disabled. 1 = OP amplifier 0 schmitt trigger non-invert buffer Enabled.
[3]	**Reserved**	Reserved.
[2]	**OPEN2**	**OP Amplifier 2 Enable Bit** 0 = OP amplifier 2 Disabled. 1 = OP amplifier 2 Enabled. **Note:** OP amplifier 2 output needs wait stable 20μs after OPEN2 is set.
[1]	**OPEN1**	**OP Amplifier 1 Enable Bit** 0 = OP amplifier 1 Disabled. 1 = OP amplifier 1 Enabled. **Note:** OP amplifier 1 output needs wait stable 20μs after OPEN1 is set.
[0]	**OPEN0**	**OP Amplifier 0 Enable Bit** 0 = OP amplifier 0 Disabled. 1 = OP amplifier 0 Enabled. **Note:** OP amplifier 0 output needs wait stable 20μs after OPEN0 is set.

OPA Mode Control Register0 (OPA_MODE0)

Register	Offset	R/W	Description	Reset Value
OPA_MODE0	OPA_BA+0x04	R/W	OP Amplifier Mode Control Register0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						SWSEL	
23	22	21	20	19	18	17	16
LMODE	Reserved		OUTOE	Reserved	GAIN		
15	14	13	12	11	10	9	8
Reserved				NEGCHEN			
7	6	5	4	3	2	1	0
Reserved			POSCHEN				

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:24]	**SWSEL**	**OP Amplifier 0 Resistor End Switch Selection** 00 = Input channel from OPA0 negative channel 0 (OPA0_N0) is enabled. 01 = Input channel connected to AV_{SS} is enabled. 10 = Input channel connected to OPA1_int_OUT is enabled. 11 = Reserved.
[23]	**LMODE**	**Low Power Mode Enable Bit** 0= OPA operated in normal mode. 1= OPA operated in low power mode.
[22:21]	**Reserved**	Reserved.
[20]	**OUTOE**	**OP Amplifier 0 Output Enable Bit** 0 = OP Amplifier 0 Output floating. 1 = OP Amplifier 0 Output to PAD.
[19]	**Reserved**	Reserved.
[18:16]	**GAIN**	**OP Amplifier 0 Gain Control** 3'b000 = Normal mode 3'b001 = Gain x1 3'b010 = Gain x2 3'b011 = Gain x4 3'b100 = Gain x8 3'b101 = Gain x16 3'b110 = Gian x32 3'b111 = Reserved **Note:** When inverting gain setting, these bits represent gain -1x, -3x, -7x, -15x, -31x from GIAIN = 3'b010 to 3'b110 respectively.
[15:12]	**Reserved**	Reserved.

Bits	Description	
[11:8]	**NEGCHEN**	**OP Amplifier 0 Negative Input Channel Enable Bit** 0000 = Disable all negative input channel. 0001 = OPA0_N0. 0010 = OPA0_N1. 0100 = OPA1_int_OUT. 1000 = Vf_int. **Note 1:** OPA_int_OUT means the input from OPA1 internal output **Note 2:** For example, if NEGCHEN is selected as 0011, OPA0_N0 and OPA0_N1 are both enabled.
[7:4]	**Reserved**	Reserved.
[3:0]	**POSCHEN**	**OP Amplifier 0 Positvie Input Channel Enable Bit** 00000 = Disable all positive input channel. 00001 = Enable OPA0_P0. 00010 = Enable OPA0_P1. 00100 = Enable DAC0_OUT. 01000 = Enable DAC1_OUT. 10000 = Enable OPA1_int_OUT. **Note 1:** OPA1_int_OUT means the input from OPA1 internal output **Note 2:** For example, if POSCHEN is selected as 00011, PGA0_P0 and PGA0_P1 are both enabled.

OPA Mode Control Register1 (OPA_MODE1)

Register	Offset	R/W	Description	Reset Value
OPA_MODE1	OPA_BA+0x08	R/W	OP Amplifier Mode Control Register1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						SWSEL	
23	22	21	20	19	18	17	16
LMODE	Reserved		OUTOE	Reserved	GAIN		
15	14	13	12	11	10	9	8
Reserved				NEGCHEN			
7	6	5	4	3	2	1	0
Reserved			POSCHEN				

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:24]	**SWSEL**	**OP Amplifier Resistor End Switch Selection** 00 = Input channel from OPA1 negative channel 0 (OPA1_N0) is enabled. 01 = Input channel connected to AV_{SS} is enabled. 10 = Input channel connected to OPA0_int_OUT is enabled. 11 = Reserved.
[23]	**LMODE**	**Low Power Mode Enable Bit** 0= OPA operated in normal mode. 1= OPA operated in low power mode.
[22:21]	**Reserved**	Reserved.
[20]	**OUTOE**	**OP Amplifier 1 Output Enable Bit** 0 = OP Amplifier 1 Output floating. 1 = OP Amplifier 1 Output to PAD.
[19]	**Reserved**	Reserved.
[18:16]	**GAIN**	**OP Amplifier 1 Gain Control** 3'b000 = Normal mode 3'b001 = Gain x1 3'b010 = Gain x2 3'b011 = Gain x4 3'b100 = Gain x8 3'b101 = Gain x16 3'b110 = Gian x32 3'b111 = Reserved **Note:** When inverting gain setting, these bits represent gain -1x, -3x, -7x, -15x, -31x from GIAIN = 3'b010 to 3'b110 respectively.
[15:12]	**Reserved**	Reserved.

Bits	Description	
[11:8]	**NEGCHEN**	**OP Amplifier 1 Negative Input Channel Enable Bit** 0000 = Disable all negative input channel. 0001 = OPA1_N0. 0010 = OPA1_N1. 0100 = OPA0_int_OUT. 1000 = Vf_int. **Note 1:** OPA0_int_OUT means the input from OPA0 internal output **Note 2:** For example, if NEGCHEN is selected as 0011, PGA1_N0 and PGA1_N1 are both enabled.
[7:4]	**Reserved**	Reserved.
[3:0]	**POSCHEN**	**OP Amplifier 1 Positvie Input Channel Enable Bit** 00000 = Disable all positive input channel. 00001 = OPA1_P0. 00010 = OPA1_P1. 00100 = DAC0_OUT. 01000 = DAC1_OUT. 10000 = OPA0_int_OUT. **Note 1:** OPA0_int_OUT means the input from OPA0 internal output **Note 2:** For example, if POSCHEN is selected as 00011, PGA1_P0 and PGA1_P1 are both enabled.

OPA Mode Control Register2 (OPA_MODE2)

Register	Offset	R/W	Description	Reset Value
OPA_MODE2	OPA_BA+0x0C	R/W	OP Amplifier Mode Control Register2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						SWSEL	
23	22	21	20	19	18	17	16
LMODE	Reserved		OUTOE	Reserved	GAIN		
15	14	13	12	11	10	9	8
Reserved				NEGCHEN			
7	6	5	4	3	2	1	0
Reserved			POSCHEN				

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:24]	**SWSEL**	**OP Amplifier Resistor End Switch Selection** 00 = Input channel from OPA2 negative channel 0 (OPA2_N0) is enabled. 01 = Input channel connected to AV_{SS} is enabled. 10 = Input channel connected to OPA1_int_OUT is enabled. 11 = Reserved.
[23]	**LMODE**	**Low Power Mode Enable Bit** 0= OPA operated in normal mode. 1= OPA operated in low power mode.
[22:21]	**Reserved**	Reserved.
[20]	**OUTOE**	**OP Amplifier 2 Output Enable Bit** 0 = OP Amplifier 2 Output floating. 1 = OP Amplifier 2 Output to PAD.
[19]	**Reserved**	Reserved.
[18:16]	**GAIN**	**OP Amplifier 2 Gain Control** 3'b000 = Normal mode 3'b001 = Gain x1 3'b010 = Gain x2 3'b011 = Gain x4 3'b100 = Gain x8 3'b101 = Gain x16 3'b110 = Gian x32 3'b111 = Reserved **Note:** When inverting gain setting, these bits represent gain -1x, -3x, -7x, -15x, -31x from GIAIN = 3'b010 to 3'b110 respectively.
[15:12]	**Reserved**	Reserved.

Bits	Description	
[11:8]	**NEGCHEN**	**OP Amplifier 2 Negative Input Channel Enable Bit** 0000 = Disable all negative input channel. 0001 = OPA2_N0. 0010 = OPA2_N1. 0100 = OPA1_int_OUT. 1000 = Vf_int. **Note 1:** OPA1_int_OUT means the input from OPA1 internal output **Note 2:** For example, if NEGCHEN is selected as 0011, PGA2_N0 and PGA2_N1 are both enabled.
[7:5]	**Reserved**	Reserved.
[4:0]	**POSCHEN**	**OP Amplifier 2 Positvie Input Channel Enable Bit** 00000 = Disable all positive input channel. 00001 = OPA2_P0. 00010 = OPA2_P1. 00100 = DAC0_OUT. 01000 = DAC1_OUT. 10000 = OPA1_int_OUT. **Note 1:** OPA1_int_OUT means the input from OPA1 internal output **Note 2:** For example, if POSCHEN is selected as 00011, PGA2_P0 and PGA2_P1 are both enabled.

OPA Status Register (OPA_STATUS)

Register	Offset	R/W	Description	Reset Value
OPA_STATUS	OPA_BA+0x20	R	OP Amplifier Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					OPDOWKF2	OPDOWKF1	OPDOWKF0
7	6	5	4	3	2	1	0
Reserved	OPDOIF2	OPDOIF1	OPDOIF0	Reserved	OPDO2	OPDO1	OPDO0

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**OPDOWKF2**	**OP Amplifier 2 Schmitt Trigger Digital Output Wake-Up Flag** OPDOWKF2 wake-up flag is set by hardware whenever OPDOWKE2 is set to 1 and the OP amplifier 2 Schmitt trigger non-inverting buffer digital output changes state in chip Power-down mode. **Note:** This bit is cleared by writing 1 to it.
[9]	**OPDOWKF1**	**OP Amplifier 1 Schmitt Trigger Digital Output Wake-Up Flag** OPDOWKF1 wake-up flag is set by hardware whenever OPDOWKE1 is set to 1 and the OP amplifier 1 Schmitt trigger non-inverting buffer digital output changes state in chip Power-down mode. **Note:** This bit is cleared by writing 1 to it.
[8]	**OPDOWKF0**	**OP Amplifier 0 Schmitt Trigger Digital Output Wake-Up Flag** OPDOWKF0 wake-up flag is set by hardware whenever OPDOWKE0 is set to 1 and the OP amplifier 0 Schmitt trigger non-inverting buffer digital output changes state in chip Power-down mode. **Note:** This bit is cleared by writing 1 to it.
[7]	**Reserved**	Reserved.
[6]	**OPDOIF2**	**OP Amplifier 2 Schmitt Trigger Digital Output Interrupt Flag** If chip is not in Power-down mode, OPDOIF2 interrupt flag is set by hardware whenever the OP amplifier 2 Schmitt trigger non-inverting buffer digital output changes state. If chip is in Power-down mode, OPDOIF2 interrupt flag is set by hardware whenever the OP amplifier 2 Schmitt trigger non-inverting buffer digital output changes state and OPDOWKE2 is set to 1. **Note:** This bit is cleared by writing 1 to it.
[5]	**OPDOIF1**	**OP Amplifier 1 Schmitt Trigger Digital Output Interrupt Flag** If chip is not in Power-down mode, OPDOIF1 interrupt flag is set by hardware whenever the OP amplifier 1 Schmitt trigger non-inverting buffer digital output changes state. If chip is in Power-down mode, OPDOIF1 interrupt flag is set by hardware whenever the OP amplifier 1 Schmitt trigger non-inverting buffer digital output changes state and OPDOWKE1 is set to 1. **Note:** This bit is cleared by writing 1 to it.

[4]	**OPDOIF0**	**OP Amplifier 0 Schmitt Trigger Digital Output Interrupt Flag** If chip is not in Power-down mode, OPDOIF0 interrupt flag is set by hardware whenever the OP amplifier 0 Schmitt trigger non-inverting buffer digital output changes state. If chip is in Power-down mode, OPDOIF0 interrupt flag is set by hardware whenever the OP amplifier 0 Schmitt trigger non-inverting buffer digital output changes state and OPDOWKE0 is set to 1. **Note:** This bit is cleared by writing 1 to it.
[3]	**Reserved**	Reserved.
[2]	**OPDO2**	**OP Amplifier 2 Digital Output** Synchronized to the APB clock to allow reading by software. Cleared when the Schmitt trigger buffer is disabled (OPDOEN2 = 0).
[1]	**OPDO1**	**OP Amplifier 1 Digital Output** Synchronized to the APB clock to allow reading by software. Cleared when the Schmitt trigger buffer is disabled (OPDOEN1 = 0).
[0]	**OPDO0**	**OP Amplifier 0 Digital Output** Synchronized to the APB clock to allow reading by software. Cleared when the Schmitt trigger buffer is disabled (OPDOEN0 = 0).

OPA Calibration Control Register (OPA_CALCTL)

Register	Offset	R/W	Description	Reset Value
OPA_CALCTL	OPA_BA+0x24	R/W	OP Amplifier Calibration Control Register	0x0000_03F0

31	30	29	28	27	26	25	24
Reserved					TRIMOPT2	TRIMOPT1	TRIMOPT0
23	22	21	20	19	18	17	16
Reserved					CALRVS2	CALRVS1	CALRVS0
15	14	13	12	11	10	9	8
Reserved						CALCLK2	
7	6	5	4	3	2	1	0
CALCLK1		CALCLK0		Reserved	CALTRG2	CALTRG1	CALTRG0

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26]	**TRIMOPT2**	**OPA2 Calibration Trim Option** 0 = Calibration trim from NMOS to PMOS. 1 = Calibration trim from PMOS to NMOS. **Note:** This option is only effective when CALRVS2 = 0 (OPA_CALCTL[6])
[25]	**TRIMOPT1**	**OPA1 Calibration Trim Option** 0 = Calibration trim from NMOS to PMOS. 1 = Calibration trim from PMOS to NMOS. **Note:** This option is only effective when CALRVS1 = 0 (OPA_CALCTL[5])
[24]	**TRIMOPT0**	**OPA0 Calibration Trim Option** 0 = Calibration trim from NMOS to PMOS 1 = Calibration trim from PMOS to NMOS **Note:** This option is only effective when CALRVS0 = 0 (OPA_CALCTL[4])
[23:19]	**Reserved**	Reserved.
[18]	**CALRVS2**	**OPA2 Calibration Reference Voltage Selection** 0 = V_{REF} is $\frac{1}{2}$ AV_{DD}. 1 = V_{REF} from high vcm to low vcm.
[17]	**CALRVS1**	**OPA1 Calibration Reference Voltage Selection** 0 = V_{REF} is $\frac{1}{2}$ AV_{DD}. 1 = V_{REF} from high vcm to low vcm.
[16]	**CALRVS0**	**OPA0 Calibration Reference Voltage Selection** 0 = V_{REF} is $\frac{1}{2}$ AV_{DD}. 1 = V_{REF} from high vcm to low vcm.
[15:10]	**Reserved**	Reserved.

Bits	Description	
[9:8]	**CALCLK2**	**OP Amplifier 2 Calibration Clock Rate Selection** 00 = 1 kHz. 01 = Reserved. 10 = Reserved. 11 = Reserved.
[7:6]	**CALCLK1**	**OP Amplifier 1 Calibration Clock Rate Selection** 00 = 1 kHz. 01 = Reserved. 10 = Reserved. 11 = Reserved.
[5:4]	**CALCLK0**	**OP Amplifier 0 Calibration Clock Rate Selection** 00 = 1 kHz. 01 = Reserved. 10 = Reserved. 11 = Reserved.
[3]	**Reserved**	Reserved.
[2]	**CALTRG2**	**OP Amplifier 2 Calibration Trigger Bit** 0 = Calibration is stopped. 1 = Calibration is triggered. **Note 1:** Before this bit is enabled, OPEN2 should be set and the internal high speed RC oscillator (HIRC) should be enabled in advance. **Note 2:** Hardware will auto clear this bit when calibration is finished.
[1]	**CALTRG1**	**OP Amplifier 1 Calibration Trigger Bit** 0 = Calibration is stopped. 1 = Calibration is triggered. **Note 1:** Before this bit is enabled, OPEN1 should be set and the internal high speed RC oscillator (HIRC) should be enabled in advance. **Note 2:** Hardware will auto clear this bit when calibration is finished.
[0]	**CALTRG0**	**OP Amplifier 0 Calibration Trigger Bit** 0 = Calibration is stopped. 1 = Calibration is triggered. **Note 1:** Before this bit is enabled, OPEN0 should be set and the internal high speed RC oscillator (HIRC) should be enabled in advance. **Note 2:** Hardware will auto clear this bit when calibration is finished.

OPA Calibration Status Register (OPA_CALST)

Register	Offset	R/W	Description	Reset Value
OPA_CALST	OPA_BA+0x28	R	OP Amplifier Calibration Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					CALPS2	CALNS2	DONE2
7	6	5	4	3	2	1	0
Reserved	CALPS1	CALNS1	DONE1	Reserved	CALPS0	CALNS0	DONE0

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**CALPS2**	**OP Amplifier 2 Calibration Result Status for PMOS** 0 = Pass. 1 = Fail.
[9]	**CALNS2**	**OP Amplifier 2 Calibration Result Status for NMOS** 0 = Pass. 1 = Fail.
[8]	**DONE2**	**OP Amplifier 2 Calibration Done Status** 0 = Calibrating. 1 = Calibration Done.
[7]	**Reserved**	Reserved.
[6]	**CALPS1**	**OP Amplifier 1 Calibration Result Status for PMOS** 0 = Pass. 1 = Fail.
[5]	**CALNS1**	**OP Amplifier 1 Calibration Result Status for NMOS** 0 = Pass. 1 = Fail.
[4]	**DONE1**	**OP Amplifier 1 Calibration Done Status** 0 = Calibrating. 1 = Calibration Done.
[3]	**Reserved**	Reserved.
[2]	**CALPS0**	**OP Amplifier 0 Calibration Result Status for PMOS** 0 = Pass. 1 = Fail.

Bits	Description	
[1]	**CALNS0**	**OP Amplifier 0 Calibration Result Status for NMOS** 0 = Pass. 1 = Fail.
[0]	**DONE0**	**OP Amplifier 0 Calibration Done Status** 0 = Calibrating. 1 = Calibration Done.

6.46 Peripherals Interconnection

6.46.1 Overview

Some peripherals have interconnections which allow autonomous communication or synchronous action between peripherals without needing to involve the CPU. Peripherals interact without CPU saves CPU resources, reduces power consumption, operates with no software latency and fast responds.

6.46.2 Peripherals Interconnect Matrix Table

Source		Destination																			
		EADC	LPADC	ACMP		DAC	OPA	TMR	EPWM	PWM	SYSTEM	ECAP	EQEI	TK	UTCPD	LPTMR	LPUART	LPI2C	LPSPI	LPDMA
		EADC	(NPD0~4)	(NPD0~5)	ACMP_CRV (NPD0~4)	Buffered DAC (NPD0~2)	(NPD0~4)	Timer w/ basic PWM (NPD0~2)	EPWM	PWM	HIRCTRIM	ECAP	EQEI	Touch (NPD0~2)	UTCPD	LPTMR0/1 NPD0~4)	LPUART0 (NPD0~4)	LPI2C0 (NPD0~4)	LPSPI0 (NPD0~4)	LPDMA (NPD0~4)
ACMP	(NPD0~5)	20	20	-	-	-	-	5,7	8,12	8	-	24	-	-	27	5,7	-	-	-	-
	ACMP_CRV (NPD0~4)	-	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
CLKCTL	HXT clock fail	-	-	-	-	-	-	-	8	8	-	-	-	-	-	-	-	-	-	-
DAC	DAC (NPD0~2)	-	-	3	-	-	22	-	-	-	-	-	-	-	-	-	-	-	-	-
EADC	EADC	-	-	-	-	-	-	-	8	8	-	-	-	-	27, 28	-	-	-	-	-
EPWM	EPWM	1	1	-	-	4	-	-	10,11	10	-	-	-	-	-	-	-	-	-	-
EQEI	EQEI	-	-	-	-	-	-	-	-	-	-	15	-	-	-	-	-	-	-	-
GPIO	EINT (NPD0~2)	-	-	-	-	-	-	-	23	23	-	-	-	-	27,29	-	-	-	-	-
LPADC	(NPD0~4)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	30
LPI2C	LPI2C	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	30
LPSPI	LPSPI	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	30
LPTMR	LPTMR (NPD0~4)	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	25	25	25	30
LPUART	LPUART	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	30
OPA	OPA output (NPD0~4)	18	18	21	-	-	22	-	-	-	-	-	-	-	-	-	-	-	-	-
PWM	PWM	1	1	-	-	4	-	-	10	10	-	-	-	-	-	-	-	-	-	-
RTC	VBAT (NPD0~5)	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	LXT (NPD0~5)	-	-	-	-	-	-	5	-	-	13	-	-	-	-	-	-	-	-	-
	RTC (NPD0~5)	-	-	-	-	-	-	-	-	-	-	-	-	15	-	-	-	-	-	-
SYSTEM	VBG (NPD0~5)	2	2	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	T. Sensor (NPD0~4)	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Internal V_{REF} (NPD0~4)	17	17	-	21	17	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	HXT	-	-	-	-	-	-	5	-	-	-	-	-	-	-	-	-	-	-	-
	HIRC48	-	-	-	-	-	-	5	-	-	-	-	-	-	-	-	-	-	-	-
	HIRC12 (NPD0~4)	-	-	-	-	-	-	5	-	-	-	-	-	-	-	-	-	-	-	-
	LIRC (NPD0~5)	-	-	-	-	-	-	5	-	-	-	-	-	-	-	-	-	-	-	-
	MIRC (NPD0~4)	-	-	-	-	-	-	5	-	-	-	-	-	-	-	-	-	-	-	-
	PLL	-	-	-	-	-	-	5	-	-	-	-	-	-	-	-	-	-	-	-
	BOD (NPD0~5)	-	-	-	-	-	-	-	8	8	-	-	-	-	-	-	-	-	-	-
	CPU Lockup	-	-	-	-	-	-	-	8	8	-	-	-	-	-	-	-	-	-	-
	SRAM parity error	-	-	-	-	-	-	-	8	8	-	-	-	-	-	-	-	-	-	-
	WKIO (NPD0~5)	-	25	-	-	-	-	-	-	-	-	-	-	-	-	-	25	25	25	-
TMR	TMR (NPD0~2)	1	1	-	-	4	-	7	9	9	-	-	16	15	-	-	-	-	-	-
TTMR	TTMR (NPD0~4)	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	25	25	25	30
USB1	USB1.1 Device	-	-	-	-	-	-	7	-	-	13	-	-	-	-	-	-	-	-	-
UTCPD	UTCPD	-	-	-	-	-	-	-	26	26	-	-	-	-	-	-	-	-	-	-

Table 6.46-1 Peripherals Interconnect Matrix Table

6.46.3 Functional Description

1. From EPWM, PWM, TMR, LPTMR, TTMR to EADC/LPADC

- EPWM Trigger EADC Conversion

  EPWM can be one of the EADC conversion trigger source.

  Setting the EADC external hardware trigger in section 6.42.5.7 (EADC Start Synchronous with EPWM/PWM Trigger)

  The detail EPWM trigger conditions are described in section 6.15.5.27(EPWM Trigger EADC/LPADC/DAC Generator）

- PWM Trigger EADC Conversion

  PWM can be one of the EADC conversion trigger source.

  Setting the EADC external hardware trigger in section 6.42.5.7 (EADC Start Synchronous with EPWM/PWM Trigger).

  The detail PWM trigger conditions are described in section 6.14.5.23(PWM Trigger EADC/LPADC Generator).

- TMR Trigger EADC Conversion

  TMR can be one of the EADC conversion trigger source. When timer counter value matches the timer compared value or when the TMx_EXT pin edge transition meets setting, timer will trigger the ADC to start the conversion.

  Setting the EADC external hardware trigger in section 6.42.5.6 (EADC Trigger by Timer Trigger and External Pin).

  The detail TMR trigger conditions are described in section 6.11.5.6 (Timer Trigger Function)and 6.11.6.12(PWM Trigger EADC, LPADC, PDMA, DAC Generator).

- EPWM Trigger LPADC Conversion

  EPWM can be one of the LPADC conversion trigger source.

  Setting the LPADC external hardware trigger in section 6.43.5.10 (EPWM Trigger)

  The detail EPWM trigger conditions are described in section 6.15.5.27 (EPWM Trigger EADC/LPADC/DAC Generator).

- PWM Trigger LPADC Conversion

  PWM can be one of the LPADC conversion trigger source.

  Setting the LPADC external hardware trigger in section 6.43.5.11 (PWM Trigger).

  The detail PWM trigger conditions are described in section 6.11.6.12 (PWM Trigger EADC, LPADC, PDMA, DAC Generator).

- TMR Trigger LPADC Conversion

  TMR can be one of the LPADC conversion trigger source. When timer counter value matches the timer compared value or when the TMx_EXT pin edge transition meets setting, TMR will trigger the LPADC to start the conversion.

  Setting the LPADC external hardware trigger in section 6.43.5.9(Timer Trigger).

  The detail TMR trigger conditions are described in section 6.11.5.6(Timer Trigger Function) and 6.11.6.12 (PWM Trigger EADC, LPADC, PDMA, DAC Generator).

- LPTMR Trigger LPADC Conversion

  LPTMR can be one of the LPADC conversion trigger source. When timer counter value matches the timer compared value, LPTMR will trigger the LPADC to start the conversion.

Setting the LPADC external hardware trigger in section 6.43.5.19 (Auto-operation Mode).

The detail LPTMR trigger conditions are described in section 6.12.5.6 (Low Power Timer Trigger Function).

- TTMR Trigger LPADC Conversion

  TTMR can be one of the LPADC conversion trigger source. When timer counter value matches the timer compared value, TTMR will trigger the LPADC to start the conversion.

  Setting the LPADC external hardware trigger in section 6.43.5.19 (Auto-operation Mode).

  The detail TTMR trigger conditions are described in section 6.13.5.3 (Tick Timer Trigger Function).

2. From VBG/T.Sensor/VBAT to EADC/LPADC

- VBG/T.Sensor/VBAT can be one of the EADC conversion source

  Setting the EADC external hardware trigger in Figure 6.42-1 (EADC Converter Block Diagram)

  The detail VBG/T.Sensor/VBAT settings are described in section 6.2.18 (Internal Voltage Source Control Register (SYS_IVSCTL))

- VBG/T.Sensor/VBAT can be one of the LPADC conversion source

  Setting the LPADC external hardware trigger in Figure 6.43-1 (LPADC Controller Block Diagram)

  The detail VBG/T.Sensor/VBAT settings are described in section 6.2.18 (Internal Voltage Source Control Register (SYS_IVSCTL)).

3. From ACMP CRV/VBG/DAC to ACMP

- ACMP CRV/VBG/DAC can be the source for ACMP.

  Setting the ACMP CRV in section 6.41.5.5 (Comparator Reference Voltage (CRV))

  Setting the DAC in Figure 6.44-1 (Digital-to-Analog Converter Block Diagram)

  The block diagram of ACMP is described in Figure 6.41-1 (Analog Comparator 0/1 Block Diagram) &Figure 6.41-2 (Analog Comparator 2 Block Diagram).

4. From EPWM, PWM, TMR to DAC

- EPWM Trigger DAC Conversion

  EPWM can also be used to trigger DAC conversion.

  Setting the DAC hardware trigger input source from EPWM trigger described in section (DAC Trigger Selection).

  The detail EPWM trigger conditions are described in section 6.15.5.27 (EPWM Trigger EADC/LPADC/DAC Generator).

- PWM Trigger DAC Conversion

  PWM can also be used to trigger DAC conversion.

  Setting the DAC hardware trigger input source from PWM trigger described in section 6.44.5.6 (DAC Trigger Selection).

  The detail PWM trigger conditions are described in section 6.14.5.24 (PWM Trigger DAC Generator).

- TMR Trigger DAC Conversion

  Setting the DAC hardware trigger input source from TMR trigger described in section 6.44.5.6 (DAC Trigger Selection).

The detail TMR trigger conditions are described in section 6.11.6.12 (PWM Trigger EADC, LPADC, PDMA, DAC Generator).

5. **Measure ACMP output or input clocks pulse width**

   - TMR measure ACMP output pulse

     Setting the TMR external clock source in Figure 6.11-2 (Clock Source of Timer Controller) and TMR mode in section 6.12.5.4 (Capture Mode).

     The detail ACMP trigger conditions are described in Figure 6.41-1 (Analog Comparator 0/1 Block Diagram) & Figure 6.41-2 (Analog Comparator 2 Block Diagram).

   - LPTMR measure ACMP output pulse

     Setting the LPTMR external clock source (Clock Source of Low Power Timer Controller) and LPTMR mode in section 6.12.5.4 (Capture Mode).

     The detail ACMP trigger conditions are described in Figure 6.41-1 (Analog Comparator 0/1 Block Diagram) & Figure 6.41-2 (Analog Comparator 2 Block Diagram).

6. **From Timer0/2 to Timer1/3**

   - Inter-Timer Trigger Capture Mode

     Timer0/2 will be forced in event counting mode, counting with external event, and will generate an internal signal (INTR_TMR_TRG) to trigger Timer1/3 start or stop counting. The Timer1/3 will be forced in capture mode and start/stop trigger-counting by Timer0/2 counter status.

     The detail of inter-timer trigger capture mode are described in section 6.11.5.7 (Inter-Timer Trigger Capture Mode).

7. **From ACMP/USB to TMR**

   - TMR event counting function provides an application which can count the input event from ACMP/USB will reflect to CNT (TIMERx_CNT[23:0]) value.

     Setting the TMR event counter clock source in section 6.11.5.3(Event Counting Mode).

     The setting of ACMP is described in Figure 6.41-1 (Analog Comparator 0/1 Block Diagram) & Figure 6.41-2 (Analog Comparator 2 Block Diagram).

     The setting of USB is described in section 6.32.5.1(Serial Interface Engine (SIE)).

8. **From ACMP, BOD, Clock Fail,SRAM Parity Error and CPU Lockup to EPWM/PWM Brake Source**

   EPWM/PWM brake source can be ACMP0/1_O output signal or EADC result monitor or several different system fail conditions include clock fail, Brown-out detect, and Core lockup and SRAM Parity Error. When system fault, EPWM/PWM/TMR brake signal generated, EPWM/PWM output will be set to protect the power switch controlled by EPWM/PWM.

   The detail setting of EPWM brake function is described in section 6.15.5.23 (EPWM Brake).

   The detail setting of PWM brake function is described in section 6.14.5.19 (PWM Brake).

9. **From TMR to EPWM/PWM**

   - TMR Generates Trigger Pulses as EPWM/PWM External Clock Source

     TMR0~ TMR3 can generates trigger pules as EPWM/PWM external clock source.

     When timer counter value matches the timer compared value or when the TMx_EXT pin edge transition meets setting, timer can generate a trigger pulse by setting described in section 6.11.5.1(Timer Interrupt Flag).

     The setting of EPWM/PWM clock source are described in Figure 6.15-3 (EPWM Clock

Source Control) & Figure 6.14-3 (PWM Clock Source Control).

10.From EPWM/ PWM to EPWM/ PWM

- EPWM Synchronous Function

  Select synchronous source from EPWM0 or EPWM1 or PWM0 or PWM1, and select EPWM channels. The chosen EPWM channels will start counting at the same time once the synchronous start function is enabled and CNTSEN(EPWM_SSTRG[0]) is set.

  The detail setting of EPWM synchronous function is described in section 6.15.5.19(Synchronous function).

- PWM Synchronous Function

  Select synchronous source from EPWM0 or EPWM1 or PWM0 or PWM1, and select PWM channels. The chosen PWM channels will start counting at the same time once the synchronous start function is enabled and CNTSEN(PWM_SSTRG[0]) is set.

  The detail setting of PWM synchronous start function is described in section 6.14.5.21(Synchronous start function).

11.From EPWM to EPWM

- EPWM Synchronous Function

  EPWM synchronous function source external pin SYNC_IN and output SYNC_OUT signal to other EPWM.

  The detail setting of EPWM synchronous function is described in section 6.15.5.19 (Synchronous Function).

12.From ACMP to EPWM

- Leading edge blanking (LEB) function is used to blank the false trigger from brake source ACMP which may be caused by EPWM output transition.

  Setting the ACMP trigger in Figure 6.41-1(Analog Comparator 0/1 Block Diagram) & Figure 6.41-2 (Analog Comparator 2 Block Diagram).

  The detail EPWM trigger conditions are described in section 6.15.5.24 (LEB Function).

13.From LXT and USB 1.1 Device to HIRC TRIM & HIRC48 MHz TRIM & MIRC TRIM

- Use LXT or USB Synchronous Mode to system auto-trim RC circuit

  This chip supports auto-trim function: the HIRC trim (12 MHz RC oscillator) and RC 48 MHz oscillator and MIRC 1 MHz oscillator, according to the accurate external 32.768 kHz crystal oscillator or internal USB synchronous mode, automatically gets accurate output frequency, 0.25 % deviation within all temperature ranges.

  The detail of HIRC trim setting is described in section 6.2.10 (HIRC Auto Trim).

14.From EQEI to ECAP

- ECAP Input noise filter

  The architecture of ECAP input noise filter is similar to that one used for EQEI. With 6 sampling-rate options, it supports a wide range of filtering noise, the duration of filtered noise and the duration of the signal that is guaranteed to be sampled.

  The detail setting of modulation is described in section 6.17.5.1 (Input Noise Filter).

15. TMR0~3 and RTC trigger touch key module to sense

- TMR0~3 and RTC can be the source to trigger touch key.

  Setting the TMR trigger in section 6.11.5.6 (Timer Trigger Function).

Setting the RTC trigger in section 6.20.5.7 (Periodic Time Tick Interrupt).

The detail touch key trigger conditions are described in section 6.39.5.2 (Touch Key Periodic Scan Mode).

16.From TMR to EQEI

- TMR TIF event to EQEI

  When EQEI bit HOLDCNT(QEI_CTL[24]) set, the CNT(QEI_CNT[31:0]) content will be captured into EQEI Counter Hold Register CNTHOLD(QEI_CNTHOLD[31:0]), the data will be held until the next HOLDCNT (QEI_CTL[24]) trigger comes. The bit HOLDCNT can be set by writing 1 to it or the rising edge of timers interrupt flags TIF (TIMERx_INTSTS[0])The detail setting of modulation is described in section 6.11.5.1 (Timer Interrupt Flag).

  The detail setting of modulation is described in section 6.16.5.16 (Capture EQEI Counter).

17.Internal reference voltage source for EADC/LPADC/DAC

- The reference voltage source of EADC/LPADC/DAC can be external VREF pin/AVDD or internal reference.

  Setting the internal reference in “VREF Control Register (SYS_VREFCTL)”

18.PGA/OPA output to EADC/LPADC as input source

- The PGA/OPA output voltage can be the input source for EADC/LPADC.

  Setting the PGA/OPA in Figure 6.45-1 (OP Amplifier Block Diagram).

  Setting the EADC/LPADC in Figure 6.42-1 (EADC Converter Block Diagram) & Figure 6.43-1(LPADC Controller Block Diagram).

19.PGA/OPA output to ACMP as input source

- The PGA/OPA output voltage can be the input source for ACMP.

  Setting the PGA/OPA in Figure 6.45-1 (OP Amplifier Block Diagram).

  The setting of ACMP is described in Figure 6.41-1(Analog Comparator 0/1 Block Diagram) & Figure 6.41-2 (Analog Comparator 2 Block Diagram).

20.From ACMP to EADC

- ACMP can be the trigger source for EADC start-of-conversion

  The setting of ACMP is described in Figure 6.41-1 (Analog Comparator 0/1 Block Diagram) & Figure 6.41-2 (Analog Comparator 2 Block Diagram).

  The detail EADC start-of-conversion are described in Figure 6.42-2 (Sample Module 0~3 Block Diagram).

21.Internal reference voltage source for ACMP

- The reference voltage source of ACMP can be external VREF pin/AVDD or internal reference.

  The setting of ACMP is described in Figure 6.41-1(Analog Comparator 0/1 Block Diagram) & Figure 6.41-2 (Analog Comparator 2 Block Diagram).

  Setting the internal reference in “VREF Control Register (SYS_VREFCTL)”

22.OPA/PGA input source form DAC/OPA/PGA

- The input voltage source of OPA/PGA can be DAC or another OPA/PGA.

  Setting the DAC in Figure 6.44-1 (Digital-to-Analog Converter Block Diagram).

  Setting the OPA/PGA in Figure 6.45-1 (OP Amplifier Block Diagram).

23.EINT event trigger EPWM/PWM.

- EINT event trigger can be the trigger source of EPWM/PWM.

  Setting the EINT event trigger in section 6.6.5.8 (INT Edge Detect and Trigger Function)

  The detail EPWM trigger conditions are described in section 6.15.5.33 (External Pin Event Trigger)

  The detail PWM trigger conditions are described in section 6.14.5.29 (External Pin Event Trigger)

24.From ACMP to ECAP

- ACMP can be the source of ECAP

  Setting the ACMP trigger in Figure 6.41-1 (Analog Comparator 0/1 Block Diagram) & Figure 6.41-2 (Analog Comparator 0/1 Block Diagram).

  The detail ECAP trigger conditions are described in Figure 6.17-1 (Input Capture Timer/Counter Architecture).

25.LPTMR/TTMR/WKIO event trigger LPSPI/LPI2C/LPUART.

- LPTMR/TTMR/WKIO event trigger can be the trigger source of LPSPI/LPI2C/LPUART.

  Setting the LPTMR event trigger in section 6.12.5.6 (Low Power Timer Trigger Function).

  Setting the TTMR event trigger in section 6.13.5.3 (Tick Timer Trigger Function).

  Setting the WKIO event trigger in section 6.2.13 (Wake-up Timer (WKTMR) and Wake-up IO (WKIO) Function)..

  The detail LPSPI trigger conditions are described in section 6.30.5.10(Automatic Operaion Mode).

  The detail LPI2C trigger conditions are described in section 6.22.5.7(Automatic Operation Modes (auto-operation)).

  The detail LPUART trigger conditions are described in section 6.28.5.12 (Auto-operation Mode).

26.UTCPD VBUS event trigger EPWM/PWM brake function.

- UTCPD VBUS sink disable or source disable can be the trigger source of EPWM/PWM.

  Setting the UTCPD VBUS event trigger in section 6.35.5.6 (Disconnection Detect) & 6.35.5.7 (Discharge).

  The detail setting of EPWM brake function is described in section 6.15.5.23 (EPWM Brake)

  The detail setting of PWM brake function is described in section 6.14.5.19 (PWM Brake).

27. ACMP, EADC, EINT event trigger UTCPD function.

- ACMP, EADC, EINT can be the input sources for external VBUS overcurrent protection fault detect disable and external VCONN overcurrent fault detect disable.

  The setting of ACMP is described inFigure 6.41-1 (Analog Comparator 0/1 Block Diagram) &Figure 6.41-2 (Analog Comparator 0/1 Block Diagram).

  Setting EADC the in section 6.42.5.12 (Conversion Result Monitor by Compare Mode).

  Setting EINT the in section 6.6.5.8 (INT Edge Detect and Trigger Function).

  The detail setting of UTCPD in Figure 6.35-1(UTCPD System Block Diagram).

28.EADC conversion data for UTCPD.

- The EADC controller provides compare registers to monitor conversion results from

EADC conversion module.

Setting the EADC in section 6.42.5.12 (Conversion Result Monitor by Compare Mode).

The detail setting of UTCPD in section 6.35.5 (Functional Description).

29.EINT event to UTCPD force off VBUS input.

- EINT can be the UTCPD force off VBUS function.

  Setting the EINT in section 6.6.5.8 (INT Edge Detect and Trigger Function).

  The detail setting of UTCPD in section 6.35.5 (Functional Description).

30.LPTMR, TTMR, LPADC, LPUART, LPSPI, LPI2C trigger LPPDMA.

- LPPDMA can transfer data from LPTMR, TTMR, LPADC, LPUART, LPSPI, LPI2C to another address without CPU intervention.

  Setting the LPTMR in section 6.12.5.6 (Low Power Timer Trigger Function).

  Setting the TTMR in section 6.13.5.3 (Tick Timer Trigger Function).

  Setting the LPADC in section 6.43.5.15(LPPDMA Transfer Mode).

  Setting the LPUART in section 6.28.5.11 (LPPDMA Transfer Function).

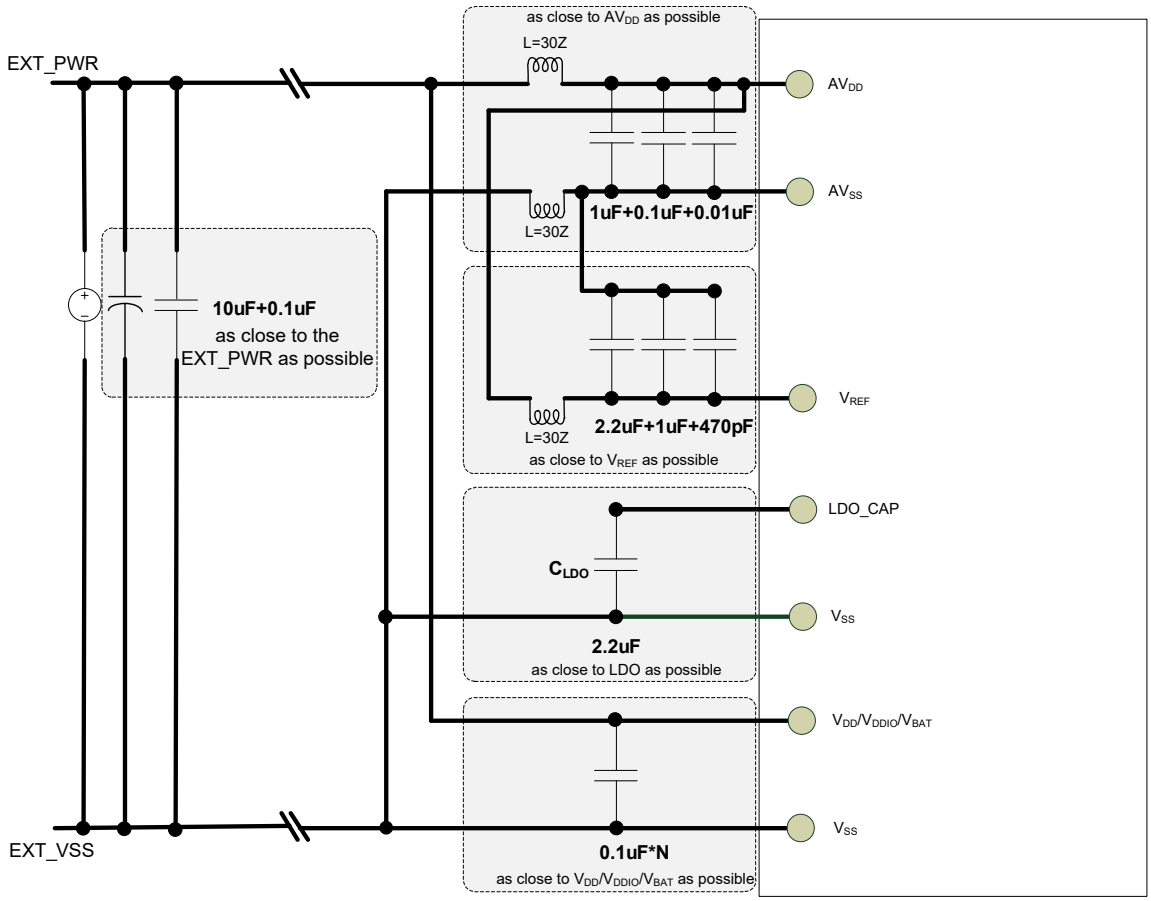
  Setting the LPSPI in section 6.30.5.7 (PDMA Transfer Function).

  Setting the LPI2C in section 6.22.5.3 (LPPDMA Transfer Function).

  The detail setting of LPPDMA in section 6.9.5 (Functional Description).

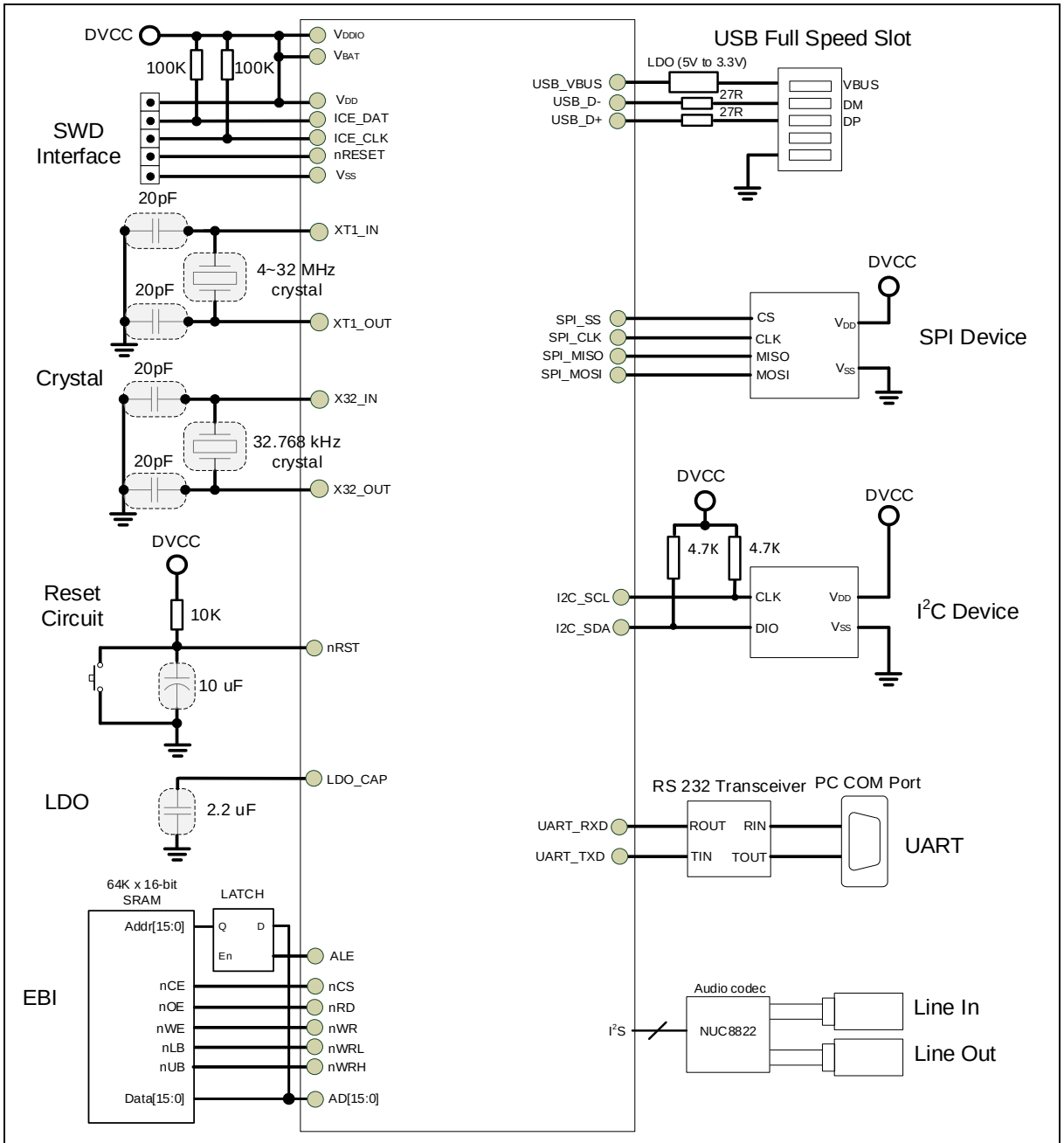
7 APPLICATION CIRCUIT

7.1 Power Supply Scheme



Note: C_{LDO} must be connected between the LDO_CAP pin and the closest GND pin of the device. If there are two LDO_CAP pins or more, the capacitor value on each LDO pin needs to be divided equally.

7.2 Peripheral Application Scheme



Note 1: It is recommended to use 100 kΩ pull-up resistor on both ICE_DAT and ICE_CLK pin.

Note 2: It is recommended to use 10 kΩ pull-up resistor and 10 uF capacitor on nRESET pin.

Note 3: C_{LDO} must be connected between the LDO_CAP pin and the closest GND pin of the device. If there are two LDO_CAP pins or more, the capacitor value on each LDO pin needs to be divided equally.

8 ELECTRICAL CHARACTERISTIC

SPlease refer to the relative Datasheet for detailed information about the M2L31 electrical characteristics.

9 ABBREVIATIONS

9.1 Abbreviations

Acronym	Description
ACMP	Analog Comparator Controller
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
BOD	Brown-out Detection
CAN	Controller Area Network
CCAP	Camera Capture Interface
DAP	Debug Access Port
DES	Data Encryption Standard
EADC	Enhanced Analog-to-Digital Converter
EBI	External Bus Interface
EMAC	Ethernet MAC Controller
EPWM	Enhanced Pulse Width Modulation
FIFO	First In, First Out
FMC	Flash Memory Controller
FPU	Floating-point Unit
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
HIRC	12 MHz Internal High Speed RC Oscillator
HXT	4~24 MHz External High Speed Crystal Oscillator
IAP	In Application Programming
ICP	In Circuit Programming
ISP	In System Programming
LDO	Low Dropout Regulator
LIN	Local Interconnect Network
LIRC	10 kHz internal low speed RC oscillator (LIRC)
MPU	Memory Protection Unit
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PDMA	Peripheral Direct Memory Access
PLL	Phase-Locked Loop

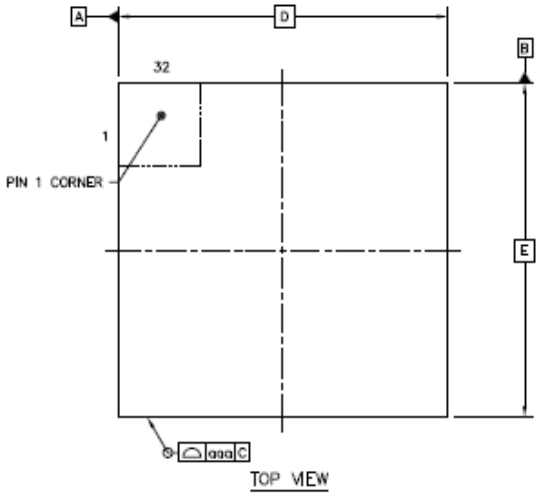
PWM	Pulse Width Modulation
EQEI	Enhanced Quadrature Encoder Interface
SD	Secure Digital
SPI	Serial Peripheral Interface
SPS	Samples per Second
TDES	Triple Data Encryption Standard
TK	Touch Key
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
UCID	Unique Customer ID
USB	Universal Serial Bus
WDT	Watchdog Timer
WWDT	Window Watchdog Timer

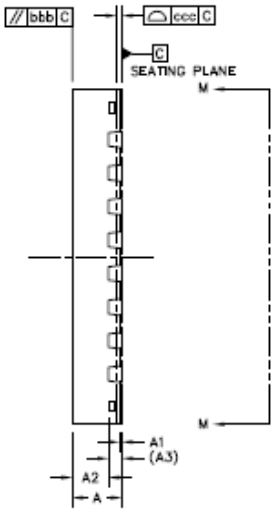
Table 9.1-1 List of Abbreviations

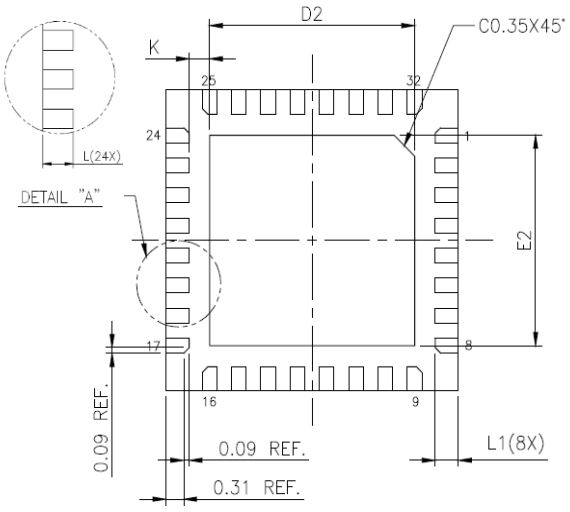
10 PACKAGE DIMENSIONS

Package is Halogen-free, RoHS-compliant and TSCA-compliant.

10.1 QFN 33L (5x5x0.8 mm, Pitch 0.5 mm, Lead Length 0.4 mm)





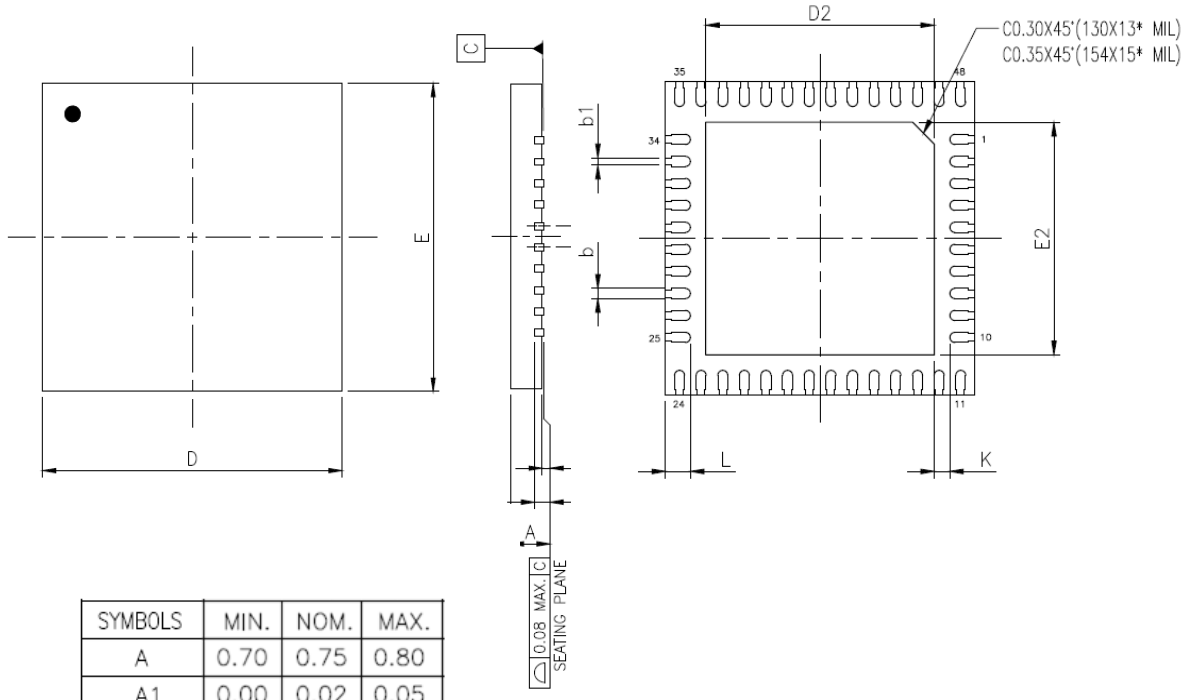


mm

SYMBOLS	Min	Nom	Max
A	0.7	0.75	0.8
A1	0	0.02	0.05
A3		0.2 REF	
b	0.18	0.25	0.3
D		5.0 BSC	
E		5.0 BSC	
e		0.5 BSC	
D2	3.45	3.5	3.55
E2	3.45	3.5	3.55
L	0.35	0.4	0.45
L1	0.33	0.4	0.43
K	0.2	-	-

QFN32_5x5_0.80_0.50_0.40_3.50x3.50

10.2 QFN 48L (5x5x0.8 mm, Pitch 0.35 mm, Lead Length 0.4 mm)

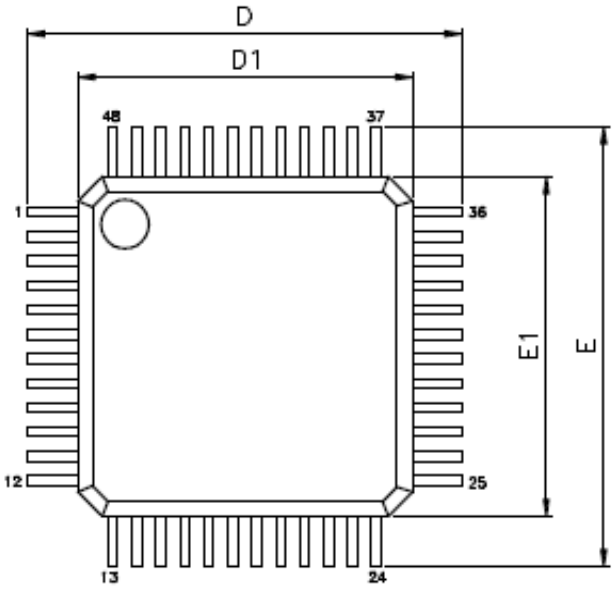


SYMBOLS	MIN.	NOM.	MAX.
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A3		0.203 REF.	
b	0.13	0.18	0.23
b1	0.07	0.12	0.17
D		5.00 BSC	
E		5.00 BSC	
e		0.35 BSC	
K	0.20	—	—

PAD SIZE	D2			E2			L		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
130X13* MIL	3.65	3.70	3.75	3.65	3.70	3.75	0.35	0.40	0.45
154X15* MIL	3.65	3.70	3.75	3.65	3.70	3.75	0.35	0.40	0.45

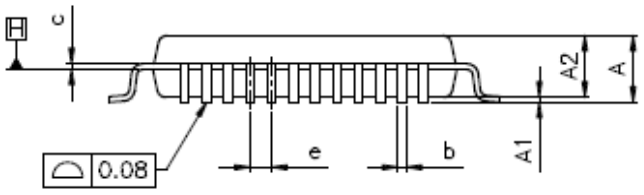
QFN48_5x5_0.80_0.35_0.40_3.70x3.70

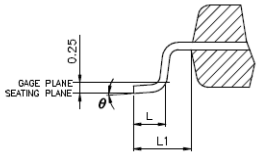
10.3 LQFP 48L (7x7x1.4 mm Footprint 2.0 mm)



VARIATIONS (ALL DIMENSIONS SHOWN IN MM)

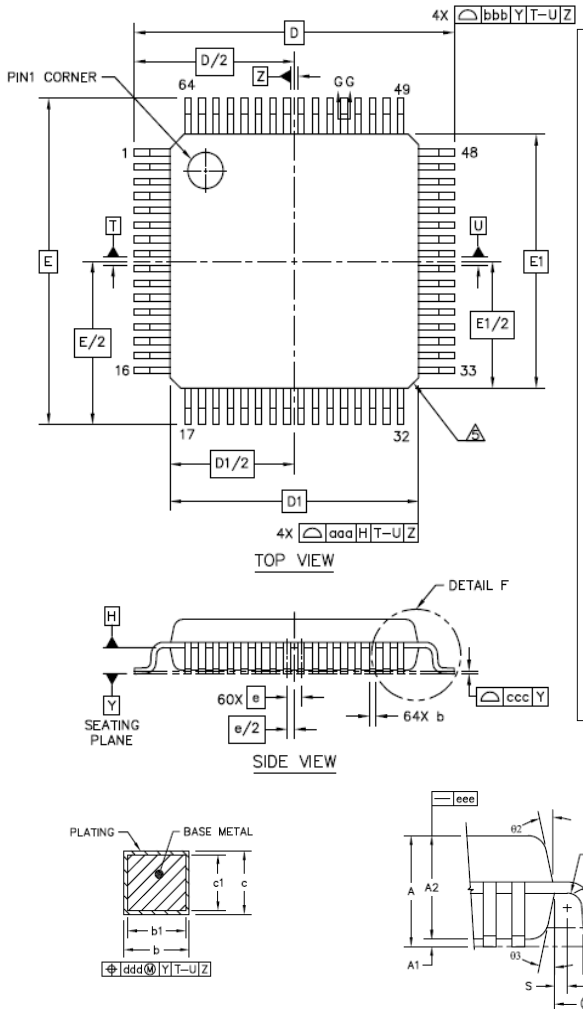
SYMBOLS	MIN.	NOM.	MAX.
A	--	--	1.60
A1	0.05	--	0.15
A2	1.35	1.40	1.45
b	0.17	0.22	0.27
c	0.09	--	0.20
D	8.80	9.00	9.20
D1	6.90	7.00	7.10
E	8.80	9.00	9.20
E1	6.90	7.00	7.10
e	0.50 BSC		
L	0.45	0.60	0.75
L1	1.00 REF		
S	0.20	--	--
θ	0°	3.5°	7°
θ1	0°	--	--
θ2	11°	12°	13°
θ3	11°	12°	13°





LQFP48_7X7_1.40_0.50_1.00

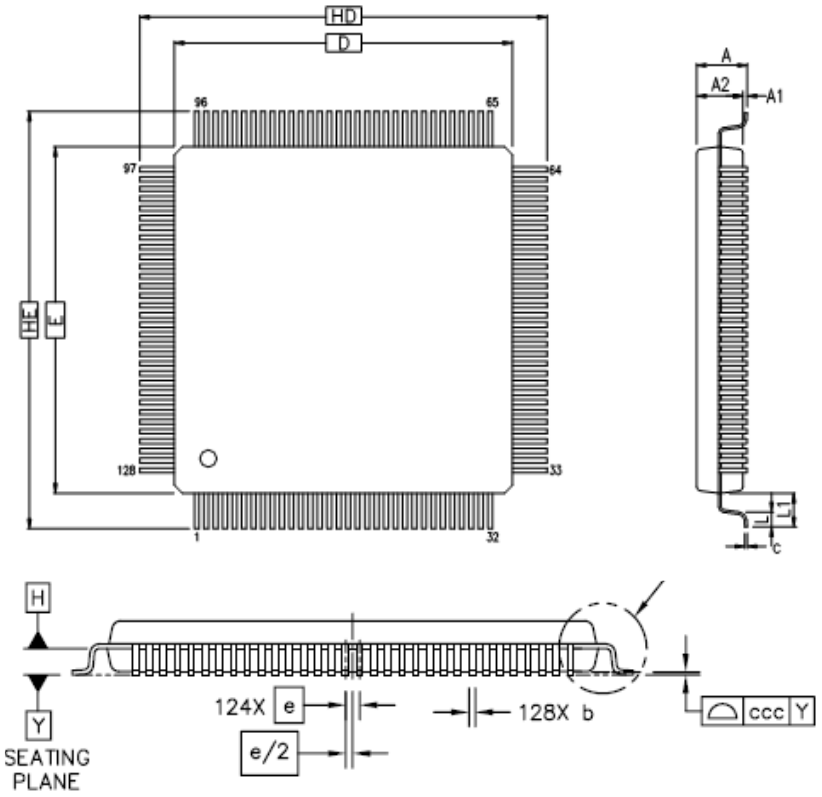
10.4 LQFP 64L (7x7x1.4 mm Footprint 2.0 mm)



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	---	---	1.6
STAND OFF		A1	0.05	---	0.15
MOLD THICKNESS		A2	1.35	1.4	1.45
LEAD WIDTH(PLATING)		b	0.13	0.18	0.23
LEAD WIDTH		b1	0.13	0.16	0.19
L/F THICKNESS(PLATING)		c	0.09	---	0.2
L/F THICKNESS		c1	0.09	---	0.16
BODY SIZE	X	D	8.8	9	9.2
	Y	E	8.8	9	9.2
	X	D1	6.8	7	7.2
	Y	E1	6.8	7	7.2
LEAD PITCH		e	0.4 BSC		
		L	0.45	0.6	0.75
FOOTPRINT		L1	1 REF		
		θ	0°	3.5°	7°
		θ1	0°	---	---
		θ2	11°	12°	13°
		θ3	11°	12°	13°
		R1	0.08	---	---
		R2	0.08	---	0.2
		S	0.2	---	---
COPLANARITY		ccc	0.08		
LEAD OFFSET		ddd	0.07		
MOLD FLATNESS		eee	0.05		

LQFP64_7X7_1.40_0.40_1.00

10.5 LQFP 128L (14x14x1.4 mm Footprint 2.0 mm)



SIDE VIEW

SYMBOL	MIN	NOM.	MAX.
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
HD	15.85	16.00	16.15
D	13.85	14.00	14.15
HE	15.85	16.00	16.15
E	13.85	14.00	14.15
b	0.13	0.16	0.23
e	-	0.40	-
Θ°	0	3.5	7
c	0.09	-	0.20
L	0.45	0.60	0.75
L1	-	1.00	-
Y	-	-	0.10

LQFP128_14X14_1.40_0.40_1.00

11 REVISION HISTORY

Date	Revision	Description
2024.06.12	1.00	Initial version.
2024.09.03	1.01	1. Revised QFN33 multi-function diagram and table in section 4.3.1. 2. Revised INT_VREF block diagram to show interconnection between external V_{REF} pin and INT_VREF in Figure 6.42-1, Figure 6.43-1 and Figure 6.44-1. 3. Added PROEN0 (SYS_UTCPDCTL[1]) bit information to turn off Rd of UTCPD_CC1/UTCPD_CC2 pin in section 6.6.5.1. 4. Added a note to inform LPADC usage in VBGFEN bit field and updated content in VREFCTL bit field in section 6.2.18. 5. Added RMC_APWPROT0, RMC_APWPROT1 and RMC_APWPKEEP in Table 6.2-1. 6. Modified the description of register DAC0_DAT and DAC1_DAT in section 6.44.7. 7. Updated DAC block diagram in Figure 6.44-1. 8. Updated OPA block diagram in Figure 6.45-1. 9. Revised clock status description of NPD5, SPD and DPD mode in Table 6.2-2.
2024.12.17	1.02	1. Revised the max. PWM clock source from 144 MHz to 72 MHz in section 2.1 and chaptor 1. 2. Removed UTCPD0_CCDB1 and UTCPD0_CCDB2 in section 4.3.1. 3. Removed SYS_RAMPGCTL register in section 6.2.17 and 6.2.18. 4. Removed LPM function in section 6.32.
2025.05.12	1.03	1. Revised RMC_APPROT register to RMC_APWROT register in section 6.5.4.6. 2. Removed unnecessary text in section 6.5.4.7. 3. Revised TTMR0_TRGCTL register bit field in section 6.13.7.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.