NuMicro® Family

Arm® Cortex®-M0-based Microcontroller

M0A21/M0A23 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 17
2 FEATURES 18
3 PARTS INFORMATION 26
3.1 Package Type 26
3.2 NuMicro® M0A21/M0A23 Series Selection Guide 27
3.2.1 NuMicro® M0A21 Series 27
3.2.2 NuMicro® M0A23 Series 27
3.2.3 NuMicro® M0A21/M0A23 Selection Code 28
4 PIN CONFIGURATION 29
4.1 Pin Configuration 29
4.1.1 M0A21 Series Pin Diagram 29
4.1.2 M0A21 Series Function Pin Table 30
4.1.3 M0A23 Series Pin Diagram 35
4.1.4 M0A23 Series Function Pin Table 37
4.2 Pin Description 41
4.2.1 M0A21/M0A23 Series Pin Description 41
4.2.2 M0A21/M0A23 Series Multi-function Summary Table 59
4.2.3 M0A21/M0A23 Series Multi-function Summary Table Sorted by GPIO 77
5 BLOCK DIAGRAM 95
6 FUNCTIONAL DESCRIPTION 96
6.1 Arm® Cortex®-M0 Core 96
6.2 System Manager 98
6.2.1 Overview 98
6.2.2 System Reset 98
6.2.3 System Power Distribution 104
6.2.4 Power Modes and Wake-up Sources 104
6.2.5 System Memory Map 108
6.2.6 SRAM Memory Orginization 110
6.2.7 Chip Bus Matrix 111
6.2.8 IRC Auto Trim 111
6.2.9 Register Lock Control 112
6.2.10 UART0_TXD/USCI0_DAT0 Modulation with PWM 112
6.2.11 Register Map 113
6.2.12 Register Description 115
6.2.13 System Timer (SysTick) 148
6.2.14 Nested Vectored Interrupt Controller (NVIC) 153
6.2.15 System Control Register 169
6.3 Clock Controller 177

6.3.1 Overview 177
6.3.2 Clock Generator 178
6.3.3 System Clock and SysTick Clock 180
6.3.4 Peripherals Clock 181
6.3.5 Power-down Mode Clock 181
6.3.6 Clock Output 182
6.3.7 Register Map 183
6.3.8 Register Description 184
6.4 Flash Memory Controller (FMC) 203
6.4.1 Overview 203
6.4.2 Features 203
6.4.3 Block Diagram 203
6.4.4 Functional Description 205
6.4.5 Register Map 222
6.4.6 Register Description 223
6.5 General Purpose I/O (GPIO) 232
6.5.1 Overview 232
6.5.2 Features 232
6.5.3 Block Diagram 233
6.5.4 Basic Configuration 233
6.5.5 Functional Description 233
6.5.6 Register Map 238
6.5.7 Register Description 240
6.6 PDMA Controller (PDMA) 254
6.6.1 Overview 254
6.6.2 Features 254
6.6.3 Block Diagram 254
6.6.4 Basic Configuration 254
6.6.5 Functional Description 255
6.6.6 Register Map 261
6.6.7 Register Description 262
6.7 Timer Controller (TMR) 289
6.7.1 Overview 289
6.7.2 Features 289
6.7.4 Block Diagram 290
6.7.5 Basic Configuration 291
6.7.6 Functional Description 292
6.7.7 Register Map 299
6.7.8 Register Description 301
6.8 Watchdog Timer (WDT) 312
6.8.1 Overview 312

6.8.2 Features 312
6.8.3 Block Diagram 312
6.8.4 Basic Configuration 312
6.8.5 Functional Description 313
6.8.6 Register Map 316
6.8.7 Register Description 317
6.9 Window Watchdog Timer (WWDT) 321
6.9.1 Overview 321
6.9.2 Features 321
6.9.3 Block Diagram 321
6.9.4 Basic Configuration 321
6.9.5 Functional Description 322
6.9.6 Register Map 326
6.9.7 Register Description 327
6.10 PWM Generator and Capture Timer (PWM) 332
6.10.1 Overview 332
6.10.2 Features 332
6.10.3 Block Diagram 333
6.10.4 Basic Configuration 336
6.10.5 Functional Description 338
6.10.6 Register Map 360
6.10.7 Register Description 363
6.11 UART Interface Controller (UART) 411
6.11.1 Overview 411
6.11.2 Features 411
6.11.3 Block Diagram 412
6.11.4 Basic Configuration 415
6.11.5 Functional Description 416
6.11.6 Register Map 441
6.11.7 Register Description 443
6.12 USCI - Universal Serial Control Interface Controller (USCI) 480
6.12.1 Overview 480
6.12.2 Features 480
6.12.3 Block Diagram 480
6.12.4 Functional Description 480
6.13 USCI – UART Mode 491
6.13.1 Overview 491
6.13.2 Features 491
6.13.3 Block Diagram 491
6.13.4 Basic Configuration 491
6.13.5 Functional Description 493

6.13.6 Register Map 502
6.13.7 Register Description 503
6.14 USCI - SPI Mode 524
6.14.1 Overview 524
6.14.2 Features 524
6.14.3 Block Diagram 525
6.14.4 Basic Configuration 525
6.14.5 Functional Description 526
6.14.6 Register Map 538
6.14.7 Register Description 540
6.15 USCI - I²C Mode 562
6.15.1 Overview 562
6.15.2 Features 562
6.15.3 Block Diagram 563
6.15.4 Basic Configuration 563
6.15.5 START or Repeated START Signal 564
6.15.6 Register Map 582
6.15.7 Register Description 583
6.16 Controller Area Network (CAN) 602
6.16.1 Overview 602
6.16.2 Features 602
6.16.3 Block Diagram 602
6.16.4 Basic Configuration 603
6.16.5 Functional Description 604
6.16.6 Test Mode 605
6.16.7 CAN Communications 607
6.16.8 Register Map 623
6.16.9 Register Description 628
6.17 CRC Controller (CRC) 662
6.17.1 Overview 662
6.17.2 Features 662
6.17.3 Block Diagram 662
6.17.4 Basic Configuration 662
6.17.5 Functional Description 663
6.17.6 Register Map 665
6.17.7 Register Description 666
6.18 Hardware Divider (HDIV) 671
6.18.1 Overview 671
6.18.2 Features 671
6.18.3 Basic Configuration 671
6.18.4 Functional Description 671

6.18.5 Register Map 672
6.18.6 Register Description 673
6.19 Analog-to-Digital Converter (ADC) 678
6.19.1 Overview 678
6.19.2 Features 678
6.19.3 Block Diagram 679
6.19.4 Basic Configuration 679
6.19.5 Functional Description 679
6.19.6 Register Map 687
6.19.7 Register Description 689
6.20 Digital to Analog Converter (DAC) 703
6.20.1 Overview 703
6.20.2 Features 703
6.20.3 Block Diagram 703
6.20.4 Basic Configuration 704
6.20.5 Functional Description 704
6.20.6 Register Map 708
6.20.7 Register Description 709
6.21 Analog Comparator Controller (ACMP) 716
6.21.1 Overview 716
6.21.2 Features 716
6.21.3 Block Diagram 716
6.21.4 Basic Configuration 717
6.21.5 Functional Description 718
6.21.6 Register Map 723
6.21.7 Register Description 724
6.22 Peripherals Interconnection 731
6.22.1 Overview 731
6.22.2 Peripherals Interconnect Matrix Table 731
6.22.3 Functional Description 731
7 APPLICATION CIRCUIT 733
7.1 Power Supply Scheme 733
7.2 Peripheral Application Scheme 734
8 ELECTRICAL CHARACTERISTICS 735
9 PACKAGE DIMENSIONS 736
9.1 SSOP 20 (5.3x7.2x1.75 mm) 736
9.2 TSSOP 28 (4.4x9.7x1.0 mm) 737
10 ABBREVIATIONS 738
10.1 Abbreviations 738

11 REVISION HISTORY 740

LIST OF FIGURES

Figure 4.1-1 M0A21 Series SSOP 20-pin Diagram 29
Figure 4.1-2 M0A21 Series TSSOP 28-pin Diagram 30
Figure 4.1-3 M0A23 Series SSOP 20-pin Diagram 35
Figure 4.1-4 M0A23 Series TSSOP 28-pin Diagram 36
Figure 4.2-1 NuMicro® M0A21/M0A23 Block Diagram 95
Figure 6.1-1 Functional Block Diagram 96
Figure 6.2-1 System Reset Sources 99
Figure 6.2-2 nRESET Reset Waveform 101
Figure 6.2-3 nRESET Reset Mode Enable Control Waveform 101
Figure 6.2-4 Power-on Reset (POR) Waveform 102
Figure 6.2-5 Low Voltage Reset (LVR) Waveform 102
Figure 6.2-6 Brown-out Detector (BOD) Waveform 103
Figure 6.2-7 NuMicro® M0A21/M0A23 Power Distribution Diagram 104
Figure 6.2-8 Power Mode State Machine 106
Figure 6.2-9 SRAM Memory Organization 110
Figure 6.2-10 NuMicro® M0A21/M0A23 Bus Matrix Diagram 111
Figure 6.3-1 Clock Generator Global View Diagram 178
Figure 6.3-2 Clock Generator Block Diagram 179
Figure 6.3-3 System Clock Block Diagram 180
Figure 6.3-4 HXT Stop Protect Procedure 181
Figure 6.3-5 SysTick Clock Control Block Diagram 181
Figure 6.3-6 Clock Output Block Diagram 182
Figure 6.4-116/32 KB Flash Memory Control Block Diagram 204
Figure 6.4-2 Data Flash Shared with APROM 205
Figure 6.4-3 16/32 Kbytes Flash Memory Map 211
Figure 6.4-4 16/32 Kbytes Flash System Memory Map with IAP Mode 212
Figure 6.4-5 LDROM with IAP Mode 213
Figure 6.4-6 APROM with IAP Mode 213
Figure 6.4-7 16/32 Kbytes Flash System Memory Map without IAP Mode 214
Figure 6.4-8 Boot Source Selection 215
Figure 6.4-9 ISP Procedure Example 217
Figure 6.4-10 Example for Accelerating Interrupt by VECMAP 218
Figure 6.4-11 ISP 32-bit Programming Procedure 219
Figure 6.4-12 CRC-32 Checksum Calculation 220
Figure 6.4-13 CRC-32 Checksum Calculation Flow 221
Figure 6.5-1 GPIO Controller Block Diagram 233

Figure 6.5-2 Input Mode 234
Figure 6.5-3 Push-Pull Output 234
Figure 6.5-4 Open-Drain Output 235
Figure 6.5-5 Quasi-Bidirectional I/O Mode 235
Figure 6.5-6 GPIO Rising Edge Trigger Interrupt 236
Figure 6.5-7 GPIO Falling Edge Trigger Interrupt 237
Figure 6.6-1 PDMA Controller Block Diagram 254
Figure 6.6-2 Descriptor Table Entry Structure 255
Figure 6.6-3 Basic Mode Finite State Machine 256
Figure 6.6-4 Descriptor Table Link List Structure 257
Figure 6.6-5 Scatter-Gather Mode Finite State Machine 258
Figure 6.6-6 Example of Single Transfer Type and Burst Transfer Type in Basic Mode 259
Figure 6.6-7 Example of PDMA Channel 0 Time-out Counter Operation 260
Figure 6.7-1 Timer Controller Block Diagram 290
Figure 6.7-2 Clock Source of Timer Controller 291
Figure 6.7-3 Continuous Counting Mode 293
Figure 6.7-4 External Capture Mode 294
Figure 6.7-5 Reset Counter Mode 295
Figure 6.7-6 Internal Timer Trigger 296
Figure 6.7-7 Inter-Timer Trigger Capture Timing 297
Figure 6.8-1 Watchdog Timer Block Diagram 312
Figure 6.8-2 Watchdog Timer Clock Control 313
Figure 6.8-3 Watchdog Timer Time-out Interval and Reset Period Timing 314
Figure 6.9-1 WWDT Block Diagram 321
Figure 6.9-2 WWDT Clock Control 321
Figure 6.9-3 WWDT Reset and Reload Behavior 323
Figure 6.9-4 WWDT Reload Counter When CNTDAT > CMPDAT 323
Figure 6.9-5 WWDT Reload Counter When WWDT_CNT < WINCMP 324
Figure 6.9-6 WWDT Interrupt and Reset Signals 324
Figure 6.10-1 PWM Generator Overview Block Diagram 333
Figure 6.10-2 PWM System Clock Source Control 334
Figure 6.10-3 PWM Clock Source Control 335
Figure 6.10-4 PWM Independent Mode Architecture Diagram 335
Figure 6.10-5 PWM Complementary Mode Architecture Diagram 336
Figure 6.10-6 PWM0_CH0 Prescaler Waveform in Up Counter Type 339
Figure 6.10-7 PWMx Counter Waveform When Setting Clear Counter 339
Figure 6.10-8 PWM Up Counter Type 340

Figure 6.10-9 PWM Down Counter Type 340
Figure 6.10-10 PWM Up-Down Counter Type 341
Figure 6.10-11 PWM Compared point Events in Up-Down Counter Type 342
Figure 6.10-12 PWM Double Buffering Illustration 343
Figure 6.10-13 Period Loading in Up-Count Mode 343
Figure 6.10-14 Immediately Loading in Up-Count Mode 344
Figure 6.10-15 Center Loading in Up-Down-Count Mode 345
Figure 6.10-16 PWM Pulse Generation 346
Figure 6.10-17 PWM 0% to 100% Pulse Generation 346
Figure 6.10-18 PWM Independent Mode Waveform 347
Figure 6.10-19 PWM Complementary Mode Waveform 348
Figure 6.10-20 PWMx_CH0 Output Control in Independent Mode 348
Figure 6.10-21 PWMx_CH0 and PWMx_CH1 Output Control in Complementary Mode 349
Figure 6.10-22 Dead-Time Insertion 349
Figure 6.10-23 Illustration of Mask Control Waveform 350
Figure 6.10-24 Brake Noise Filter Block Diagram 350
Figure 6.10-25 Brake Block Diagram for PWMx_CH0 and PWMx_CH1 Pair 351
Figure 6.10-26 Edge Detector Waveform for PWMx_CH0 and PWMx_CH1 Pair 352
Figure 6.10-27 Level Detector Waveform for PWMx_CH0 and PWMx_CH1 Pair 352
Figure 6.10-28 Brake Source Block Diagram 353
Figure 6.10-29 Brake System Fail Block Diagram 353
Figure 6.10-30 Initial State and Polarity Control with Rising Edge Dead-Time Insertion 354
Figure 6.10-31 PWM_CH0 and PWM_CH1 Pair Interrupt Architecture Diagram 355
Figure 6.10-32 PWMx_CH0 and PWMx_CH1 Pair Trigger ADC Block Diagram 356
Figure 6.10-33 PWM Trigger ADC in Up-Down Counter Type Timing Waveform 356
Figure 6.10-34 PWM_CH0 Capture Block Diagram 357
Figure 6.10-35 Capture Operation Waveform 358
Figure 6.10-36 Capture PDMA Operation Waveform of Channel 0 359
Figure 6.11-1 UART Clock Control Diagram 413
Figure 6.11-2 UART Block Diagram 414
Figure 6.11-3 Auto-Baud Rate Measurement 420
Figure 6.11-4 Transmit Delay Time Operation 420
Figure 6.11-5 UART nCTS Wake-up Case1 421
Figure 6.11-6 UART nCTS Wake-up Case2 421
Figure 6.11-7 UART Data Wake-up 422
Figure 6.11-8 UART Received Data FIFO Reached Threshold Wake-up 422
Figure 6.11-9 UART RS-485 AAD Mode Address Match Wake-up 423

Figure 6.11-10 UART Received Data FIFO threshold time-out wake-up 423
Figure 6.11-11 Auto-Flow Control Block Diagram 427
Figure 6.11-12 UART nCTS Auto-Flow Control Enabled 427
Figure 6.11-13 UART nRTS Auto-Flow Control Enabled 428
Figure 6.11-14 UART nRTS Auto-Flow with Software Control 428
Figure 6.11-15 IrDA Control Block Diagram 429
Figure 6.11-16 IrDA TX/RX Timing Diagram 430
Figure 6.11-17 Structure of LIN Frame 430
Figure 6.11-18 Structure of LIN Byte 431
Figure 6.11-19 Break Detection in LIN Mode 433
Figure 6.11-20 LIN Frame ID and Parity Format 433
Figure 6.11-21 LIN Sync Field Measurement 435
Figure 6.11-22 UART_BAUD Update Sequence in AR mode if SLVDUEN is 1 436
Figure 6.11-23 UART_BAUD Update Sequence in AR mode if SLVDUEN is 0 436
Figure 6.11-24 RS-485 nRTS Driving Level in Auto Direction Mode 438
Figure 6.11-25 RS-485 nRTS Driving Level with Software Control 439
Figure 6.11-26 Structure of RS-485 Frame 439
Figure 6.12-1 USCI Block Diagram 480
Figure 6.12-2 Input Conditioning for USCIx_DAT[1:0] and USCIx_CTL[1:0] 481
Figure 6.12-3 Input Conditioning for USCIx_CLK 482
Figure 6.12-4 Block Diagram of Data Buffering 483
Figure 6.12-5 Data Access Structure 484
Figure 6.12-6 Transmit Data Path 484
Figure 6.12-7 Receive Data Path 485
Figure 6.12-8 Protocol-Relative Clock Generator 486
Figure 6.12-9 Basic Clock Divider Counter 487
Figure 6.12-10 Block of Timing Measurement Counter 487
Figure 6.12-11 Sample Time Counter 488
Figure 6.12-12 Event and Interrupt Structure 489
Figure 6.13-1 USCI-UART Mode Block Diagram 491
Figure 6.13-2 UART Signal Connection for Full-Duplex Communication 493
Figure 6.13-3 UART Standard Frame Format 494
Figure 6.13-4 UART Bit Timing (data sample time) 496
Figure 6.13-5 UART Auto Baud Rate Control 498
Figure 6.13-6 Incoming Data Wake-Up 499
Figure 6.13-7 nCTS Wake-Up Case 1 499
Figure 6.13-8 nCTS Wake-Up Case 2 499

Figure 6.14-1 SPI Master Mode Application Block Diagram 524
Figure 6.14-2 SPI Slave Mode Application Block Diagram 524
Figure 6.14-3 USCI SPI Mode Block Diagram 525
Figure 6.14-44-Wire Full-Duplex SPI Communication Signals (Master Mode) 527
Figure 6.14-54-Wire Full-Duplex SPI Communication Signals (Slave Mode) 527
Figure 6.14-6 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x0) .. 528
Figure 6.14-7 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x1) .. 529
Figure 6.14-8 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x2) .. 529
Figure 6.14-9 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x3) .. 530
Figure 6.14-1016-bit Data Length in One Word Transaction with MSB First Format 531
Figure 6.14-11 Word Suspend Interval between Two Transaction Words 531
Figure 6.14-12 Auto Slave Select (SUSPITV ≧ 0x3) 532
Figure 6.14-13 Auto Slave Select (SUSPITV < 0x3) 532
Figure 6.14-14 One Output Data Channel Half-duplex (SPI Master Mode) 533
Figure 6.14-15 One Input Data Channel Half-duplex (SPI Master Mode) 534
Figure 6.14-16 SPI Timing in Master Mode 536
Figure 6.14-17 SPI Timing in Master Mode (Alternate Phase of Serial Bus Clock) 536
Figure 6.14-18 SPI Timing in Slave Mode 537
Figure 6.14-19 SPI Timing in Slave Mode (Alternate Phase of Serial Bus Clock) 537
Figure 6.15-1 I²C Bus Timing 562
Figure 6.15-2 USCI I²C Mode Block Diagram 563
Figure 6.15-3 I²C Protocol 564
Figure 6.15-4 START and STOP Conditions 565
Figure 6.15-5 Bit Transfer on the I²C Bus 566
Figure 6.15-6 Acknowledge on the I²C Bus 566
Figure 6.15-7 Arbitration Lost 567
Figure 6.15-8 Control I²C Bus According to Current I²C Status 570
Figure 6.15-9 Master Transmits Data to Slave with a 7-bit Address 570
Figure 6.15-10 Master Reads Data from Slave with a 7-bit Address 570
Figure 6.15-11 Master Transmits Data to Slave by 10-bit Address 570
Figure 6.15-12 Master Reads Data from Slave by 10-bit Address 571
Figure 6.15-13 Master Transmitter Mode Control Flow with 7-bit Address 571
Figure 6.15-14 Master Receiver Mode Control Flow with 7-bit Address 572
Figure 6.15-15 Master Transmitter Mode Control Flow with 10-bit Address 573
Figure 6.15-16 Master Recevier Mode Control Flow with 10-bit Address 574
Figure 6.15-17 Save Mode Control Flow with 7-bit Address 575
Figure 6.15-18 Save Mode Control Flow with 10-bit Address 576

Figure 6.15-19 GC Mode with 7-bit Address 577
Figure 6.15-20 Setup Time Wrong Adjustment 579
Figure 6.15-21 Hold Time Wrong Adjustment 579
Figure 6.15-22 I²C Time-out Count Block Diagram 580
Figure 6.15-23 EEPROM Random Read 581
Figure 6.15-24 Protocol of EEPROM Random Read 581
Figure 6.16-1 CAN Peripheral Block Diagram 603
Figure 6.16-2 CAN Core in Silent Mode 605
Figure 6.16-3 CAN Core in Loop Back Mode 606
Figure 6.16-4 CAN Core in Loop Back Mode Combined with Silent Mode 606
Figure 6.16-5 Data Transfer between IFn Registers and Message 608
Figure 6.16-6 Application Software Handling of a FIFO Buffer 613
Figure 6.16-7 Bit Timing 615
Figure 6.16-8 Propagation Time Segment 616
Figure 6.16-9 Synchronization on “late” and “early” Edges 617
Figure 6.16-10 Filtering of Short Dominant Spikes 618
Figure 6.16-11 Structure of the CAN Core’s CAN Protocol Controller 619
Figure 6.17-1 CRC Generator Block Diagram 662
Figure 6.17-2 CHECKSUM Bit Order Reverse Functional Block 663
Figure 6.17-3 Write Data Bit Order Reverse Functional Block 664
Figure 6.18-1 Hardware Divider Operation Flow 671
Figure 6.19-1 AD Controller Block Diagram 679
Figure 6.19-2 ADC Peripheral Clock Control 680
Figure 6.19-3 Single Mode Conversion Timing Diagram 681
Figure 6.19-4 Burst Mode Conversion Timing Diagram 682
Figure 6.19-5 Single-Cycle Scan Mode on Enabled Channels Timing Diagram 683
Figure 6.19-6 Continuous Scan Mode on Enabled Channels Timing Diagram 684
Figure 6.19-7 A/D Conversion Result Monitor Logic Diagram 685
Figure 6.19-8 A/D Controller Interrupt 686
Figure 6.19-9 Conversion Result Mapping Diagram of ADC Single-end Input 690
Figure 6.19-10 Conversion Result Mapping Diagram of ADC Differential Input 691
Figure 6.20-1 Digital-to-Analog Converter Block Diagram 703
Figure 6.20-2 DAC Conversion Started by Software Write Trigger 704
Figure 6.20-3 DAC Conversion Started by Hardware Trigger Event 705
Figure 6.20-4 DAC PDMA Underrun Condition Example 706
Figure 6.20-5 DAC Continuous Conversion with Software PDMA Mode 706
Figure 6.20-6 DAC Interrupt Source 707

Figure 6.21-1 Analog Comparator Block Diagram 717
Figure 6.21-2 Comparator Hysteresis Function of ACMP0 718
Figure 6.21-3 Window Latch Mode 719
Figure 6.21-4 Filter Function Example 719
Figure 6.21-5 Comparator Controller Interrupt 720
Figure 6.21-6 Comparator Reference Voltage Block Diagram 720
Figure 6.21-7 Example of Window Compare Mode 721
Figure 6.21-8 Example of Window Compare Mode 722

List of Tables

Table 1-1 NuMicro® M0A21/M0A23 Series Key Features Table 17
Table 6.2-1 Reset Value of Registers 100
Table 6.2-2 Power Mode Table 105
Table 6.2-3 Power Mode Difference Table 105
Table 6.2-4 Power Mode Difference Table 105
Table 6.2-5 Clocks in Power Modes 107
Table 6.2-6 Condition of Entering Power-down Mode Again 108
Table 6.2-7 Address Space Assignments for On-Chip Controllers 109
Table 6.2-8 Exception Model 154
Table 6.2-9 Interrupt Number Table 155
Table 6.2-10 Priority Grouping 173
Table 6.4-1 Vector Mapping Support 215
Table 6.4-2 ISP Command List 216
Table 6.4-3 FMC Control Registers for Flash Programming 218
Table 6.6-1 Channel Priority Table 256
Table 6.7-1 Timer0 ~ Timer3 MFP Table 292
Table 6.8-1 Watchdog Timer Time-out Interval Period Selection 314
Table 6.9-1 WWDT Prescaler Value Selection 322
Table 6.9-2 CMPDAT Setting Limitation 325
Table 6.10-1 PWM Clock Source Control Registers Setting Table 334
Table 6.10-2 PWM Pulse Generation Event Priority for Up-Counter 346
Table 6.10-3 PWM Pulse Generation Event Priority for Down-Counter 347
Table 6.10-4 PWM Pulse Generation Event Priority for Up-Down-Counter 347
Table 6.11-1 NuMicro® M0A21/M0A23 Series UART Features 412
Table 6.11-2 UART Interrupt 415
Table 6.11-3 UART Interface Controller Pin 416
Table 6.11-4 UART Controller Baud Rate Equation Table 417
Table 6.11-5 UART Controller Baud Rate Parameter Setting Example Table 417
Table 6.11-6 UART Controller Baud Rate Register Setting Example Table 418
Table 6.11-7 Baud Rate Compensation Example Table 1 418
Table 6.11-8 Baud Rate Compensation Example Table 2 419
Table 6.11-9 UART controller Interrupt Source and Flag List 425
Table 6.11-10 UART Line Control of Word and Stop Length Setting 426
Table 6.11-11 UART Line Control of Parity Bit Setting 426
Table 6.11-12 LIN Header Selection in Master Mode 431
Table 6.12-1 Input Signals for Different Protocols 481

Table 6.12-2 Output Signals for Different Protocols 482
Table 6.12-3 Data Transfer Events and Interrupt Handling 489
Table 6.12-4 Protocol-specific Events and Interrupt Handling 489
Table 6.13-1 Input Signals for UART Protocol 494
Table 6.13-2 Output Signals for Different Protocol 494
Table 6.14-1 Serial Bus Clock Configuration 528
Table 6.15-1 Relationship between I^2C Baud Rate and PCLK 579
Table 6.16-1 Initialization of a Transmit Object 610
Table 6.16-2 Initialization of a Receive Object 611
Table 6.16-3 CAN Bit Time Parameters 615
Table 6.16-4 CAN Register Map for Each Bit Function 627
Table 6.16-5 Last Error Code 631
Table 6.16-6 Source of Interrupts 634
Table 6.16-7 IF1 and IF2 Message Interface Register 637
Table 6.16-8 Structure of a Message Object in the Message Memory 651
Table 6.21-1 Truth Table of Window Compare Logic 721
Table 6.22-1 Peripherals Interconnect Matrix Table 731
Table 10.1-1 List of Abbreviations 739

1 GENERAL DESCRIPTION

The NuMicro® M0A21/M0A23 series is a 32-bit microcontroller based on Arm® Cortex®-M0 core. It provides compact package with highly flexible digital pin function assignment, rich analog peripherals, -40°C to 125°C operating temperature, 2.4V ~ 5.5V operating voltage, CAN 2.0B and LIN interface for robust communication. The NuMicro® M0A23/M0A21 series targets robust and high operating temperature applications, such as 24 GHz mmWave radar, Battery Management System (BMS), car lighting, electric window lifter, and power seat.

The NuMicro® M0A21/M0A23 series provide SSOP20 and TSSOP28 package with rich analog and digital functions, which are especially suitable for small form factor applications. SSOP20 provides up to 18 IO pins and TSSOP28 provides up to 26 IO pins. Each IO pin of the M0A21/M0A23 series can be arbitrarily assigned to digital peripherals, such as UART, SPI, and PWM. The M0A21/M0A23 series provides rich analog functions including 17-ch 12-bit 800 kSPS ADC, 1 set of 5-bit DAC and 2 sets of ACMP in both SSOP20 and TSSOP28 package. Moreover, it provides low voltage reset (LVR) and brown-out detector (BOD) to ensure the system safety.

The NuMicro® M0A21/M0A23 series runs up to 48 MHz and supports hardware divider. It provides 32 Kbytes Flash memory, 4 Kbytes SRAM and 2 Kbytes LDROM for ISP (In-System Programming) feature for easily firmware update. It is equipped with plenty of peripherals including up to four 32-bit timers, 6-ch 16-bit PWM generators, 1 set of CAN 2.0B controller, 2 sets of LIN functions, 5-ch PDMA, 2 sets of UART with One-Wire mode, IrDA and RS485 functions. Besides, the M0A21/M0A23 series provides two sets of Universal Serial Control Interfaces (USCI) that can be configured as UART, SPI or I^2C.

The package types of the M0A21/M0A23 series are included SSOP20 (5.3x7.2x1.75 mm) and TSSOP28 (4.4x9.7x1.0 mm).

Product Line	CAN	UART	LIN	USCI	Timer	PWM	PDMA	ADC	ACMP	Divider
M0A21	0	2	2	2	4	6	5	17	2	1
M0A23	1	2	2	2	4	6	5	17	2	1

Table 1-1 NuMicro® M0A21/M0A23 Series Key Features Table

The M0A21/M0A23 is targeted at applications such as:

- 24GHz mmWave radar
- BMS (Battery Management System)
- Car lighting
- Car windows
- Power seat

2 FEATURES

Core and System	
Arm® Cortex®-M0	• Arm® Cortex®-M0 core, running up to 48 MHz • Built-in Nested Vectored Interrupt Controller (NVIC) • 24-bit system tick timer • Programmble and maskable interrupt • Low Power Sleep mode by WFI and WFE instructions
Brown-out Detector (BOD)	• Four-level BOD with brown-out interrupt and reset option. (4.4V/3.7V/2.7V/2.3V)
Low Voltage Reset (LVR)	• LVR with 2.22V threshold voltage level
Security	• 96-bit Unique ID (UID). • 128-bit Unique Customer ID (UCID). • One built-in temperature sensor.
32-bit H/W Divider(HDIV)	• Signed (two's complement) integer calculation • 32-bit dividend with 16-bit divisor calculation capacity • 32-bit quotient and 32-bit remainder outputs (16-bit remainder with sign extends to 32-bit) • 6 HCLK clocks taken for one cycle calculation
Memories	
Boot Loader	• Nuvoton ISP (In-System-Programming) tool for firmware upgrade via UART • ISP/IAP libraries
Flash	• Up to 32 KB application ROM (APROM) • 2 KB on-chip Flash for user-defined loader (LDROM) • All on-chip Flash support 512 bytes page erase • Fast Flash programming verification with CRC • On-chip Flash programming with In-Chip Programming (ICP), In-System Programming (ISP) and In-Application Programming (IAP) capabilities • Configurable boot up sources including boot loader, user-defined loader (LDROM) or Application ROM (APROM)

	• Data Flash with configurable memory size • 2-wired ICP Flash updating through SWD interface • 32-bit and multi-word Flash programming function
SRAM	• Up to 4 KB embedded SRAM • Supports byte-, half-word- and word-access • Supports PDMA mode
Cyclic Redundancy Calculation (CRC)	• Supports CRC-CCITT, CRC-8, CRC-16 and CRC-32 polynomials • Programmable initial value • Programmable order reverse setting and one's complement setting for input data and CRC checksum • 8-bit, 16-bit, and 32-bit data width • 8-bit write mode with 1-AHB clock cycle operation • 16-bit write mode with 2-AHB clock cycle operation • 32-bit write mode with 4-AHB clock cycle operation • Uses DMA to write data with performing CRC operation
Peripheral DMA (PDMA)	• Supports up to 5 independent configurable channels for automatic data transfer between memories and peripherals • Basic and Scatter-Gather transfer modes • Each channel supports circular buffer management using Scatter-Gather Transfer mode • Stride function for rectangle image data movement • Fixed-priority and Round-robin priorities modes • Single and burst transfer types • Byte-, half-word- and word tranfer unit with count up to 65536 • Request source can be from software, UART, ADC, PWM and Timer
Clocks	
External Clock Source	• 4~24 MHz High-speed eXternal crystal oscillator (HXT) for precise timing operation • 32.768 kHz Low-speed eXternal crystal oscillator (LXT) for low-power system operation • Supports clock failure detection for external crystal oscillators and exception generatation (NMI)
Internal Clock Source	• 48 MHz High-speed Internal RC oscillator (HIRC) trimmed to

	±0.25% accuracy that can optionally be used as a system clock • 38.4 kHz Low-speed Internal RC oscillator (LIRC) for watchdog timer and wakeup operation
Timers	
32-bit Timer	• Four sets of 32-bit timers with 24-bit up counter and one 8-bit pre-scale counter from independent clock source • One-shot, Periodic, Toggle and Continuous Counting operation modes • Supports event counting function to count the event from external pins • Supports external capture pin for enhanced interval measurement and resetting 24-bit up counter • Supports 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 PWM, ADC and PDMA.
PWM	• Three 16-bit counters with 12-bit prescale for six 48 MHz PWM 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 16-bit resolution PWM counter • Up, down or up-down PWM counter type • Supports mask function and tri-state enable for each PWM pin • Supports brake function • Up to 6 independent input capture channels with 16-bit resolution counter • Counter synchronous start function • Able to trigger ADC to start conversion.
Watchdog	• 20-bit free running up counter for WDT time-out interval. • Selectable time-out interval (24 ~ 220) and the time-out interval is 416us ~ 27.3 s if WDT_CLK = 38.4 kHz (LIRC). • System kept in reset state for a period of (1 / WDT_CLK) * 63. • Able to wake up from Power-down or Idle mode

	• Interrupt or reset selectable on watchdog time-out • 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.
Window Watchdog	• Clock sources from HCLK/2048 (default selection) or LIRC • Window set by 6-bit down counter with 11-bit prescale • WWDT counter suspends in Idle/Power-down mode • Supports Interrupt
Analog Interfaces	
Analog-to-Digital Converter (ADC)	• Analog input voltage range: 0 ~ AV_{DD} (voltage of V_{DD} pin). • 12-bit resolution and 10-bit accuracy is guaranteed. • Up to 17 single-end analog input channels or 8 differential analog input channels • Maximum ADC peripheral clock frequency is 16 MHz. • Up to 800 kSPS sampling rate. • Four operation modes: - Single mode: A/D conversion is performed one time on a specified channel. - Burst mode: A/D converter samples and converts the specified single channel and sequentially stores the result in FIFO. - 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 (STADC). • Timer 0~3 overflow pulse trigger. • PWM trigger. • Each conversion result is 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. • 4 internal channels band-gap voltage (VBG), temperature sensor (V_{TEMP}), internal reference voltage and DAC0 output • Supports PDMA transfer mode
Digital to Analog Converter (DAC)	• Supports one 5-bit 100 kSPS voltage type DAC • Analog output voltage range: 0~AV_{DD} (voltage of V_{DD} pin) • Reference voltage from internal reference voltage V_{REF} pin or AV_{DD} • DAC maximum conversion updating rate 100K sps • Rail to rail settle time 10us • Supports software and timer0~3 trigger to start DAC conversion • Supports PDMA mode
Analog Comparator (ACMP)	• Analog input voltage range: 0 ~ AV_{DD} (voltage of V_{DD} pin) • Up to two rail-to-rail analog comparators • Supports hysteresis function • Supports wake-up function • Selectable input sources of positive input and negative input **ACMP0 supports:** • 3 multiplexed I/O pins at positive sources: - ACMP0_P0, Comparator Reference Voltage (CRV), and DAC0 output • 5 negative sources: - ACMP0_N0, ACMP0_N1, ACMP0_N2, ACMP0_N3 - Comparator Reference Voltage (CRV) **ACMP1 supports:** • 3 multiplexed I/O pins at positive sources: - ACMP1_P0, Comparator Reference Voltage (CRV), and DAC0 output • 5 negative sources: **ACMP1_N** • ACMP1_N0, ACMP1_N1, ACMP1_N2, ACMP1_N3 • Comparator Reference Voltage (CRV) • Shares one ACMP interrupt vector for all comparators • Interrupts generated when compare results change (Interrupt event condition is programmable)

	• Supports triggers for break events and cycle-by-cycle control for PWM • Supports window compare mode and window latch mode
Communication Interfaces	
UART	• Supports up to 2 UARTs: UART0, UART1 • UART baud rate clock from LXT (32.768 kHz) with 9600bps can work normally in power down mode even system clock is stopped • Full-duplex asynchronous communications • Separates receive and transmit 16/16 bytes entry FIFO for data payloads • Supports hardware auto-flow control (RX, TX, CTS and RTS) and programmable receiver buffer trigger level • Supports programmable baud rate generator for each channel individually • 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 • 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 LIN function mode • Supports LIN master/slave mode • Supports programmable break generation function for transmitter • Supports break detection function for receiver • Supports LIN slave header time-out detection function • Supports LIN response time-out detection function • Supports LIN wake-up function • Supports IrDA SIR function mode • Supports for 3/16 bit duration for normal mode

	• Supports RS-485 mode • Supports RS-485 9-bit mode • Supports hardware or software enables to program nRTS pin to control RS-485 transmission direction • Supports nCTS, incoming data, Received Data FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function • Supports PDMA mode • Supports Single-wire function mode.
CAN	• Supports CAN protocol version 2.0 part A and B • Bit rates up to 1 MBit/s • 32 Message Objects • Supports wake-up function
Universal Serial Control Interface (USCI)	• Supports one set of USCI • USCI supports UART, SPI and I^2C function • Single byte TX and RX buffer mode **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 **SPI** • Master or Slave mode operation (maximum frequency: master = fPCLK / 2, slave < fPCLK / 5) • 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
I²C	
	• Full master and slave device capability • 7-bit addressing mode (10-bit mode Not supported) • Communication in standard mode (100 kbps) or in fast mode (up to 400 kbps) • Multi-master bus • One transmit buffer and two receive buffer for data payload • Supports 10-bit bus time-out capability • Supports Bus monitor mode • Power down wake-up by data toggle or address match • Multiple address recognition • Device address flag • Setup/hold time programmable
GPIO	• Four I/O modes: - Quasi-bidirectional mode - Push-Pull Output mode - Open-Drain Output mode - Input only with high impendence mode • Schmitt trigger input • I/O pin configured as interrupt source with edge/level trigger setting • Supports high drive and high sink current I/O • Supports independent pull-up control • Maximum I/O Speed is 24 MHz when V_{DD} = 2.4 ~ 5.5V. • Supports up to 18/26 GPIOs for SSOP20 TSSOP28 respectively • Enabling the pin interrupt function will also enable the wake-up function

3 PARTS INFORMATION

3.1 Package Type

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

SSOP20	TSSOP28
M0A21OB1AC	M0A21EB1AC
M0A21OC1AC	M0A21EC1AC
M0A23OC1AC	M0A23EC1AC

3.2 NuMicro® M0A21/M0A23 Series Selection Guide

3.2.1 NuMicro® M0A21 Series

Part Number	Flash (KB)	SRAM (KB)	ISP ROM (KB)	I/O	Timer	PWM	PDMA	Connectivity				DAC	LXT	ACMP	ADC(12-Bit)	Package
								USCI*	UART	LIN	CAN					
M0A21OB1AC	16	4	2	18	4	6	5	2	2	2	-	1	1	2	17-ch	SSOP20
M0A21OC1AC	32	4	2	18	4	6	5	2	2	2	-	1	1	2	17-ch	SSOP20
M0A21EB1AC	16	4	2	26	4	6	5	2	2	2	-	1	1	2	17-ch	TSSOP28
M0A21EC1AC	32	4	2	26	4	6	5	2	2	2	-	1	1	2	17-ch	TSSOP28

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

3.2.2 NuMicro® M0A23 Series

Part Number	Flash (KB)	SRAM (KB)	ISP ROM (KB)	I/O	Timer	PWM	PDMA	Connectivity				DAC	LXT	ACMP	ADC(12-Bit)	Package
								USCI*	UART	LIN	CAN					
M0A23OC1AC	32	4	2	18	4	6	5	2	2	2	1	1	1	2	17-ch	SSOP20
M0A23EC1AC	32	4	2	26	4	6	5	2	2	2	1	1	1	2	17-ch	TSSOP28

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

3.2.3 NuMicro® M0A21/M0A23 Selection Code

Core	Series	Package	Flash Size	SRAM Size	Revision	Temperature
M0	A21	O	C	1	A	C
Cortex®-M0	A21: without CAN A23: with CAN	O: SSOP20 (5.3x7.2 mm) E: SSOP28 (4.4x9.7 mm)	B: 16 KB C: 32 KB	1: 4 KB		C:-40°C ~ 125°C

4 PIN CONFIGURATION

The pin configuaration information can be found in the M0A21/M0A23 Series Multi-function Summary Table section or by using NuTool - PinConfigure. The NuTool - PinConfigure contains all Nuvoton NuMicro® Family chip series with all part number, and helps configure GPIO multi-function pins correctly and handily.

4.1 Pin Configuration

4.1.1 M0A21 Series Pin Diagram

4.1.1.1 SSOP20 Package

Corresponding Part Number: M0A21OB1AC, M0A21OC1AC

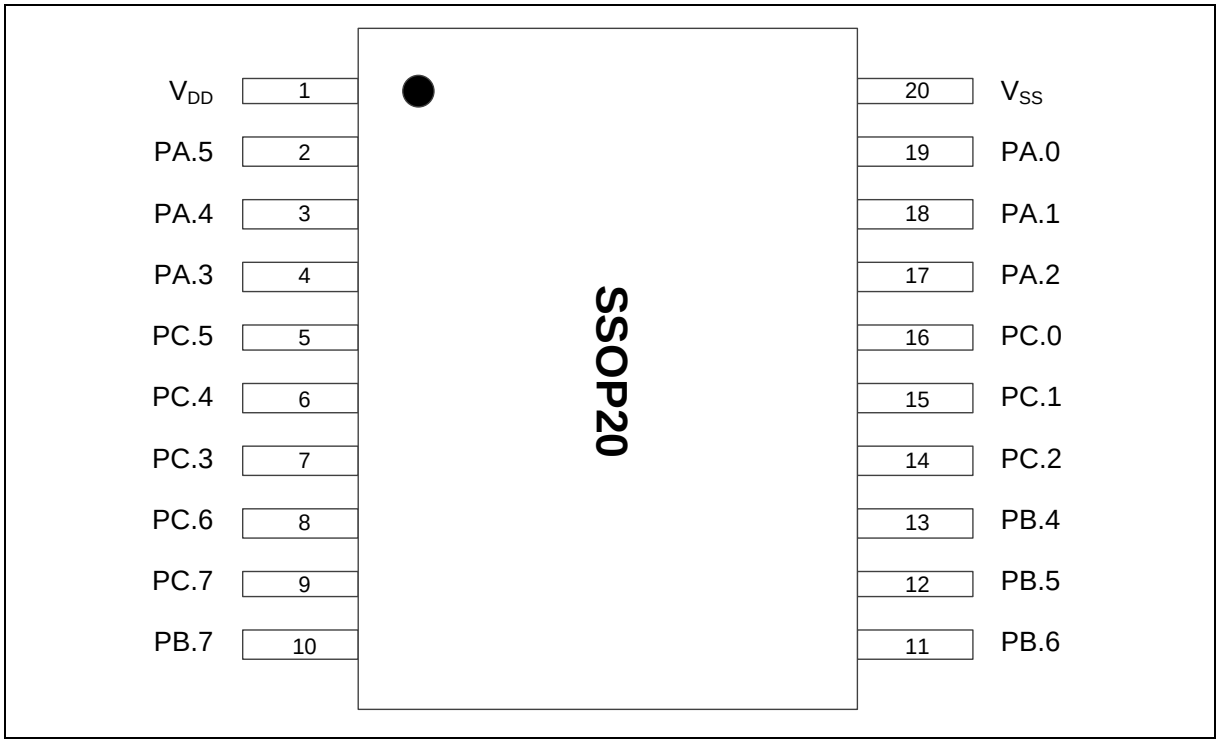


Figure 4.1-1 M0A21 Series SSOP 20-pin Diagram

4.1.1.2 TSSOP28 Package

Corresponding Part Number: M0A21EB1AC, M0A21EC1AC

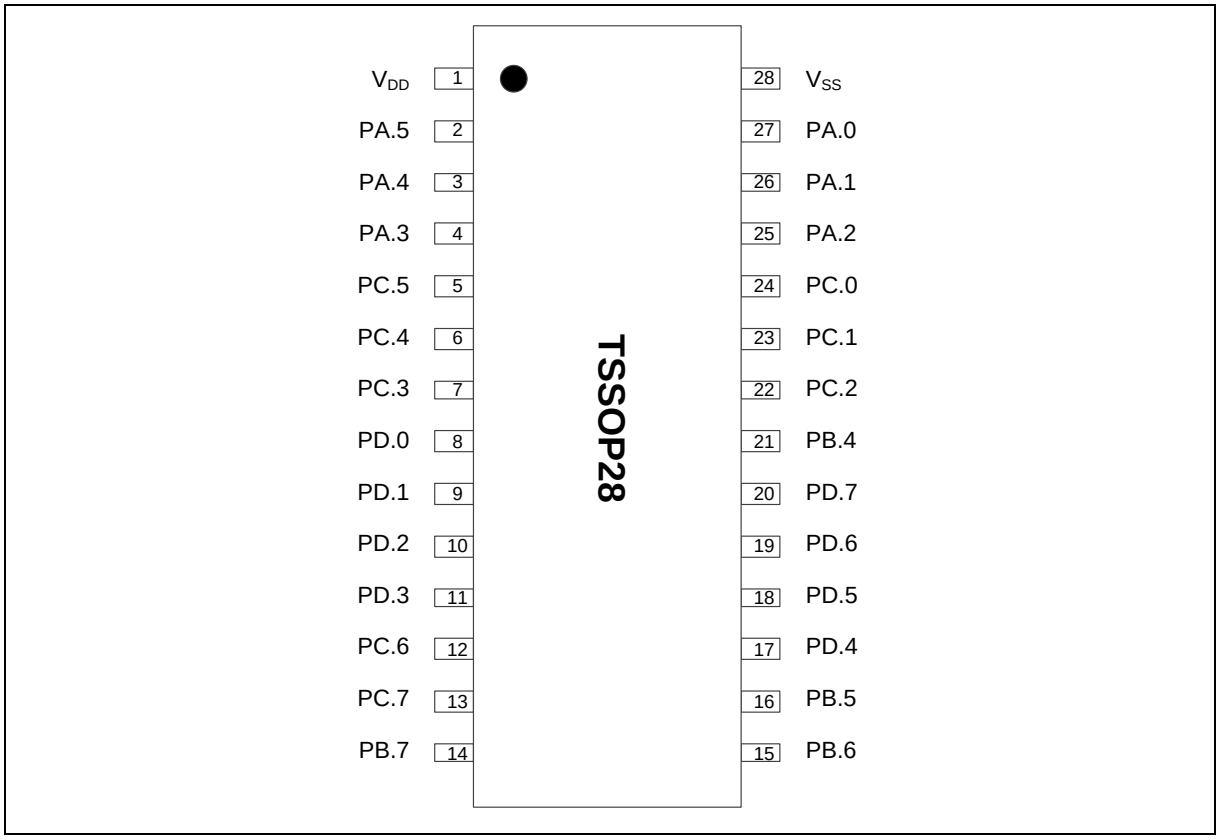


Figure 4.1-2 M0A21 Series TSSOP 28-pin Diagram

4.1.2 M0A21 Series Function Pin Table

4.1.2.1 SSOP20 Package

Corresponding Part Number: M0A21OB1AC, M0A21OC1AC

M0A21OB1AC / M0A21OC1AC Multi-function Pin Table

Pin	M0A21OB1AC / M0A21OC1AC Pin Function
1	V_{DD}
2	PA.5 / ADC0_CH16 / UART0_nRTS / XT1_IN / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / ACMP0_WLAT
3	PA.4 / ADC0_CH15 / UART0_nRTS / XT1_OUT / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / ACMP1_WLAT
4	PA.3 / nRESET / UART0_nCTS / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_RXD / INT0
5	PC.5 / ADC0_CH14 / X32_IN / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 /

Pin	M0A21OB1AC / M0A21OC1AC Pin Function
	USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT1
6	PC.4 / ADC0_CH13 / X32_OUT / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT2
7	PC.3 / ADC0_CH12 / ACMP0_N3 / ACMP1_N3 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT3
8	PC.6 / ADC0_CH11 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT4
9	PC.7 / ADC0_CH10 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT5
10	PB.7 / ADC0_CH9 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / ACMP0_WLAT
11	PB.6 / ADC0_CH8 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / ACMP1_WLAT
12	PB.5 / ADC0_CH7 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT0
13	PB.4 / ADC0_CH6 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / UART1_nCTS
14	PC.2 / ADC0_CH5 / ACMP0_N2 / ACMP1_N2 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / UART1_nRTS
15	PC.1 / ADC0_CH4 / ACMP0_N1 / ACMP1_N1 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD
16	PC.0 / ADC0_CH3 / ACMP1_P0 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE0
17	PA.2 / ADC0_CH2 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE1

Pin	M0A21OB1AC / M0A21OC1AC Pin Function
18	PA.1 / ADC0_CH1 / ACMP0_N0 / ACMP1_N0 / VREF / ICE_CLK / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE0
19	PA.0 / ADC0_CH0 / DAC0_OUT / ACMP0_P0 / ICE_DAT / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE1
20	V_{SS}

4.1.2.2 TSSOP28 Package

Corresponding Part Number: M0A21EB1AC, M0A21EC1AC

M0A21EB1AC / M0A21EC1AC Multi-function Pin Table

Pin	M0A21EB1AC / M0A21EC1AC Pin Function
1	V_{DD}
2	PA.5 / ADC0_CH16 / UART0_nRTS / XT1_IN / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / ACMP0_WLAT
3	PA.4 / ADC0_CH15 / UART0_nRTS / XT1_OUT / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / ACMP1_WLAT
4	PA.3 / nRESET / UART0_nCTS / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_RXD / INT0
5	PC.5 / ADC0_CH14 / X32_IN / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT1
6	PC.4 / ADC0_CH13 / X32_OUT / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT2
7	PC.3 / ADC0_CH12 / ACMP0_N3 / ACMP1_N3 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT3
8	PD.0 / PWM0_CH4 / UART0_TXD / USCI1_CLK / TM0
9	PD.1 / PWM0_CH5 / UART0_RXD / USCI1_DAT0 / TM1
10	PD.2 / PWM0_CH0 / USCI1_DAT1 / TM2 / UART1_nCTS
11	PD.3 / PWM0_CH1 / USCI1_CTL0 / TM3 / UART1_nRTS

Pin	M0A21EB1AC / M0A21EC1AC Pin Function
12	PC.6 / ADC0_CH11 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT4
13	PC.7 / ADC0_CH10 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT5
14	PB.7 / ADC0_CH9 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / ACMP0_WLAT
15	PB.6 / ADC0_CH8 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / ACMP1_WLAT
16	PB.5 / ADC0_CH7 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT0
17	PD.4 / PWM0_CH0 / UART0_TXD / USCI0_CLK / TM0
18	PD.5 / PWM0_CH1 / UART0_RXD / USCI0_DAT0 / TM1
19	PD.6 / PWM0_CH2 / USCI0_DAT1 / TM2 / UART1_nCTS
20	PD.7 / PWM0_CH3 / USCI0_CTL0 / TM3 / UART1_nRTS
21	PB.4 / ADC0_CH6 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / UART1_nCTS
22	PC.2 / ADC0_CH5 / ACMP0_N2 / ACMP1_N2 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / UART1_nRTS
23	PC.1 / ADC0_CH4 / ACMP0_N1 / ACMP1_N1 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD
24	PC.0 / ADC0_CH3 / ACMP1_P0 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE0
25	PA.2 / ADC0_CH2 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE1
26	PA.1 / ADC0_CH1 / ACMP0_N0 / ACMP1_N0 / VREF / ICE_CLK / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD /

Pin	M0A21EB1AC / M0A21EC1AC Pin Function
	PWM0_BRAKE0
27	PA.0 / ADC0_CH0 / DAC0_OUT / ACMP0_P0 / ICE_DAT / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE1
28	V_{SS}

4.1.3 M0A23 Series Pin Diagram

4.1.3.1 SSOP20 Package

Corresponding Part Number: M0A23OC1AC

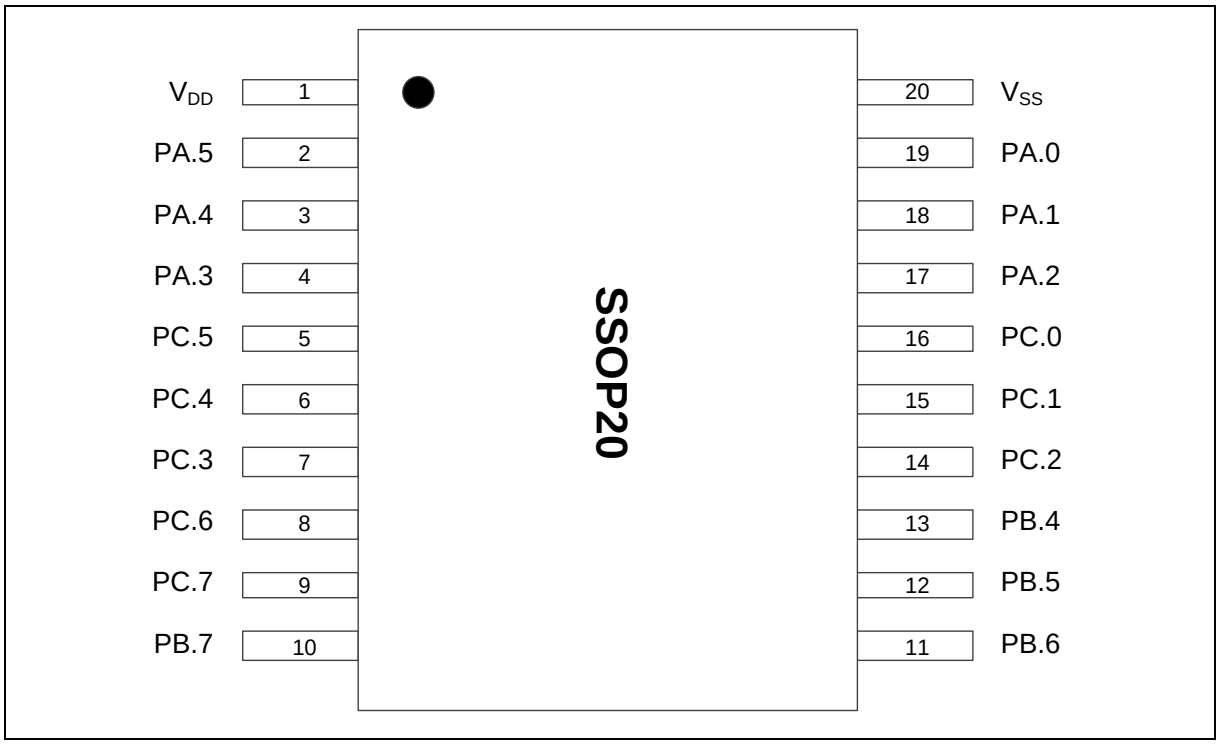


Figure 4.1-3 M0A23 Series SSOP 20-pin Diagram

4.1.3.2 TSSOP28 Package

Corresponding Part Number: M0A23EC1AC.

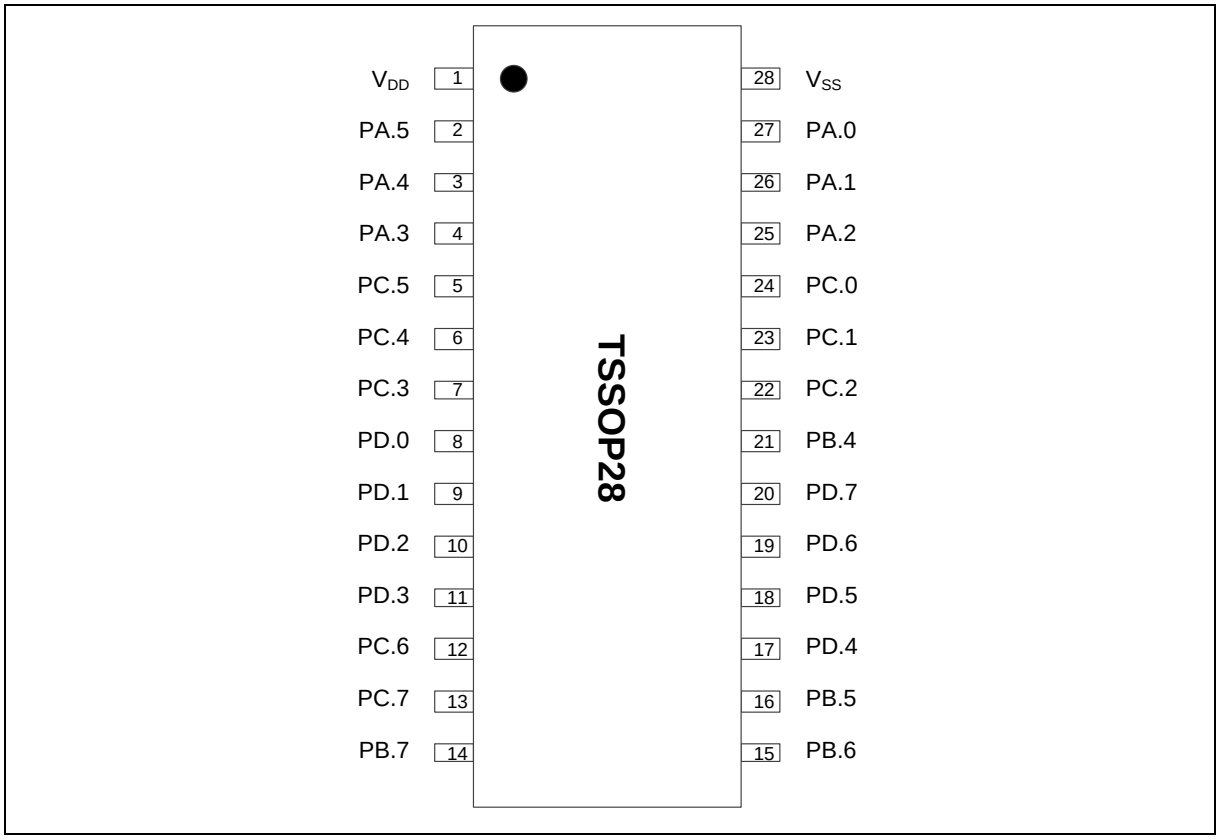


Figure 4.1-4 M0A23 Series TSSOP 28-pin Diagram

4.1.4 M0A23 Series Function Pin Table

4.1.4.1 SSOP20 Package

Corresponding Part Number: M0A23OC1AC.

M0A23OC1AC Multi-function Pin Table

Pin	M0A23OC1AC Pin Function
1	V_{DD}
2	PA.5 / ADC0_CH16 / UART0_nRTS / XT1_IN / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / ACMP0_WLAT
3	PA.4 / ADC0_CH15 / UART0_nRTS / XT1_OUT / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / ACMP1_WLAT
4	PA.3 / nRESET / UART0_nCTS / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_RXD / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_RXD / INT0
5	PC.5 / ADC0_CH14 / X32_IN / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT1
6	PC.4 / ADC0_CH13 / X32_OUT / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT2
7	PC.3 / ADC0_CH12 / ACMP0_N3 / ACMP1_N3 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT3
8	PC.6 / ADC0_CH11 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT4
9	PC.7 / ADC0_CH10 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT5
10	PB.7 / ADC0_CH9 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / ACMP0_WLAT
11	PB.6 / ADC0_CH8 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / ACMP1_WLAT
12	PB.5 / ADC0_CH7 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O /

Pin	M0A23OC1AC Pin Function
	ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT0
13	PB.4 / ADC0_CH6 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / UART1_nCTS
14	PC.2 / ADC0_CH5 / ACMP0_N2 / ACMP1_N2 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / UART1_nRTS
15	PC.1 / ADC0_CH4 / ACMP0_N1 / ACMP1_N1 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD
16	PC.0 / ADC0_CH3 / ACMP1_P0 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE0
17	PA.2 / ADC0_CH2 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE1
18	PA.1 / ADC0_CH1 / ACMP0_N0 / ACMP1_N0 / VREF / ICE_CLK / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE0
19	PA.0 / ADC0_CH0 / DAC0_OUT / ACMP0_P0 / ICE_DAT / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE1
20	V_{SS}

4.1.4.2 TSSOP28 Package

Corresponding Part Number: M0A23EC1AC.

M0A23EC1AC Multi-function Pin Table

Pin	M0A23EC1AC Pin Function
1	V_{DD}
2	PA.5 / ADC0_CH16 / UART0_nRTS / XT1_IN / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / ACMP0_WLAT

Pin	M0A23EC1AC Pin Function
3	PA.4 / ADC0_CH15 / UART0_nRTS / XT1_OUT / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / ACMP1_WLAT
4	PA.3 / nRESET / UART0_nCTS / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_RXD / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_RXD / INT0
5	PC.5 / ADC0_CH14 / X32_IN / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT1
6	PC.4 / ADC0_CH13 / X32_OUT / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT2
7	PC.3 / ADC0_CH12 / ACMP0_N3 / ACMP1_N3 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT3
8	PD.0 / PWM0_CH4 / UART0_TXD / USCI1_CLK / TM0
9	PD.1 / PWM0_CH5 / UART0_RXD / USCI1_DAT0 / TM1
10	PD.2 / PWM0_CH0 / CAN0_TXD / USCI1_DAT1 / TM2 / UART1_nCTS
11	PD.3 / PWM0_CH1 / CAN0_RXD / USCI1_CTL0 / TM3 / UART1_nRTS
12	PC.6 / ADC0_CH11 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT4
13	PC.7 / ADC0_CH10 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / INT5
14	PB.7 / ADC0_CH9 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / ACMP0_WLAT
15	PB.6 / ADC0_CH8 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / ACMP1_WLAT
16	PB.5 / ADC0_CH7 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / INT0
17	PD.4 / PWM0_CH0 / UART0_TXD / USCI0_CLK / TM0
18	PD.5 / PWM0_CH1 / UART0_RXD / USCI0_DAT0 / TM1

Pin	M0A23EC1AC Pin Function
19	PD.6 / PWM0_CH2 / CAN0_TXD / USCI0_DAT1 / TM2 / UART1_nCTS
20	PD.7 / PWM0_CH3 / CAN0_RXD / USCI0_CTL0 / TM3 / UART1_nRTS
21	PB.4 / ADC0_CH6 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / UART1_nCTS
22	PC.2 / ADC0_CH5 / ACMP0_N2 / ACMP1_N2 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / UART1_nRTS
23	PC.1 / ADC0_CH4 / ACMP0_N1 / ACMP1_N1 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD
24	PC.0 / ADC0_CH3 / ACMP1_P0 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE0
25	PA.2 / ADC0_CH2 / UART0_nRTS / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE1
26	PA.1 / ADC0_CH1 / ACMP0_N0 / ACMP1_N0 / VREF / ICE_CLK / UART0_nCTS / CLKO / PWM0_CH0 / PWM0_CH2 / PWM0_CH4 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI1_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM0 / TM2 / TM0_EXT / TM2_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE0
27	PA.0 / ADC0_CH0 / DAC0_OUT / ACMP0_P0 / ICE_DAT / UART0_nCTS / CLKO / PWM0_CH1 / PWM0_CH3 / PWM0_CH5 / UART0_TXD / UART0_RXD / USCI0_CLK / USCI0_DAT0 / USCI0_DAT1 / USCI0_CTL0 / USCI0_CTL1 / USCI1_CLK / USCI1_DAT0 / USCI1_DAT1 / USCI1_CTL0 / CAN0_TXD / CAN0_RXD / ACMP0_O / ACMP1_O / ADC0_ST / TM1 / TM3 / TM1_EXT / TM3_EXT / UART1_TXD / UART1_RXD / PWM0_BRAKE1
28	V_{SS}

4.2 Pin Description

Different part number with same package might have different functions. Please refer to the selection guide in section 3.2.

4.2.1 M0A21/M0A23 Series Pin Description

20 Pin	28 Pin	Pin Name	Type	MFP	Description
1	1	V_{DD}	P	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
2	2	PA.5	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH16	A	MFP1	ADC0 channel 16 analog input.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		XT1_IN	I	MFP3	External 4~24 MHz (high speed) crystal input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
		PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
		PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
		TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
		TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
		TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		ACMP0_WLAT	I	MFP30	Analog comparator 0 window latch input pin
3	3	PA.4	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH15	A	MFP1	ADC0 channel 15 analog input.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		XT1_OUT	O	MFP3	External 4~24 MHz (high speed) crystal output pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
		PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
		PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
		TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
		TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
		TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		ACMP1_WLAT	I	MFP30	Analog comparator 1 window latch input pin

20 Pin	28 Pin	Pin Name	Type	MFP	Description
4	4	PA.3	I	MFP0	General purpose digital input pin.
		nRESET	I	MFP2	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.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		PWM0_CH0	I	MFP5	PWM0 channel 0 capture input.
		PWM0_CH2	I	MFP6	PWM0 channel 2 capture input.
		PWM0_CH4	I	MFP7	PWM0 channel 4 capture input.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I	MFP10	USCI0 clock pin.
		USCI0_DAT0	I	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I	MFP13	USCI0 control 0 pin.
		USCI0_CTL1	I	MFP14	USCI0 control 1 pin.
		USCI1_CLK	I	MFP15	USCI1 clock pin.
		USCI1_DAT0	I	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I	MFP18	USCI1 control 0 pin.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM0	I	MFP24	Timer0 event counter input pin.
		TM2	I	MFP25	Timer2 event counter input pin.
		TM0_EXT	I	MFP26	Timer0 external capture input pin.
		TM2_EXT	I	MFP27	Timer2 external capture input pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		INT0	I	MFP30	External interrupt 0 input pin.
5	5	PC.5	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH14	A	MFP1	ADC0 channel 14 analog input.
		X32_IN	I	MFP2	External 32.768 kHz crystal input pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
		PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
		PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
		TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
		TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
		TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		INT1	I	MFP30	External interrupt 1 input pin.
6	6	PC.4	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH13	A	MFP1	ADC0 channel 13 analog input.
		X32_OUT	O	MFP2	External 32.768 kHz crystal output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
		PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
		PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
		TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
		TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
		TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		INT2	I	MFP30	External interrupt 2 input pin.
7	7	PC.3	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH12	A	MFP1	ADC0 channel 12 analog input.
		ACMP0_N3	A	MFP1	Analog comparator 0 negative input 3 pin.
		ACMP1_N3	A	MFP1	Analog comparator 1 negative input 3 pin.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
		PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
		PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
		TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
		TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
		TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		INT3	I	MFP30	External interrupt 3 input pin.
	8	PD.0	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH4	I/O	MFP2	PWM0 channel 4 output/capture input.
		UART0_TXD	O	MFP3	UART0 data transmitter output pin.
		USCI1_CLK	I/O	MFP4	USCI1 clock pin.
		TM0	I/O	MFP5	Timer0 event counter input/toggle output pin.
	9	PD.1	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH5	I/O	MFP2	PWM0 channel 5 output/capture input.
		UART0_RXD	I	MFP3	UART0 data receiver input pin.
		USCI1_DAT0	I/O	MFP4	USCI1 data 0 pin.
		TM1	I/O	MFP5	Timer1 event counter input/toggle output pin.
	10	PD.2	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH0	I/O	MFP2	PWM0 channel 0 output/capture input.
		CAN0_TXD	O	MFP3	CAN0 bus transmitter output.
		USCI1_DAT1	I/O	MFP4	USCI1 data 1 pin.
		TM2	I/O	MFP5	Timer2 event counter input/toggle output pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		UART1_nCTS	I	MFP6	UART1 clear to Send input pin.
	11	PD.3	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH1	I/O	MFP2	PWM0 channel 1 output/capture input.
		CAN0_RXD	I	MFP3	CAN0 bus receiver input.
		USCI1_CTL0	I/O	MFP4	USCI1 control 0 pin.
		TM3	I/O	MFP5	Timer3 event counter input/toggle output pin.
		UART1_nRTS	O	MFP6	UART1 request to Send output pin.
8	12	PC.6	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH11	A	MFP1	ADC0 channel 11 analog input.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
		PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
		PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
		TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
		TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		INT4	I	MFP30	External interrupt 4 input pin.
9	13	PC.7	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH10	A	MFP1	ADC0 channel 10 analog input.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
		PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
		PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
		TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
		TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
		TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		INT5	I	MFP30	External interrupt 5 input pin.
10	14	PB.7	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH9	A	MFP1	ADC0 channel 9 analog input.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
		PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
		PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
		TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
		TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
		TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		ACMP0_WLAT	I	MFP30	Analog comparator 0 window latch input pin
11	15	PB.6	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH8	A	MFP1	ADC0 channel 8 analog input.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
		PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
		PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
		TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
		TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
		TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		ACMP1_WLAT	I	MFP30	Analog comparator 1 window latch input pin
12	16	PB.5	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH7	A	MFP1	ADC0 channel 7 analog input.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
		PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
		PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
		TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
		TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
		TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		INT0	I	MFP30	External interrupt 0 input pin.
	17	PD.4	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH0	I/O	MFP2	PWM0 channel 0 output/capture input.
		UART0_TXD	O	MFP3	UART0 data transmitter output pin.
		USCI0_CLK	I/O	MFP4	USCI0 clock pin.
		TM0	I/O	MFP5	Timer0 event counter input/toggle output pin.
	18	PD.5	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH1	I/O	MFP2	PWM0 channel 1 output/capture input.
		UART0_RXD	I	MFP3	UART0 data receiver input pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		USCI0_DAT0	I/O	MFP4	USCI0 data 0 pin.
		TM1	I/O	MFP5	Timer1 event counter input/toggle output pin.
	19	PD.6	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH2	I/O	MFP2	PWM0 channel 2 output/capture input.
		CAN0_TXD	O	MFP3	CAN0 bus transmitter output.
		USCI0_DAT1	I/O	MFP4	USCI0 data 1 pin.
		TM2	I/O	MFP5	Timer2 event counter input/toggle output pin.
		UART1_nCTS	I	MFP6	UART1 clear to Send input pin.
	20	PD.7	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH3	I/O	MFP2	PWM0 channel 3 output/capture input.
		CAN0_RXD	I	MFP3	CAN0 bus receiver input.
		USCI0_CTL0	I/O	MFP4	USCI0 control 0 pin.
		TM3	I/O	MFP5	Timer3 event counter input/toggle output pin.
		UART1_nRTS	O	MFP6	UART1 request to Send output pin.
13	21	PB.4	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH6	A	MFP1	ADC0 channel 6 analog input.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
		PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
		PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
		TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
		TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
		TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		UART1_nCTS	I	MFP30	UART1 clear to Send input pin.
14	22	PC.2	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH5	A	MFP1	ADC0 channel 5 analog input.
		ACMP0_N2	A	MFP1	Analog comparator 0 negative input 2 pin.
		ACMP1_N2	A	MFP1	Analog comparator 1 negative input 2 pin.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
		PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
		PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
		TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
		TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
		TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		UART1_nRTS	O	MFP30	UART1 request to Send output pin.
15	23	PC.1	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH4	A	MFP1	ADC0 channel 4 analog input.
		ACMP0_N1	A	MFP1	Analog comparator 0 negative input 1 pin.
		ACMP1_N1	A	MFP1	Analog comparator 1 negative input 1 pin.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
		PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
		PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
		TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
		TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
		TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
16	24	PC.0	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH3	A	MFP1	ADC0 channel 3 analog input.
		ACMP1_P0	A	MFP1	Analog comparator 1 positive input 0 pin.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
		PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
		PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
		TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
		TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		PWM0_BRAKE0	I	MFP30	PWM0 Brake 0 input pin.
17	25	PA.2	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH2	A	MFP1	ADC0 channel 2 analog input.
		UART0_nRTS	O	MFP2	UART0 request to Send output pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
		PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
		PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
		TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
		TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
		TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		PWM0_BRAKE1	I	MFP30	PWM0 Brake 1 input pin.
18	26	PA.1	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH1	A	MFP1	ADC0 channel 1 analog input.
		ACMP0_N0	A	MFP1	Analog comparator 0 negative input 0 pin.
		ACMP1_N0	A	MFP1	Analog comparator 1 negative input 0 pin.
		V_{REF}	A	MFP1	ADC reference voltage input. **Note:** This pin needs to be connected with a 1uF capacitor.
		ICE_CLK	I	MFP2	Serial wired debugger clock pin. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_CLK pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
		PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
		PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
		TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
		TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		PWM0_BRAKE0	I	MFP30	PWM0 Brake 0 input pin.
19	27	PA.0	I/O	MFP0	General purpose digital I/O pin.
		ADC0_CH0	A	MFP1	ADC0 channel 0 analog input.
		DAC0_OUT	A	MFP1	DAC0 channel analog output.
		ACMP0_P0	A	MFP1	Analog comparator 0 positive input 0 pin.
		ICE_DAT	I/O	MFP2	Serial wired debugger data pin. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_DAT pin.
		UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
		CLKO	O	MFP4	Clock Out
		PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
		PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
		PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
		UART0_TXD	O	MFP8	UART0 data transmitter output pin.
		UART0_RXD	I	MFP9	UART0 data receiver input pin.
		USCI0_CLK	I/O	MFP10	USCI0 clock pin.
		USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
		USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
		USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
		USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
		USCI1_CLK	I/O	MFP15	USCI1 clock pin.
		USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
		USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
		USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
		CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
		CAN0_RXD	I	MFP20	CAN0 bus receiver input.
		ACMP0_O	O	MFP21	Analog comparator 0 output pin.
		ACMP1_O	O	MFP22	Analog comparator 1 output pin.
		ADC0_ST	I	MFP23	ADC0 external trigger input pin.
		TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
		TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.

20 Pin	28 Pin	Pin Name	Type	MFP	Description
		TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
		TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
		UART1_TXD	O	MFP28	UART1 data transmitter output pin.
		UART1_RXD	I	MFP29	UART1 data receiver input pin.
		PWM0_BRAKE1	I	MFP30	PWM0 Brake 1 input pin.
20	28	V_{SS}	P	MFP0	Ground pin for digital circuit.

4.2.2 M0A21/M0A23 Series Multi-function Summary Table

Group	Pin Name	GPIO	MFP	Type	Description
ACMP0	ACMP0_N0	PA.1	MFP1	A	Analog comparator 0 negative input 0 pin.
	ACMP0_N1	PC.1	MFP1	A	Analog comparator 0 negative input 1 pin.
	ACMP0_N2	PC.2	MFP1	A	Analog comparator 0 negative input 2 pin.
	ACMP0_N3	PC.3	MFP1	A	Analog comparator 0 negative input 3 pin.
	ACMP0_O	PA.5	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PA.4	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PC.5	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PC.4	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PC.3	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PC.6	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PC.7	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PB.7	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PB.6	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PB.5	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PB.4	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PC.2	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PC.1	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PC.0	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PA.2	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PA.1	MFP21	O	Analog comparator 0 output pin.
	ACMP0_O	PA.0	MFP21	O	Analog comparator 0 output pin.
	ACMP0_P0	PA.0	MFP1	A	Analog comparator 0 positive input 0 pin.

Group	Pin Name	GPIO	MFP	Type	Description
	ACMP0_WLAT	PA.5	MFP30	I	Analog comparator 0 window latch input pin
	ACMP0_WLAT	PB.7	MFP30	I	Analog comparator 0 window latch input pin
ACMP1	ACMP1_N0	PA.1	MFP1	A	Analog comparator 1 negative input 0 pin.
	ACMP1_N1	PC.1	MFP1	A	Analog comparator 1 negative input 1 pin.
	ACMP1_N2	PC.2	MFP1	A	Analog comparator 1 negative input 2 pin.
	ACMP1_N3	PC.3	MFP1	A	Analog comparator 1 negative input 3 pin.
	ACMP1_O	PA.5	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PA.4	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PC.5	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PC.4	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PC.3	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PC.6	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PC.7	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PB.7	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PB.6	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PB.5	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PB.4	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PC.2	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PC.1	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PC.0	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PA.2	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PA.1	MFP22	O	Analog comparator 1 output pin.
	ACMP1_O	PA.0	MFP22	O	Analog comparator 1 output pin.
	ACMP1_P0	PC.0	MFP1	A	Analog comparator 1 positive input 0 pin.
	ACMP1_WLAT	PA.4	MFP30	I	Analog comparator 1 window latch input pin
	ACMP1_WLAT	PB.6	MFP30	I	Analog comparator 1 window latch input pin
ADC0	ADC0_CH0	PA.0	MFP1	A	ADC0 channel 0 analog input.
	ADC0_CH1	PA.1	MFP1	A	ADC0 channel 1 analog input.
	ADC0_CH10	PC.7	MFP1	A	ADC0 channel 10 analog input.
	ADC0_CH11	PC.6	MFP1	A	ADC0 channel 11 analog input.
	ADC0_CH12	PC.3	MFP1	A	ADC0 channel 12 analog input.
	ADC0_CH13	PC.4	MFP1	A	ADC0 channel 13 analog input.
	ADC0_CH14	PC.5	MFP1	A	ADC0 channel 14 analog input.
	ADC0_CH15	PA.4	MFP1	A	ADC0 channel 15 analog input.

Group	Pin Name	GPIO	MFP	Type	Description
	ADC0_CH16	PA.5	MFP1	A	ADC0 channel 16 analog input.
	ADC0_CH2	PA.2	MFP1	A	ADC0 channel 2 analog input.
	ADC0_CH3	PC.0	MFP1	A	ADC0 channel 3 analog input.
	ADC0_CH4	PC.1	MFP1	A	ADC0 channel 4 analog input.
	ADC0_CH5	PC.2	MFP1	A	ADC0 channel 5 analog input.
	ADC0_CH6	PB.4	MFP1	A	ADC0 channel 6 analog input.
	ADC0_CH7	PB.5	MFP1	A	ADC0 channel 7 analog input.
	ADC0_CH8	PB.6	MFP1	A	ADC0 channel 8 analog input.
	ADC0_CH9	PB.7	MFP1	A	ADC0 channel 9 analog input.
	ADC0_ST	PA.5	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PA.4	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PA.3	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PC.5	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PC.4	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PC.3	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PC.6	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PC.7	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PB.7	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PB.6	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PB.5	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PB.4	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PC.2	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PC.1	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PC.0	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PA.2	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PA.1	MFP23	I	ADC0 external trigger input pin.
	ADC0_ST	PA.0	MFP23	I	ADC0 external trigger input pin.
CAN0	CAN0_RXD	PA.5	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PA.4	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PA.3	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PC.5	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PC.4	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PC.3	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PD.3	MFP3	I	CAN0 bus receiver input.

Group	Pin Name	GPIO	MFP	Type	Description
	CAN0_RXD	PC.6	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PC.7	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PB.7	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PB.6	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PB.5	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PD.7	MFP3	I	CAN0 bus receiver input.
	CAN0_RXD	PB.4	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PC.2	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PC.1	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PC.0	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PA.2	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PA.1	MFP20	I	CAN0 bus receiver input.
	CAN0_RXD	PA.0	MFP20	I	CAN0 bus receiver input.
	CAN0_TXD	PA.5	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PA.4	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PC.5	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PC.4	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PC.3	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PD.2	MFP3	O	CAN0 bus transmitter output.
	CAN0_TXD	PC.6	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PC.7	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PB.7	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PB.6	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PB.5	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PD.6	MFP3	O	CAN0 bus transmitter output.
	CAN0_TXD	PB.4	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PC.2	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PC.1	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PC.0	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PA.2	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PA.1	MFP19	O	CAN0 bus transmitter output.
	CAN0_TXD	PA.0	MFP19	O	CAN0 bus transmitter output.
CLKO	CLKO	PA.5	MFP4	O	Clock Out
	CLKO	PA.4	MFP4	O	Clock Out

Group	Pin Name	GPIO	MFP	Type	Description
	CLKO	PC.5	MFP4	O	Clock Out
	CLKO	PC.4	MFP4	O	Clock Out
	CLKO	PC.3	MFP4	O	Clock Out
	CLKO	PC.6	MFP4	O	Clock Out
	CLKO	PC.7	MFP4	O	Clock Out
	CLKO	PB.7	MFP4	O	Clock Out
	CLKO	PB.6	MFP4	O	Clock Out
	CLKO	PB.5	MFP4	O	Clock Out
	CLKO	PB.4	MFP4	O	Clock Out
	CLKO	PC.2	MFP4	O	Clock Out
	CLKO	PC.1	MFP4	O	Clock Out
	CLKO	PC.0	MFP4	O	Clock Out
	CLKO	PA.2	MFP4	O	Clock Out
	CLKO	PA.1	MFP4	O	Clock Out
	CLKO	PA.0	MFP4	O	Clock Out
DAC0	DAC0_OUT	PA.0	MFP1	A	DAC0 channel analog output.
ICE	ICE_CLK	PA.1	MFP2	I	Serial wired debugger clock pin. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_CLK pin.
	ICE_DAT	PA.0	MFP2	I/O	Serial wired debugger data pin. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_DAT pin.
INT0	INT0	PA.3	MFP30	I	External interrupt 0 input pin.
	INT0	PB.5	MFP30	I	External interrupt 0 input pin.
INT1	INT1	PC.5	MFP30	I	External interrupt 1 input pin.
INT2	INT2	PC.4	MFP30	I	External interrupt 2 input pin.
INT3	INT3	PC.3	MFP30	I	External interrupt 3 input pin.
INT4	INT4	PC.6	MFP30	I	External interrupt 4 input pin.
INT5	INT5	PC.7	MFP30	I	External interrupt 5 input pin.
PWM0	PWM0_BRAKE0	PC.0	MFP30	I	PWM0 Brake 0 input pin.
	PWM0_BRAKE0	PA.1	MFP30	I	PWM0 Brake 0 input pin.
	PWM0_BRAKE1	PA.2	MFP30	I	PWM0 Brake 1 input pin.
	PWM0_BRAKE1	PA.0	MFP30	I	PWM0 Brake 1 input pin.
	PWM0_CH0	PA.5	MFP5	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH0	PA.3	MFP5	I	PWM0 channel 0 capture input.
	PWM0_CH0	PC.4	MFP5	I/O	PWM0 channel 0 output/capture input.

Group	Pin Name	GPIO	MFP	Type	Description
	PWM0_CH0	PD.2	MFP2	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH0	PC.6	MFP5	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH0	PB.7	MFP5	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH0	PB.5	MFP5	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH0	PD.4	MFP2	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH0	PC.2	MFP5	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH0	PC.0	MFP5	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH0	PA.1	MFP5	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH1	PA.4	MFP5	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH1	PC.5	MFP5	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH1	PC.3	MFP5	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH1	PD.3	MFP2	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH1	PC.7	MFP5	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH1	PB.6	MFP5	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH1	PD.5	MFP2	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH1	PB.4	MFP5	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH1	PC.1	MFP5	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH1	PA.2	MFP5	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH1	PA.0	MFP5	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH2	PA.5	MFP6	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH2	PA.3	MFP6	I	PWM0 channel 2 capture input.
	PWM0_CH2	PC.4	MFP6	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH2	PC.6	MFP6	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH2	PB.7	MFP6	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH2	PB.5	MFP6	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH2	PD.6	MFP2	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH2	PC.2	MFP6	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH2	PC.0	MFP6	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH2	PA.1	MFP6	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH3	PA.4	MFP6	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH3	PC.5	MFP6	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH3	PC.3	MFP6	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH3	PC.7	MFP6	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH3	PB.6	MFP6	I/O	PWM0 channel 3 output/capture input.

Group	Pin Name	GPIO	MFP	Type	Description
	PWM0_CH3	PD.7	MFP2	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH3	PB.4	MFP6	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH3	PC.1	MFP6	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH3	PA.2	MFP6	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH3	PA.0	MFP6	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH4	PA.5	MFP7	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH4	PA.3	MFP7	I	PWM0 channel 4 capture input.
	PWM0_CH4	PC.4	MFP7	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH4	PD.0	MFP2	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH4	PC.6	MFP7	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH4	PB.7	MFP7	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH4	PB.5	MFP7	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH4	PC.2	MFP7	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH4	PC.0	MFP7	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH4	PA.1	MFP7	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH5	PA.4	MFP7	I/O	PWM0 channel 5 output/capture input.
	PWM0_CH5	PC.5	MFP7	I/O	PWM0 channel 5 output/capture input.
	PWM0_CH5	PC.3	MFP7	I/O	PWM0 channel 5 output/capture input.
	PWM0_CH5	PD.1	MFP2	I/O	PWM0 channel 5 output/capture input.
	PWM0_CH5	PC.7	MFP7	I/O	PWM0 channel 5 output/capture input.
	PWM0_CH5	PB.6	MFP7	I/O	PWM0 channel 5 output/capture input.
	PWM0_CH5	PB.4	MFP7	I/O	PWM0 channel 5 output/capture input.
	PWM0_CH5	PC.1	MFP7	I/O	PWM0 channel 5 output/capture input.
	PWM0_CH5	PA.2	MFP7	I/O	PWM0 channel 5 output/capture input.
	PWM0_CH5	PA.0	MFP7	I/O	PWM0 channel 5 output/capture input.
TM0	TM0	PA.5	MFP24	I/O	Timer0 event counter input/toggle output pin.
	TM0	PA.3	MFP24	I	Timer0 event counter input pin.
	TM0	PC.4	MFP24	I/O	Timer0 event counter input/toggle output pin.
	TM0	PD.0	MFP5	I/O	Timer0 event counter input/toggle output pin.
	TM0	PC.6	MFP24	I/O	Timer0 event counter input/toggle output pin.
	TM0	PB.7	MFP24	I/O	Timer0 event counter input/toggle output pin.
	TM0	PB.5	MFP24	I/O	Timer0 event counter input/toggle output pin.
	TM0	PD.4	MFP5	I/O	Timer0 event counter input/toggle output pin.
	TM0	PC.2	MFP24	I/O	Timer0 event counter input/toggle output pin.

Group	Pin Name	GPIO	MFP	Type	Description
	TM0	PC.0	MFP24	I/O	Timer0 event counter input/toggle output pin.
	TM0	PA.1	MFP24	I/O	Timer0 event counter input/toggle output pin.
TM1	TM1	PA.4	MFP24	I/O	Timer1 event counter input/toggle output pin.
	TM1	PC.5	MFP24	I/O	Timer1 event counter input/toggle output pin.
	TM1	PC.3	MFP24	I/O	Timer1 event counter input/toggle output pin.
	TM1	PD.1	MFP5	I/O	Timer1 event counter input/toggle output pin.
	TM1	PC.7	MFP24	I/O	Timer1 event counter input/toggle output pin.
	TM1	PB.6	MFP24	I/O	Timer1 event counter input/toggle output pin.
	TM1	PD.5	MFP5	I/O	Timer1 event counter input/toggle output pin.
	TM1	PB.4	MFP24	I/O	Timer1 event counter input/toggle output pin.
	TM1	PC.1	MFP24	I/O	Timer1 event counter input/toggle output pin.
	TM1	PA.2	MFP24	I/O	Timer1 event counter input/toggle output pin.
	TM1	PA.0	MFP24	I/O	Timer1 event counter input/toggle output pin.
TM2	TM2	PA.5	MFP25	I/O	Timer2 event counter input/toggle output pin.
	TM2	PA.3	MFP25	I	Timer2 event counter input pin.
	TM2	PC.4	MFP25	I/O	Timer2 event counter input/toggle output pin.
	TM2	PD.2	MFP5	I/O	Timer2 event counter input/toggle output pin.
	TM2	PC.6	MFP25	I/O	Timer2 event counter input/toggle output pin.
	TM2	PB.7	MFP25	I/O	Timer2 event counter input/toggle output pin.
	TM2	PB.5	MFP25	I/O	Timer2 event counter input/toggle output pin.
	TM2	PD.6	MFP5	I/O	Timer2 event counter input/toggle output pin.
	TM2	PC.2	MFP25	I/O	Timer2 event counter input/toggle output pin.
	TM2	PC.0	MFP25	I/O	Timer2 event counter input/toggle output pin.
	TM2	PA.1	MFP25	I/O	Timer2 event counter input/toggle output pin.
TM3	TM3	PA.4	MFP25	I/O	Timer3 event counter input/toggle output pin.
	TM3	PC.5	MFP25	I/O	Timer3 event counter input/toggle output pin.
	TM3	PC.3	MFP25	I/O	Timer3 event counter input/toggle output pin.
	TM3	PD.3	MFP5	I/O	Timer3 event counter input/toggle output pin.
	TM3	PC.7	MFP25	I/O	Timer3 event counter input/toggle output pin.
	TM3	PB.6	MFP25	I/O	Timer3 event counter input/toggle output pin.
	TM3	PD.7	MFP5	I/O	Timer3 event counter input/toggle output pin.
	TM3	PB.4	MFP25	I/O	Timer3 event counter input/toggle output pin.
	TM3	PC.1	MFP25	I/O	Timer3 event counter input/toggle output pin.
	TM3	PA.2	MFP25	I/O	Timer3 event counter input/toggle output pin.

Group	Pin Name	GPIO	MFP	Type	Description
	TM3	PA.0	MFP25	I/O	Timer3 event counter input/toggle output pin.
TM0	TM0_EXT	PA.5	MFP26	I/O	Timer0 external capture input/toggle output pin.
	TM0_EXT	PA.3	MFP26	I	Timer0 external capture input pin.
	TM0_EXT	PC.4	MFP26	I/O	Timer0 external capture input/toggle output pin.
	TM0_EXT	PC.6	MFP26	I/O	Timer0 external capture input/toggle output pin.
	TM0_EXT	PB.7	MFP26	I/O	Timer0 external capture input/toggle output pin.
	TM0_EXT	PB.5	MFP26	I/O	Timer0 external capture input/toggle output pin.
	TM0_EXT	PC.2	MFP26	I/O	Timer0 external capture input/toggle output pin.
	TM0_EXT	PC.0	MFP26	I/O	Timer0 external capture input/toggle output pin.
	TM0_EXT	PA.1	MFP26	I/O	Timer0 external capture input/toggle output pin.
TM1	TM1_EXT	PA.4	MFP26	I/O	Timer1 external capture input/toggle output pin.
	TM1_EXT	PC.5	MFP26	I/O	Timer1 external capture input/toggle output pin.
	TM1_EXT	PC.3	MFP26	I/O	Timer1 external capture input/toggle output pin.
	TM1_EXT	PC.7	MFP26	I/O	Timer1 external capture input/toggle output pin.
	TM1_EXT	PB.6	MFP26	I/O	Timer1 external capture input/toggle output pin.
	TM1_EXT	PB.4	MFP26	I/O	Timer1 external capture input/toggle output pin.
	TM1_EXT	PC.1	MFP26	I/O	Timer1 external capture input/toggle output pin.
	TM1_EXT	PA.2	MFP26	I/O	Timer1 external capture input/toggle output pin.
	TM1_EXT	PA.0	MFP26	I/O	Timer1 external capture input/toggle output pin.
TM2	TM2_EXT	PA.5	MFP27	I/O	Timer2 external capture input/toggle output pin.
	TM2_EXT	PA.3	MFP27	I	Timer2 external capture input pin.
	TM2_EXT	PC.4	MFP27	I/O	Timer2 external capture input/toggle output pin.
	TM2_EXT	PC.6	MFP27	I/O	Timer2 external capture input/toggle output pin.
	TM2_EXT	PB.7	MFP27	I/O	Timer2 external capture input/toggle output pin.
	TM2_EXT	PB.5	MFP27	I/O	Timer2 external capture input/toggle output pin.
	TM2_EXT	PC.2	MFP27	I/O	Timer2 external capture input/toggle output pin.
	TM2_EXT	PC.0	MFP27	I/O	Timer2 external capture input/toggle output pin.
	TM2_EXT	PA.1	MFP27	I/O	Timer2 external capture input/toggle output pin.
TM3	TM3_EXT	PA.4	MFP27	I/O	Timer3 external capture input/toggle output pin.
	TM3_EXT	PC.5	MFP27	I/O	Timer3 external capture input/toggle output pin.
	TM3_EXT	PC.3	MFP27	I/O	Timer3 external capture input/toggle output pin.
	TM3_EXT	PC.7	MFP27	I/O	Timer3 external capture input/toggle output pin.
	TM3_EXT	PB.6	MFP27	I/O	Timer3 external capture input/toggle output pin.
	TM3_EXT	PB.4	MFP27	I/O	Timer3 external capture input/toggle output pin.

Group	Pin Name	GPIO	MFP	Type	Description
	TM3_EXT	PC.1	MFP27	I/O	Timer3 external capture input/toggle output pin.
	TM3_EXT	PA.2	MFP27	I/O	Timer3 external capture input/toggle output pin.
	TM3_EXT	PA.0	MFP27	I/O	Timer3 external capture input/toggle output pin.
UART0	UART0_RXD	PA.5	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PA.4	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PA.3	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PC.5	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PC.4	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PC.3	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PD.1	MFP3	I	UART0 data receiver input pin.
	UART0_RXD	PC.6	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PC.7	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PB.7	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PB.6	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PB.5	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PD.5	MFP3	I	UART0 data receiver input pin.
	UART0_RXD	PB.4	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PC.2	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PC.1	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PC.0	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PA.2	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PA.1	MFP9	I	UART0 data receiver input pin.
	UART0_RXD	PA.0	MFP9	I	UART0 data receiver input pin.
	UART0_TXD	PA.5	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PA.4	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PC.5	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PC.4	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PC.3	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PD.0	MFP3	O	UART0 data transmitter output pin.
	UART0_TXD	PC.6	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PC.7	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PB.7	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PB.6	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PB.5	MFP8	O	UART0 data transmitter output pin.

Group	Pin Name	GPIO	MFP	Type	Description
	UART0_TXD	PD.4	MFP3	O	UART0 data transmitter output pin.
	UART0_TXD	PB.4	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PC.2	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PC.1	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PC.0	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PA.2	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PA.1	MFP8	O	UART0 data transmitter output pin.
	UART0_TXD	PA.0	MFP8	O	UART0 data transmitter output pin.
	UART0_nCTS	PA.3	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PC.5	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PC.4	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PC.3	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PC.6	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PC.7	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PB.7	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PB.6	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PB.5	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PB.4	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PC.2	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PC.1	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PC.0	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PA.2	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PA.1	MFP3	I	UART0 clear to Send input pin.
	UART0_nCTS	PA.0	MFP3	I	UART0 clear to Send input pin.
	UART0_nRTS	PA.5	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PA.4	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PC.3	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PC.6	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PC.7	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PB.7	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PB.6	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PB.5	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PB.4	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PC.2	MFP2	O	UART0 request to Send output pin.

Group	Pin Name	GPIO	MFP	Type	Description
	UART0_nRTS	PC.1	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PC.0	MFP2	O	UART0 request to Send output pin.
	UART0_nRTS	PA.2	MFP2	O	UART0 request to Send output pin.
UART1	UART1_RXD	PA.5	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PA.4	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PA.3	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PC.5	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PC.4	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PC.3	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PC.6	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PC.7	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PB.7	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PB.6	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PB.5	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PB.4	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PC.2	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PC.1	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PC.0	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PA.2	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PA.1	MFP29	I	UART1 data receiver input pin.
	UART1_RXD	PA.0	MFP29	I	UART1 data receiver input pin.
	UART1_TXD	PA.5	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PA.4	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PC.5	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PC.4	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PC.3	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PC.6	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PC.7	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PB.7	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PB.6	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PB.5	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PB.4	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PC.2	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PC.1	MFP28	O	UART1 data transmitter output pin.

Group	Pin Name	GPIO	MFP	Type	Description
	UART1_TXD	PC.0	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PA.2	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PA.1	MFP28	O	UART1 data transmitter output pin.
	UART1_TXD	PA.0	MFP28	O	UART1 data transmitter output pin.
	UART1_nCTS	PD.2	MFP6	I	UART1 clear to Send input pin.
	UART1_nCTS	PD.6	MFP6	I	UART1 clear to Send input pin.
	UART1_nCTS	PB.4	MFP30	I	UART1 clear to Send input pin.
	UART1_nRTS	PD.3	MFP6	O	UART1 request to Send output pin.
	UART1_nRTS	PD.7	MFP6	O	UART1 request to Send output pin.
	UART1_nRTS	PC.2	MFP30	O	UART1 request to Send output pin.
USCI0	USCI0_CLK	PA.5	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PA.4	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PA.3	MFP10	I	USCI0 clock pin.
	USCI0_CLK	PC.5	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PC.4	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PC.3	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PC.6	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PC.7	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PB.7	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PB.6	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PB.5	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PD.4	MFP4	I/O	USCI0 clock pin.
	USCI0_CLK	PB.4	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PC.2	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PC.1	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PC.0	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PA.2	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PA.1	MFP10	I/O	USCI0 clock pin.
	USCI0_CLK	PA.0	MFP10	I/O	USCI0 clock pin.
	USCI0_CTL0	PA.5	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PA.4	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PA.3	MFP13	I	USCI0 control 0 pin.
	USCI0_CTL0	PC.5	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PC.4	MFP13	I/O	USCI0 control 0 pin.

Group	Pin Name	GPIO	MFP	Type	Description
	USCI0_CTL0	PC.3	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PC.6	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PC.7	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PB.7	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PB.6	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PB.5	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PD.7	MFP4	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PB.4	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PC.2	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PC.1	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PC.0	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PA.2	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PA.1	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL0	PA.0	MFP13	I/O	USCI0 control 0 pin.
	USCI0_CTL1	PA.5	MFP14	I/O	USCI0 control 1 pin.
	USCI0_CTL1	PA.3	MFP14	I	USCI0 control 1 pin.
	USCI0_CTL1	PC.5	MFP14	I/O	USCI0 control 1 pin.
	USCI0_CTL1	PC.3	MFP14	I/O	USCI0 control 1 pin.
	USCI0_CTL1	PC.7	MFP14	I/O	USCI0 control 1 pin.
	USCI0_CTL1	PB.6	MFP14	I/O	USCI0 control 1 pin.
	USCI0_CTL1	PB.4	MFP14	I/O	USCI0 control 1 pin.
	USCI0_CTL1	PC.1	MFP14	I/O	USCI0 control 1 pin.
	USCI0_CTL1	PA.2	MFP14	I/O	USCI0 control 1 pin.
	USCI0_CTL1	PA.0	MFP14	I/O	USCI0 control 1 pin.
	USCI0_DAT0	PA.5	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PA.4	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PA.3	MFP11	I	USCI0 data 0 pin.
	USCI0_DAT0	PC.5	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PC.4	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PC.3	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PC.6	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PC.7	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PB.7	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PB.6	MFP11	I/O	USCI0 data 0 pin.

Group	Pin Name	GPIO	MFP	Type	Description
	USCI0_DAT0	PB.5	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PD.5	MFP4	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PB.4	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PC.2	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PC.1	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PC.0	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PA.2	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PA.1	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT0	PA.0	MFP11	I/O	USCI0 data 0 pin.
	USCI0_DAT1	PA.5	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PA.4	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PA.3	MFP12	I	USCI0 data 1 pin.
	USCI0_DAT1	PC.5	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PC.4	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PC.3	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PC.6	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PC.7	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PB.7	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PB.6	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PB.5	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PD.6	MFP4	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PB.4	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PC.2	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PC.1	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PC.0	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PA.2	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PA.1	MFP12	I/O	USCI0 data 1 pin.
	USCI0_DAT1	PA.0	MFP12	I/O	USCI0 data 1 pin.
USCI1	USCI1_CLK	PA.5	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PA.4	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PA.3	MFP15	I	USCI1 clock pin.
	USCI1_CLK	PC.5	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PC.4	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PC.3	MFP15	I/O	USCI1 clock pin.

Group	Pin Name	GPIO	MFP	Type	Description
	USCI1_CLK	PD.0	MFP4	I/O	USCI1 clock pin.
	USCI1_CLK	PC.6	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PC.7	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PB.7	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PB.6	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PB.5	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PB.4	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PC.2	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PC.1	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PC.0	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PA.2	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PA.1	MFP15	I/O	USCI1 clock pin.
	USCI1_CLK	PA.0	MFP15	I/O	USCI1 clock pin.
	USCI1_CTL0	PA.5	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PA.4	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PA.3	MFP18	I	USCI1 control 0 pin.
	USCI1_CTL0	PC.5	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PC.4	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PC.3	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PD.3	MFP4	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PC.6	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PC.7	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PB.7	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PB.6	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PB.5	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PB.4	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PC.2	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PC.1	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PC.0	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PA.2	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PA.1	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL0	PA.0	MFP18	I/O	USCI1 control 0 pin.
	USCI1_CTL1	PA.4	MFP14	I/O	USCI1 control 1 pin.
	USCI1_CTL1	PC.4	MFP14	I/O	USCI1 control 1 pin.

Group	Pin Name	GPIO	MFP	Type	Description
	USCI1_CTL1	PC.6	MFP14	I/O	USCI1 control 1 pin.
	USCI1_CTL1	PB.7	MFP14	I/O	USCI1 control 1 pin.
	USCI1_CTL1	PB.5	MFP14	I/O	USCI1 control 1 pin.
	USCI1_CTL1	PC.2	MFP14	I/O	USCI1 control 1 pin.
	USCI1_CTL1	PC.0	MFP14	I/O	USCI1 control 1 pin.
	USCI1_CTL1	PA.1	MFP14	I/O	USCI1 control 1 pin.
	USCI1_DAT0	PA.5	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PA.4	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PA.3	MFP16	I	USCI1 data 0 pin.
	USCI1_DAT0	PC.5	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PC.4	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PC.3	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PD.1	MFP4	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PC.6	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PC.7	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PB.7	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PB.6	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PB.5	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PB.4	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PC.2	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PC.1	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PC.0	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PA.2	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PA.1	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT0	PA.0	MFP16	I/O	USCI1 data 0 pin.
	USCI1_DAT1	PA.5	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PA.4	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PA.3	MFP17	I	USCI1 data 1 pin.
	USCI1_DAT1	PC.5	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PC.4	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PC.3	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PD.2	MFP4	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PC.6	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PC.7	MFP17	I/O	USCI1 data 1 pin.

Group	Pin Name	GPIO	MFP	Type	Description
	USCI1_DAT1	PB.7	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PB.6	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PB.5	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PB.4	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PC.2	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PC.1	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PC.0	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PA.2	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PA.1	MFP17	I/O	USCI1 data 1 pin.
	USCI1_DAT1	PA.0	MFP17	I/O	USCI1 data 1 pin.
V_{REF}	V_{REF}	PA.1	MFP1	A	ADC reference voltage input. **Note:** This pin needs to be connected with a 1uF capacitor.
X32	X32_IN	PC.5	MFP2	I	External 32.768 kHz crystal input pin.
	X32_OUT	PC.4	MFP2	O	External 32.768 kHz crystal output pin.
XT1	XT1_IN	PA.5	MFP3	I	External 4~24 MHz (high speed) crystal input pin.
	XT1_OUT	PA.4	MFP3	O	External 4~24 MHz (high speed) crystal output pin.
nRESET	nRESET	PA.3	MFP2	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.

4.2.3 M0A21/M0A23 Series Multi-function Summary Table Sorted by GPIO

	Pin Name	Type	MFP	Description
PA.0	PA.0	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH0	A	MFP1	ADC0 channel 0 analog input.
	DAC0_OUT	A	MFP1	DAC0 channel analog output.
	ACMP0_P0	A	MFP1	Analog comparator 0 positive input 0 pin.
	ICE_DAT	I/O	MFP2	Serial wired debugger data pin. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_DAT pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
	PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
	PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
	TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
	TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
	TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.

	Pin Name	Type	MFP	Description
	PWM0_BRAKE1	I	MFP30	PWM0 Brake 1 input pin.
PA.1	PA.1	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH1	A	MFP1	ADC0 channel 1 analog input.
	ACMP0_N0	A	MFP1	Analog comparator 0 negative input 0 pin.
	ACMP1_N0	A	MFP1	Analog comparator 1 negative input 0 pin.
	V_{REF}	A	MFP1	ADC reference voltage input. **Note:** This pin needs to be connected with a 1uF capacitor.
	ICE_CLK	I	MFP2	Serial wired debugger clock pin. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_CLK pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
	PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
	PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
	TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
	TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
	TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.

	Pin Name	Type	MFP	Description
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	PWM0_BRAKE0	I	MFP30	PWM0 Brake 0 input pin.
PA.2	PA.2	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH2	A	MFP1	ADC0 channel 2 analog input.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
	PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
	PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
	TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
	TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
	TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	PWM0_BRAKE1	I	MFP30	PWM0 Brake 1 input pin.

	Pin Name	Type	MFP	Description
PA.3	PA.3	I	MFP0	General purpose digital input pin.
	nRESET	I	MFP2	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.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	PWM0_CH0	I	MFP5	PWM0 channel 0 capture input.
	PWM0_CH2	I	MFP6	PWM0 channel 2 capture input.
	PWM0_CH4	I	MFP7	PWM0 channel 4 capture input.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I	MFP10	USCI0 clock pin.
	USCI0_DAT0	I	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I	MFP13	USCI0 control 0 pin.
	USCI0_CTL1	I	MFP14	USCI0 control 1 pin.
	USCI1_CLK	I	MFP15	USCI1 clock pin.
	USCI1_DAT0	I	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I	MFP18	USCI1 control 0 pin.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM0	I	MFP24	Timer0 event counter input pin.
	TM2	I	MFP25	Timer2 event counter input pin.
	TM0_EXT	I	MFP26	Timer0 external capture input pin.
	TM2_EXT	I	MFP27	Timer2 external capture input pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	INT0	I	MFP30	External interrupt 0 input pin.
PA.4	PA.4	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH15	A	MFP1	ADC0 channel 15 analog input.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	XT1_OUT	O	MFP3	External 4~24 MHz (high speed) crystal output pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
	PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
	PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.

	Pin Name	Type	MFP	Description
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
	TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
	TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
	TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	ACMP1_WLAT	I	MFP30	Analog comparator 1 window latch input pin
PA.5	PA.5	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH16	A	MFP1	ADC0 channel 16 analog input.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	XT1_IN	I	MFP3	External 4~24 MHz (high speed) crystal input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
	PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
	PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.

	Pin Name	Type	MFP	Description
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
	TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
	TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
	TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	ACMP0_WLAT	I	MFP30	Analog comparator 0 window latch input pin
PB.4	PB.4	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH6	A	MFP1	ADC0 channel 6 analog input.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
	PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
	PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.

	Pin Name	Type	MFP	Description
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
	TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
	TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
	TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	UART1_nCTS	I	MFP30	UART1 clear to Send input pin.
PB.5	PB.5	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH7	A	MFP1	ADC0 channel 7 analog input.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
	PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
	PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.

	Pin Name	Type	MFP	Description
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
	TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
	TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
	TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	INT0	I	MFP30	External interrupt 0 input pin.
PB.6	PB.6	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH8	A	MFP1	ADC0 channel 8 analog input.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
	PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
	PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.

	Pin Name	Type	MFP	Description
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
	TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
	TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
	TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	ACMP1_WLAT	I	MFP30	Analog comparator 1 window latch input pin
PB.7	PB.7	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH9	A	MFP1	ADC0 channel 9 analog input.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
	PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
	PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.

	Pin Name	Type	MFP	Description
	TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
	TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
	TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
	TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	ACMP0_WLAT	I	MFP30	Analog comparator 0 window latch input pin
PC.0	PC.0	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH3	A	MFP1	ADC0 channel 3 analog input.
	ACMP1_P0	A	MFP1	Analog comparator 1 positive input 0 pin.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
	PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
	PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
	TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.

	Pin Name	Type	MFP	Description
	TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
	TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	PWM0_BRAKE0	I	MFP30	PWM0 Brake 0 input pin.
PC.1	PC.1	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH4	A	MFP1	ADC0 channel 4 analog input.
	ACMP0_N1	A	MFP1	Analog comparator 0 negative input 1 pin.
	ACMP1_N1	A	MFP1	Analog comparator 1 negative input 1 pin.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
	PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
	PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
	TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
	TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.

	Pin Name	Type	MFP	Description
	TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
PC.2	PC.2	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH5	A	MFP1	ADC0 channel 5 analog input.
	ACMP0_N2	A	MFP1	Analog comparator 0 negative input 2 pin.
	ACMP1_N2	A	MFP1	Analog comparator 1 negative input 2 pin.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
	PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
	PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
	TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
	TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
	TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.

	Pin Name	Type	MFP	Description
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	UART1_nRTS	O	MFP30	UART1 request to Send output pin.
PC.3	PC.3	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH12	A	MFP1	ADC0 channel 12 analog input.
	ACMP0_N3	A	MFP1	Analog comparator 0 negative input 3 pin.
	ACMP1_N3	A	MFP1	Analog comparator 1 negative input 3 pin.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
	PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
	PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
	TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
	TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
	TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.

	Pin Name	Type	MFP	Description
	INT3	I	MFP30	External interrupt 3 input pin.
PC.4	PC.4	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH13	A	MFP1	ADC0 channel 13 analog input.
	X32_OUT	O	MFP2	External 32.768 kHz crystal output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
	PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
	PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
	TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
	TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
	TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	INT2	I	MFP30	External interrupt 2 input pin.
PC.5	PC.5	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH14	A	MFP1	ADC0 channel 14 analog input.

	Pin Name	Type	MFP	Description
	X32_IN	I	MFP2	External 32.768 kHz crystal input pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
	PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
	PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
	TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
	TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
	TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	INT1	I	MFP30	External interrupt 1 input pin.
PC.6	PC.6	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH11	A	MFP1	ADC0 channel 11 analog input.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out

	Pin Name	Type	MFP	Description
	PWM0_CH0	I/O	MFP5	PWM0 channel 0 output/capture input.
	PWM0_CH2	I/O	MFP6	PWM0 channel 2 output/capture input.
	PWM0_CH4	I/O	MFP7	PWM0 channel 4 output/capture input.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI1_CTL1	I/O	MFP14	USCI1 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM0	I/O	MFP24	Timer0 event counter input/toggle output pin.
	TM2	I/O	MFP25	Timer2 event counter input/toggle output pin.
	TM0_EXT	I/O	MFP26	Timer0 external capture input/toggle output pin.
	TM2_EXT	I/O	MFP27	Timer2 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	INT4	I	MFP30	External interrupt 4 input pin.
PC.7	PC.7	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH10	A	MFP1	ADC0 channel 10 analog input.
	UART0_nRTS	O	MFP2	UART0 request to Send output pin.
	UART0_nCTS	I	MFP3	UART0 clear to Send input pin.
	CLKO	O	MFP4	Clock Out
	PWM0_CH1	I/O	MFP5	PWM0 channel 1 output/capture input.
	PWM0_CH3	I/O	MFP6	PWM0 channel 3 output/capture input.
	PWM0_CH5	I/O	MFP7	PWM0 channel 5 output/capture input.

	Pin Name	Type	MFP	Description
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
	USCI0_CLK	I/O	MFP10	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP11	USCI0 data 0 pin.
	USCI0_DAT1	I/O	MFP12	USCI0 data 1 pin.
	USCI0_CTL0	I/O	MFP13	USCI0 control 0 pin.
	USCI0_CTL1	I/O	MFP14	USCI0 control 1 pin.
	USCI1_CLK	I/O	MFP15	USCI1 clock pin.
	USCI1_DAT0	I/O	MFP16	USCI1 data 0 pin.
	USCI1_DAT1	I/O	MFP17	USCI1 data 1 pin.
	USCI1_CTL0	I/O	MFP18	USCI1 control 0 pin.
	CAN0_TXD	O	MFP19	CAN0 bus transmitter output.
	CAN0_RXD	I	MFP20	CAN0 bus receiver input.
	ACMP0_O	O	MFP21	Analog comparator 0 output pin.
	ACMP1_O	O	MFP22	Analog comparator 1 output pin.
	ADC0_ST	I	MFP23	ADC0 external trigger input pin.
	TM1	I/O	MFP24	Timer1 event counter input/toggle output pin.
	TM3	I/O	MFP25	Timer3 event counter input/toggle output pin.
	TM1_EXT	I/O	MFP26	Timer1 external capture input/toggle output pin.
	TM3_EXT	I/O	MFP27	Timer3 external capture input/toggle output pin.
	UART1_TXD	O	MFP28	UART1 data transmitter output pin.
	UART1_RXD	I	MFP29	UART1 data receiver input pin.
	INT5	I	MFP30	External interrupt 5 input pin.
PD.0	PD.0	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH4	I/O	MFP2	PWM0 channel 4 output/capture input.
	UART0_TXD	O	MFP3	UART0 data transmitter output pin.
	USCI1_CLK	I/O	MFP4	USCI1 clock pin.
	TM0	I/O	MFP5	Timer0 event counter input/toggle output pin.
PD.1	PD.1	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH5	I/O	MFP2	PWM0 channel 5 output/capture input.
	UART0_RXD	I	MFP3	UART0 data receiver input pin.
	USCI1_DAT0	I/O	MFP4	USCI1 data 0 pin.
	TM1	I/O	MFP5	Timer1 event counter input/toggle output pin.
PD.2	PD.2	I/O	MFP0	General purpose digital I/O pin.

	Pin Name	Type	MFP	Description
	PWM0_CH0	I/O	MFP2	PWM0 channel 0 output/capture input.
	CAN0_TXD	O	MFP3	CAN0 bus transmitter output.
	USCI1_DAT1	I/O	MFP4	USCI1 data 1 pin.
	TM2	I/O	MFP5	Timer2 event counter input/toggle output pin.
	UART1_nCTS	I	MFP6	UART1 clear to Send input pin.
PD.3	PD.3	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH1	I/O	MFP2	PWM0 channel 1 output/capture input.
	CAN0_RXD	I	MFP3	CAN0 bus receiver input.
	USCI1_CTL0	I/O	MFP4	USCI1 control 0 pin.
	TM3	I/O	MFP5	Timer3 event counter input/toggle output pin.
	UART1_nRTS	O	MFP6	UART1 request to Send output pin.
PD.4	PD.4	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH0	I/O	MFP2	PWM0 channel 0 output/capture input.
	UART0_TXD	O	MFP3	UART0 data transmitter output pin.
	USCI0_CLK	I/O	MFP4	USCI0 clock pin.
	TM0	I/O	MFP5	Timer0 event counter input/toggle output pin.
PD.5	PD.5	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH1	I/O	MFP2	PWM0 channel 1 output/capture input.
	UART0_RXD	I	MFP3	UART0 data receiver input pin.
	USCI0_DAT0	I/O	MFP4	USCI0 data 0 pin.
	TM1	I/O	MFP5	Timer1 event counter input/toggle output pin.
PD.6	PD.6	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP2	PWM0 channel 2 output/capture input.
	CAN0_TXD	O	MFP3	CAN0 bus transmitter output.
	USCI0_DAT1	I/O	MFP4	USCI0 data 1 pin.
	TM2	I/O	MFP5	Timer2 event counter input/toggle output pin.
	UART1_nCTS	I	MFP6	UART1 clear to Send input pin.
PD.7	PD.7	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP2	PWM0 channel 3 output/capture input.
	CAN0_RXD	I	MFP3	CAN0 bus receiver input.
	USCI0_CTL0	I/O	MFP4	USCI0 control 0 pin.
	TM3	I/O	MFP5	Timer3 event counter input/toggle output pin.
	UART1_nRTS	O	MFP6	UART1 request to Send output pin.

5 BLOCK DIAGRAM

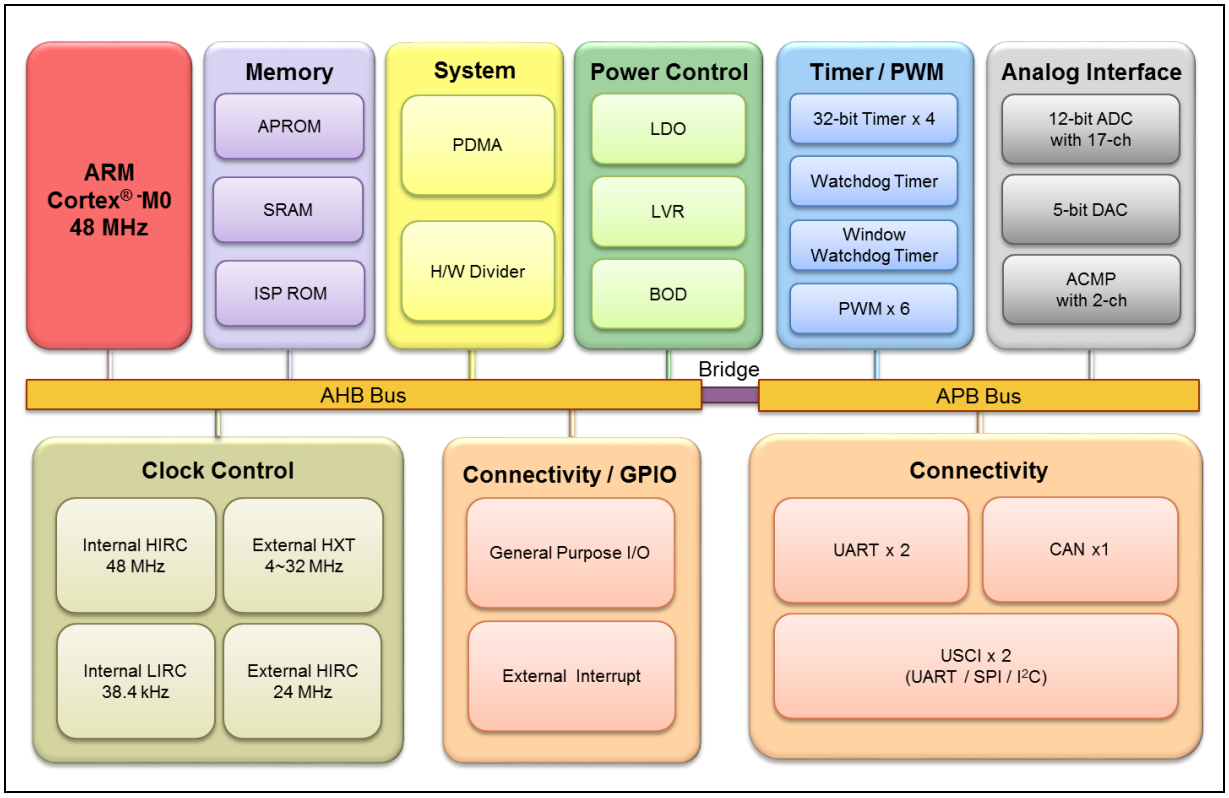


Figure 4.2-1 NuMicro® M0A21/M0A23 Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 Arm® Cortex®-M0 Core

The Cortex®-M0 core is a configurable, multistage, 32-bit RISC processor, which has an AMBA AHB-Lite interface and includes an NVIC component. It also has optional hardware debug functionality. The core can execute Thumb code and is compatible with other Cortex®-M profile processor. The profile supports two modes -Thread mode and Handler mode. Handler mode is entered as a result of an exception. An exception return can only be issued in Handler mode. Thread mode is entered on Reset, and can be entered as a result of an exception return. Figure 6.1-1 shows the functional block diagram of processor.

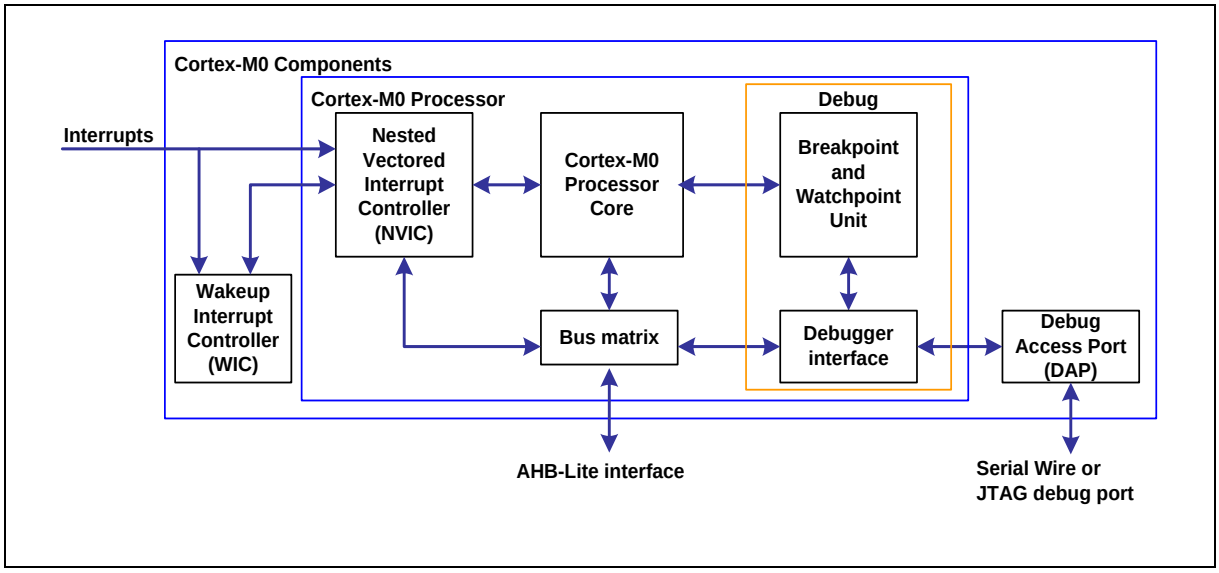


Figure 6.1-1 Functional Block Diagram

The implemented device provides:

- A low gate count processor:
  - Arm®6-M Thumb® instruction set
  - Thumb-2 technology
  - Arm®6-M compliant 24-bit SysTick timer
  - A 32-bit hardware multiplier
  - System interface supported with little-endian data accesses
  - Ability to have deterministic, fixed-latency, interrupt handling
  - Load/store-multiples and multicycle-multiplies that can be abandoned and restarted to facilitate rapid interrupt handling
  - C Application Binary Interface compliant exception model. This is the Armv6-M, C Application Binary Interface (C-ABI) compliant exception model that enables the use of pure C functions as interrupt handlers
  - Low Power Sleep mode entry using the Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or return from interrupt sleep-on-exit feature
- NVIC:
  - 32 external interrupt inputs, each with four levels of priority

  - Dedicated Non-maskable Interrupt (NMI) input
  - Supports for both level-sensitive and pulse-sensitive interrupt lines
  - Supports Wake-up Interrupt Controller (WIC) and, providing Ultra-low Power Sleep mode
- Debug support:
  - Four hardware breakpoints
  - Two watchpoints
  - Program Counter Sampling Register (PCSR) for non-intrusive code profiling
  - Single step and vector catch capabilities
- Bus interfaces:
  - Single 32-bit AMBA-3 AHB-Lite system interface that provides simple integration to all system peripherals and memory
  - Single 32-bit slave port that supports the DAP (Debug Access Port)

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®-M0 core Only by writing 1 to CPURST (SYS_IPRST0[1])
  - nRESET glitch filter time 32us

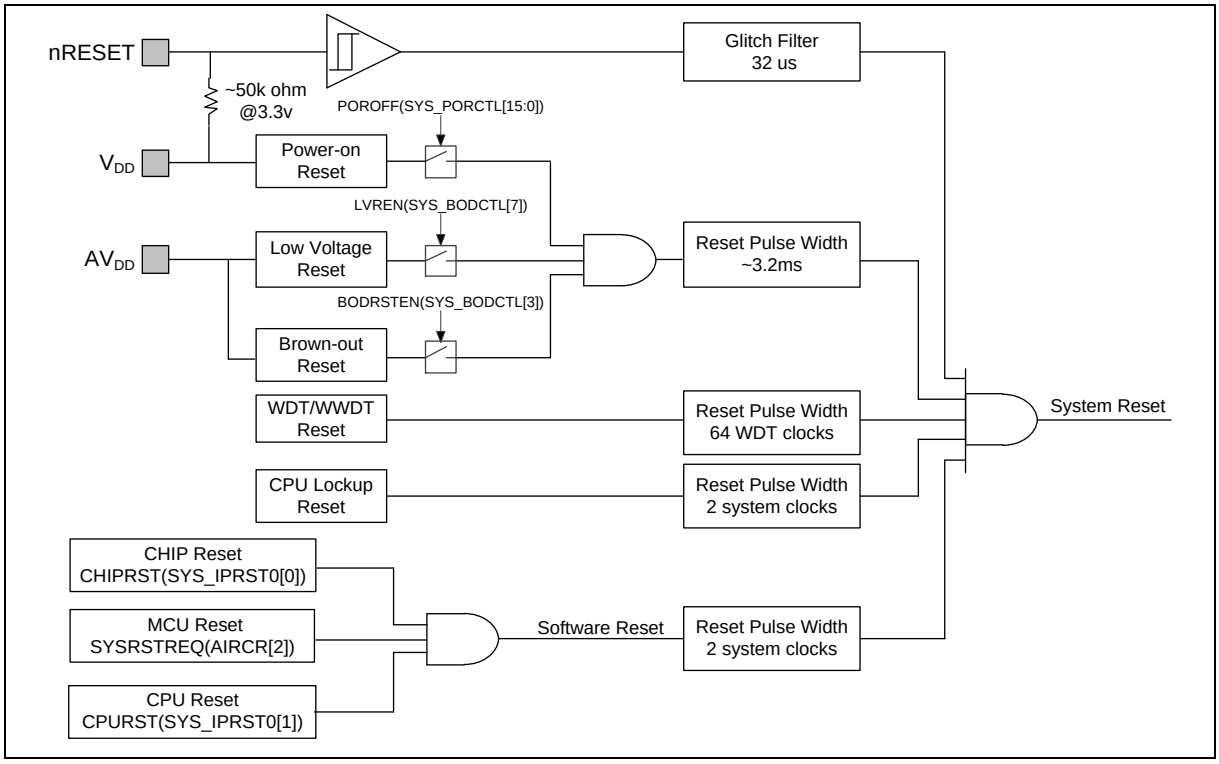


Figure 6.2-1 System Reset Sources

There are a total of 9 reset sources in the NuMicro® family. In general, CPU reset is used to reset Cortex®-M0 only; the other reset sources will reset Cortex®-M0 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[16])									
BODRSTEN (SYS_BODCTL[3])									
HXTEN (CLK_PWRCTL[0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	
LXTEN (CLK_PWRCTL[1])	0x0	-	-	-	-	-	-	-	-
WDTCKEN (CLK_APBCLK0[0])	0x1	-	0x1	-	-	-	0x1	-	-

HCLKSEL (CLK_CLKSEL0[2:0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-
WDTSEL (CLK_CLKSEL1[1:0])	0x3	0x3	-	-	-	-	-	-	-
HXTSTB (CLK_STATUS[0])	0x0	-	-	-	-	-	-	-	-
LXTSTB (CLK_STATUS[1])	0x0	-	-	-	-	-	-	-	-
HIRCSTB (CLK_STATUS[4])	0x0	-	-	-	-	-	-	-	-
CLKSFAIL (CLK_STATUS[7])	0x0	0x0	-	-	-	-	-	-	-
RSTEN (WDT_CTL[1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
WDTEN (WDT_CTL[7])									
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 (FMC_ISPCTL[1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
FMC_DFBA	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	-	Reload from CONFIG1	-	-
CBS (FMC_ISPSTS[2:1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
VECMAP (FMC_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	-	-
Other Peripheral Registers	Reset Value								-
FMC 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 0.2 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.

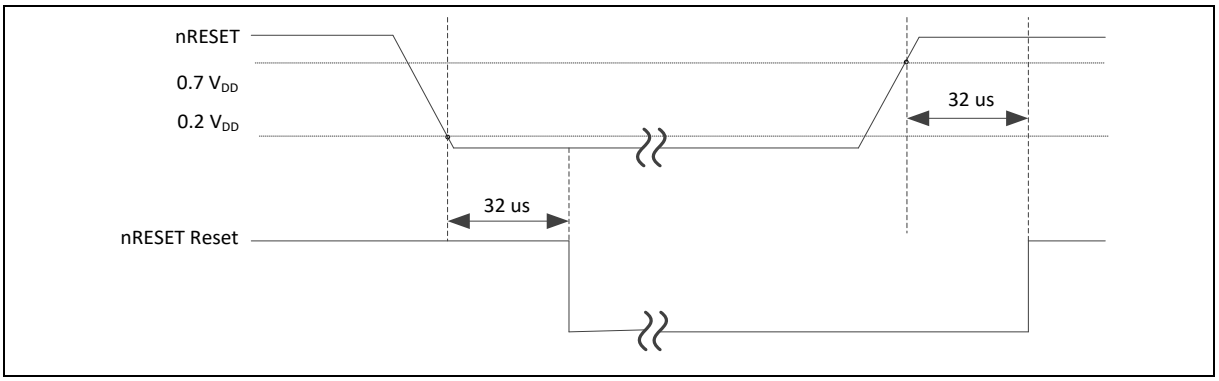


Figure 6.2-2 nRESET Reset Waveform

The special mode can enable nRESET pin function when system select other function for GPA.3, user can input special control signal for GPA.3 make system force enable nRESET pin.

Figure 6.2-3 shows the method of entry special mode.

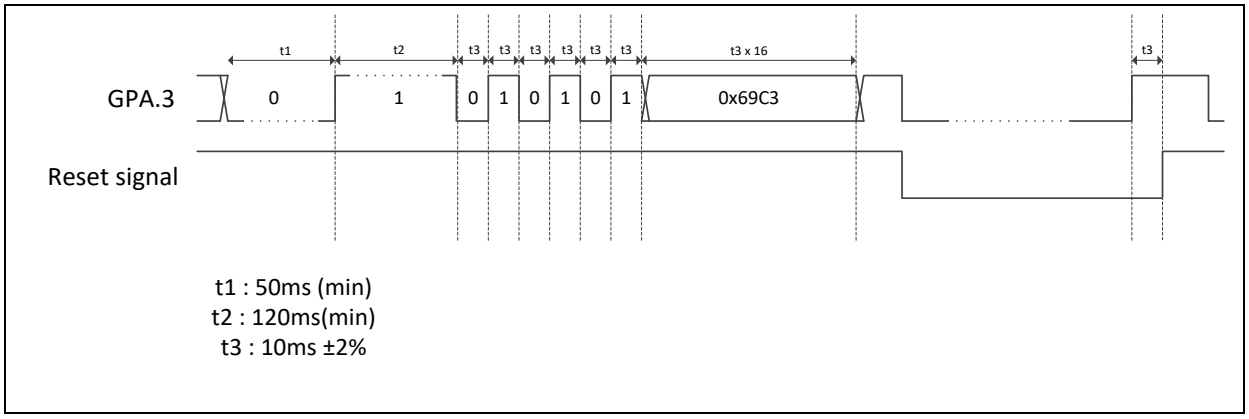


Figure 6.2-3 nRESET Reset Mode Enable Control Waveform

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 6.2-4 shows the power-on reset waveform.

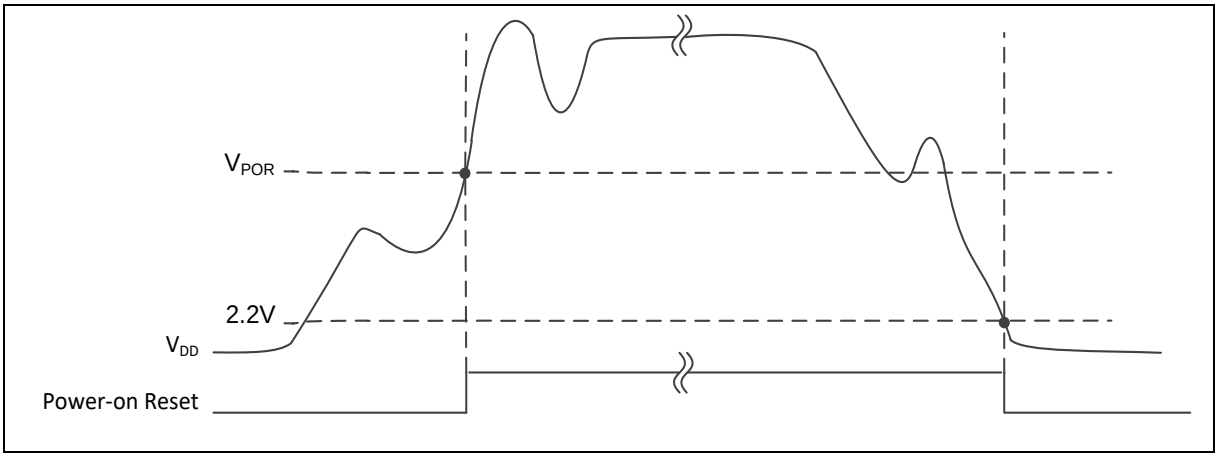


Figure 6.2-4 Power-on Reset (POR) Waveform

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 200us 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 6.2-5 shows the Low Voltage Reset waveform.

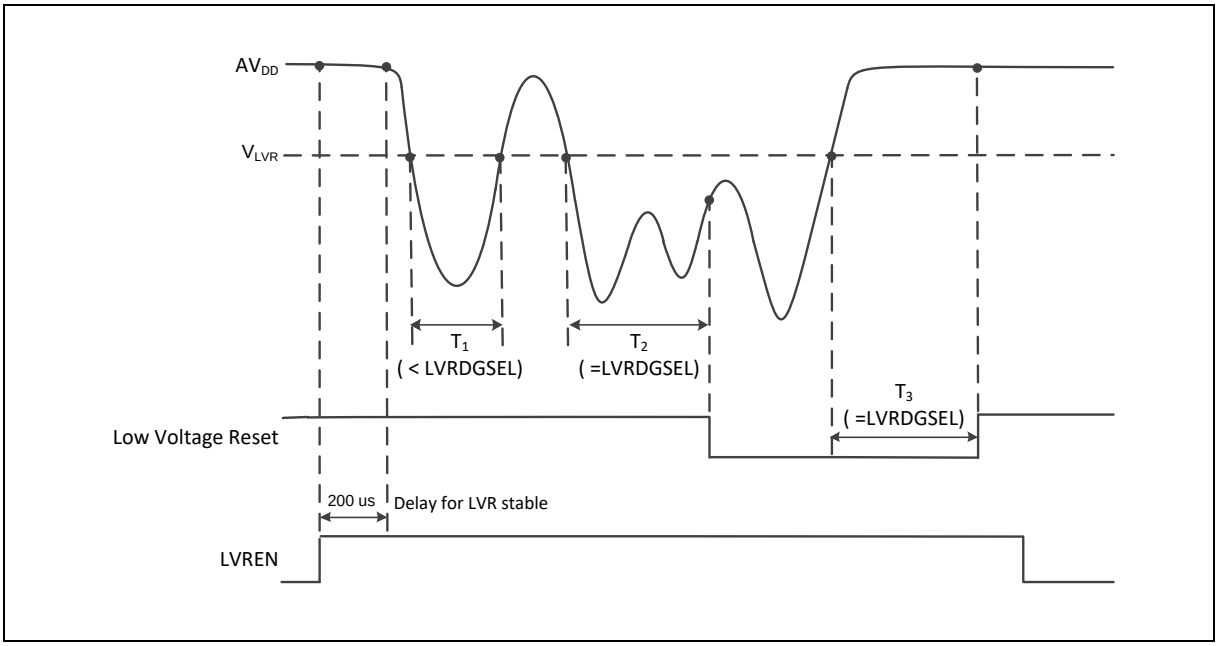


Figure 6.2-5 Low Voltage Reset (LVR) Waveform

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[17:16]) 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 Flash 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 6.2-6 shows the Brown-out Detector waveform.

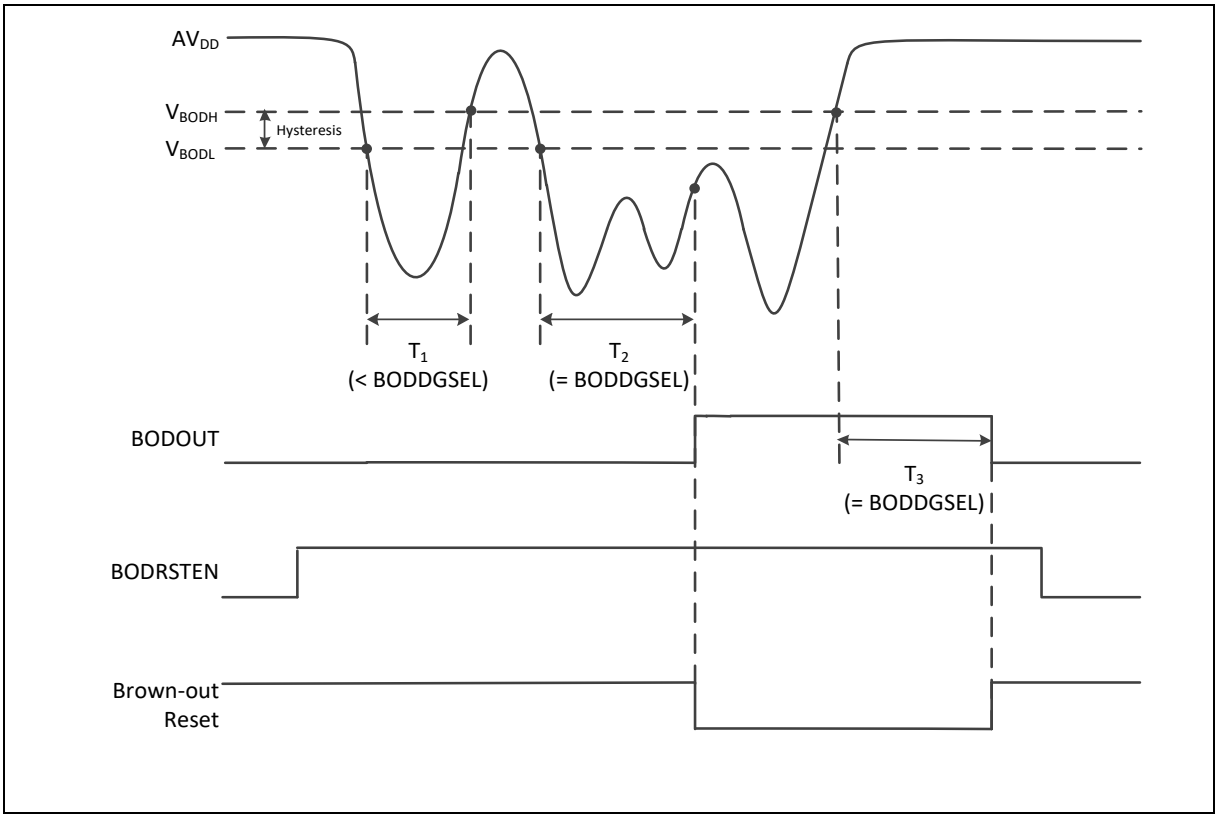


Figure 6.2-6 Brown-out Detector (BOD) Waveform

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®-M0 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(FMC_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(FMC_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 three segments:

- Analog power from V_{DD} and V_{SS} provides the power for analog components operation.
- Digital power from V_{DD} and V_{SS} supplies the power to the internal regulator which provides a fixed 1.8V power for digital operation and I/O pins.

The outputs of internal voltage regulators, LDO and V_{DD}, require an external capacitor which should be located close to the corresponding pin. Figure 6.2-7 shows the NuMicro® M0A21/M0A23 power distribution.

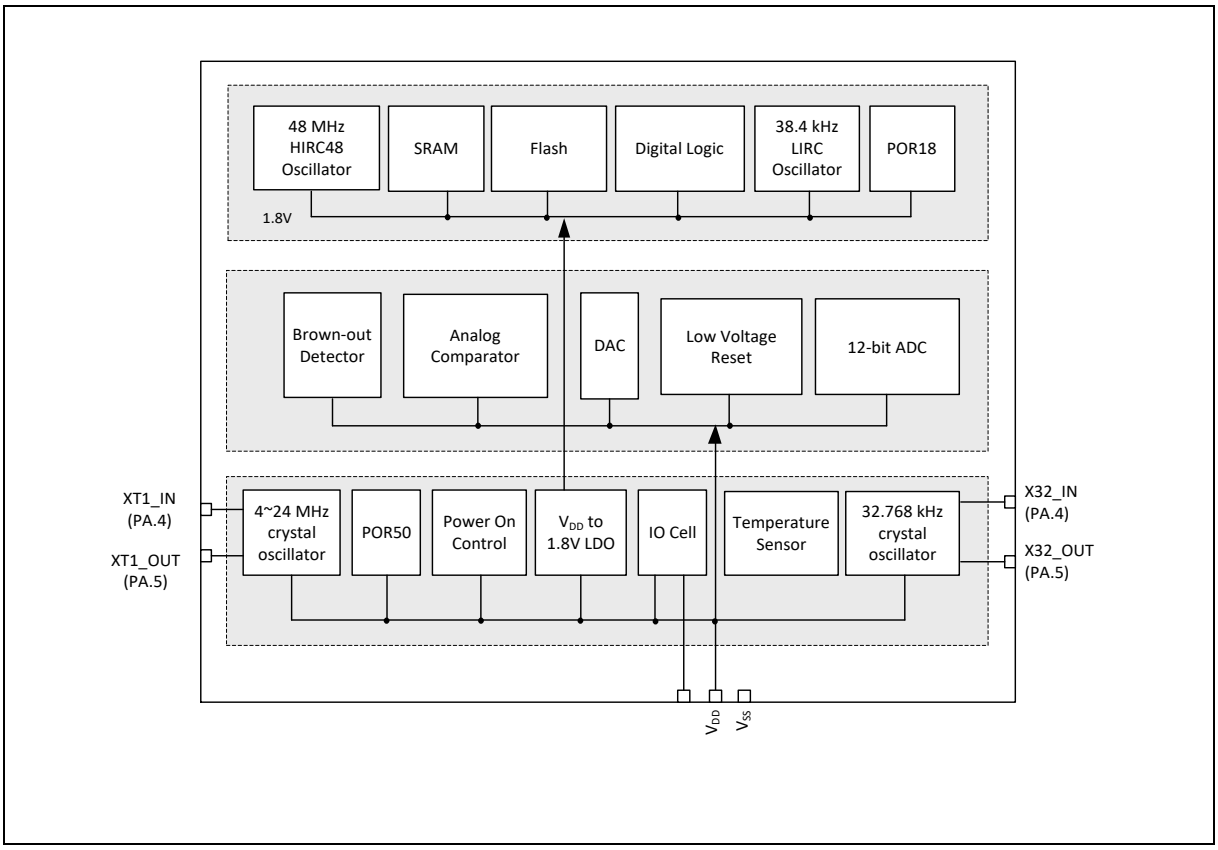


Figure 6.2-7 NuMicro® M0A21/M0A23 Power Distribution Diagram

6.2.4 Power Modes and Wake-up Sources

The M0A21/M0A23 series has power manager unit to support several operating modes for saving power. Table 6.2-2 lists all power mode in the M0A21/M0A23 series.

Mode	CPU Operating Maximum Speed(MHz)	Clock Disable
Normal mode	48	All clocks are disabled by control register.

Idle mode	CPU enter Sleep mode	Only CPU clock is disabled.
Power-down mode	CPU enters Power-down mode	Most clocks are disabled except LIRC/LXT, and only WDT/Timer/UART peripheral clocks still enable if their clock sources are selected as LIRC/LXT.

Table 6.2-2 Power Mode Table

There are different power mode entry settings and leaving condition for each power mode. Table 6.2-3 shows the entry setting for each power mode. When chip power-on, chip is running in normal mode. User can enter each mode by setting SLEEPDEEP (SCR[2]), PDEN (CLK_PWRCT:[7]) and execute WFI instruction.

Register/Instruction Mode	**SLEEPDEEP (SCR[2])**	**PDEN (CLK_PWRCTL[7])**	**CPU Run WFI Instruction**
Normal mode	0	0	NO
Idle mode (CPU enter Sleep mode)	0	0	YES
Power-down mode (CPU enters Deep Sleep mode)	1	1	YES

Table 6.2-3 Power Mode Difference Table

There are several wake-up sources in Idle mode and Power-down mode. Table 6.2-4 lists the available clocks for each power mode.

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	WDT, Timer, UART, BOD, GPIO, EINT, USCI, CAN and ACMP
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 6.2-4 Power Mode Difference Table

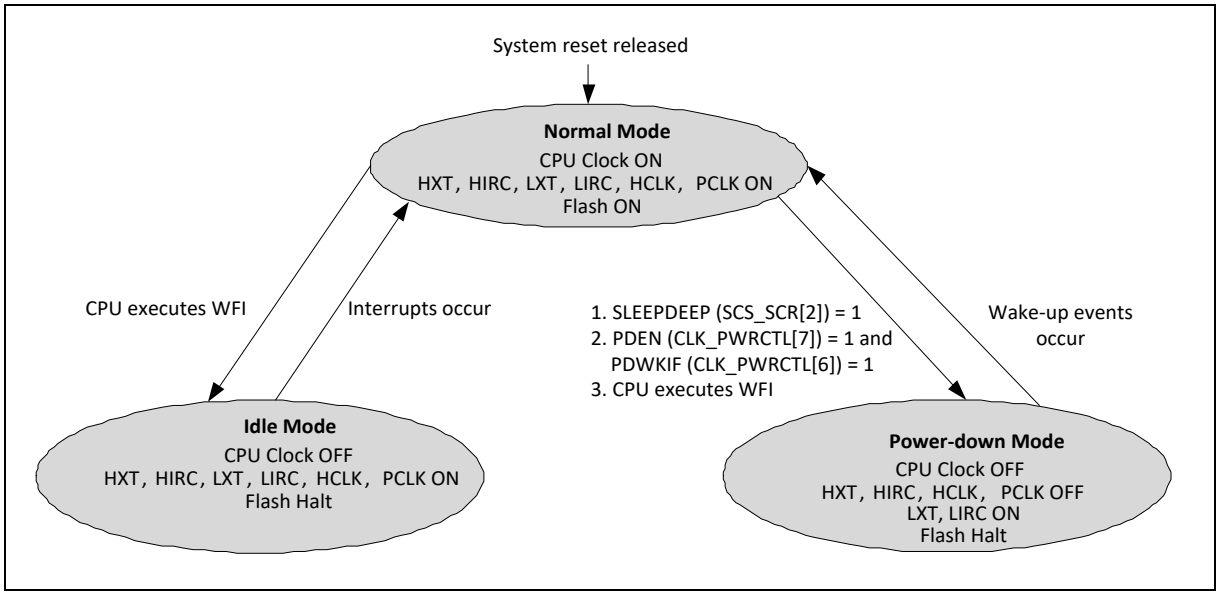


Figure 6.2-8 Power Mode State Machine

1. LXT (32768 Hz XTL) ON or OFF depends on SW setting in normal mode.
2. LIRC (38.4 kHz OSC) ON or OFF depends on S/W setting in normal mode.
3. If TIMER clock source is selected as LIRC/LXT and LIRC/LXT is on.
4. If WDT clock source is selected as LIRC and LIRC is on.
5. If UART clock source is selected as LXT and LXT is on.

	Normal Mode	Idle Mode	Power-Down Mode
HXT (4~24 MHz XTL)	ON	ON	Halt
HIRC48 (48 MHz OSC)	ON	ON	Halt
LXT (32768 Hz XTL)	ON	ON	ON/OFF[1]
LIRC (38.4 kHz OSC)	ON	ON	ON/OFF[2]
MLDO	ON	ON	OFF
ULDO	ON	ON	ON
CPU	ON	Halt	Halt
HCLK/PCLK	ON	ON	Halt
SRAM retention	ON	ON	ON
FLASH	ON	ON	Halt
GPIO	ON	ON	Halt
PDMA	ON	ON	Halt
TIMER	ON	ON	ON/OFF[3]
PWM	ON	ON	Halt
WDT	ON	ON	ON/OFF[4]
WWDT	ON	Halt	Halt
UART	ON	ON	ON/OFF[6]
USCI	ON	ON	Halt
ADC	ON	ON	Halt
ACMP	ON	ON	Halt

Table 6.2-5 Clocks in Power Modes

Wake-up sources in Power-down mode:

WDT, Timer, UART, USCI, BOD, GPIO, CAN, and ACMP.

After chip enters power down, the following wake-up sources can wake chip up to normal mode. Table 6.2-5 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 Source	Wake-Up Condition	System Can Enter Power-Down Mode Again Condition*
BOD	Brown-Out Detector Interrupt	After software writes 1 to clear (SYS_BODCTL[4]).
INT	External Interrupt	After software write 1 to clear the Px_INTSRC[n] bit.
GPIO	GPIO Interrupt	After software write 1 to clear the Px_INTSRC[n] bit.
TIMER	Timer Interrupt	After software writes 1 to clear TWKF (TIMERx_INTSTS[1]) and TIF (TIMERx_INTSTS[0]).
WDT	WDT Interrupt	After software writes 1 to clear WKF (WDT_CTL[5]) (Write Protect).

UART0/1	nCTS wake-up	After software writes 1 to clear CTSWKF (UARTx_WKSTS[0]).
	RX Data wake-up	After software writes 1 to clear DATWKF (UARTx_WKSTS[1]).
	Received FIFO Threshold Wake-up	After software writes 1 to clear RFRTWKF (UARTx_WKSTS[2]).
	RS-485 AAD Mode Wake-up	After software writes 1 to clear RS485WKF (UARTx_WKSTS[3]).
	Received FIFO Threshold Time-out Wake-up	After software writes 1 to clear TOUTWKF (UARTx_WKSTS[4]).
CAN	RX Data wake-up	After software writes 1 to clear WAKUP_STS (CAN_WU_STATUS[0]).
ACMP	Comparator Power-Down Wake-Up Interrupt	After software writes 1 to clear WKIF0 (ACMP_STATUS[8]) and WKIF1 (ACMP_STATUS[9]).

Table 6.2-6 Condition of Entering Power-down Mode Again

6.2.5 System Memory Map

The NuMicro® M0A21/M0A23 series provides 4G-byte addressing space. The memory locations assigned to each on-chip controllers are shown in Table 6.2-7. The detailed register definition, memory space, and programming will be described in the following sections for each on-chip peripheral. The M0A21/M0A23 series only supports little-endian data format.

Address Space	Token	Controllers
Flash and SRAM Memory Space		
0x0000_0000 – 0x0000_7FFF	FLASH_BA	FLASH Memory Space (32 Kbytes)
0x2000_0000 – 0x2000_0FFF	SRAM0_BA	SRAM Memory Space (4 Kbytes)
Peripheral Controllers Space (0x4000_0000 – 0x400F_FFFF)		
0x4000_0000 – 0x4000_01FF	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	PDMA_BA	Peripheral DMA Control Registers
0x4000_C000 – 0x4000_CFFF	FMC_BA	Flash Memory Control Registers
0x4001_4000 – 0x4001_7FFF	HDIV_BA	Hardware Divider Register
0x4003_1000 – 0x4003_1FFF	CRC_BA	CRC Generator Registers
APB Controllers Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4004_0000 – 0x4004_0FFF	WDT_BA	Watchdog Timer Control Registers
0x4004_3000 – 0x4004_3FFF	ADC_BA	Analog-Digital-Converter (ADC) Control Registers
0x4004_5000 – 0x4004_5FFF	ACMP01_BA	Analog Comparator 0/ 1 Control Registers
0x4004_7000 – 0x4004_7FFF	DAC0_BA	DAC 0 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	PWM0_BA	PWM0 Control Registers

0x4007_0000 – 0x4007_0FFF	UART0_BA	UART0 Control Registers
0x4007_1000 – 0x4007_1FFF	UART1_BA	UART1 Control Registers
0x400A_0000 – 0x400A_0FFF	CAN0_BA	CAN0 Control Registers
0x400D_0000 – 0x400D_0FFF	USCI0_BA	USCI0 Control Registers
0x400D_1000 – 0x400D_1FFF	USCI1_BA	USCI1 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 6.2-7 Address Space Assignments for On-Chip Controllers

6.2.6 SRAM Memory Orginization

The M0A21/M0A23 supports embedded SRAM with total 4 Kbytes size

- Supports total 4 Kbytes SRAM
- Supports byte / half word / word write
- Supports oversize response error

Table 6.2-9 shows the SRAM organization of M0A21/M0A23. The address between 0x2000_1000 to 0x3FFF_FFFF is illegal memory space and chip will enter hardfault if CPU accesses these illegal memory addresses.

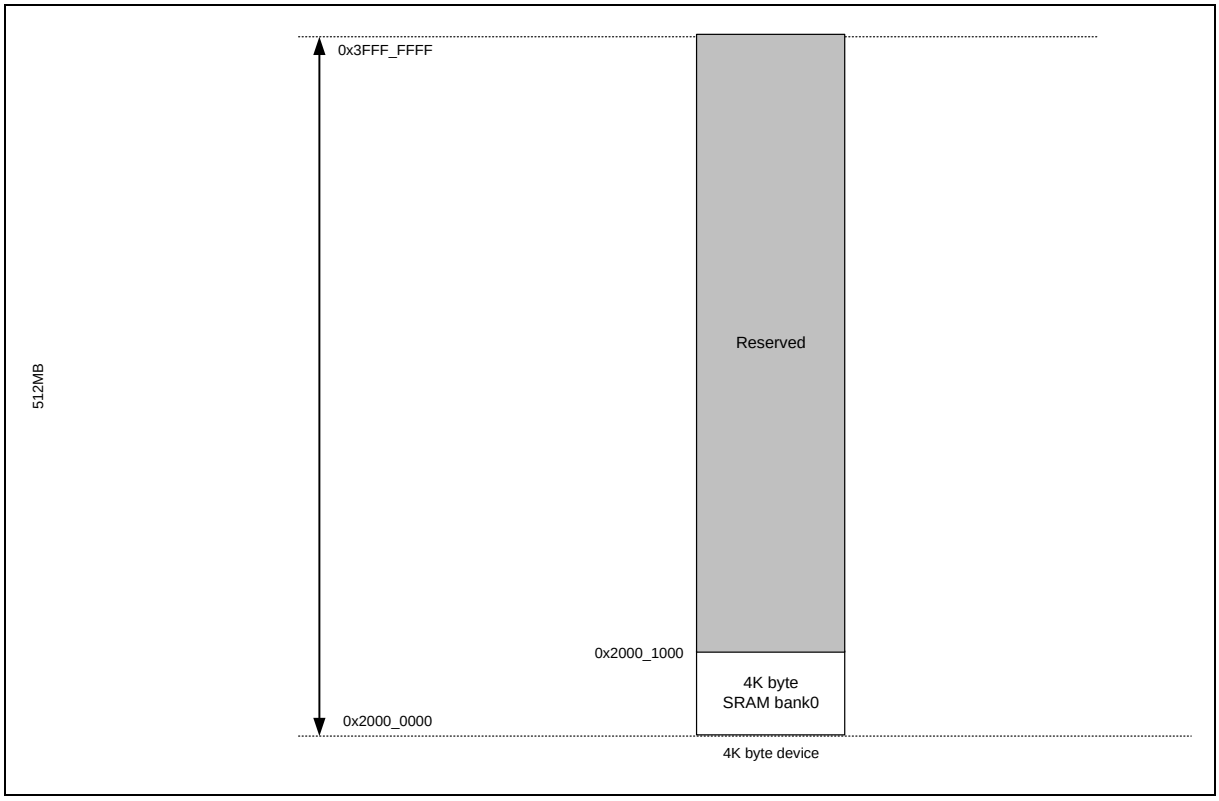


Figure 6.2-9 SRAM Memory Organization

6.2.7 Chip Bus Matrix

The M0A21/M0A23 series supports Bus Matrix to manage the access arbitration between masters. The access arbitration use round-robin algorithm as the bus priority.

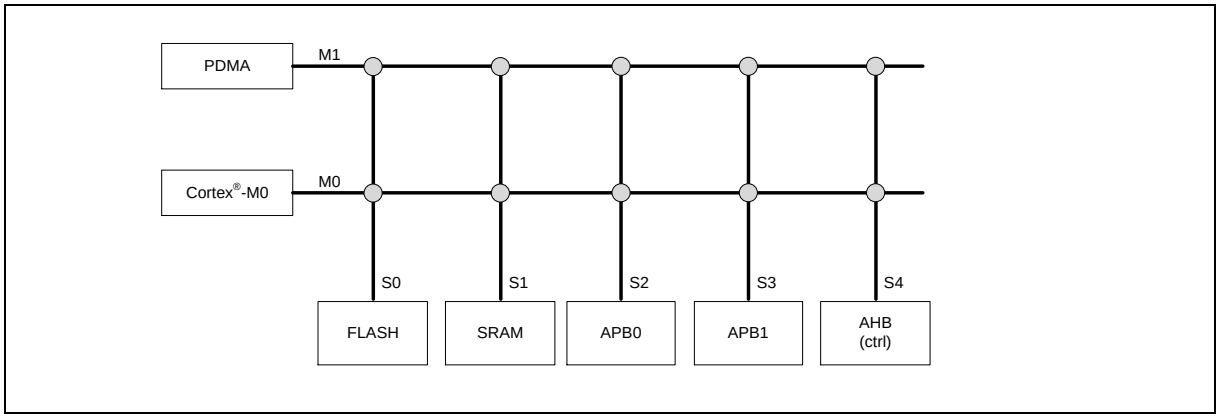


Figure 6.2-10 NuMicro® M0A21/M0A23 Bus Matrix Diagram

6.2.8 IRC Auto Trim

This chip supports auto-trim function: the HIRC trim (48 MHz RC oscillator), according to the accurate external 32.768 kHz crystal oscillator, automatically gets accurate output frequency, 0.25 % deviation within all temperature ranges.

In HIRC case, the system needs an accurate 48 MHz clock. In such case, if not soldering 32.768 kHz crystal or 12 MHz crystal in system, user has to set REFCKSEL (SYS_HIRCTRIMCTL [10] reference clock selection) to “0”, set FREQSEL (SYS_HIRCTRIMCTL [1:0] trim frequency selection) to “01”, and the auto-trim function will be enabled. Interrupt status bit FREQLOCK (SYS_HIRCTRIMSTS[0] HIRC frequency lock status) “1” indicates the HIRC output frequency is accurate within 0.25% deviation.

6.2.9 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. All proteced control registers are noted “(Write Protect)” and add an note “**Note:** This bit is write protected. Refer to the SYS_REGLCTL register in register description field.

6.2.10 UART0_TXD/USCI0_DAT0 Modulation with PWM

This chip supports UART0_TXD/USCI_DAT0 to modulate with PWM channel. User can set MODPWMSEL(SYS_MODCTL[7:4]) to choose which PWM0 channel to modulate with UART0_TXD/USCI0_DAT0 and set MODEN(SYS_MODCTL[0]) to enable modulation function. User can set TXDINV(UART_LINE[8]) to inverse UART0_TXD or DATOINV(UUART_LINECTL[5]) to inverse USCI0_DAT0 before modulating with PWM.

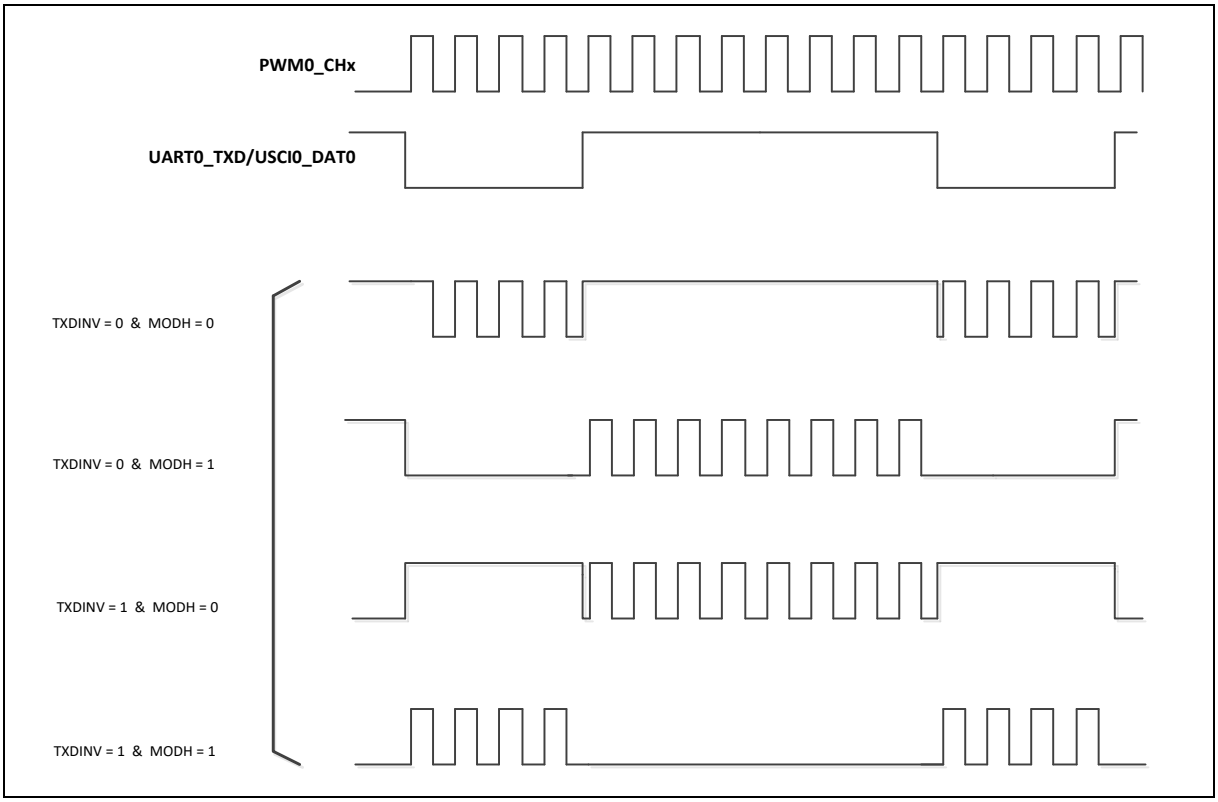


Figure 6.2-11 UART0_TXD/USCI0_DAT0 Modulated with PWM Channel

6.2.11 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
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	0x00XX_038X
SYS_IVSCTL	SYS_BA+0x1C	R/W	Internal Voltage Source Control Register	0x0000_0000
SYS_PORCTL	SYS_BA+0x24	R/W	Power-On-reset Controller Register	0x0000_0000
SYS_VREFCTL	SYS_BA+0x28	R/W	V_{REF} Control Register	0x0000_0100
SYS_GPA_MFP0	SYS_BA+0x30	R/W	GPIOA Multiple Function Control Register 0	0x0000_0202
SYS_GPA_MFP1	SYS_BA+0x34	R/W	GPIOA Multiple Function Control Register 1	0x0000_0000
SYS_GPB_MFP1	SYS_BA+0x44	R/W	GPIOB Multiple Function Control Register 1	0x0000_0000
SYS_GPC_MFP0	SYS_BA+0x50	R/W	GPIOC Multiple Function Control Register 0	0x0000_0000
SYS_GPC_MFP1	SYS_BA+0x54	R/W	GPIOC Multiple Function Control Register 1	0x0000_0000
SYS_GPD_MFP0	SYS_BA+0x60	R/W	GPIOD Multiple Function Control Register 0	0x0000_0000
SYS_GPD_MFP1	SYS_BA+0x64	R/W	GPIOD Multiple Function Control Register 1	0x0000_0000
SYS_GPA_MFOS	SYS_BA+0xB0	R/W	GPIOA Multiple Function Output Select Register	0x0000_0000
SYS_GPB_MFOS	SYS_BA+0xB4	R/W	GPIOB Multiple Function Output Select Register	0x0000_0000
SYS_GPC_MFOS	SYS_BA+0xB8	R/W	GPIOC Multiple Function Output Select Register	0x0000_0000
SYS_GPD_MFOS	SYS_BA+0xBC	R/W	GPIOD Multiple Function Output Select 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_MODCTL	SYS_BA+0xE8	R/W	Modulation Control Register	0x0000_0000
SYS_HIRCTRIMCTL	SYS_BA+0xF0	R/W	HIRC Trim Control Register	0x0008_0000
SYS_HIRCTRIMIEN	SYS_BA+0xF4	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000
SYS_HIRCTRIMSTS	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_PORDISAN	SYS_BA+0x1EC	R/W	Analog POR Disable Control Register	0x0000_0000

6.2.12 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] 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	Reserved	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®-M0 lockup happened and chip is reset. **Note:** 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®- M0 Core and Flash Memory Controller (FMC). 0 = No reset from CPU. 1 = The Cortex®-M0 Core and FMC are reset by software setting CPURST to 1. **Note:** Write to clear this bit to 0.
[6]	**Reserved**	Reserved.
[5]	**SYSRF**	**System Reset Flag** The system reset flag is set by the "Reset Signal" from the Cortex®-M0 Core to indicate the previous reset source. 0 = No reset from Cortex®-M0. 1 = The Cortex®- M0 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®-M0 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							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CRCRST	Reserved		HDIV_RST	Reserved	PDMARST	CPURST	CHIPRST

Bits	Description	
[31:8]	**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]	**HDIV_RST**	**HDIV Controller Reset (Write Protect)** Set this bit to 1 will generate a reset signal to the hardware divider. User need to set this bit to 0 to release from the reset state. 0 = Hardware divider controller normal operation. 1 = Hardware divider controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Controller Reset (Write Protect)** Setting this bit to 1 will generate a reset signal to the PDMA. User needs to set this bit to 0 to release from reset state. 0 = PDMA controller normal operation. 1 = PDMA 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 Flash Memory Controller(FMC), 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 Flash are also reload. About the difference between CHIPRST and SYSRESETREQ(AIRCR[2]), please refer to section 6.2.2 0 = Chip normal operation. 1 = Chip one-shot reset. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** Reset by powr on reset.

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
Reserved			ADCRST	Reserved			CAN0RST
23	22	21	20	19	18	17	16
Reserved						UART1RST	UART0RST
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ACMP01RST	Reserved	TMR3RST	TMR2RST	TMR1RST	TMR0RST	GPIORST	Reserved

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**ADCRST**	**ADC Controller Reset** 0 = ADC controller normal operation. 1 = ADC controller reset.
[27:25]	**Reserved**	Reserved.
[24]	**CAN0RST**	**CAN0 Controller Reset** 0 = CAN0 controller normal operation. 1 = CAN0 controller reset.
[23:18]	**Reserved**	Reserved.
[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.
[15:8]	**Reserved**	Reserved.
[7]	**ACMP01RST**	**Analog Comparator 0/1 Controller Reset** 0 = Analog Comparator 0/1 controller normal operation. 1 = Analog Comparator 0/1 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							
23	22	21	20	19	18	17	16
Reserved							PWM0RST
15	14	13	12	11	10	9	8
Reserved			DAC0RST	Reserved		USCI1RST	USCI0RST
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**PWM0RST**	**PWM0 Controller Reset** 0 = PWM0 controller normal operation. 1 = PWM0 controller reset.
[15:13]	**Reserved**	Reserved.
[12]	**DAC0RST**	**DAC0 Controller Reset** 0 = DAC0 controller normal operation. 1 = DAC0 controller reset.
[11:10]	**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.
[7:0]	**Reserved**	Reserved.

Brown-out Detector Control Register (SYS_BODCTL)

Partial of the SYS_BODCTL control registers values are initiated by the Flash 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	0x00XX_038X

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
Reserved	LVRDGSEL			Reserved	BODDGSEL		
7	6	5	4	3	2	1	0
LVREN	BODOUT	BODLPM	BODIF	BODRSTEN	Reserved		BODEN

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**BODVL**	**Brown-out Detector Threshold Voltage Selection (Write Protect)** The default value is set by Flash controller user configuration register CBOV (CONFIG0 [22:21]). 00 = Brown-Out Detector threshold voltage is 2.3V. 01 = Brown-Out Detector threshold voltage is 2.7V. 10 = Brown-Out Detector threshold voltage is 3.7V. 11 = Brown-Out Detector threshold voltage is 4.4V. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. Note : reset by powr on reset
[15]	**Reserved**	Reserved.
[14:12]	**LVRDGSEL**	**LVR Output De-glitch Time Select (Write Protect)** 000 = Without de-glitch function. 001 = 64 system clock (HCLK). 010 = 128 system clock (HCLK). 011 = 256 system clock (HCLK). 100 = 512 system clock (HCLK). 101 = 1024 system clock (HCLK). 110 = 2048 system clock (HCLK). 111 = 4096 system clock (HCLK). **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[11]	**Reserved**	Reserved.

Bits	Description	
[10:8]	**BODDGSEL**	**Brown-out Detector Output De-glitch Time Select (Write Protect)** 000 = BOD output is sampled by RC32K clock. 001 = 64 system clock (HCLK). 010 = 128 system clock (HCLK). 011 = 256 system clock (HCLK). 100 = 512 system clock (HCLK). 101 = 1024 system clock (HCLK). 110 = 2048 system clock (HCLK). 111 = 4096 system clock (HCLK). **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[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 200us 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.
[4]	**BODIF**	**Brown-out Detector Interrupt Flag** 0 = Brown-out Detector does not detect any voltage draft at V_{DD} down through or up through the voltage of BODVL setting. 1 = When Brown-out Detector detects the V_{DD} is dropped down through the voltage of BODVL setting or the V_{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.

Bits	Description	
[3]	**BODRSTEN**	**Brown-out Reset Enable Bit (Write Protect)** The default value is set by Flash 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. **Note 3:** Reset by powr on reset.
[2:1]	**Reserved**	Reserved.
[0]	**BODEN**	**Brown-out Detector Enable Bit (Write Protect)** The default value is set by Flash controller user configuration register CBODEN (CONFIG0 [19]). 0 = Brown-out Detector function Disabled. 1 = Brown-out Detector function Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** Reset by powr on reset.

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
Reserved							VTEMPEN

Bits	Description	
[31:2]	**Reserved**	Reserved.
[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:** After this bit is set to 1, the value of temperature sensor output can be obtained from ADC conversion result. Please refer to ADC function chapter for details.

Power-on Reset Controller Register (SYS_PORCTL)

Register	Offset	R/W	Description	Reset Value
SYS_PORCTL	SYS_BA+0x24	R/W	Power-On-reset Controller Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**POROFF**	**Power-on Reset Enable Bit (Write Protect)** When powered on, the POR circuit generates a reset signal to reset the whole chip function, but noise on the power may cause the POR active again. User can disable internal POR circuit to avoid unpredictable noise to cause chip reset by writing 0x5AA5 to this field. The POR function 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:** This bit is write protected. Refer to the SYS_REGLCTL register.

V_{REF} Control Register (SYS_VREFCTL)

Register	Offset	R/W	Description	Reset Value
SYS_VREFCTL	SYS_BA+0x28	R/W	V_{REF} Control Register	0x0000_0100

31	30	29	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	PRELOADSEL	Reserved	ADCPRESEL	VREFCTL			

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**PRELOADSEL**	**Pre-load Timing Selection (Write Protect)** 0 = Pre-load time is 60us for 0.1uF Capacitor. 1 = Pre-load time is 310us for 1uF Capacitor. **Note:** These bits is write protected. Refer to the SYS_REGLCTL register.
[5]	**Reserved**	Reserved.
[4]	**ADCPRESEL**	**ADC Voltage Reference** 0 = ADC positive reference voltage comes from AV_{DD} (voltage of V_{DD} pin) 1 = ADC positive reference voltage comes from internal or external V_{REF}. **Note:** These bits is write protected. Refer to the SYS_REGLCTL register.
[3:0]	**VREFCTL**	**V_{REF} Control Bits (Write Protect)** 0000 = V_{REF} is from external pin. 0001 = V_{REF} is internal 1.536V. 0011 = V_{REF} is internal 2.048V. 0101 = V_{REF} is internal 2.56V. 0111 = V_{REF} is internal 3.072V. 1001 = V_{REF} is internal 4.096V. Others = Reserved. **Note 1:** GPA1 needs to keep folating if Internal voltage reference is enabled. **Note 2:** These bits are write protected. Refer to the SYS_REGLCTL register.

GPIOA Multiple Function Control Register 0 (SYS_GPA_MFP0)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP0	SYS_BA+0x30	R/W	GPIOA Multiple Function Control Register 0	0x0000_0202

31	30	29	28	27	26	25	24
GPA3MFP							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
GPA2MFP							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
GPA1MFP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPA0MFP							

Bits	Description	
[31:24]	**GPA3MFP**	**PA.3 Multi-function Pin Selection**
[23:16]	**GPA2MFP**	**PA.2 Multi-function Pin Selection**
[15:8]	**GPA1MFP**	**PA.1 Multi-function Pin Selection**
[7:0]	**GPA0MFP**	**PA.0 Multi-function Pin Selection**

GPIOA Multiple Function Control Register 1 (SYS_GPA_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP1	SYS_BA+0x34	R/W	GPIOA 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**
GPA5MFP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPA4MFP							

Bits	Description	
[30:16]	**Reserved**	Reserved.
[15:8]	**GPA5MFP**	**PA.5 Multi-function Pin Selection**
[7:0]	**GPA4MFP**	**PA.4 Multi-function Pin Selection**

GPIOB Multiple Function Control Register 1 (SYS_GPB_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFP1	SYS_BA+0x44	R/W	GPIOB Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
GPB7MFP							
23	22	21	20	19	18	17	16
GPB6MFP							
15	14	13	12	11	10	9	8
GPB5MFP							
7	6	5	4	3	2	1	0
GPB4MFP							

Bits	Description	
[31:24]	GPB7MFP	PB.7 Multi-function Pin Selection
[23:16]	GPB6MFP	PB.6 Multi-function Pin Selection
[15:8]	GPB5MFP	PB.5 Multi-function Pin Selection
[7:0]	GPB4MFP	PB.4 Multi-function Pin Selection

GPIOC Multiple Function Control Register 0 (SYS_GPC_MFP0)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFP0	SYS_BA+0x50	R/W	GPIOC Multiple Function Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
GPC3MFP							
23	22	21	20	19	18	17	16
GPC2MFP							
15	14	13	12	11	10	9	8
GPC1MFP							
7	6	5	4	3	2	1	0
GPC0MFP							

Bits	Description	
[31:24]	GPC3MFP	PC3 Multi-function Pin Selection
[23:16]	GPC2MFP	PC.2 Multi-function Pin Selection
[15:8]	GPC1MFP	PC.1 Multi-function Pin Selection
[7:0]	GPC0MFP	PC.0 Multi-function Pin Selection

GPIOC Multiple Function Control Register 1 (SYS_GPC_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFP1	SYS_BA+0x54	R/W	GPIOC Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
GPC7MFP							
23	22	21	20	19	18	17	16
GPC6MFP							
15	14	13	12	11	10	9	8
GPC5MFP							
7	6	5	4	3	2	1	0
GPC4MFP							

Bits	Description	
[31:24]	GPC7MFP	PC.7Multi-function Pin Selection
[23:16]	GPC6MFP	PC.6 Multi-function Pin Selection
[15:8]	GPC5MFP	PC.5 Multi-function Pin Selection
[7:0]	GPC4MFP	PC.4 Multi-function Pin Selection

GPIOD Multiple Function Control Register 0 (SYS_GPD_MFP0)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFP0	SYS_BA+0x60	R/W	GPIOD Multiple Function Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
GPD3MFP							
23	22	21	20	19	18	17	16
GPD2MFP							
15	14	13	12	11	10	9	8
GPD1MFP							
7	6	5	4	3	2	1	0
GPD0MFP							

Bits	Description	
[31:24]	GPD3MFP	PD3 Multi-function Pin Selection
[23:16]	GPD2MFP	PD.2 Multi-function Pin Selection
[15:8]	GPD1MFP	PD.1 Multi-function Pin Selection
[7:0]	GPD0MFP	PD.0 Multi-function Pin Selection

GPIOD Multiple Function Control Register 1 (SYS_GPD_MFP1)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFP1	SYS_BA+0x64	R/W	GPIOD Multiple Function Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
GPD7MFP							
23	22	21	20	19	18	17	16
GPD6MFP							
15	14	13	12	11	10	9	8
GPD5MFP							
7	6	5	4	3	2	1	0
GPD4MFP							

Bits	Description	
[31:24]	GPD7MFP	PD.7Multi-function Pin Selection
[23:16]	GPD6MFP	PD.6 Multi-function Pin Selection
[15:8]	GPD5MFP	PD.5 Multi-function Pin Selection
[7:0]	GPD4MFP	PD.4 Multi-function Pin Selection

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

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFOS	SYS_BA+0xB0	R/W	GPIOA Multiple Function Output Select Register	0x0000_0000
SYS_GPB_MFOS	SYS_BA+0xB4	R/W	GPIOB Multiple Function Output Select Register	0x0000_0000
SYS_GPC_MFOS	SYS_BA+0xB8	R/W	GPIOC Multiple Function Output Select Register	0x0000_0000
SYS_GPD_MFOS	SYS_BA+0xBC	R/W	GPIOD 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 funtion pin output mode type is unchanged. 1 = Multiple funtion pin output mode type is Open-drain mode. **Note:** Max. n=5 for port A. Max. n=7 for port B. The PB.0/ PB.1/ PB.2/ PB.3 is ignored. Max. n=7 for port C/D. If MFOS is enabled then GPIO mode setting is ignored when MFP setting be GPIO.

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							
7	6	5	4	3	2	1	0
PDMABIST	Reserved						

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**PDMABIST**	**PDMA BIST Enable Bit (Write Protect)** This bit enables BIST test for PDMA RAM 0 = system PDMA BIST Disabled. 1 = system PDMA BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6:0]	**Reserved**	Reserved.

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							
23	22	21	20	19	18	17	16
PDMAEND	Reserved						
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PDMABISTF	Reserved						

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**PDMAEND**	**PDMA SRAM BIST Test Finish** 0 = PDMA SRAM BIST is active. 1 = PDMA SRAM BIST test finished.
[22:8]	**Reserved**	Reserved.
[7]	**PDMABISTF**	**PDMA SRAM BIST Failed Flag** 0 = PDMA SRAM BIST pass. 1 = PDMA SRAM BIST failed.
[6:0]	**Reserved**	Reserved.

Modulation Control Register (SYS_MODCTL)

Register	Offset	R/W	Description	Reset Value
SYS_MODCTL	SYS_BA+0xE8	R/W	Modulation 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
MODPWMSEL				Reserved		MODH	MODEN

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:4]	**MODPWMSEL**	**PWM0 Channel Select for Modulation** Select the PWM0 channel to modulate with the UART0_TXD or USCI0_DAT0. 0000: PWM0 Channel 0 modulate with UART0_TXD. 0001: PWM0 Channel 1 modulate with UART0_TXD. 0010: PWM0 Channel 2 modulate with UART0_TXD. 0011: PWM0 Channel 3 modulete with UART0_TXD. 0100: PWM0 Channel 4 modulete with UART0_TXD. 0101: PWM0 Channel 5 modulete with UART0_TXD. 0110: Reserved. 0111: Reserved. 1000: PWM0 Channel 0 modulate with USCI0_DAT0. 1001: PWM0 Channel 1 modulate with USCI0_DAT0. 1010: PWM0 Channel 2 modulate with USCI0_DAT0. 1011: PWM0 Channel 3 modulete with USCI0_DAT0. 1100: PWM0 Channel 4 modulete with USCI0_DAT0. 1101: PWM0 Channel 5 modulete with USCI0_DAT0. 1110: Reserved. 1111: Reserved. **Note:** This bis is valid while MODEN (SYS_MODCTL[0]) is set to 1.
[3:2]	**Reserved**	Reserved.
[1]	**MODH**	**Modulation at Data High** Select modulation pulse(PWM0) at high or low of UART0_TXD or USCI0_DAT0 0 = Modulation pulse at UART0_TXD low or USCI0_DAT0 low. 1 = Modulation pulse at UART0_TXD high or USCI0_DAT0 high.
[0]	**MODEN**	**Modulation Function Enable Bit** This bit enables modulation funcion by modulating with PWM0 channel output and USCI0(USCI0_DAT0) or UART0(UART0_TXD) output.

		0 = Modulation Function Disabled. 1 = Modulation Function Enabled.

HIRC Trim Control Register (SYS_HIRCTRIMCTL)

Register	Offset	R/W	Description	Reset Value
SYS_HIRCTRIMCTL	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		Reserved		FREQSEL	

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20:16]	**BOUNDARY**	**Boundary Selection** Fill the boundary range from 0x1 to 0x1F, 0x0 is reserved. **Note:** This field is effective only when the BOUNDEN(SYS_HIRCTRIMCTL[9]) is enable.
[15:11]	**Reserved**	Reserved.
[10]	**REFCKSEL**	**Reference Clock Selection** 0 = HIRC trim reference clock is from LXT (32.768 kHz). 1 = Reserved. **Note:** If there is no reference clock (LXT) when the rc_trim is enabled, **CLKERIF (SYS_HIRCTRIMCTL[2]) will be set to 1.**
[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]	**Reserved**	Reserved.
[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.

HIRC Trim Interrupt Enable Register (SYS_HIRCTRIMIEN)

Register	Offset	R/W	Description	Reset Value
SYS_HIRCTRIMIEN	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	TFALIEN	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_HIRCTRIMSTS[2]) is set during auto trim operation, an interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = Disable CLKERRIF(SYS_HIRCTRIMSTS[2]) status to trigger an interrupt to CPU. 1 = Enable CLKERRIF(SYS_HIRCTRIMSTS[2]) status to trigger an interrupt to CPU.
[1]	**TFALIEN**	**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_HIRCTRIMCTL[1:0]). If this bit is high and TFAILIF(SYS_HIRCTRIMSTS[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_HIRCTRIMSTS[1]) status to trigger an interrupt to CPU. 1 = Enable TFAILIF(SYS_HIRCTRIMSTS[1]) status to trigger an interrupt to CPU.
[0]	**Reserved**	Reserved.

HIRC Trim Interrupt Status Register (SYS_HIRCTRIMSTS)

Register	Offset	R/W	Description	Reset Value
SYS_HIRCTRIMSTS	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	CLKERIF	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]	**CLKERIF**	**Clock Error Interrupt Status** When the frequency relation between reference clock and 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_HIRCTRIMCTL[1:0]) will be cleared to 00 by hardware automatically if CESTOPEN(SYS_HIRCTRIMCTL[8]) is set to 1. If this bit is set and CLKEIEN(SYS_HIRCTIEN[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. **Note** : reset by powr on reset
[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_HIRCTRIMCTL[1:0]) will be cleared to 00 by hardware automatically. If this bit is set and TFAILIEN(SYS_HIRCIEN[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. **Note** : reset by powr on reset
[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 frequecy 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. **Note** : Reset by powr on reset.

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 writen 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.

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:** This bit is write protected. Refer to the SYS_REGLCTL register.

6.2.13 System Timer (SysTick)

The Cortex®-M0 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®-M0 Technical Reference Manual” and “Arm® v6-M Architecture Reference Manual”.

6.2.13.1 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_CTRL	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

6.2.13.2 System Timer Control Register Description

SysTick Control and Status Register (SYST_CTRL)

Register	Offset	R/W	Description	Reset Value
SYST_CTRL	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** The 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 protection. The register is write-clear. A software write of any value will clear the register to 0.

6.2.14 Nested Vectored Interrupt Controller (NVIC)

The Cortex®-M0 provides an interrupt controller as an integral part of the exception mode, named as "Nested Vectored Interrupt Controller (NVIC)", which is closely coupled to the processor core and provides following features:

- Nested and Vectored interrupt support
- Automatic processor state saving and restoration
- Reduced and deterministic interrupt latency

The NVIC prioritizes and handles all supported exceptions. All exceptions are handled in "Handler Mode". This NVIC architecture supports 32 (IRQ[31:0]) discrete interrupts with 4 levels of priority. All of the interrupts and most of the system exceptions can be configured to different priority levels. When an interrupt occurs, the NVIC will compare the priority of the new interrupt to the current running one's priority. If the priority of the new interrupt is higher than the current one, the new interrupt handler will override the current handler.

When an interrupt is accepted, the starting address of the interrupt service routine (ISR) is fetched from a vector table in memory. There is no need to determine which interrupt is accepted and branch to the starting address of the correlated ISR by software. While the starting address is fetched, NVIC will also automatically save processor state including the registers "PC, PSR, LR, R0~R3, R12" to the stack. At the end of the ISR, the NVIC will restore the mentioned registers from stack and resume the normal execution. Thus it will take less and deterministic time to process the interrupt request.

The NVIC supports "Tail Chaining" which handles back-to-back interrupts efficiently without the overhead of states saving and restoration and therefore reduces delay time in switching to pending ISR at the end of current ISR. The NVIC also supports "Late Arrival" which improves the efficiency of concurrent ISRs. When a higher priority interrupt request occurs before the current ISR starts to execute (at the stage of state saving and starting address fetching), the NVIC will give priority to the higher one without delay penalty. Thus it advances the real-time capability.

For more detailed information, please refer to the "Arm® Cortex®-M0 Technical Reference Manual" and "Arm® v6-M Architecture Reference Manual".

6.2.14.1 Exception Model and System Interrupt Map

Table 6.2-8 lists the exception model supported by the M0A21/M0A23 series. Software can set four levels of priority on some of these exceptions as well as on all interrupts. The highest user-configurable priority is denoted as "0" and the lowest priority is denoted as "3". The default priority of all the user-configurable interrupts is "0". Note that priority "0" is treated as the fourth priority on the system, after three system exceptions "Reset", "NMI" and "Hard Fault".

Exception Name	Vector Number	Priority
Reset	1	-3
NMI	2	-2
Hard Fault	3	-1
Reserved	4 ~ 10	Reserved
SVCall	11	Configurable
Reserved	12 ~ 13	Reserved
PendSV	14	Configurable

SysTick	15	Configurable
Interrupt (IRQ0 ~ IRQ31)	16 ~ 47	Configurable

Table 6.2-8 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	WDT_INT	Watchdog Timer interrupt
18	2	EINT024	External interrupt from EINT0,2,4.
19	3	EINT135	External interrupt from EINT1.3.5
20	4	GPAB_INT	External interrupt from PA, PB pin
21	5	GPCD_INT	External interrupt from PC, PD pin
22	6	PWM0_INT	PWM0 interrupt
23	7	Reserved	Reserved
24	8	TMR0_INT	Timer 0 interrupt
25	9	TMR1_INT	Timer 1 interrupt
26	10	TMR2_INT	Timer 2 interrupt
27	11	TMR3_INT	Timer 3 interrupt
28	12	UART0_INT	UART0 interrupt
29	13	UART1_INT	UART1 interrupt
30	14	Reserved	Reserved
31	15	CAN0_INT	CAN0 interrupt
32	16	Reserved	Reserved
33	17	Reserved	Reserved
34	18	Reserved	Reserved
35	19	Reserved	Reserved
36	20	Reserved	Reserved
37	21	Reserved	Reserved
38	22	USCI0	USCI0 interrupt
39	23	Reserved	Reserved
40	24	DAC0_INT	DAC0 interrupt
41	25	ACMP01_INT	ACMP0 and ACMP1 interrupt
42	26	PDMA_INT	PDMA interrupt

43	27	USCI1	USCI1 interrupt
44	28	PWRWU_INT	Clock controller interrupt for chip wake-up from power-down state
45	29	ADC_INT	ADC interrupt
46	30	CLKFAIL	Clock fail detected or IRC Auto Trim interrupt
47	31	Reserved	Reserved

Table 6.2-9 Interrupt Number Table

6.2.14.2 Vector Table

When an interrupt is accepted, the processor will automatically fetch the starting address of the interrupt service routine (ISR) from a vector table in memory. For Armv6-M, the vector table base address is fixed at 0x00000000. 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.

Vector Table Word Offset	Description
0	SP_main – The Main stack pointer
Vector Number	Exception Entry Pointer using that Vector Number

Table 7.2-10 Vector Figure Format

6.2.14.3 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 be activated. 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.

6.2.14.4 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+0x000	R/W	IRQ0 ~ IRQ31 Set-enable Control Register	0x0000_0000
NVIC_ICER0	NVIC_BA+0x080	R/W	IRQ0 ~ IRQ31 Clear-enable Control Register	0x0000_0000
NVIC_ISPR0	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-pending Control Register	0x0000_0000
NVIC_ICPR0	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-pending Control Register	0x0000_0000
NVIC_IABR0	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ31 Active Bit Register	0x0000_0000
NVIC_IPRn **n=0,1..7**	0xE000E400 +0x4*n	R/W	IRQ0 ~ IRQ31 Priority Control Register	0x0000_0000
STIR	0xE000EF00	R/W	Software Trigger Interrupt Registers	0x0000_0000

IRQ0 ~ IRQ31 Set-enable Control Register (NVIC_ISER0)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER0	NVIC_BA+0x000	R/W	IRQ0 ~ IRQ31 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 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 ~ IRQ31 Clear-enable Control Register (NVIC_ICER0)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER0	NVIC_BA+0x080	R/W	IRQ0 ~ IRQ31 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 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 ~ IRQ31 Set-pending Control Register (NVIC_ISPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR0	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ31 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 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 ~ IRQ31 Clear-pending Control Register (NVIC_ICPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR0	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ31 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 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 ~ IRQ31 Active Bit Register (NVIC_IABR0)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR0	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ31 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 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ0 ~ IRQ31 Interrupt Priority Register (NVIC_IPRn)

Register	Offset	R/W	Description	Reset Value
NVIC_IPRn **n=0,1..7**	0xE000E400 +0x4*n	R/W	IRQ0 ~ IRQ31 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	0xE000EF00	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 0-31. For example, a value of 0x03 specifies interrupt IRQ3.

6.2.14.5 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**
Reserved			**CLKFAIL**	**Reserved**	**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 PC.7 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from PC.7 pin NMI source Disabled. 1 = External interrupt from PC.7 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[12]	**EINT4**	**External Interrupt From PC.6 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from PC.6 pin NMI source Disabled. 1 = External interrupt from PC.6 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[11]	**EINT3**	**External Interrupt From PC.3 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from PC.3 pin NMI source Disabled. 1 = External interrupt from PC.3 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[10]	**EINT2**	**External Interrupt From PC.4 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from PC.4 pin NMI source Disabled. 1 = External interrupt from PC.4 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[9]	**EINT1**	**External Interrupt From PC.5 Pin NMI Source Enable (Write Protect)**

		0 = External interrupt from PC.5 pin NMI source Disabled. 1 = External interrupt from PC.5 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[8]	**EINT0**	**External Interrupt From PA.3 or PB.5 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from PA.3 or PB.5 pin NMI source Disabled. 1 = External interrupt from PA.3 or PB.5 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7:5]	**Reserved**	Reserved.
[4]	**CLKFAIL**	**Clock Fail Detected and IRC Auto Trim Interrupt NMI Source Enable (Write Protect)** 0 = Clock fail detected and IRC Auto Trim interrupt NMI source Disabled. 1 = Clock fail detected and IRC Auto Trim interrupt NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**Reserved**	Reserved.
[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**
Reserved			**CLKFAIL**	**Reserved**	**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 PC.7 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PC.7 interrupt is deasserted. 1 = External Interrupt from PC.7 interrupt is asserted.
[12]	**EINT4**	**External Interrupt From PC.6 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PC.6 interrupt is deasserted. 1 = External Interrupt from PC.6 interrupt is asserted.
[11]	**EINT3**	**External Interrupt From PC.3 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PC.3 interrupt is deasserted. 1 = External Interrupt from PC.3 interrupt is asserted.
[10]	**EINT2**	**External Interrupt From PC.4 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PC.4 interrupt is deasserted. 1 = External Interrupt from PC.4 interrupt is asserted.
[9]	**EINT1**	**External Interrupt From PC.5 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PC.5 interrupt is deasserted. 1 = External Interrupt from PC.5 interrupt is asserted.
[8]	**EINT0**	**External Interrupt From PA.3 or PB.5 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PA.3 or PB.5 interrupt is deasserted. 1 = External Interrupt from PA.3 or PB.5 interrupt is asserted.
[7:5]	**Reserved**	Reserved.

[4]	**CLKFAIL**	**Clock Fail Detected or IRC Auto Trim Interrupt Flag (Read Only)** 0 = Clock fail detected or IRC Auto Trim interrupt is deasserted. 1 = Clock fail detected or IRC Auto Trim interrupt is asserted.
[3]	**Reserved**	Reserved.
[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.

6.2.15 System Control Register

The Cortex®-M0 status and operation mode control are managed by System Control Registers. Including CPUID, Cortex®-M0 interrupt priority and Cortex®-M0 power management can be controlled through these system control registers.

For more detailed information, please refer to the "*Arm® Cortex®-M0 Technical Reference Manual*" and "*Arm® v6-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
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

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	PENDSVCLR	PENDSTSET	PENDSTCLR	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 = Changes 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 = Changes 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]	**PENDSVCLR**	**PendSV Clear-pending Bit** Write Operation: 0 = No effect. 1 = Removes 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 = Changes SysTick exception state to pending. Read Operation: 0 = SysTick exception is not pending. 1 = SysTick exception is pending.
[25]	**PENDSTCLR**	**SysTick Exception Clear-pending Bit** Write Operation: 0 = No effect. 1 = Removes 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.
[10:6]	**Reserved**	Reserved.
[5:0]	**VECTACTIVE**	**Number of the Current Active Exception** 0 = Thread mode. Non-zero = The exception number of the currently active exception.

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 6.2-10 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. **Note:** 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	
[31:30]	**PRI_11**	**Priority of System Handler 11 – SVCall** “0” denotes the highest priority and “3” denotes the lowest priority.
[29:0]	**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	
[31:30]	**PRI_15**	**Priority of System Handler 15 – SysTick** “0” denotes the highest priority and “3” denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_14**	**Priority of System Handler 14 – PendSV** “0” denotes the highest priority and “3” denotes the lowest priority.

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®-M0 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 4~24 MHz external high speed crystal (HXT) and 48 MHz internal high speed RC oscillator (HIRC) to reduce the overall system power consumption. Figure 6.3-1 shows the clock generator and the overview of the clock source control.

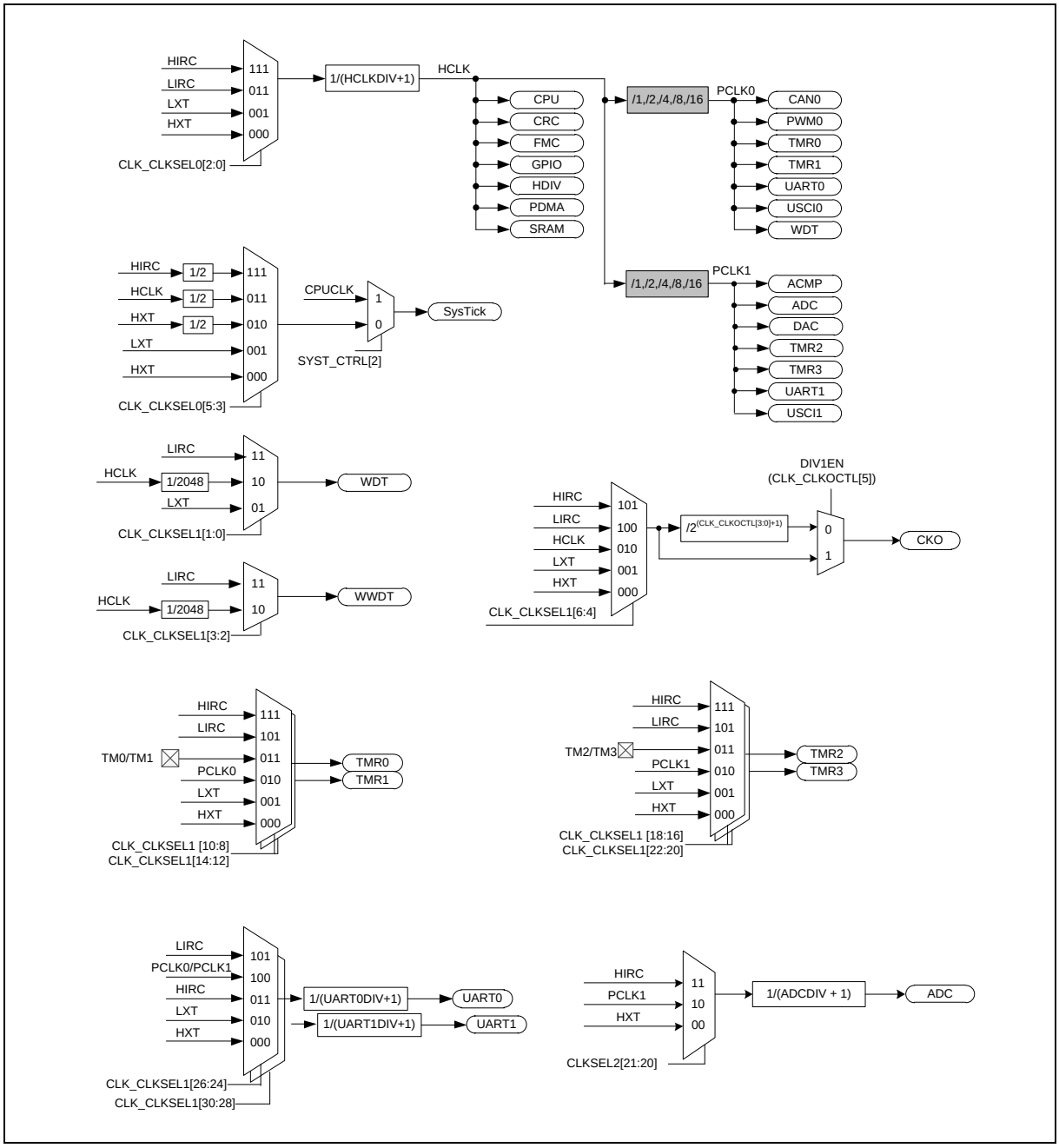


Figure 6.3-1 Clock Generator Global View Diagram

6.3.2 Clock Generator

The clock generator consists of 4 clock sources, which are listed below:

- 32.768 kHz external low speed crystal oscillator (LXT)
- 4~24 MHz external high speed crystal oscillator (HXT)
- 48 MHz internal high speed RC oscillator (HIRC)
- 38.4 kHz internal low speed RC oscillator (LIRC)

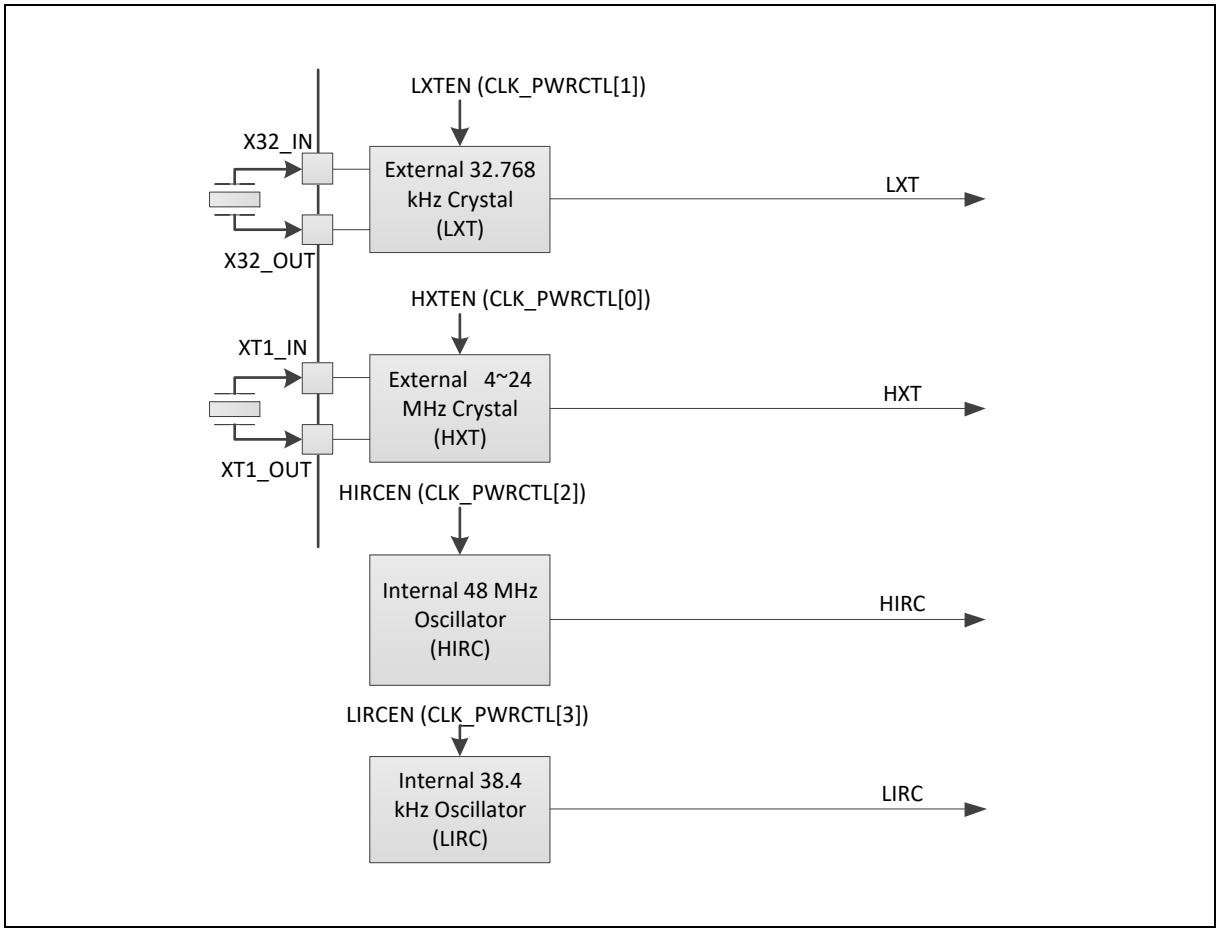


Figure 6.3-2 Clock Generator Block Diagram

6.3.3 System Clock and SysTick Clock

The system clock has 4 clock sources, which were generated from clock generator block. The clock source switch depends on the register HCLKSEL (CLK_CLKSEL0[2:0]). The block diagram is shown in Figure 6.3-3

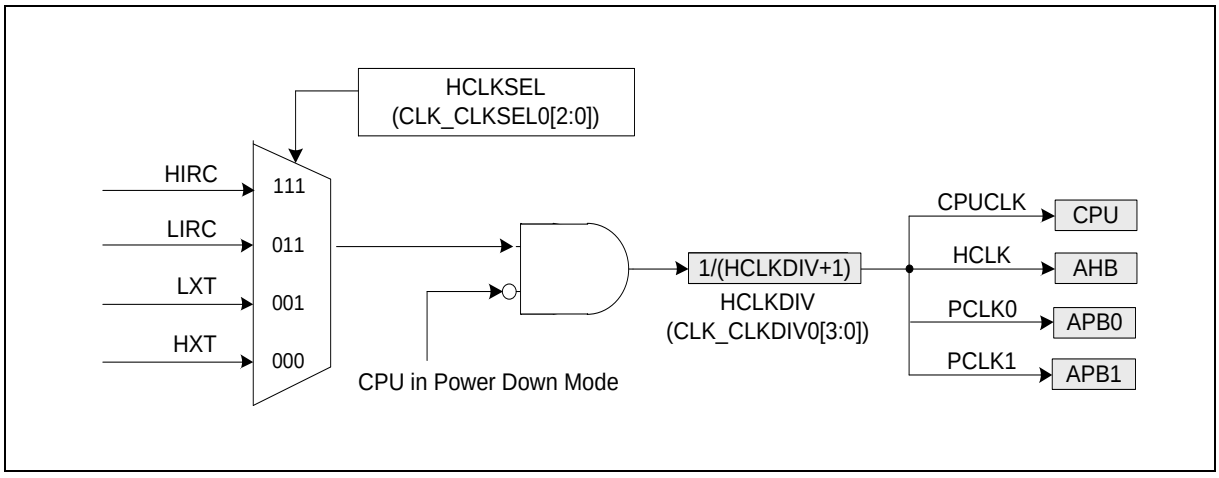


Figure 6.3-3 System Clock Block Diagram

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. 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.

The HXT clock stop detect and system clock switch to HIRC procedure is shown in Figure 6.3-4.

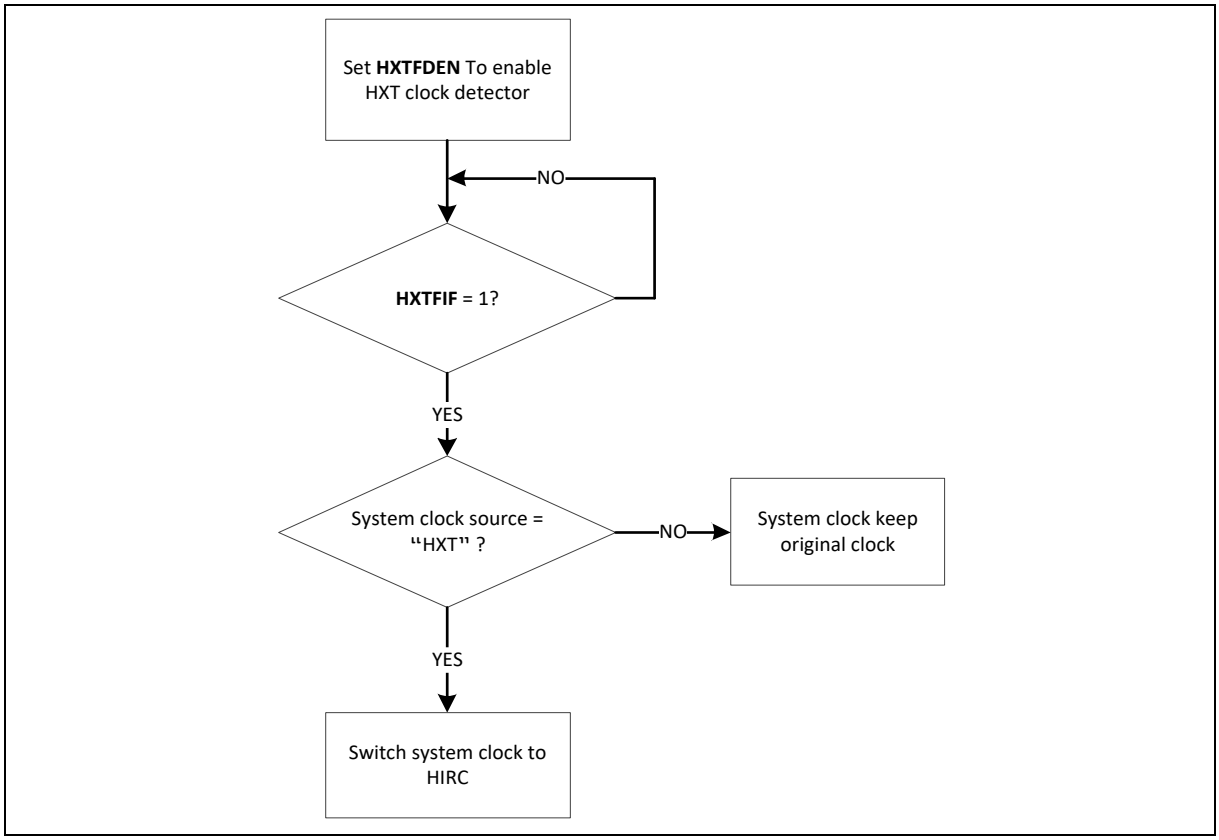


Figure 6.3-4 HXT Stop Protect Procedure

The clock source of SysTick in Cortex®-M0 core can use CPU clock or external clock (SYST_CTRL[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.

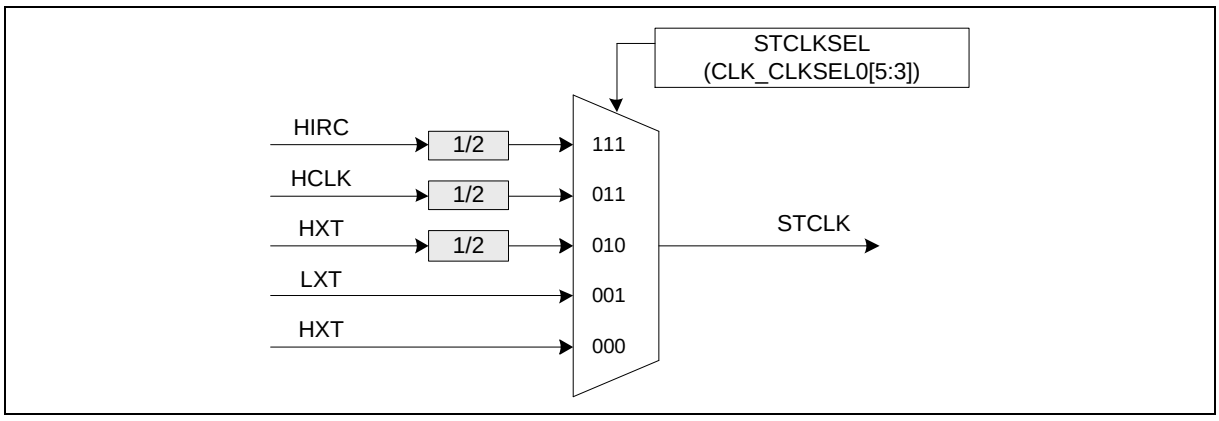


Figure 6.3-5 SysTick Clock Control Block Diagram

6.3.4 Peripherals Clock

The peripherals clock has different clock source switch setting, which depends on the different peripheral. Please refer to the CLK_CLKSELx register description in section 6.3.8.

6.3.5 Power-down Mode Clock

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

For theses clocks, which still keep active, are listed below:

- Clock Generator
  - 38.4 kHz internal low speed RC oscillator (LIRC) clock
  - 32.768 kHz external low speed crystal oscillator (LXT) clock
- Peripherals Clock (When the modules adopt LXT or LIRC as clock source)

6.3.6 Clock Output

This device is equipped with a power-of-2 frequency divider which is composed by16 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.

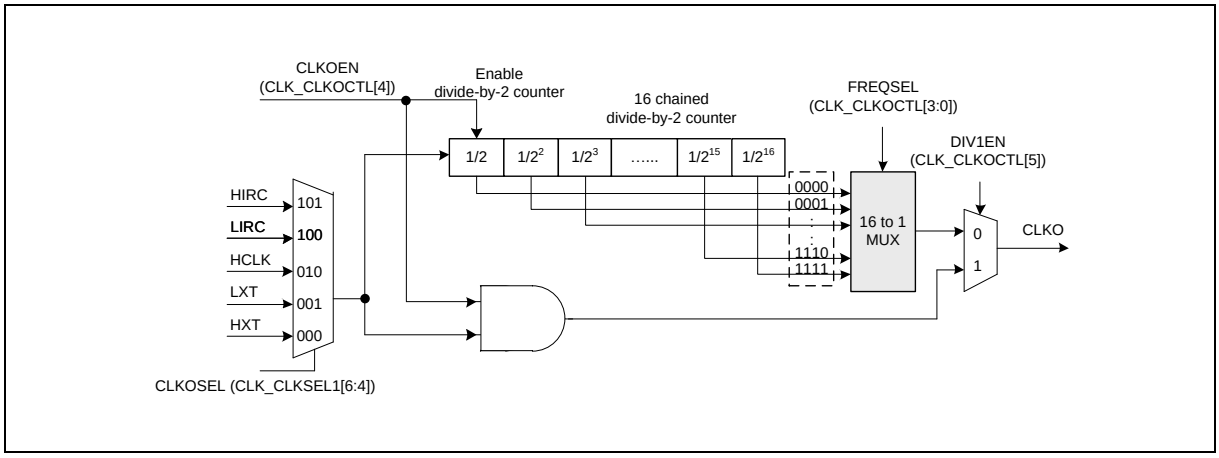


Figure 6.3-6 Clock Output Block Diagram

6.3.7 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	0xB901_001X
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_0004
CLK_APBCLK0	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register 0	0x0000_0001
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	0x0000_003F
CLK_CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0x4477_773B
CLK_CLKSEL2	CLK_BA+0x18	R/W	Clock Source Select Control Register 2	0x0020_032B
CLK_CLKDIV0	CLK_BA+0x20	R/W	Clock Divider Number Register 0	0x0000_0000
CLK_PCLKDIV	CLK_BA+0x34	R/W	APB Clock Divider Register	0x0000_0000
CLK_STATUS	CLK_BA+0x50	R	Clock Status Monitor Register	0x0000_00XX
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_HXTFSEL	CLK_BA+0xB4	R/W	HXT Filter Select Control Register	0x0000_0000

6.3.8 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	0xB901_001X

31	30	29	28	27	26	25	24
HXTSELXT	HXTGAIN			LXTSELXT	LXTGAIN		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PDEN	PDWKIF	PDWKIEN	PDWKDLY	LIRCEN	HIRCEN	LXTEN	HXTEN

Bits	Description	
[31]	**HXTSELXT**	**HXT Crystal Mode Selection** 0 = HXT works as external clock mode. PA.5 is configured as external clock input pin. 1 = HXT works as crystal mode. PA.4 and PA.5 are configured as high speed crystal (HXT) pins. **Note 1:** When HXTSELXT = 0, PA.4 MFP should be set as GPIO mode. The DC characteristic of XT1_OUT is the same as GPIO. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[30:28]	**HXTGAIN**	**HXT Gain Control Bit (Write Protect)** Please refer to HXT Charateristic. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[27]	**LXTSELXT**	**LXT Crystal Mode Selection** 0 = LXT works as external clock mode. PC.5 is configured as external clock input pin. 1 = LXT works as crystal mode. PC.4 and PC.5 are configured as low speed crystal (LXT) pins. **Note 1:** When LXTSELXT = 0, PC.4 MFP should be set as GPIO mode. The DC characteristic of X32_OUT is the same as GPIO. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[26:24]	**LXTGAIN**	**LXT Gain Control Bit (Write Protect)** Please refer to LXT Charateristic. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[23:8]	**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 utill 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 and the HIRC will be disabled in this mode, but LXT and LIRC are not controlled by Power-down mode. If user disable LIRC before entering power-down mode, this bit should be set after LIRC disabled 50us.

		In Power-down mode, 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 operating normally or chip in idle mode because of WFI command. 1 = Chip enters Power-down mode instant or wait 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]	**PDWKDLY**	**Enable the Wake-up Delay Counter (Write Protect)** When the chip wakes up from Power-down mode, the clock control will delay certain clock cycles to wait system clock stable. The delayed clock cycle is 4096 clock cycles when chip works at external high speed crystal oscillator (HXT), and 512 clock cycles when chip works at internal high speed RC oscillator (HIRC). 0 = Clock cycles delay Disabled. 1 = Clock cycles delay Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**LIRCEN**	**LIRC Enable Bit (Write Protect)** 0 = Internal low speed RC oscillator (LIRC) Disabled. 1 = 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 = Internal high speed RC oscillator (HIRC) Disabled. 1 = 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 = External low speed crystal (LXT) Disabled. 1 = External low speed crystal (LXT) Enabled. **Note 1:** Reset by power on reset. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**HXTEN**	**HXT Enable Bit (Write Protect)** 0 = Eexternal high speed crystal (HXT) Disabled. 1 = External high speed crystal (HXT) Enabled. **Note 1:** Reset by power on reset. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.

AHB Devices Clock Enable Control Register (CLK_AHBCLK)

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_AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control 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
CRCCKEN	Reserved		HDIV_EN	Reserved	ISPCKEN	PDMACKEN	Reserved

Bits	Description	
[31:8]	**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]	**HDIV_EN**	**Divider Controller Clock Enable Control** 0 = Divider controller peripheral clock Disabled. 1 = Divider controller peripheral clock Enabled.
[3]	**Reserved**	Reserved.
[2]	**ISPCKEN**	**Flash ISP Controller Clock Enable Bit** 0 = Flash ISP peripheral clock Disabled. 1 = Flash ISP peripheral clock Enabled.
[1]	**PDMACKEN**	**PDMA Controller Clock Enable Bit** 0 = PDMA peripheral clock Disabled. 1 = PDMA 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_0001

31	30	29	28	27	26	25	24
Reserved			ADCCKEN	Reserved			CAN0CKEN
23	22	21	20	19	18	17	16
Reserved						UART1CKEN	UART0CKEN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ACMP01CKEN	CLKOCKEN	TMR3CKEN	TMR2CKEN	TMR1CKEN	TMR0CKEN	Reserved	WDTCKEN

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**ADCCKEN**	**Analog-digital-converter Clock Enable Bit** 0 = ADC clock Disabled. 1 = ADC clock Enabled.
[27:25]	**Reserved**	Reserved.
[24]	**CAN0CKEN**	**CAN0 Clock Enable Bit** 0 = CAN0 clock Disabled. 1 = CAN0 clock Enabled.
[23:18]	**Reserved**	Reserved.
[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.
[15:8]	**Reserved**	Reserved.
[7]	**ACMP01CKEN**	**Analog Comparator 0/1 Clock Enable Bit** 0 = Analog comparator 0/1 clock Disabled. 1 = Analog comparator 0/1 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]	**Reserved**	Reserved.
[0]	**WDTCKEN**	**Watchdog Timer Clock Enable Bit (Write Protect)** 0 = Watchdog timer clock Disabled. 1 = Watchdog timer clock Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** This bit is reset by power on reset, Watchdog reset or software chip reset.

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							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**PWM0CKEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**DACCKEN**	**Reserved**		**USCI1CKEN**	**USCI0CKEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**PWM0CKEN**	**PWM0 Clock Enable Bit** 0 = PWM0 clock Disabled. 1 = PWM0 clock Enabled.
[15:13]	**Reserved**	Reserved.
[12]	**DACCKEN**	**DAC Clock Enable Bit** 0 = DAC clock Disabled. 1 = DAC clock Enabled.
[11:10]	**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.
[7:0]	**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	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		**STCLKSEL**			**HCLKSEL**		

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:3]	**STCLKSEL**	**Cortex®-M0 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. Other = Reserved. **Note 1:** If SysTick clock source is not from HCLK (i.e. SYST_CTRL[2] = 0), SysTick clock source must less than or equal to HCLK/2. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2:0]	**HCLKSEL**	**HCLK 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. 011 = Clock source from LIRC. 111= Clock source from HIRC. Other = Reserved. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** Reset by power on reset

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	0x4477_773B

31	30	29	28	27	26	25	24
Reserved	UART1SEL			Reserved	UART0SEL		
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
Reserved	CLKOSEL			WWDTSEL		WDTSEL	

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**UART1SEL**	**UART1 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 010 = Clock source from external low speed crystal oscillator (LXT). 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from PCLK1. 101 = Clock source from internal low speed RC oscillator (LIRC). Other = Reserved.
[27]	**Reserved**	Reserved.
[26:24]	**UART0SEL**	**UART0 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 010 = Clock source from external low speed crystal oscillator (LXT). 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from PCLK0. 101 = Clock source from internal low speed RC oscillator (LIRC). Other = Reserved.
[23]	**Reserved**	Reserved.
[22:20]	**TMR3SEL**	**TIMER3 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from external low speed crystal oscillator (LXT). 010 = Clock source from PCLK1. 011 = Clock source from external clock T3 pin. 101 = Clock source from internal low speed RC oscillator (LIRC). 111 = Clock source from internal high speed RC oscillator (HIRC). Others = Reserved.

[19]	**Reserved**	Reserved.
[18:16]	**TMR2SEL**	**TIMER2 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from external low speed crystal oscillator (LXT). 010 = Clock source from PCLK1. 011 = Clock source from external clock T2 pin. 101 = Clock source from internal low speed RC oscillator (LIRC). 111 = Clock source from internal high speed RC oscillator (HIRC). Others = Reserved.
[15]	**Reserved**	Reserved.
[14:12]	**TMR1SEL**	**TIMER1 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from external low speed crystal oscillator (LXT). 010 = Clock source from PCLK0. 011 = Clock source from external clock T1 pin. 101 = Clock source from internal low speed RC oscillator (LIRC). 111 = Clock source from internal high speed RC oscillator (HIRC). Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**TMR0SEL**	**TIMER0 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from external low speed crystal oscillator (LXT). 010 = Clock source from PCLK0. 011 = Clock source from external clock T0 pin. 101 = Clock source from internal low speed RC oscillator (LIRC). 111 = Clock source from internal high speed RC oscillator (HIRC). Others = Reserved.
[7]	**Reserved**	Reserved.
[6:4]	**CLKOSEL**	**Clock Divider Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from external low speed crystal oscillator (LXT). 010 = Clock source from HCLK. 100 = Clock source from internal low speed RC oscillator (LIRC). 101 = Clock source from internal high speed RC oscillator (HIRC). Others = Reserved.
[3:2]	**WWDTSEL**	**Window Watchdog Timer Clock Source Selection (Write Protect)** 10 = Clock source from HCLK/2048. 11 = Clock source from internal low speed RC oscillator (LIRC). Others = Reserved.
[1:0]	**WDTSEL**	**Watchdog Timer Clock Source Selection (Write Protect)** 01 = Clock source from external low speed crystal oscillator (LXT). 10 = Clock source from HCLK/2048. 11 = Clock source from internal low speed RC oscillator (LIRC). Others = Reserved. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. 2. It will be forced to 11 when CONFIG0[31], CONFIG0[4], CONFIG0[3] are all ones.

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	0x0020_032B

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**ADCSEL**		**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.
[21:20]	**ADCSEL**	**ADC Clock Source Selection** 00 = Clock source from external high speed crystal oscillator (HXT) clock. 01 = Reserved. 10 = Clock source from PCLK1. 11 = Clock source from internal high speed RC oscillator (HIRC) clock.
[19:0]	**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**
ADCDIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
UART1DIV				**UART0DIV**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**HCLKDIV**			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**ADCDIV**	**ADC Clock Divide Number From ADC Clock Source** ADC clock frequency = (ADC clock source frequency) / (ADCDIV + 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]	**Reserved**	Reserved.
[3:0]	**HCLKDIV**	**HCLK Clock Divide Number From HCLK Clock Source** HCLK clock frequency = (HCLK clock source frequency) / (HCLKDIV + 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 HCLK 000: PCLK1 = HCLK. 001: PCLK1 = 1/2 HCLK. 010: PCLK1 = 1/4 HCLK. 011: PCLK1 = 1/8 HCLK. 100: PCLK1 = 1/16 HCLK. Others: Reserved.
[3]	**Reserved**	Reserved.
[2:0]	**APB0DIV**	**APB0 Clock DIvider** APB0 clock can be divided from HCLK 000: PCLK0 = HCLK. 001: PCLK0 = 1/2 HCLK. 010: PCLK0 = 1/4 HCLK. 011: PCLK0 = 1/8 HCLK. 100: PCLK0 = 1/16 HCLK. 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_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
CLKSFAIL	Reserved		HIRCSTB	LIRCSTB	Reserved	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:** Write 1 to clear the bit to 0.
[6:5]	**Reserved**	Reserved.
[4]	**HIRCSTB**	**HIRC Clock Source Stable Flag (Read Only)** 0 = Internal high speed RC oscillator (HIRC) clock is not stable or disabled. 1 = Internal high speed RC oscillator (HIRC) clock is stable and enabled.
[3]	**LIRCSTB**	**LIRC Clock Source Stable Flag (Read Only)** 0 = Internal low speed RC oscillator (LIRC) clock is not stable or disabled. 1 = Internal low speed RC oscillator (LIRC) clock is stable and enabled.
[2]	**Reserved**	Reserved.
[1]	**LXTSTB**	**LXT Clock Source Stable Flag (Read Only)** 0 = External low speed crystal oscillator (LXT) clock is not stable or disabled. 1 = External low speed crystal oscillator (LXT) clock is stabled and enabled.
[0]	**HXTSTB**	**HXT Clock Source Stable Flag (Read Only)** 0 = External high speed crystal oscillator (HXT) clock is not stable or disabled. 1 = External high speed crystal oscillator (HXT) clock is stable and enabled.

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		DIV1EN	CLKOEN	FREQSEL			

Bits	Description	
[31:6]	**Reserved**	Reserved.
[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 = External high speed crystal oscillator (HXT) clock frequency range detector fail interrupt Disabled. 1 = External high speed crystal oscillator (HXT) clock frequency range detector fail interrupt Enabled.
[16]	**HXTFQDEN**	**HXT Clock Frequency Range Detector Enable Bit** 0 = External high speed crystal oscillator (HXT) clock frequency range detector Disabled. 1 = External high speed crystal oscillator (HXT) clock frequency range detector Enabled.
[15:14]	**Reserved**	Reserved.
[13]	**LXTFIEN**	**LXT Clock Fail Interrupt Enable Bit** 0 = External low speed crystal oscillator (LXT) clock fail interrupt Disabled. 1 = External low speed crystal oscillator (LXT) clock fail interrupt Enabled.
[12]	**LXTFDEN**	**LXT Clock Fail Detector Enable Bit** 0 = External low speed crystal oscillator (LXT) clock fail detector Disabled. 1 = External low speed crystal oscillator (LXT) clock fail detector Enabled.
[11:6]	**Reserved**	Reserved.
[5]	**HXTFIEN**	**HXT Clock Fail Interrupt Enable Bit** 0 = External high speed crystal oscillator (HXT) clock fail interrupt Disabled. 1 = External high speed crystal oscillator (HXT) clock fail interrupt Enabled.
[4]	**HXTFDEN**	**HXT Clock Fail Detector Enable Bit** 0 = External high speed crystal oscillator (HXT) clock fail detector Disabled. 1 = 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 = External high speed crystal oscillator (HXT) clock frequency is normal. 1 = External high speed crystal oscillator (HXT) clock frequency is abnormal. **Note:** Write 1 to clear the bit to 0.
[7:2]	**Reserved**	Reserved.
[1]	**LXTFIF**	**LXT Clock Fail Interrupt Flag (Write Protect)** 0 = External low speed crystal oscillator (LXT) clock is normal. 1 = External low speed crystal oscillator (LXT) stops. **Note:** Write 1 to clear the bit to 0.
[0]	**HXTFIF**	**HXT Clock Fail Interrupt Flag (Write Protect)** 0 = External high speed crystal oscillator (HXT) clock is normal. 1 = External high speed crystal oscillator (HXT) clock stops. **Note:** Write 1 to clear the bit to 0.

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	
[31:10]	**Reserved**	Reserved.
[9:0]	**UPERBD**	**HXT Clock Frequency Range Detector Upper Boundary Value** The bits define the maximum value of frequency range detector window. When HXT frequency is higher than this maximum frequency value, the HXT Clock Frequency Range Detector Interrupt Flag will be 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	
[31:10]	**Reserved**	Reserved.
[9:0]	**LOWERBD**	**HXT Clock Frequency Range Detector Lower Boundary Value** The bits define the minimum value of frequency range detector window. When HXT frequency is lower than this minimum frequency value, the HXT Clock Frequency Range Detector Interrupt Flag will be set to 1.

Frequency out of range will be asserted when HIRC_period*1024> HXT_period*CLK_DUPB or HIRC_period*1024 < HXT_period*CLK_CDLOWB

HXT Filter Select Control Register (CLK_HXTFSEL)

Register	Offset	R/W	Description	Reset Value
CLK_HXTFSEL	CLK_BA+0xB4	R/W	HXT Filter 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**HXTFSEL**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**HXTFSEL**	**HXT Filter Select** 0 = HXT frequency is greater than12 MHz. 1 = HXT frequency is less than or equal to 12 MHz. **Note:** This bit should not be changed during HXT running.

6.4 Flash Memory Controller (FMC)

6.4.1 Overview

This chip is equipped with 16/32 Kbytes on-chip embedded Flash. A User Configuration block is provided for system initialization. A loader ROM (LDROM) is used for In-System-Programming (ISP) function. This chip also supports In-Application-Programming (IAP) function. User switches the code executing without the chip reset after the embedded Flash is updated.

6.4.2 Features

- Supports 16/32 Kbytes application ROM (APROM).
- Supports 512 bytes page size for 16/32 Kbytes Flash.
- Supports 2 Kbytes loader ROM (LDROM).
- Supports configurable Data Flash size to share with APROM.
- Supports 12 bytes User Configuration block to control system initialization.
- Supports 512 bytes page erase for all embedded Flash.
- Supports CRC-32 checksum calculation function.
- Supports In-System-Programming (ISP) / In-Application-Programming (IAP) to update embedded Flash memory.

6.4.3 Block Diagram

The Flash memory controller (FMC) consists of AHB slave interface, Flash control registers, Flash initialization controller, Flash operation control and embedded Flash memory. Figure 6.4-1 shows the block diagram of Flash memory controller.

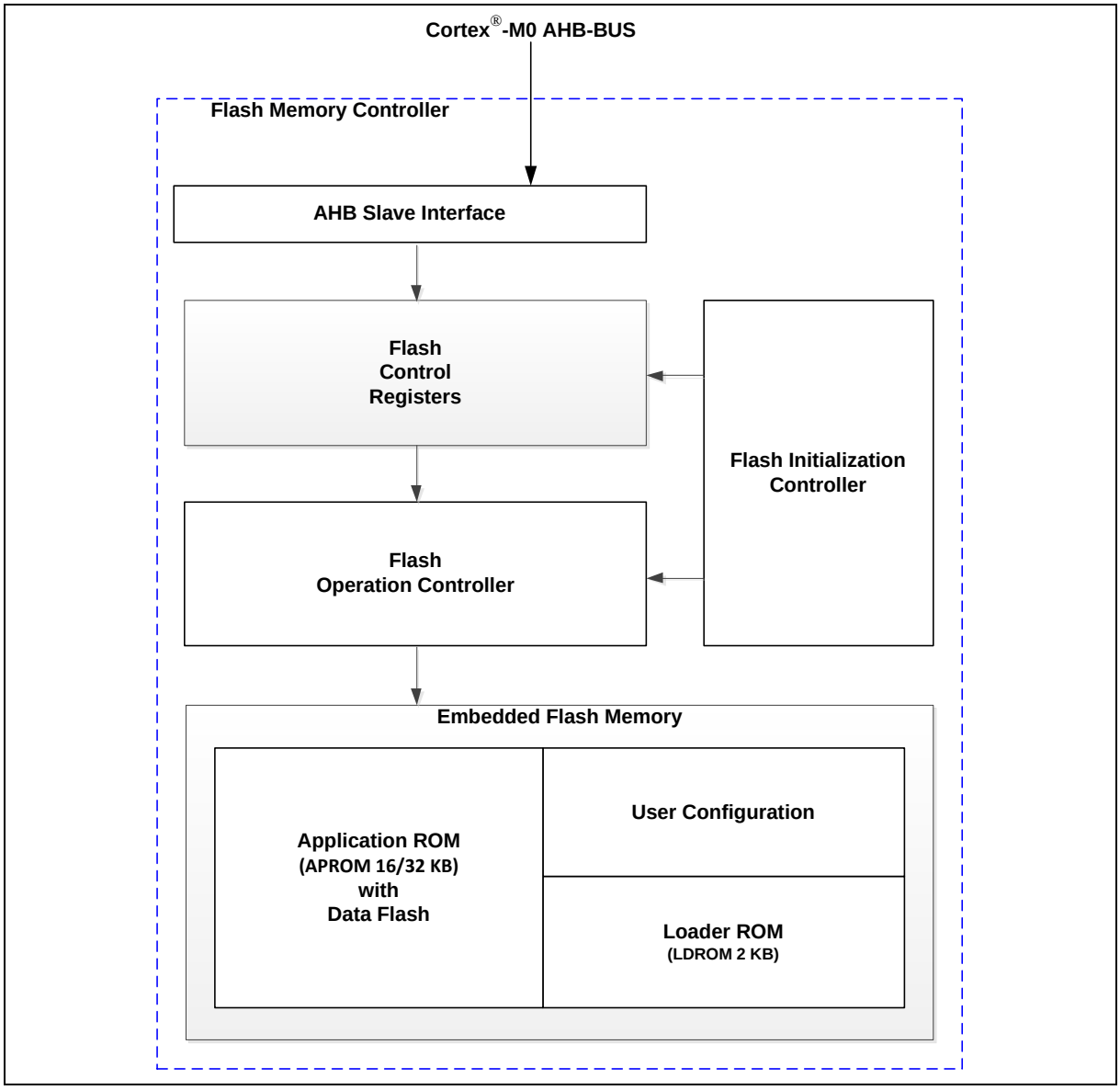


Figure 6.4-116/32 KB Flash Memory Control Block Diagram

AHB Slave Interface

There is single AHB slave interface in Flash memory controller for Cortex®-M0 to perform the instruction, data fetch and ISP control registers.

Flash Control Registers

All of ISP control and status registers are in the Flash control registers. Refer to the Register Description section for the detailed register description.

Flash Initialization Controller

When the chip is powered on or active from reset, the Flash initialization controller will start to access Flash automatically and check the Flash stability, and also reload User Configuration content to the Flash control registers as system initialization stage.

Flash Operation Controller

The Flash operations, such as checksum, Flash erase, Flash program, and Flash read operation, have

specific control timing for embedded Flash memory. The Flash operation controller generates those control timing by requirement from the Flash control registers and the Flash initialization controller.

Embedded Flash Memory

The embedded Flash memory is the main memory for user application code and parameters. It consists of the user configuration block, 2 Kbytes LDROM and 16/32 Kbytes APROM with Data Flash. The page erase Flash size is 512 bytes, and program bit width is 32 bits.

6.4.4 Functional Description

FMC functions include the memory organization, boot selection, IAP, ISP, the embedded Flash programming, and checksum calculation. The Flash memory map and system memory map are also introduced in the memory organization.

6.4.4.1 Memory Organization

The FMC memory consists of the embedded Flash memory. The embedded Flash memory is programmable, and includes APROM, LDROM, Data Flash and the User Configuration block. The address map includes Flash memory map and four system address maps: LDROM with IAP, LDROM without IAP, APROM with IAP, and APROM without IAP functions.

6.4.4.2 LDROM, APROM and Data Flash

LDROM is designed for a loader to implement In-System-Programming (ISP) function by user. LDROM is a 2 KB embedded Flash memory, the Flash address range is from 0x0010_0000 to 0x0010_07FF. APROM is main memory for user applications. The APROM size is 16/32 KB. Data Flash is used to store application parameters (not instruction). Data Flash is shared with APROM and its size is configurable. The base address of Data Flash is determined by DFBA (CONFIG1[19:0]). All of embedded Flash memory is 512 bytes page erased.

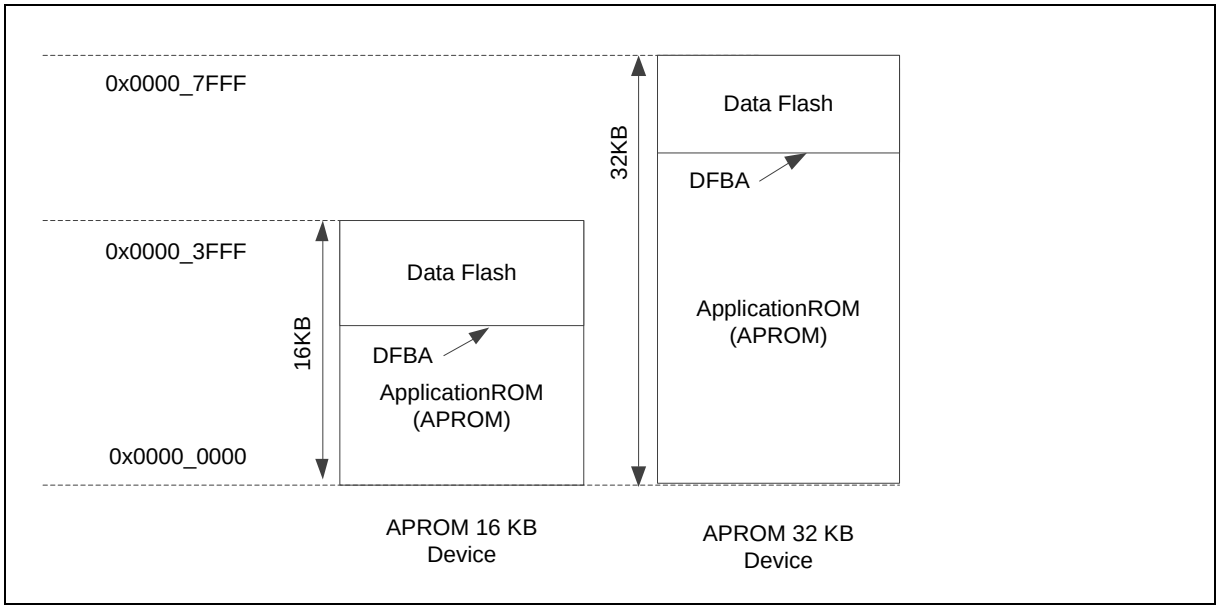


Figure 6.4-2 Data Flash Shared with APROM

6.4.4.3 User Configuration Block

User Configuration block is internal programmable configuration area for boot options, such as Flash security lock, boot selection, brown-out voltage level, and Data Flash base address. It works like a fuse

for power on setting. It is loaded from Flash memory to its corresponding control registers during chip power on. User can set these bits according to different application requirements. User Configuration block can be updated by ISP function and its address located at 0x0030_0000 with three 32 bits words (CONFIG0, CONFIG1 and CONFIG2). Any change on User Configuration block will take effect after system reboot

CONFIG0 (Address = 0x0030_0000)

31	30	29	28	27	26	25	24
CWDTEN[2]	CWDTPDEN	Reserved		CFGXT1	Reserved	CFGRPS	Reserved
23	22	21	20	19	18	17	16
Reserved	CBOV		CBORST	CBODEN	Reserved		
15	14	13	12	11	10	9	8
Reserved			ICELOCK	Reserved	CIOINI	RSTEXT	RSTWSEL
7	6	5	4	3	2	1	0
CBS		Reserved	CWDTEN[1:0]		Reserved	LOCK	DFEN

Bits	Description	
[31]	**CWDTEN[2]**	**Watchdog Timer Hardware Enable Bit** When the 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 in normal operation mode. However, in Power-down mode, the LIRC may be able to be disabled by setting CWDTPDEN=1 and LIRCEN=0 (CLK_PWRCTL[3]). **CWDTEN[2:0]** is CONFIG0[31][4][3], 011 = WDT hardware enable function is active. WDT clock is always on except chip enters Power- down mode. When chip enters Power-down mode, WDT clock is always on if CWDTPDEN is 0 or WDT clock is controlled by LIRCEN (CLK_PWRCTL[3]) if CWDTPDEN is 1. Please refer to bit field description of CWDTPDEN. 111 = WDT hardware enable function is inactive, WDT clock source only can be changed in this case. Others = WDT hardware enable function is active. WDT clock is always on.
[30]	**CWDTPDEN**	**Watchdog Clock Power-down Enable Bit** This bit should be used with CWDTEN. When WDT enabled by CWDTEN, user can use this bit to control WDT wakeup when system is in Power-down mode. If it is necessary to wakeup system by WDT, then user can set CWDTPDEN=0 to make sure WDT keep working at Power-down mode. If user don't want to wakeup system by WDT, user may just set CWDTPDEN=1 and LIRCEN=0 to let WDT suspend in power down. 0 = Watchdog Timer clock kept enabled when chip enters power-down. 1 = Watchdog Timer clock is controlled by LIRCEN (CLK_PWRCTL[3]) when the chip enters power-down. **Note:** This bit only works if CWDTEN[2:0] is set to 011.
[29:28]	**Reserved**	Reserved.
[27]	**CFGXT1**	**HXT Mode Selection** 0 = HXT works as external clock mode. PA.5 is configured as external clock input pin. 1 = HXT works as crystal mode. PA.4 and PA.5 are configured as external high speed crystal(HXT) pins. **Note:** When CFGXT1 = 0, P4.5 MFP should be set as GPIO mode. The DC characteristic of XT1_IN is the same as GPIO.
[26]	**Reserved**	Reserved.
[25]	**CFGRPS**	**Reset Pin Selection** 0 = Set GPA[3] to GPIO mode. 1 = Set GPA[3] to nReset mode.
[24:23]	**Reserved**	Reserved.

[22:21]	**CBOV**	**Brown-out Voltage Selection** 00 = Brown-out voltage is 2.3V. 01 = Brown-out voltage is 3.3V. 10 = Brown-out voltage is 3.7V. 11 = Brown-out voltage is 4.4V.
[20]	**CBORST**	**Brown-out Reset Enable Bit** 0 = Brown-out reset Enabled after power on or active from reset pin. 1 = Brown-out reset Disabled after power on or active from reset pin.
[19]	**CBODEN**	**Brown-out Detector Enable Bit** 0= Brown-out detect Enabled after powered on. 1= Brown-out detect Disabled after powered on.
[18:13]	**Reserved**	Reserved.
[12]	**ICELOCK**	**ICE Lock Bit** This bit is only used to disable ICE function. User may use it with LOCK (CONFIG0[1]) bit to increase security level. 0 = ICE function Disabled. 1 = ICE function Enabled.
[11]	**Reserved**	Reserved.
[10]	**CIOINI**	**I/O Initial State Selection** 0 = All GPIO set as Quasi-bidirectional mode after chip powered on or active from reset pin. 1 = All GPIO set as input tri-state mode after powered on or active from reset pin.
[9]	**RSTEXT**	**Chip Reset Time Extend** 0 = Extend reset time to 26.6 ms if chip release from power-on reset/LVR/BOD/nReset pin reset happened. 1 = Extend reset time to 3.2 ms if chip release from power-on reset/LVR/BOD/nReset pin reset happened.
[8]	**RSTWSEL**	**RST Pin Width Selection** 0 = nReset pin debounce width is 2 us. 1 = nReset pin debounce width is 32 us.
[7:6]	**CBS**	**Chip Booting Selection** When CBS[0] = 0 with IAP mode, 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. In other words, the code in LDROM and APROM can be called by each other. CBS value is valid. 00 = Boot from LDROM with IAP mode. 01 = Boot from LDROM without IAP mode. 10 = Boot from APROM with IAP mode. 11 = Boot from APROM without IAP mode. **Note:** BS (FMC_ISPCTL[1]) is only be used to control boot switching when CBS[0] = 1. VECMAP (FMC_ISPSTS[23:9]) is only used to remap 0x0~0x1FF when CBS[0] = 0.
[5]	**Reserved**	Reserved.
[4:3]	**CWDTEN[1:0]**	**Watchdog Timer Hardware Enable Bit** Please refer to CWDTEN[2] (CONFIG0[31]) for details.
[2]	**Reserved**	Reserved.

[1]	**LOCK**	**Security Lock Control** 0 = Flash memory content is locked. 1 = Flash memory content is unlocked if ALOCK (CONFIG2[7:0]) is also equal to 0x5A.
[0]	**DFEN**	**Data Flash Enable Bit** The Data Flash is shared with APROM, and the base address of Data Flash is decided by DFBA (CONFIG1[19:0]) when DFEN is 0. 0 = Data Flash Enabled. 1 = Data Flash Disabled.

CONFIG1 (Address = 0x0030_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	Description	
[31:20]	**Reserved**	Reserved.
[19:0]	**DFBA**	**Data Flash Base Address** This register works only when DFEN (CONFIG0[0]) is set to 0. If DFEN (CONFIG0[0]) is set to 0, the Data Flash base address is defined by user. Since on-chip Flash erase unit is 512 bytes, it is mandatory to keep bit 8-0 or bit 10-0 respectively as 0.

CONFIG2 (Address = 0x0030_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
ALOCK							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**ALOCK**	**Advance Security Lock Control** 0x5A = Flash memory content is unlocked if LOCK (CONFIG0[1]) is set to 1. Others = Flash memory content is locked. **Note:** ALOCK will be programmed as 0x5A after executing ISP page erase or ISP/ICP whole chip erase

6.4.4.4 Flash Memory Map

In the M0A21/M0A23, the Flash memory map is different from system memory map. The system memory map is used by CPU fetch code or data from FMC memory. The Flash memory map is used for ISP function to read, program or erase FMC memory. Figure 6.4-3 shows the Flash memory map.

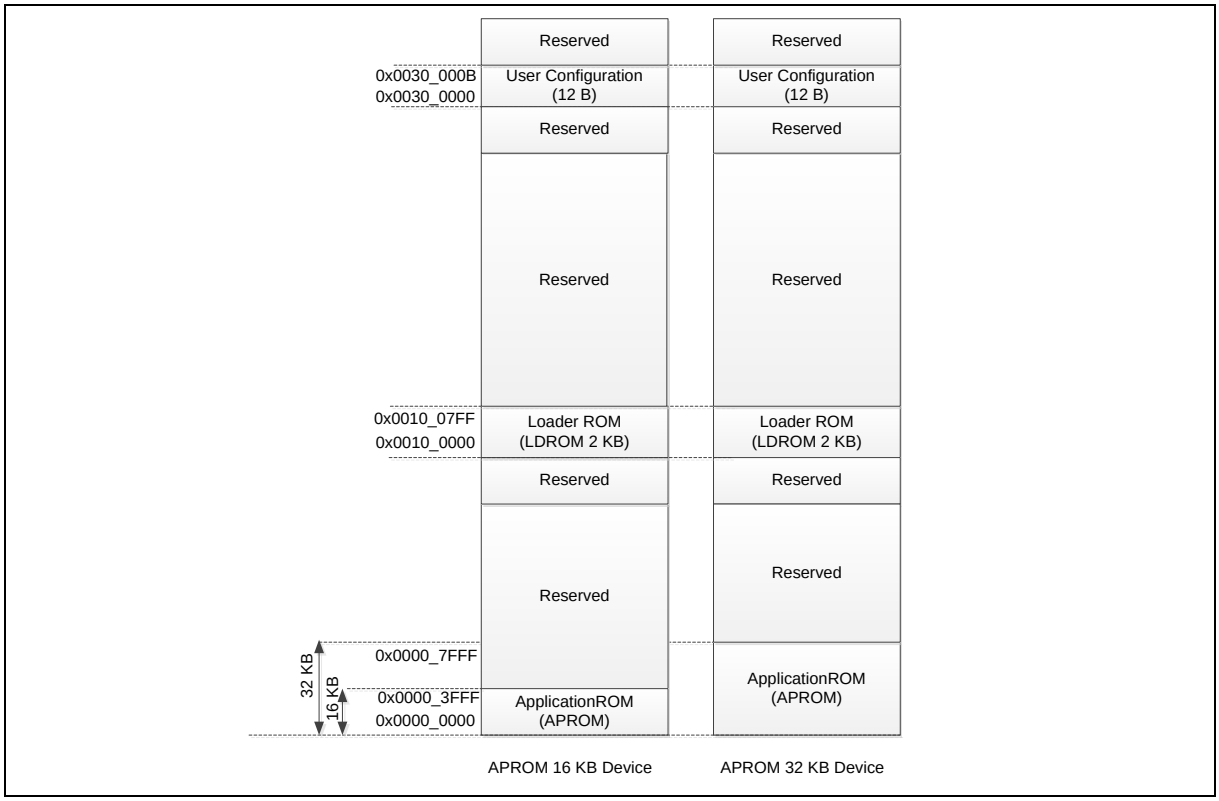


Figure 6.4-3 16/32 Kbytes Flash Memory Map

6.4.4.5 System Memory Map with IAP Mode

The system memory map is used by CPU to fetch code or data from FMC memory and LDROM(0x0010_0000~0x0010_07FF) address map are the same as in the Flash memory map. The Data Flash is shared with APROM and the Data Flash base address is defined by CONFIG1. The content of CONFIG1 is loaded into DFBA (Data Flash Base Address Register) at the Flash initialization. The DFBA~APROM maximum size is the Data Flash region for Cortex®-M0 data access, and 0x0000_0200~(DFBA-1) is APROM region for Cortex®-M0 instruction access.

The address from 0x0000_0000 to 0x0000_01FF is called system memory vector. 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.

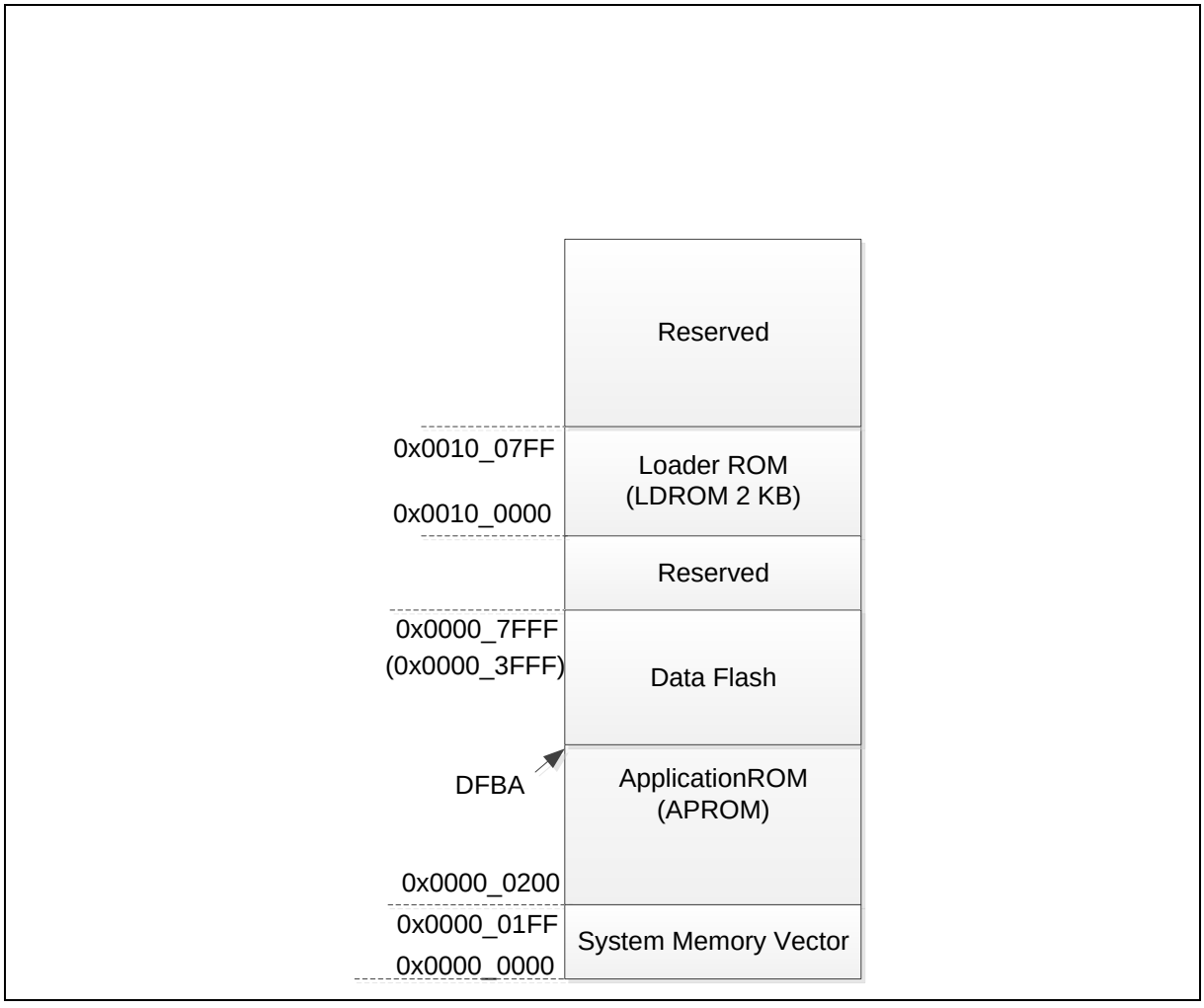


Figure 6.4-4 16/32 Kbytes Flash System Memory Map with IAP Mode

In LDROM with IAP mode, the default value of {VECMAP[11:0], 9'h000} is 0x100000 and first page of LDROM (0x0010_0000 ~ 0x0010_01FF) is mapping to the system memory vector for Cortex®-M0 instruction or data access.

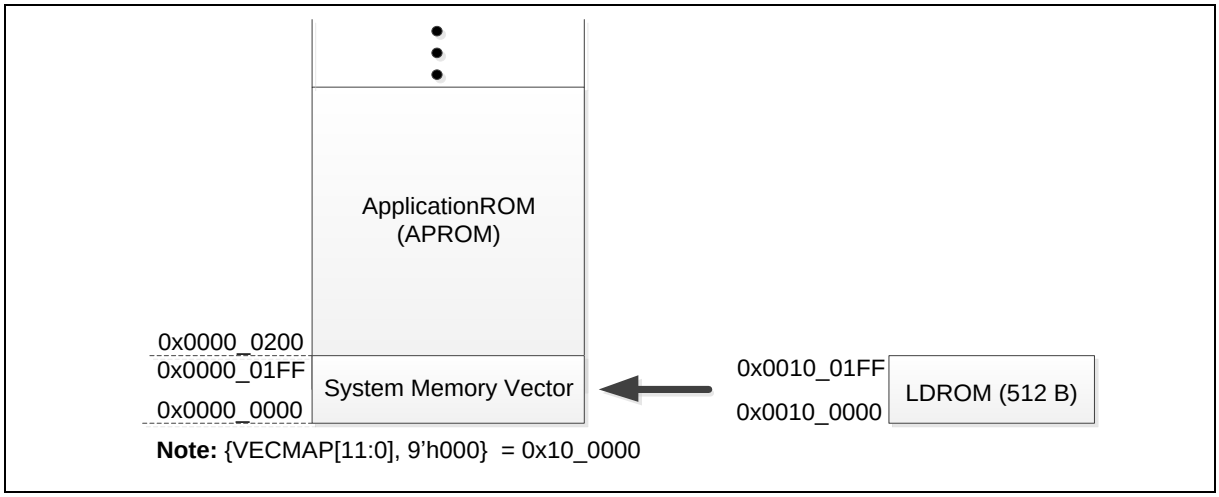


Figure 6.4-5 LDROM with IAP Mode

In APROM with IAP mode, the default value of {VECMAP[11:0], 9'h000} is 0x000000 and first page of APROM (0x0000_0000~0x0000_01FF) is mapping to the system memory vector for Cortex®-M0 instruction or data access.

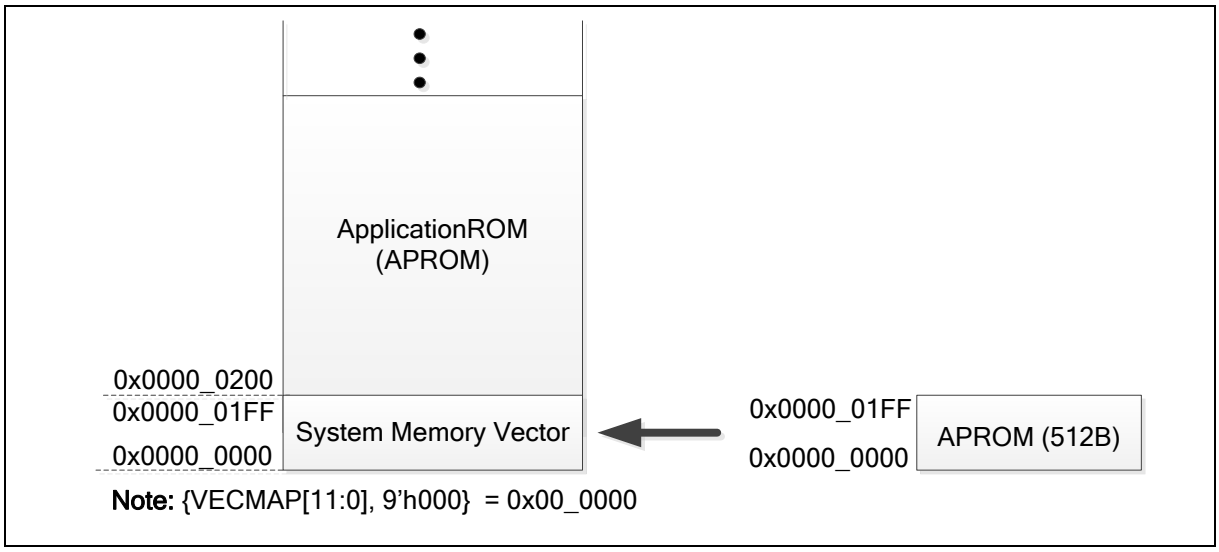


Figure 6.4-6 APROM with IAP Mode

In system memory map with IAP mode, APROM and LDROM can remap to the system memory vector when CPU running. User can write the target remap address to FMC_ISPADDR register and then trigger ISP procedure with the "Vector Page Remap" command (0x2E). In VECMAP (FMC_ISPSTS[23:9]), shows the finial system memory vector mapping address.

6.4.4.6 System Memory Map without IAP Mode

In system memory map without IAP mode, the system memory vector mapping is not supported. There are two kinds of system memory map without IAP mode when chip booting: (1) LDROM without IAP, (2) APROM without IAP. In LDROM without IAP mode, LDROM base is mapping to 0x0000_0000. CPU program cannot run to access APROM. In APROM without IAP mode, APROM base is mapping to 0x0000_0000. CPU program cannot run to access LDROM. The Data Flash is shared with APROM and the Data Flash base address is defined by CONFIG1. The content of CONFIG1 is loaded into DFBA (Data Flash Base Address Register) at the Flash initialization. The DFBA~0x0000_7FFF is the Data Flash region for Cortex®-M0 data access, and 0x0000_0000~(DFBA-1) is APROM region for Cortex®-

M0 instruction access.

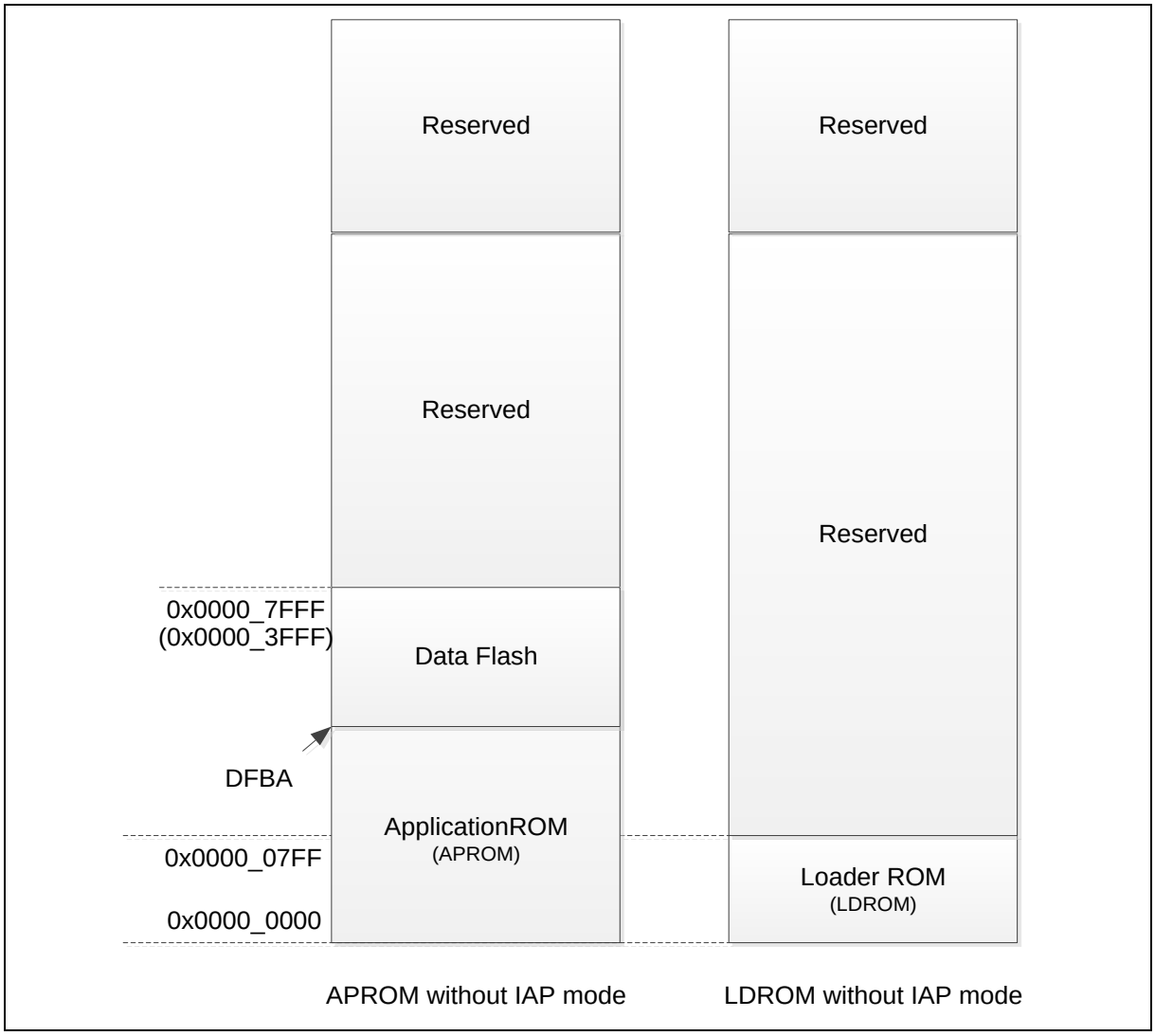


Figure 6.4-7 16/32 Kbytes Flash System Memory Map without IAP Mode

6.4.4.7 Boot Selection

The M0A21/M0A23 provides four booting modes for application field. They are LDROM with IAP, LDROM without IAP, APROM with IAP, and APROM without IAP. The booting modes and system memory map are setting by CBS (CONFIG0[7:6]).

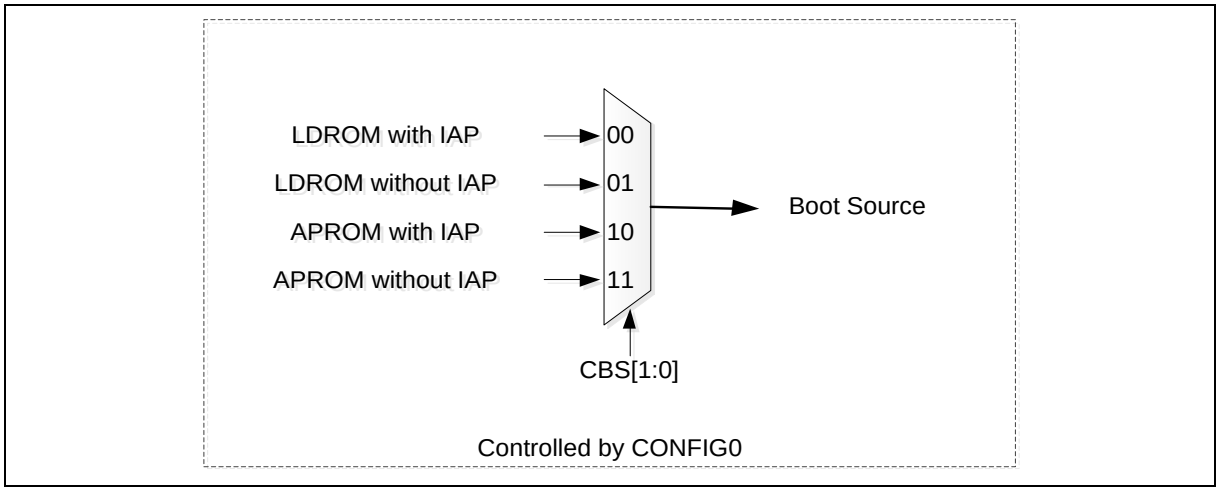


Figure 6.4-8 Boot Source Selection

CBS[1:0]	Boot Selection/System Memory Map	Vector Mapping Supporting
00	LDROM with IAP	Yes
01	LDROM without IAP	No
10	APROM with IAP	Yes
11	APROM without IAP	No

Table 6.4-1 Vector Mapping Support

6.4.4.8 In-Application-Programming (IAP)

The M0A21/M0A23 provides In-Application-Programming (IAP) function for user to switch the code executing between APROM and LDROM. User can enable the IAP function by booting chip and setting the chip boot selection bits in CBS (CONFIG0[7:6]) as 10 or 00.

When chip boots with IAP function enabled, any executable code (align to 512 bytes) is allowed to map to the system memory vector(0x0000_0000~0x0000_01FF) any time. User can change the remap address to FMC_ISPADDR and then trigger ISP procedure with the “Vector Page Remap” command.

6.4.4.9 In-System-Programming (ISP)

The M0A21/M0A23 supports In-System-Programming (ISP) function allowing the embedded Flash 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, I^2C, and SPI.

The M0A21/M0A23 ISP provides the following functions for embedded Flash memory.

- Supports Flash page erase function
- Supports Flash data program function
- Supports Flash data read function
- Supports company ID read function
- Supports unique ID read function
- Supports memory checksum calculation function

- Supports system memory vector remap function

ISP Commands

ISP Command	FMC_ISPCMD	FMC_ISPADDR	FMC_ISPDAT
FLASH Page Erase	0x22	Valid address of Flash memory origination. It must be 512 bytes page alignment.	N/A
FLASH 32-bit Program	0x21	Valid address of Flash memory origination	FMC_ISPDAT: Programming Data
FLASH 32-bit Read	0x00	Valid address of Flash memory origination. It must be word alignment.	FMC_ISPDAT: Return Data
Read Company ID	0x0B	0x0000_0000	FMC_ISPDAT: 0x0000_00DA
Read Checksum	0x0D	Keep address of “Run Checksum Calculation”	FMC_ISPDAT: Return Checksum
Run Checksum Calculation	0x2D	Valid start address of memory origination It must be 512 bytes page alignment	FMC_ISPDAT: Size It must be 512 bytes alignment
Read Unique ID	0x04	0x0000_0000	FMC_ISPDAT: Unique ID Word 0
		0x0000_0004	FMC_ISPDAT: Unique ID Word 1
		0x0000_0008	FMC_ISPDAT: Unique ID Word 2
		0x0000_0070	FMC_ISPDAT[11:0]: Built-in VBG ADC conversion result
Vector Remap	0x2E	Valid address in APROM or LDROM. It must be 512 bytes alignment	N/A

Table 6.4-2 ISP Command List

ISP Procedure

The FMC controller provides embedded Flash memory read, erase and program operation. Several control bits of FMC control register are write-protected, thus it is necessary to unlock before setting.

After unlocking the protected register bits, user needs to set the FMC_ISPCTL control register to decide to update LDROM, APROM or User Configuration block, and then set ISPEN (FMC_ISPCTL[0]) to enable ISP function.

Once the FMC_ISPCTL register is set properly, user can set FMC_ISPCMD (refer to above ISP command list) for specify operation. Set FMC_ISPADDR for target Flash memory based on Flash memory origination. FMC_ISPDAT can be used to set the data to program or used to return the read data according to FMC_ISPCMD.

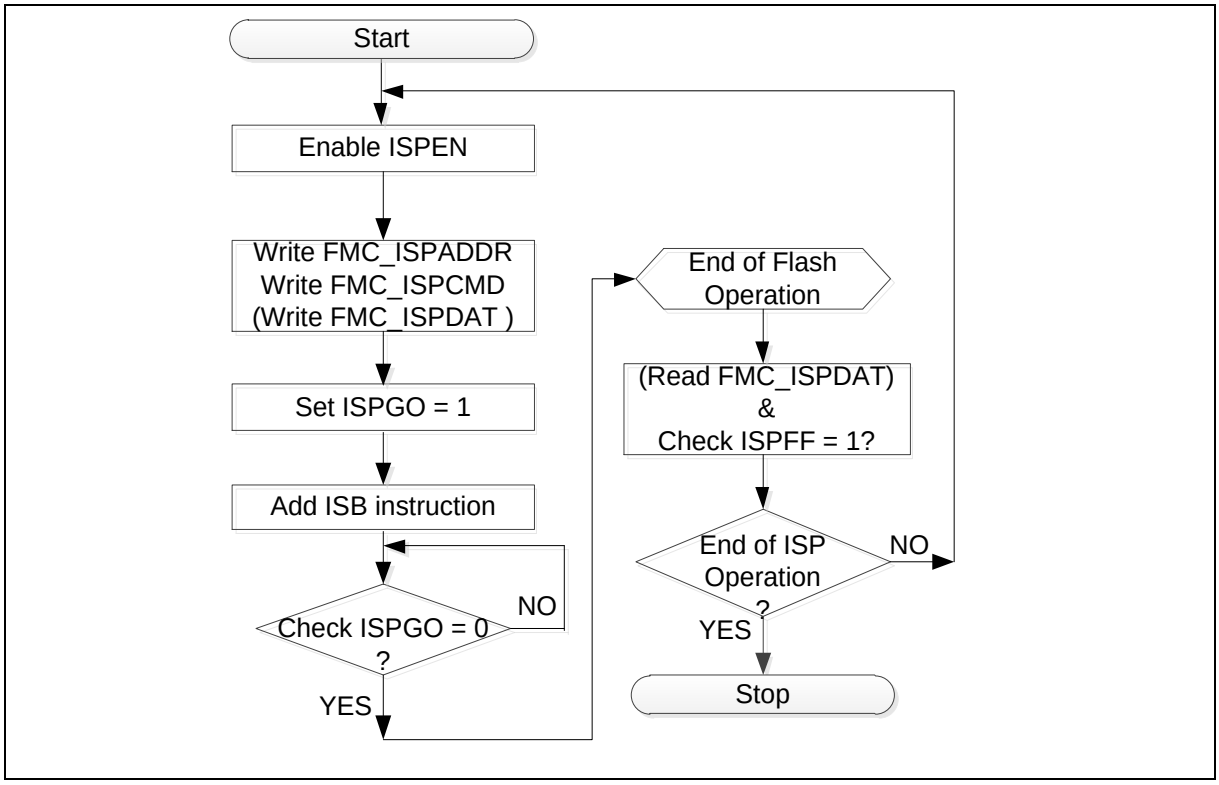


Figure 6.4-9 ISP Procedure Example

Finally, set ISPGO (FMC_ISPTRG[0]) register to perform the relative ISP function. The ISPGO(FMC_ISPTRG[0]) bit is self-cleared when ISP function has been done. To make sure ISP function has been finished before CPU goes ahead, ISB (Instruction Synchronization Barrier) instruction is used right after ISPGO(FMC_ISPTRG[0]) setting.

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(FMC_ISPSTS[6]) flag can only be cleared by software. The next ISP procedure can be started even ISPFF(FMC_ISPSTS[6]) bit is kept as 1. Therefore, it is recommended to check the ISPFF(FMC_ISPSTS[6]) bit and clear it after each ISP operation if it is set to 1.

When the ISPGO(FMC_ISPTRG[0]) bit is set, 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(FMC_ISPTRG[0]) bit. User should add ISB (Instruction Synchronization Barrier) instruction next to the instruction in which ISPGO (FMC_ISPTRG[0]) bit is set 1 to ensure correct execution of the instructions following ISP operation.

6.4.4.10 VECMAP for Interrupt and Memory Programming

Accelerate interrupt by VECMAP

In IAP mode, VECMAP function could be used to map 512 bytes SRAM to vector map space. It means it is possible to store all exception vectors to SRAM. Then, if any exceptions assert, CPU can read exception handler from SRAM with zero wait state to speed up exception latency.

Because the vector map space is fixed to be 512 bytes, user must copy all 512 bytes to SRAM before remapping SRAM to vector map space. Otherwise, CPU may get wrong data from vector map space after remapping. Figure 6.4-10 shows an example to accelerating interrupt by VECMAP.

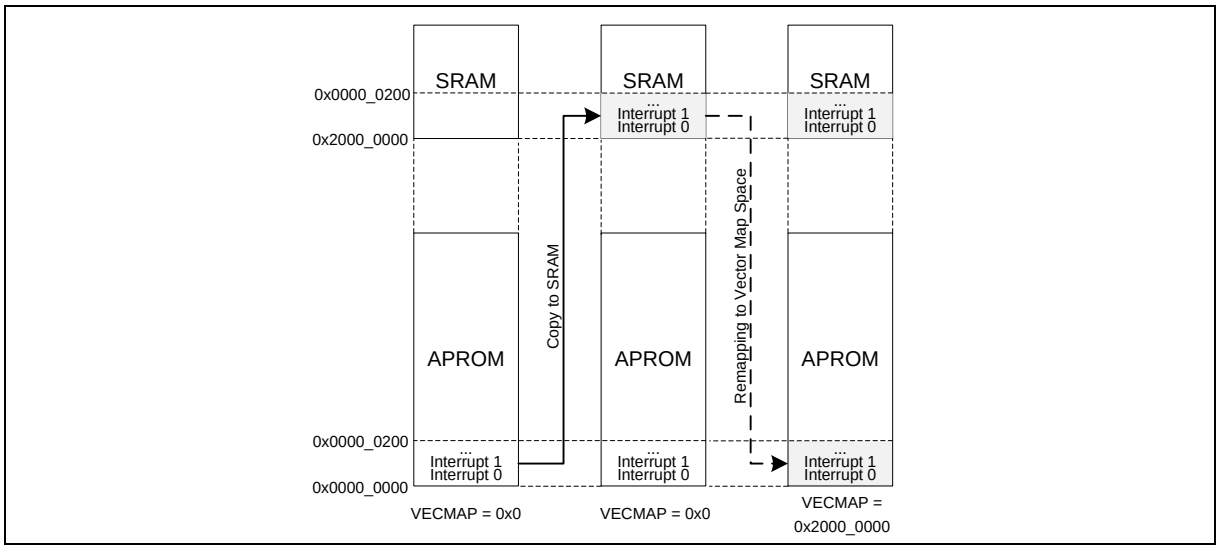


Figure 6.4-10 Example for Accelerating Interrupt by VECMAP

Avoid CPU halt when Flash programming

When Flash memory controller is busy, any CPU access to Flash memory will cause CPU halt for waiting Flash controller ready. If Flash controller is busy in page erasing, it may cause CPU halt for a long time to erase pages. To avoid this situation, user needs to avoid CPU access Flash memory when page erasing. The easiest way is to execute code in SRAM and use VECMAP to map all exceptions to SRAM. By executing code in SRAM, CPU will not access Flash to get instructions. By mapping all exceptions to SRAM, all interrupts will not need to get exception handler from Flash memory.

6.4.4.11 Embedded Flash Memory Programming

This chip provides 32-bit Flash memory programming function to updated procedure. Table 6.4-3 lists required FMC control registers in each embedded Flash programming function.

Register	Description	32-Bit Programming
FMC_ISPCTL	ISP Control Register	●
FMC_ISPADDR	ISP Address Register	●
FMC_ISPDAT	ISP Data Register	●
FMC_ISPCMD	ISP Command Register	0x21
FMC_ISPTRG	ISP Trigger Control Register	●
FMC_ISPSTS	ISP Status Register	●

Table 6.4-3 FMC Control Registers for Flash Programming

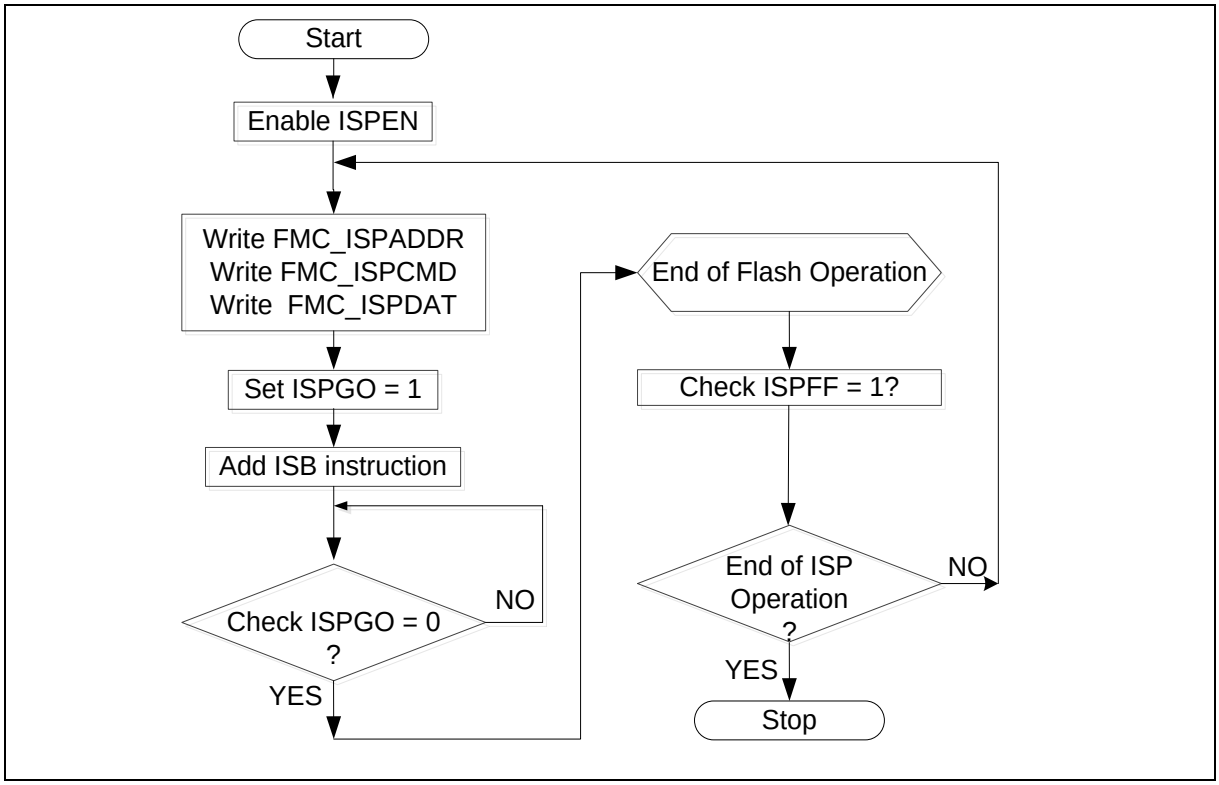


Figure 6.4-11 ISP 32-bit Programming Procedure

6.4.4.12 CRC32 Checksum Calculation

The M0A21/M0A23 supports the Cyclic Redundancy Check (CRC-32) checksum calculation function to help user quickly check the memory content includes APROM and LDROM. The CRC-32 polynomial is as below.

CRC-32: $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$

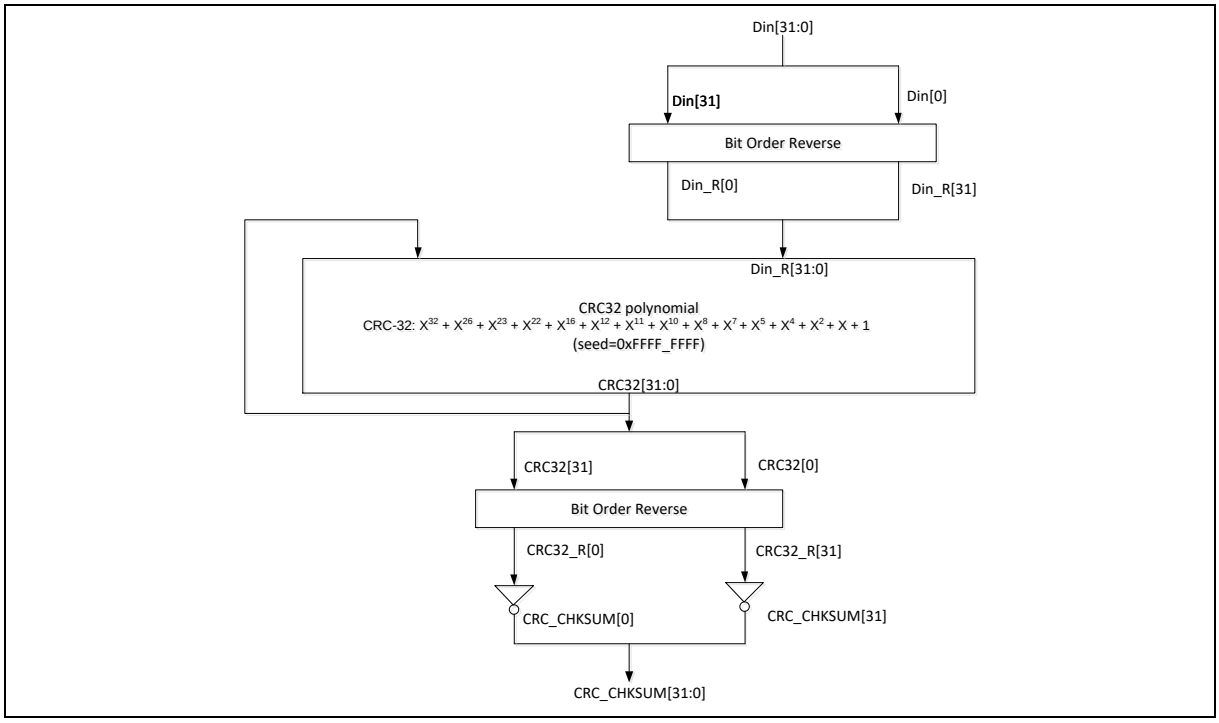


Figure 6.4-12 CRC-32 Checksum Calculation

The following three steps complete the CRC-32 checksum calculation.

1. Perform ISP “Run Memory Checksum” operation: user has to set the memory starting address (FMC_ISPADDR) and size (FMC_ISPDAT) to calculate. Both address and size have to be 512 bytes alignment, the size must be multiples of 512 bytes and the starting address includes APROM and LDROM.
2. Perform ISP “Read Memory Checksum” operation: the FMC_ISPADDR should be kept the same as step 1.
3. Read FMC_ISPDAT to get checksum: The checksum is read from FMC_ISPDAT. If the checksum is 0x0000_0000, It must be one of two conditions (1) Checksum calculation is in-progress, (2) Address and size is over device limitation.

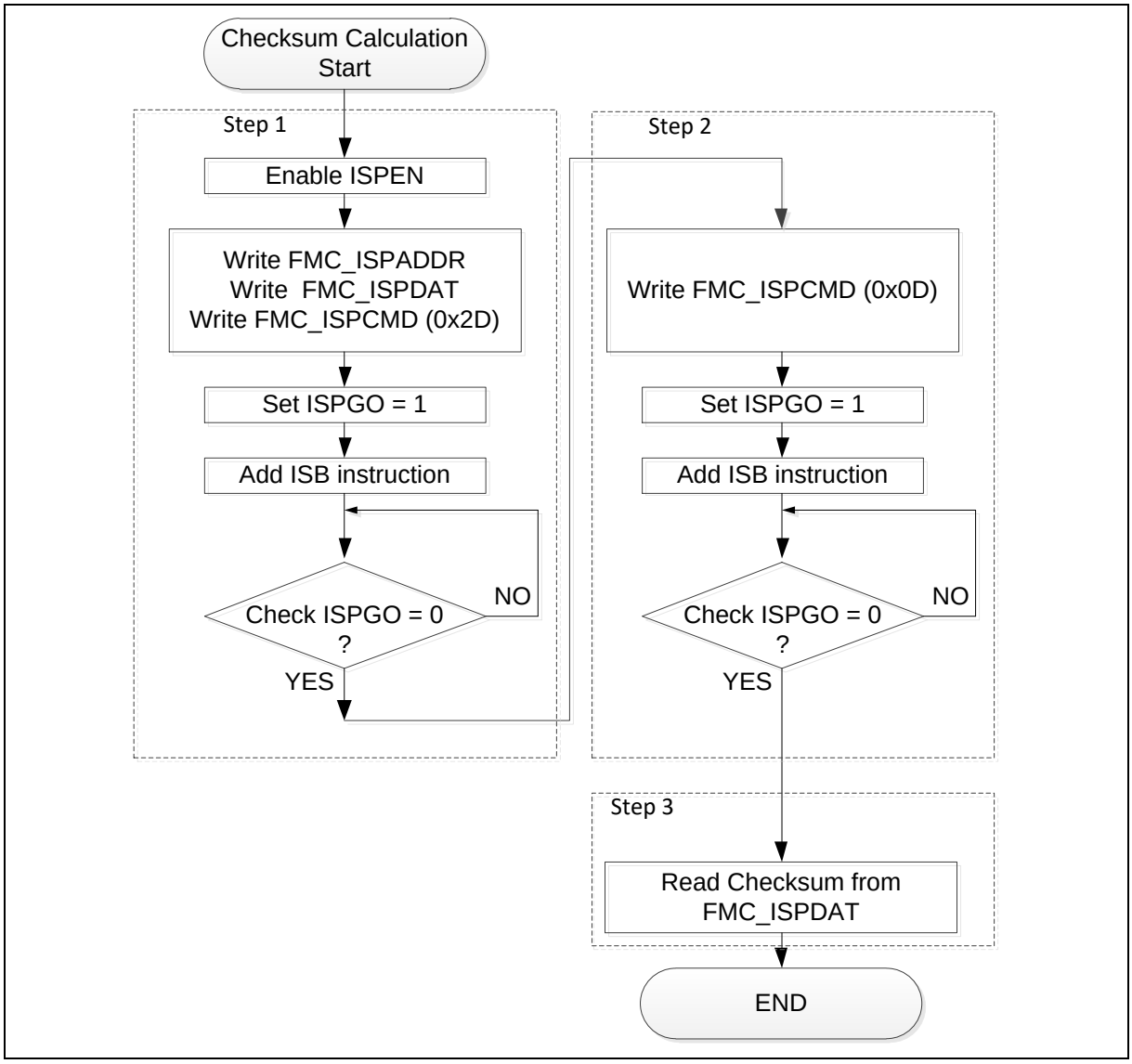


Figure 6.4-13 CRC-32 Checksum Calculation Flow

6.4.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
FMC Base Address: **FMC_BA = 0x4000_C000**				
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control Register	0x0000_000X
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000
FMC_DFBA	FMC_BA+0x14	R	Data Flash Base Address	0xXXXX_XXXX
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP Status Register	0xX0X0_000X

6.4.6 Register Description

ISP Control Register (FMC_ISPCTL)

Register	Offset	R/W	Description	Reset Value
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control 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	**ISPFF**	**LDUEN**	**CFGUEN**	**APUEN**	**Reserved**	**BS**	**ISPEN**

Bits	Description	
[31:7]	**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 erased/programmed if CFGUEN is set to 0. ● Page Erase command at LOCK mode with ICE connection. ● Erase or Program command at brown-out detected. ● Destination address is illegal, such as over an available range. ● Invalid ISP commands. This bit needs to be cleared by writing 1 to it. **Note:** This bit is write-protected. Refer to the SYS_REGLCTL register.
[5]	**LDUEN**	**LDROM Update Enable Bit (Write Protect)** LDROM update enable bit. 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 Selection (Write Protect)** Set/clear this bit to select next booting from LDROM/APROM, respectively. This bit also functions as chip booting status flag, which can be used to check where chip booted from. This bit is initiated with the inversed value of CBS[1] (CONFIG0[7]) after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened 0 = Booting from APROM. 1 = Booting from LDROM. **Note:** This bit is write-protected. Refer to the SYS_REGLCTL register.
[0]	**ISPEN**	**ISP Enable Bit (Write Protect)** 0 = ISP function Disabled. 1 = ISP function Enabled. **Note:** This bit is write-protected. Refer to the SYS_REGLCTL register.

ISP Address (FMC_ISPADDR)

Register	Offset	R/W	Description	Reset Value
FMC_ISPADDR	FMC_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 M0A21/M0A23 is equipped with embedded Flash. ISPADDR[1:0] must be kept 00 for ISP 32-bit operation. For CRC32 Checksum Calculation command, this field is the Flash starting address for checksum calculation, 512 bytes alignment is necessary for checksum calculation. 16/32 Kbytes Flash: ISPADR[8:0] must be kept all 0 for Vector Page Remap Command.

ISP Data Register (FMC_ISPDAT)

Register	Offset	R/W	Description	Reset Value
FMC_ISPDAT	FMC_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. For Run Checksum Calculation command, ISPDAT is the memory size (byte) and 512 bytes alignment. For Read 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 (FMC_ISPCMD)

Register	Offset	R/W	Description	Reset Value
FMC_ISPCMD	FMC_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 CMD** ISP command table is shown below: 0x00= Flash 32-bit Read. 0x04= Read Unique ID. 0x0B= Read Company ID. 0x0D= Read CRC32 Checksum. 0x21= Flash 32-bit Program. 0x22= Flash Page Erase. 0x2D= Run CRC32 Checksum Calculation. 0x2E= Vector Remap. The other commands are invalid.

ISP Trigger Control Register (FMC_ISPTRG)

Register	Offset	R/W	Description	Reset Value
FMC_ISPTRG	FMC_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 Flash Base Address Register (FMC_DFBA)

Register	Offset	R/W	Description	Reset Value
FMC_DFBA	FMC_BA+0x14	R	Data Flash 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 Flash Base Address** This register indicates Data Flash start address. It is a read only register. The Data Flash is shared with APROM. the content of this register is loaded from CONFIG1. This register is valid when DFEN (CONFIG0[0]) = 0.

ISP Status Register (FMC_ISPSTS)

Register	Offset	R/W	Description	Reset Value
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP Status Register	0xX0X0_000X

31	30	29	28	27	26	25	24
Reserved		**VECMAP**					
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**
Reserved	**ISPFF**	**Reserved**			**CBS**		**ISPBUSY**

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:9]	**VECMAP**	**Vector Page Mapping Address (Read Only)** All access to 0x0000_0000~0x0000_01FF is remapped to the Flash memory or SRAM address {VECMAP[20:0], 9'h000} ~ {VECMAP[20:0], 9'h1FF}. VECMAP [20:19] = 00 system vector address is mapped to Flash memory. VECMAP [20:19] = 10 system vector address is mapped to SRAM memory. VECMAP [18:12] should be 0.
[8:7]	**Reserved**	Reserved.
[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is the mirror of ISPFF (FMC_ISPCTL[6]), it needs to be cleared by writing 1 to FMC_ISPCTL[6] or FMC_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 erased/programmed if CFGUEN is set to 0. ● Page Erase command at LOCK mode with ICE connection ● Erase or Program command at brown-out detected ● Destination address is illegal, such as over an available range. ● Invalid ISP commands
[5:3]	**Reserved**	Reserved.
[2:1]	**CBS**	**Boot Selection of CONFIG (Read Only)** This bit is initiated with the CBS (CONFIG0[7:6]) after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened. 00 = LDROM with IAP mode. 01 = LDROM without IAP mode. 10 = APROM with IAP mode. 11 = APROM without IAP mode.

[0]	**ISPBUSY**	**ISP BUSY (Read Only)** 0 = ISP operation is finished. 1 = ISP operation is busy.

6.5 General Purpose I/O (GPIO)

6.5.1 Overview

This chip has up to 26 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 26 pins are arranged in 4 ports named as PA, PB, PC, PD. PA has 6 pins on port, PB has 4 pins on port. PC and PD have 8 pins on port. Each 26 pins is independent and has the corresponding register bits to control the pin mode function and data, except GPA[3]. GPA[3] is a input pin.

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. After the chip is reset, the I/O mode of all pins are depending on CIOINI (CONFIG0[10]). Each I/O pin has a very weakly individual pull-up resistor which is about 50 kΩ. Please refer to the M0A21/M0A23 Datasheet for detailed pin operation voltage information about V_{DD} electrical characteristics.

6.5.2 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
- Configurable default I/O mode of all pins after reset by CIOINI (CONFIG0[10]) setting
  - CIOINI = 0, all GPIO pins in Quasi-bidirectional mode after chip reset
  - CIOINI = 1, all GPIO pins in input mode after chip reset
- Supports independent pull-up control
- Enabling the pin interrupt function will also enable the wake-up function

6.5.3 Block Diagram

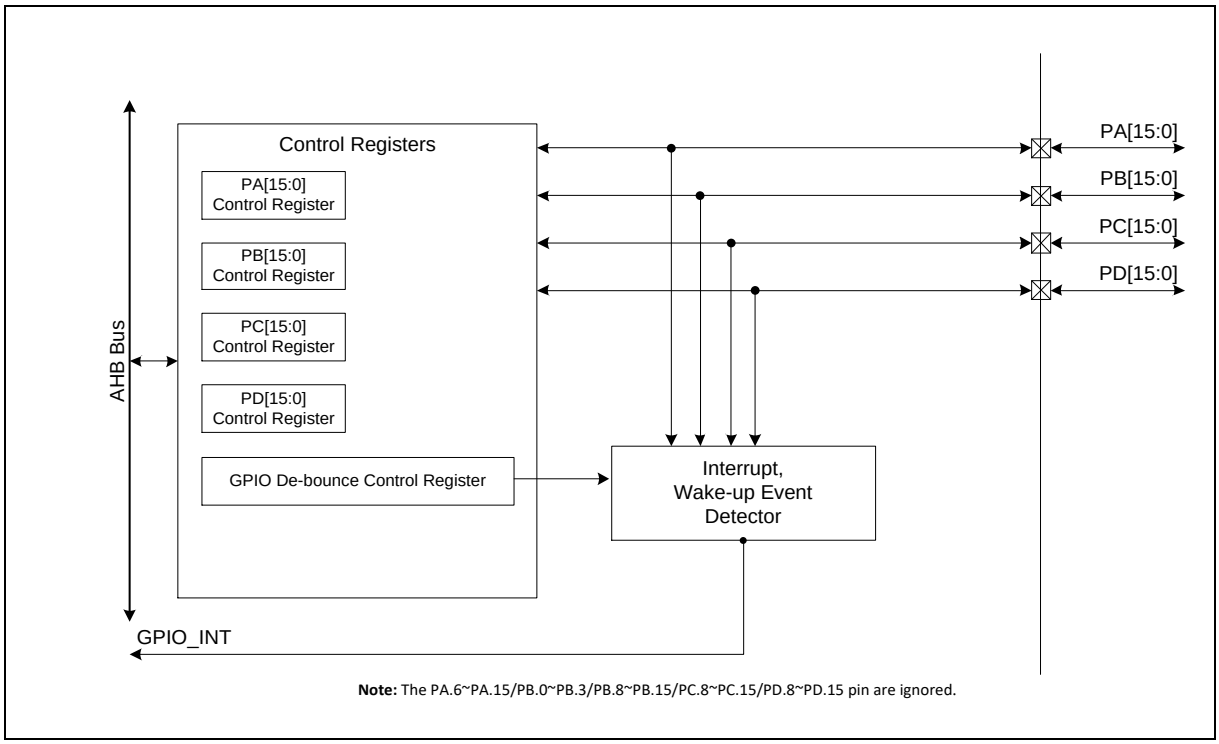


Figure 6.5-1 GPIO Controller Block Diagram

Note: The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~PC.15/PD.8~PD.15 pins are not avaliable.

6.5.4 Basic Configuration

- Reset configuration
  - Reset GPIO in GPIORST (SYS_IPRST1[1])
- Pin configuration

Group	Pin Name	GPIO	MFP
INT0	INT0	PA.3, PB.5	MFP30
INT1	INT1	PC.5	MFP30
INT2	INT2	PC.4	MFP30
INT3	INT3	PC.3	MFP30
INT4	INT4	PC.6	MFP30
INT5	INT5	PC.7	MFP30

6.5.5 Functional Description

6.5.5.1 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.

Each I/O pin includes an internal resistor. Set (Px_PUSEL[n]) to 1 to enable internal pull-up resistor.

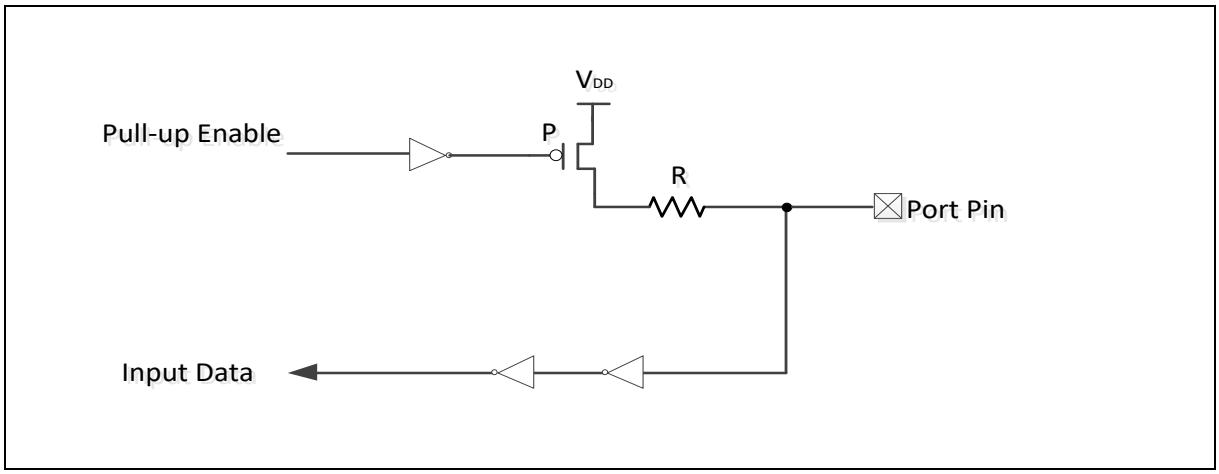


Figure 6.5-2 Input Mode

6.5.5.2 Push-pull Output Mode

Figure 6.5-3 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.

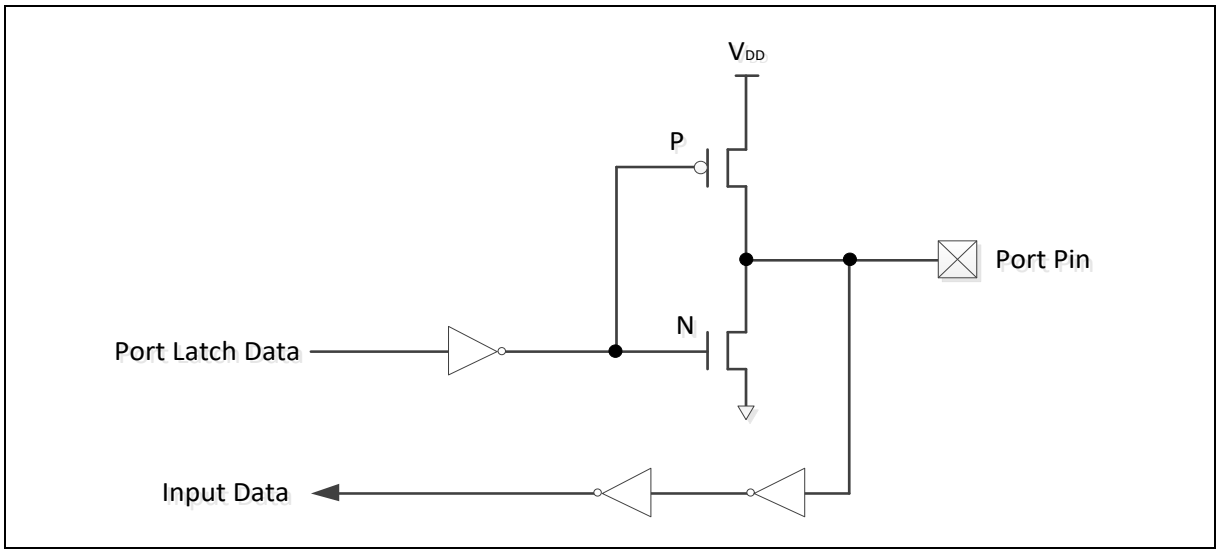


Figure 6.5-3 Push-Pull Output

6.5.5.3 Open-drain Mode

Figure 6.5-4 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.

Each I/O pin includes an internal resistor. Set (Px_PUSEL[2n+1:2n]) to 01 to enable internal pull-up resistor. Internal pull-up resistor is available in Open-drain mode.

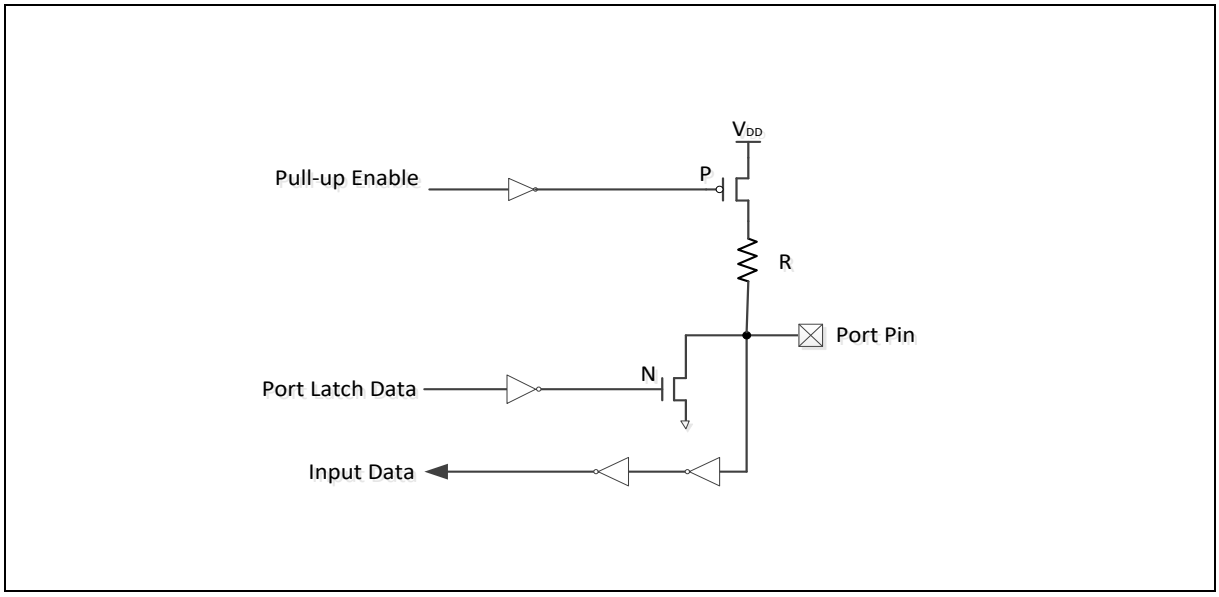


Figure 6.5-4 Open-Drain Output

6.5.5.4 Quasi-bidirectional Mode

Figure 6.5-5 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,please refer to the M0A21/M0A23 Datasheet for detailed information.

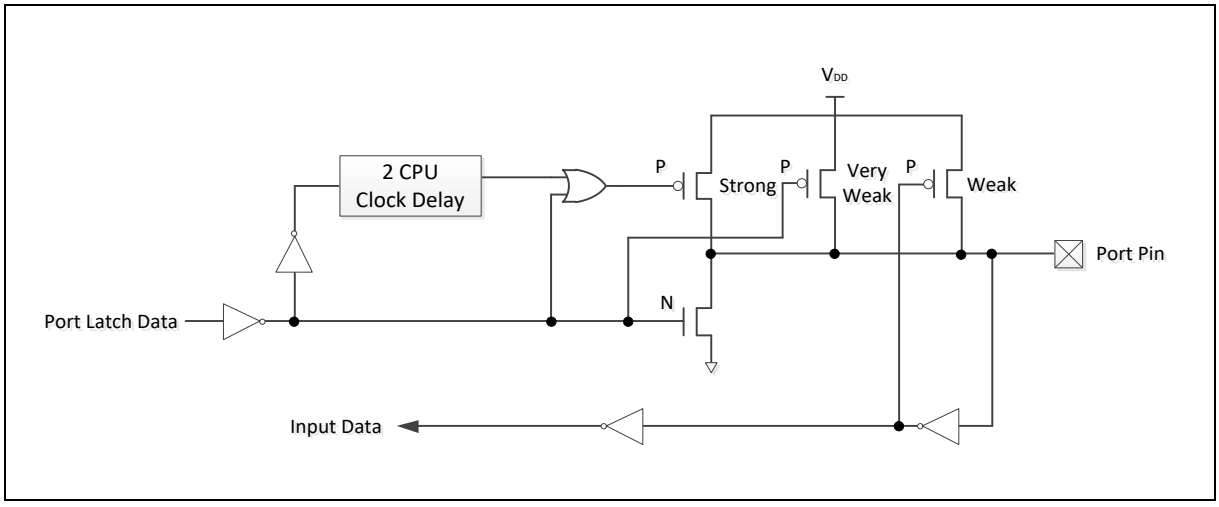


Figure 6.5-5 Quasi-Bidirectional I/O Mode

6.5.5.5 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. The GPIO can also be the chip wake-up source when chip enters Idle/Power-down mode. The setting of wake-up trigger condition is the same as GPIO interrupt trigger.

6.5.5.6 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 support 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. 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 (38.4 kHz) by setting DBCLKSRC (GPIO_DBCTL[4]) register. And DBCLKSEL (GPIO_DBCTL[3:0]) register can control sampling cycle period.

Figure 6.5-6 shows GPIO rising edge trigger interrupt. The interval of time between the two valid sample signal is determined by DBCLKSRC (GPIO_DBCTL[4]) and DBCLKSEL (GPIO_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 6.5-7 shows the situation is opposite to rising edge trigger.

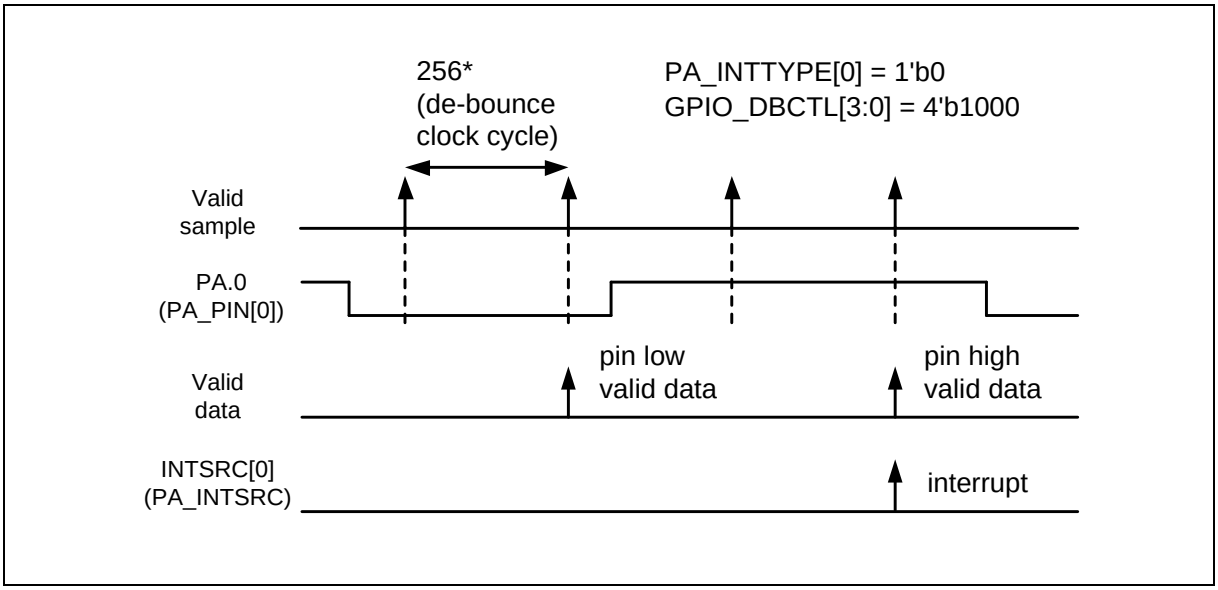


Figure 6.5-6 GPIO Rising Edge Trigger Interrupt

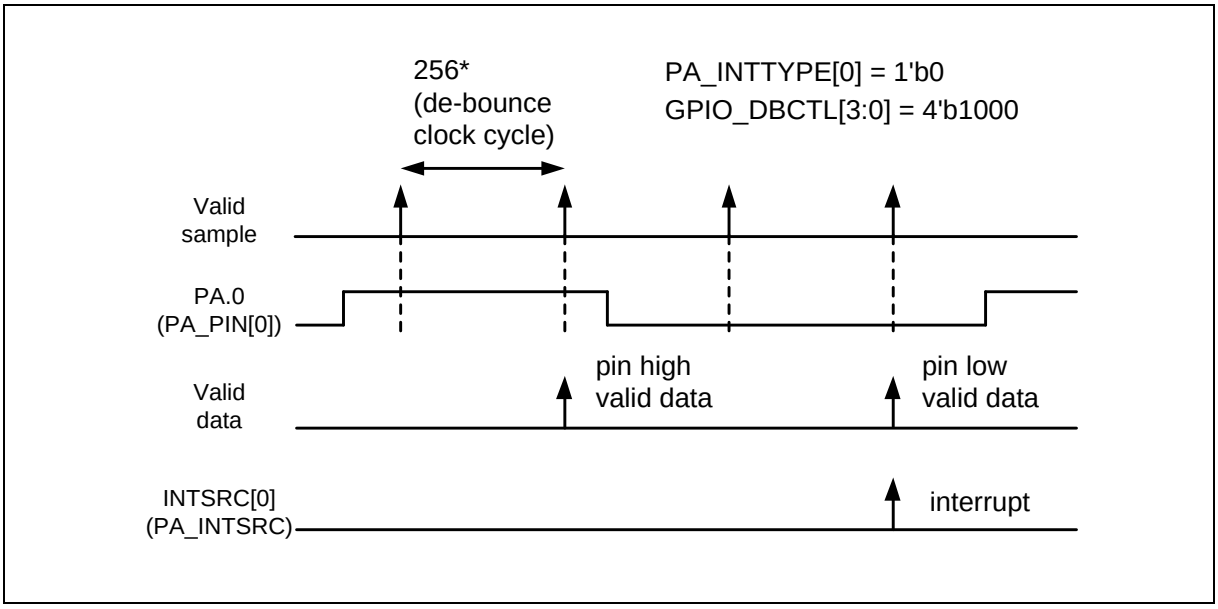


Figure 6.5-7 GPIO Falling Edge Trigger Interrupt

6.5.5.7 GPIO Digital Input Path Disable Control

User can disable GPIO digital input path by setting DINOFF (Px_DINOFF[n+16]). When GPIO digital input path is disabled, the digital input pin value PIN (Px_PIN[n]) is tied to low. By the way, the GPIO digital input path is force disabled by hardware and DINOFF control is useless when I/O function configure as ADC/ACMP/ext. XTL

6.5.5.8 GPIO Group Clock On-Off Control

User can disable GPIO group clock by setting GPIO_CLKON. When GPxOn is diabled, the group clock is Off, the GPIO register and pin control and PDIO circiut are not workable. Only GPIO_DBCTL and GPIO_CLKON can be wrote.

6.5.6 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	0xXXXX_XXXX
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_0037
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_PUSEL	GPIO_BA+0x030	R/W	PA Pull-up Selection Register	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0xXXXX_XXXX
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_00F0
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_PUSEL	GPIO_BA+0x070	R/W	PB Pull-up Selection Register	0x0000_0000
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0xXXXX_XXXX
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_00FF
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_PUSEL	GPIO_BA+0x0B0	R/W	PC Pull-up Selection Register	0x0000_0000
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0xXXXX_XXXX
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_00FF
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_PUSEL	GPIO_BA+0x0F0	R/W	PD Pull-up Selection Register	0x0000_0000
GPIO_DBCTL	GPIO_BA+0x440	R/W	Interrupt De-bounce Control Register	0x000F_0000
GPIO_CLKON	GPIO_BA+0x444	R/W	GPIO Clock On-off Register	0x0000_000F
PAn_PDIO n=0,1..5	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO n=3..7	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO n=0,1..7	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO n=0,1..7	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.n Pin Data Input/Output Register	0x0000_000X

6.5.7 Register Description

Port A-D 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	0xXXXX_XXXX
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0xXXXX_XXXX
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0xXXXX_XXXX
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0xXXXX_XXXX

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	**MODE**	**Port A-D 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 1:** The initial value of this field is defined by CIOINI (CONFIG0 [10]). If CIOINI is set to 0, the default value is 0xFFFF_FFFF and all pins will be quasi-bidirectional mode after chip powered on. If CIOINI is set to 1, the default value is 0x0000_0000 and all pins will be input mode after chip powered on. **Note 2:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective. **Note 3:** The GPA.3 is a input pin. **Note 4:** If MFOS is enabled then GPIO mode setting is ignored.

Port A-D 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

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**	**Port A-D 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:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective.
[15:0]	**Reserved**	Reserved.

Port A-D Data Output Value (Px_DOUT)

Register	Offset	R/W	Description	Reset Value
PA_DOUT	GPIO_BA+0x008	R/W	PA Data Output Value	0x0000_0037
PB_DOUT	GPIO_BA+0x048	R/W	PB Data Output Value	0x0000_00F0
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_00FF
PD_DOUT	GPIO_BA+0x0C8	R/W	PD 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**
DOUT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DOUT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DOUT**	**Port A-D 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 1:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective. **Note 2:**The GPA.3 is a input pin.

Port A-D 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

31	30	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**	**Port A-D 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 ineffective. 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:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective.

Port A-D 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

31	30	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**	**Port A-D Pin[n] Pin Value** Each bit of the register reflects the actual status of the respective Px.n pin. 0 = The corresponding pin status is low. 1 = The corresponding pin status is high. **Note:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective.

Port A-D 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

31	30	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**	**Port A-D 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 ineffective. **Note:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective.

Port A-D 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

31	30	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**	**Port A-D 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 has no effect and no interrupt 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 ineffective. **Note:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective.

Port A-D 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

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**	**Port A-D 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:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective.
[n] n=0,1..15	**FLIEN**	**Port A-D 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:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective.

Port A-D 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

31	30	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**	**Port A-D 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:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective.

Port A-D 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

31	30	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**	**Port A-D Pin[n] Input Schmitt Trigger Enable Bit** 0 = Px.n input schmitt trigger function Disabled. 1 = Px.n input schmitt trigger function Enabled. **Note:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective.

Port A-D Pull-up Selection Register (Px_PUSEL)

Register	Offset	R/W	Description	Reset Value
PA_PUSEL	GPIO_BA+0x030	R/W	PA Pull-up Selection Register	0x0000_0000
PB_PUSEL	GPIO_BA+0x070	R/W	PB Pull-up Selection Register	0x0000_0000
PC_PUSEL	GPIO_BA+0x0B0	R/W	PC Pull-up Selection Register	0x0000_0000
PD_PUSEL	GPIO_BA+0x0F0	R/W	PD Pull-up Selection Register	0x0000_0000

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

Bits	Description	
[n] n=0,1..15	**PUSEL**	**Port A-D Pin[n] Pull-up Enable Register** Determine each I/O Pull-up of Px.n pins. 0 = Px.n pull-up disable. 1 = Px.n pull-up enable. **Note 1:** The independent pull-up control register only valid when MODEn set as input and open-drain mode even if I/O function is switched to multi-function pin. Ex: UARTx_RXD. **Note2:** The PA.3/PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective.

Interrupt De-bounce Control Register (GPIO_DBCTL)

Register	Offset	R/W	Description	Reset Value
GPIO_DBCTL	GPIO_BA+0x440	R/W	Interrupt De-bounce Control Register	0x000F_0000

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16] x=A,B,C,D	**ICLKONx**	**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. 1 = All I/O pins edge detection circuit is always active after reset. **Note:** It is recommended to disable this bit to save system power if no special application concern. Each bit control each GPIO group.
[15:5]	**Reserved**	Reserved.
[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 38.4 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.

GPIO Clock On-off Register (GPIO_CLKON)

Register	Offset	R/W	Description	Reset Value
GPIO_CLKON	GPIO_BA+0x444	R/W	GPIO Clock On-off Register	0x0000_000F

31	30	29	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				GPDOn	GPCOn	GPBOn	GPAOn

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**GPxOn**	**GPIO Group Clock On-off** The GPIO port clock can be disabled to reduce power consumption by setting GPIO_CLKON if the GPIO port isn't used. When GPxOn is set to 0 to diable GPIO port clock, the GPIO register, pin control and PDIO function are not workable. Only GPIO_CLKON and GPIO_DBCTL register can be updated 0 = Disable GPIO group clock, include register & pin control & PDIO circuit. 1 = Enable GPIO group clock, include register & pin control & PDIO circuit.

GPIO Px.n Pin Data Input/Outut Register (Pxn_PDIO)

Register	Offset	R/W	Description	Reset Value
PAn_PDIO n=0,1..5	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO n=4,5,6,7	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO n=0,1..7	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO n=0,1..7	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.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:** The PA.6~PA.15/PB.0~PB.3/PB.8~PB.15/PC.8~15/PD.8~15 pin is ineffective. **Note 3:**The GPA.3 is a input pin.

6.6 PDMA Controller (PDMA)

6.6.1 Overview

The peripheral direct memory access (PDMA) controller is used to provide high-speed data transfer. The PDMA controller 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. The PDMA controller has a total of 5 channels and each channel can perform transfer between memory and peripherals or between memory and memory.

6.6.2 Features

- Supports 5 independently configurable channels
- 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 UART, USCI, ADC, PWM, DAC and TIMER 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 time-out function on channel 0 and channel1

6.6.3 Block Diagram

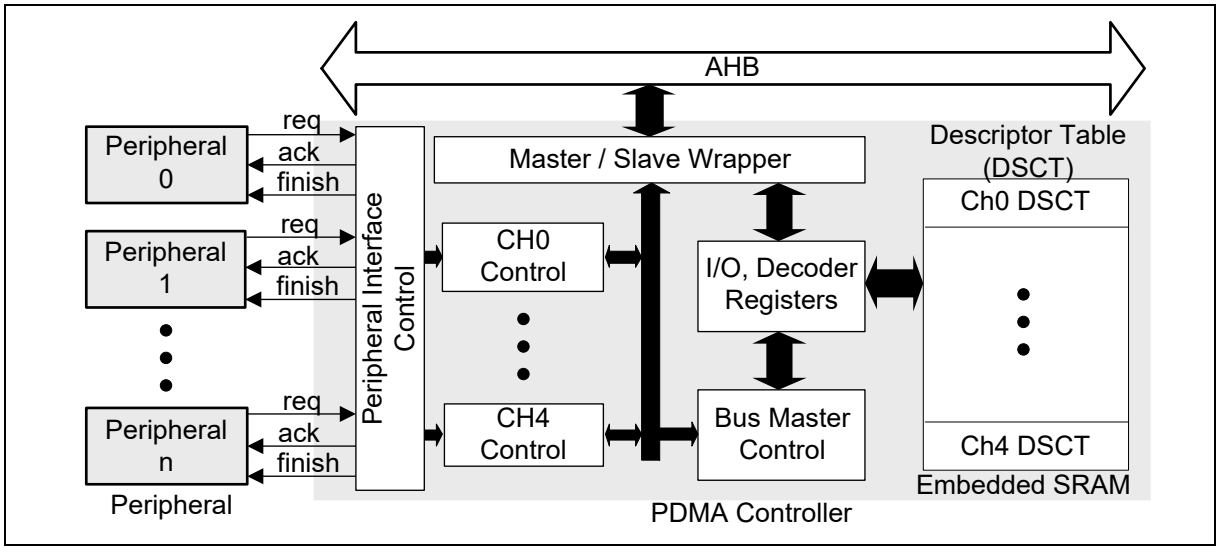


Figure 6.6-1 PDMA Controller Block Diagram

6.6.4 Basic Configuration

- Clock Source Configuration
  - Enable PDMA controller clock in PDMACKEN (CLK_AHBCLK [1]).
- Reset Configuration
  - Reset PDMA controller in PDMARST (SYS_IPRST0[2]).

6.6.5 Functional Description

The PDMA controller transfers data from one address to another without CPU intervention. The PDMA controller supports 5 independent channels and serves only one channel at one time, as the result, PDMA controller supports two level channel priorities: fixed and round-robin priority, PDMA controller serves channel in order from highest to lowest priority channel. The PDMA controller supports two operation modes: Basic mode and Scatter-gather mode. Basic mode is used to perform one descriptor table transfer. Scatter-gather mode has more entries for each PDMA channel, and thus the PDMA controller 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 6.6-2 shows the diagram of descriptor table (DSCT) data structure.

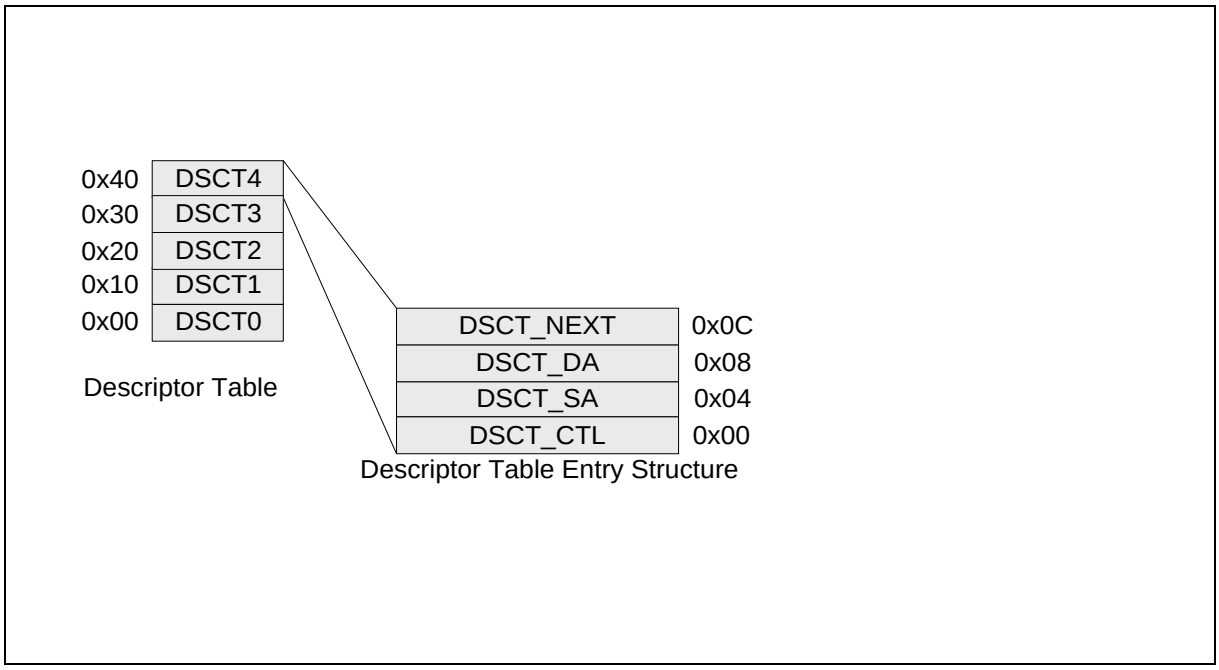


Figure 6.6-2 Descriptor Table Entry Structure

The PDMA controller also supports single and burst transfer type and the request source can be from software or peripheral request, transfer between memory to memory using 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).

6.6.5.1 Channel Priority

The PDMA controller 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 6.6-1.

PDMA_PRISET	Channel Number	Priority Setting	Arbitration Priority In Descending Order
1	4	Channel4, Fixed Priority	Highest
1	3	Channel3, Fixed Priority	---
---	---	---	---
1	0	Channel0, Fixed Priority	---

0	4	Channel4, Round-Robin Priority	---
0	3	Channel3, Round-Robin Priority	---
---	---	---	---
0	0	Channel0, Round-Robin Priority	Lowest

Table 6.6-1 Channel Priority Table

6.6.5.2 PDMA Operation Mode

The PDMA controller supports two operation modes including Basic mode and Scatter-Gather mode.

Basic Mode

Basic mode is used to perform one descriptor table transfer mode. This mode can be used to transfer data between memory and memory, peripherals and memory or peripherals and peripherals, but if user want to transfer data between peripherals and peripherals, one thing must be sured is that the request from peripherals knows that the data is ready for transfer or not. PDMA controller operation mode can be set from OPMODE (PDMA_DSCTn_CTL[1:0], n denotes PDMA channel), the default setting is in idle state (OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x0) 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 the PDMA controller 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[4:0]) is enabled and the operation mode will be updated to idle state as shown in Figure 6.6-3. If software configures the operation mode to idle state, the PDMA controller 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.

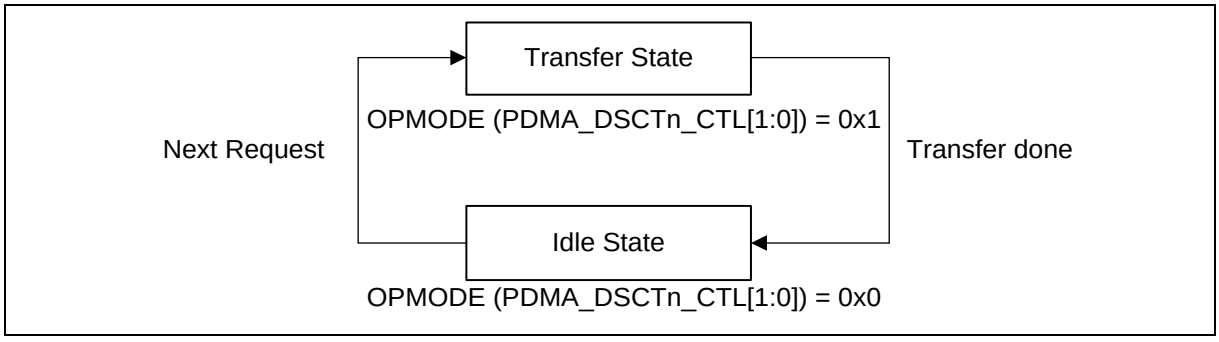


Figure 6.6-3 Basic Mode Finite State Machine

Scatter-Gather Mode

Scatter-Gather mode is a complex mode and can perform sophisticated transfer through the use of the description link list table as shown in Figure 6.6-4. Through operation mode user can perform peripheral wrapper-around, and multiple PDMA task can be used for 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 entries task till the last task with OPMODE (PDMA_DSCTn_CTL[1:0]) is idle state without ack. It also means scatter-gather mode can be used to transfer data between memory to memory without handshaking.

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[4:0]) flag will be asserted and if this bit is “1” TDIFn will not be active).

If channel 4 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 16bits without last two bits [1:0] valid in PDMA_DSCTn_NEXT), and then the next DSCT entry start address is 0x2000_0100.

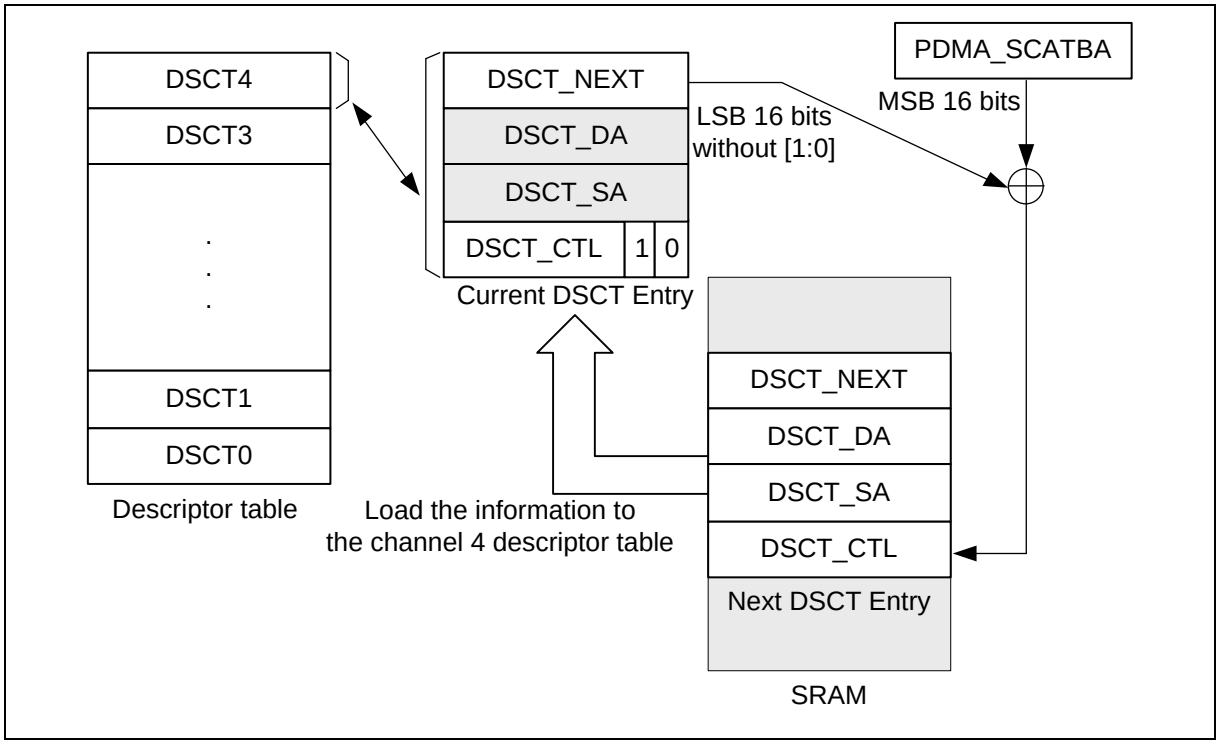


Figure 6.6-4 Descriptor Table Link List Structure

The above link list table operation is DSCT state in Scatter-Gather Mode as shown in Figure 6.6-5. 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.

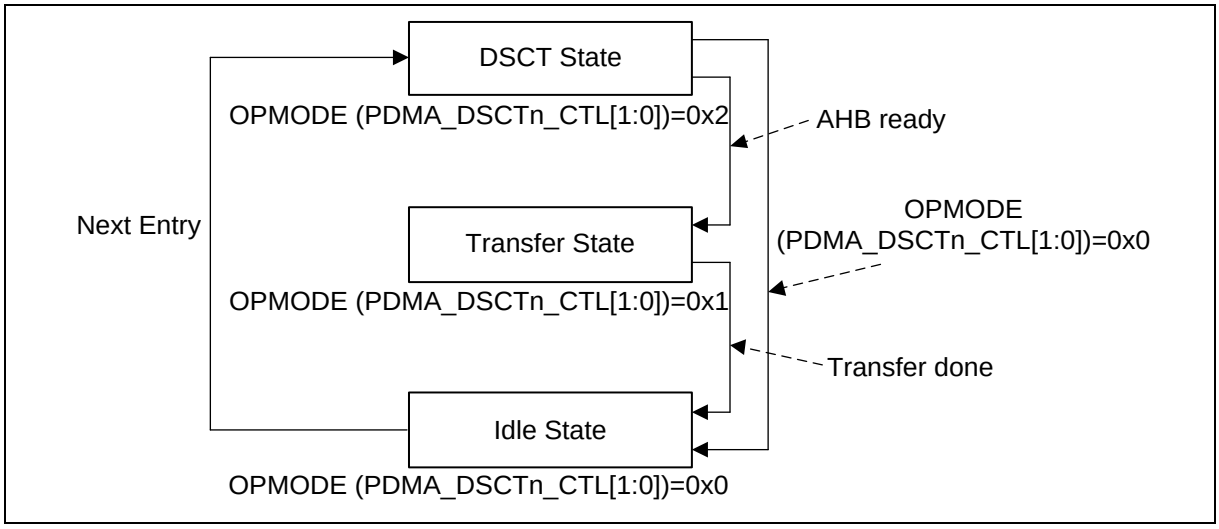


Figure 6.6-5 Scatter-Gather Mode Finite State Machine

6.6.5.3 Transfer Type

The PDMA controller supports two transfer types: single transfer type and burst transfer type, configure by setting TXTYPE (PDMA_DSCTn_CTL[2]).

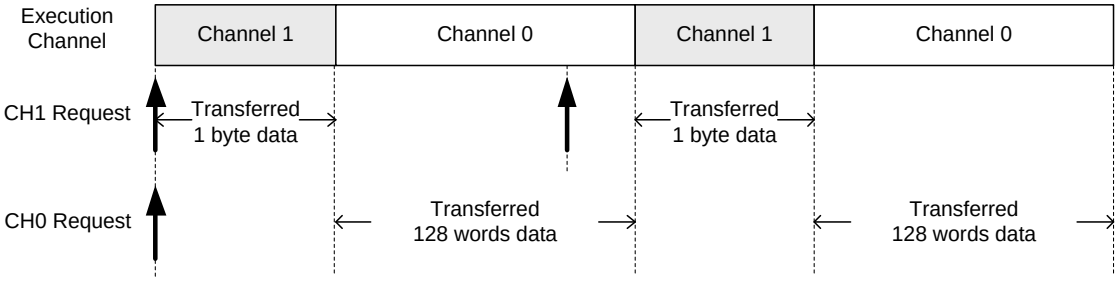
When the PDMA controller is operated in single transfer type, each transfer data needs one request signal for one transfer, after transferred data, TXCNT (PDMA_DSCTn_CTL[31:16]) will decrease 1. Transfer will be finished after the TXCNT (PDMA_DSCTn_CTL[31:16]) decreases to 0. In this mode, the BURSIZE (PDMA_DSCTn_CTL[6:4]) is not useful to control the transfer size. The BURSIZE (PDMA_DSCTn_CTL[6:4]) will be fixed as one.

For the burst transfer type, the PDMA controller transfers TXCNT (PDMA_DSCTn_CTL[31:16]) of data and need only one request signal. After transferred BURSIZE (PDMA_DSCTn_CTL[6:4]) of data, TXCNT (PDMA_DSCTn_CTL[31:16]) will decrease BURSIZE number. Transfer will be done after the transfer count TXCNT (PDMA_DSCTn_CTL[31:16]) decreases to 0. Note that burst transfer type can only be used for PDMA controller to do burst transfer between memory and memory. User must use single request type for memory-to-peripheral and peripheral-to-memory transfers.

Figure 6.6-6 shows an example about single and burst transfer type in basic mode. In this example, channel 1 uses single transfer type and TXCNT (PDMA_DSCTn_CTL[31:16]) = 127. Channel 0 uses burst transfer type, BURSIZE (PDMA_DSCTn_CTL[6:4]) = 128 and TXCNT (PDMA_DSCTn_CTL[31:16]) = 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; the PDMA controller will load the channel 1 descriptor table first and executing. But channel 1 is single transfer type, and thus the PDMA controller will only transfer one transfer data.
3. Then, the PDMA controller 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, the PDMA controller 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, the PDMA controller will turn to channel 1 and transfer next one data.
5. After channel 1 transfers data, the PDMA controller 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. The PDMA controller 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 6.6-6 Example of Single Transfer Type and Burst Transfer Type in Basic Mode

6.6.5.4 Channel Time-out

Only PDMA channel 0 and channel 1 support time-out function. When the transfer channel is enabled and selected to the peripheral, corresponding channel time-out TOUTENn (PDMA_TOUTEN [n], n=0,1) 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 TOUTPSCn (PDMA_TOUTPSC [2+4n:4n], n=0,1). If time-out counter counts up from 0 to corresponding channel's TOCn (PDMA_TOC0_1 [16(n+1)-1):16n], n=0,1), the PDMA controller will generate interrupt signal when corresponding TOUTIENn (PDMA_TOUTIEN [n], n=0,1) 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 (PDMA_TOC0_1 [16(n+1)-1:16n], n=0,1), 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 6.6-7 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.

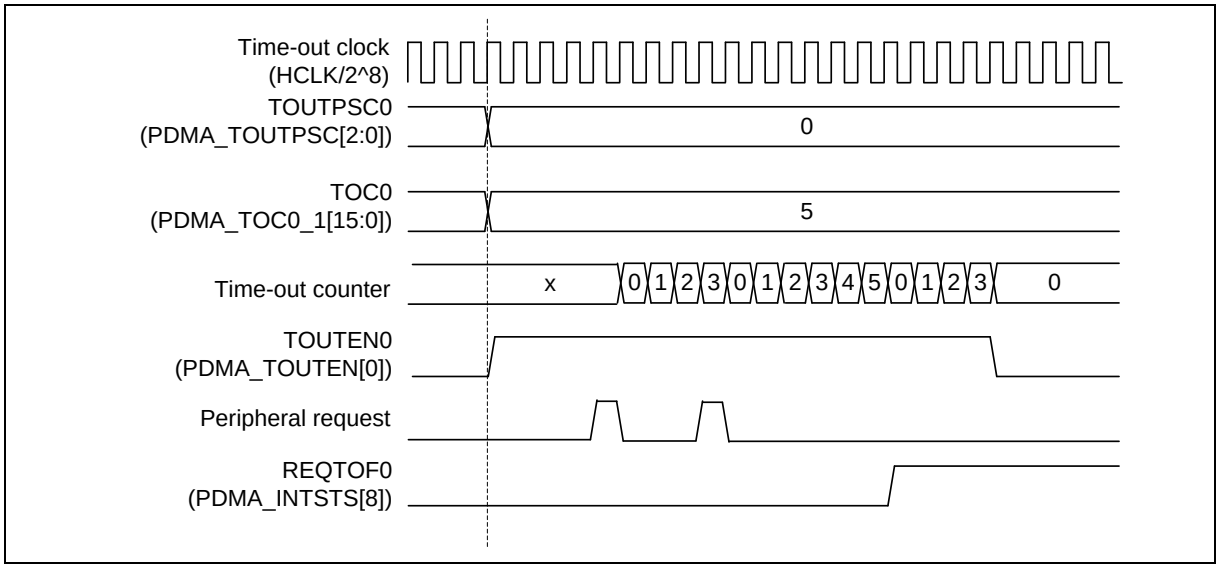


Figure 6.6-7 Example of PDMA Channel 0 Time-out Counter Operation

6.6.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMA_BA = 0x4000_8000**				
PDMA_DSCTn_CTL **n = 0,1..4**	PDMA_BA+0x10*n	R/W	Descriptor Table Control Register of PDMA Channel n	0xXXXX_XXXX
PDMA_DSCTn_SA **n = 0,1..4**	PDMA_BA+0x0004+0x10*n	R/W	Source Address Register of PDMA Channel n	0xXXXX_XXXX
PDMA_DSCTn_DA **n = 0,1..4**	PDMA_BA+0x0008+0x10*n	R/W	Destination Address Register of PDMA Channel n	0xXXXX_XXXX
PDMA_DSCTn_NEXT **n = 0,1..4**	PDMA_BA+0x000c+0x10*n	R/W	Next Scatter-gather Descriptor Table Offset Address of PDMA Channel n	0xXXXX_XXXX
PDMA_CURSCATn **n = 0,1..4**	PDMA_BA+0x0100+0x004*n	R	Current Scatter-gather Descriptor Table Address of PDMA Channel n	0xXXXX_XXXX
PDMA_CHCTL	PDMA_BA + 0x400	R/W	PDMA Channel Control Register	0x0000_0000
PDMA_PAUSE	PDMA_BA + 0x404	W	PDMA Transfer Pause Control Register	0x0000_0000
PDMA_SWREQ	PDMA_BA + 0x408	W	PDMA Software Request Register	0x0000_0000
PDMA_TRGSTS	PDMA_BA + 0x40C	R	PDMA Channel Request Status Register	0x0000_0000
PDMA_PRISET	PDMA_BA + 0x410	R/W	PDMA Fixed Priority Setting Register	0x0000_0000
PDMA_PRICLR	PDMA_BA + 0x414	W	PDMA Fixed Priority Clear Register	0x0000_0000
PDMA_INTEN	PDMA_BA + 0x418	R/W	PDMA Interrupt Enable Register	0x0000_0000
PDMA_INTSTS	PDMA_BA + 0x41C	R/W	PDMA Interrupt Status Register	0x0000_0000
PDMA_ABTSTS	PDMA_BA + 0x420	R/W	PDMA Channel Read/Write Target Abort Flag Register	0x0000_0000
PDMA_TDSTS	PDMA_BA + 0x424	R/W	PDMA Channel Transfer Done Flag Register	0x0000_0000
PDMA_ALIGN	PDMA_BA + 0x428	R/W	PDMA Transfer Alignment Status Register	0x0000_0000
PDMA_TACTSTS	PDMA_BA + 0x42C	R	PDMA Transfer Active Flag Register	0x0000_0000
PDMA_TOUTPSC	PDMA_BA + 0x430	R/W	PDMA Time-out Prescaler Register	0x0000_0000
PDMA_TOUTEN	PDMA_BA + 0x434	R/W	PDMA Time-out Enable Register	0x0000_0000
PDMA_TOUTIEN	PDMA_BA + 0x438	R/W	PDMA Time-out Interrupt Enable Register	0x0000_0000
PDMA_SCATBA	PDMA_BA + 0x43C	R/W	PDMA Scatter-gather Descriptor Table Base Address Register	0x2000_0000
PDMA_TOC0_1	PDMA_BA + 0x440	R/W	PDMA Time-out Counter Ch1 and Ch0 Register	0xFFFF_FFFF
PDMA_CHRST	PDMA_BA + 0x460	R/W	PDMA Channel Reset Register	0x0000_0000
PDMA_REQSEL0_3	PDMA_BA + 0x480	R/W	PDMA Request Source Select Register 0	0x0000_0000
PDMA_REQSEL4	PDMA_BA + 0x484	R/W	PDMA Request Source Select Register 1	0x0000_0000

6.6.7 Register Description

Descriptor Table Control Register (PDMA_DSCTn_CTL)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCTn_CTL	PDMA_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 16384, 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 decrease 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:** The PDMA transfer source address (PDMA_DSCT_SA) and PDMA transfer destination address (PDMA_DSCT_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 enabled it will not generates TDIFn(PDMA_TDSTS[4:0]) when PDMA controller finishes transfer task.

Bits	Description	
		0 = Table interrupt Enabled. 1 = Table interrupt Disabled. **Note:** This function is only for scatter-gather mode.
[6:4]	**BURSIZE**	**Burst 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, the 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_DSCT_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 the PDMA_INTSTS[1] to make sure the curren task is complete.

Start Source Address Register (PDMA_DSCTn_SA)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCTn_SA	PDMA_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 (PDMA_DSCTn_DA)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCTn_DA	PDMA_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 (PDMA_DSCTn_NEXT)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCTn_NEXT	PDMA_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 (PDMA_CURSCATn)

Register	Offset	R/W	Description	Reset Value
PDMA_CURSCATn	PDMA_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 (PDMA_CHCTL)

Register	Offset	R/W	Description	Reset Value
PDMA_CHCTL	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**CHEN4**	**CHEN3**	**CHEN2**	**CHEN1**	**CHEN0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**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 (PDMA_PAUSE)

Register	Offset	R/W	Description	Reset Value
PDMA_PAUSE	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**PAUSE4**	**PAUSE3**	**PAUSE2**	**PAUSE1**	**PAUSE0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**PAUSEn**	**PDMA Channel n Transfer Pause Control (Write Only)** User can set PAUSEn bit field to pause the PDMA transfer. When user sets PAUSEn bit, the PDMA controller will pause the on-going transfer, then clear the channel enable bit CHEN(PDMA_CHCTL [n], n=0,1..4) and clear request active flag(PDMA_TRGSTS[n:0], n=0,1..4). If the paused channel is re-enabled again, the remaining transfers will be processed. 0 = No effect. 1 = Pause PDMA channel n transfer.

PDMA Software Request Register (PDMA_SWREQ)

Register	Offset	R/W	Description	Reset Value
PDMA_SWREQ	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**SWREQ4**	**SWREQ3**	**SWREQ2**	**SWREQ1**	**SWREQ0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**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 PDMA_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 (PDMA_TRGSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_TRGSTS	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**REQSTS4**	**REQSTS3**	**REQSTS2**	**REQSTS1**	**REQSTS0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**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 (PDMA_PRISET)

Register	Offset	R/W	Description	Reset Value
PDMA_PRISET	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**FPRISET4**	**FPRISET3**	**FPRISET2**	**FPRISET1**	**FPRISET0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**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 (PDMA_PRICLR)

Register	Offset	R/W	Description	Reset Value
PDMA_PRICLR	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**FPRICLR4**	**FPRICLR3**	**FPRICLR2**	**FPRICLR1**	**FPRICLR0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**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 (PDMA_INTEN)

Register	Offset	R/W	Description	Reset Value
PDMA_INTEN	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**INTEN4**	**INTEN3**	**INTEN2**	**INTEN1**	**INTEN0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**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 (PDMA_INTSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_INTSTS	PDMA_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 PDMA_TOC1, user can write 1 to clear these bits. 0 = No request time-out. 1 = Peripheral request time-out. **Note:** Please disable time-out function before clearing 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 PDMA_TOC0, user can write 1 to clear these bits. 0 = No request time-out. 1 = Peripheral request time-out. **Note:** Please disable time-out function before clearing 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. 1 = AHB bus ERROR response received.

PDMA Channel Read/Write Target Abort Flag Register (PDMA_ABTSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_ABTSTS	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**ABTIF4**	**ABTIF3**	**ABTIF2**	**ABTIF1**	**ABTIF0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**ABTIFn**	**PDMA Read/Write Target Abort Interrupt Status Flag** This bit indicates which PDMA controller has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel n transfer. 1 = AHB bus ERROR response received when channel n transfer.

PDMA Channel Transfer Done Flag Register (PDMA_TDSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_TDSTS	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**TDIF4**	**TDIF3**	**TDIF2**	**TDIF1**	**TDIF0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**TDIFn**	**Transfer Done Flag** This bit indicates whether PDMA controller channel transfer has been finished or not, user can write 1 to clear these bits. 0 = PDMA channel transfer has not finished. 1 = PDMA channel has finished transmission.

PDMA Transfer Alignment Status Register (PDMA_ALIGN)

Register	Offset	R/W	Description	Reset Value
PDMA_ALIGN	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**ALIGN4**	**ALIGN3**	**ALIGN2**	**ALIGN1**	**ALIGN0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**ALIGNn**	**Transfer Alignment Flag** This bit indicates whether source and destination address both follow transfer width setting, user can write 1 to clear these bits. 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.

PDMA Transfer Active Flag Register (PDMA_TACTSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_TACTSTS	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**TXACTF4**	**TXACTF3**	**TXACTF2**	**TXACTF1**	**TXACTF0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**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.

PDMA Time-out Prescaler Register (PDMA_TOUTPSC)

Register	Offset	R/W	Description	Reset Value
PDMA_TOUTPSC	PDMA_BA + 0x430	R/W	PDMA Time-out Prescaler Register	0x0000_0000

31	30	29	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 (PDMA_TOUTEN)

Register	Offset	R/W	Description	Reset Value
PDMA_TOUTEN	PDMA_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 (PDMA_TOUTIEN)

Register	Offset	R/W	Description	Reset Value
PDMA_TOUTIEN	PDMA_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 (PDMA_SCATBA)

Register	Offset	R/W	Description	Reset Value
PDMA_SCATBA	PDMA_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_DSCT_NEXT. **Note:** Only useful in Scatter-Gather mode.
[15:0]	**Reserved**	Reserved.

PDMA Time-out Period Counter Register 0 (PDMA_TOC0_1)

Register	Offset	R/W	Description	Reset Value
PDMA_TOC0_1	PDMA_BA + 0x440	R/W	PDMA Time-out Counter Ch1 and Ch0 Register	0xFFFF_FFFF

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. For the example of time-out period, refer to 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 (PDMA_CHRST)

Register	Offset	R/W	Description	Reset Value
PDMA_CHRST	PDMA_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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**CH4RST**	**CH3RST**	**CH2RST**	**CH1RST**	**CH0RST**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:0]	**CHnRST**	**Channel n Reset** 0 = corresponding channel n is not reset. 1 = corresponding channel n is reset.

PDMA Request Source Select Register 0 (PDMA_REQSEL0_3)

Register	Offset	R/W	Description	Reset Value
PDMA_REQSEL0_3	PDMA_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	
[31:30]	**Reserved**	Reserved.
[29:24]	**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.
[23:22]	**Reserved**	Reserved.
[21:16]	**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.
[15:14]	**Reserved**	Reserved.
[13:8]	**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.
[7:6]	**Reserved**	Reserved.
[5:0]	**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 = Channel connects to DAC0_TX.** **4 = Channel connects to UART0_TX.** **5 = Channel connects to UART0_RX.** **6 = Channel connects to UART1_TX.** **7 = Channel connects to UART1_RX.**

Bits	Description	
		8 = Reserved. **9 = Reserved.** **10 = Channel connects to USCI0_TX.** **11 = Channel connects to USCI0_RX.** **12 = Channel connects to USCI1_TX.** **13 = Channel connects to USCI1_RX.** **14 = Reserved.** **15 = Reserved.** **16 = Reserved.** **17 = Reserved.** **18 = Reserved.** **19 = Reserved.** **20 = Channel connects to ADC_RX.** **21 = Channel connects to PWM0_P1_RX.** **22 = Channel connects to PWM0_P2_RX.** **23 = Channel connects to PWM0_P3_RX.** **24 = Reserved.** **25 = Reserved.** **26 = Reserved.** **27 = Reserved.** **28 = Reserved.** **29 = Reserved.** **30 = Reserved.** **31 = Reserved.** **32 = Channel connects to TMR0.** **33 = Channel connects to TMR1.** **34 = Channel connects to TMR2.** **35 = Channel connects to TMR3.** **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 (PDMA_REQSEL4)

Register	Offset	R/W	Description	Reset Value
PDMA_REQSEL4	PDMA_BA + 0x484	R/W	PDMA Request 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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**REQSRC4**					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**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.

6.7 Timer Controller (TMR)

6.7.1 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.

6.7.2 Features

6.7.2.1 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])
- Supports event counting function
- 24-bit capture value is readable through CAPDAT (TIMERx_CAP[23:0])
- Supports external capture pin event for interval measurement
- Supports external capture pin event to reset 24-bit up counter
- Supports internal capture triggered while internal ACMP output signal and LIRC transition
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated
- Support Timer0 ~ Timer3 time-out interrupt signal or capture interrupt signal to trigger PWM, ADC, PDMA function
- Supports Inter-Timer trigger mode

6.7.4 Block Diagram

The timer controller block diagram and clock control are shown as follows.

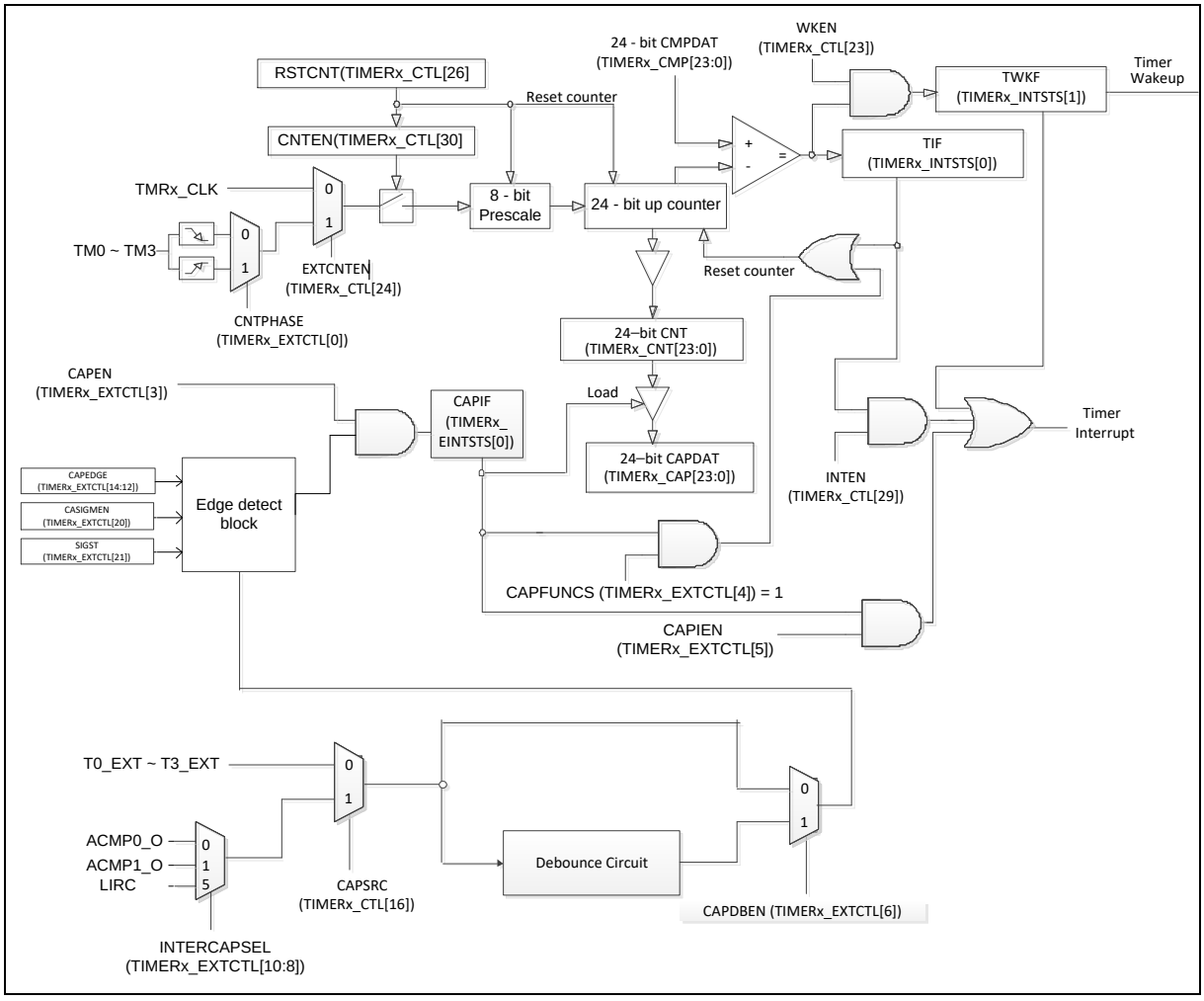


Figure 6.7-1 Timer Controller Block Diagram

6.7.5 Basic Configuration

- Clock Source Configuration
  - The clock source of Timer0 ~ Timer3 in timer mode can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]).
  - Select the source of Timer0 ~ Timer3 on 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.

The Timer0 ~ Timer3 clock control is shown in Figure 6.7-2.

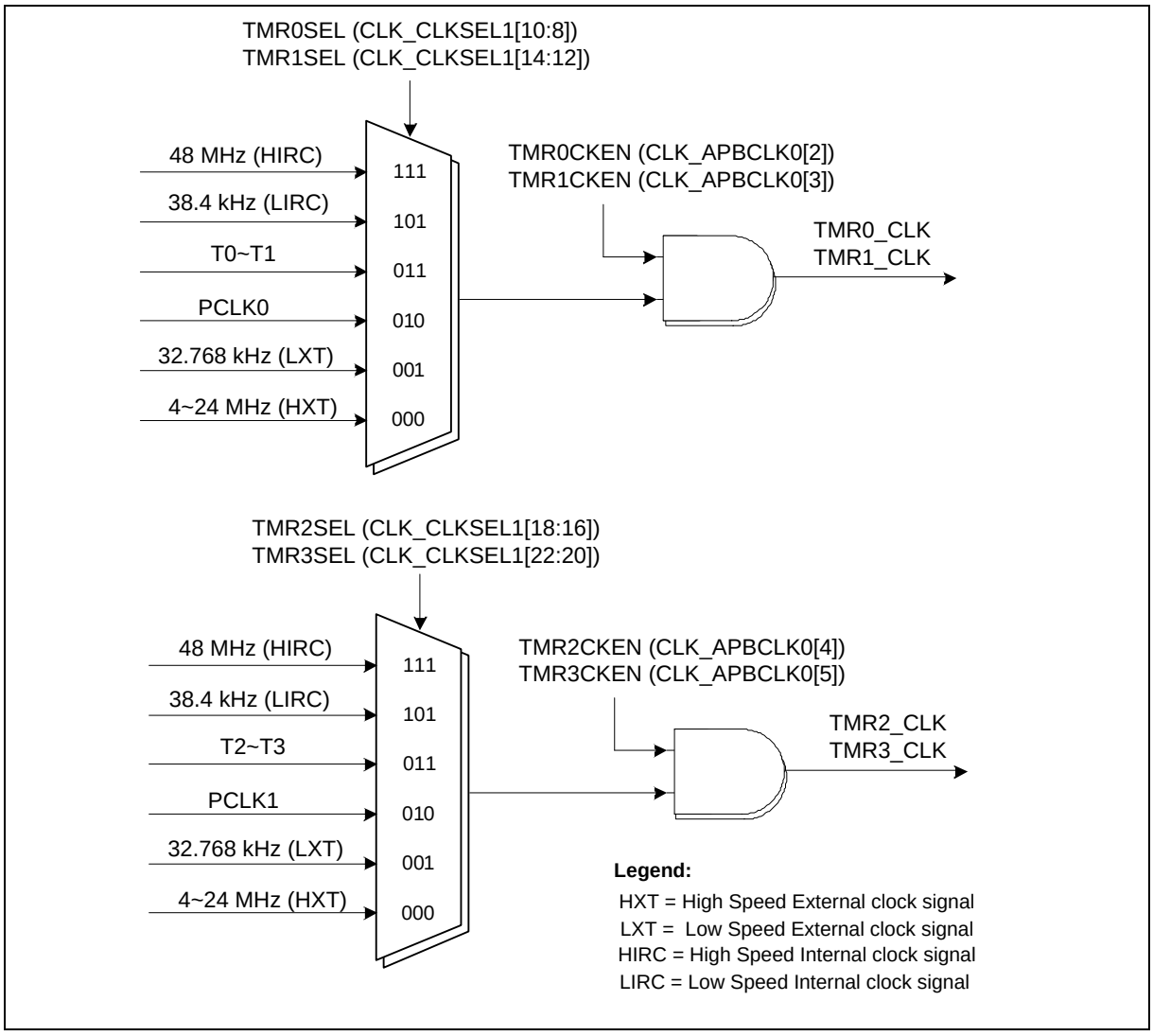


Figure 6.7-2 Clock Source of Timer Controller

- Timer0 ~ Timer3 MFP configurations
  - The MFP table for Timer0 ~ Timer3 is shown as Table 6.7-1.

Signal Name	Pin Name	MFP
TM0	PA.1, PA.5, PA.3, PB.7, PB.5, PC.0, PC.2, PC.4, PC.6	MFP24

	PD.0, PD.4	MFP5
TM1	PA.2, PA.0, PA.4, PB.4, PB.6, PC.1, PC.3, PC.5, PC.7	MFP24
	PD.1, PD.5	MFP5
TM2	PA.1, PA.5, PA.3, PB.7, PB.5, PC.0, PC.2, PC.4, PC.6	MFP25
	PD.2, PD.6	MFP5
TM3	PA.2, PA.0, PA.4, PB.4, PB.6, PC.1, PC.3, PC.5, PC.7	MFP25
	PD.3, PD.7	MFP5
TM0_EXT	PA.1, PA.5, PA.3, PB.7, PB.5, PC.0, PC.2, PC.4, PC.6	MFP26
TM1_EXT	PA.2, PA.0, PA.4, PB.4, PB.6, PC.1, PC.3, PC.5, PC.7	MFP26
TM2_EXT	PA.1, PA.5, PA.3, PB.7, PB.5, PC.0, PC.2, PC.4, PC.6	MFP27
TM3_EXT	PA.2, PA.0, PA.4, PB.4, PB.6, PC.1, PC.3, PC.5, PC.7	MFP27

Table 6.7-1 Timer0 ~ Timer3 MFP Table

6.7.6 Functional Description

6.7.6.1 Timer Interrupt Flag

The timer controller supports the following interrupt flags; one is TIF (TIMERx_INTSTS[0]) and it is set while timer counter value CNT (TIMERx_CNT[23:0]) matches the timer compared value CMPDAT (TIMERx_CMP[23:0]). The other situation that CAPIF (TIMERx_EINTSTS[0]) is set means when the transition on the TMx_EXT pin, LIRC, or ACMP is associated with CAPEDGE (TIMERx_EXTCTL[14:12]) setting. User can set CAPSRC (TIMERx_CTL[16]) and INTERCAPSEL (TIMERx_EXTCTL[10:8]) to select capture source. The TWKF (TIMERx_INTSTS[1]) bit indicates the interrupt wake-up flag status of timer. Set WKEN (TIMERx_CTL[23]) to 1 to use wake-up function.

6.7.6.2 Timer Counting Mode

The timer controller provides four timer counting modes: one-shot, periodic, toggle-output and continuous counting operation modes:

6.7.6.3 One–Shot Mode

If the timer controller is configured in 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 set, the timer interrupt signal is generated and sent to NVIC to inform CPU.

User can set ICEDEBUG (TIMERx_CTL[31]) to 1 to disable ICE debug mode acknowledgement that affects TIMER counting when CPU is halted.

6.7.6.4 Periodic Mode

If the timer controller is configured in 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 set, the timer interrupt signal is generated and sent to NVIC to inform CPU. In this mode, the timer controller operates counting and compares with CMPDAT value periodically until the CNTEN bit is cleared by user.

6.7.6.5 Toggle-Output Mode

If the timer controller is configured in 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. User can set TGLPINSEL (TIMERx_CTL[22]) 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.

6.7.6.6 Continuous Counting Mode

If the timer controller is configured in 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 set, the timer interrupt signal is generated and sent to NVIC to inform CPU. User can change different CMPDAT value immediately without disabling timer counting and restarting timer counting in this mode.

For example, the CMPDAT value is set as 80, first. The TIF will be set to 1 when the CNT value is equal to 80, the timer counter is kept counting and the CNT value will not go 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 be set to 1 again when the CNT value reaches 200. At last, user programs CMPDAT as 500 and clears TIF, and the TIF will be set to 1 again when the CNT value reaches 500.

In this mode, the timer counting is continuous. So, this operation mode is called as continuous counting mode.

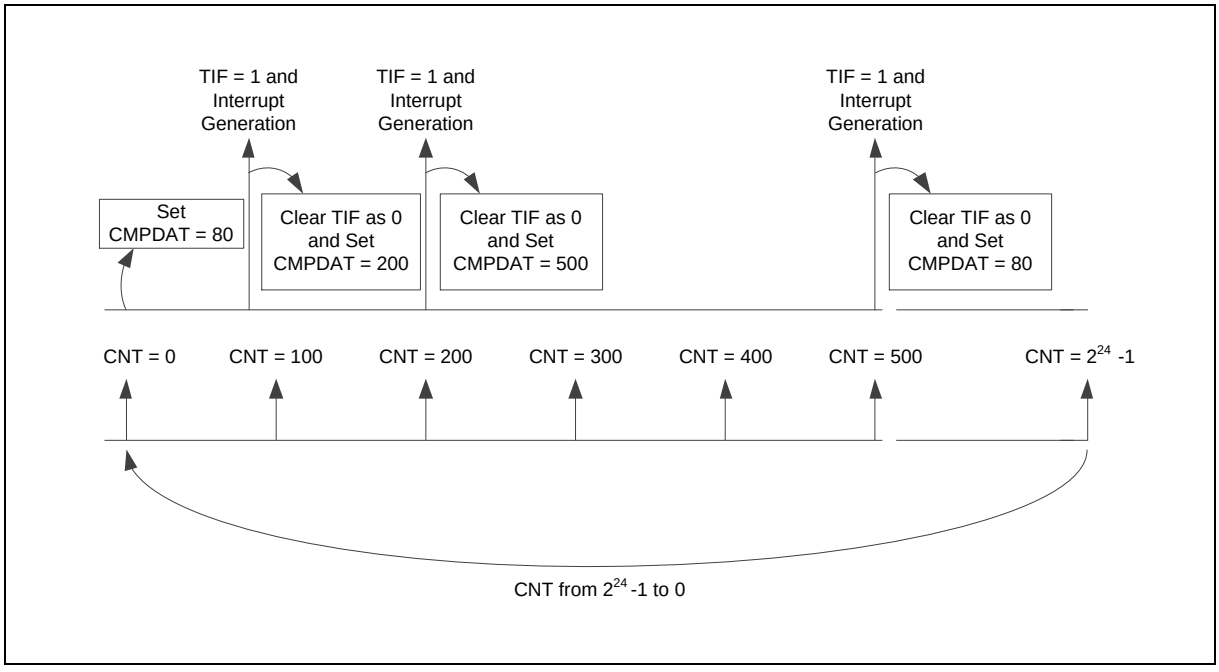


Figure 6.7-3 Continuous Counting Mode

6.7.6.7 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 PCLK.

If ECNTSSEL (TIMERx_EXTCTL[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 PCLK if TMx pin de-bounce disabled or less than 1/8 PCLK 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 pin.

6.7.6.8 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, LIRC, or ACMP. 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 PCLK.

If CAPSRC (TIMERx_CTL[16]) is 0, the capture event is triggered by TMx_EXT pin transition. User can 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 PCLK if TMx_EXT pin de-bounce disabled or less than 1/8 PCLK 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, if user did 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 fuction. 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.

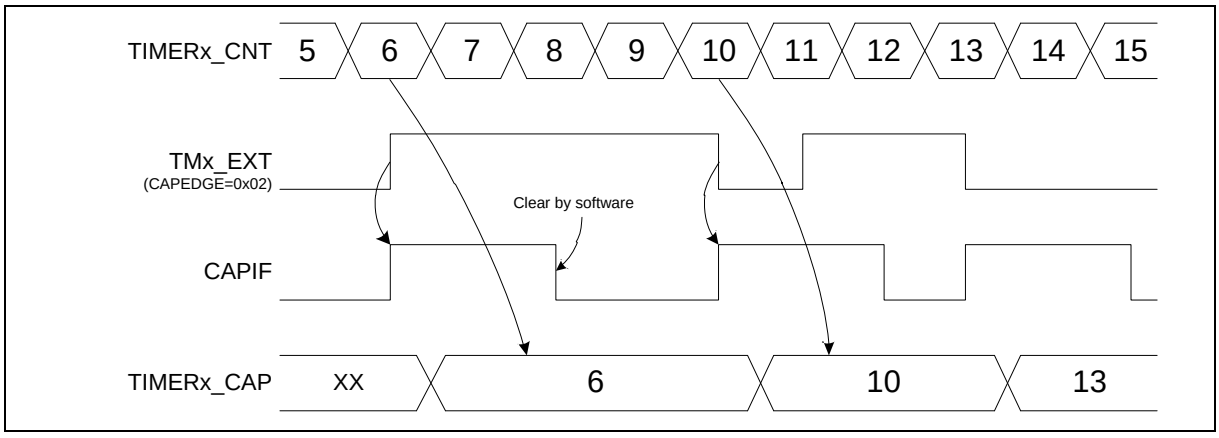


Figure 6.7-4 External Capture Mode

If CAPSRC (TIMERx_CTL[16]) 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 is 000, ACMP1 if INTERCAPSEL is 001, or LIRC if INTERCAPSEL is 101.

6.7.6.9 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 as 1. User must set CAPSRC and INTERCAPSEL to select TMx_EXT transition, internal ACMPx output signal, or LIRC

to trigger reset counter value.

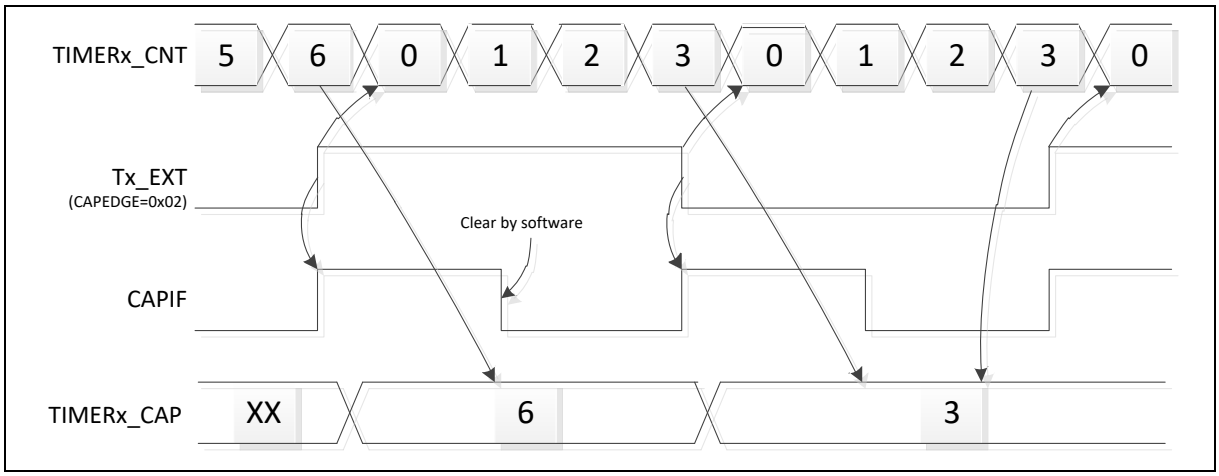


Figure 6.7-5 Reset Counter Mode

6.7.6.10 Timer Trigger Function

The timer controller provides timer time-out interrupt or capture interrupt to trigger PWM, DAC, ADC and PDMA. If TRGSSEL (TIMERx_CTL[18]) is 0, time-out interrupt signal is used to trigger PWM, DAC, ADC and PDMA. If TRGSSEL (TIMERx_CTL[18]) is 1, capture interrupt signal is used to trigger PWM, DAC, ADC and PDMA.

When the TRGPWM (TIMERx_CTL[19]) is set, if the timer interrupt signal is generated, the timer controller will generate a trigger pulse as PWM external clock source.

When the TRGDAC (TIMERx_CTL[20]) is set, if the timer interrupt signal is generated, the timer controller will trigger DAC to start converter.

When the TRGADC (TIMERx_CTL[21]) is set, if the timer interrupt signal is generated, the timer controller will trigger ADC to start converter.

When the TRGPDMA (TIMERx_CTL[8]) is set, if the timer interrupt signal is generated, the timer controller will trigger PDMA.

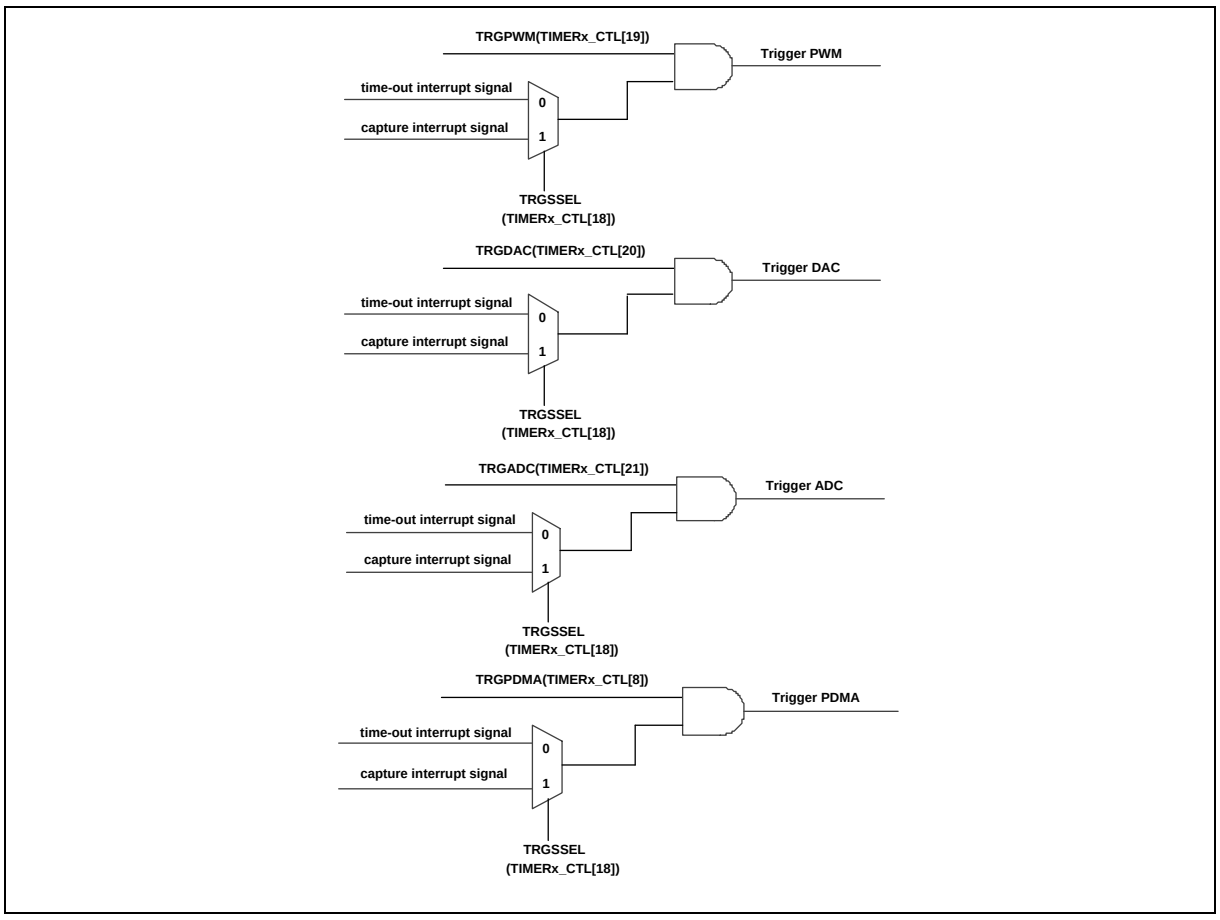


Figure 6.7-6 Internal Timer Trigger

6.7.6.11 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.

When 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[10]) 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 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[10]) 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]).

The inter-timer trigger mode can be used to measure the period of external event (TMx) more precisely. Figure 6.7-7 shows the sample flow of Inter-Timer Trigger Capture Mode for Timer0 as event counting mode and Timer1 as trigger-counting capture mode.

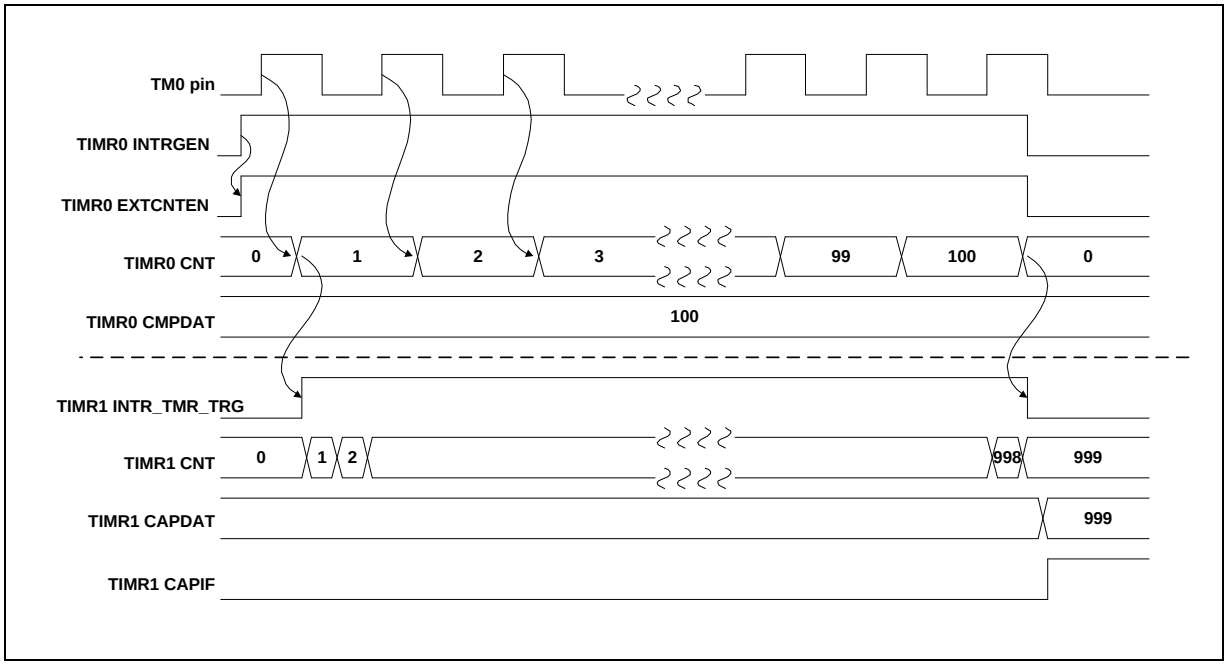


Figure 6.7-7 Inter-Timer Trigger Capture Timing

User must clear Timer1/3 CAPIF if user wants to use the second inter-timer trigger function.

6.7.6.12 Capture Single Measure Mode

When user sets CAPEN (TIMERx_EXTCTL[3]) = 1 and CASIGMEN(TIMERx_EXTCTL[20]) = 1, Timer will enter Single Meaure Mode. User can use timer to measure full period or half period. When user writes one to SIGST (TIMERx_EXTCTL[21]), timer will start to measure TMx_EXT. When timer detects edge finish, SIGST will be auto cleared by hardware and the measured value will be uploaded to Timer Capture Data Register. User can write one to SIGST to trigger timer measure again.

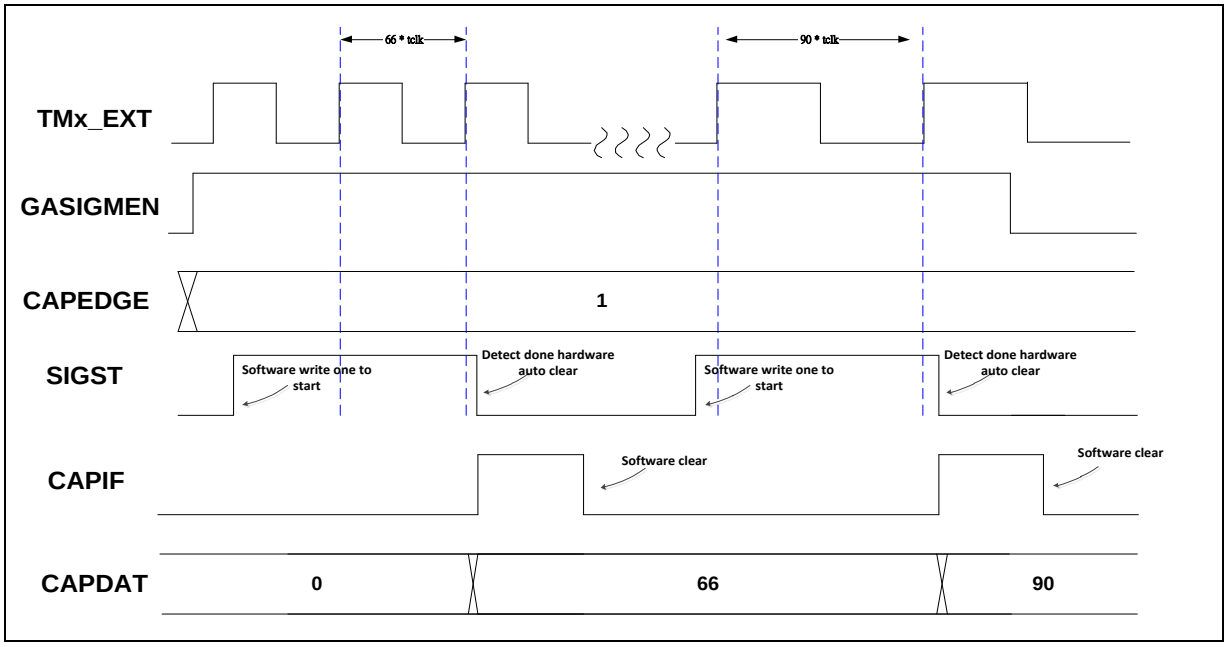


Figure 6.7-8 Capture Single Measure Mode

6.7.7 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	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
TIMER1_CTL	TMR01_BA+0x20	R/W	Timer1 Control Register	0x0000_0005
TIMER1_CMP	TMR01_BA+0x24	R/W	Timer1 Comparator Register	0x0000_0000
TIMER1_INTSTS	TMR01_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER1_CNT	TMR01_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TIMER1_CAP	TMR01_BA+0x30	R	Timer1 Capture Data Register	0x0000_0000
TIMER1_EXTCTL	TMR01_BA+0x34	R/W	Timer1 External Control Register	0x0000_0000
TIMER1_EINTSTS	TMR01_BA+0x38	R/W	Timer1 External Interrupt Status 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	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
TIMER3_CTL	TMR23_BA+0x20	R/W	Timer3 Control Register	0x0000_0005
TIMER3_CMP	TMR23_BA+0x24	R/W	Timer3 Comparator Register	0x0000_0000
TIMER3_INTSTS	TMR23_BA+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000
TIMER3_CNT	TMR23_BA+0x2C	R	Timer3 Data Register	0x0000_0000
TIMER3_CAP	TMR23_BA+0x30	R	Timer3 Capture Data Register	0x0000_0000

TIMER3_EXTCTL	TMR23_BA+0x34	R/W	Timer3 External Control Register	0x0000_0000
TIMER3_EINTSTS	TMR23_BA+0x38	R/W	Timer3 External Interrupt Status Register	0x0000_0000

6.7.8 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+0x20	R/W	Timer1 Control Register	0x0000_0005
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2 Control Register	0x0000_0005
TIMER3_CTL	TMR23_BA+0x20	R/W	Timer3 Control Register	0x0000_0005

31	30	29	28	27	26	25	24
ICEDEBUG	CNTEN	INTEN	OPMODE		RSTCNT	ACTSTS	EXTCNTEN
23	22	21	20	19	18	17	16
WKEN	TGLPINSEL	TRGADC	TRGDAC	TRGPWM	TRGSSEL	Reserved	CAPSRC
15	14	13	12	11	10	9	8
Reserved					INTRGEN	Reserved	TRGPDMA
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 setting 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:** Setting 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]	**RSTCNT**	**Timer Counter Reset Bit** Setting this bit will reset the 24-bit up counter value CNT (TIMERx_CNT[23:0]) and also force CNTEN (TIMERx_CTL[30]) to 0 if ACTSTS (TIMERx_CTL[25]) is 1. 0 = No effect. 1 = Reset internal 8-bit prescale counter, 24-bit up counter value and CNTEN bit. **Note:** This bit will be auto cleared.
[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 be 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 1:** When timer is used as an event counter, this bit should be set to 1 and select PCLKx (x=0~1) as timer clock source. **Note 2:** When TMR0/TMR2 INTRGEN is set to 1, this bit is forced to 1. When INTRGEN is 1 and TMR1/TMR3 CAPIF (TIMERx_EINTSTS[0]) is 1, this bit is forced to 0.
[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]	**TGLPINSEL**	**Toggle-output Pin Select** 0 = Toggle mode output to Tx (Timer Event Counter Pin). 1 = Toggle mode output to Tx_EXT (Timer External Capture Pin).
[21]	**TRGADC**	**Trigger ADC Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger ADC. 0 = Timer interrupt trigger ADC Disabled. 1 = Timer interrupt trigger ADC Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger ADC. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger ADC.
[20]	**TRGDAC**	**Trigger DAC Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger DAC. 0 = Timer interrupt trigger DAC Disabled. 1 = Timer interrupt trigger DAC Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger DAC. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger DAC.
[19]	**TRGPWM**	**Trigger PWM Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger PWM. 0 = Timer interrupt trigger PWM Disabled. 1 = Timer interrupt trigger PWM Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger PWM.

		If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger PWM.
[18]	**TRGSSEL**	**Trigger Source Select Bit** This bit is used to select trigger source is from Timer time-out interrupt signal or capture interrupt signal. 0 = Timer time-out interrupt signal is used to trigger PWM, DAC, ADC and PDMA. 1 = Capture interrupt signal is used to trigger PWM, DAC, ADC and PDMA.
[17]	**Reserved**	Reserved.
[16]	**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 or LIRC. User can set INTERCAPSEL (TIMERx_EXTCTL[10:8]) to decide which internal ACMP output signal or LIRC is timer capture source.
[15:11]	**Reserved**	Reserved.
[10]	**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 mode Disabled. 1 = Inter-Timer Trigger mode Enabled. **Note:** For Timer1/3, this bit is ignored and the read back value is always 0.
[9]	**Reserved**	Reserved.
[8]	**TRGPDMA**	**Trigger PDMA Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger PDMA. 0 = Timer interrupt trigger PDMA Disabled. 1 = Timer interrupt trigger PDMA Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger PDMA. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger PDMA.
[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:** Updating 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+0x24	R/W	Timer1 Comparator Register	0x0000_0000
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2 Comparator Register	0x0000_0000
TIMER3_CMP	TMR23_BA+0x24	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 the 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+0x28	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+0x28	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 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	Timer0 Data Register	0x0000_0000
TIMER1_CNT	TMR01_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TIMER2_CNT	TMR23_BA+0x0C	R	Timer2 Data Register	0x0000_0000
TIMER3_CNT	TMR23_BA+0x2C	R	Timer3 Data Register	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**	**Timer Data Register** 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.

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+0x30	R	Timer1 Capture Data Register	0x0000_0000
TIMER2_CAP	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TIMER3_CAP	TMR23_BA+0x30	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, 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+0x34	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+0x34	R/W	Timer3 External Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		SIGST	CASIGMEN	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	
[31:22]	**Reserved**	Reserved.
[21]	**SIGST**	**Single Measure Start Bit** User can write 1'b1 to this bit to let timer start measure TMx_EXT pin. When capture measure event finishes this bit will auto clear by hardware.
[20]	**CASIGMEN**	**Capture Single Measure Mode Enable Bit** Enable Single Pulse Mode function, user can write one to SIGST that can start measure full period or half period on TMx_EXT(x=0~3),. 0 = Single Measure Mode Disabled. 1 = Single Measure Mode Enabled. **Note:** these bits only available when CAPEN (TIMERx_EXTCTL[3]) is 1.
[19:17]	**Reserved**	Reserved.
[16]	**ECNTSSEL**	**Event Counter Source Selection to Trigger Event Counter Function** 0 = Event Counter input source is from TMx (x= 0~3) pin. 1 = 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]) will be set to 0. Disable Single Pulse Mode : 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. Enable Single Pulse Mode: 000 = Measure **falling edge → falling edge** transfer on TMx_EXT (x= 0~3) pin. 001 = Measure **rising edge → rising edge** transfer on TMx_EXT (x= 0~3) pin. 110 = Measure **falling edge → rising edge** transfer on TMx_EXT (x= 0~3) pin. 111 = Measure **rising edge → falling edge** transfer on TMx_EXT (x= 0~3) pin. 010, 011, 100, 101 = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**INTERCAPSEL**	**Internal Capture Source Selection to Trigger Capture Function** 000 = Capture Function source is from internal ACMP0 output signal. 001 = Capture Function source is from internal ACMP1 output signal. 101 = Capture Function source is from LIRC. Others = Reserved. **Note:** these bits only available when CAPSRC (TIMERx_CTL[16]) 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, LIRC, or ACMP detection Interrupt Disabled. 1 = TMx_EXT (x= 0~3) pin, LIRC, or ACMP 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 Tx_EXT (x= 0~3) pin, or ACMP 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: TMR1/TMR3 CAPEN will be forced to 1 when TMR0/TMR2 INTRGEN is enabled.
[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+0x38	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+0x38	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							**CAPIF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CAPIF**	**Timer External Capture Interrupt Flag** This bit indicates the timer external capture interrupt flag status. 0 = TMx_EXT (x= 0~3) pin interrupt did not occur. 1 = TMx_EXT (x= 0~3) pin interrupt occurred. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** When CAPEN (TIMERx_EXTCTL[3]) bit is set and a transition on Tx_EXT (x= 0~3) pin matched the CAPEDGE (TIMERx_EXTCTL[14:12]) 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.

6.8 Watchdog Timer (WDT)

6.8.1 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.

6.8.2 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 416us ~ 27.3 s if WDT_CLK = 38.4 kHz (LIRC).
- 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.

6.8.3 Block Diagram

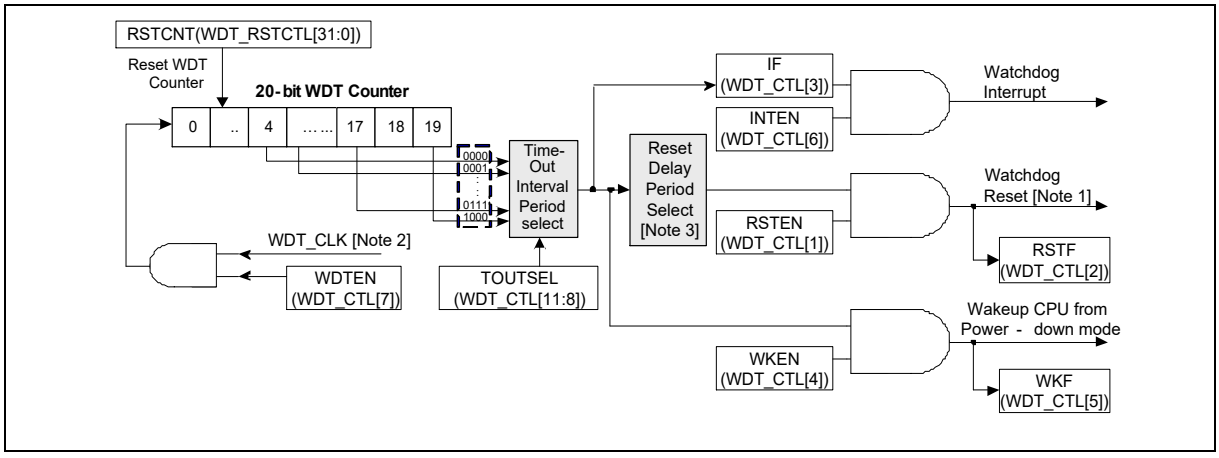


Figure 6.8-1 Watchdog Timer Block Diagram

Note1: WDT resets CPU and lasts 63 WDT_CLK.

Note2: Chip can be woken up by WDT time-out interrupt signal generated only if WDT clock source is selected to LIRC or LXT.

Note3: The WDT reset delay period can be selected as 3/18/130/1026 WDT_CLK.

6.8.4 Basic Configuration

- Clock Source Configuration
  - Select the source of WDT peripheral clock on WDTSEL (CLK_CLKSEL1[1:0])
  - Enable WDT peripheral clock in WDTCKEN (CLK_APBCLK0[0]).
  - Force 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])

- WDT clock source can be changed only if CWDTEN[2:0] is 111.

The WDT clock control is shown in Figure 6.8-2.

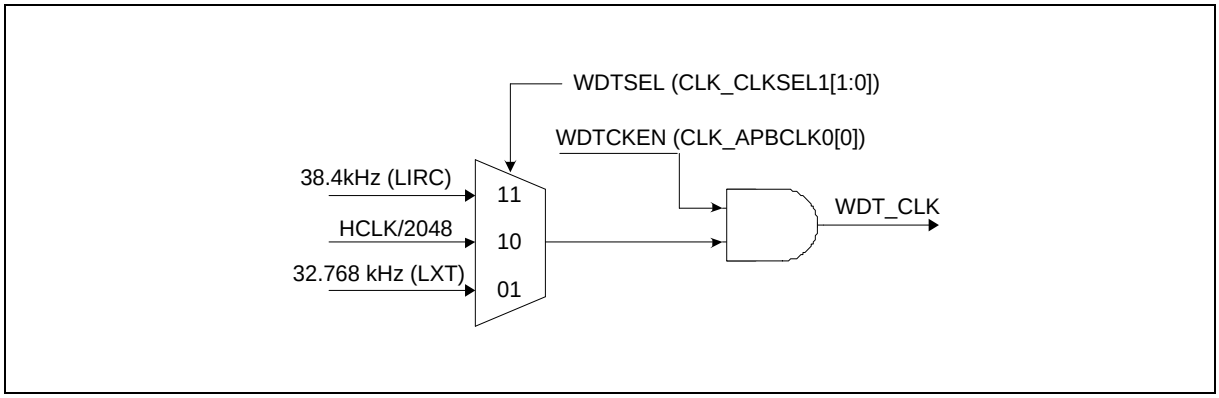


Figure 6.8-2 Watchdog Timer Clock Control

6.8.5 Functional Description

The WDT includes an 20-bit free running up counter with programmable time-out intervals. Table 6.8-1 shows the WDT time-out interval period selection and Figure 6.8-3 shows the WDT time-out interval and reset period timing.

6.8.5.1 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 eight 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.

6.8.5.2 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 set WDT_RSTCNT 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 reset state immediately. Refer to Figure 6.8-3. T_{RST} reset period will keep last 63 WDT clocks then chip restart 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}
000	$2^4 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
001	$2^6 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
010	$2^8 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
011	$2^{10} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
100	$2^{12} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
101	$2^{14} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
110	$2^{16} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
111	$2^{18} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
1000	$2^{20} * T_{WDT}$	(3/18/130/1026) * T_{WDT}

Table 6.8-1 Watchdog Timer Time-out Interval Period Selection

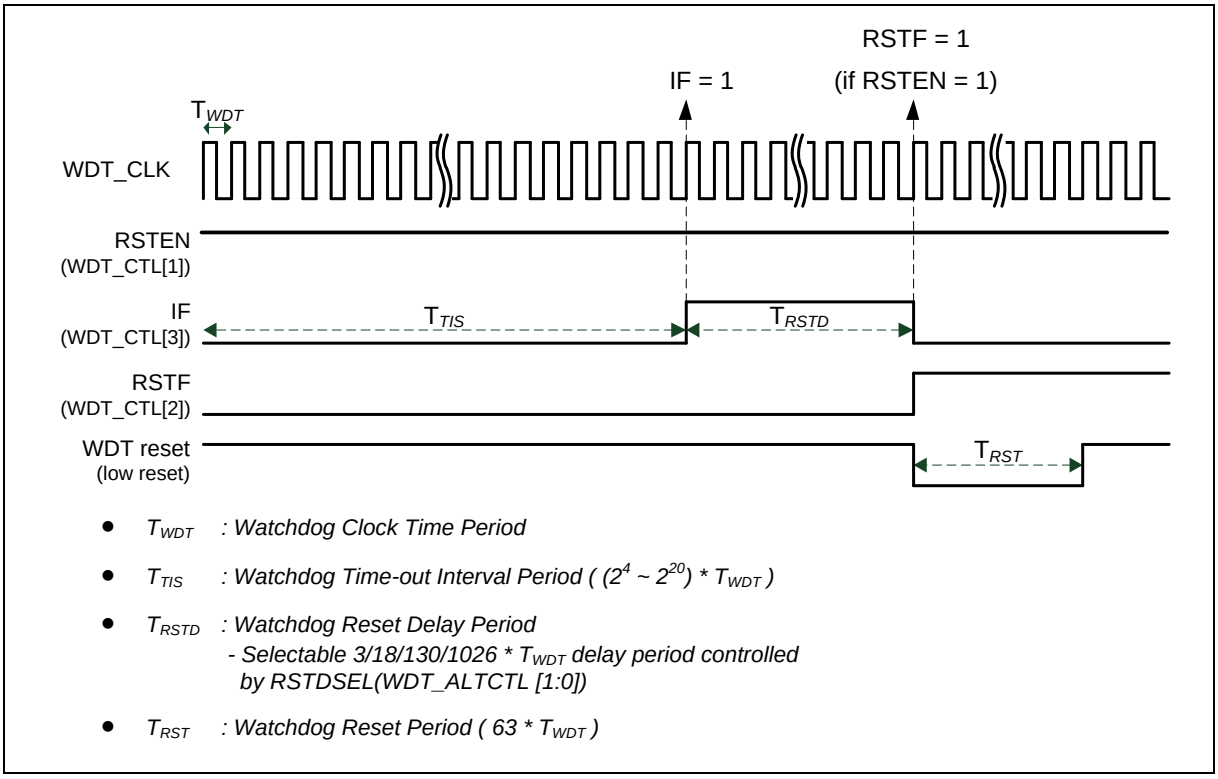


Figure 6.8-3 Watchdog Timer Time-out Interval and Reset Period Timing

6.8.5.3 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 clock 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.

6.8.5.4 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, 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.

6.8.6 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

6.8.7 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**	**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 = Set WDTEN bit is completed. 1 = Set WDTEN bit is synchronizing and not become active yet. **Note:** Performing enable or disable WDTEN bit needs 2 * WDT_CLK period to become active.
[29:12]	**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 (Write Protect)** 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 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** 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 38.4 kHz 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 writing 0x00005aa5 to RSTCNT (WDT_RSTCNT[31:0]) 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**.**

6.9 Window Watchdog Timer (WWDT)

6.9.1 Overview

The Window Watchdog Timer (WWDT) is used to perform a system reset within a specified window period to prevent software run to uncontrollable status by any unpredictable condition.

6.9.2 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

6.9.3 Block Diagram

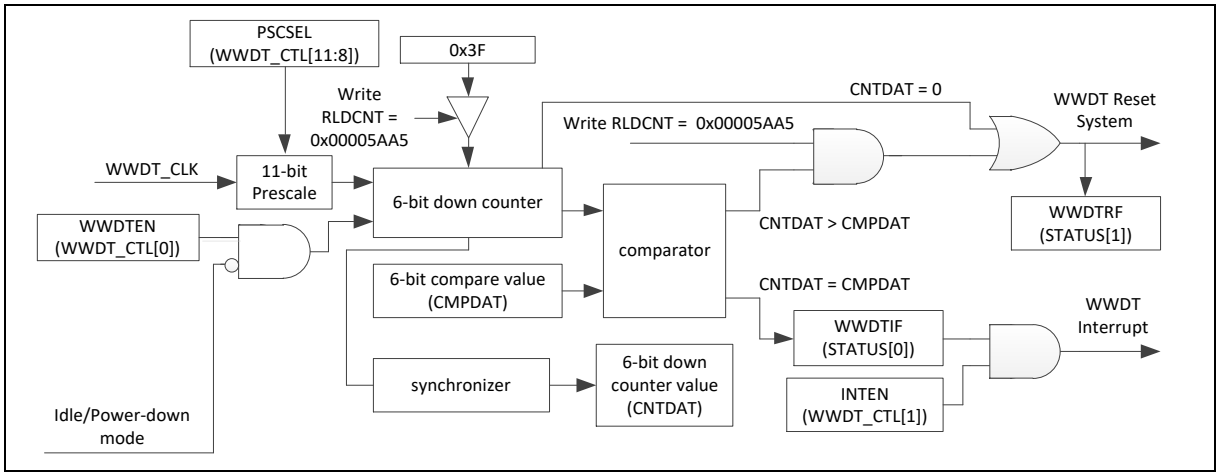


Figure 6.9-1 WWDT Block Diagram

6.9.4 Basic Configuration

- Clock Source Configuration
  - Select the source of WWDT peripheral clock on WWDTSEL (CLK_CLKSEL1[31:30])
  - Enable WWDT peripheral clock in WDTCKEN (CLK_APBCLK0[0]).

The WWDT clock control is shown in Figure 6.9-2.

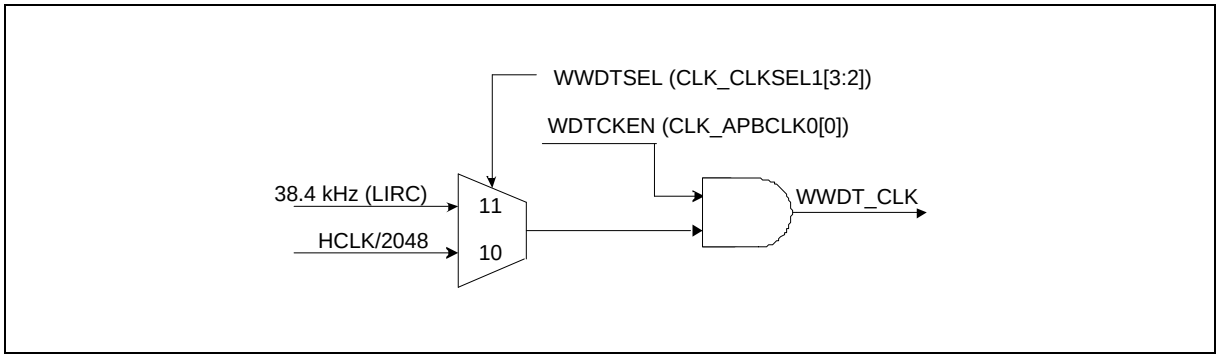


Figure 6.9-2 WWDT Clock Control

6.9.5 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 38.4 kHz 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 6.9-1.

PSCSEL	Prescaler Value	Max. Time-Out Period	Max. Time-Out Interval (WWDT_CLK=38.4 kHz)
0000	1	$1 * 64 * T_{WWDT}$	1.666667 ms
0001	2	$2 * 64 * T_{WWDT}$	3.33333 ms
0010	4	$4 * 64 * T_{WWDT}$	6.6667 ms
0011	8	$8 * 64 * T_{WWDT}$	13.333 ms
0100	16	$16 * 64 * T_{WWDT}$	26.667 ms
0101	32	$32 * 64 * T_{WWDT}$	53.333 ms
0110	64	$64 * 64 * T_{WWDT}$	106.66 ms
0111	128	$128 * 64 * T_{WWDT}$	213.33 ms
1000	192	$192 * 64 * T_{WWDT}$	320 ms
1001	256	$256 * 64 * T_{WWDT}$	426.66 ms
1010	384	$384 * 64 * T_{WWDT}$	640 ms
1011	512	$512 * 64 * T_{WWDT}$	853.33 ms
1100	768	$768 * 64 * T_{WWDT}$	1.28 s
1101	1024	$1024 * 64 * T_{WWDT}$	1.706 s
1110	1536	$1536 * 64 * T_{WWDT}$	2.56 s
1111	2048	$2048 * 64 * T_{WWDT}$	3.413 s

Table 6.9-1 WWDT Prescaler Value Selection

6.9.5.1 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.

6.9.5.2 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.

6.9.5.3 WWDT Reset System

Figure 6.9-3 shows three cases of WWDT reset and reload behavior.

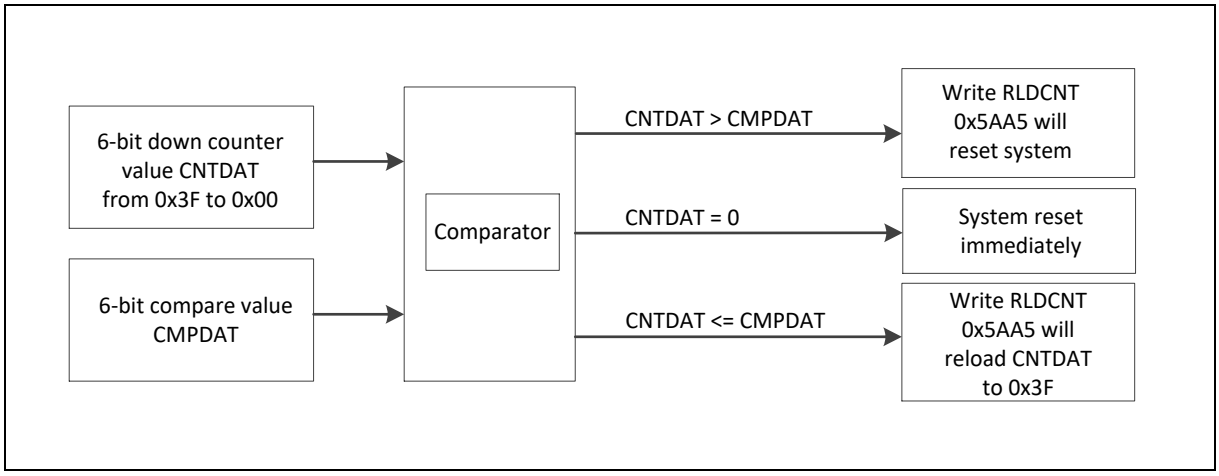


Figure 6.9-3 WWDT Reset and Reload Behavior

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 6.9-4.

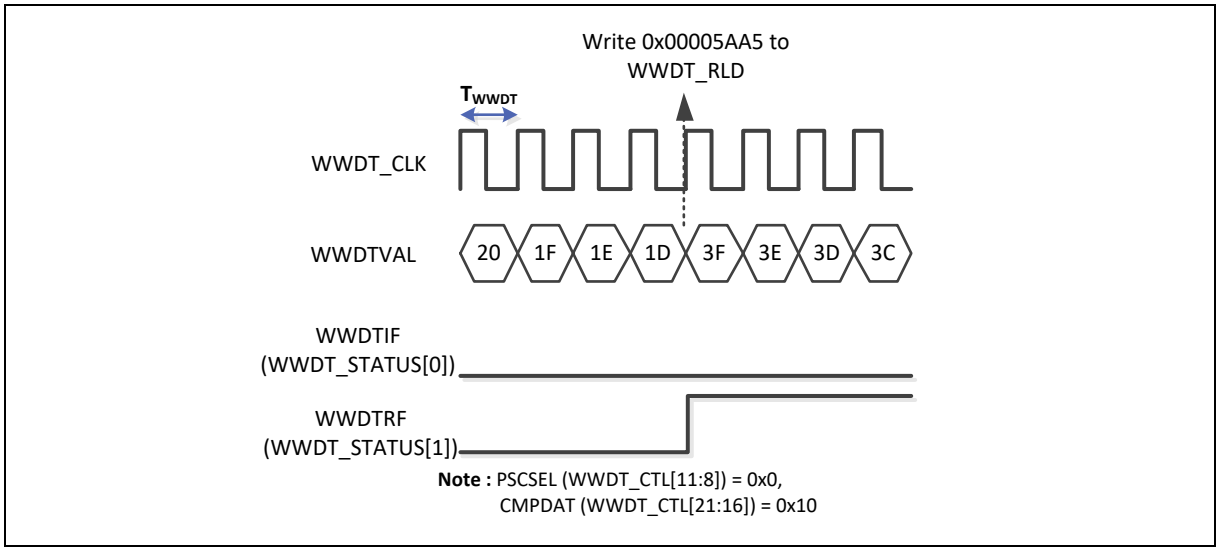


Figure 6.9-4 WWDT Reload Counter When CNTDAT > CMPDAT

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 WWDT counter value reached 0 and generate WWDT reset system signal to inform system reset. Figure 6.9-5 shows the waveform of WWDT reload counter when CNTDAT < CMPDAT and Figure 6.9-6 shows that WWDT generates reset system signal (WWDTRF) if user doesn't write 0x00005AA5 to WWDT_RLD before WWDT counter value reaches 0.

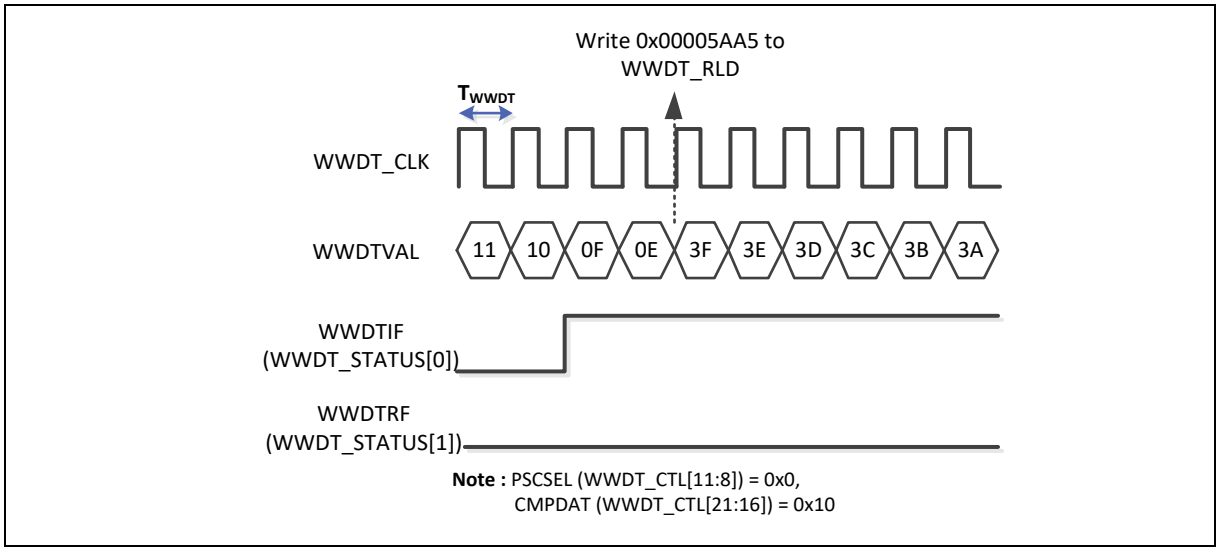


Figure 6.9-5 WWDT Reload Counter When WWDT_CNT < WINCMP

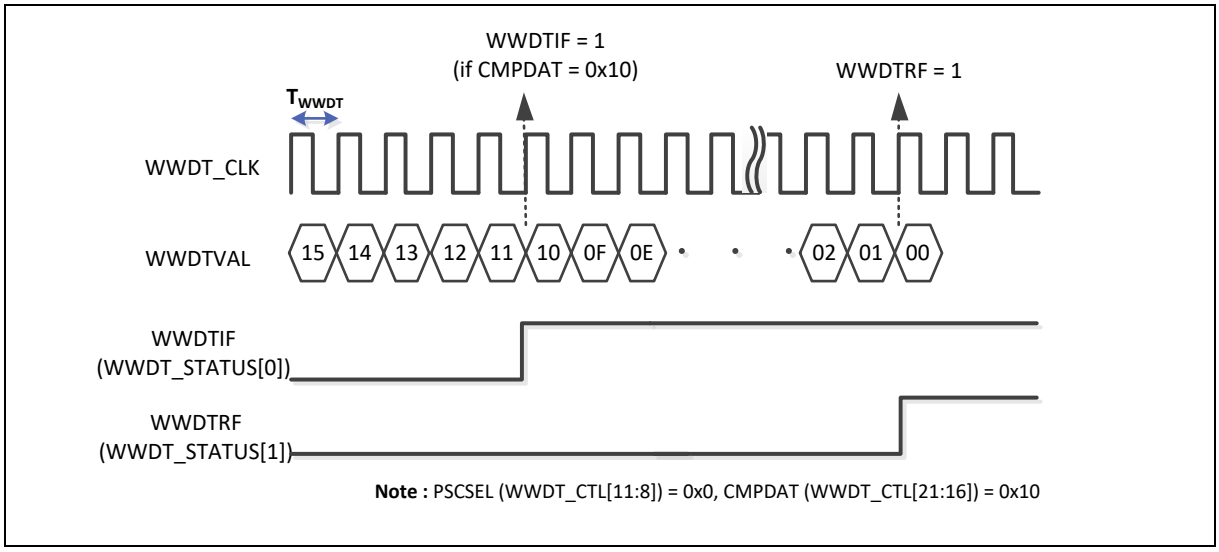


Figure 6.9-6 WWDT Interrupt and Reset Signals

6.9.5.4 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 6.9-2.

If user sets CMPDATA as 0x3F and 0x0, the interrupt doesn't occur. The reset occurs when WWDT counts to 0x0, so the interrupt doesn't occur when CMPDATA is 0x0.

PSCSEL	Prescale Value	Valid CMPDAT Value
0000	1	0x3 ~ 0x3E
0001	2	0x2 ~ 0x3E

Others	Others	0x1 ~ 0x3E

Table 6.9-2 CMPDAT Setting Limitation

6.9.5.5 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, the WWDT counter will keep counting no matter CPU is held by ICE or not.

6.9.6 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 = 0x4004_0100**				
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

6.9.7 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 Register** 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 is larger than CMPDAT, WWDT reset signal will be generatec 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.

6.10 PWM Generator and Capture Timer (PWM)

6.10.1 Overview

The chip provides one PWM generator. 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 are used to generate PWM pulse, interrupt and trigger signal for ADC to start conversion.

The PWM generator supports two standard PWM output modes: Independent mode and Complementary mode. They have different 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.

6.10.2 Features

6.10.2.1 PWM Function Features

- Supports maximum clock frequency up to 48 MHz
- Supports up to one PWM module and 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 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 matches 0, period value or compared value
  - Brake condition happened
- Supports trigger ADC on the following events:
  - PWM counter matches 0, period value or compared value

6.10.2.2 Capture Function Features

- Supports up to 6 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

6.10.3 Block Diagram

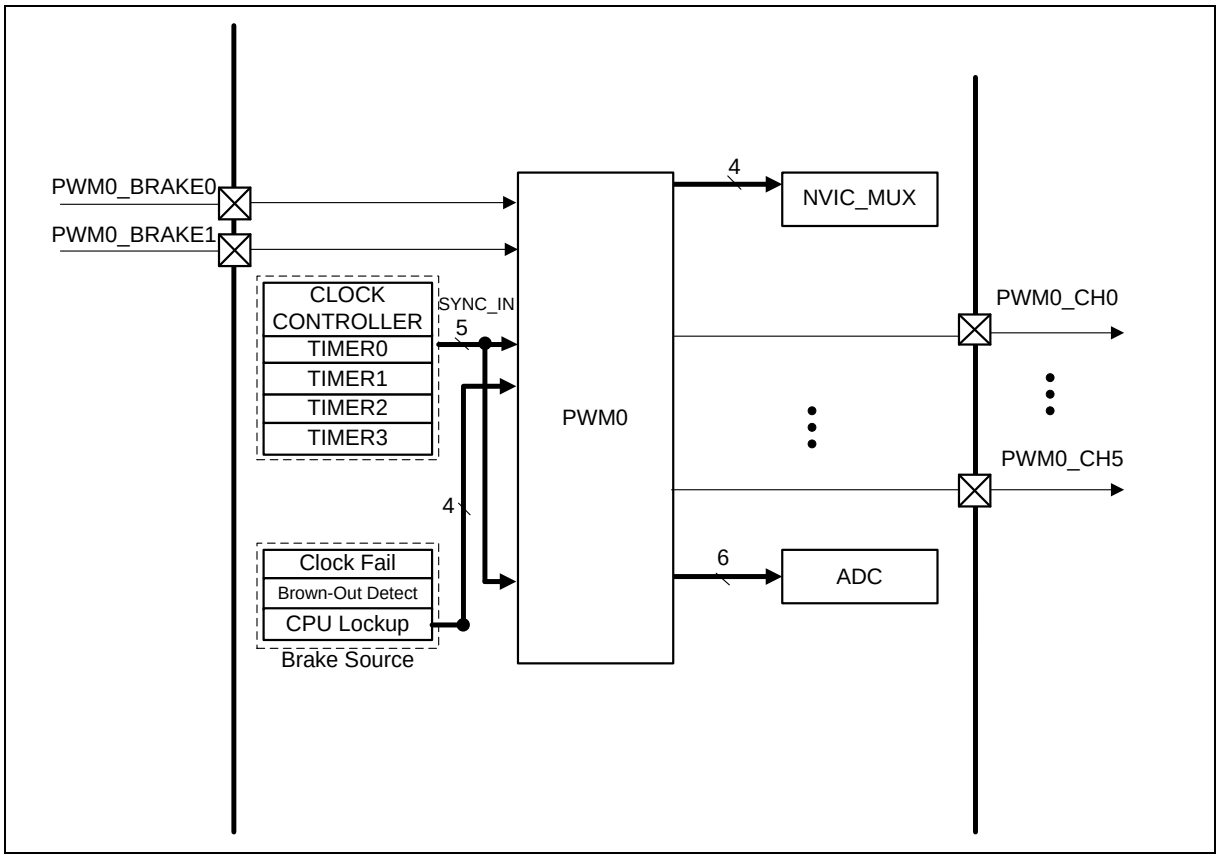


Figure 6.10-1 PWM Generator Overview Block Diagram

The PWM Clock frequency can be set equal to PCLK frequency as Figure 6.10-2.

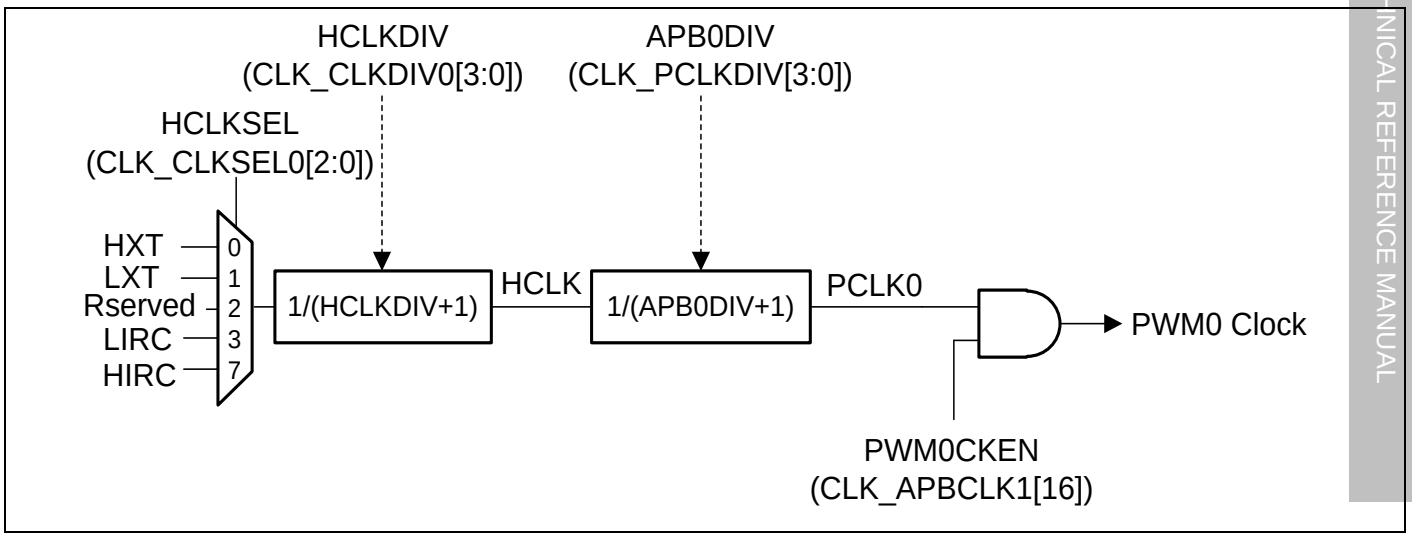


Figure 6.10-2 PWM System Clock Source Control

For the detailed register setting, please refer to Table 6.10-1. 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 6.10-3 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.

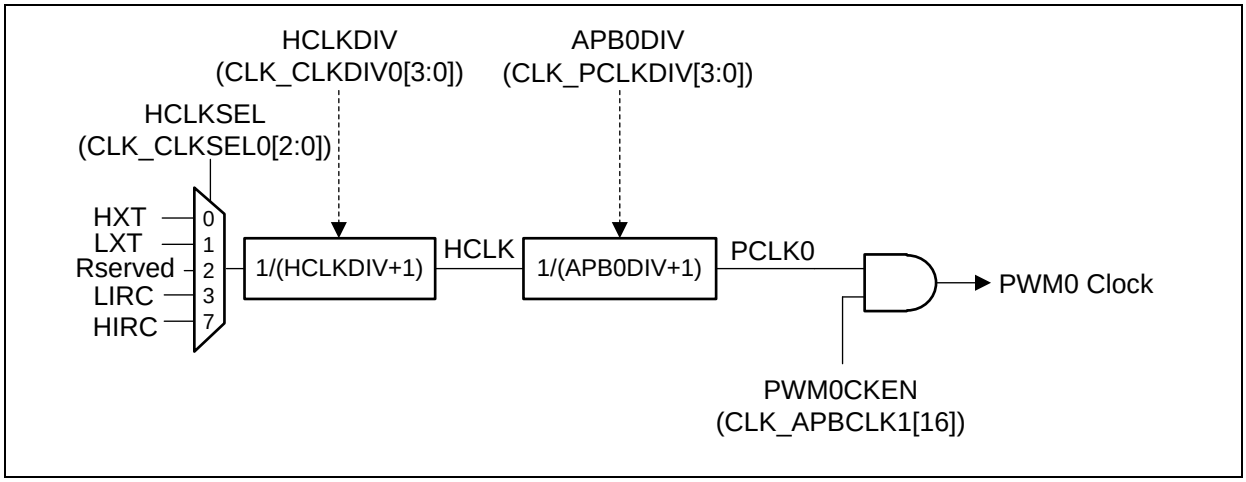


Figure 6.10-2 PWM System Clock Source Control

Frequency Ratio PCLK:PWM Clock	**HCLK**	**PCLK**	**PWM Clock**	**HCLKSEL CLK_CLKSEL0[2:0]**	**HCLKDIV CLK_CLKDIV0[3:0]**	**APB0DIV (CLK_PCLKDIV [2:0]),**
1:1	HCLK	PCLK	PCLK	Don't care	Don't care	Don't care

Table 6.10-1 PWM Clock Source Control Registers Setting Table

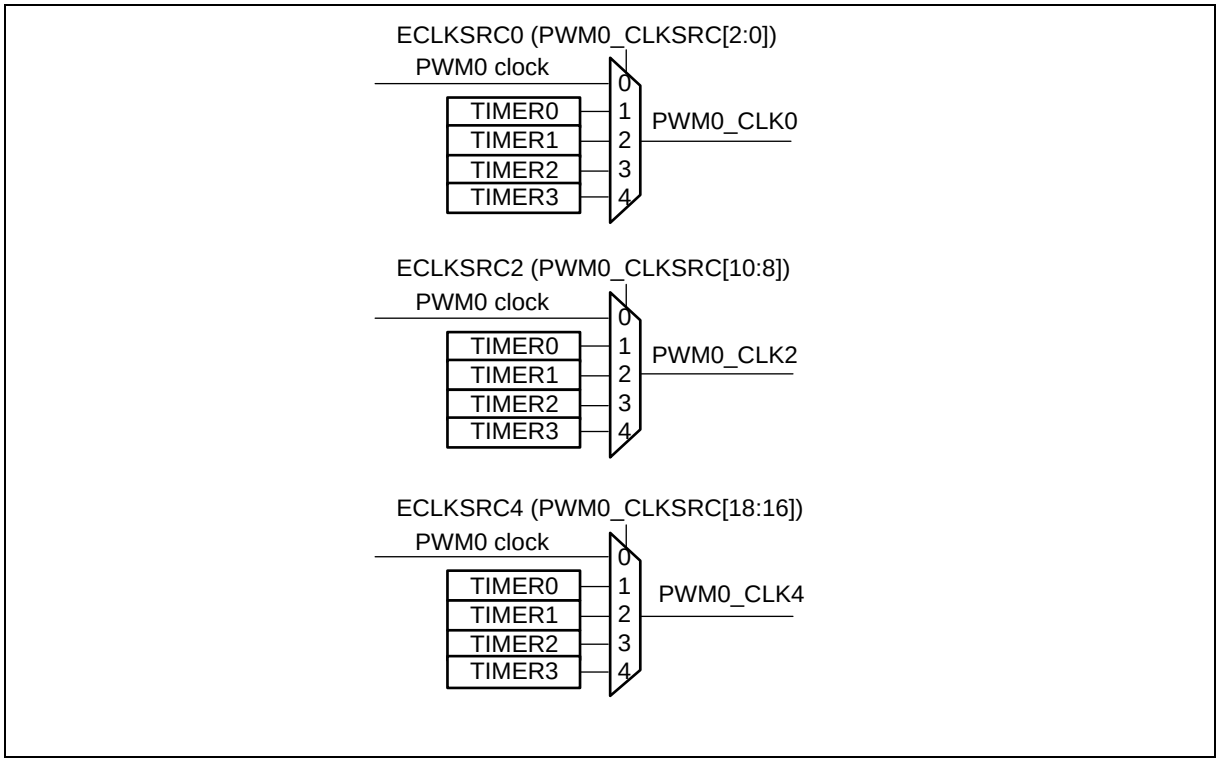


Figure 6.10-3 PWM Clock Source Control

Figure 6.10-4 and Figure 6.10-5 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 ADC 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.

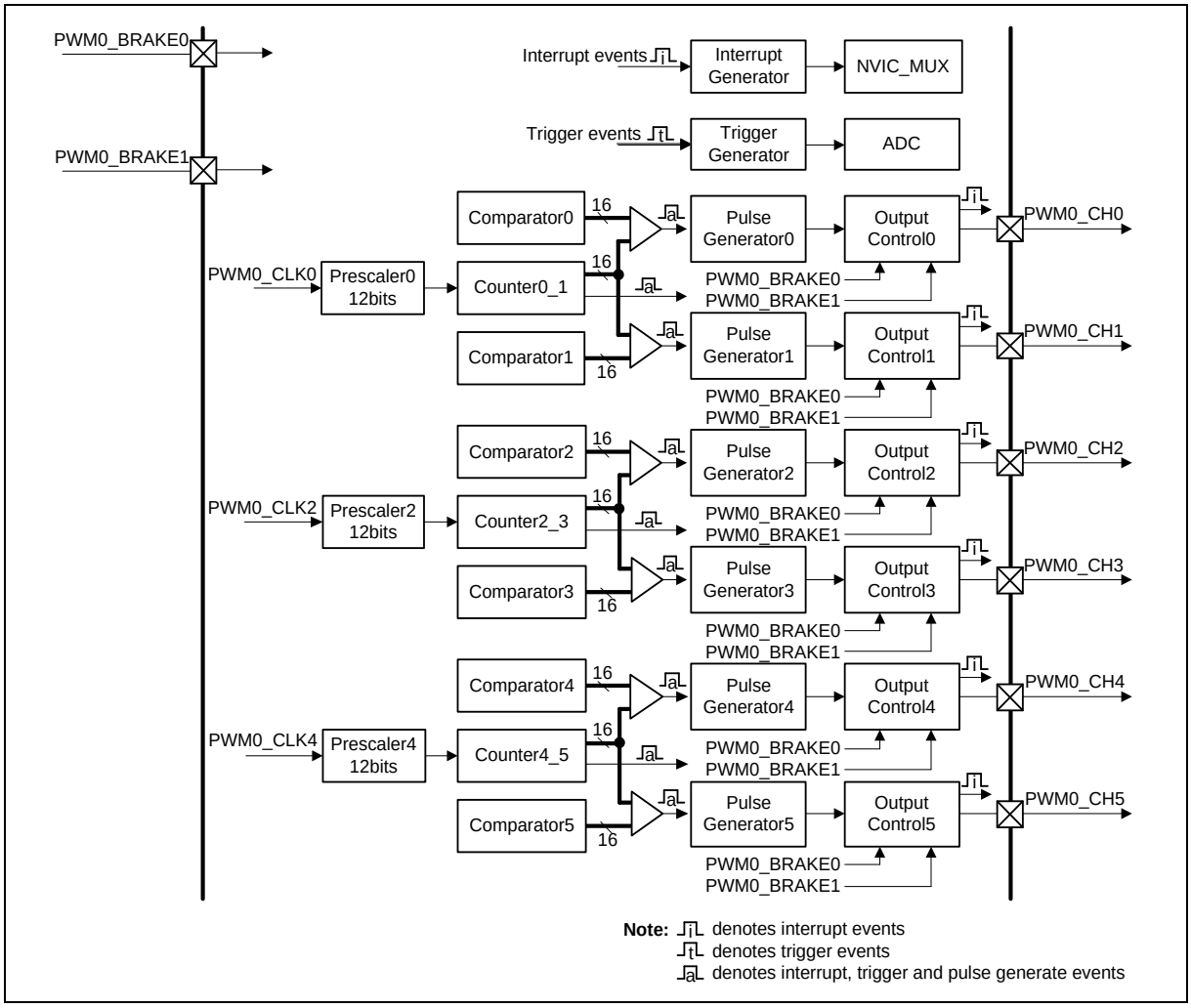


Figure 6.10-4 PWM Independent Mode Architecture Diagram

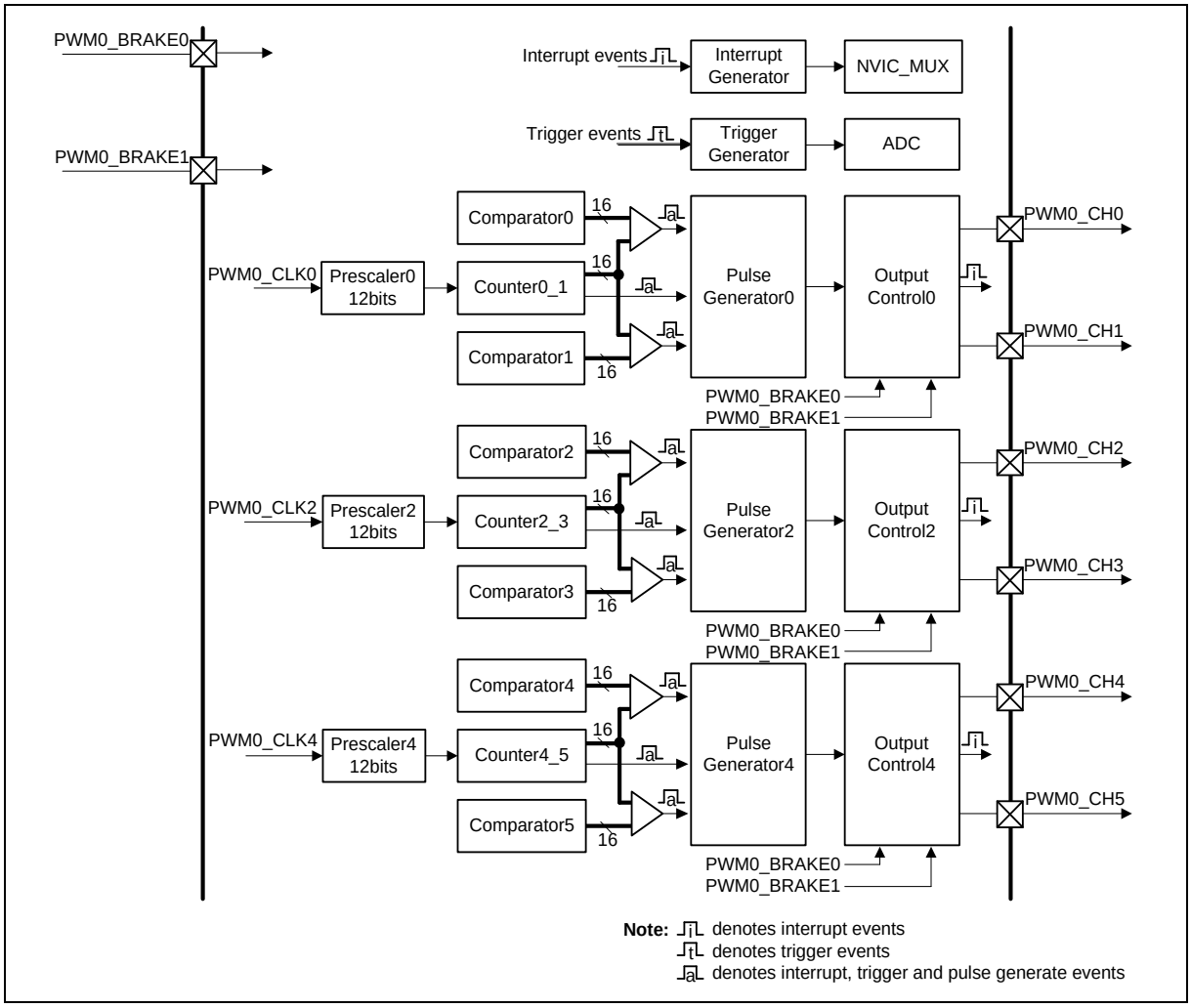


Figure 6.10-5 PWM Complementary Mode Architecture Diagram

6.10.4 Basic Configuration

6.10.4.1 PWM0 Basic Configuration

- Clock Source Configuration
  - Select the source of PWM0 peripheral clock on PWM0SEL (CLK_CLKSEL2[0])
  - Enable PWM0 peripheral clock in PWM0CKEN CLK_APBCLK1[16]).
- Reset Configuration
  - Reset PWM0 in PWM0RST (SYS_IPRST2[16])
- Pin Configuration

Group	Pin Name	GPIO	MFP
PWM0	PWM0_BRAKE0	PA.1	MFP30
		PC.0	MFP30
	PWM0_BRAKE1	PA.0	MFP30
		PA.2	MFP30

	PWM0_CH0	PA.1	MFP5
		PA.3	MFP5
		PA.5	MFP5
		PB.5	MFP5
		PB.7	MFP5
		PC.0	MFP5
		PC.2	MFP5
		PC.4	MFP5
		PC.6	MFP5
		PD.2	MFP2
		PD.4	MFP2
	PWM0_CH1	PA.0	MFP5
		PA.2	MFP5
		PA.4	MFP5
		PB.4	MFP5
		PB.6	MFP5
		PC.1	MFP5
		PC.3	MFP5
		PC.5	MFP5
		PC.7	MFP5
		PD.3	MFP2
		PD.5	MFP2
	PWM0_CH2	PA.1	MFP6
		PA.3	MFP6
		PA.5	MFP6
		PB.5	MFP6
		PB.7	MFP6
		PC.0	MFP6
		PC.2	MFP6
		PC.4	MFP6
		PC.6	MFP6
		PD.6	MFP2
	PWM0_CH3	PA.0	MFP6
		PA.2	MFP6
		PA.4	MFP6

		PB.4	MFP6
		PB.6	MFP6
		PC.1	MFP6
		PC.3	MFP6
		PC.5	MFP6
		PC.7	MFP6
		PD.7	MFP2
	PWM0_CH4	PA.1	MFP7
		PA.3	MFP7
		PA.5	MFP7
		PB.5	MFP7
		PB.7	MFP7
		PC.0	MFP7
		PC.2	MFP7
		PC.4	MFP7
		PC.6	MFP7
		PD.0	MFP2
	PWM0_CH5	PA.0	MFP7
		PA.2	MFP7
		PA.4	MFP7
		PB.4	MFP7
		PB.6	MFP7
		PC.1	MFP7
		PC.3	MFP7
		PC.5	MFP7
		PC.7	MFP7
		PD.1	MFP7

6.10.5 Functional Description

6.10.5.1 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 6.10-6 is an example of PWM channel 0 prescale waveform. The prescale counter will reload CLKPSC at the beginning of the next prescale counter down-count.

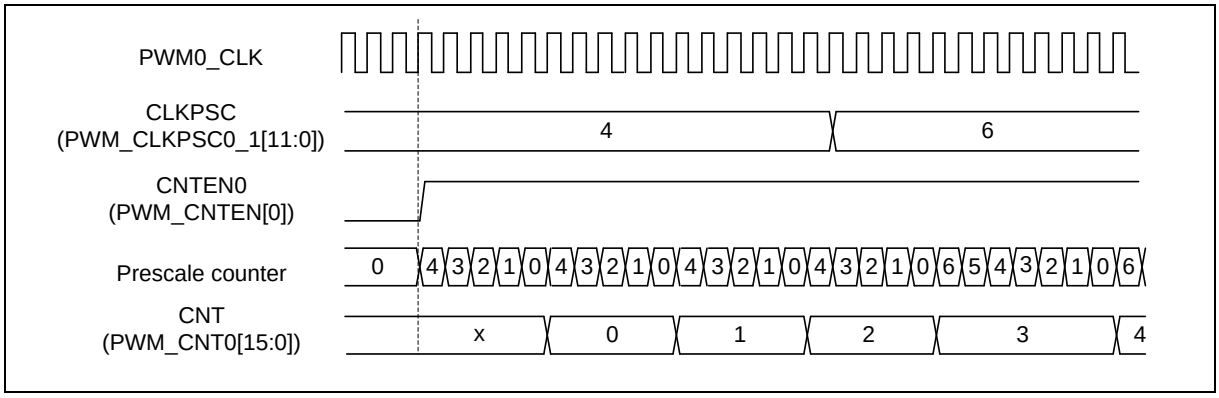


Figure 6.10-6 PWM0_CH0 Prescaler Waveform in Up Counter Type

6.10.5.2 PWM Counter

PWM supports 3 counter types operation: Up Counter, Down Counter and Up-Down Counter types.

To clear PWM counter, take PWM channel0 for an example, CNT(PWM_CNT0[15:0]) can clear to 0x00 by CNTCLR0 (PWM_CNTCLR[0]). CNT will be cleared when prescale counter counts to 0, and CNTCLR0 will be set 0 by hardware automatically.

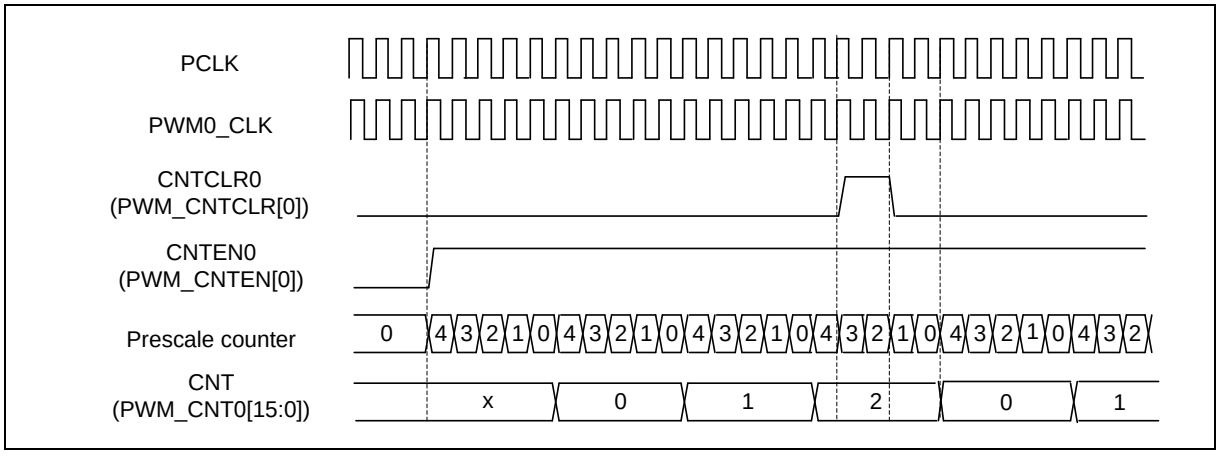


Figure 6.10-7 PWMx Counter Waveform When Setting Clear Counter

6.10.5.3 Up Counter Type

When PWM counter is set to up counter type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) is 0x0, it starts up-counting from 0 to PERIOD (PWM_PERIODn[15:0], where n denotes channel number) to complete a PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0]) 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 6.10-8 shows an example of up counter, wherein

PWM period time = (PERIOD+1) *(CLKPSC+1)* PWMx_CLK.

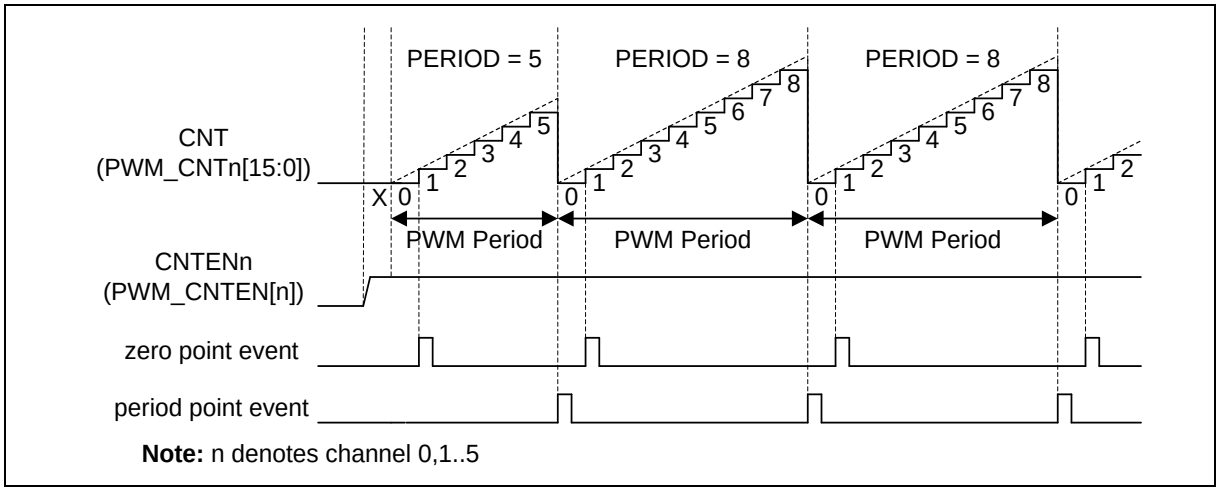


Figure 6.10-8 PWM Up Counter Type

6.10.5.4 Down Counter Type

When PWM counter is set to down counter type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) 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]) 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 6.10-9 shows an example of down counter, wherein

PWM period time = (PERIOD+1) * (CLKPSC+1) * PWMx_CLK.

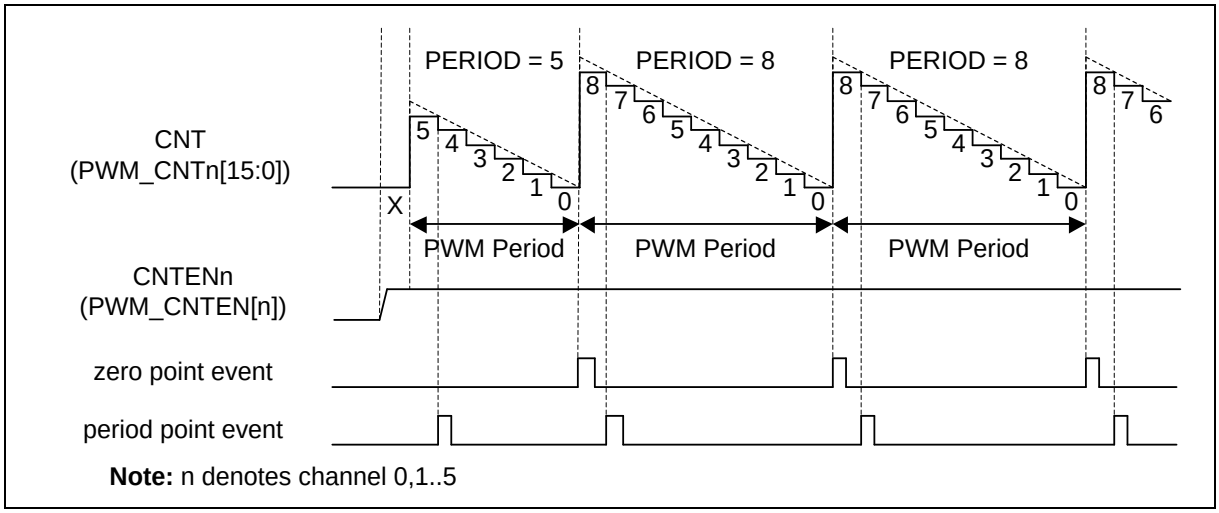


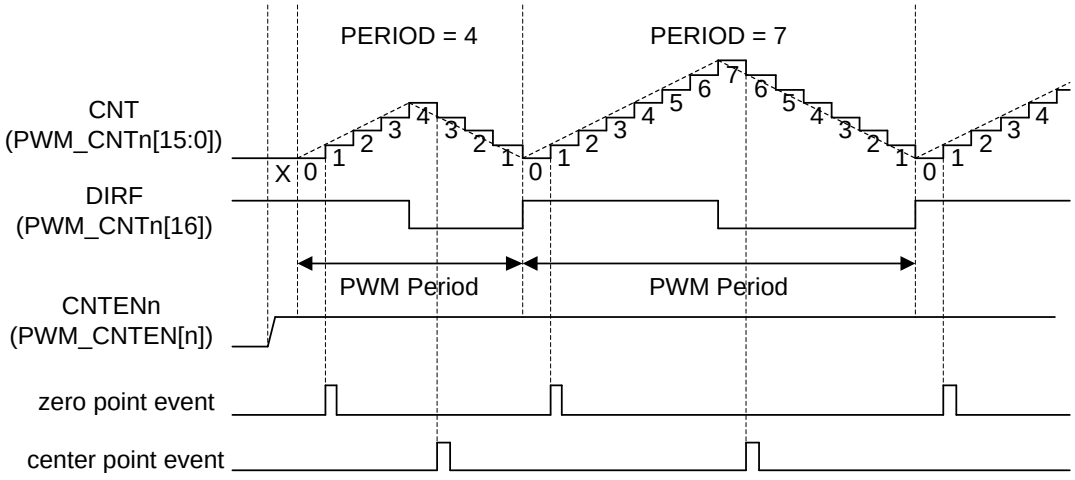
Figure 6.10-9 PWM Down Counter Type

6.10.5.5 Up-Down Counter Type

When PWM counter is set to up-down count type, CNTTYPEn (PWM_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 PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0]) 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 6.10-10 shows an example of up-down counter, wherein

PWM period time =(2*PERIOD) * (CLKPSC+1) * PWMx_CLK.

The DIRF (PWM_CNTn[16]) bit is counter direction indicator flag, where high is up counting, and low is down counting.

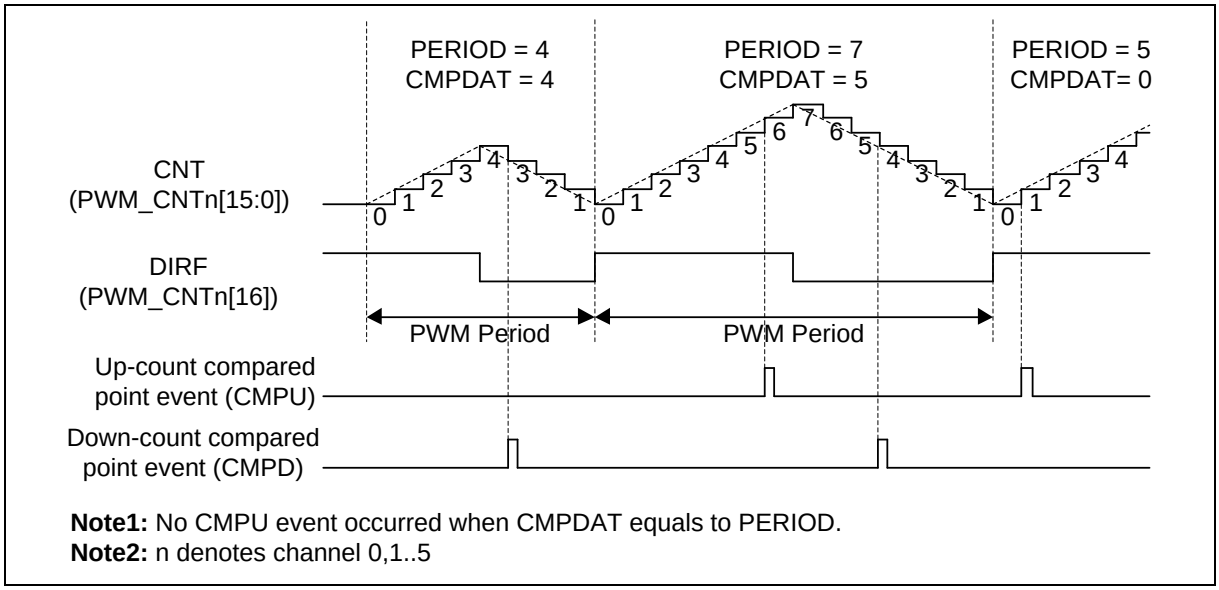


Note1: When in up-down count type, period interrupt flag occurs at center point event.
Note2: n denotes channel 0,1..5

Figure 6.10-10 PWM Up-Down Counter Type

6.10.5.6 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 ADC. In up-down counter type, two events will be generated in a PWM period as shown in Figure 6.10-11. 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 6.10-11 PWM Compared point Events in Up-Down Counter Type

6.10.5.7 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]) as the active PERIOD buffer register, CMPBUF (PWM_CMPBUFn[15:0]) 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 Figure 6.10-12, in period loading mode, writing PERIOD and CMPDAT through software, PWM will load new values to their buffer PBUF (PWM_PBUFn[15:0]) and CMPBUF (PWM_CMPBUFn[15:0]) at start of the next period without affecting the current period counter operation.

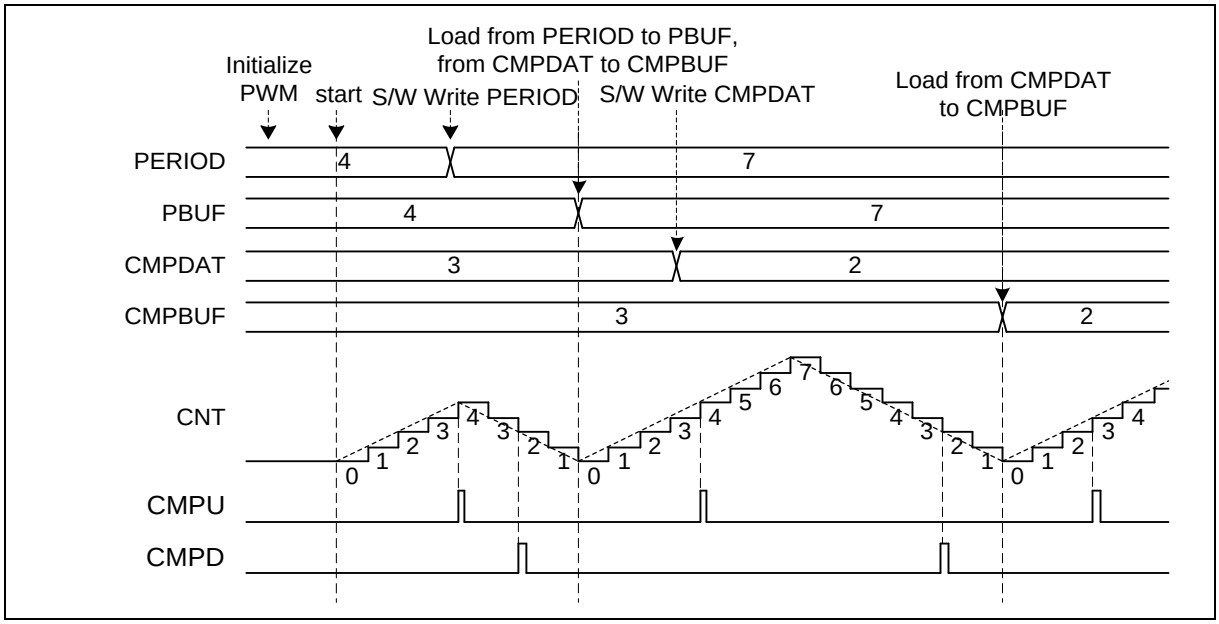


Figure 6.10-12 PWM Double Buffering Illustration

6.10.5.8 Period Loading Mode

When immediately loading mode and center loading mode are disabled that IMMLDENn (PWM_CTL0[21:16]) and CTRLDn (PWM_CTL0[5:0]) are set to 0, PWM operates at period Loading mode. In period Loading mode, PERIOD(PWM_PERIODn[15:0]) and CMP(PWM_CMPDATn[15:0]) 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 6.10-13 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 6.10-13. 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.

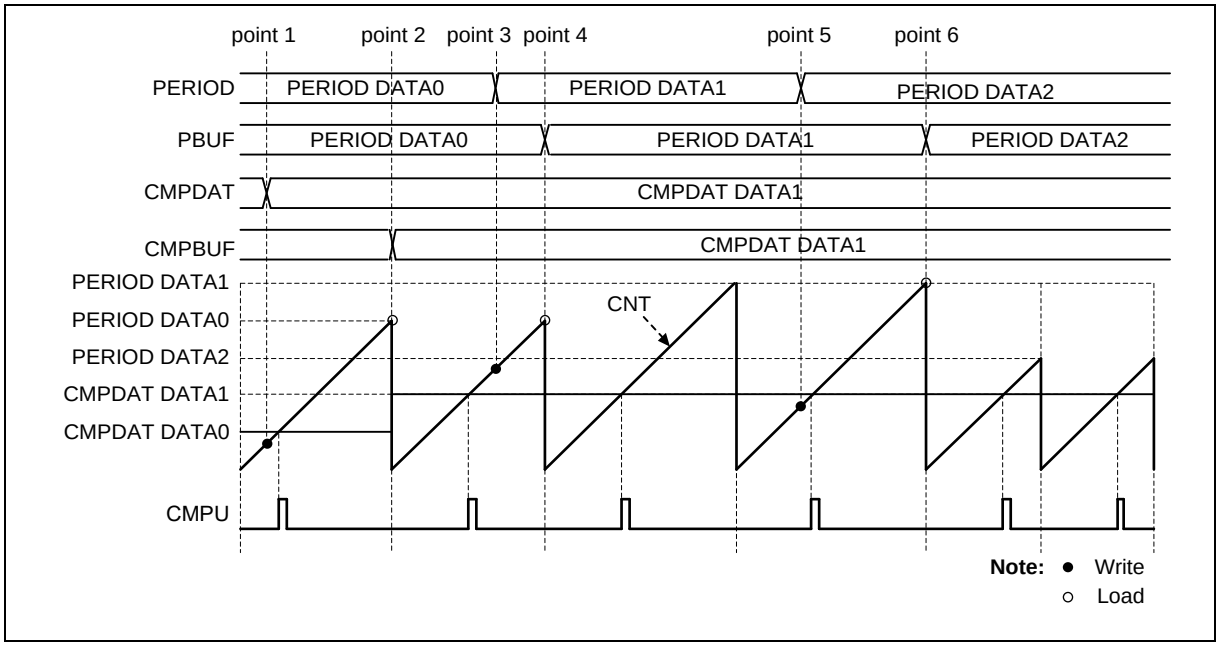


Figure 6.10-13 Period Loading in Up-Count Mode

6.10.5.9 Immediately Loading Mode

If the IMMLDENn (PWM_CTL0[21:16]) bit is set to 1, PWM operates at immediately loading mode. In immediately loading mode, when user update PERIOD (PWM_PERIODn[15:0]) or CMP (PWM_CMPDATn[15:0]), PERIOD or CMPDAT will be load to active PBUF (PWM_PBUFn[15:0]) or CMPBUF (PWM_CMPBUFn[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 counts to 0xFFFF and prescale also counts to 0, the flag CNTMAXF(PWMx_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 6.10-14 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.

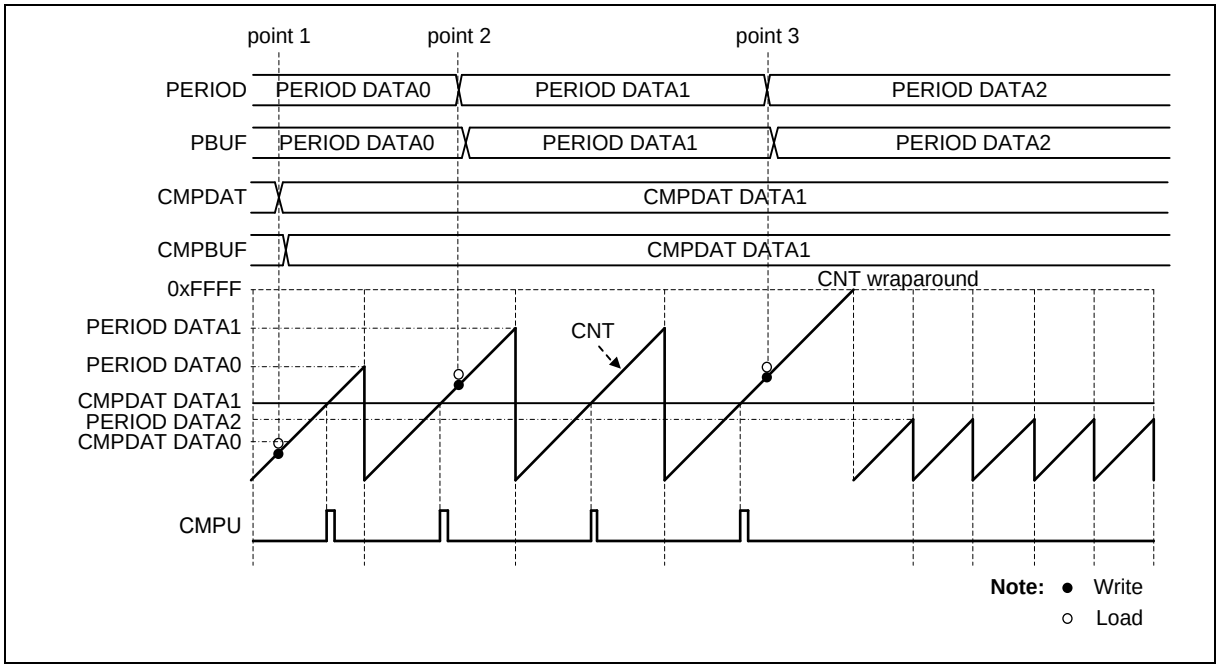


Figure 6.10-14 Immediately Loading in Up-Count Mode

6.10.5.10 Center Loading Mode

When the CTRLDn (PWM_CTL0[5:0]) bit is set to 1 and PWM counter is set to up-down count type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) is 0x2, PWM operates at center loading mode. In center loading mode, CMP(PWM_CMPDATn[15:0]) will load to active CMPBUF register in center of each period, that is, counter counts to PERIOD. PERIOD(PWM_PERIODn[15:0]) will all load to their active PBUF registers while each period is completed. Figure 6.10-15 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.

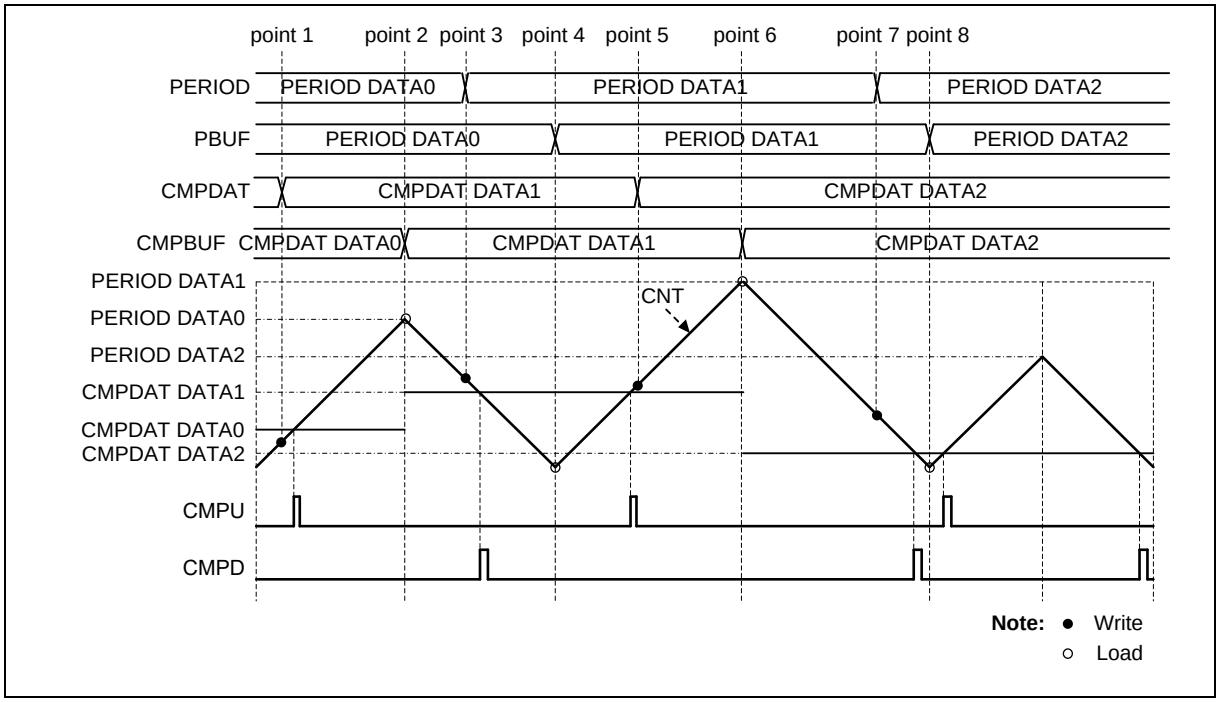


Figure 6.10-15 Center Loading in Up-Down-Count Mode

6.10.5.11 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 the CNTENn(PWM_CNTEN[n]) bit is set to 1 to enable PWM prescaler and start to run counter. The value of CLKPSC(PWM_CLKPSCn_m[11:0]), PERIOD(PWM_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, PWM counter will be set to 0.

6.10.5.12 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 6.10-16. 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]) is equal to CMP(PWM_CMPDATn[15:0]) when counting up. CMPD denotes CNT bits is equal to CMP bits when counting down.

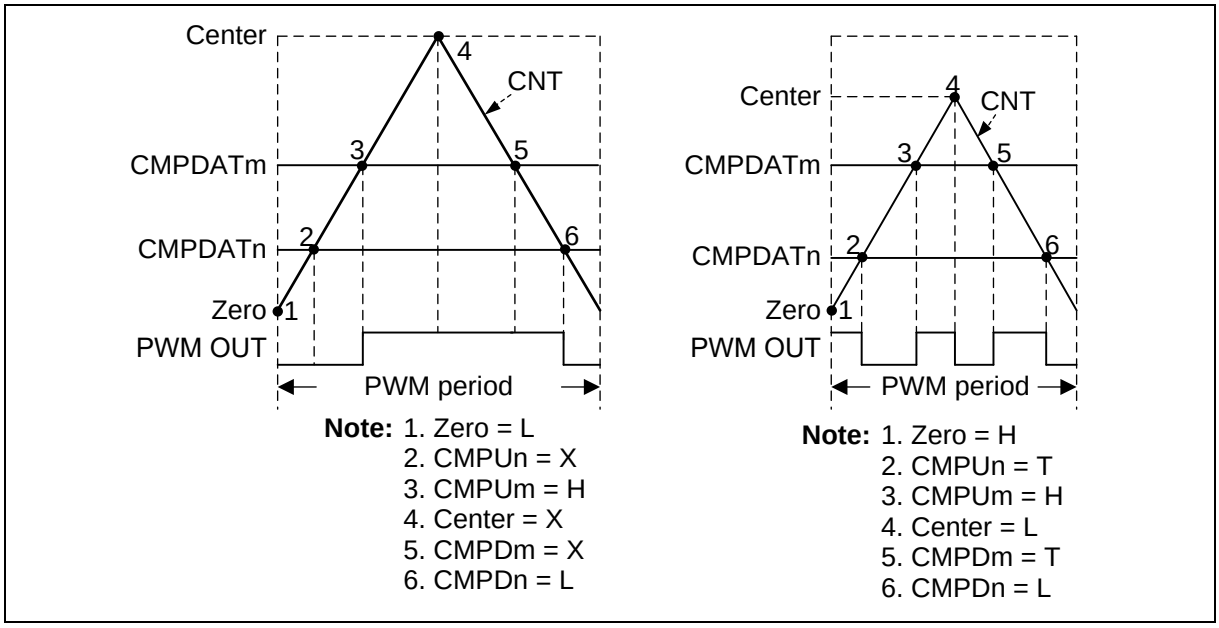


Figure 6.10-16 PWM Pulse Generation

The generation events may sometimes be set to the same value, as the reason, events priority between different counter types are list below, up counter type (Table 6.10-2), down counter type (Table 6.10-3) and up-down counter type (Table 6.10-4). By using event priority, user can easily generate 0% to 100% duty pulse as shown in Figure 6.10-17.

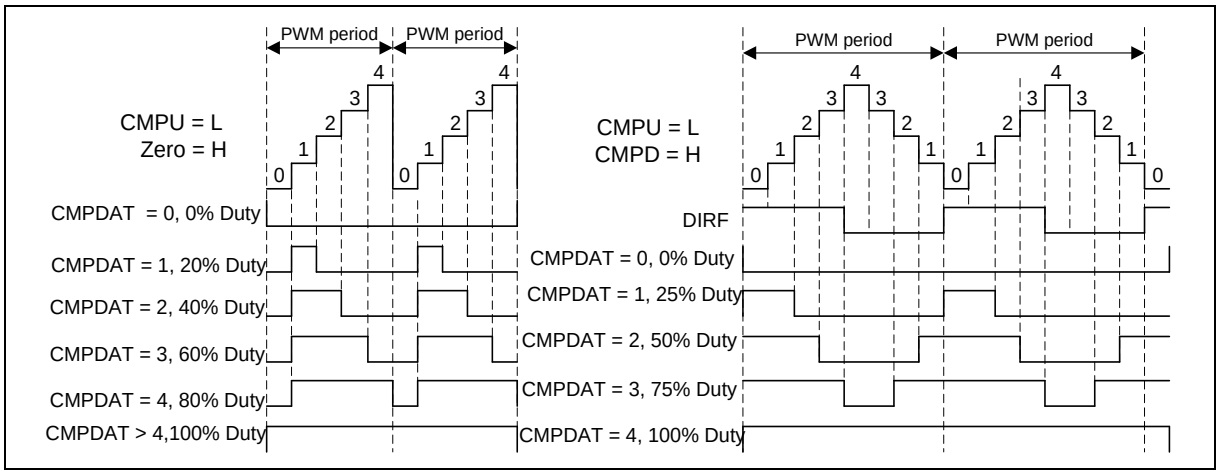


Figure 6.10-17 PWM 0% to 100% Pulse Generation

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 6.10-2 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 6.10-3 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 6.10-4 PWM Pulse Generation Event Priority for Up-Down-Counter

6.10.5.13 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.

6.10.5.14 Independent Mode

By default, the PWM is operating in independent mode, independent mode is enabled when channel n corresponding PWMMODEn (PWM_CTL1[26:24]) bit is set to 0. In this mode six PWM channels: PWM_CH0, PWM_CH1, PWM_CH2, PWM_CH3, PWM_CH4 and PWM_CH5 are running off its own period and duty as shown in Figure 6.10-18.

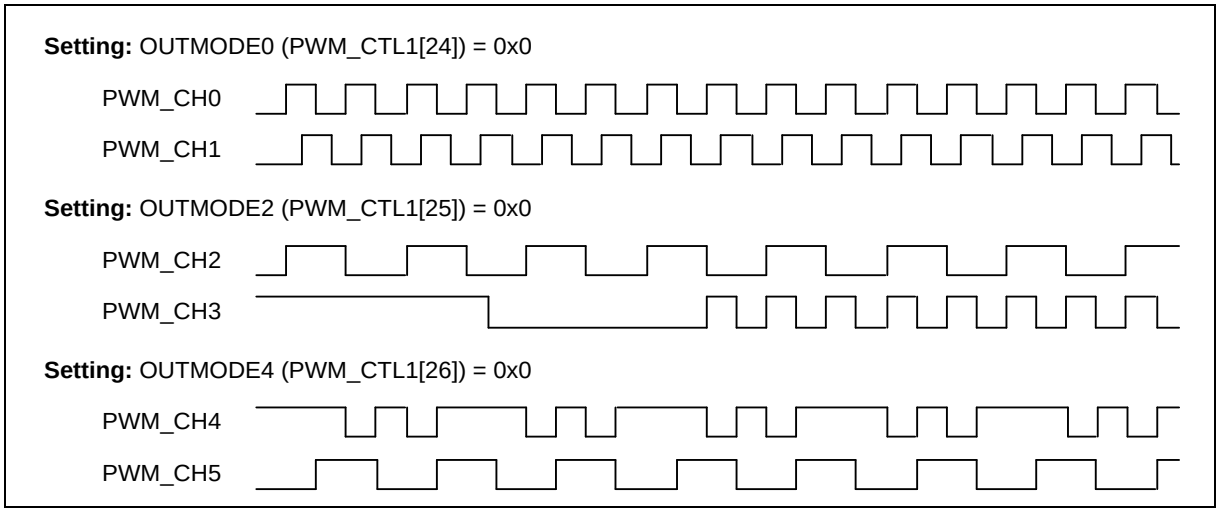


Figure 6.10-18 PWM Independent Mode Waveform

6.10.5.15 Complementary mode

Complementary mode is enabled when the paired channel corresponding PWMMODEn (PWM_CTL1[26:24]) 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 6.10-19.

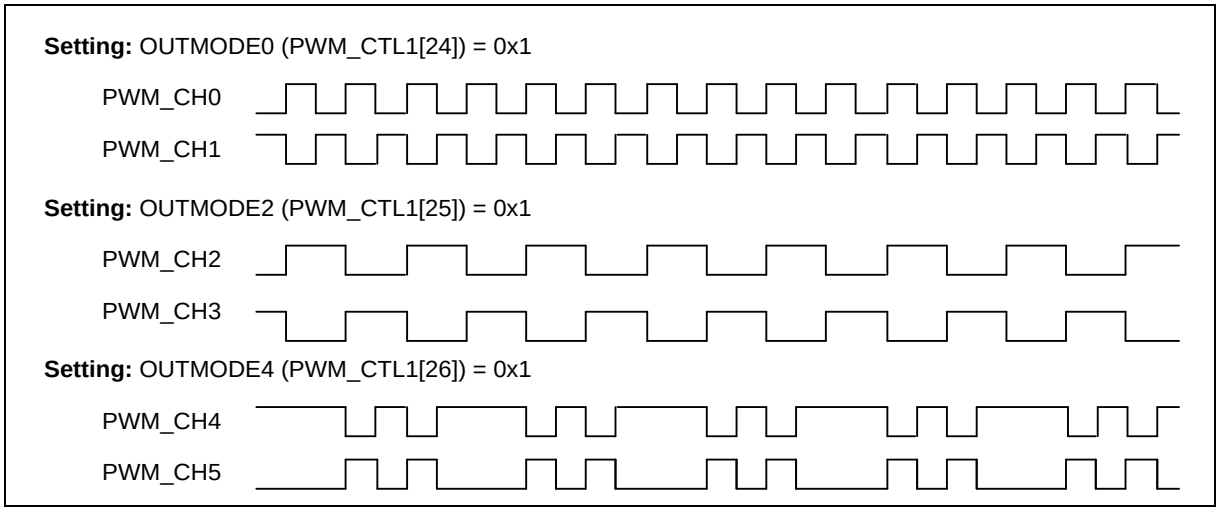


Figure 6.10-19 PWM Complementary Mode Waveform

6.10.5.16 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 6.10-20. In complementary mode, it needs two more steps to precede these four steps, Complementary channels and Dead-Time Insertion as shown in Figure 6.10-21.

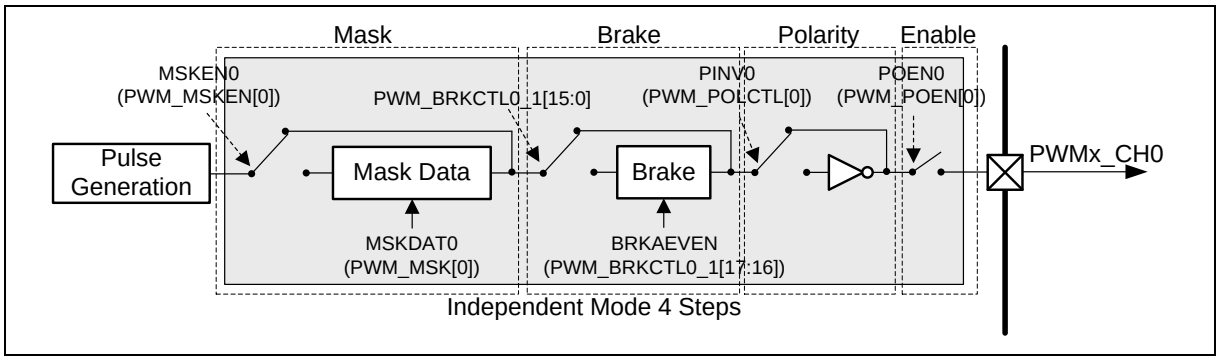


Figure 6.10-20 PWMx_CH0 Output Control in Independent Mode

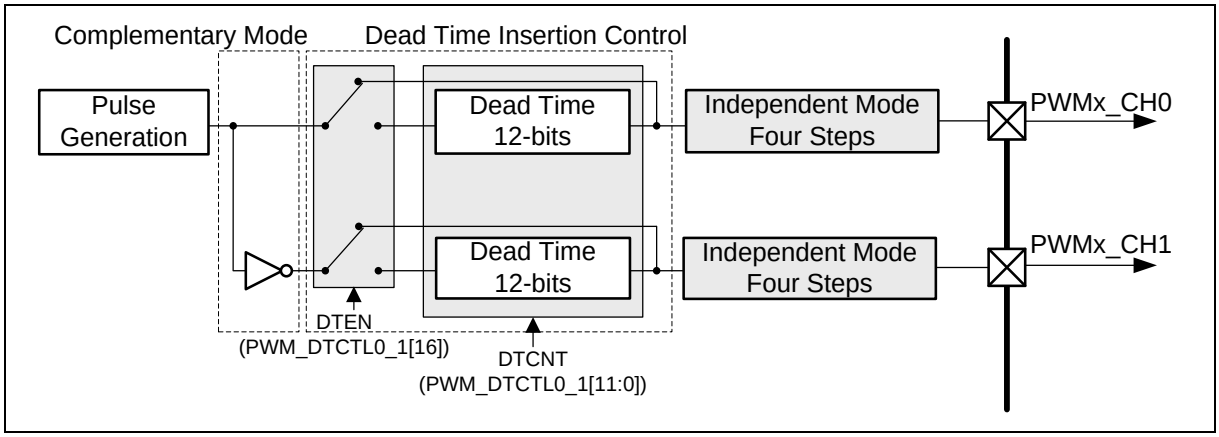


Figure 6.10-21 PWMx_CH0 and PWMx_CH1 Output Control in Complementary Mode

6.10.5.17 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 mechansism to the proper operation of the complementary system. By setting corresponding channel n 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 6.10-22 indicates the dead-time insertion for one pair of PWM signals.

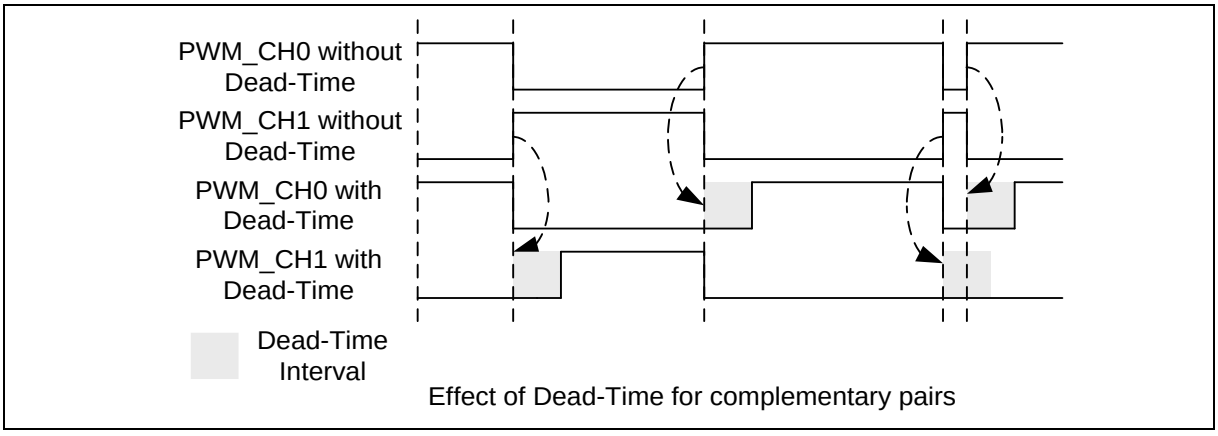


Figure 6.10-22 Dead-Time Insertion

6.10.5.18 PWM Mask Output Function

Each of the PWM channel output value can be manually overridden with the settings in the PWM Mask Enable Control Register (PWM_MSKEN) and the PWM Masked 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 6.10-23 shows an example of how PWM mask control can be used for the override feature.

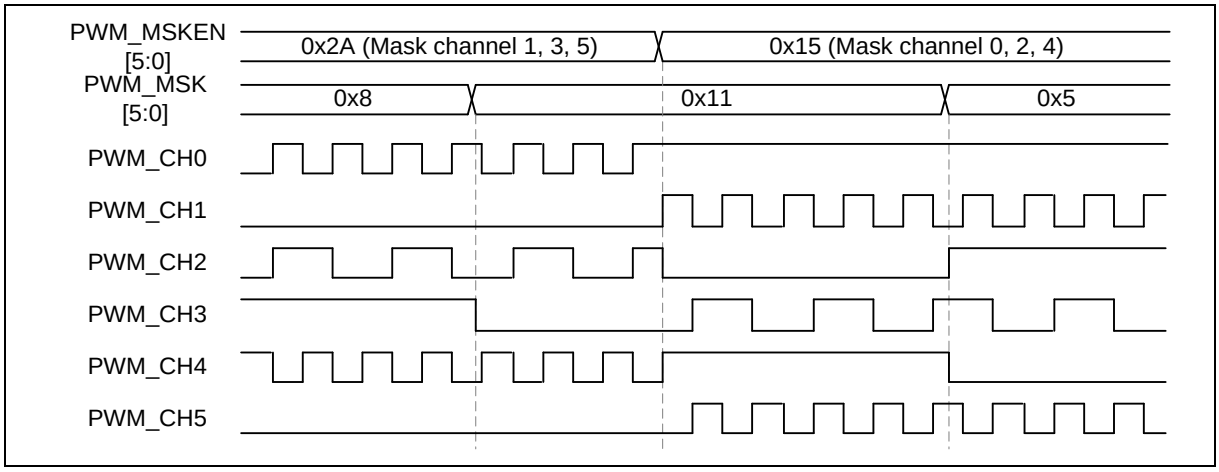


Figure 6.10-23 Illustration of Mask Control Waveform

6.10.5.19 PWM Brake

Each PWM module has two external input brake control signals. User can select active brake pin source is from PWMx_BRAKEy pin by BKxSRC bits of BNF register(x=0,1, y=0,1). The external signals will be filtered by a 3-bit noise filter. User can enable the noise filter function by BRKxNFEN bits of BNF reigster, and noise filter sampling clock can be selected by setting BRKxNFSEL bits of BNF register to fit different noise properties. Moreover, by setting the BRKxFCNT 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 the BRKxPINV (x denotes input external pin 0 or 1) bits of BNF register to realize the polarity setup for the brake control signals. Set BRKxPINV bit to 0, brake event will occurred when PWMx_BRAKEy(x=0,1, y=0,1) pin status is from low to high; set BRKxPINV to 1, brake event will occurred when PWMx_BRAKEy pin status is from high to low.

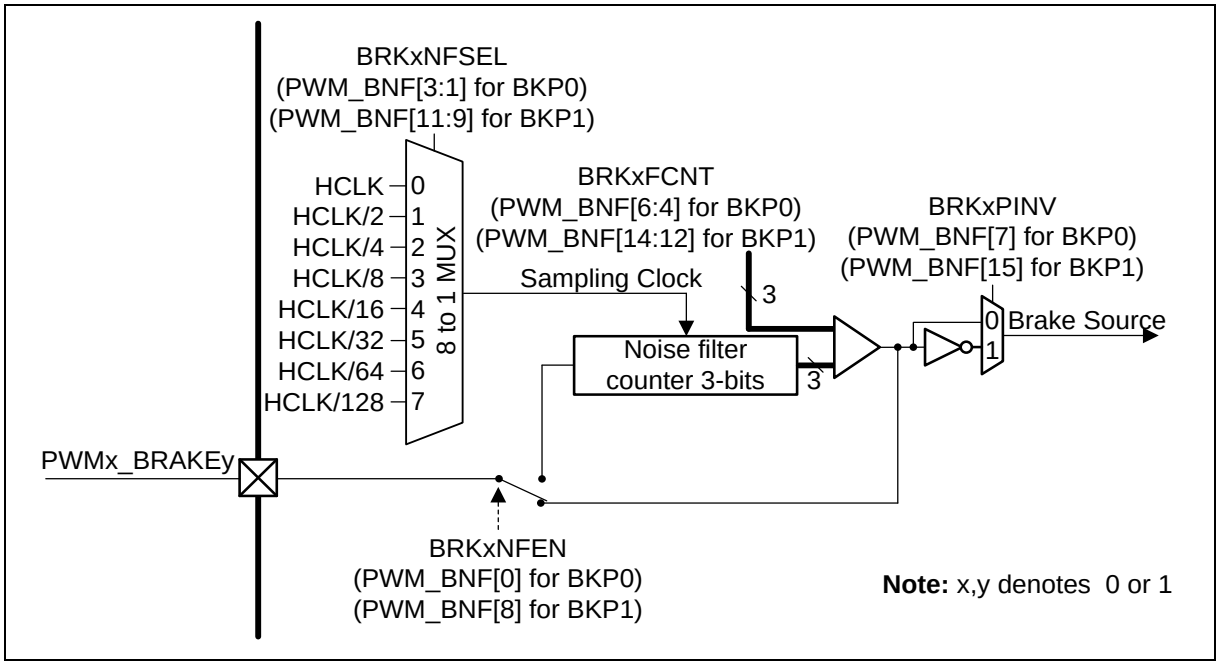


Figure 6.10-24 Brake Noise Filter Block Diagram

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 Figure 6.10-25. 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.

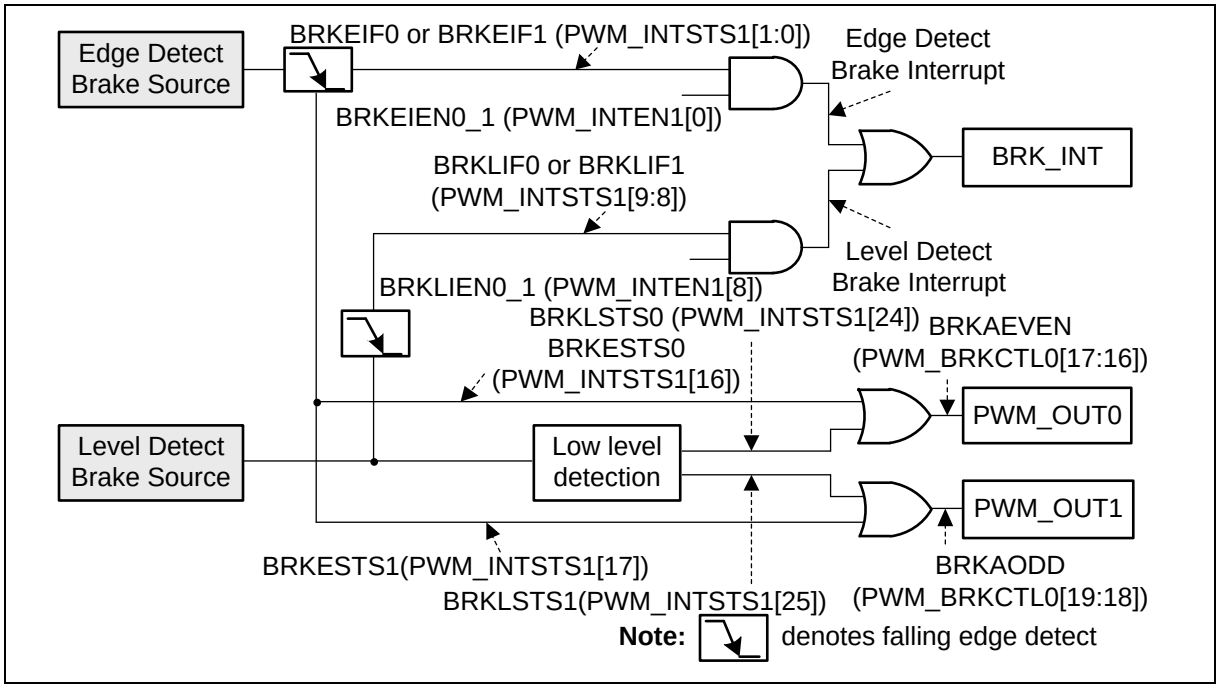


Figure 6.10-25 Brake Block Diagram for PWMx_CH0 and PWMx_CH1 Pair

Figure 6.10-26 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 6.10-27 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.

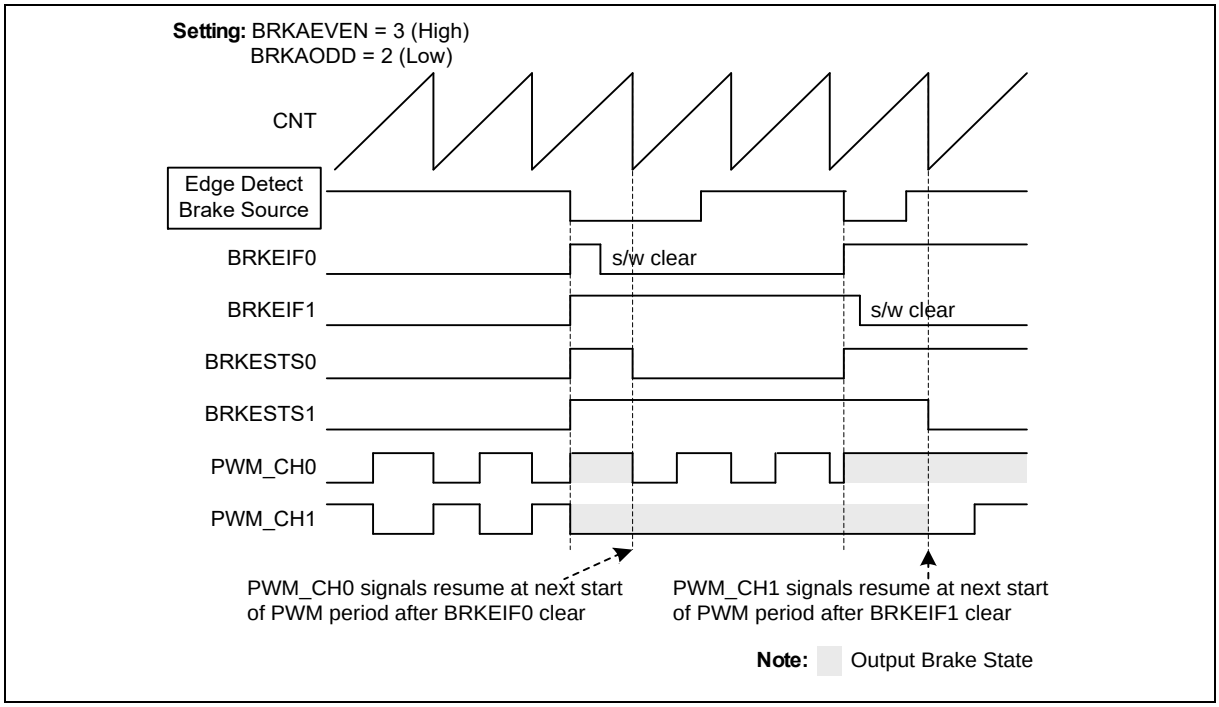


Figure 6.10-26 Edge Detector Waveform for PWMx_CH0 and PWMx_CH1 Pair

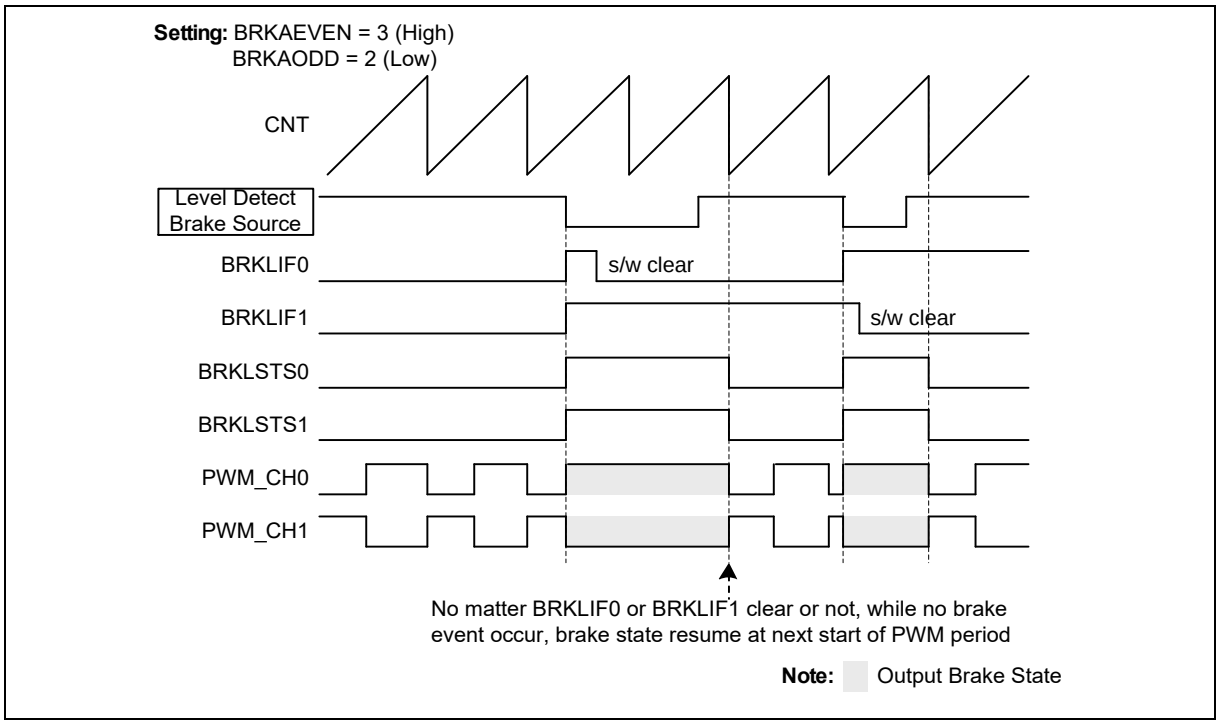


Figure 6.10-27 Level Detector Waveform for PWMx_CH0 and PWMx_CH1 Pair

The two kinds of detectors detect the same six brake sources: two from external input signals, two from analog comparators(ACMP), and one from system fail, and one from software triggered, that are shown in Figure 6.10-28. 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, and Core lockup. Figure 6.10-29 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.

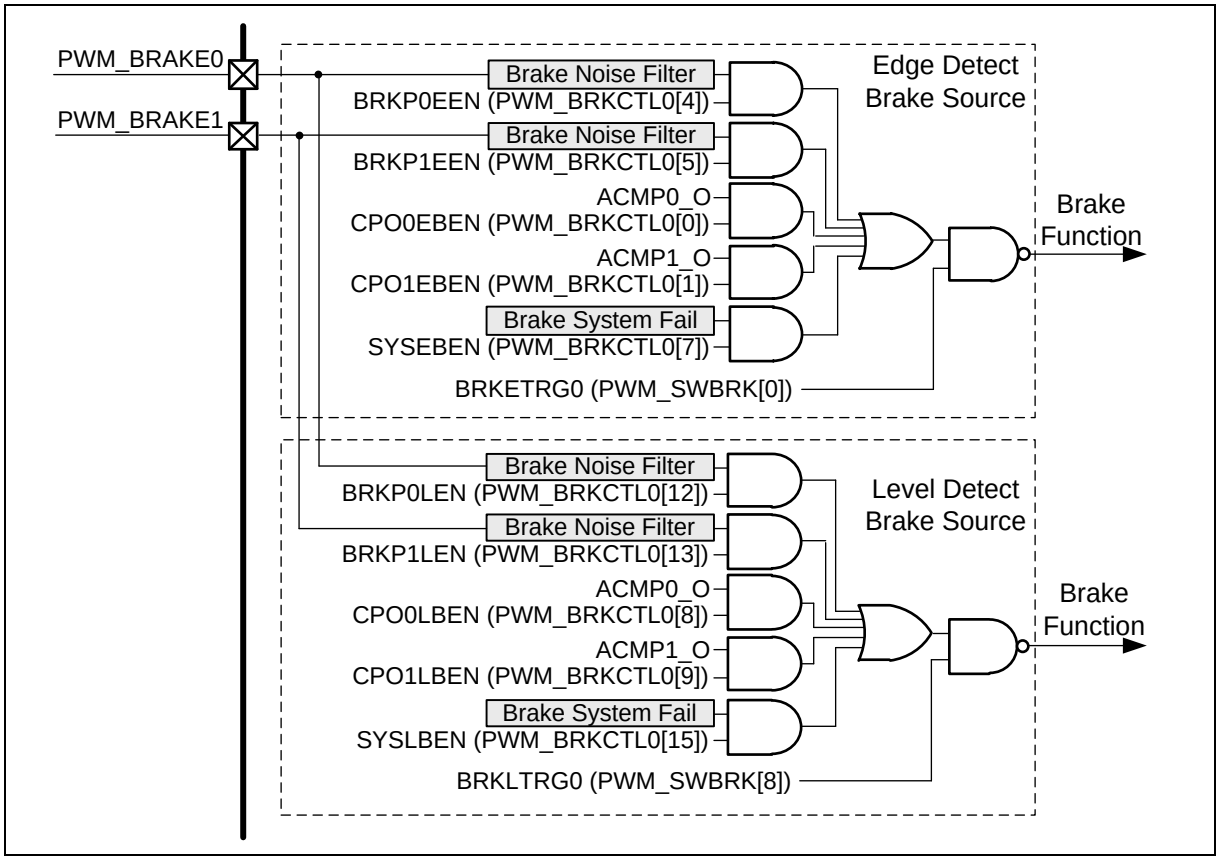


Figure 6.10-28 Brake Source Block Diagram

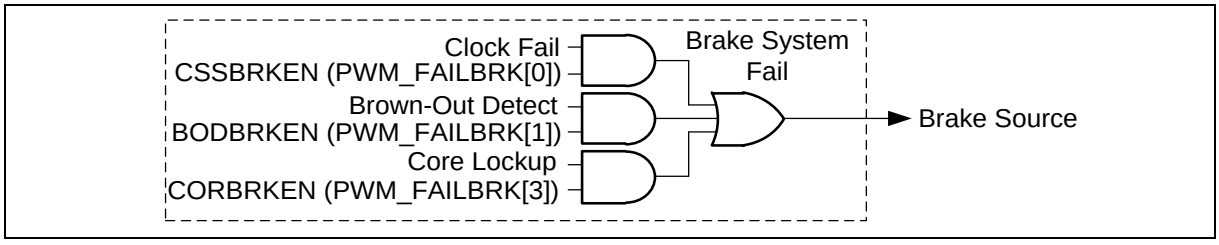


Figure 6.10-29 Brake System Fail Block Diagram

6.10.5.20 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 Negative Polarity Control Register (PWM_POLCTL), for each individual PWM channel. Figure 6.10-30 shows the initial state before PWM starting with different polarity settings.

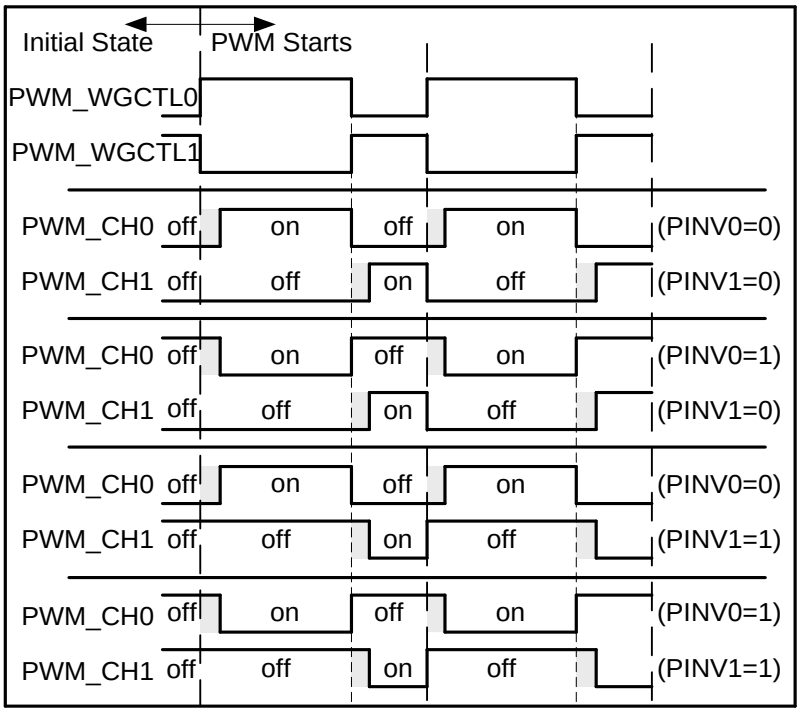


Figure 6.10-30 Initial State and Polarity Control with Rising Edge Dead-Time Insertion

6.10.5.21 Synchronous start function

The synchronous start function can be enabled when SSEN0 (PWM_SSCTL[0]) is set. User can select synchronous source which is from PWM0 by SSRC (PWM_SSCTL[9:8]). The selected PWM channels (include channel0 to channel5 of PWM) will start counting at the same time once the synchronous start function is enabled and set CNTSEN (PWM_SSTRG). It is noted that set CNTSEN (PWM_SSTRG) will also set the counter enable bit (CNTENn, n denotes channel 0 to 5) to start counting.

6.10.5.22 PWM Interrupt Generator.

There are three independent interrupts for each PWM as shown in Figure 6.10-31.

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.

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 CFLIFn (PWM_CAPIF[13:8]) can be triggered when 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 6.10-31 demonstrates the architecture of the PWM interrupts.

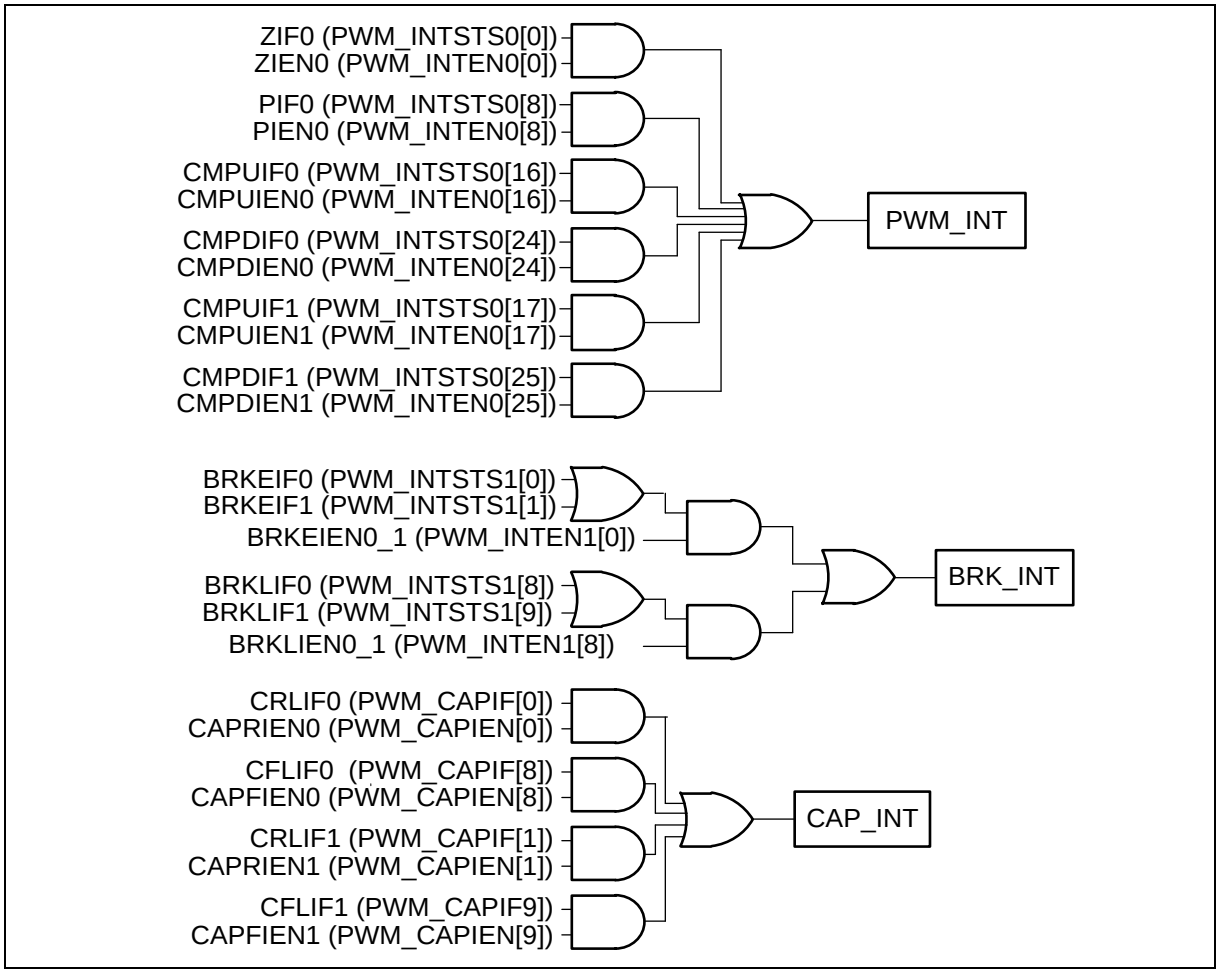


Figure 6.10-31 PWM_CH0 and PWM_CH1 Pair Interrupt Architecture Diagram

6.10.5.23 PWM Trigger ADC Generator

PWM can be one of the ADC conversion trigger source. Each PWM paired 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_ADCTS0[3:0], PWM_ADCTS0[11:8], PWM_ADCTS0[19:16], PWM_ADCTS0[27:24], PWM_ADCTS1[3:0] and PWM_ADCTS1[11:8], respectively. Setting TRGENn is to enable the trigger output to ADC, where TRGENn is TRGEN0, TRGEN1, …, TRGEN5, which are located in PWM_ADCTS0[7], PWM_ADCTS0[15], PWM_ADCTS0[23], PWM_ADCTS0[31], PWM_ADCTS1[7] and PWM_ADCTS1[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 6.10-32 is an example of PWM_CH0 and PWM_CH1. PWM can trigger ADC to start conversion in different timings by setting PERIOD and CMPDAT. Figure 6.10-33 is the trigger ADC timing waveform in the up-down counter type.

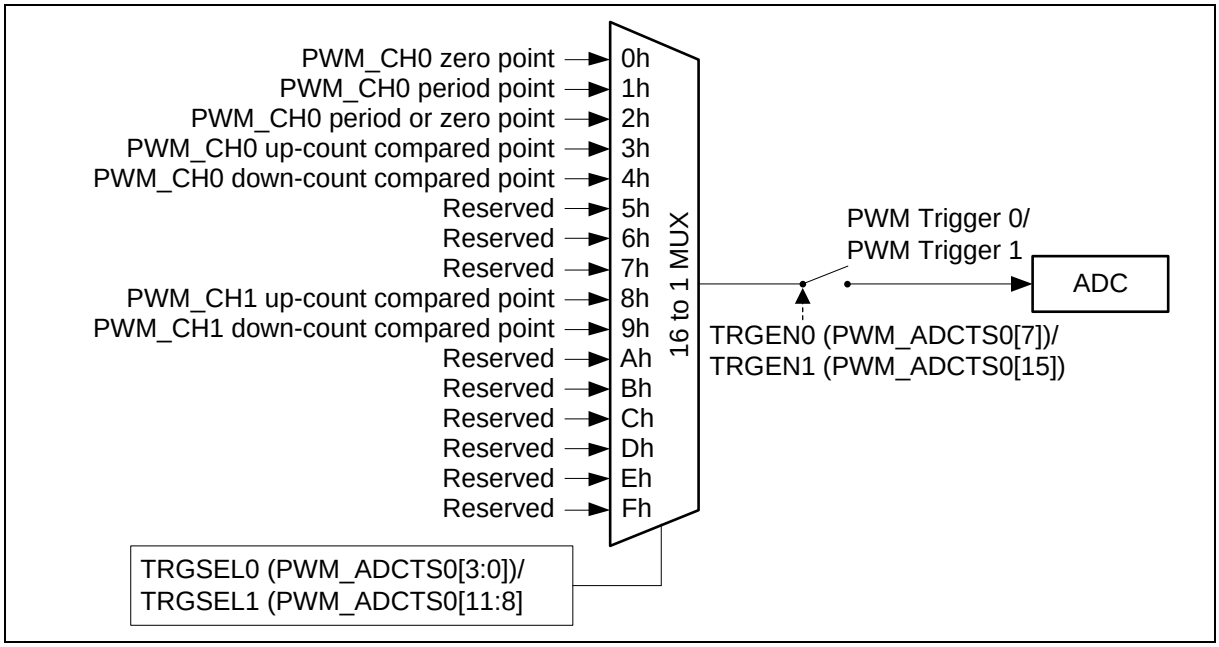


Figure 6.10-32 PWMx_CH0 and PWMx_CH1 Pair Trigger ADC Block Diagram

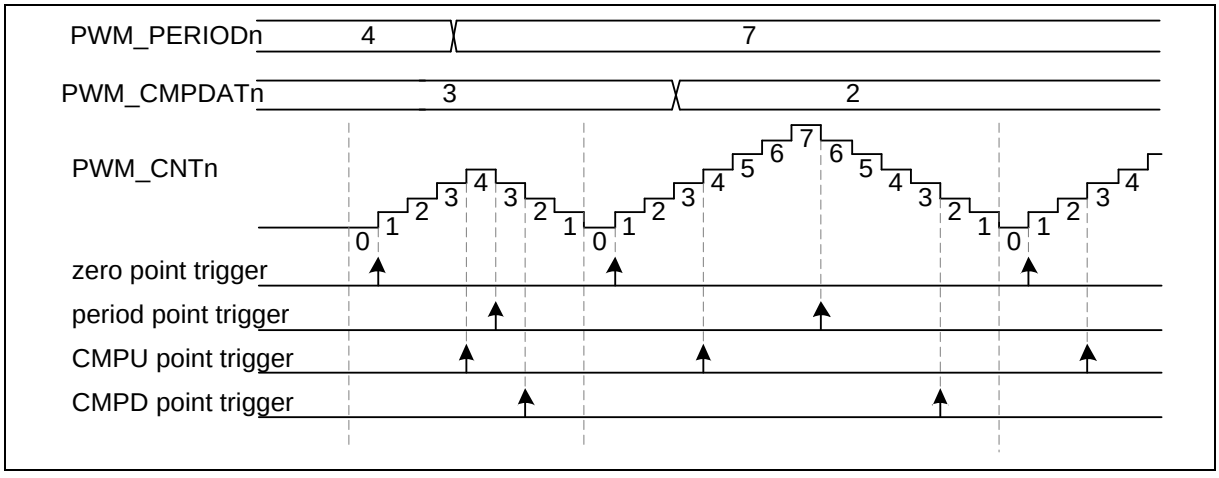


Figure 6.10-33 PWM Trigger ADC in Up-Down Counter Type Timing Waveform

6.10.5.24 Capture Operation

The channels of the capture input and the PWM output share the same pin and counter. The counter can operate in up or down counter type. 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 6.10-34 is the capture block diagram of channel 0.

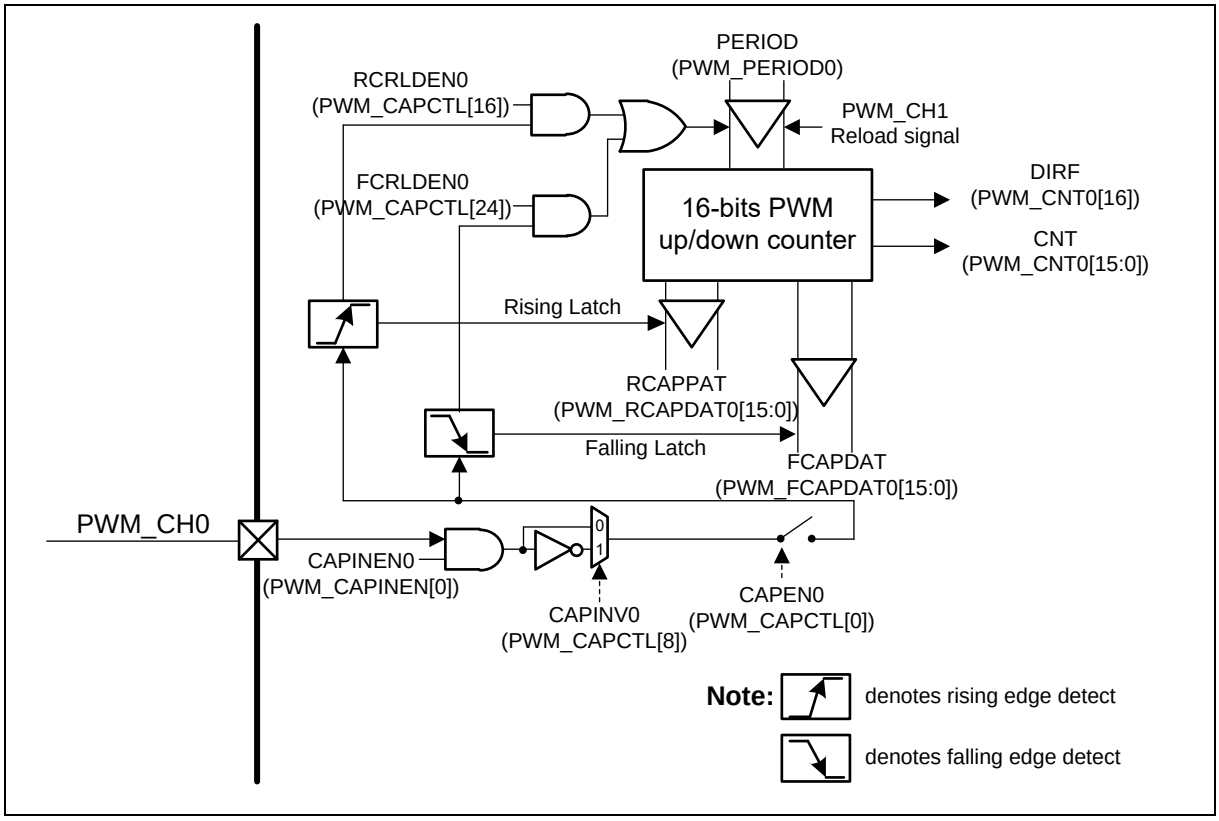


Figure 6.10-34 PWM_CH0 Capture Block Diagram

Figure 6.10-35 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 6.10-35 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 6.10-35 is: if the rising latch happens again when the CRLIFn bit is already set, the Over run 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 Over run status CFLIFOVn (PWM_CAPSTS[13:8]).

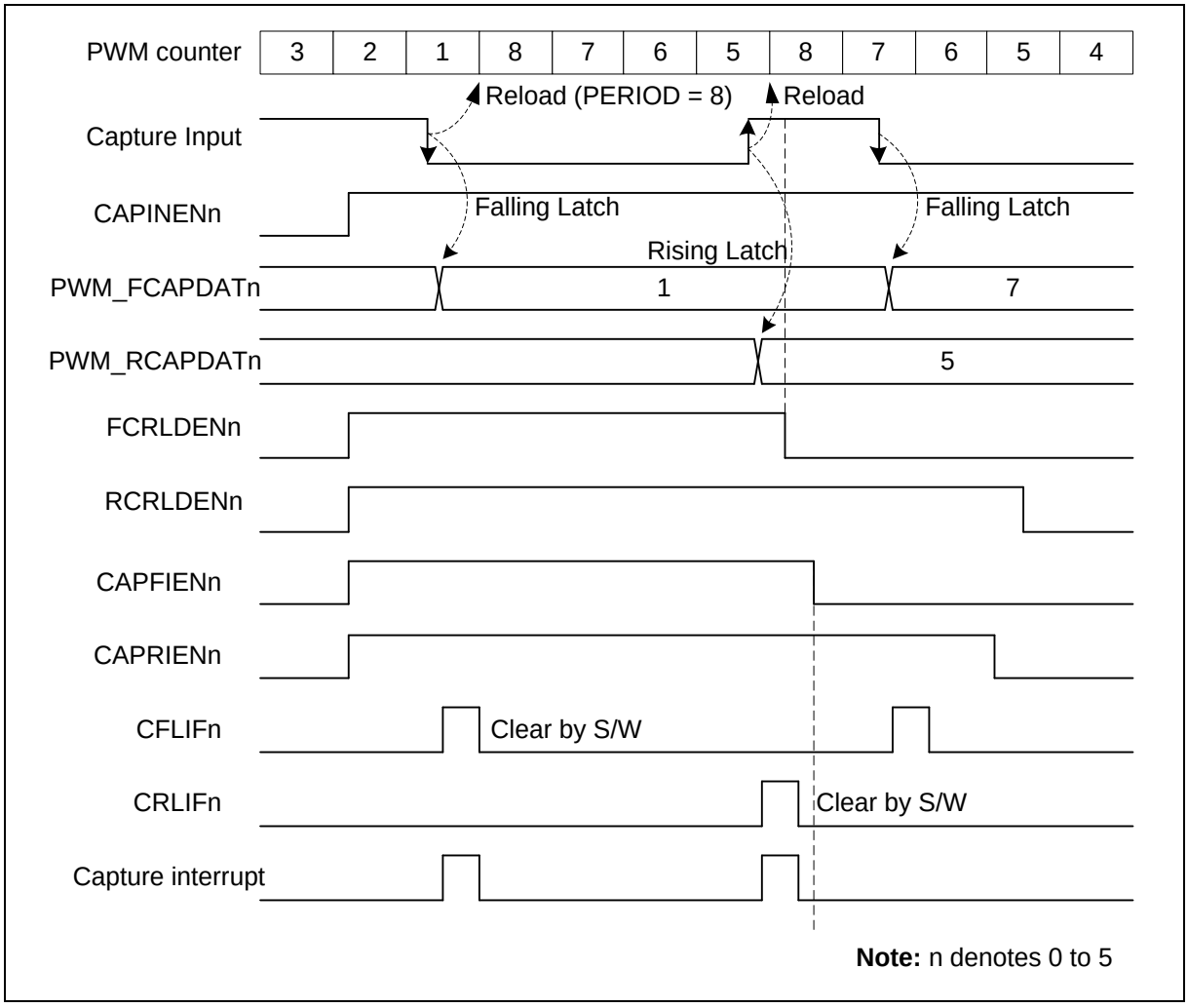


Figure 6.10-35 Capture Operation Waveform

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 PWM_CAPCTL register to 1.

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 6.10-35, 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 6.10-35, the high pulse width is 8+1-7 = 2 PWM counter time.

6.10.5.25 Capture 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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.). 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. 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 6.10-36 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 6.10-36, 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 Figure 6.10-36 denotes the data which are being transferred by PDMA.

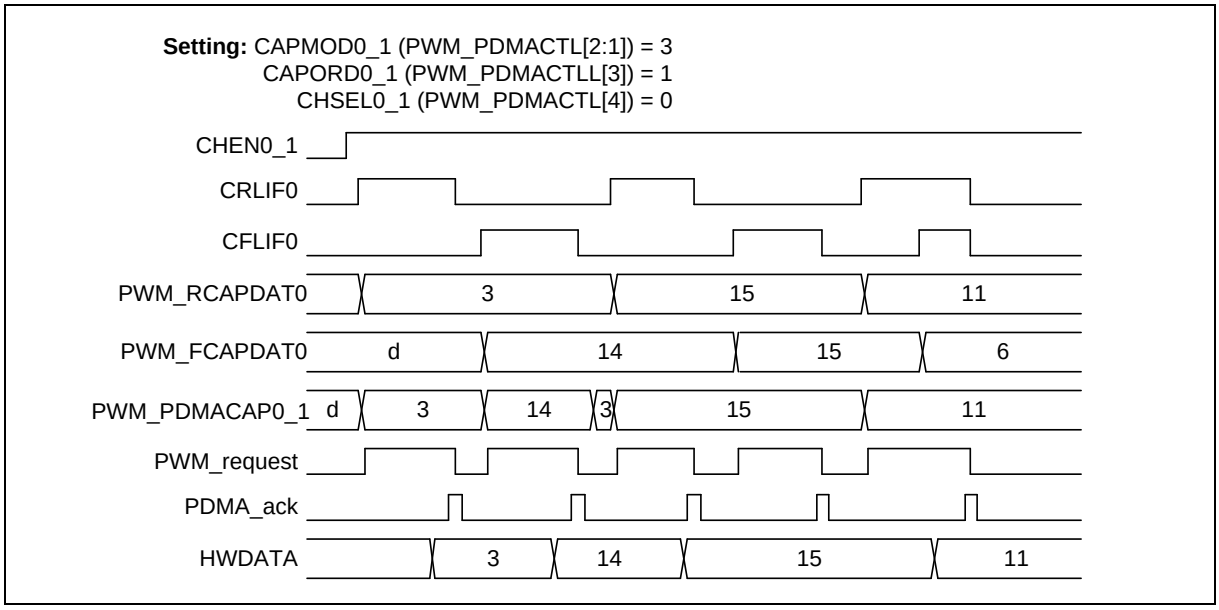


Figure 6.10-36 Capture PDMA Operation Waveform of Channel 0

6.10.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
PWM Base Address: **PWM_BA = 0x4005_8000**				
PWM_CTL0	PWM_BA+0x00	R/W	PWM Control Register 0	0x0000_0000
PWM_CTL1	PWM_BA+0x04	R/W	PWM Control Register 1	0x0000_0000
PWM_CLKSRC	PWM_BA+0x10	R/W	PWM Clock Source Register	0x0000_0000
PWM_CLKPSC0_1	PWM_BA+0x14	R/W	PWM Clock Prescale Register 0/1	0x0000_0000
PWM_CLKPSC2_3	PWM_BA+0x18	R/W	PWM Clock Prescale Register 2/3	0x0000_0000
PWM_CLKPSC4_5	PWM_BA+0x1C	R/W	PWM Clock Prescale Register 4/5	0x0000_0000
PWM_CNTEN	PWM_BA+0x20	R/W	PWM Counter Enable Register	0x0000_0000
PWM_CNTCLR	PWM_BA+0x24	R/W	PWM Clear Counter Register	0x0000_0000
PWM_PERIOD0	PWM_BA+0x30	R/W	PWM Period Register 0	0x0000_0000
PWM_PERIOD2	PWM_BA+0x38	R/W	PWM Period Register 2	0x0000_0000
PWM_PERIOD4	PWM_BA+0x40	R/W	PWM Period Register 4	0x0000_0000
PWM_CMPDAT0	PWM_BA+0x50	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1	PWM_BA+0x54	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2	PWM_BA+0x58	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3	PWM_BA+0x5C	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CMPDAT4	PWM_BA+0x60	R/W	PWM Comparator Register 4	0x0000_0000
PWM_CMPDAT5	PWM_BA+0x64	R/W	PWM Comparator Register 5	0x0000_0000
PWM_DTCTL0_1	PWM_BA+0x70	R/W	PWM Dead-time Control Register 0/1	0x0000_0000
PWM_DTCTL2_3	PWM_BA+0x74	R/W	PWM Dead-time Control Register 2/3	0x0000_0000
PWM_DTCTL4_5	PWM_BA+0x78	R/W	PWM Dead-time Control Register 4/5	0x0000_0000
PWM_CNT0	PWM_BA+0x90	R	PWM Counter Register 0	0x0000_0000
PWM_CNT2	PWM_BA+0x98	R	PWM Counter Register 2	0x0000_0000
PWM_CNT4	PWM_BA+0xA0	R	PWM Counter Register 4	0x0000_0000
PWM_WGCTL0	PWM_BA+0xB0	R/W	PWM Generation Register 0	0x0000_0000
PWM_WGCTL1	PWM_BA+0xB4	R/W	PWM Generation Register 1	0x0000_0000
PWM_MSKEN	PWM_BA+0xB8	R/W	PWM Mask Enable Register	0x0000_0000
PWM_MSK	PWM_BA+0xBC	R/W	PWM Mask Data Register	0x0000_0000

PWM_BNF	PWM_BA+0xC0	R/W	PWM Brake Noise Filter Register	0x0000_0000
PWM_FAILBRK	PWM_BA+0xC4	R/W	PWM System Fail Brake Control Register	0x0000_0000
PWM_BRKCTL0_1	PWM_BA+0xC8	R/W	PWM Brake Edge Detect Control Register 0/1	0x0000_0000
PWM_BRKCTL2_3	PWM_BA+0xCC	R/W	PWM Brake Edge Detect Control Register 2/3	0x0000_0000
PWM_BRKCTL4_5	PWM_BA+0xD0	R/W	PWM Brake Edge Detect Control Register 4/5	0x0000_0000
PWM_POLCTL	PWM_BA+0xD4	R/W	PWM Pin Polar Inverse Register	0x0000_0000
PWM_POEN	PWM_BA+0xD8	R/W	PWM Output Enable Register	0x0000_0000
PWM_SWBRK	PWM_BA+0xDC	W	PWM Software Brake Control Register	0x0000_0000
PWM_INTEN0	PWM_BA+0xE0	R/W	PWM Interrupt Enable Register 0	0x0000_0000
PWM_INTEN1	PWM_BA+0xE4	R/W	PWM Interrupt Enable Register 1	0x0000_0000
PWM_INTSTS0	PWM_BA+0xE8	R/W	PWM Interrupt Flag Register 0	0x0000_0000
PWM_INTSTS1	PWM_BA+0xEC	R/W	PWM Interrupt Flag Register 1	0x0000_0000
PWM_ADCTS0	PWM_BA+0xF8	R/W	PWM Trigger ADC Source Select Register 0	0x0000_0000
PWM_ADCTS1	PWM_BA+0xFC	R/W	PWM Trigger ADC Source Select Register 1	0x0000_0000
PWM_SSCTL	PWM_BA+0x110	R/W	PWM Synchronous Start Control Register	0x0000_0000
PWM_SSTRG	PWM_BA+0x114	W	PWM Synchronous Start Trigger Register	0x0000_0000
PWM_STATUS	PWM_BA+0x120	R/W	PWM Status Register	0x0000_0000
PWM_CAPINEN	PWM_BA+0x200	R/W	PWM Capture Input Enable Register	0x0000_0000
PWM_CAPCTL	PWM_BA+0x204	R/W	PWM Capture Control Register	0x0000_0000
PWM_CAPSTS	PWM_BA+0x208	R	PWM Capture Status Register	0x0000_0000
PWM_RCAPDAT0	PWM_BA+0x20C	R	PWM Rising Capture Data Register 0	0x0000_0000
PWM_FCAPDAT0	PWM_BA+0x210	R	PWM Falling Capture Data Register 0	0x0000_0000
PWM_RCAPDAT1	PWM_BA+0x214	R	PWM Rising Capture Data Register 1	0x0000_0000
PWM_FCAPDAT1	PWM_BA+0x218	R	PWM Falling Capture Data Register 1	0x0000_0000
PWM_RCAPDAT2	PWM_BA+0x21C	R	PWM Rising Capture Data Register 2	0x0000_0000
PWM_FCAPDAT2	PWM_BA+0x220	R	PWM Falling Capture Data Register 2	0x0000_0000
PWM_RCAPDAT3	PWM_BA+0x224	R	PWM Rising Capture Data Register 3	0x0000_0000
PWM_FCAPDAT3	PWM_BA+0x228	R	PWM Falling Capture Data Register 3	0x0000_0000
PWM_RCAPDAT4	PWM_BA+0x22C	R	PWM Rising Capture Data Register 4	0x0000_0000
PWM_FCAPDAT4	PWM_BA+0x230	R	PWM Falling Capture Data Register 4	0x0000_0000
PWM_RCAPDAT5	PWM_BA+0x234	R	PWM Rising Capture Data Register 5	0x0000_0000

PWM_FCAPDAT5	PWM_BA+0x238	R	PWM Falling Capture Data Register 5	0x0000_0000
PWM_PDMACTL	PWM_BA+0x23C	R/W	PWM PDMA Control Register	0x0000_0000
PWM_PDMACAP0_1	PWM_BA+0x240	R	PWM Capture Channel 01 PDMA Register	0x0000_0000
PWM_PDMACAP2_3	PWM_BA+0x244	R	PWM Capture Channel 23 PDMA Register	0x0000_0000
PWM_PDMACAP4_5	PWM_BA+0x248	R	PWM Capture Channel 45 PDMA Register	0x0000_0000
PWM_CAPIEN	PWM_BA+0x250	R/W	PWM Capture Interrupt Enable Register	0x0000_0000
PWM_CAPIF	PWM_BA+0x254	R/W	PWM Capture Interrupt Flag Register	0x0000_0000
PWM_PBUF0	PWM_BA+0x304	R	PWM PERIOD0 Buffer	0x0000_0000
PWM_PBUF2	PWM_BA+0x30C	R	PWM PERIOD2 Buffer	0x0000_0000
PWM_PBUF4	PWM_BA+0x314	R	PWM PERIOD4 Buffer	0x0000_0000
PWM_CMPBUF0	PWM_BA+0x31C	R	PWM CMPDAT0 Buffer	0x0000_0000
PWM_CMPBUF1	PWM_BA+0x320	R	PWM CMPDAT1 Buffer	0x0000_0000
PWM_CMPBUF2	PWM_BA+0x324	R	PWM CMPDAT2 Buffer	0x0000_0000
PWM_CMPBUF3	PWM_BA+0x328	R	PWM CMPDAT3 Buffer	0x0000_0000
PWM_CMPBUF4	PWM_BA+0x32C	R	PWM CMPDAT4 Buffer	0x0000_0000
PWM_CMPBUF5	PWM_BA+0x330	R	PWM CMPDAT5 Buffer	0x0000_0000

6.10.7 Register Description

PWM Control Register 0 (PWM_CTL0)

Register	Offset	R/W	Description	Reset Value
PWM_CTL0	PWM_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	PWM_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	PWM_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	PWM_BA+0x14	R/W	PWM Clock Prescale Register 0/1	0x0000_0000
PWM_CLKPSC2_3	PWM_BA+0x18	R/W	PWM Clock Prescale Register 2/3	0x0000_0000
PWM_CLKPSC4_5	PWM_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	PWM_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	PWM_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	PWM_BA+0x30	R/W	PWM Period Register 0	0x0000_0000
PWM_PERIOD2	PWM_BA+0x38	R/W	PWM Period Register 2	0x0000_0000
PWM_PERIOD4	PWM_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	PWM_BA+0x50	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1	PWM_BA+0x54	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2	PWM_BA+0x58	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3	PWM_BA+0x5C	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CMPDAT4	PWM_BA+0x60	R/W	PWM Comparator Register 4	0x0000_0000
PWM_CMPDAT5	PWM_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 CNTR to generate PWM waveform, interrupt and trigger ADC. 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	PWM_BA+0x70	R/W	PWM Dead-time Control Register 0/1	0x0000_0000
PWM_DTCTL2_3	PWM_BA+0x74	R/W	PWM Dead-time Control Register 2/3	0x0000_0000
PWM_DTCTL4_5	PWM_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 REGWRPROT register.
[23:17]	**Reserved**	Reserved.
[16]	**DTEN**	**Enable Dead-time Insertion for PWM Pair (Write Protect)** **PWM_CH0 andPWM_CH1** **PWM_CH2 andPWM_CH3** **PWM_CH4 andPWM_CH5** 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.
[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	PWM_BA+0x90	R	PWM Counter Register 0	0x0000_0000
PWM_CNT2	PWM_BA+0x98	R	PWM Counter Register 2	0x0000_0000
PWM_CNT4	PWM_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 CNTR 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	PWM_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 or CenterPoint Control** 00 = Do nothing. 01 = PWM period orcente point output Low. 10 = PWM period orcenter point output High. 11 = PWM period orcenter 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	PWM_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	PWM_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	PWM_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	PWM_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**			**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 PWM0 setting: 0 = Brake 1 pin source come from PWM0_BRAKE1. 1 = Reserved
[23:17]	**Reserved**	Reserved.
[16]	**BK0SRC**	**Brake 0 Pin Source Select** For PWM0 setting: 0 = Brake 0 pin source come from PWM0_BRAKE0. 1 = Reserved.
[15]	**BRK1PINV**	**Brake 1 Pin Inverse** 0 = The state of pin PWMx_BRAKE1 is passed to the negative edge detector. 1 = The inversed state of pin PWMx_BRAKE1 is passed to the negative edge detector.
[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**	**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 = The state of pin PWMx_BRAKE0 is passed to the negative edge detector. 1 = The inversed state of pin PWMx_BRAKE10 is passed to the negative edge detector.
[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]	**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**	**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	PWM_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**	**Reserved**	**BODBRKEN**	**CSSBRKEN**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**CORBRKEN**	**Core Lockup Detection Trigger PWM Brake Function 0 Enable Bit** 0 = Brake Function triggered by Core lockup detection Disabled. 1 = Brake Function triggered by Core lockup detection Enabled.
[2]	**Reserved**	Reserved.
[1]	**BODBRKEN**	**Brown-out Detection Trigger PWM Brake Function 0 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 0 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	PWM_BA+0xC8	R/W	PWM Brake Edge Detect Control Register 0/1	0x0000_0000
PWM_BRKCTL2_3	PWM_BA+0xCC	R/W	PWM Brake Edge Detect Control Register 2/3	0x0000_0000
PWM_BRKCTL4_5	PWM_BA+0xD0	R/W	PWM Brake Edge Detect Control Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				BRKAODD		BRKAEVEN	
15	14	13	12	11	10	9	8
SYSLBEN	Reserved	BRKP1LEN	BRKP0LEN	Reserved		CPO1LBEN	CPO0LBEN
7	6	5	4	3	2	1	0
SYSEBEN	Reserved	BRKP1EEN	BRKP0EEN	Reserved		CPO1EBEN	CPO0EBEN

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:18]	**BRKAODD**	**PWM Brake Action Select for Odd Channel (Write Protect)** 00 = PWM odd channel level-detect brake function not affect channel output. 01 = PWM odd channel output tri-state when level-detect brake happened. 10 = PWM odd channel output low level when level-detect brake happened. 11 = PWM odd channel output high level when level-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 brake function not affect channel output. 01 = PWM even channel output tri-state when level-detect brake happened. 10 = PWM even channel output low level when level-detect brake happened. 11 = PWM even channel output high level when level-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]	**Reserved**	Reserved.
[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.
[11:10]	**Reserved**	Reserved.
[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]	**Reserved**	Reserved.
[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.
[3:2]	**Reserved**	Reserved.
[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	PWM_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	PWM_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	PWM_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	PWM_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 always read 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 always read 0 at complementary mode.

PWM Interrupt Enable Register 1 (PWM_INTEN1)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN1	PWM_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	PWM_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	PWM_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 returns 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, 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**	**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 ADC Source Select Register 0 (PWM_ADCTS0)

Register	Offset	R/W	Description	Reset Value
PWM_ADCTS0	PWM_BA+0xF8	R/W	PWM Trigger ADC 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 ADC Enable Bit** 0 = PWM_CH3 Trigger ADC function Disabled. 1 = PWM_CH3 Trigger ADC function Enabled.
[30:28]	**Reserved**	Reserved.
[27:24]	**TRGSEL3**	**PWM_CH3 Trigger ADC 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 ADC Enable Bit** 0 = PWM_CH2 Trigger ADC function Disabled. 1 = PWM_CH2 Trigger ADC function Enabled.
[22:20]	**Reserved**	Reserved.
[19:16]	**TRGSEL2**	**PWM_CH2 Trigger ADC 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 ADC Enable Bit** 0 = PWM_CH1 Trigger ADC function Disabled. 1 = PWM_CH1 Trigger ADC function Enabled.
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL1**	**PWM_CH1 Trigger ADC 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 ADC Enable Bit** 0 = PWM_CH0 Trigger ADC function Disabled. 1 = PWM_CH0 Trigger ADC function Enabled.
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL0**	**PWM_CH0 Trigger ADC 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 ADC Source Select Register 1 (PWM_ADCTS1)

Register	Offset	R/W	Description	Reset Value
PWM_ADCTS1	PWM_BA+0xFC	R/W	PWM Trigger ADC 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 ADC Enable Bit** 0 = PWM_CH5 Trigger ADC function Disabled. 1 = PWM_CH5 Trigger ADC function Enabled.
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL5**	**PWM_CH5 Trigger ADC 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 ADC Enable Bit** 0 = PWM_CH4 Trigger ADC function Disabled. 1 = PWM_CH4 Trigger ADC function Enabled.
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL4**	**PWM_CH4 Trigger ADC 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	PWM_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 PWM0. 01 = Reserved. 10 = Reserved. 11 = Reserved.
[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	PWM_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) 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 PWM channel counter synchronous start function is enabled.

PWM Status Register (PWM_STATUS)

Register	Offset	R/W	Description	Reset Value
PWM_STATUS	PWM_BA+0x120	R/W	PWM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**ADCTRG5**	**ADCTRG4**	**ADCTRG3**	**ADCTRG2**	**ADCTRG1**	**ADCTRG0**
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	
[31:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**ADCTRGn**	**ADC Start of Conversion Status** 0 = Indicates no ADC start of conversion trigger event has occurred. 1 = An ADC 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 Capture Input Enable Register (PWM_CAPINEN)

Register	Offset	R/W	Description	Reset Value
PWM_CAPINEN	PWM_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	PWM_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	PWM_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	PWM_BA+0x20C	R	PWM Rising Capture Data Register 0	0x0000_0000
PWM_RCAPDAT1	PWM_BA+0x214	R	PWM Rising Capture Data Register 1	0x0000_0000
PWM_RCAPDAT2	PWM_BA+0x21C	R	PWM Rising Capture Data Register 2	0x0000_0000
PWM_RCAPDAT3	PWM_BA+0x224	R	PWM Rising Capture Data Register 3	0x0000_0000
PWM_RCAPDAT4	PWM_BA+0x22C	R	PWM Rising Capture Data Register 4	0x0000_0000
PWM_RCAPDAT5	PWM_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	PWM_BA+0x210	R	PWM Falling Capture Data Register 0	0x0000_0000
PWM_FCAPDAT1	PWM_BA+0x218	R	PWM Falling Capture Data Register 1	0x0000_0000
PWM_FCAPDAT2	PWM_BA+0x220	R	PWM Falling Capture Data Register 2	0x0000_0000
PWM_FCAPDAT3	PWM_BA+0x228	R	PWM Falling Capture Data Register 3	0x0000_0000
PWM_FCAPDAT4	PWM_BA+0x230	R	PWM Falling Capture Data Register 4	0x0000_0000
PWM_FCAPDAT5	PWM_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	PWM_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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.
[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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.

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	PWM_BA+0x240	R	PWM Capture Channel 01 PDMA Register	0x0000_0000
PWM_PDMACAP2_3	PWM_BA+0x244	R	PWM Capture Channel 23 PDMA Register	0x0000_0000
PWM_PDMACAP4_5	PWM_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. Please refer to section 3.2 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.

PWM Capture Interrupt Enable Register (PWM_CAPIEN)

Register	Offset	R/W	Description	Reset Value
PWM_CAPIEN	PWM_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	PWM_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 Period Register Buffer 0, 2, 4 (PWM_PBUF0, 2, 4)

Register	Offset	R/W	Description	Reset Value
PWM_PBUF0	PWM_BA+0x304	R	PWM PERIOD0 Buffer	0x0000_0000
PWM_PBUF2	PWM_BA+0x30C	R	PWM PERIOD2 Buffer	0x0000_0000
PWM_PBUF4	PWM_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	PWM_BA+0x31C	R	PWM CMPDAT0 Buffer	0x0000_0000
PWM_CMPBUF1	PWM_BA+0x320	R	PWM CMPDAT1 Buffer	0x0000_0000
PWM_CMPBUF2	PWM_BA+0x324	R	PWM CMPDAT2 Buffer	0x0000_0000
PWM_CMPBUF3	PWM_BA+0x328	R	PWM CMPDAT3 Buffer	0x0000_0000
PWM_CMPBUF4	PWM_BA+0x32C	R	PWM CMPDAT4 Buffer	0x0000_0000
PWM_CMPBUF5	PWM_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.

6.11 UART Interface Controller (UART)

6.11.1 Overview

The chip provides two 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 LIN, IrDA SIR, RS-485, and Single-wire function modes and auto-baud rate measuring function.

6.11.2 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
  - Supports 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
  - Supports for 3/16 bit duration for normal mode
- Supports LIN function mode
  - Supports LIN master/slave mode
  - Supports programmable break generation function for transmitter
  - Supports break detection function for receiver
  - Supports LIN slave header time-out detection function
  - Supports LIN response time-out detection function
  - Supports LIN wake-up function
- Supports RS-485 function mode
  - Supports RS-485 9-bit mode
  - Supports hardware or software enables to program nRTS pin to control RS-485

transmission direction

- Supports PDMA transfer function
- Supports Single-wire function mode.

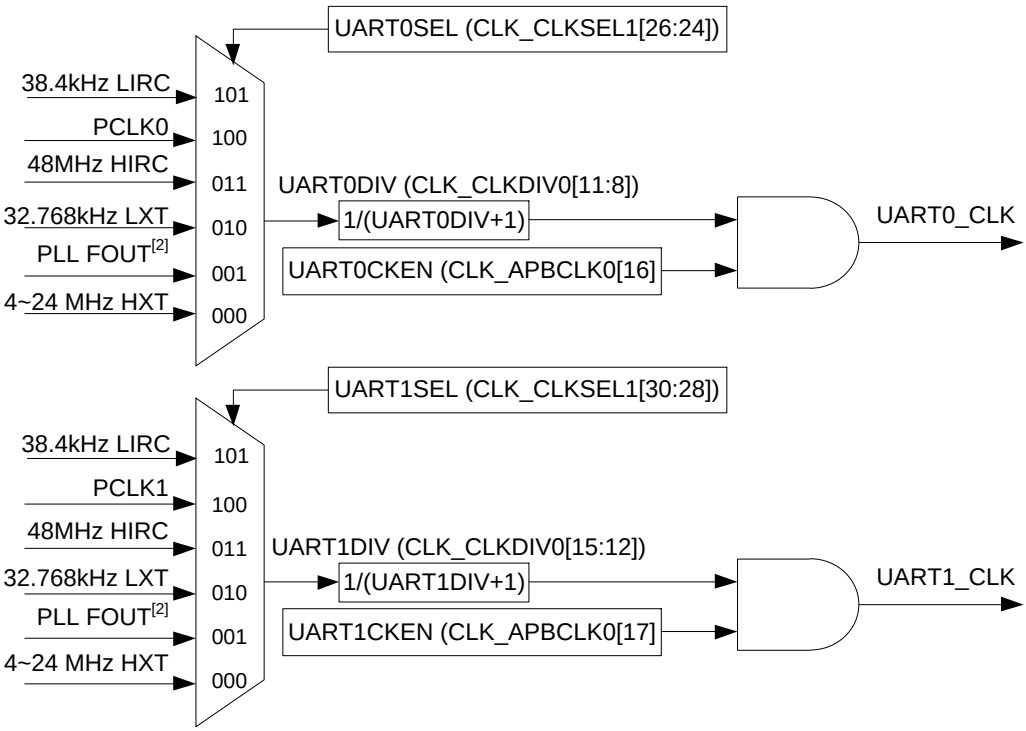
UART Feature	UART0/ UART1	USCI-UART
FIFO	16 Bytes	TX: 1 Byte RX: 2 Bytes
Auto Flow Control (CTS/RTS)	√	√
IrDA	√	-
LIN	√	-
RS-485 Function Mode	√	√
nCTS Wake-up	√	√
Imcoming Data Wake-up	√	√
Received Data FIFO reached threshold Wake-up	√	-
RS-485 Address Match (AAD mode) Wake-up	√	-
Auto-Baud Rate Measurement	√	√
STOP Bit Length	1, 1.5, 2 bit	1, 2 bit
Word Length	5, 6, 7, 8 bits	6~13 bits
Even / Odd Parity	√	√
Stick Bit	√	-
Note: √= Supported		

Table 6.11-1 NuMicro® M0A21/M0A23 Series UART Features

6.11.3 Block Diagram

The UART clock control and block diagram are shown in Figure 6.11-1 and Figure 6.11-2 respectively.

Note: The frequency of UARTx_CLK should not be greater than 30 times HCLK.



Note1: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.
Note2: Without PLL function, the clock source will be fixed at PCLK0/1, the same as UARTxSEL=3'b100. Please refer to section 4.3 NuMicro® M0A21/M0A23 Series Selection Guide for detailed information.

Figure 6.11-1 UART Clock Control Diagram

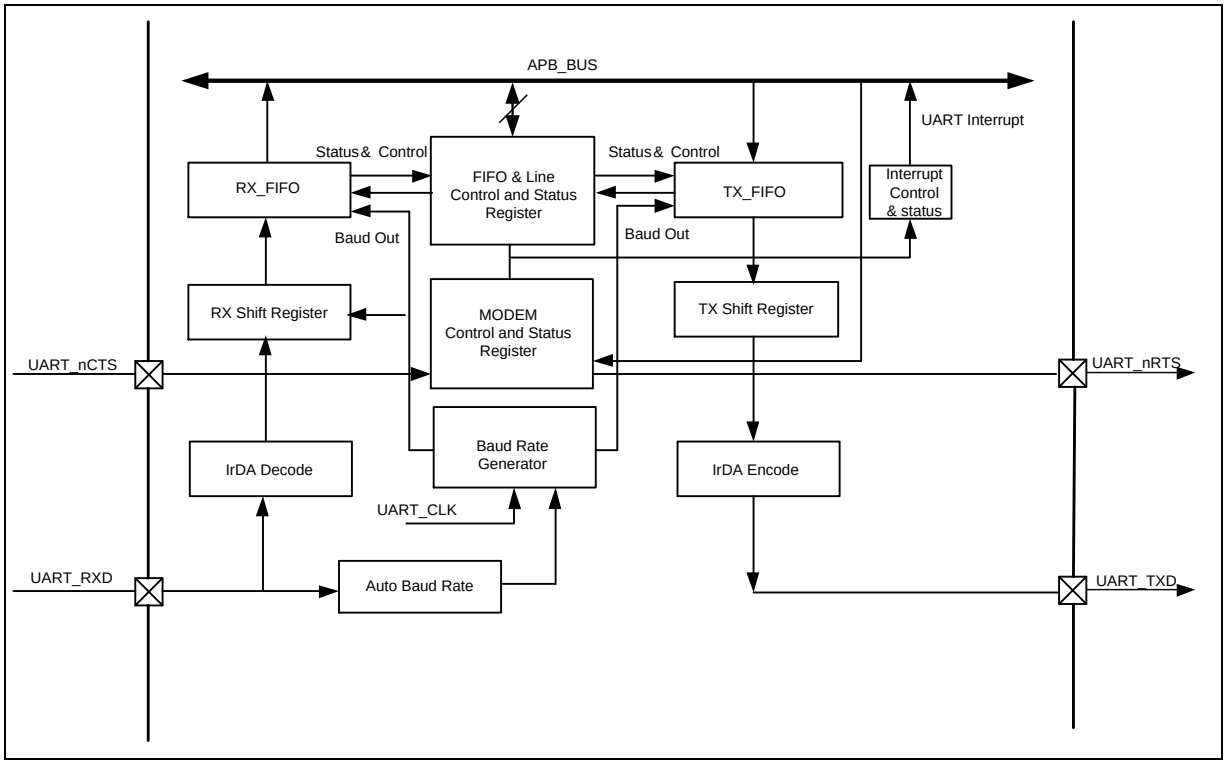


Figure 6.11-2 UART Block Diagram

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.
THERINT	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 6.11-2 UART Interrupt

6.11.4 Basic Configuration

The basic configurations of UART0 are as follows:

- Clock Source Configuration
  - Select the source of UART0 peripheral clock on UART0SEL (CLK_CLKSEL1[26:24]).
  - 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]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
UART0	UART0_RXD	PD.1, PD.5	MFP3
		PA.0, PA.1, PA.2, PA.3, PA.4, PA.5 PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	MFP9
	UART0_TXD	PD.0, PD.4	MFP3
		PA.0, PA.1, PA.2, PA.4, PA.5	MFP8

		PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	
	UART0_nCTS	PA.0, PA.1, PA.2, PA.3 PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	MFP3
	UART0_nRTS	PA.2, PA.4, PA.5 PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	MFP2

The basic configurations of UART1 are as follows:

- Clock Source Configuration
  - Select the source of UART1 peripheral clock on UART1SEL (CLK_CLKSEL1[30:28]).
  - 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]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
UART1	UART1_RXD	PA.0, PA.1, PA.2, PA.3, PA.4, PA.5 PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	MFP29
	UART1_TXD	PA.0, PA.1, PA.2, PA.4, PA.5 PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	MFP28
	UART1_nCTS	PD.2, PD.6	MFP6
		PB.4	MFP30
	UART1_nRTS	PD.3, PD.7	MFP6
		PC.2	MFP30

UART Interface Controller Pin description is shown in Table 6.11-3:

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 6.11-3 UART Interface Controller Pin

6.11.5 Functional Description

The UART controller supports five function modes including UART, LIN, IrDA, 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 the following section.

6.11.5.1 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 6.11-4 lists the UART baud rate equations in the various conditions. Table 6.11-5 and Table 6.11-6 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 3*HCLK < UART_CLK =< 4*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 6.11-4 UART Controller Baud Rate Equation Table

UART Peripheral Clock = 12 MHz			
Baud Rate	Mode 0	Mode 1	Mode 2
921600	Not supported	Not recommended	BRD=11
460800	Not recommended	BRD=0, EDIVM1 =13	BRD=24
230400	Not recommended	BRD =2, EDIVM1 =13	BRD =50
115200	Not recommended	BRD =6, EDIVM1 =13	BRD =102
57600	BRD =11	BRD =14, EDIVM1 =13	BRD =206
38400	BRD =18	BRD =22, EDIVM1 =13	BRD =311
19200	BRD =37	BRD =123, EDIVM1 =5	BRD =623
9600	BRD =76	BRD =123, EDIVM1 =10	BRD =1248
4800	BRD =154	BRD =248, EDIVM1 =10	BRD =2498

Table 6.11-5 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 supported	Not recommended	0x3000_000B

460800	Not recommended	0x2D00_0000	0x3000_0018
230400	Not recommended	0x2D00_0002	0x3000_0032
115200	Not recommended	0x2D00_0006	0x3000_0066
57600	0x0000_000B	0x2D00_000E	0x3000_00CE
38400	0x0000_0012	0x2D00_0016	0x3000_0137
19200	0x0000_0025	0x2500_007B	0x3000_026F
9600	0x0000_004C	0x2A00_007B	0x3000_04E0
4800	0x0000_009A	0x2A00_00F8	0x3000_09C2

Table 6.11-6 UART Controller Baud Rate Register Setting Example Table

6.11.5.2 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 BRCOMPDEC (UART_BRCOMP[31]) to define the positive or negative compensation in each bit. If the BRCOMPDEC (UART_BRCOMP[31]) = 0, it is positive compensation for each bit, one more module clock will be appended 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-bit 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

Baud rate is 9600, UART peripheral clock is 32.768 kHz → 3.413 peripheral clock/bit

if the baud divider is set as 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 6.11-7 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 as 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 6.11-8 Baud Rate Compensation Example Table 2

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.

The $2^{ABRDBITS}$ bit time from Start bit to the 1st rising edge is calculated by setting ABRDBITS (UART_ALTCTL[20:19]). Setting ABRDEN (UART_ALTCTL[18]) is to enable auto-baud rate function. In beginning stage, the UART RX is kept at 1. Once falling edge is detected, START bit is received. The auto-baud rate counter is reset and starts counting. The auto-baud rate counter will be stop when the 1^{st} rising edge is detected. Then, 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 is shown in Figure 6.11-3. 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]) is 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.

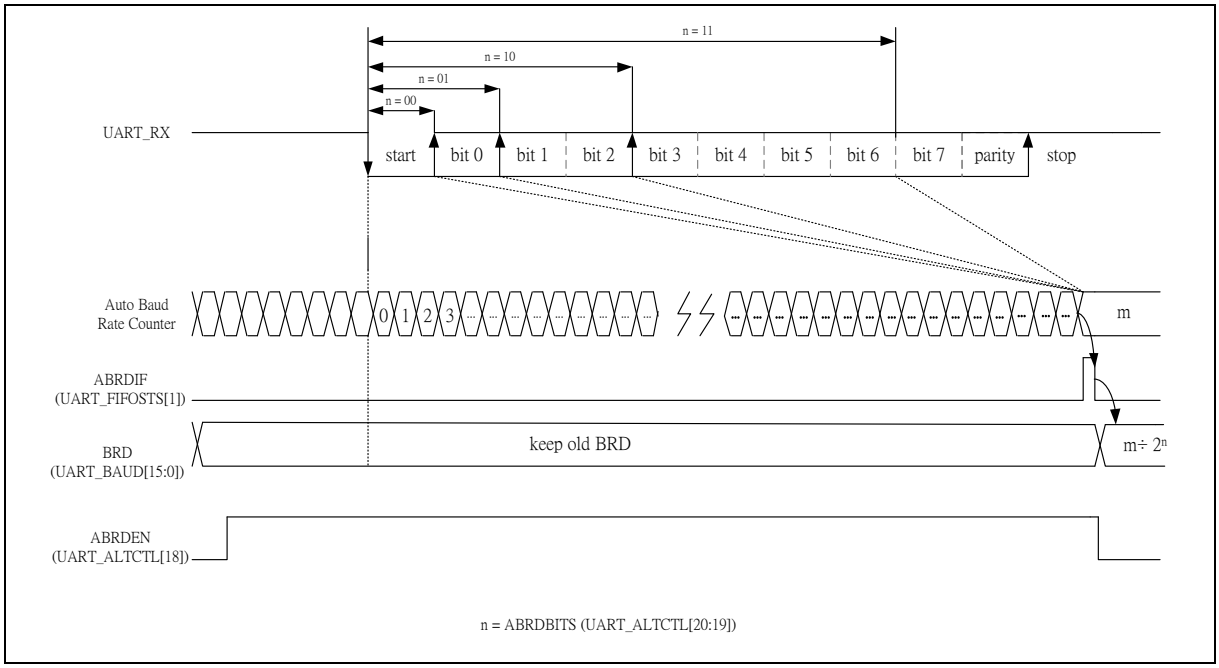


Figure 6.11-3 Auto-Baud Rate Measurement

6.11.5.3 Programming Sequence Example

1. Program ABRDBITS (UART_ALTCTL[20:19]) to determines 2 ^{ABRDBITS} bit time from UART RX receive START bit falling edge to data 1st rising edge.
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.

6.11.5.4 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 6.11-4.

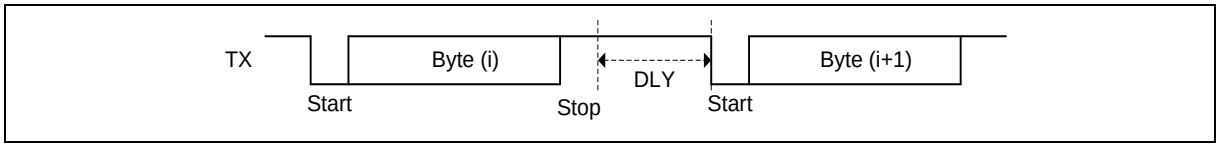


Figure 6.11-4 Transmit Delay Time Operation

6.11.5.5 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.

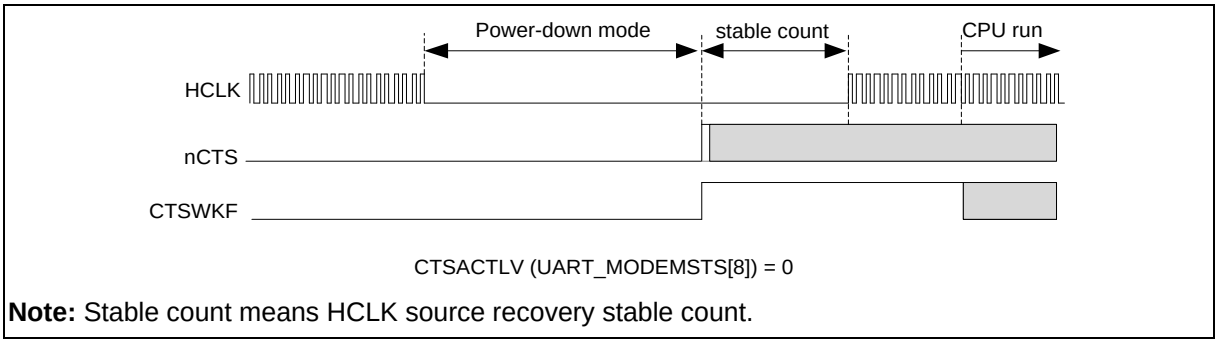
6.11.5.6 UART Controller Wake-up Function

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]) is generated. If the WKIEN (UART_INTEN[6]) 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 6.11-5 and Figure 6.11-6.

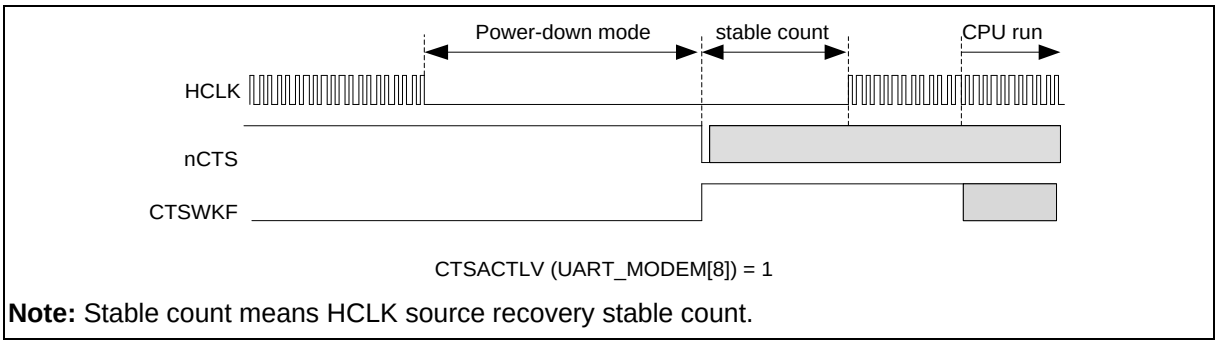
nCTS Wake-up Case 1 (nCTS transition from low to high)



Note: Stable count means HCLK source recovery stable count.

Figure 6.11-5 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 6.11-6 UART nCTS Wake-up Case2

Incoming Data Wake-up

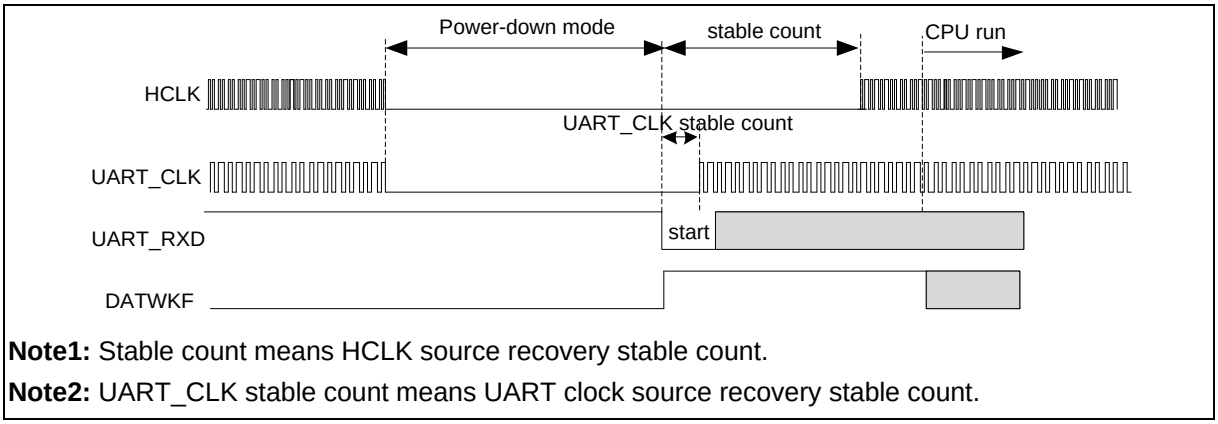
When system is in Power-down mode and the WKDATEN (UART_WKCTL[1]) is set, the toggle of incoming data (UART_RXD) pin can wake-up the system. In order to receive the incoming data after the system wake-up, the STCOMP (UART_DWKCOMP[15:0]) shall be set. These bits field of STCOMP indicate how many clock 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]) is generated. The incoming data wake-up is shown in Figure 6.11-7.

Note1: The UART controller clock source should be selected as HIRC and the compensation time for

start bit is about 91.129us. It means that the value of STCOMP (UART_DWKCOMP[15:0]) can be set as 0x1116.

Note2: The value of BRD (UART_BAUD[15:0]) should be greater than STCOMP (UART_DWKCOMP[15:0]).



Note1: Stable count means HCLK source recovery stable count.

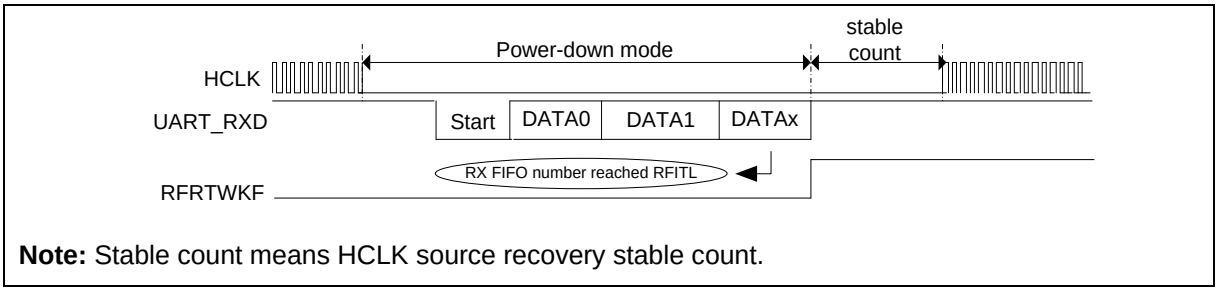
Note2: UART_CLK stable count means UART clock source recovery stable count.

Figure 6.11-7 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]) is generated. The Received Data FIFO reached threshold wake-up is shown in Figure 6.11-8.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



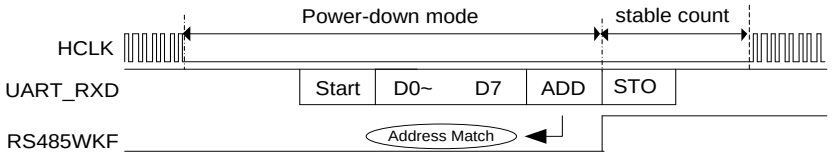
Note: Stable count means HCLK source recovery stable count.

Figure 6.11-8 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]) that cause the RS485 address match (AAD mode) wake-up flag RS485WKF (UART_WKSTS[3]) is generated. The RS-485 Address Match (AAD mode) wake-up is shown in Figure 6.11-9.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



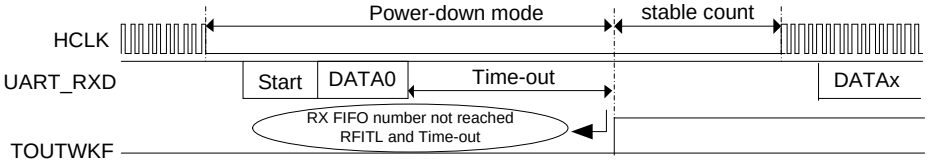
Note: Stable count means HCLK source recovery stable count.

Figure 6.11-9 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]) that cause the Received Data FIFO threshold time-out wake-up wake-up flag TOUTWKF (UART_WKSTS[4]) is generated. The Received Data FIFO threshold time-out wake-up is shown in Figure 6.11-10.

Note: The UART controller clock source should be selected as LXT or LIRC in Power-down mode to receive data.



Note: Stable count means HCLK source recovery stable count.

Figure 6.11-10 UART Received Data FIFO threshold time-out wake-up

6.11.5.7 UART Controller Interrupt and Status

Each UART controller supports eleven types of interrupts including:

- Receive Data Available Interrupt (RDAINT)
- Transmit Holding Register Empty Interrupt (THERINT)
- 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)
  - LIN Slave Header Time-out Detect (SLVHTOF)
  - LIN Response Time-out Detect (RTOUTF)
- 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)
  - LIN Wake-up Flag (LINWKF)
- 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 6.11-9 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	THERINT	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-	RXTOINT	RXTOIEN	RXTOIF	N/A	Read UART_DAT

out Interrupt					
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
				LINIF = SLVHTOF	Write ‘1’ to LINIF and Write ‘1’ to SLVHTOF
				LINIF = RTOUTF	Write ‘1’ to LINIF and Write ‘1’ to RTOUTF
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
				WKIF = LINWKF	Write ‘1’ to LINWKF
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 6.11-9 UART controller Interrupt Source and Flag List

6.11.5.8 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 1 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 6.11-10 and Table 6.11-11 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 6.11-10 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 6.11-11 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 6.11-11.

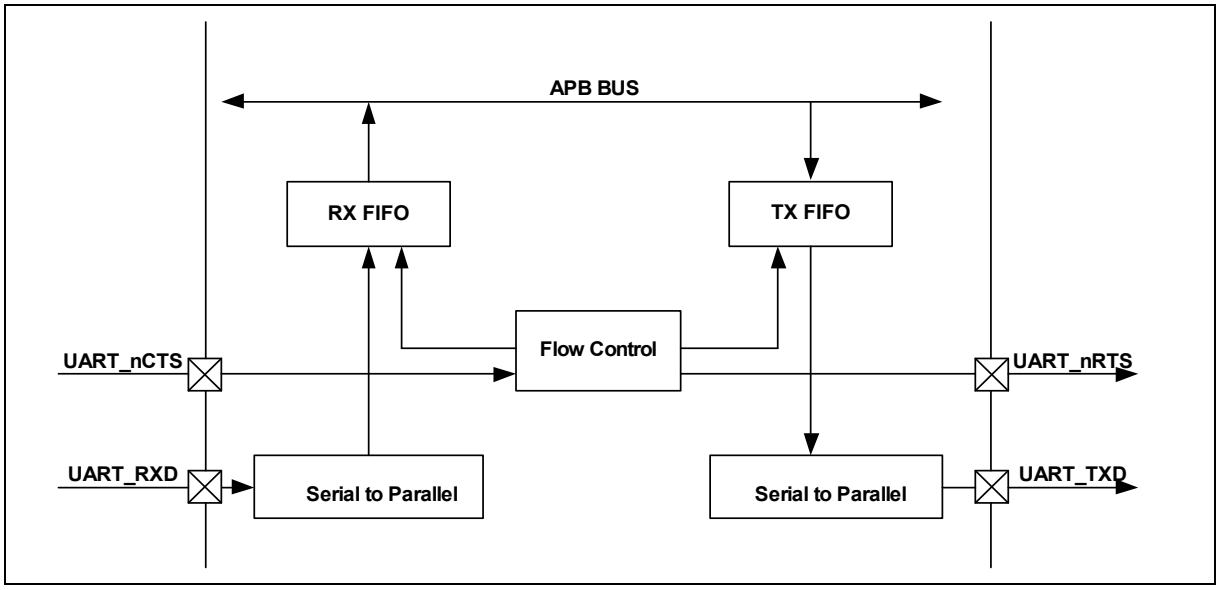


Figure 6.11-11 Auto-Flow Control Block Diagram

Figure 6.11-12 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.

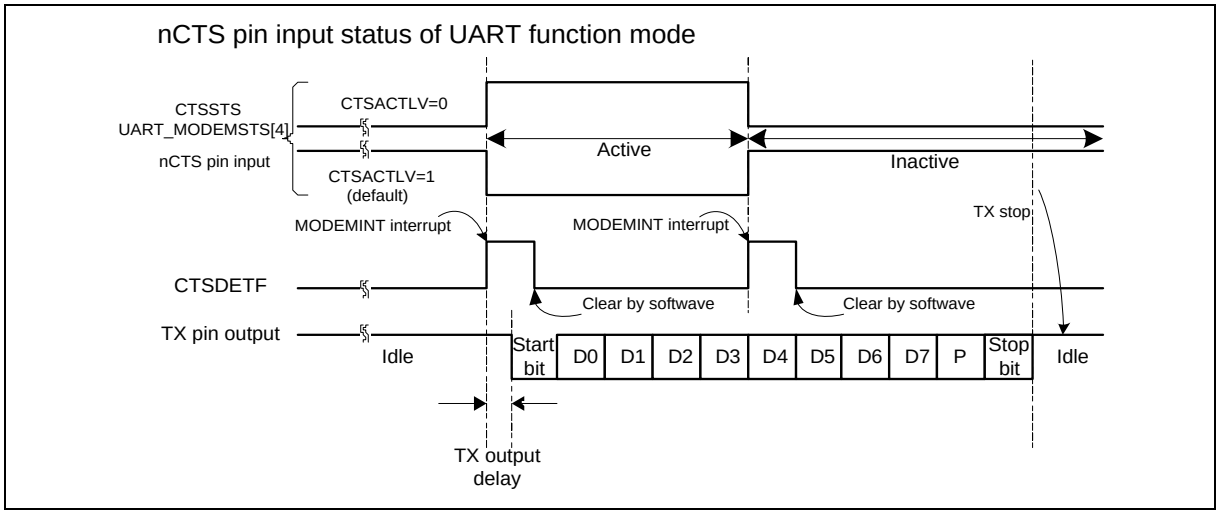


Figure 6.11-12 UART nCTS Auto-Flow Control Enabled

As shown in Figure 6.11-13, 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.

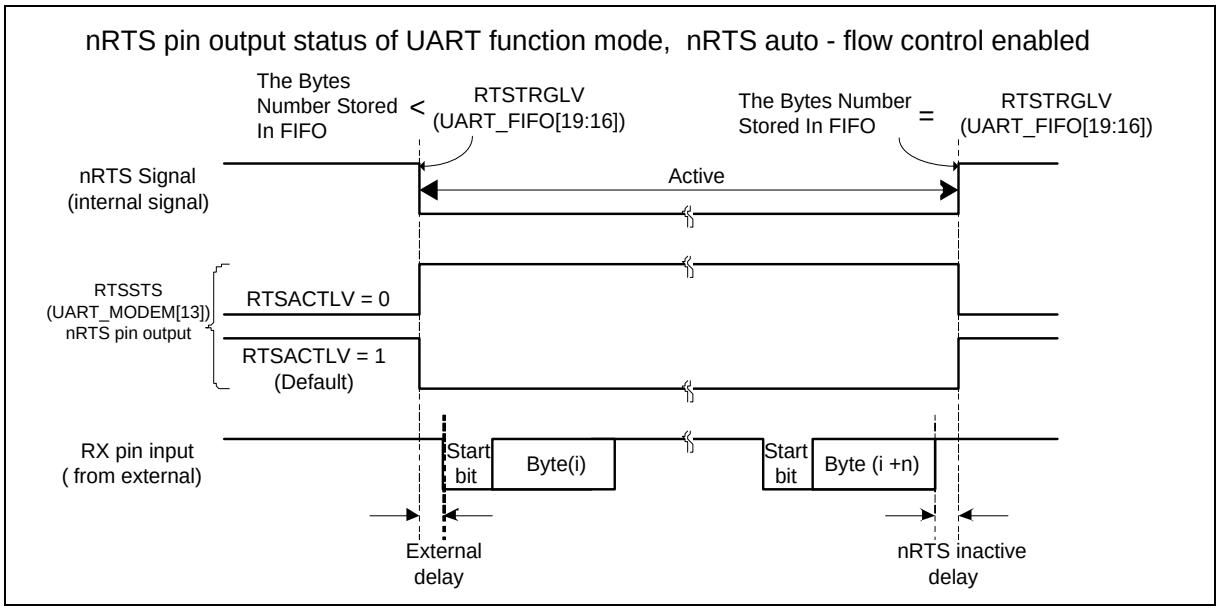


Figure 6.11-13 UART nRTS Auto-Flow Control Enabled

As shown in Figure 6.11-14, 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.

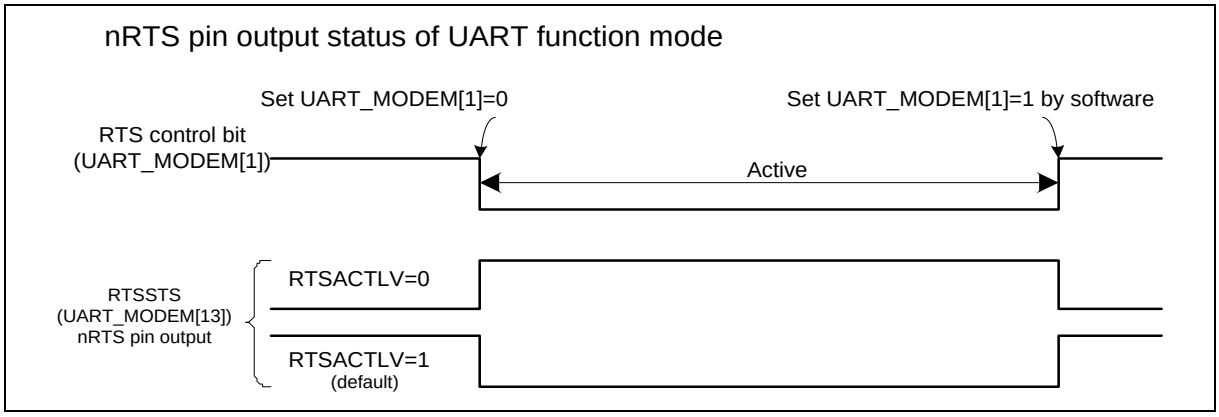


Figure 6.11-14 UART nRTS Auto-Flow with Software Control

6.11.5.9 IrDA Function Mode

The UART controller also provides Serial IrDA (SIR, Serial Infrared) function (Setting 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 6.11-15.

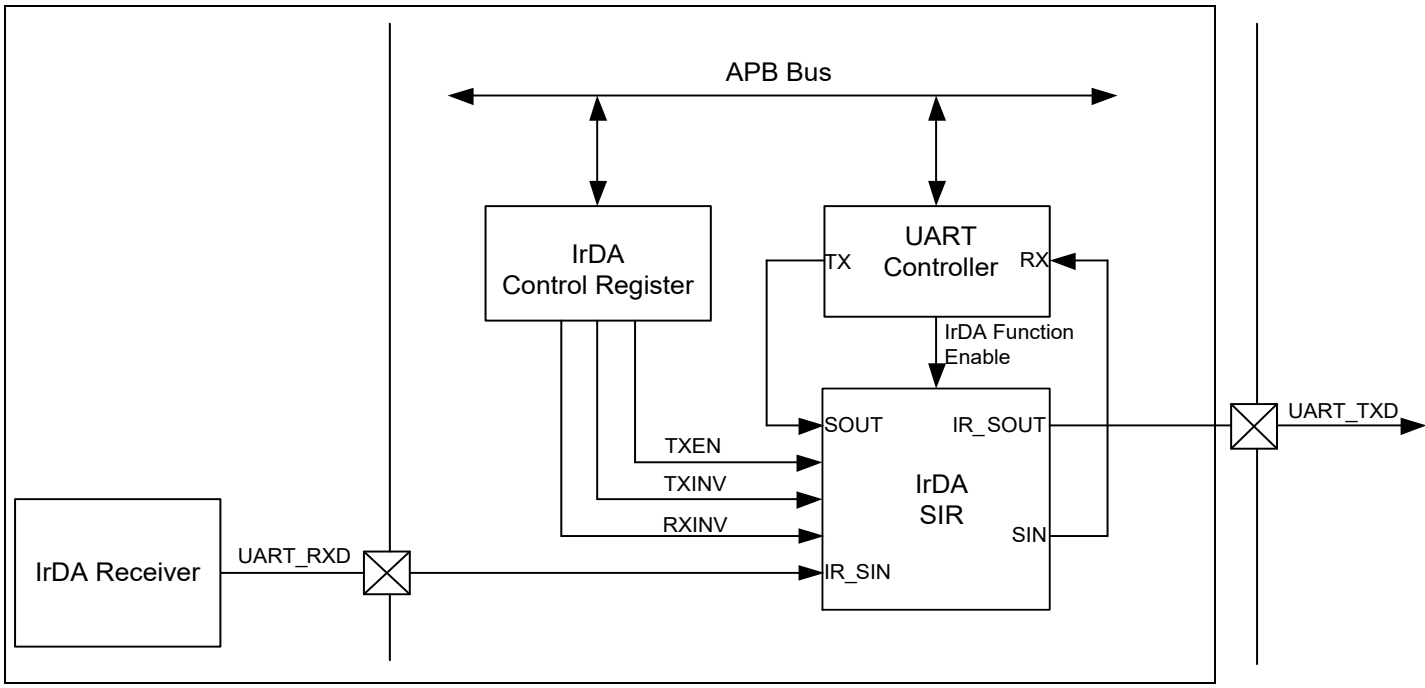


Figure 6.11-15 IrDA Control Block Diagram

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 6.11-16 is IrDA encoder/decoder waveform.

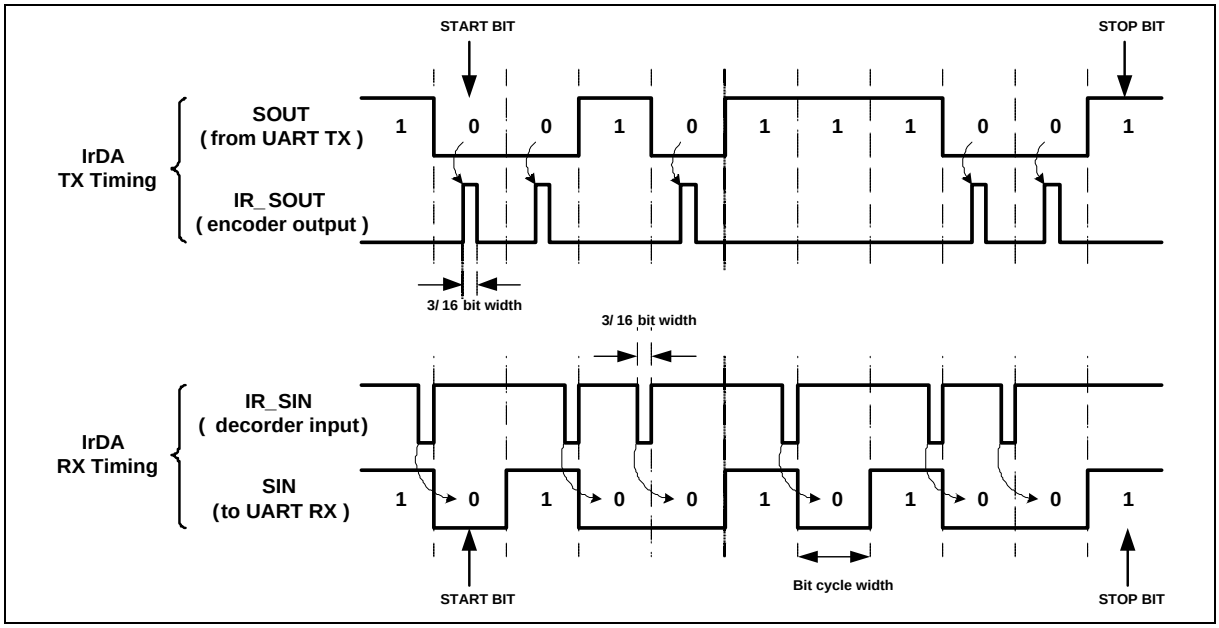


Figure 6.11-16 IrDA TX/RX Timing Diagram

6.11.5.10 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 6.11-17 is the structure of LIN Frame.

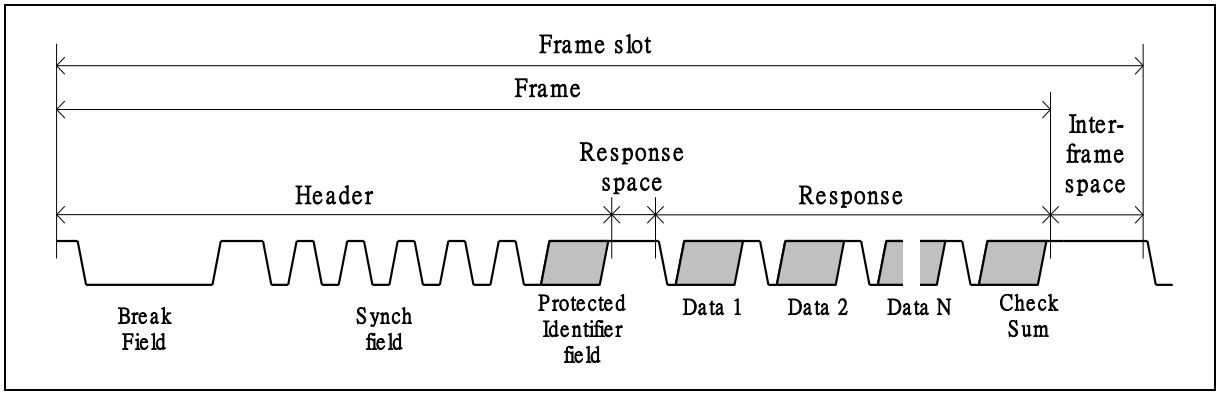


Figure 6.11-17 Structure of LIN Frame

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 6.11-18.

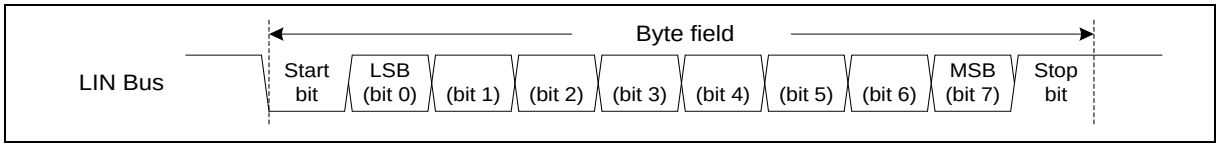


Figure 6.11-18 Structure of LIN Byte

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 on 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 6.11-12 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.

Note1: 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.

Note2: 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.

Note3: 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 6.11-19.

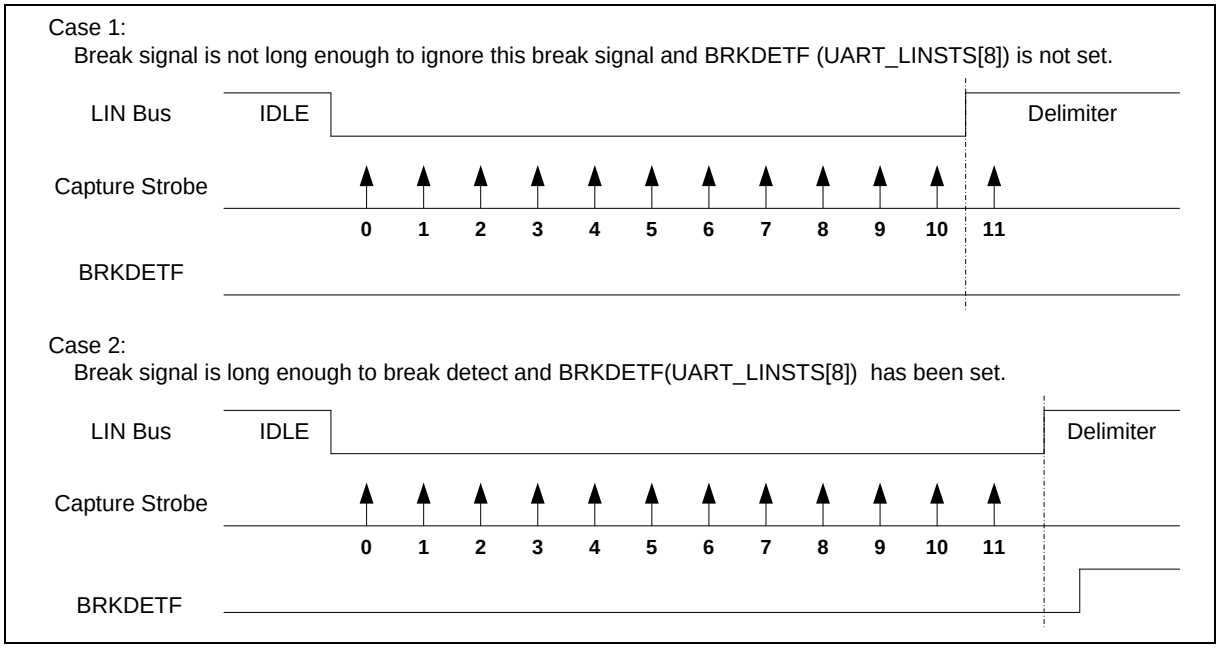


Figure 6.11-19 Break Detection in LIN Mode

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.

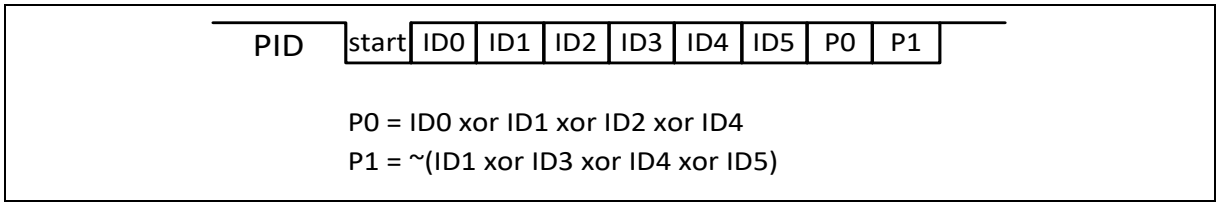


Figure 6.11-20 LIN Frame ID and Parity Format

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]) and the header time-out flag SLVHTOF (UART_LINSTS[4]) 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 UART_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.

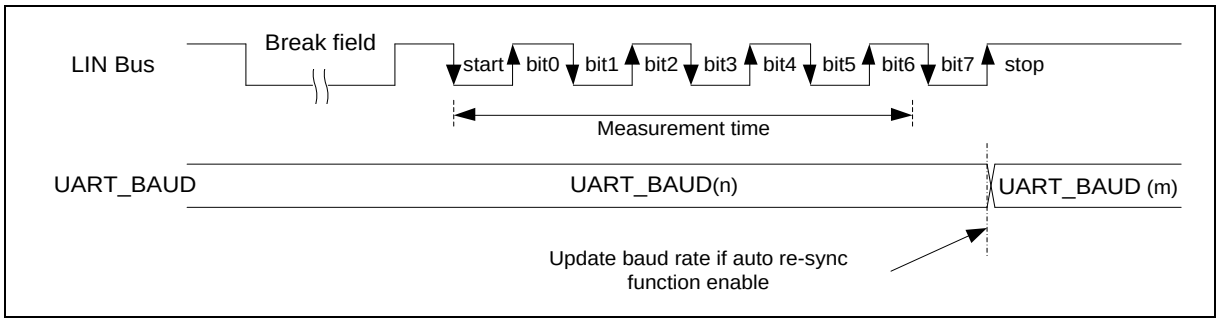


Figure 6.11-21 LIN Sync Field Measurement

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 Figure 6.11-22.

Note1: It is recommended to set the SLVDUEN bit before every checksum reception.

Note2: 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.

Note3: When operating in Automatic Resynchronization mode, the baud rate setting must be operated at mode2 (BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) must be 1).

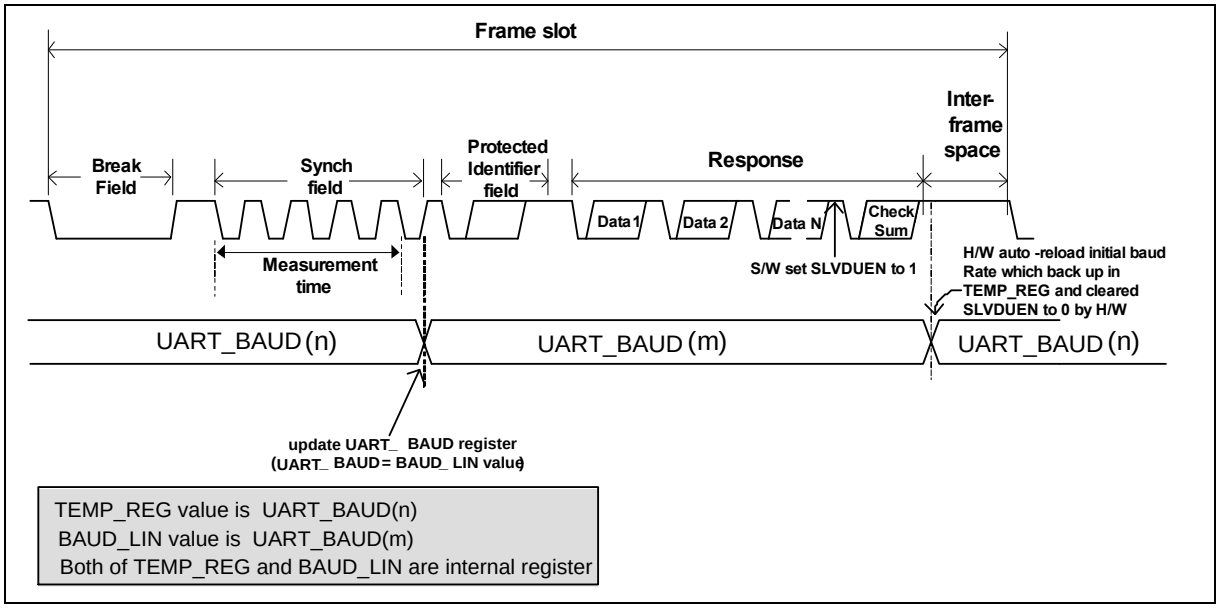


Figure 6.11-22 UART_BAUD Update Sequence in AR mode if SLVDUEN is 1

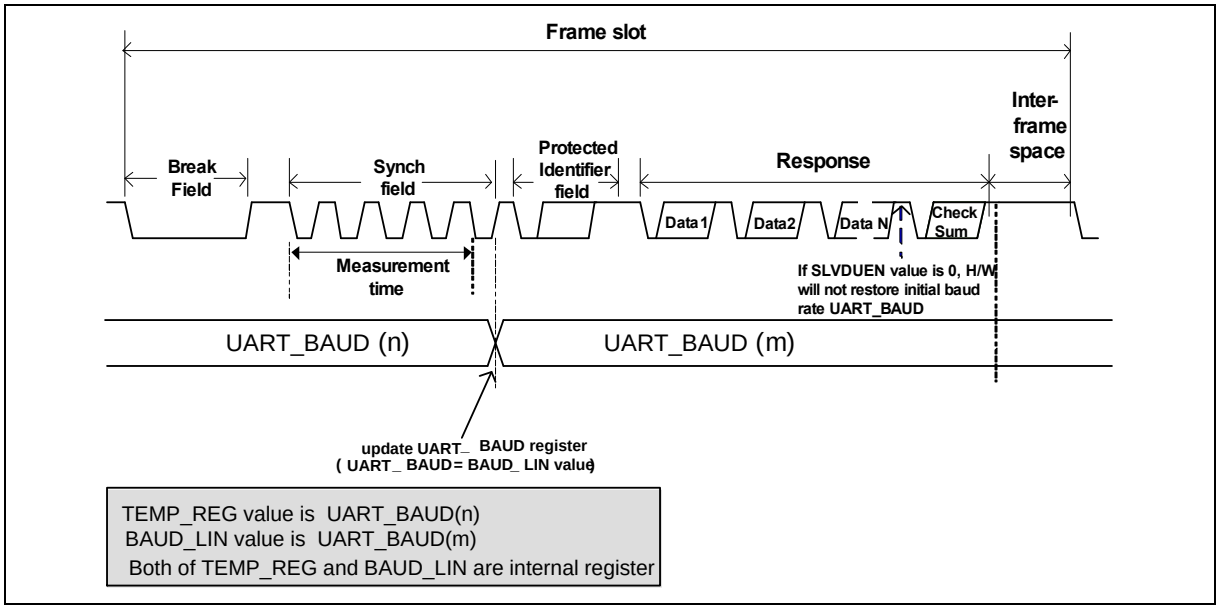


Figure 6.11-23 UART_BAUD Update Sequence in AR mode if SLVDUEN is 0

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.

6.11.5.11 RS-485 Function Mode

Another alternate function of UART controller is RS-485 function (user must set 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 same as UART in RS-485 mode.

The UART controller can be configured as an RS-485 addressable slave and 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 UART_LINE register to control the 9-th bit (When the PBE, EPE and SPE are set, the 9-th bit is transmitted 0 and when PBE and SPE are set and EPE is cleared, the 9-th 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), in first, software must decide the data which before the address byte be detected will be stored in RX FIFO or not. If software wants to ignore any data before address byte detected, the flow is set RXOFF (UART_FIFO[8]) then enable RS485NMM (UART_ALTCTL[8]) and the receiver will ignore any data until an address byte is detected (bit 9 = 1) and the address byte data will be stored in the RX FIFO. If software wants to

receive any data before address byte detected, the flow is disables RXOFF (UART_FIFO[8]) then enable RS485NMM (UART_ALTCTL[8]) and the receiver will received any data.

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)

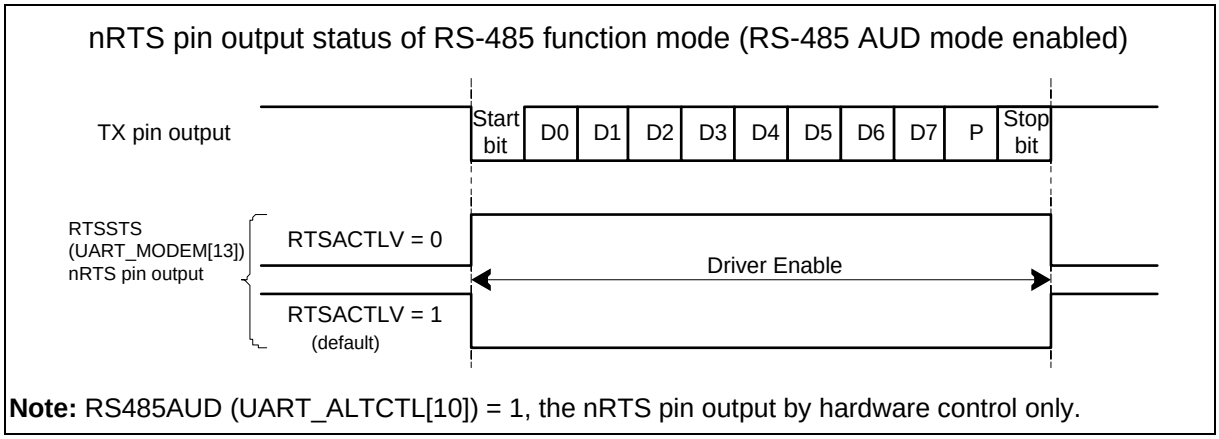
In RS-485 Auto Address Detection Operation Mode (RS485AAD (UART_ALTCTL[9]) = 1), the receiver will ignore any data until an address byte is detected (bit 9 = 1) and the address byte data matches the ADDRMV (UART_ALTCTL[31:24]) value. The address byte data will be stored in the RX FIFO. The all received 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)

Another option function of RS-485 controllers is 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 to disabled. User can set RTSACTLV in UART_MODEM register to change the nRTS driving level.

Figure 6.11-24 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.

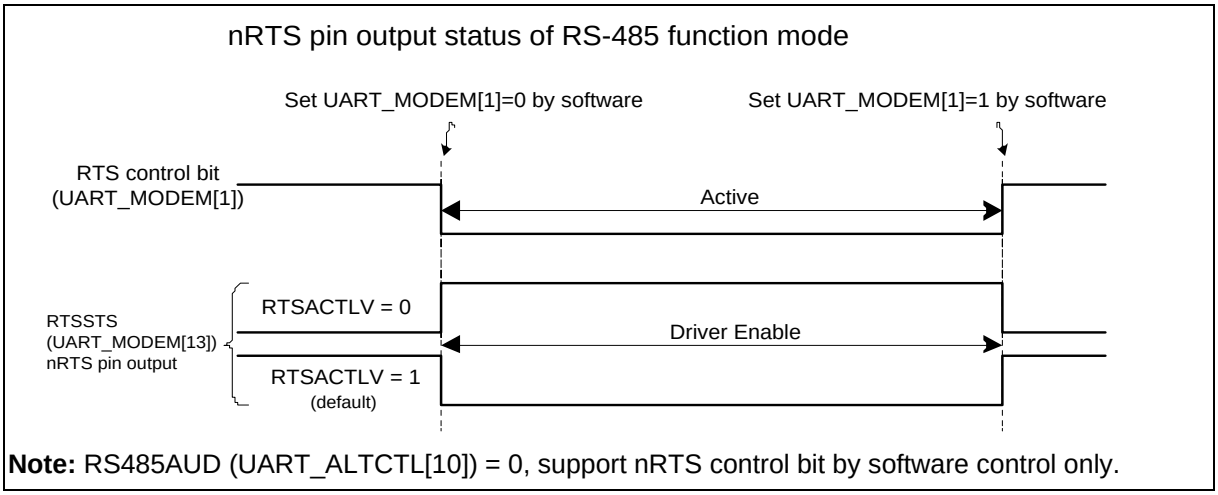


Note: RS485AUD (UART_ALTCTL[10]) = 1, the nRTS pin output by hardware control only.

Figure 6.11-24 RS-485 nRTS Driving Level in Auto Direction Mode

Figure 6.11-25 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 6.11-26.

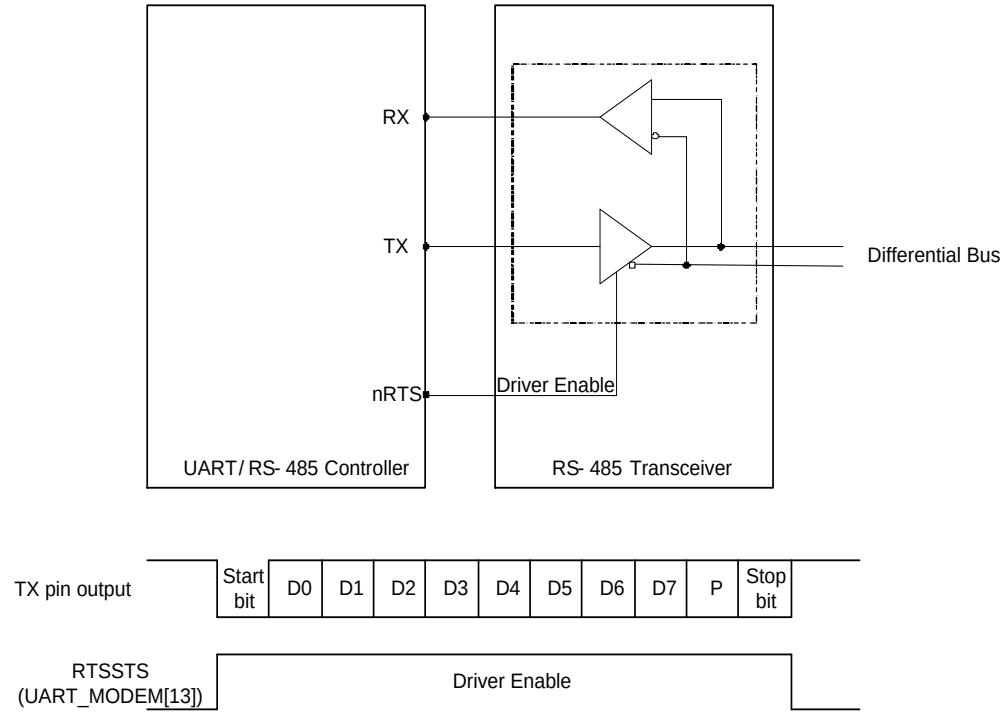


Note: RS485AUD (UART_ALTCTL[10]) = 0, support nRTS control bit by software control only.

Figure 6.11-25 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 6.11-26 Structure of RS-485 Frame

6.11.5.12 UART Single-wire Half Duplex

The UART controller provides single-wire half duplex function in UART function mode (Setting UART_FUNCSEL[2:0] to '100' to enable the UART Single-wire function). The single-wire bus keeps at RX state during the single-wire bus is idle. By writing data to TX buffer DAT (UART_DAT[7:0]), the single wire transfers the bus state to TX state immediately. After end of trasmition, the single wire transfer bus type from TX state to RX state.

To reduce the bus conflict problem, the UART controller supports flow control and bit error detection. The nRTS is inactivated during the bus keeping in TX state. The default state of the UART is RX mode, and the UART only changes to TX mode to send data after the nCTS is inactivated when ATOCTSEN (UART_INTEN[13]) is enabled. Under the TX state, the UART controller will monitor bus state. If the bus state is not equal to the UART controller TX state, the SWBEIF (UART_INTSTS[16]) will be set.

Note1: Before writing data to TX buffer, the bus state should be checked in idle state by RXIDLE (UART_FIFOSTS[29]). And the bus confliction may cause RX receive broken data.

Note2: Single-wire does not support auto-flow control. Because the nRTS is actived automatic during TX transmitted.

6.11.5.13 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.

Note: If STOPn (PDMA_STOP[n]) is set to stop UART RXPDMA task and the UART receive is not finished, the UART controller will complete the transfer and store the current receive data in receive buffer. Reading RXEMPTY (UART_FIFOSTS[14]) will check whether there is valid data in receive buffer or not.

6.11.6 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**				
UART_DAT **x=0,1**	UARTx_BA+0x00	R/W	UART Receive/Transmit Buffer Register	Undefined
UART_INTEN **x=0,1**	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000
UART_FIFO **x=0,1**	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101
UART_LINE **x=0,1**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000
UART_MODEM **x=0,1**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200
UART_MODEMSTS **x=0,1**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110
UART_FIFOSTS **x=0,1**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000
UART_INTSTS **x=0,1**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002
UART_TOUT **x=0,1**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000
UART_BAUD **x=0,1**	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000
UART_IRDA **x=0,1**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040
UART_ALTCTL **x=0,1**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C
UART_FUNCSEL **x=0,1**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000
UART_LINCTL **x=0,1**	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000
UART_LINSTS **x=0,1**	UARTx_BA+0x38	R/W	UART LIN Status Register	0x0000_0000

UART_BRCOMP **x=0,1**	UARTx_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000
UART_WKCTL **x=0,1**	UARTx_BA+0x40	R/W	UART Wake-up Control Register	0x0000_0000
UART_WKSTS **x=0,1**	UARTx_BA+0x44	R/W	UART Wake-up Status Register	0x0000_0000
UART_DWKCOMP **x=0,1**	UARTx_BA+0x48	R/W	UART Incoming Data Wake-up Compensation Register	0x0000_0000
UART_LINRTOUT **x=0,1**	UARTx_BA+0x4C	R/W	UART LIN Response Time-out Register	0x00FF_FFFF
UART_LINWKCTL **x=0,1**	UARTx_BA+0x50	R/W	UART LIN Wake-up Control Register	0x0000_00C0

6.11.7 Register Description

UART Receive/Transmit Buffer Register (UART_DAT)

Register	Offset	R/W	Description	Reset Value
UART_DAT x=0,1	UARTx_BA+0x00	R/W	UART Receive/Transmit Buffer Register	Undefined

31	30	29	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
UART_INTEN x=0,1	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 Inerrupt Disabled. 1 = Single-wire Bit Error Detect Inerrupt 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
UART_FIFO x=0,1	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. **Note1:** This bit will automatically clear at least 3 UART peripheral clock cycles. **Note2:** 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. **Note1:** This bit will automatically clear at least 3 UART peripheral clock cycles. **Note2:** 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
UART_LINE **x=0,1**	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. **Note1:** 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. **Note2:** 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. **Note1:** 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. **Note2:** 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. **Note1:** This bit has effect only when PBE (UART_LINE[3]) is set. **Note2:** 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
UART_MODEM x=0,1	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) **Note1:** Refer to Figure 6.11-13 and Figure 6.11-14 for UART function mode. **Note2:** Refer to Figure 6.11-24 and Figure 6.11-25 for RS-485 function mode. **Note3:** 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 (Request-to-send) Signal Control** This bit is direct control internal nRTS 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. **Note1:** The nRTS signal control bit is not effective when nRTS auto-flow control is enabled in UART function mode. **Note2:** The nRTS signal control bit is not effective when RS-485 auto direction mode (AUD) is enabled in RS-485 function mode. **Note3:** Single-wire mode supports this feature.
[0]	**Reserved**	Reserved.

UART Modem Status Register (UART_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
UART_MODEMSTS x=0,1	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
UART_FIFOSTS x=0,1	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 can not 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’). **Note1:** This field is used for RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1 to enable Address detection mode. **Note2:** 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
UART_INTSTS x=0,1	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. **Note1:** In RS-485 function mode, this field include “receiver detect any address byte received address byte character (bit9 = ‘1’) bit". **Note2:** 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. **Note3:** 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. **Note1:** This bit is active when FUNCSEL (UART_FUNCSEL[2:0]) is select UART Single-wire mode. **Note2:** 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 slave header error detect (SLVHEF (UART_LINSTS[1] = 1)), LIN slave ID parity error (SLVIDPEF (UART_LINSTS[2] = 1)), LIN slave header time-out detect (SLVHTOF (UART_LINSTS[4] = 1)), LIN response time-out detect (RTOUTF (UART_LINSTS[5] = 1)), LIN break detect (BRKDETF (UART_LINSTS[8] = 1)), or bit error detect (BITEF (UART_LINSTS[9] = 1)). If LINIEN (UART_INTEN[8]) is enabled the LIN interrupt will be generated. 0 = None of SLVHDETF, SLVHEF, SLVIDPEF, SLVHTOF, RTOUTF, BITEF, and BRKDETF is generated. 1 = At least one of SLVHDETF, SLVHEF, SLVIDPEF, SLVHTOF, RTOUTF, BITEF, and BRKDETF is generated. **Note:** This bit is cleared when SLVHDETF (UART_LINSTS[0]), SLVHEF (UART_LINSTS[1]), SLVIDPEF (UART_LINSTS[2]), SLVHTOF (UART_LINSTS[4]), RTOUTF (UART_LINSTS[5]), BRKDETF (UART_LINSTS[8]), and BITEF (UART_LINSTS[9]) 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. **Note1:** 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. **Note2:** 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. **Note3:** 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** 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 it will be cleared when writing data into UART_DAT (TX FIFO not empty).
[0]	**RDAIF**	**Receive Data Available Interrupt Flag** 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 it 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
UART_TOUT x=0,1	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000

31	30	29	28	27	26	25	24
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	
[31:16]	**Reserved**	Reserved.
[15:8]	**DLY**	**TX Delay Time Value** This field is used to programming 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
UART_BAUD x=0,1	UARTx_BA+0x24	R/W	UART 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. 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 6.11-4. **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 6.11-4.
[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 6.11-4.
[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 6.11-4.

UART IrDA Control Register (UART_IRDA)

Register	Offset	R/W	Description	Reset Value
UART_IRDA x=0,1	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) **Note1:** 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. **Note2:** 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. **Note1:** 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. **Note2:** 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
UART_ALTCTL x=0,1	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 (AUD)** 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 (AAD)** 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 (NMM)** 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. **Note1:** This break field length is BRKFL + 1. **Note2:** 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
UART_FUNCSEL x=0,1	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
Reserved	DGE	Reserved		TXRXDIS	FUNCSEL		

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**DGE**	**Deglitch Enable Bit** 0 = Deglitch Disabled. 1 = Deglitch Enabled. **Note:** When this bit is set to logic 1, any pulse width less than about 300 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.
[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 disable immediately when this bit is set. The TX and RX compelet current task before disable TX and RX. When TX and RX disable, 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
UART_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		RTOUTEN	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. **Note1:** User can fill any 8-bit value to this field and the bit 24 indicates ID0 (LSB first). **Note2:** 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 indicates 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. **Note1:** 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]). **Note2:** This break field length is BRKFL + 1. **Note3:** According to LIN spec, the reset value is 12 (break field length = 13).
[16:15]	**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. **Note1:** 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. **Note2:** 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. **Note1:** 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]). **Note2:** 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.
[7:6]	**Reserved**	Reserved.
[5]	**RTOUTEN**	**LIN Response Time-out Detection Enable Bit** 0 = LIN response time-out detection Disabled. 1 = LIN response time-out detection Enabled. **Note1:** This bit detects both master/slave response time-out. When detecting slave response time-out, SLVEN (UART_LINCTL[0]) and SLVHDEN (UART_LINCTL[1]) must be set to 1. **Note2:** There would be an internal counter up counting after LIN master sends a header or LIN slave detects a header. The master sends header event and sends response event and the slave detects header event and receives response event will reset the counter. When the counter is equal to or bigger than LINRTOIC (UART_LINRTOUT[23:0]), a LIN response time-out is generated. **Note3:** In LIN function mode, when detecting response time-out, RTOUTF (UART_LINSTS[5]) flag will be asserted. If the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated.

[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 6.11.5.10 (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. **Note1:** This bit only is valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note2:** This bit used for LIN Slave Automatic Resynchronization mode. (for Non-Automatic Resynchronization mode, this bit should be kept cleared) **Note3:** The control and interactions of this field are explained in 6.11.5.10 (Slave mode with automatic resynchronization).
[2]	**SLVAREN**	**LIN Slave Automatic Resynchronization Mode Enable Bit** 0 = LIN automatic resynchronization Disabled. 1 = LIN automatic resynchronization Enabled. **Note1:** This bit only is valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note2:** When operation in Automatic Resynchronization mode, the baud rate setting must be mode2 (BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) must be 1). **Note3:** The control and interactions of this field are explained in 6.11.5.10 (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. **Note1:** This bit is only valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note2:** 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
UART_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		RTOUTF	SLVHTOF	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. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** 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. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid when LIN break detection function is enabled (BRKDETEN (UART_LINCTL[10]) =1).
[7:6]	**Reserved**	Reserved.
[5]	**RTOUTF**	**LIN Response Time-out Flag** This bit is set when no LIN response received and the time-out counter equal to or bigger than LINRTOIC (UART_LINRTOUT[23:0]). If LINIEN (UART_INTEN[8]) is enabled, the LIN Bus Interrupt will be generated. 0 = LIN response time-out not detected. 1 = LIN response time-out detected. **Note1:** This bit can be cleared by writing 1 to it.
[4]	**SLVHTOF**	**LIN Slave Header Time-out Flag** This bit is set by hardware when a LIN header reception time-out is detected in LIN slave mode and be cleared by writing 1 to it. When this bit is set, SLVHEF (UART_LINSTS[1]) will

		also be set. 0 = LIN header time-out not detected. 1 = LIN header time-out detected. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** 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])).
[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. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid in LIN Slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note3:** 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. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** 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. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** 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). **Note1:** This bit can be cleared by writing 1 to it. **Note2:** 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])). **Note3:** 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
UART_BRCOMP x=0,1	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
UART_WKCTL x=0,1	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. **Note1:** When the system is in Power-down mode, Received Data FIFO reached threshold time-out will wake up system from Power-down mode. **Note2:** It is suggested the function is enabled when the WKRFRTEN (UART_WKCTL[2]) is set to 1.
[3]	**WKRS485EN**	**RS-485 Address Match (AAD Mode) 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. **Note1:** When the system is in.Power-down mode, RS-485 Address Match will wake-up system from Power-down mode. **Note2:** 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
UART_WKSTSx=0,1	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. **Note1:** If WKTOUTEN (UART_WKCTL[4]) is enabled, the Received Data FIFO reached threshold time-out wake-up cause this bit is set to '1'. **Note2:** This bit can be cleared by writing '1' to it.
[3]	**RS485WKF**	**RS-485 Address Match (AAD Mode) 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. **Note1:** If WKRS485EN (UART_WKCTL[3]) is enabled, the RS-485 Address Match (AAD mode) wake-up cause this bit is set to '1'. **Note2:** 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. **Note1:** If WKRFRTEN (UART_WKCTL[2]) is enabled, the Received Data FIFO Reached Threshold wake-up cause this bit is set to '1'. **Note2:** 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. **Note1:** If WKDATEN (UART_WKCTL[1]) is enabled, the Incoming Data wake-up cause this bit is set to '1'. **Note2:** 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. **Note1:** If WKCTSEN (UART_WKCTL[0]) is enabled, the nCTS wake-up cause this bit is set to '1'. **Note2:** 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
UART_DWKCOMP **x=0,1**	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 LIN Response Time-out Register (UART_LINRTOUT)

Register	Offset	R/W	Description	Reset Value
UART_LINRTOUT **x=0,1**	UARTx_BA+0x4C	R/W	UART LIN Response Time-out Register	0x00FF_FFFF

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**LINRTOIC**	**LIN Response Time-out Comparator** The time-out counter resets and starts counting (the counting clock = UART_CLK) whenever LIN master sends a header or LIN slave detects a header. Once the content of time-out counter is equal to or bigger than that of time-out comparator (LINRTOIC (UART_LINRTOUT[23:0])), a LIN response time-out (RTOUTF (UART_LINSTS[5])) is generated if RTOUTEN (UART_LINCTL[5]) enabled.

UART LIN Wake-up Control Register (UART_LINWKCTL)

Register	Offset	R/W	Description	Reset Value
UART_LINWKCTL x=0,1	UARTx_BA+0x50	R/W	UART LIN Wake-up Control Register	0x0000_00C0

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

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**LINWKF**	**LIN Wake-up Flag** This bit is set if chip wake-up from power-down state by LIN wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by LIN wake-up. **Note1:** If LINWKEN (UART_LINWKCTL[28]) is enabled, the LIN wake-up event will cause this bit set to '1'. **Note2:** This bit can be cleared by writing '1' to it.
[28]	**LINWKEN**	**LIN Wake-up Enable Bit** 0 = LIN wake-up system function Disabled. 1 = LIN wake-up system function Enabled. **Note1:** When the system is in Power-down mode, LIN wake-up event will wake up system from Power-down mode.
[27:25]	**Reserved**	Reserved.
[24]	**SENDLINW**	**LIN Send Wake-up Enable Bit** 0 = Send LIN Wake-up Disabled. 1 = Send LIN Wake-up Enabled. **Note1:** When this bit is set, the UART will send LIN wake-up automatically. When LIN wake-up transfer operation finished, this bit will be cleared automatically.
[23:0]	**LINWKC**	**LIN Send Wake-up Signal Length Counter** When SENDLINW (UART_LINWKCTL[24]) is enabled, the UART will send LIN wake-up signal for LINWKC (UART_LINWKCTL[23:0]) period (the counting clock = UART_CLK). LINWKC (UART_LINWKCTL[23:0]) should be set to a value between 250 us to 5 ms according to UART_CLK rate.

6.12 USCI - Universal Serial Control Interface Controller (USCI)

6.12.1 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.

6.12.2 Features

The controller can be individually configured to match the application needs. The following protocols are supported:

- UART
- SPI
- I²C

6.12.3 Block Diagram

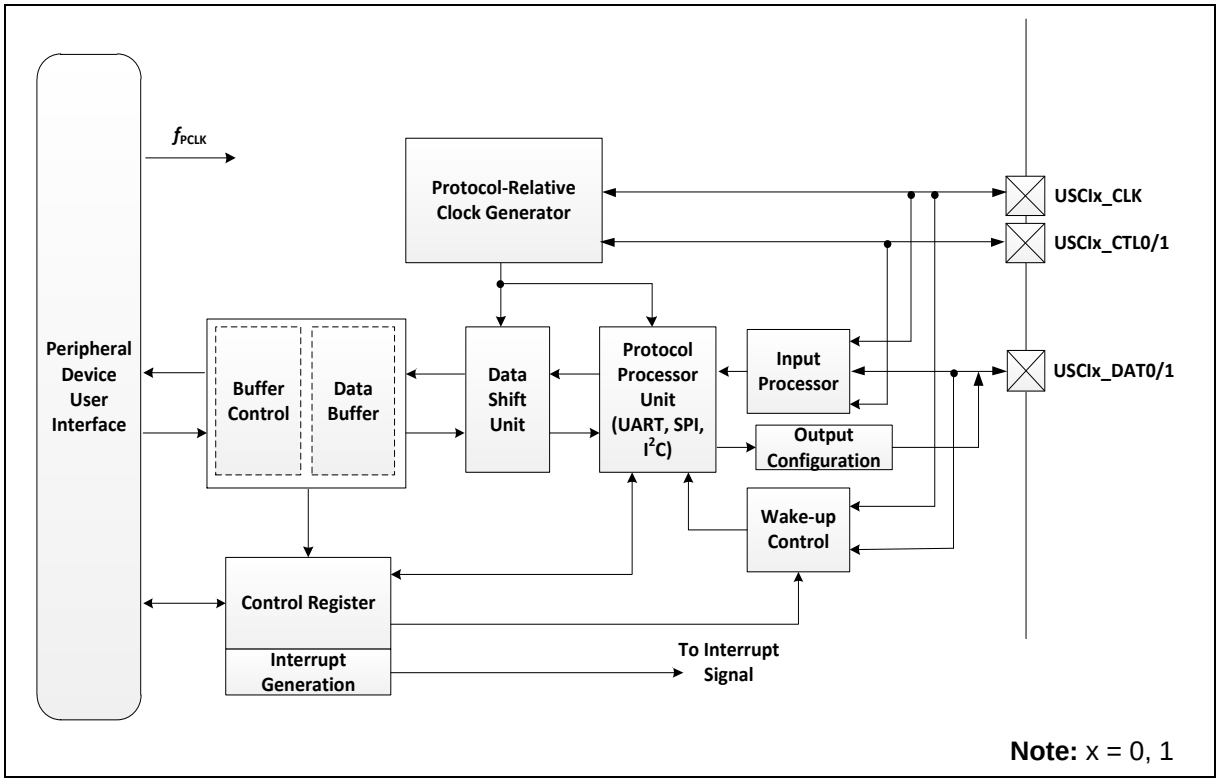


Figure 6.12-1 USCI Block Diagram

6.12.4 Functional Description

The structure of the Universal Serial Control Interface (USCI) controller is shown in Figure 6.12-1. 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.

6.12.4.1 I/O Processer

Input Signal

All input stages offer the similar feature set. They are used for all protocols.

Table 6.12-1lists 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	I^2C
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 6.12-1 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).

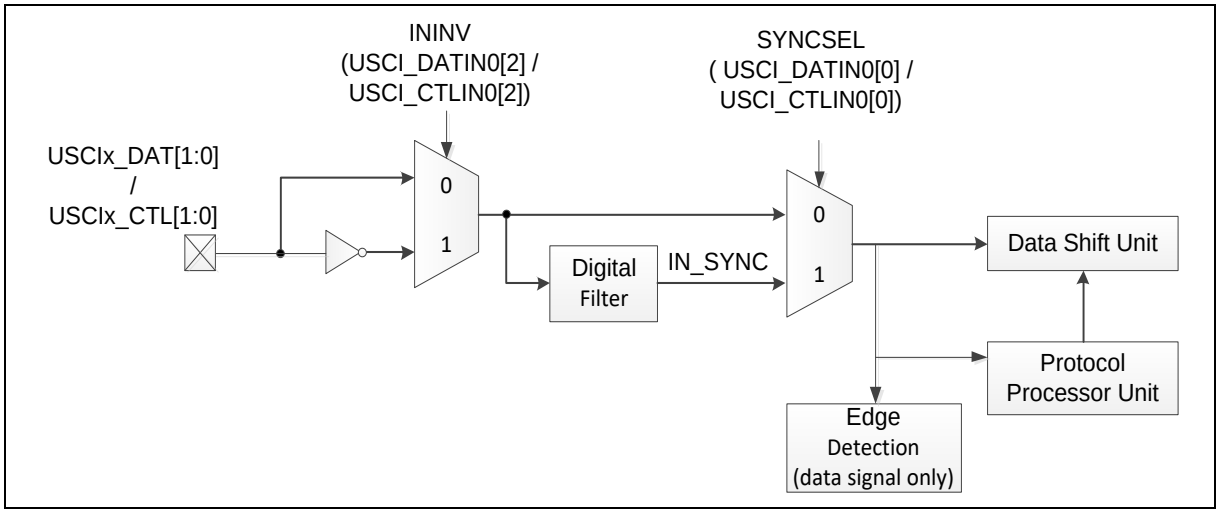


Figure 6.12-2 Input Conditioning for USCIx_DAT[1:0] and USCIx_CTL[1:0]

The input structure of USCIx_CLK is similar to USCIx_CTL[1:0] input structure, except it does not support inverse function.

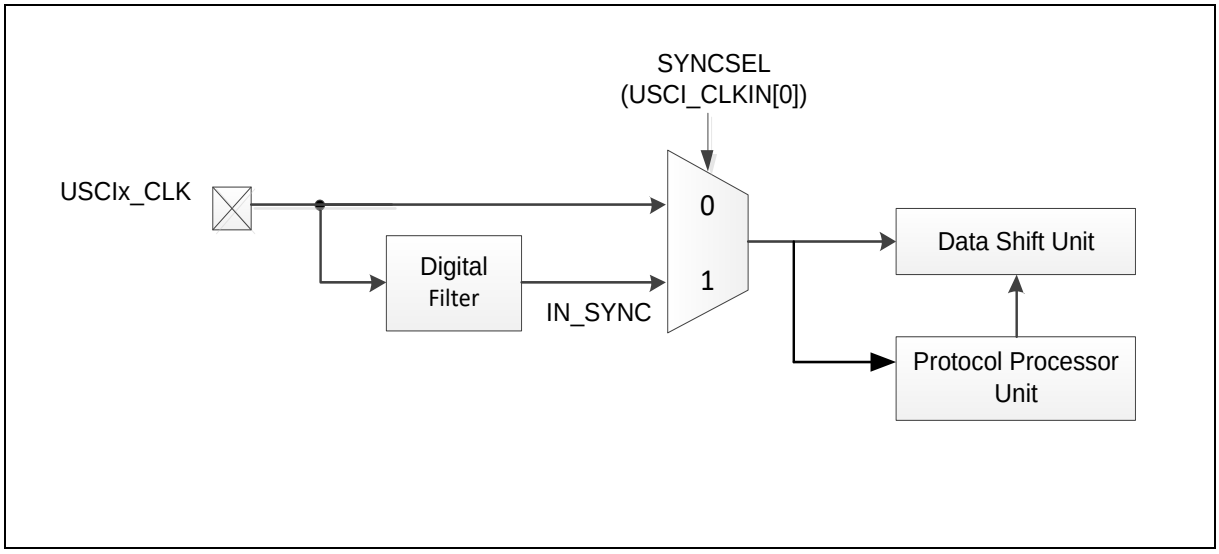


Figure 6.12-3 Input Conditioning for USCIx_CLK

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 6.12-2 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 6.12-2 Output Signals for Different Protocols

The description of protocol-specific items are given in the related protocol chapters.

6.12.4.2 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.

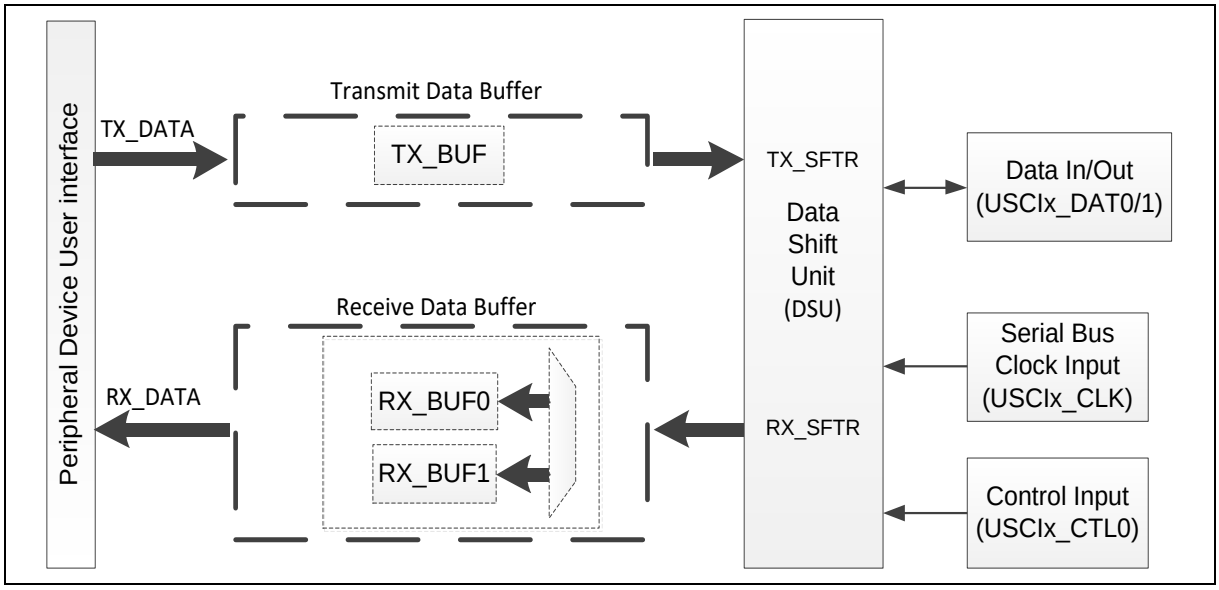


Figure 6.12-4 Block Diagram of Data Buffering

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.

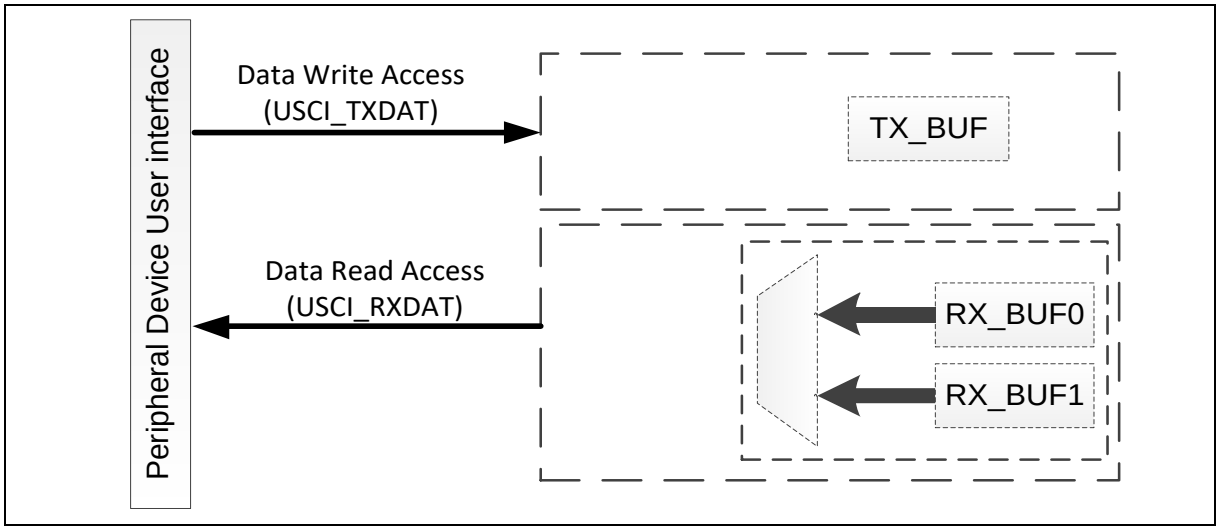


Figure 6.12-5 Data Access Structure

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.

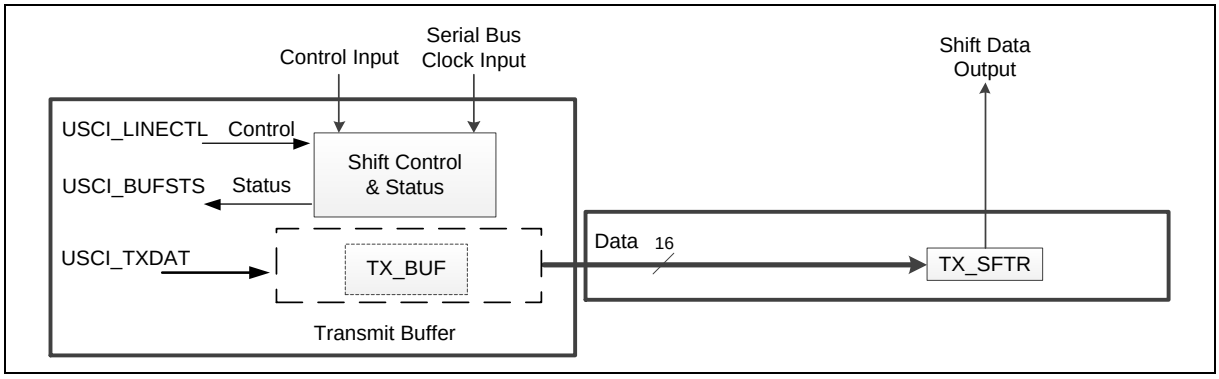


Figure 6.12-6 Transmit Data Path

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.

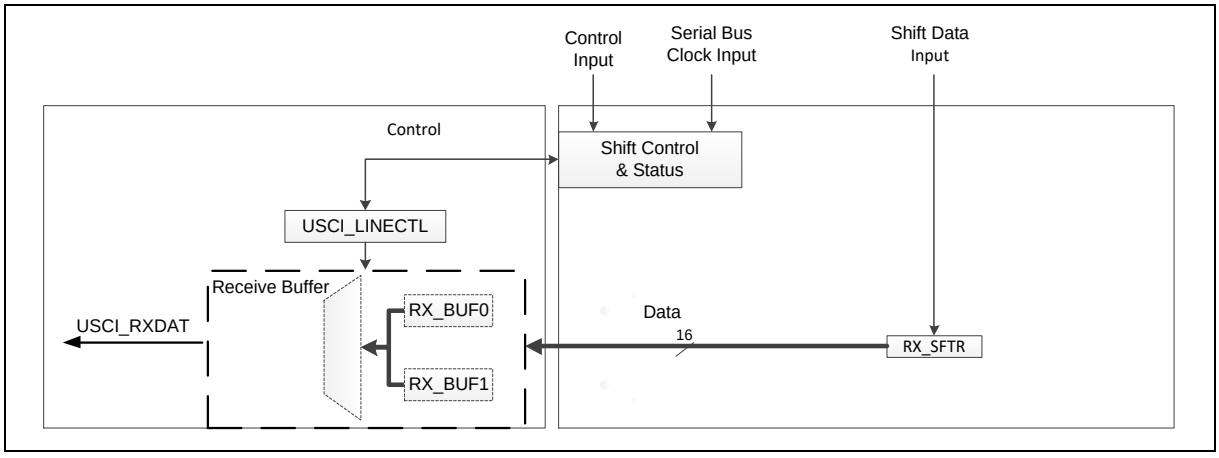


Figure 6.12-7 Receive Data Path

6.12.4.3 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.

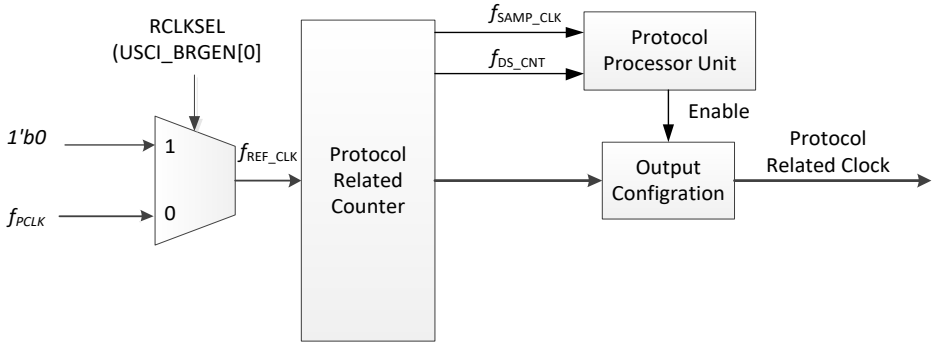
The direction of selected pin is controlled by PORTDIR (USCI_TXDAT[16]). When user writes USCI_TXDAT register, the transmit data and its port direction are settled simultaneously.

6.12.4.4 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.

6.12.4.5 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 to the **Basic Clock Divider Counter** section to get the f_{SAMP_CLK}

Figure 6.12-8 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 (fSAMP_CLK and fDS_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.

$$f_{\mathrm{DIV_CLK}} = f_{REF_CLK} \times \frac{1}{\mathrm{CLKDIV} + 1} \text{ if } \mathrm{PTCLKSEL} = 0$$

$$f_{\mathrm{DIV_CLK}} = f_{REF_CLK} \times \frac{1}{(\mathrm{CLKDIV} + 1) \times 2} \text{ if } \mathrm{PTCLKSEL} = 1$$

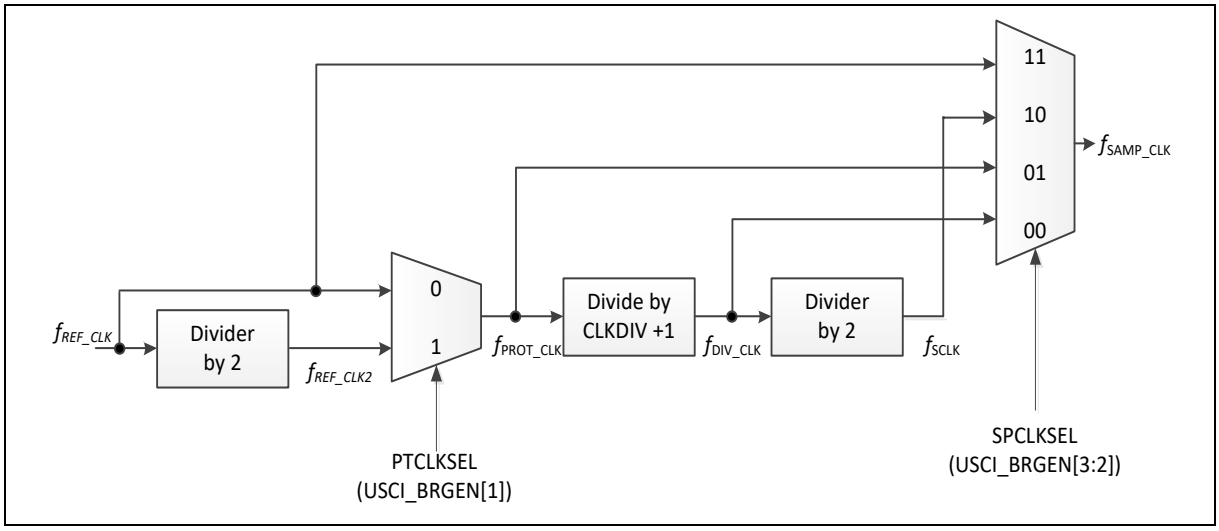


Figure 6.12-9 Basic Clock Divider Counter

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.

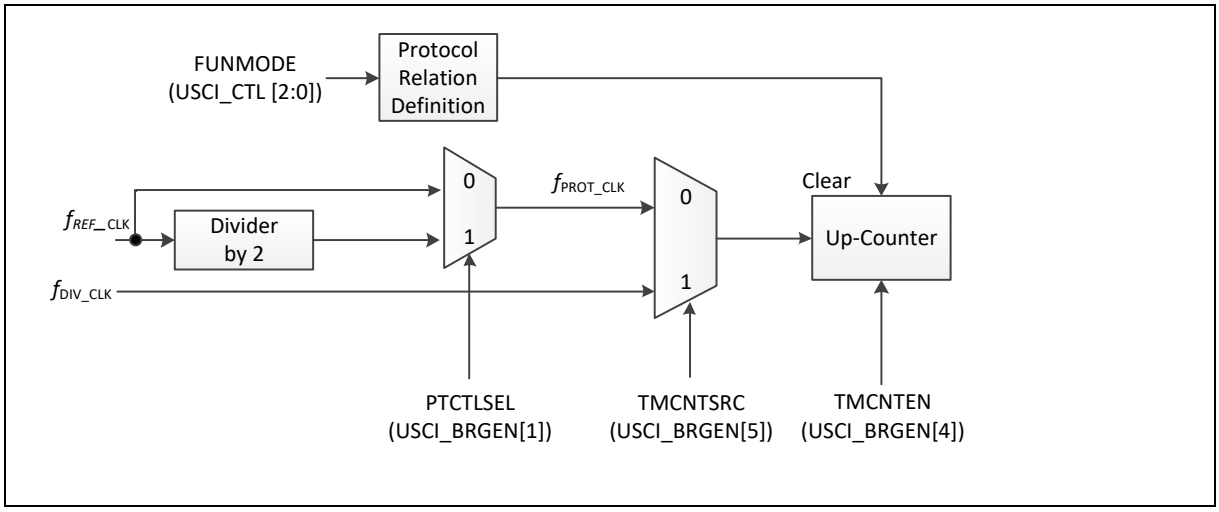


Figure 6.12-10 Block of Timing Measurement Counter

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.

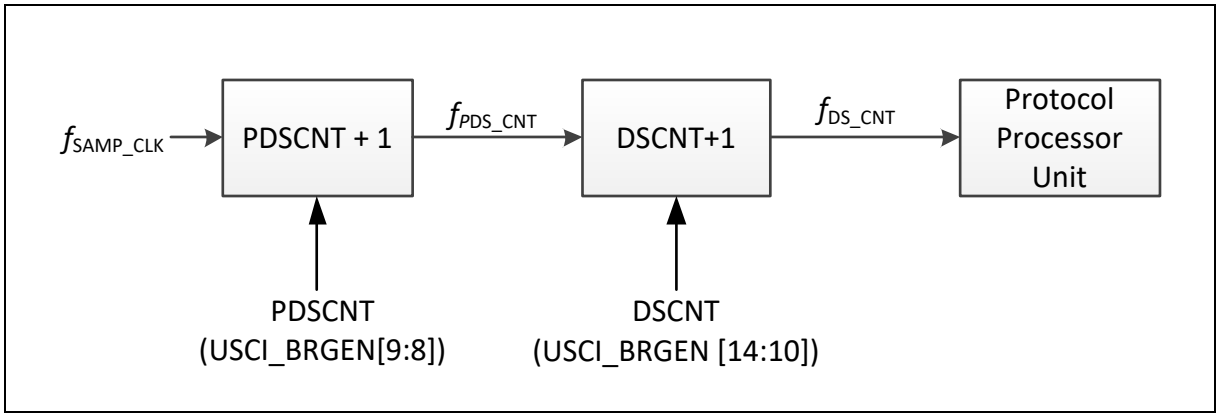


Figure 6.12-11 Sample Time Counter

6.12.4.6 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 6.12-12.

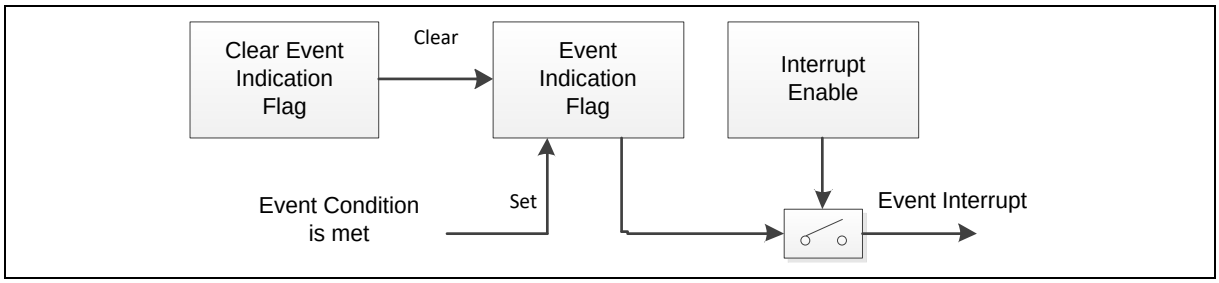


Figure 6.12-12 Event and Interrupt Structure

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 6.12-3.

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 6.12-3 Data Transfer Events and Interrupt Handling

6.12.4.7 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 6.12-4 Protocol-specific Events and Interrupt Handling

6.12.4.8 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.

6.12.4.9 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.

6.13 USCI – UART Mode

6.13.1 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 being independent, frames can start at different points in time for transmission and reception.

The UART controller also provides auto flow control. There are two conditions to wake-up the system.

6.13.2 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 by built-in capture event of baud rate generator
- Supports PDMA capability
- Supports Wake-up function (Data and nCTS Wakeup Only)

6.13.3 Block Diagram

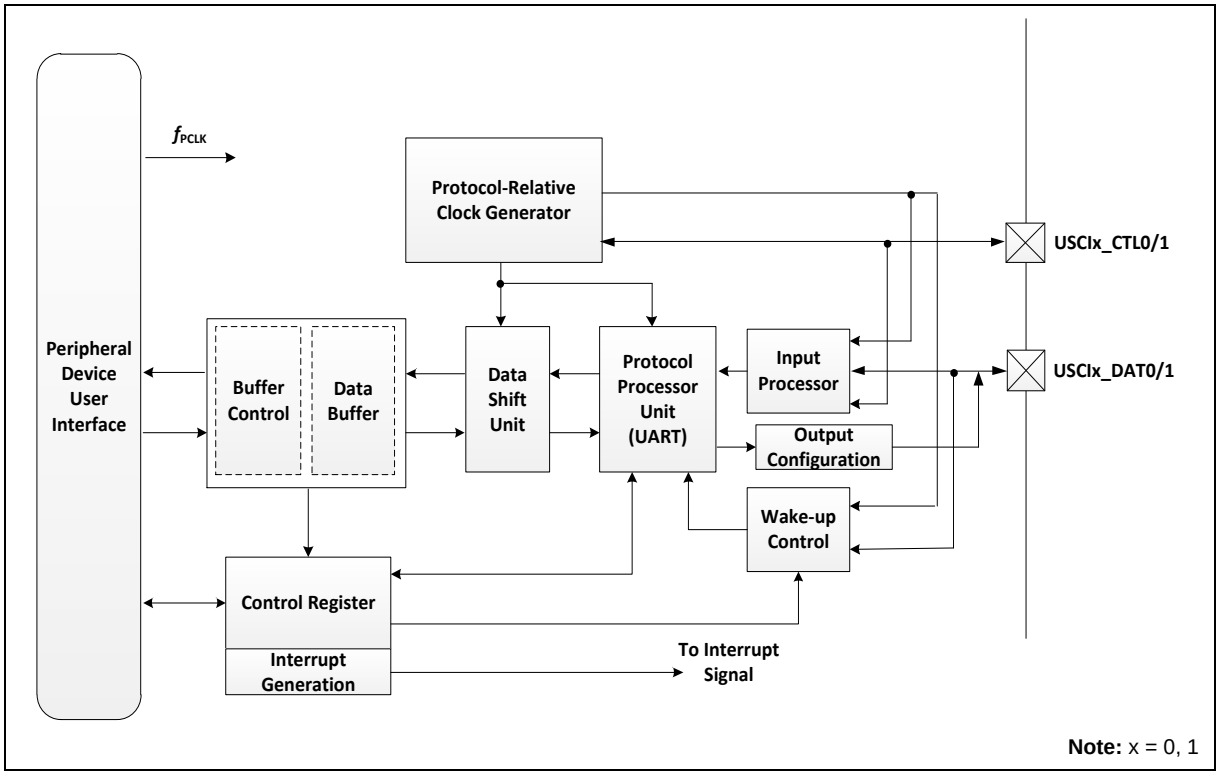


Figure 6.13-1 USCI-UART Mode Block Diagram

6.13.4 Basic Configuration

The basic configurations of USCI0_UART are as follows:

- Clock Source Configuration
- Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK1[8]).
- Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).
- Enable USCI0_UART function UUART_CTL[2:0] register, UUART_CTL[2:0]=3'b010.
- Pin Configuration

Group	Pin Name	GPIO	MFP
USCI0	USCI0_CTL0	PD.7	MFP4
		PA.0, PA.1, PA.2, PA.3, PA.4, PA.5 PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	MFP13
	USCI0_CTL1	PA.0, PA.2, PA.5 PB.4, PB.6 PC.1, PC.3, PC.5, PC.7	MFP14
	USCI0_DAT0	PD.5	MFP4
		PA.0, PA.1, PA.2, PA.3, PA.4, PA.5 PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	MFP11
	USCI0_DAT1	PD.6	MFP4
		PA.0, PA.1, PA.2, PA.4, PA.5 PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	MFP12

The basic configurations of USCI1_UART are as follows:

- Clock Source Configuration
- Enable USCI1 peripheral clock in USCI1CKEN (CLK_APBCLK1[9]).
- Reset USCI1 controller in USCI1RST (SYS_IPRST2[9]).
- Enable USCI1_UART function UUART_CTL[2:0] register, UUART_CTL[2:0]=3'b010.
- Pin Configuration

Group	Pin Name	GPIO	MFP
USCI1	USCI1_CTL0	PD.3	MFP4
		PA.0, PA.1, PA.2, PA.3, PA.4, PA.5 PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	MFP18
	USCI1_CTL1	PA.1, PA.4 PB.5, PB.7 PC.0, PC.2, PC.4, PC.6	MFP14
	USCI1_DAT0	PD.1	MFP4
		PA.0, PA.1, PA.2, PA.3, PA.4, PA.5	MFP16

		PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	
	USCI1_DAT1	PD.2	MFP4
		PA.0, PA.1, PA.2, PA.4, PA.5 PB.4, PB.5, PB.6, PB.7 PC.0, PC.1, PC.2, PC.3, PC.4, PC.5, PC.6, PC.7	MFP17

6.13.5 Functional Description

6.13.5.1 USCI Common Function Description

Please refer to section 6.12.4 for detailed information.

6.13.5.2 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 6.13-2 shows an example with a point-to-point full-duplex connection between two communication partners UART module A and UART module B.

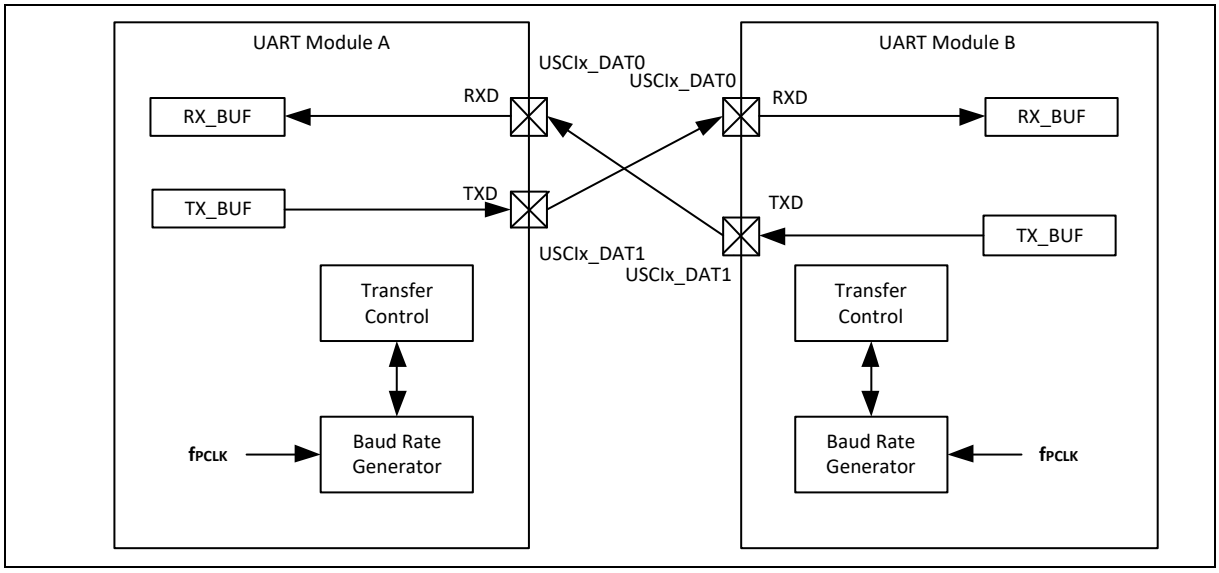


Figure 6.13-2 UART Signal Connection for Full-Duplex Communication

Input Signals

For UART protocol, the number of input signals is shown in Table 6.13-1. 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 6.13-1).

Selected Protocol		UART
Control Input	USCIx_CTL0	nCTS
	USCIx_CTL1	X
Data Input	USCIx_DAT0	RX

	USCIx_DAT1	X

Table 6.13-1 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	X
	USCIx_CTL1	nRTS
Data Output	USCIx_DAT0	X
	USCIx_DAT1	TX

Table 6.13-2 Output Signals for Different Protocol

6.13.5.3 Frame Format

A standard UART frame is shown in Figure 6.13-3. 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.

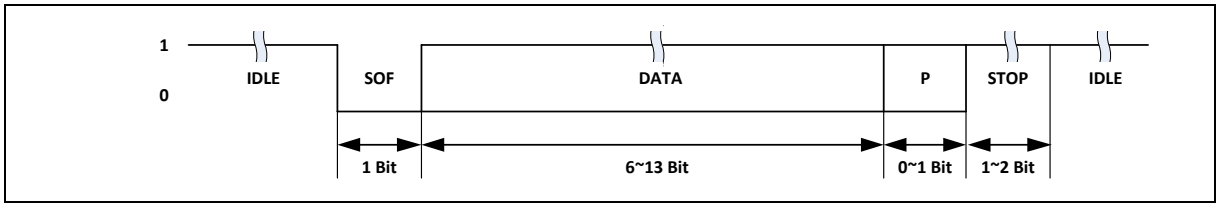


Figure 6.13-3 UART Standard Frame Format

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.

6.13.5.4 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 FUNMODOE (UUART_CTL[2:0]) to 010B 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.

6.13.5.5 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 6.13-4, 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.

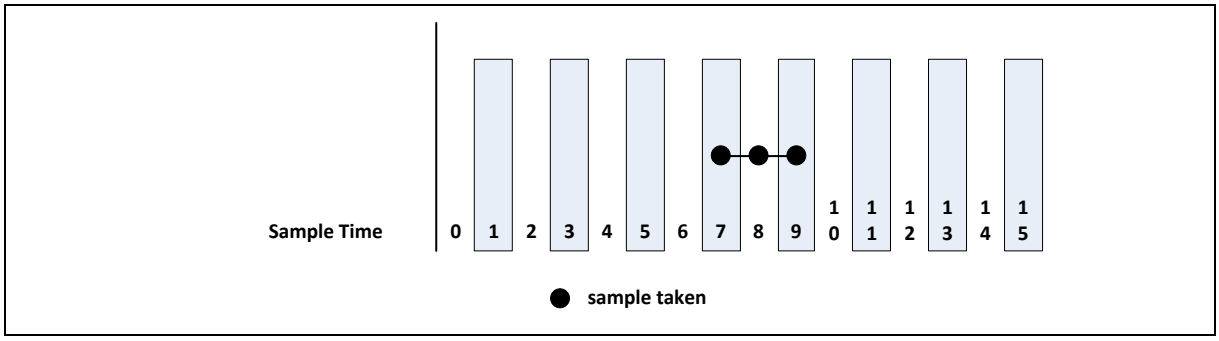


Figure 6.13-4 UART Bit Timing (data sample time)

6.13.5.6 Baud Rate Generation

The baud rate f_{UART} 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_{SAM_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 = 2'b00 ($f_{SAMP_CLK} = f_{DIV_CLK}$). 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'b10 ($f_{SAMP_CLK} = f_{SCLK}$), 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 6.15-1 lists the relative error percentage examples for user to calculate his relative baud rate setting.

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 6.13-1 Baud Rate Relationship

Note: {SPCLKSEL, PTCLKSEL, RCLKSEL = 2'b0,1'b0,1'b0}

6.13.5.7 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_PROTCTL[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_BRGEN[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 6.13-5.

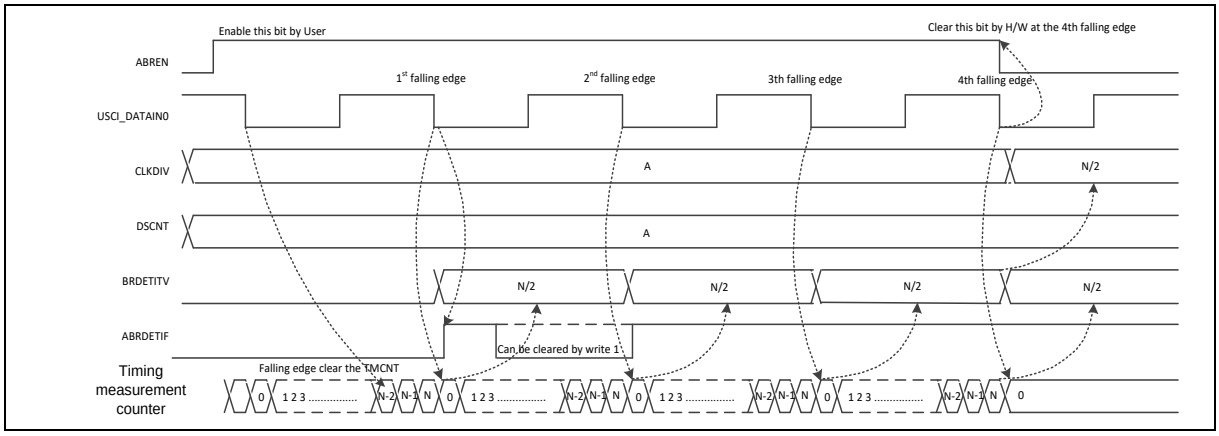


Figure 6.13-5 UART Auto Baud Rate Control

6.13.5.8 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.

6.13.5.9 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.

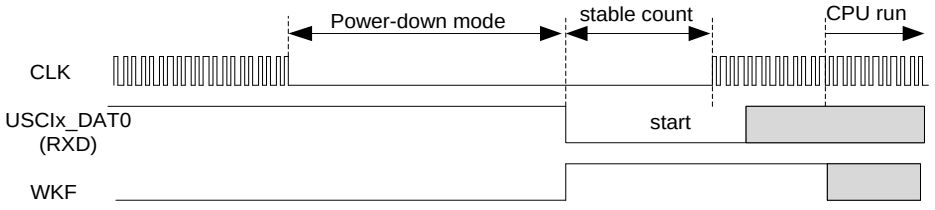
The UART controller can also play as an RS-485 addressable slave, the protocol-related error of PARITYERR (UUART_PROTSTS[5]) can be acted as the address bit detection when the PARITYEN (UUART_PROTCTL[1]), EVENPARITY (UUART_PROTCTL[2]) and STICKEN (UUART_PROTCTL[26]) were set. If the PARITYERR was set, it means that the address bit in the received bus is detected otherwise, the data is received into Buffer.

6.13.5.10 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-wp is shown in Figure 6.13-6.



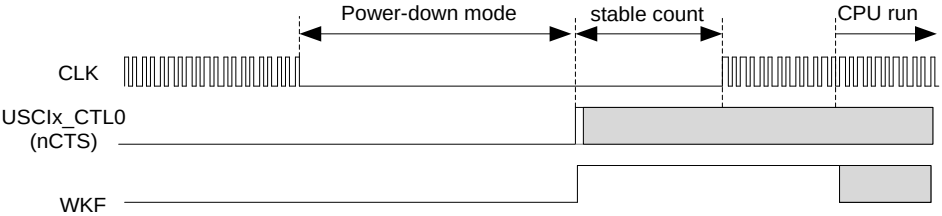
Note: Stable count means HCLK source recovery stable count.

Figure 6.13-6 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-wp is shown in Figure 6.13-7 and Figure 6.13-8.

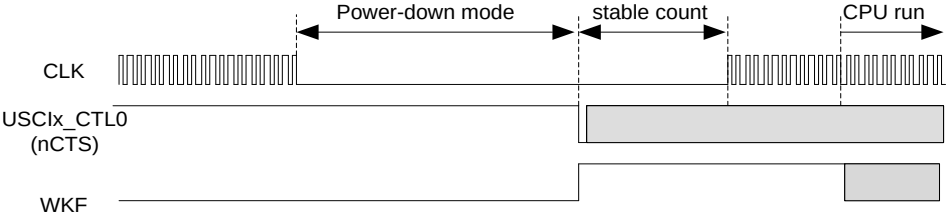
Case 1 (nCTS transition from low to high):



Note: Stable count means HCLK source recovery stable count.

Figure 6.13-7 nCTS Wake-Up Case 1

Case 2 (nCTS transition from high to low):



Note 1: Stable count means HCLK source recovery stable count.

Figure 6.13-8 nCTS Wake-Up Case 2

6.13.5.11 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.

6.13.5.12 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.

6.13.6 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**				
UUART_CTL	UUARTn_BA+0x00	R/W	USCI Control Register	0x0000_0000
UUART_INTEN	UUARTn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000
UUART_BRGEN	UUARTn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
UUART_DATIN0	UUARTn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000
UUART_CTLIN0	UUARTn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000
UUART_CLKIN	UUARTn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000
UUART_LINECTL	UUARTn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
UUART_TXDAT	UUARTn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
UUART_RXDAT	UUARTn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
UUART_BUFCTL	UUARTn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000
UUART_BUFSTS	UUARTn_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101
UUART_PDMACTL	UUARTn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000
UUART_WKCTL	UUARTn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
UUART_WKSTS	UUARTn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
UUART_PROTCTL	UUARTn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000
UUART_PROTIEN	UUARTn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
UUART_PROTSTS	UUARTn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

6.13.7 Register Description

USCI Control Register (UUART_CTL)

Register	Offset	R/W	Description	Reset Value
UUART_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. **Note:** Other bit combinations are reserved.

USCI Interrupt Enable Register (UUART_INTEN)

Register	Offset	R/W	Description	Reset Value
UUART_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 Blt** 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
UUART_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} = 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 Input Data Signal Configuration (UUART_DATIN0)

Register	Offset	R/W	Description	Reset Value
UUART_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 10.
[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
UUART_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
UUART_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
UUART_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. 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]. **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
UUART_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
UUART_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
UUART_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
UUART_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
UUART_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
UUART_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
UUART_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
UUART_PROTCTL	UUARTn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	DGE	BCEN	Reserved	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:** When this bit is set to logic 1, any pulse width less than about 300 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.
[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.
[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_PROTSTS[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.
[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
UUART_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
UUART_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.

6.14 USCI - SPI Mode

6.14.1 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.

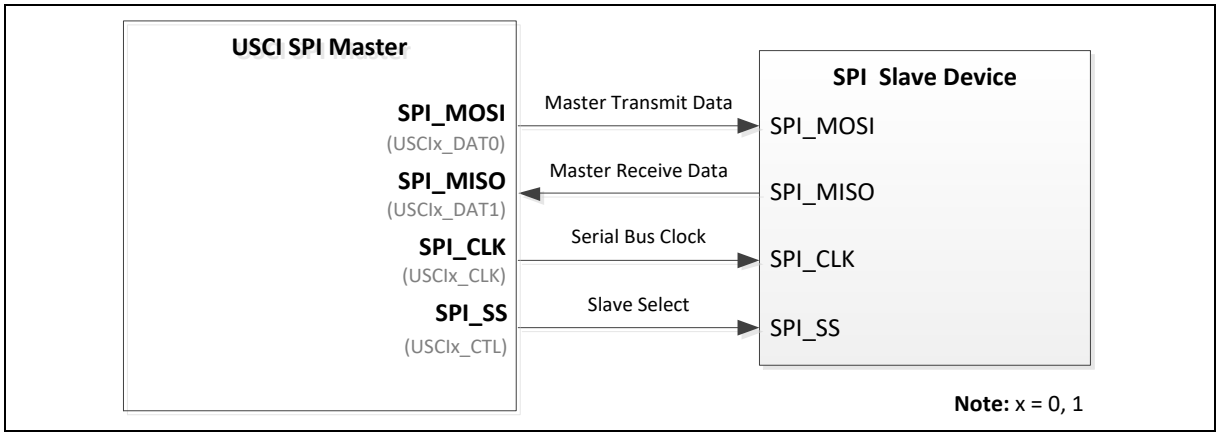


Figure 6.14-1 SPI Master Mode Application Block Diagram

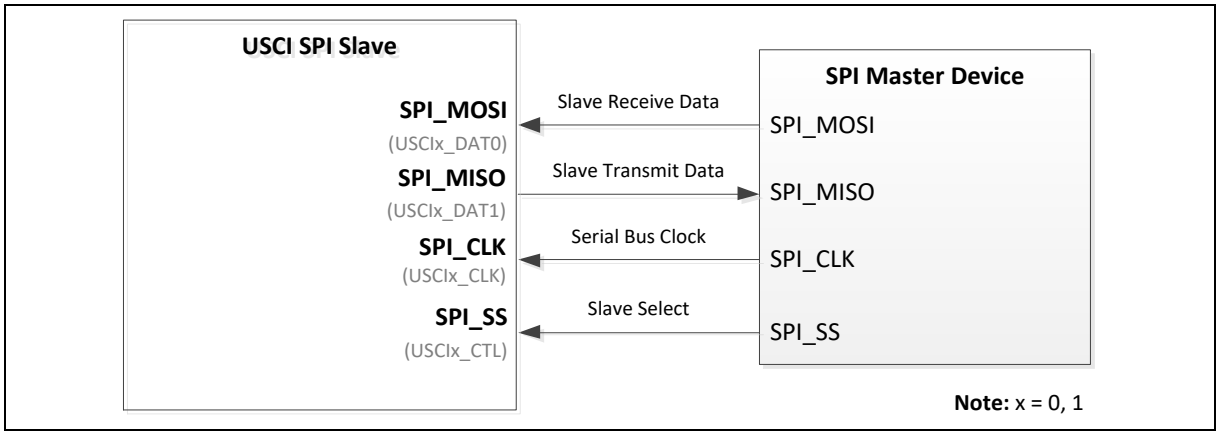


Figure 6.14-2 SPI Slave Mode Application Block Diagram

6.14.2 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

6.14.3 Block Diagram

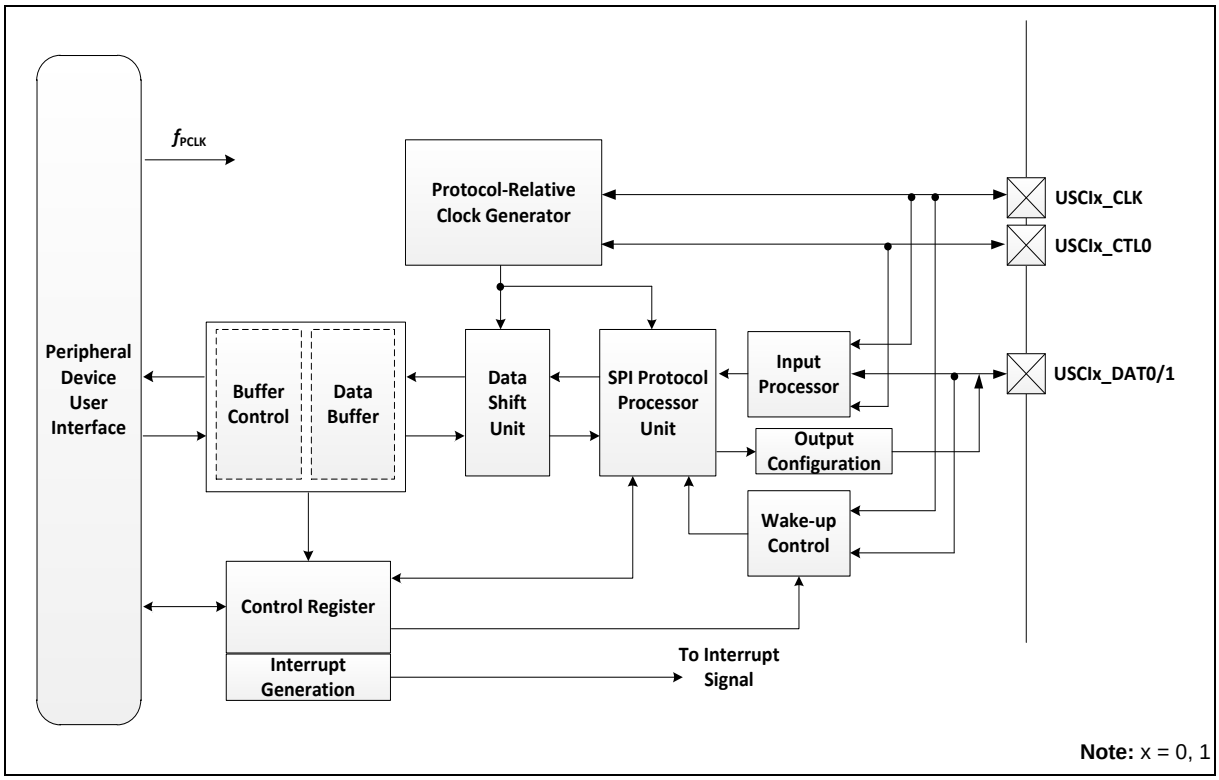


Figure 6.14-3 USCI SPI Mode Block Diagram

6.14.4 Basic Configuration

6.14.4.1 USCI0 SPI Basic Configurations

- Clock Source Configuration
  - Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK1[8]).
  - Enable USCI0_SPI function on USCI0 USPI_CTL[2:0] register, USPI_CTL[2:0]=3'b001
- Reset Configuration
  - Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
USCI0	USCI0_CLK	PA.0~PA.5, PB.4~PB.7, PC.0~PC.7	MFP10
		PD.4	MFP4

	USCI0_CTL0	PA.0~PA.5, PB.4~PB.7, PC.0~PC.7	MFP13
		PD.7	MFP4
	USCI0_DAT0	PA.0~PA.5, PB.4~PB.7, PC.0~PC.7	MFP11
		PD.5	MFP4
	USCI0_DAT1	PA.0~PA.5, PB.4~PB.7, PC.0~PC.7	MFP12
		PD.6	MFP4

6.14.4.2 USCI1 SPI Basic Configurations

- Clock Source Configuration
  - Enable USCI1 peripheral clock in USCI1CKEN (CLK_APBCLK1[9]).
  - Enable USCI1_SPI function on USCI1 USPI_CTL[2:0] register, USPI_CTL[2:0]=3'b001
- Reset Configuration
  - Reset USCI1 controller in USCI0RST (SYS_IPRST2[9]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
USCI1	USCI1_CLK	PA.0~PA.5, PB.4~PB.7, PC.0~PC.7	MFP15
		PD.0	MFP4
	USCI1_CTL0	PA.0~PA.5, PB.4~PB.7, PC.0~PC.7	MFP18
		PD.3	MFP4
	USCI1_DAT0	PA.0~PA.5, PB.4~PB.7, PC.0~PC.7	MFP16
		PD.1	MFP4
	USCI1_DAT1	PA.0~PA.5, PB.4~PB.7, PC.0~PC.7	MFP17
		PD.2	MFP4

6.14.5 Functional Description

6.14.5.1 USCI Common Function Description

Please refer to section 6.12.4 for detailed information.

6.14.5.2 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

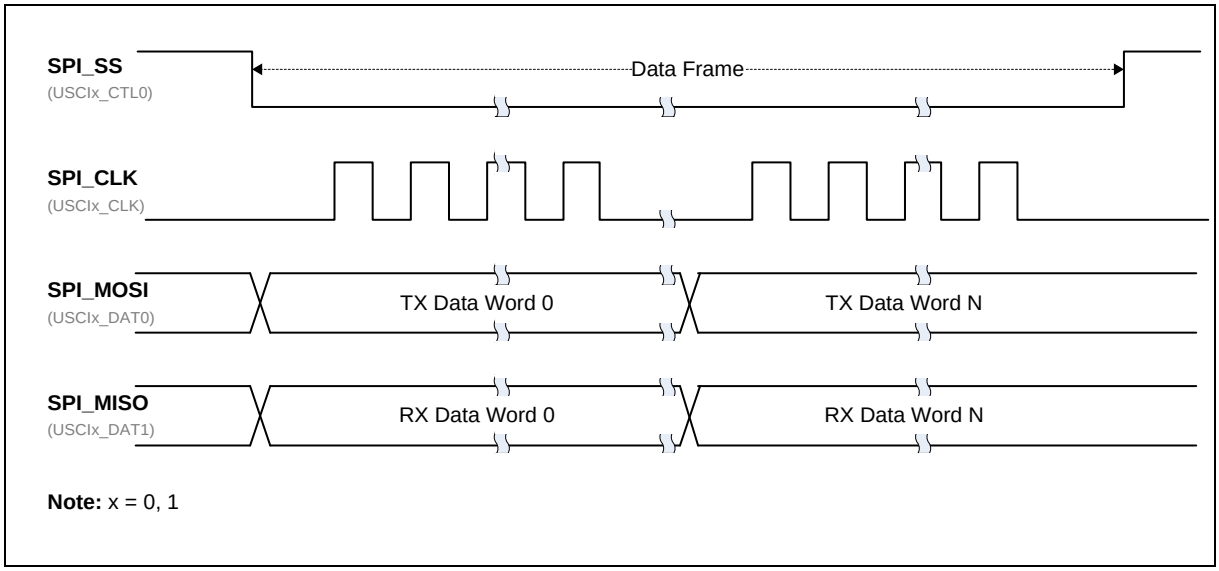


Figure 6.14-4 4-Wire Full-Duplex SPI Communication Signals (Master Mode)

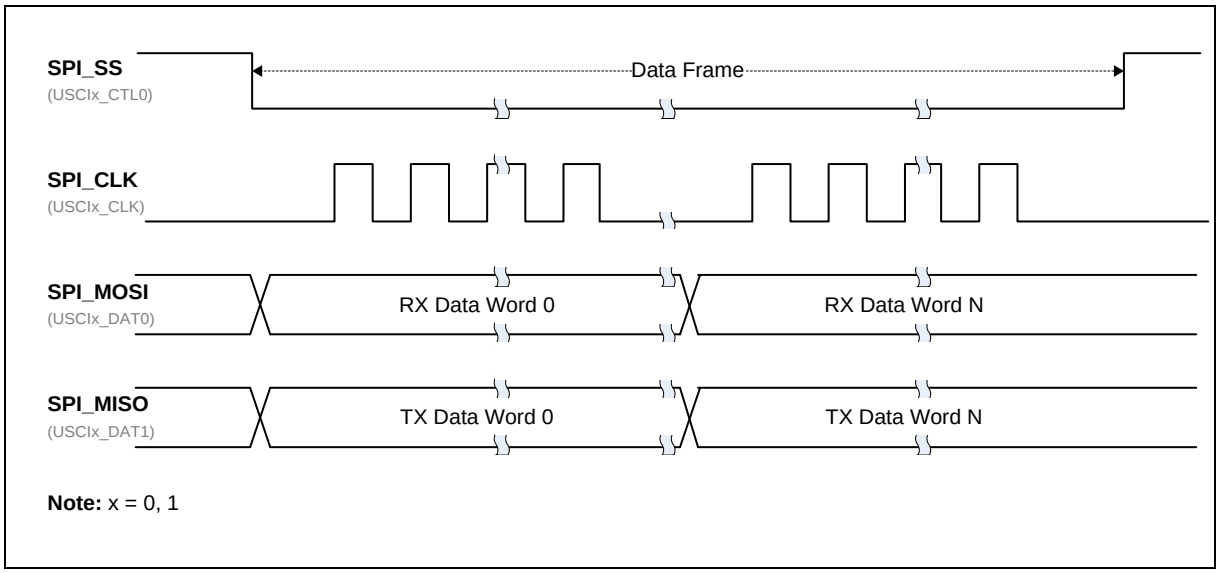


Figure 6.14-5 4-Wire Full-Duplex SPI Communication Signals (Slave Mode)

6.14.5.3 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 section 6.12.4 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 6.14-1 Serial Bus Clock Configuration

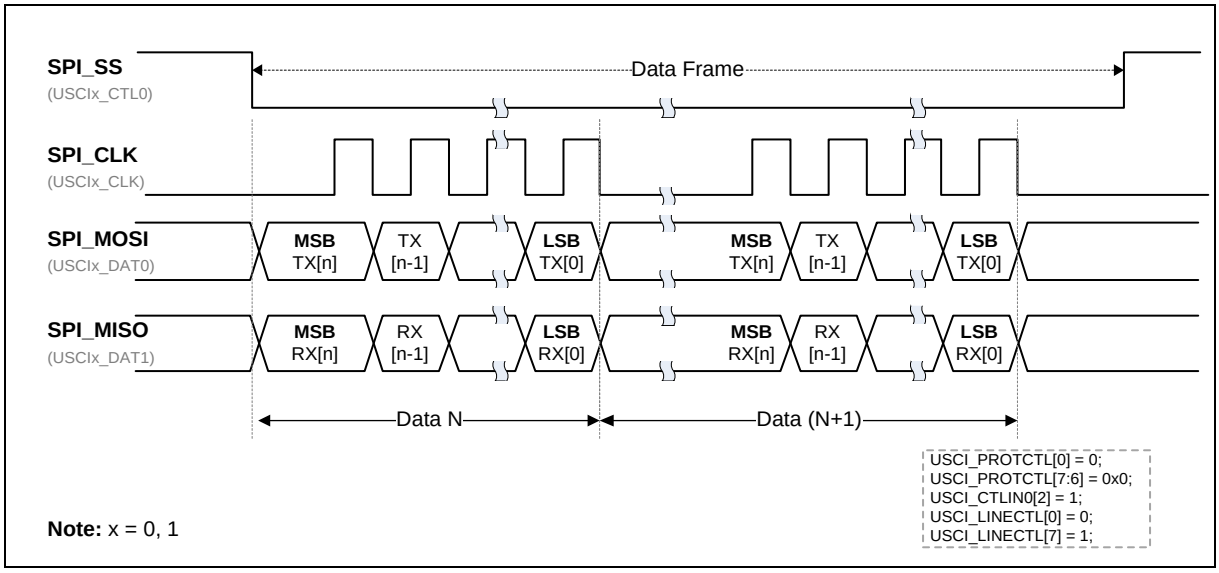


Figure 6.14-6 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x0)

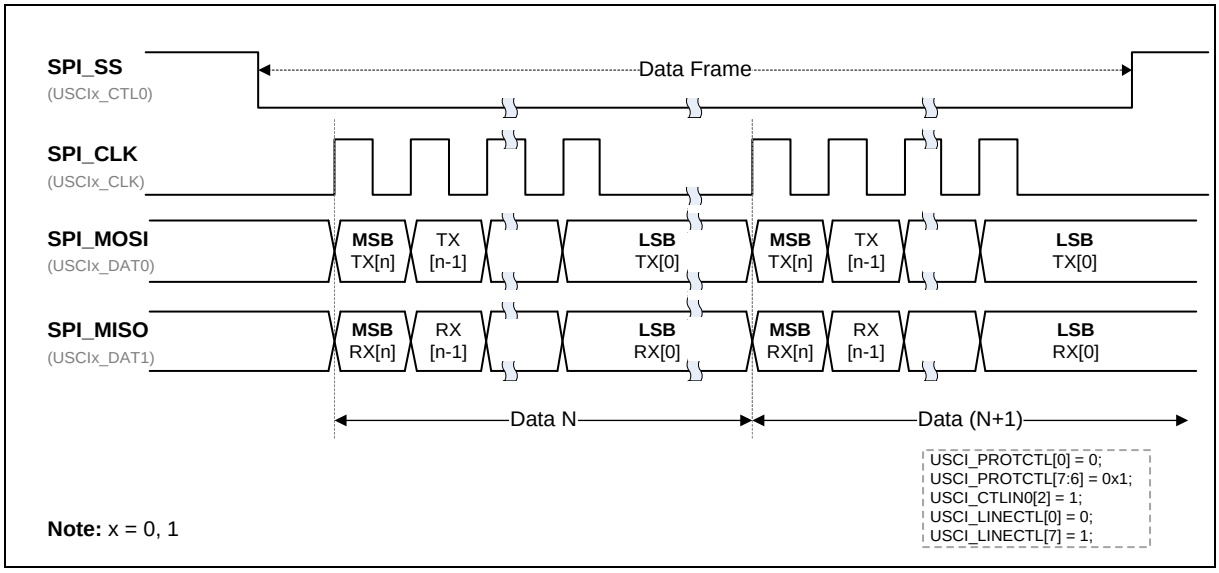


Figure 6.14-7 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x1)

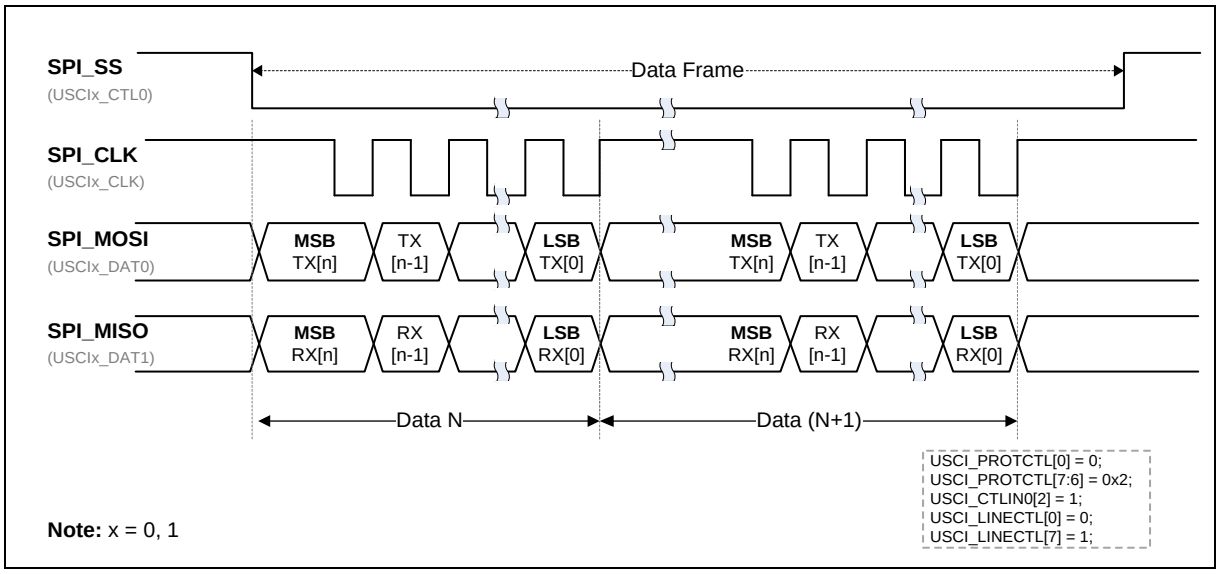


Figure 6.14-8 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x2)

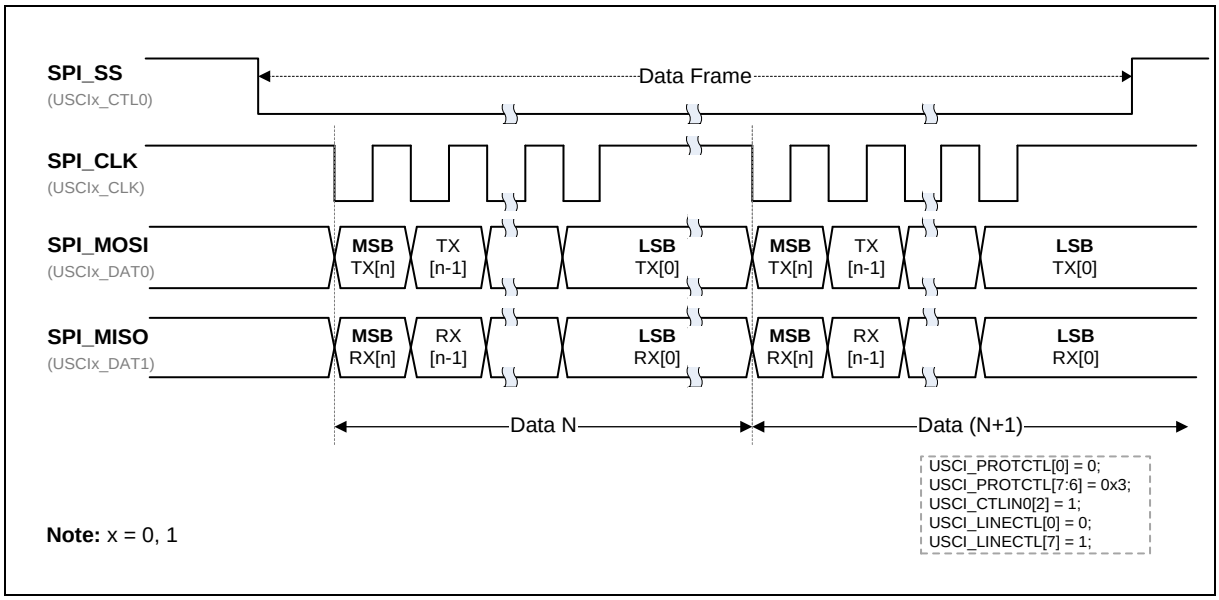


Figure 6.14-9 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x3)

6.14.5.4 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 keep inactive for at least 2 USCI peripheral clock cycles between two consecutive frames in order to correctly detect the end of a frame.

6.14.5.5 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.

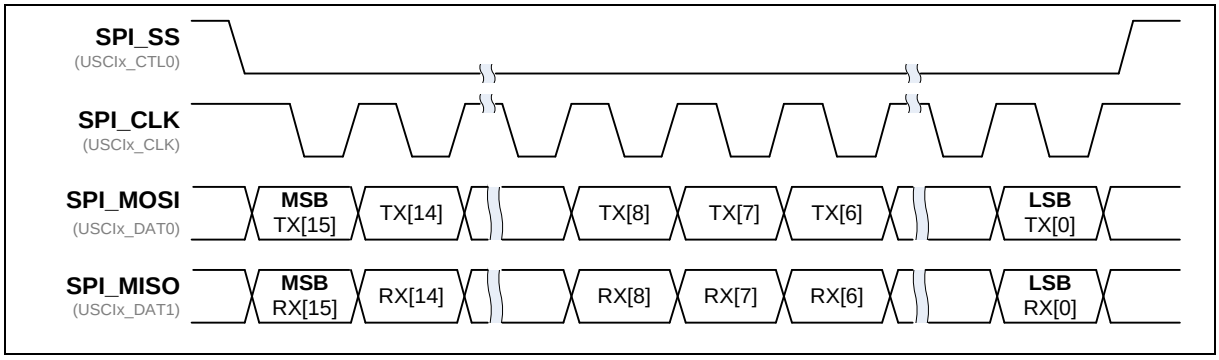


Figure 6.14-1016-bit Data Length in One Word Transaction with MSB First Format

6.14.5.6 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).

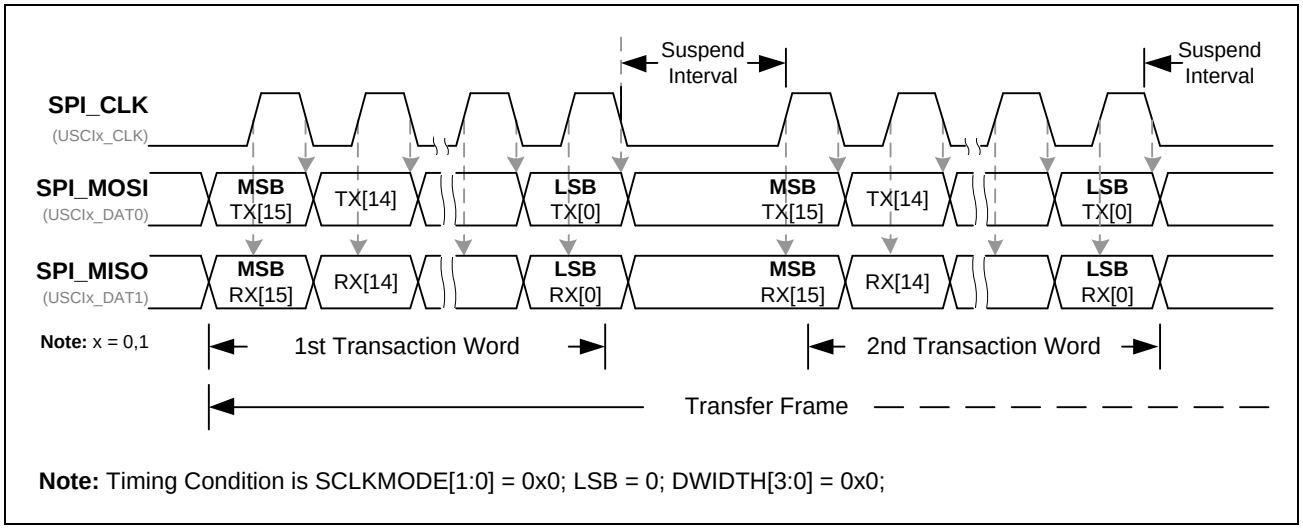


Figure 6.14-11 Word Suspend Interval between Two Transaction Words

6.14.5.7 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 great 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.

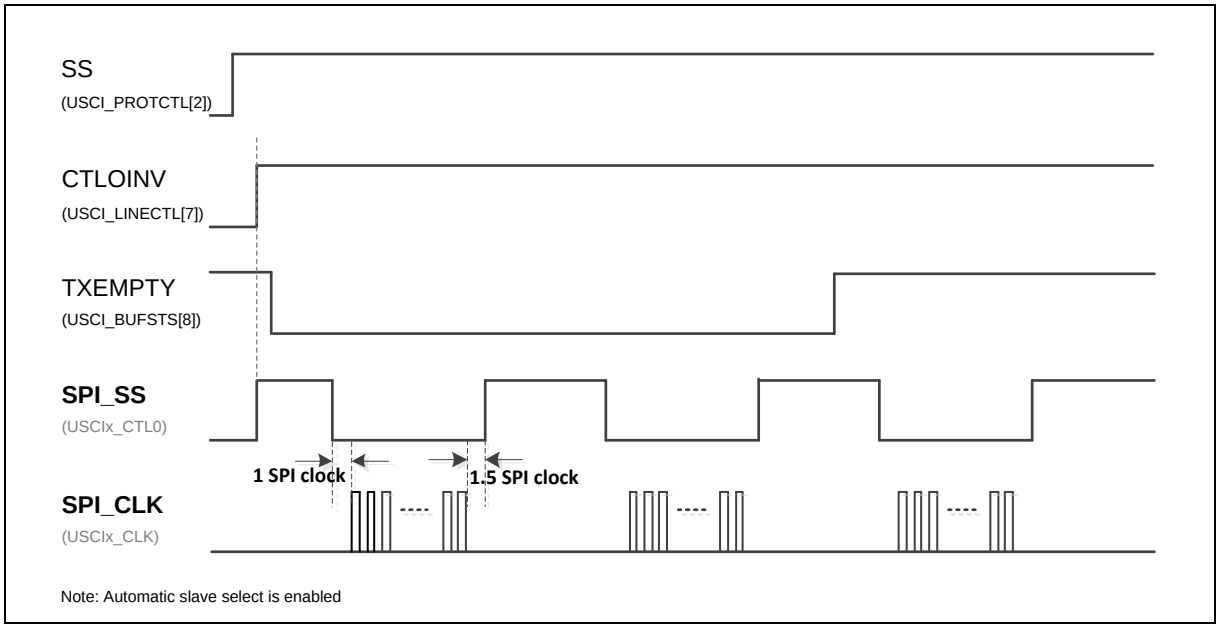


Figure 6.14-12 Auto Slave Select (SUSPITV ≧ 0x3)

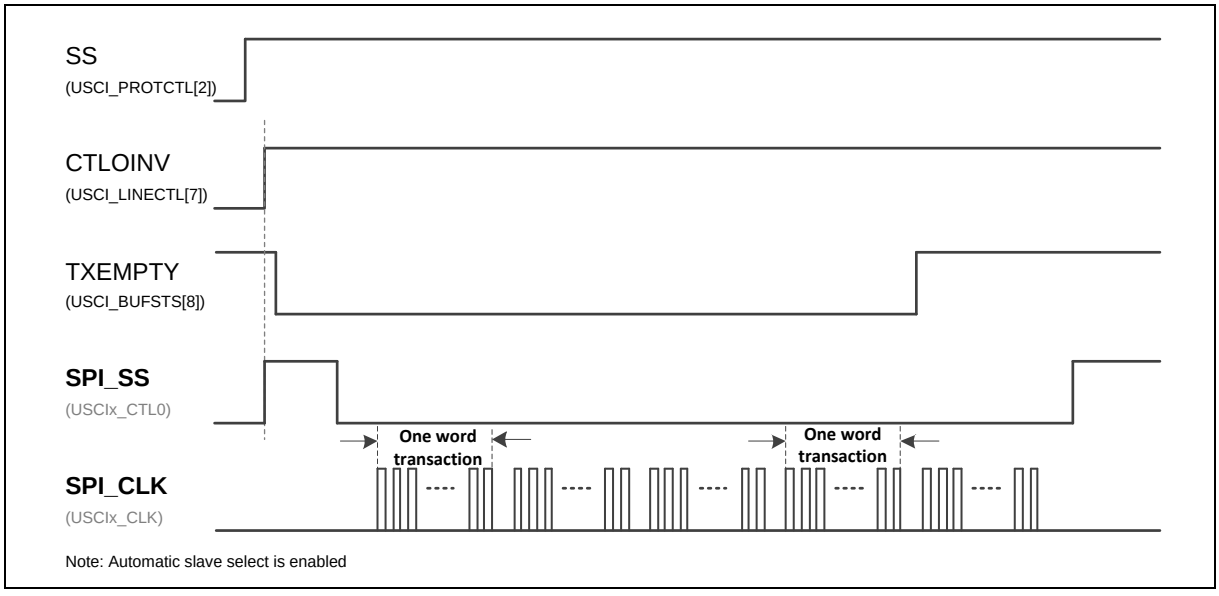


Figure 6.14-13 Auto Slave Select (SUSPITV < 0x3)

6.14.5.8 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.

6.14.5.9 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 detail 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 6.14-14 shows the one output data channel and one input data channel half-duplex transfer diagrams with the external device.

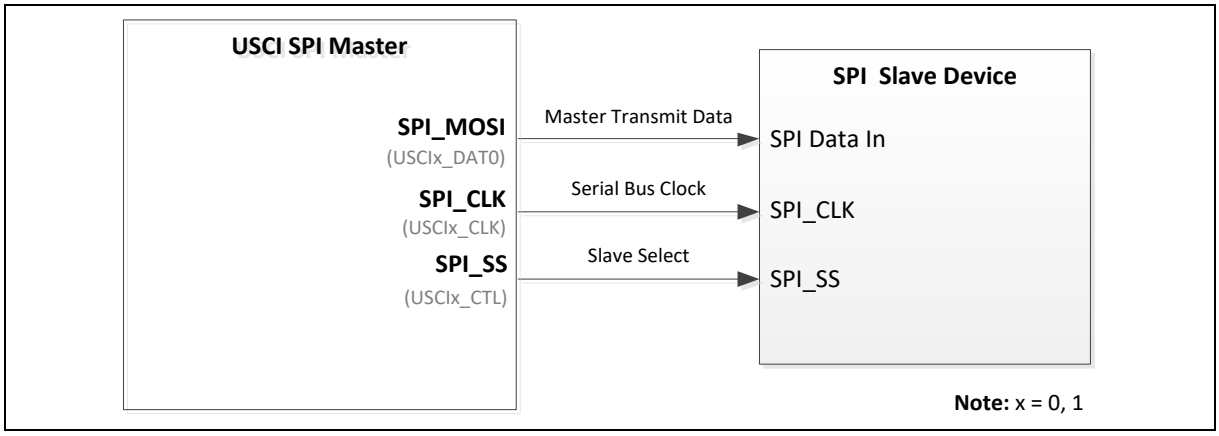


Figure 6.14-14 One Output Data Channel Half-duplex (SPI Master Mode)

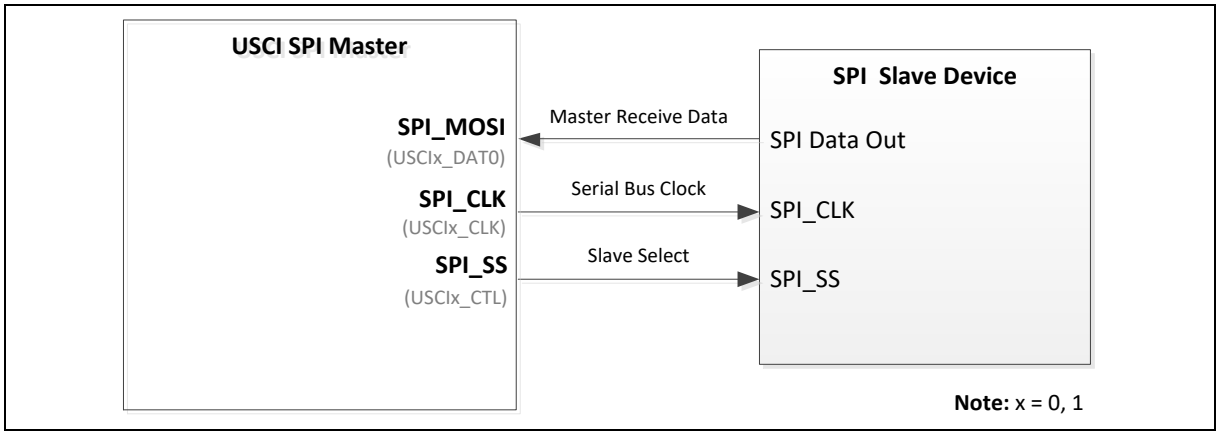


Figure 6.14-15 One Input Data Channel Half-duplex (SPI Master Mode)

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.

6.14.5.10 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]), are 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.

6.14.5.11 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.

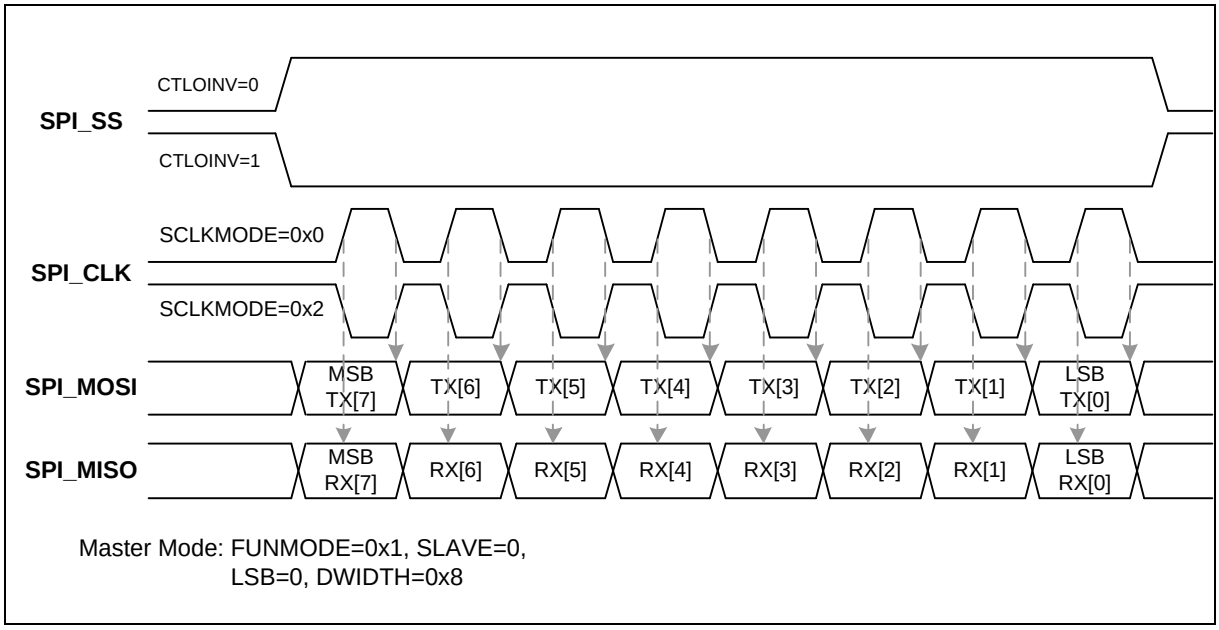


Figure 6.14-16 SPI Timing in Master Mode

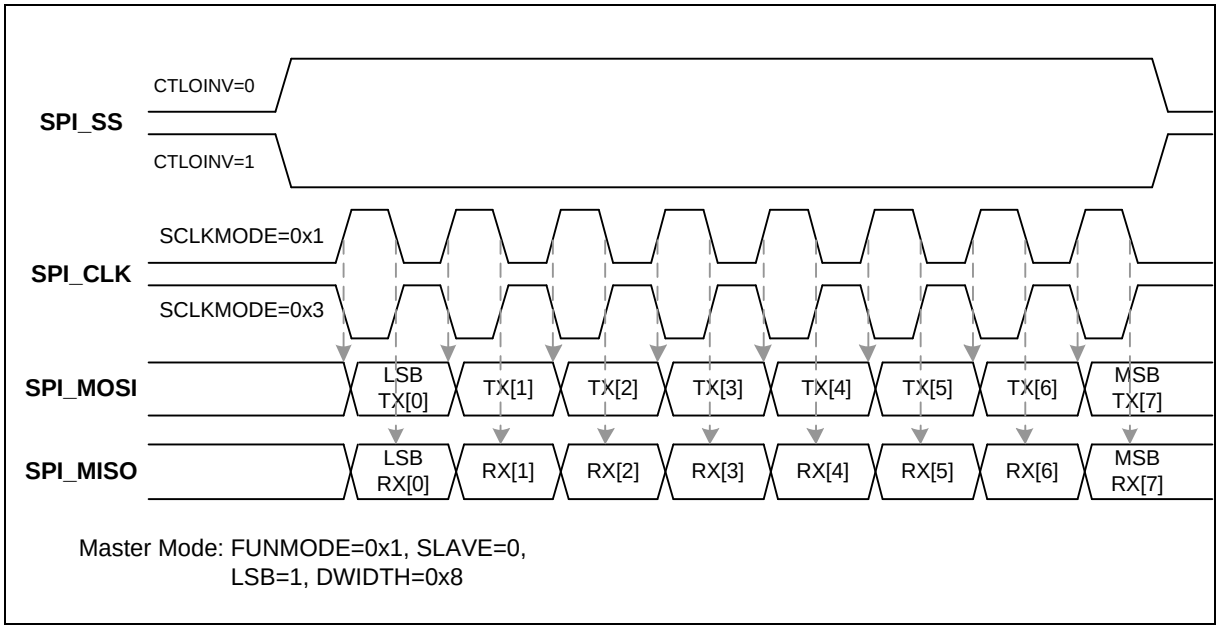


Figure 6.14-17 SPI Timing in Master Mode (Alternate Phase of Serial Bus Clock)

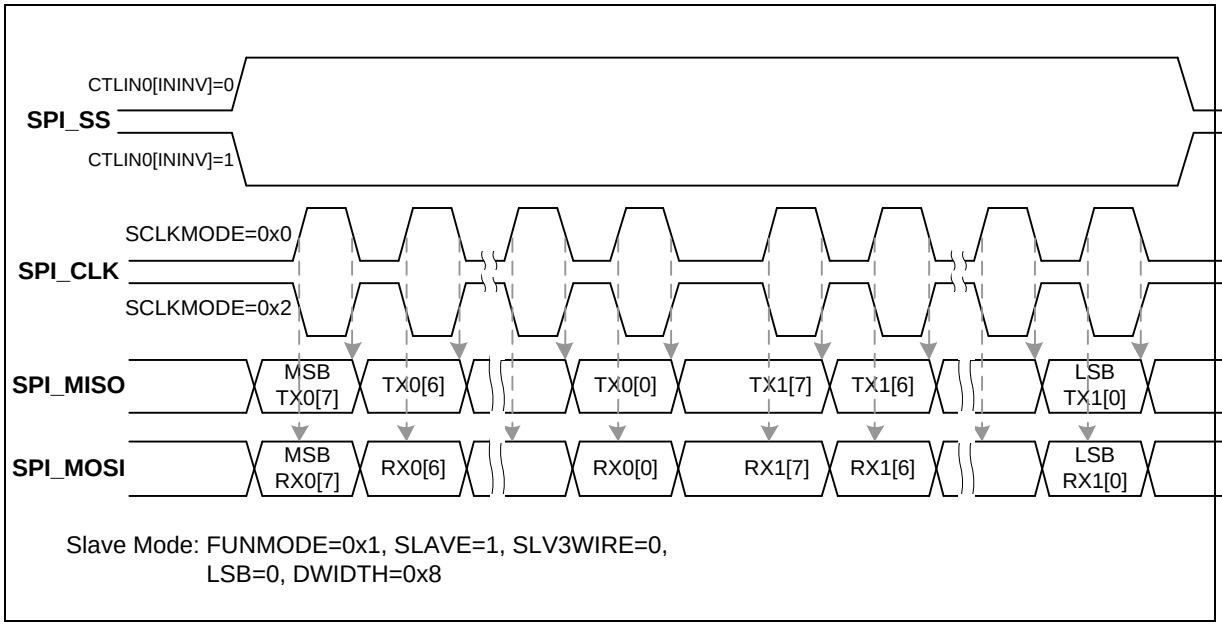


Figure 6.14-18 SPI Timing in Slave Mode

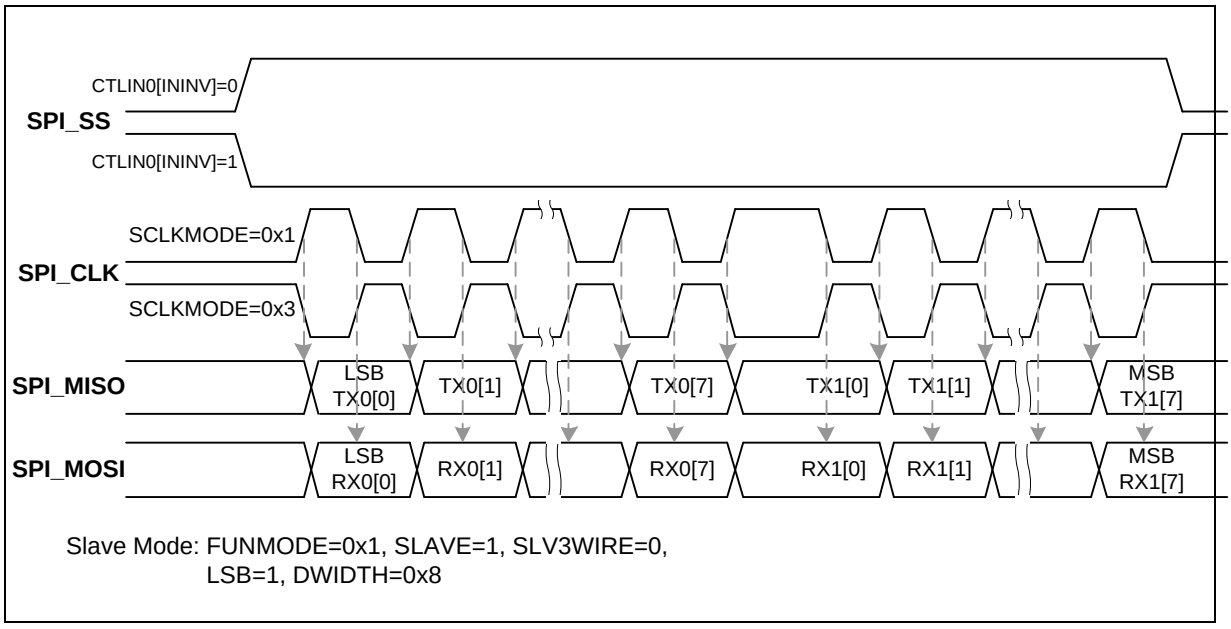


Figure 6.14-19 SPI Timing in Slave Mode (Alternate Phase of Serial Bus Clock)

6.14.5.12 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.

6.14.5.13 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.

6.14.6 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**				
USPI_CTL	USPIn_BA+0x00	R/W	USCI Control Register	0x0000_0000
USPI_INTEN	USPIn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000
USPI_BRGEN	USPIn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
USPI_DATIN0	USPIn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000
USPI_CTLIN0	USPIn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000
USPI_CLKIN	USPIn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000
USPI_LINECTL	USPIn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
USPI_TXDAT	USPIn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
USPI_RXDAT	USPIn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
USPI_BUFCTL	USPIn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000
USPI_BUFSTS	USPIn_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101
USPI_PDMACTL	USPIn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000
USPI_WKCTL	USPIn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
USPI_WKSTS	USPIn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
USPI_PROTCTL	USPIn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0300
USPI_PROTIEN	USPIn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
USPI_PROTSTS	USPIn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

6.14.7 Register Description

USCI Control Register (USPI_CTL)

Register	Offset	R/W	Description	Reset Value
USPI_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
USPI_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. **Note:** The receive finish event happens when hardware receives the last bit of RX data into shift data unit.
[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. **Note:** For SPI master mode, the receive start event happens when SPI master sends slave select active and spi clock to the external SPI slave. For SPI slave mode, the receive start event happens when slave select of SPI slave is active and spi clock of SPI slave is inputed from the external SPI master.
[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. **Note:** The transmit finish event happens when hardware sends the last bit of TX data from shift data unit.
[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. **Note:** The transmit start event happens when hardware starts to move TX data from data buffer to shift data unit.
[0]	**Reserved**	Reserved.

USCI Baud Rate Generator Register (USPI_BRGEN)

Register	Offset	R/W	Description	Reset Value
USPI_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} ($f_{DIV_CLK} = f_{PROT_CLK} / (CLKDIV+1)$).
[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
USPI_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, which is synchronized with PCLK, 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
USPI_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, which is synchronized with PCLK, 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
USPI_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, which is synchronized with PCLK, 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
USPI_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 pins. 0 = Data output values of USCIx_DAT0/1 pins are not inverted. 1 = Data output values of USCIx_DAT0/1 pins are 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
USPI_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
USPI_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 Transmitter/Receive Buffer Control Register (USPI_BUFCTL)

Register	Offset	R/W	Description	Reset Value
USPI_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
USPI_BUFSTS	USPIn_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				**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
USPI_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.
[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
USPI_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
USPI_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
USPI_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** This bit defines the transmitting data value of USCIx_DAT1 when no data is available for transferring. 0 = The output data value is 0 if TX under run event occurs. 1 = The output data value is 1 if TX under run event occurs. **Note:** for Slave
[27:26]	**Reserved**	Reserved.
[25:16]	**SLVTOCNT**	**Slave Mode Time-out Period** 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}. **Note:** Slave only
[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 values are 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** 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. **Note:** Master only
[7:6]	**SCLKMODE**	**Serial Bus Clock Mode** This bit field defines the SCLK idle status, data transmit, and data receive edge. MODE0 = The idle state of SPI clock is low level. Data is transmitted with falling edge and received with rising edge. MODE1 = The idle state of SPI clock is low level. Data is transmitted with rising edge and received with falling edge. MODE2 = The idle state of SPI clock is high level. Data is transmitted with rising edge and received with falling edge. 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** 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. **Note:** Master only
[2]	**SS**	**Slave Select Control** 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 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. **Note:** Master only
[1]	**SLV3WIRE**	**Slave 3-wire Mode Selection** 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. **Note:** Slave only
[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
USPI_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	
[31:5]	**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 = The Slave mode bit count error interrupt Disabled. 1 = The 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 = The Slave time-out interrupt Disabled. 1 = The 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
USPI_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 listing are the bus busy conditions: a. USPI_PROTCTL[31] = 1 and the TXEMPTY = 0. b. For SPI Master mode, the TXEMPTY = 1 but the current transaction is not finished yet. c. 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. d. 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** 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.

		Note: Slave only
[8]	**SSINAIF**	**Slave Select Inactive Interrupt Flag** 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. **Note:** for Slave only
[7]	**Reserved**	Reserved.
[6]	**SLVBEIF**	**Slave Bit Count Error Interrupt Flag** 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. If the transmit/receive data bit count does not match the setting of DWIDTH (USPI_LINECTL[11:8]), bit count error event occurs. **Note:** for Slave only
[5]	**SLVTOIF**	**Slave Time-out Interrupt Flag** 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 **Note:** for Slave only
[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. The receive end event happens when hardware receives the last bit of RX data into shift data unit.
[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. For SPI master mode, the receive start event happens when SPI master sends slave select active and spi clock to the external SPI slave. For SPI slave mode, the receive start event happens when slave select of SPI slave is active and spi clock of SPI slave is inputed from the external SPI master.
[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. The transmit end event happens when hardware sends the last bit of TX data from shift data unit.
[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. The transmit start event happens when hardware starts to move TX data from data buffer to shift data unit.
[0]	**Reserved**	Reserved.

6.15 USCI - I²C Mode

6.15.1 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 6.15-1 for more detailed I²C BUS Timing.

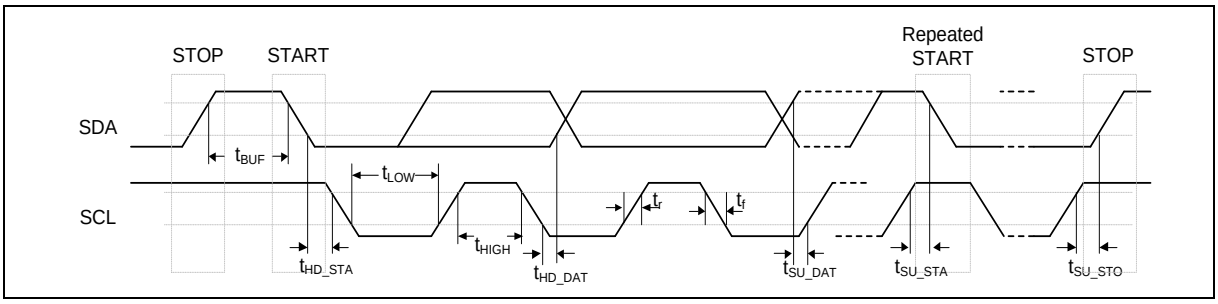


Figure 6.15-1 I²C Bus Timing

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 enabling 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.

6.15.2 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 data toggle or address match
- Supports setup/hold time programmable
- Supports multiple address recognition (two slave address with mask option)

6.15.3 Block Diagram

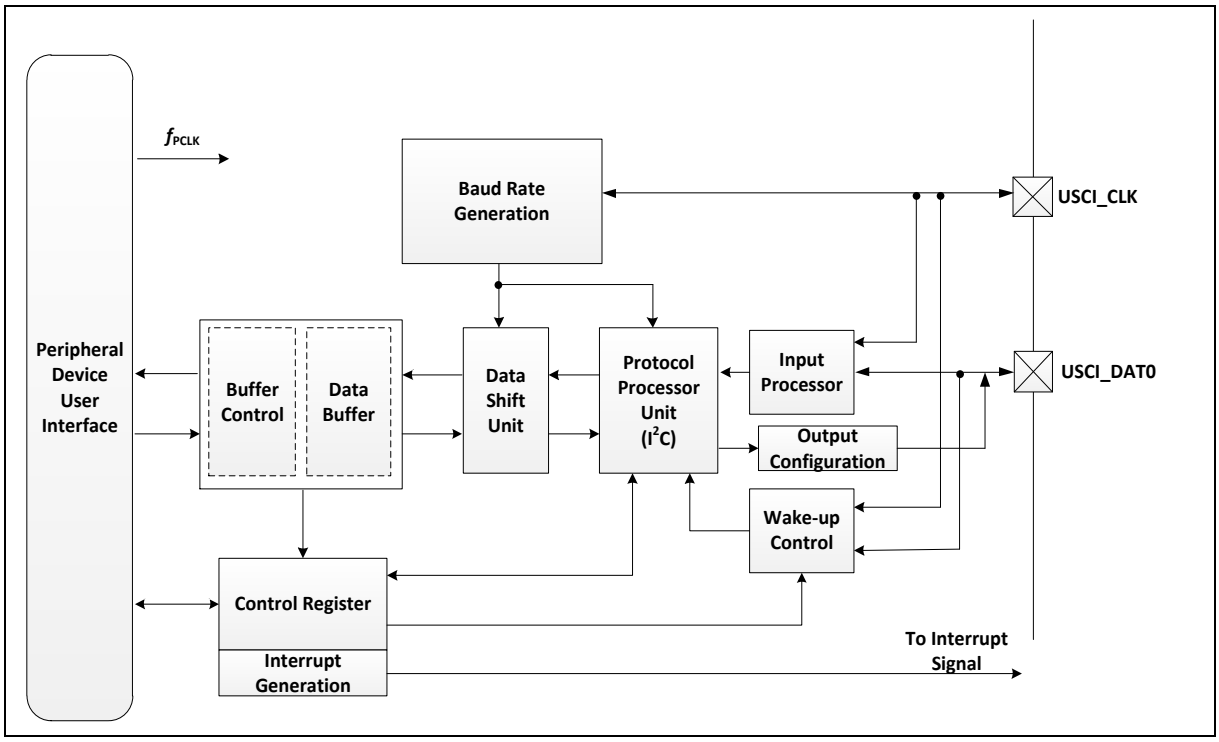


Figure 6.15-2 USCI I²C Mode Block Diagram

6.15.4 Basic Configuration

6.15.4.1 USCI0 I²C Basic Configurations

The basic configurations of USCI0_I2C are as follows:

- Clock Source Configuration
  - Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK1[8]).
  - Enable USCI0_I2C function UI2C_CTL[2:0]) register, UI2C_CTL[2:0]=3'b100
- Reset Configuration
  - Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
USCI0	USCI0_CLK	PA.5, PA.4, PC.5, PC.4, PC.3, PC.6, PC.7, PB.7, PB.6, PB.5, PB.4, PC.2, PC.1, PC.0, PA.2, PA.1, PA.0	MFP10
		PD.4	MFP4
	USCI0_DAT0	PA.5, PA.4, PC.5, PC.4, PC.3, PC.6, PC.7, PB.7, PB.6, PB.5, PB.4, PC.2, PC.1, PC.0, PA.2, PA.1, PA.0	MFP11
		PD.5	MFP4

6.15.4.2 Functional Description USCI1 I²C Basic Configurations

The basic configurations of USCI1_I2C are as follows:

- Clock Source Configuration
  - Enable USCI1 peripheral clock in USCI1CKEN (CLK_APBCLK1[9]).
  - Enable USCI1_I2C function UI2C_CTL[2:0]) register, UI2C_CTL[2:0]=3'b100
- Reset Configuration
  - Reset USCI1 controller in USCI1RST (SYS_IPRST2[9]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
USCI1	USCI1_CLK	PA.5, PA.4, PC.5, PC.4, PC.3, PC.6, PC.7, PB.7, PB.6, PB.5, PB.4, PC.2, PC.1, PC.0, PA.2, PA.1, PA.0	MFP15
		PD.0	MFP4
	USCI1_DAT0	PA.5, PA.4, PC.5, PC.4, PC.3, PC.6, PC.7, PB.7, PB.6, PB.5, PB.4, PC.2, PC.1, PC.0, PA.2, PA.1, PA.0	MFP16
		PD.1	MFP4

6.15.5 START or Repeated START Signal

Figure 6.15-3 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

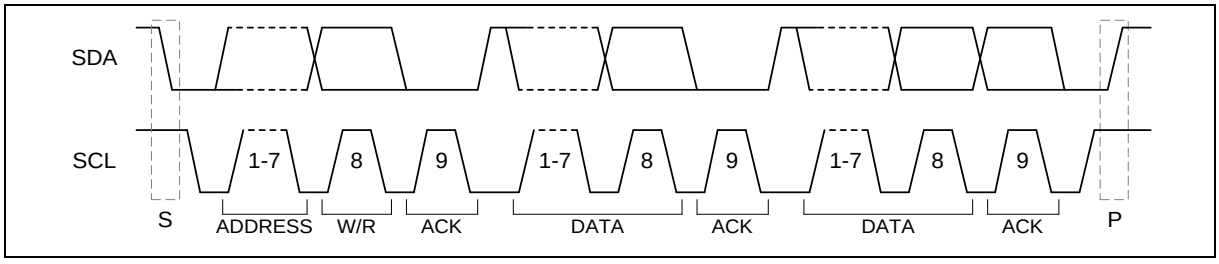


Figure 6.15-3 I²C Protocol

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.

6.15.5.1 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 6.15-4 shows the waveform of START, Repeat START and STOP.

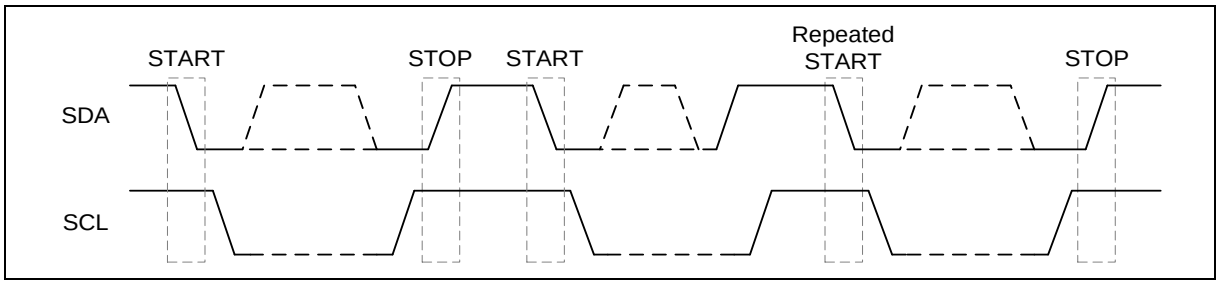


Figure 6.15-4 START and STOP Conditions

6.15.5.2 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 I^{2}C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.
- The I^{2}C 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 I^{2}C 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 I^{2}C port is equipped multiple address recognition with one address mask registers UI2C_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.

6.15.5.3 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.

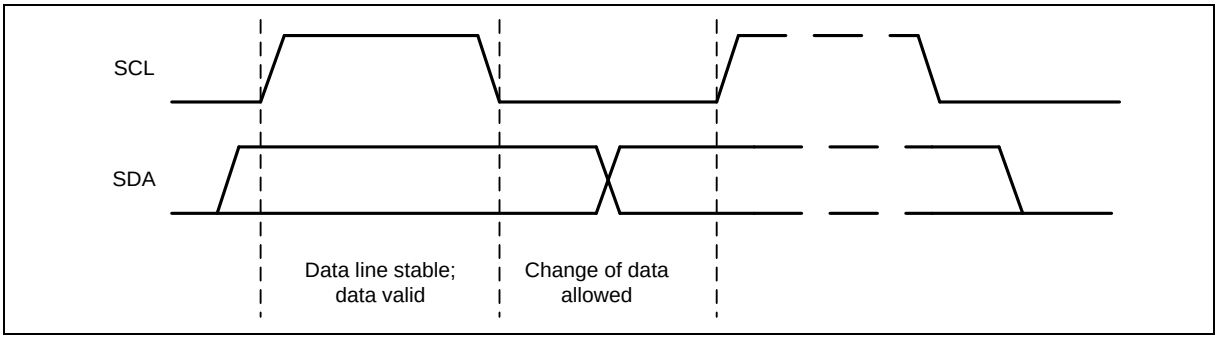


Figure 6.15-5 Bit Transfer on the I²C Bus

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.

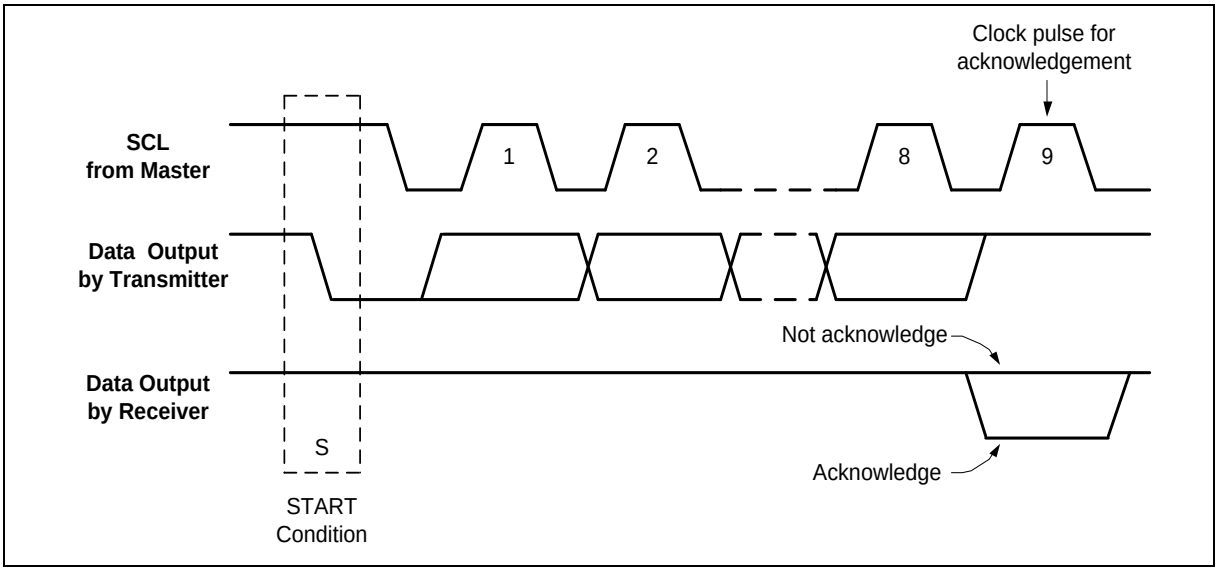


Figure 6.15-6 Acknowledge on the I²C Bus

6.15.5.4 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}
- PDSCNT (UI2C_BRGEN [9:8])

  to define the length of a data sample time (division of f_{REF_CLK} by 1, 2, 3, or 4)
- DSCNT (UI2C_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 = 2'b00 ($f_{SAMP_CLK} = f_{DIV_CLK}$). Under these conditions, the baud

rate is given by:

$$f_{I2C} = f_{REF_CLK} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

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:

$$f_{I2C} = \frac{f_{REF_CLK}}{2} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

If SPCLKSEL = 2'b10 (f_{SAMP_CLK} = f_{SCLK}), and RCLKSEL = 0 (f_{REF_CLK2}= f_{PCLK}), PTCLKSEL = 0 (f_{PROT_CLK} = f_{REF_CLK}). The baud rate is given by:

$$f_{I2C} = f_{REF_CLK} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{2} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

6.15.5.5 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]).

6.15.5.6 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 6.15-7 describes master1 data and master2 data are compete arbitration.

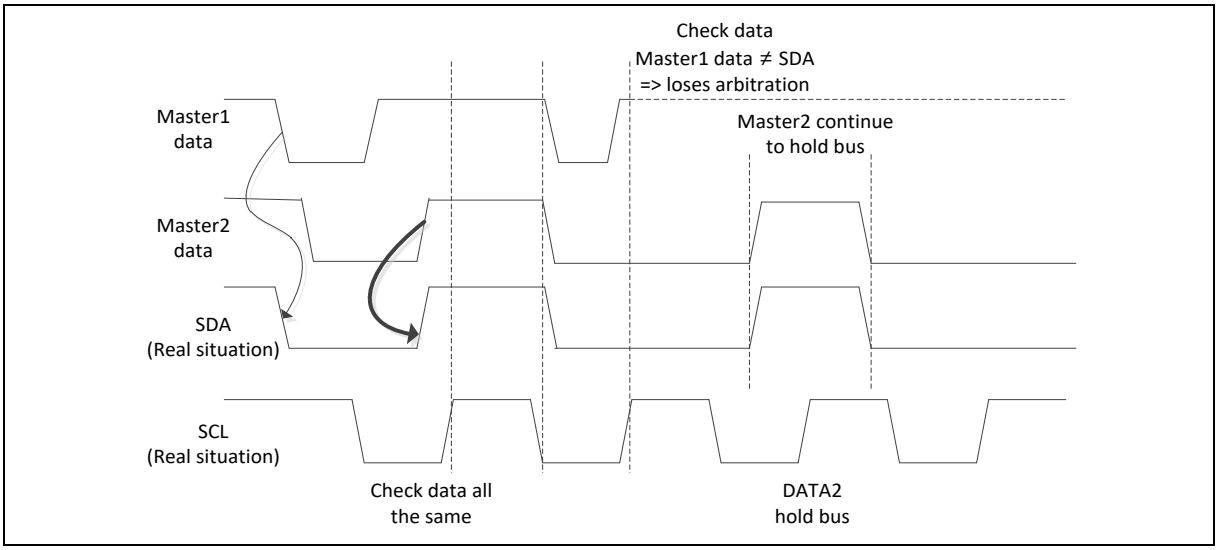


Figure 6.15-7 Arbitration Lost

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.

6.15.5.7 Transmission Chain

The I²C 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}.

6.15.5.8 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.

6.15.5.9 I²C Protocol Interrupt Events

The following protocol-related events are generated in I²C 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]))

6.15.5.10 Operating the I2C

To operate the I²C protocol, the following issues have to be considered:

Select I²C Mode

It is recommended to configure all parameters of the I²C that do not change during run time while

FUNMODE (UI2C_CTL [2:0]) = 000B. The I²C control flow has to be done while FUNMODE (UI2C_CTL [2:0]) = 000B to avoid unintended edges of the input signals and the I²C 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 I²C bus lines.

Note: The step to enable the alternate output port functions should only be done after the I²C 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 I²C 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 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(UI2C_PROTCTL[1]) 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, 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 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 6.15-8 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.

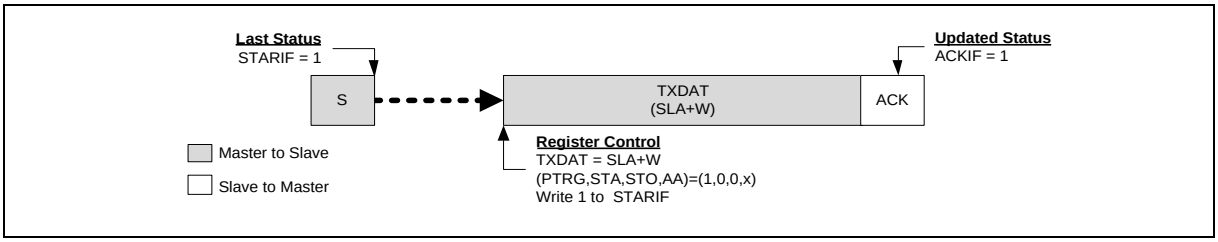


Figure 6.15-8 Control I²C Bus According to Current I²C Status

Data Transfer on the I²C Bus

Figure 6.15-9 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.

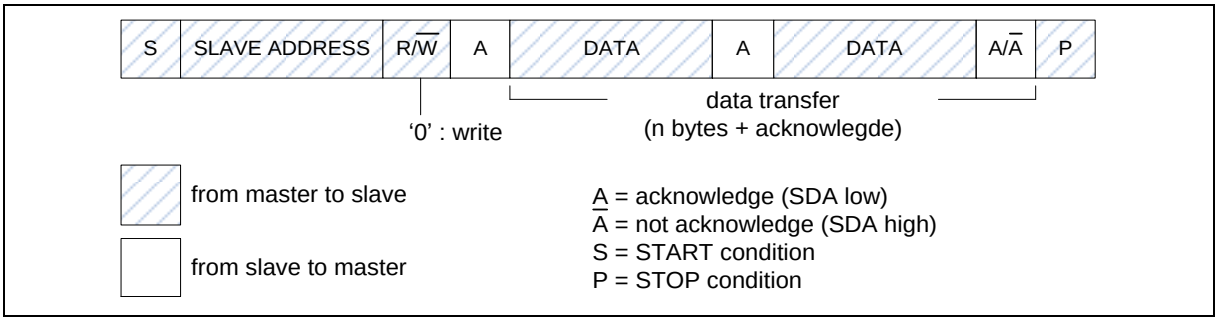


Figure 6.15-9 Master Transmits Data to Slave with a 7-bit Address

Figure 6.15-10 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.

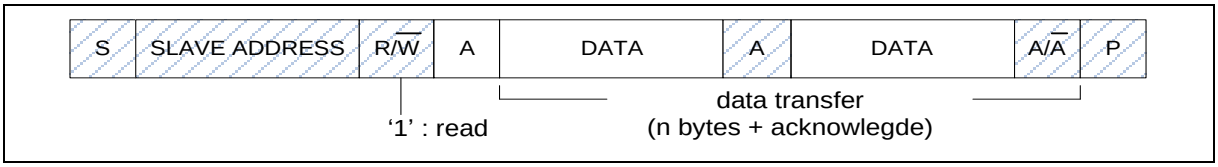


Figure 6.15-10 Master Reads Data from Slave with a 7-bit Address

Figure 6.15-11 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.

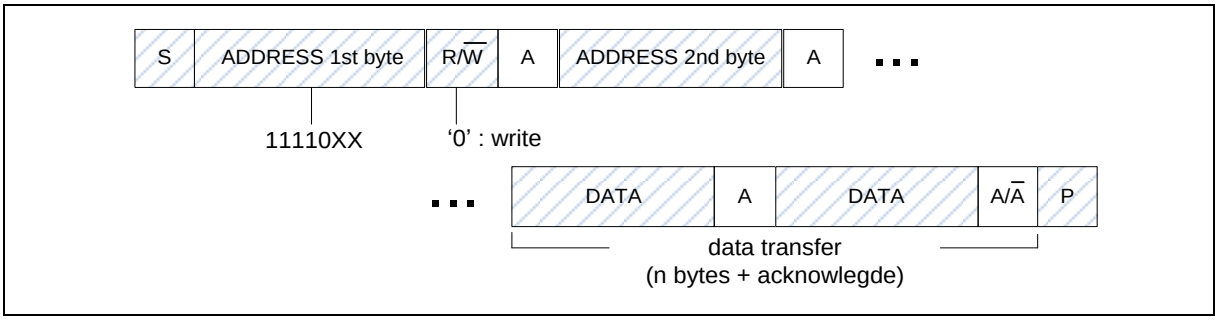


Figure 6.15-11 Master Transmits Data to Slave by 10-bit Address

Figure 6.15-12 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.

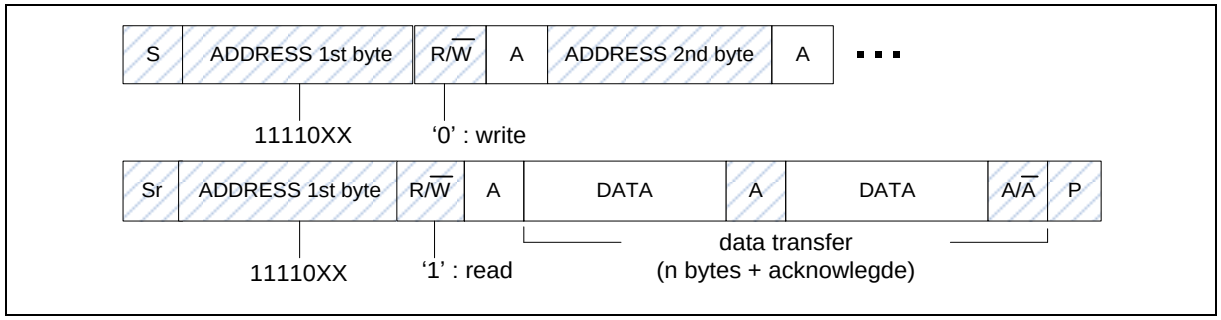


Figure 6.15-12 Master Reads Data from Slave by 10-bit Address

Master Mode

In Figure 6.15-13 and Figure 6.15-14, 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 mode (Figure 6.15-13) or Master receiver mode (Figure 6.15-14) 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 I²C protocol.

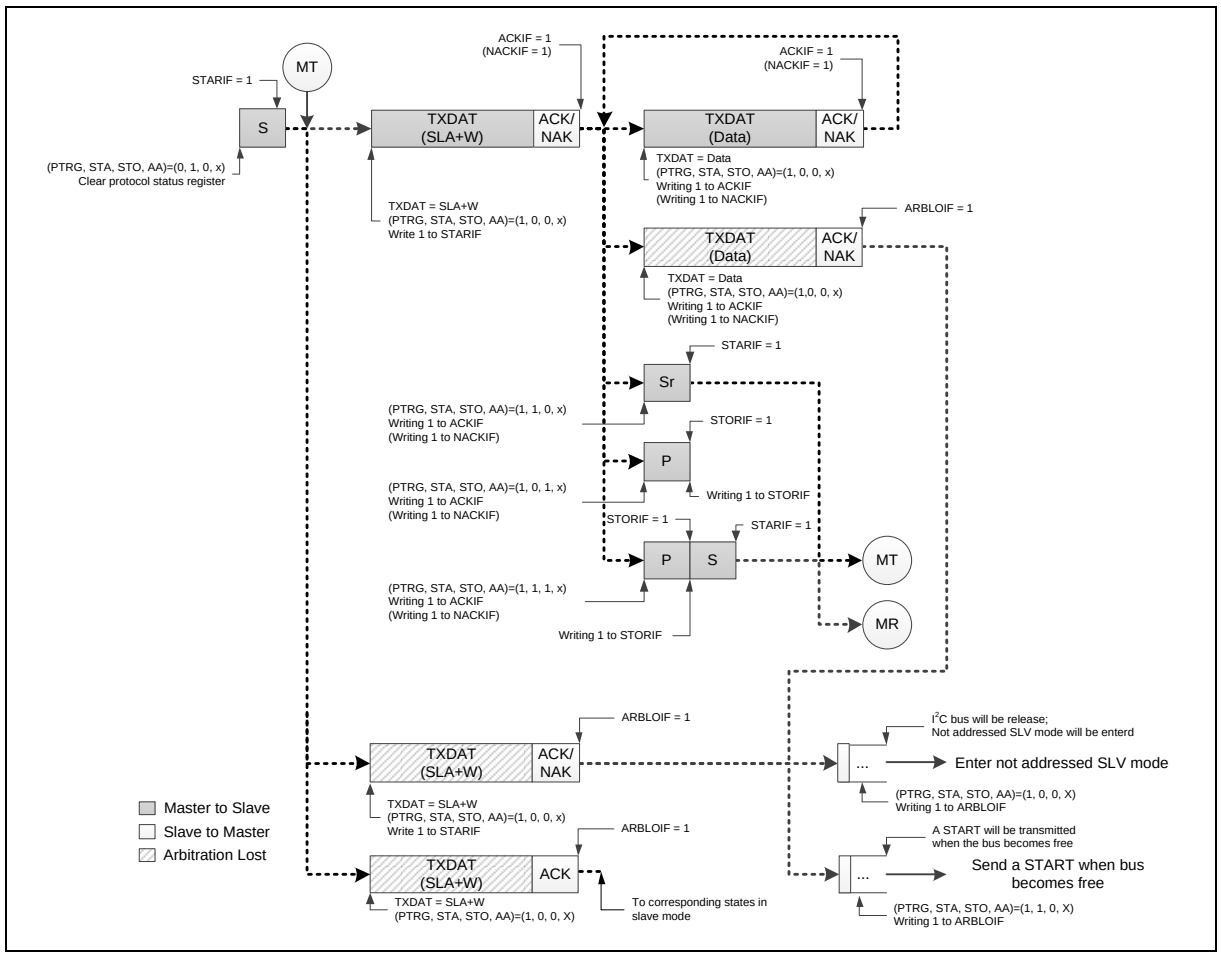


Figure 6.15-13 Master Transmitter Mode Control Flow with 7-bit Address

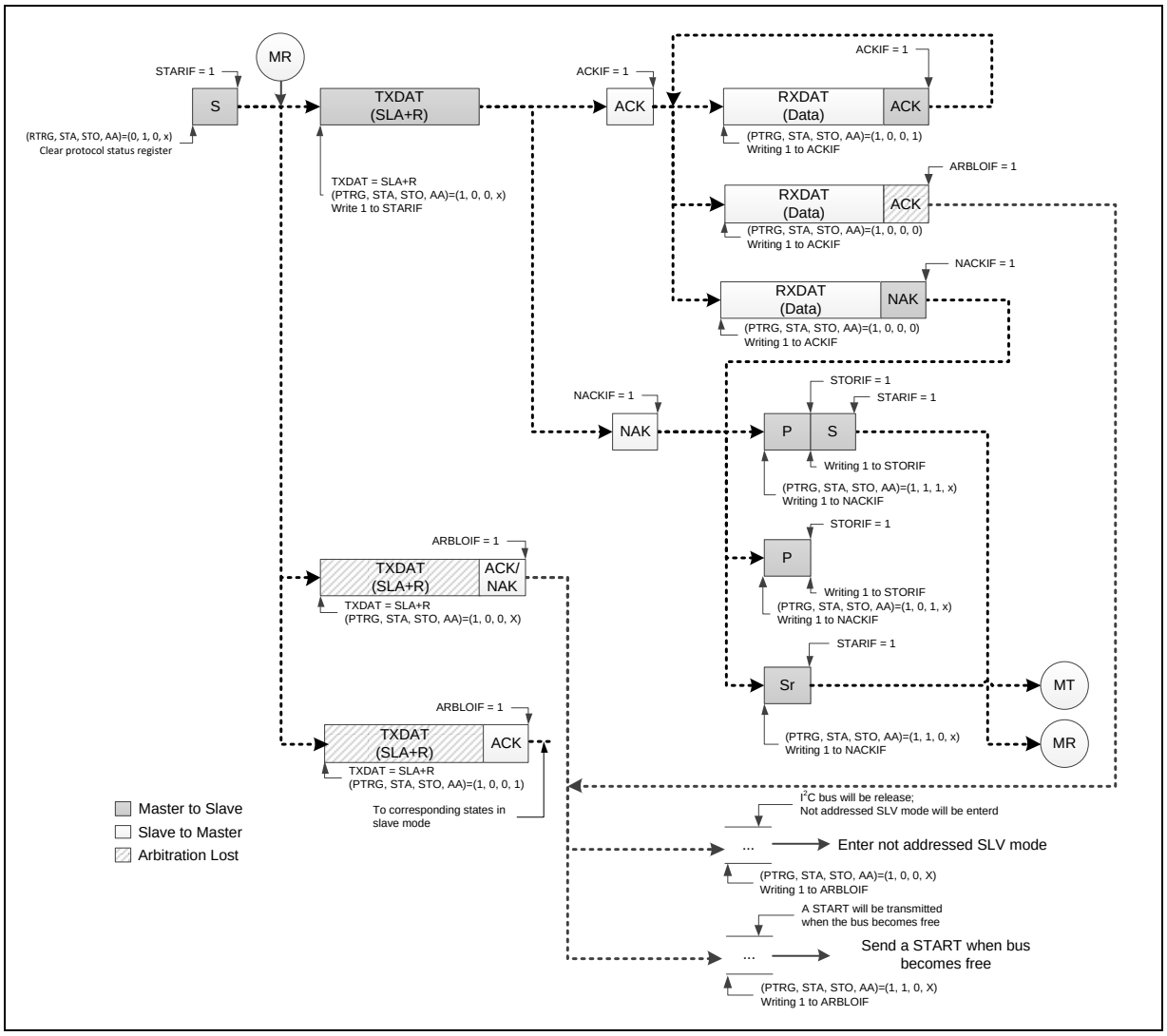


Figure 6.15-14 Master Receiver Mode Control Flow with 7-bit Address

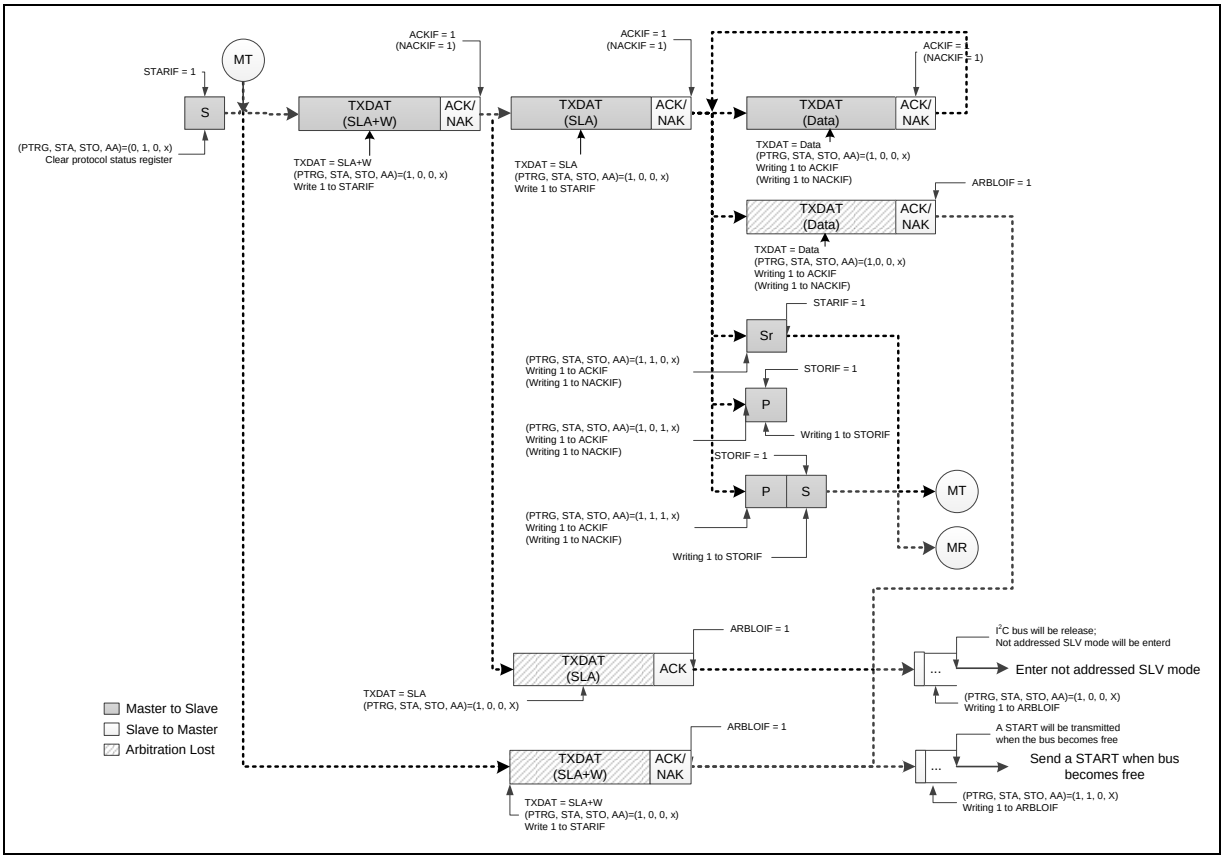


Figure 6.15-15 Master Transmitter Mode Control Flow with 10-bit Address

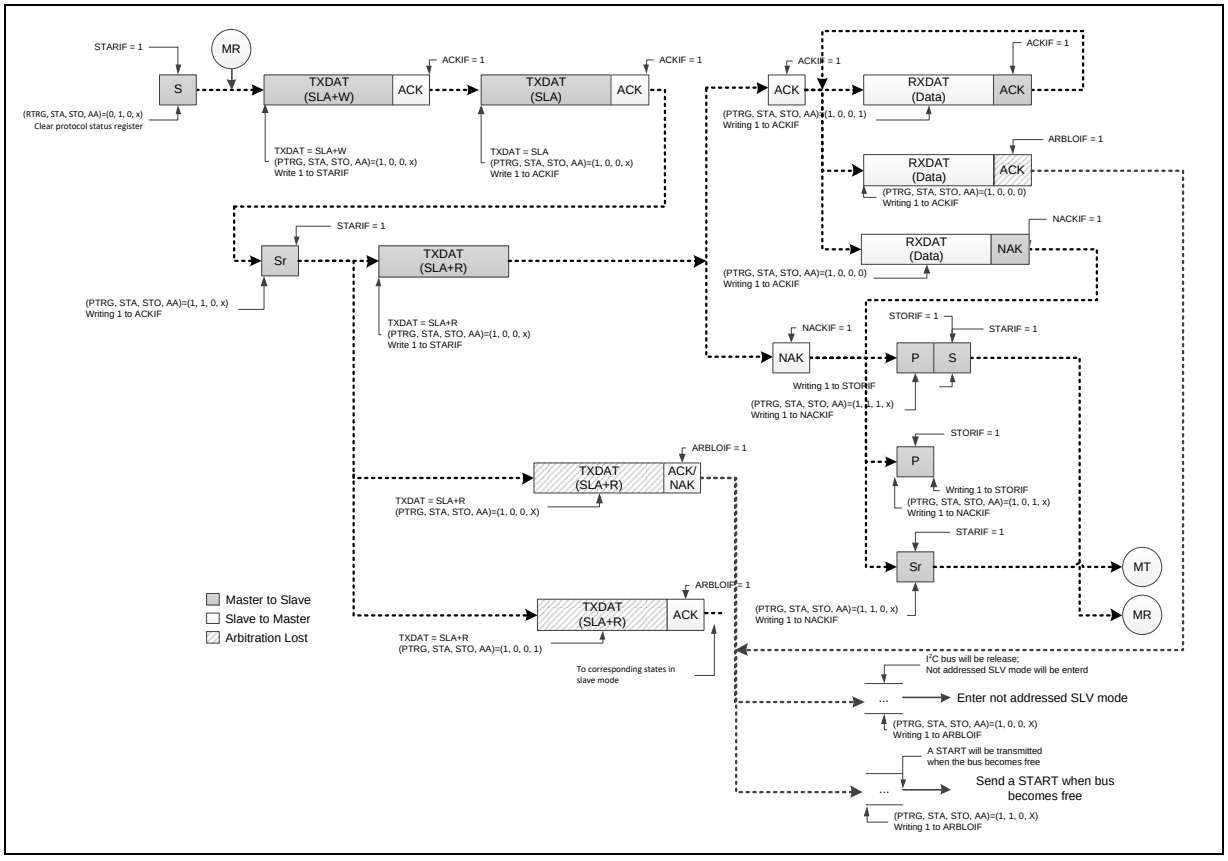


Figure 6.15-16 Master Recevier Mode Control Flow with 10-bit Address

If the I²C is in Master mode and gets arbitration lost, the bit of ARBLOIF (UI2C_PROTSTS [11]) will be set. User may writing 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 writing 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 6.15-17 shows all the possible flow for I²C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.15-17 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.

The I²C controller supports two slave address match flags, are ADMAT0 and ADMAT1 on UI2C_ADMAT[1:0] register. Every control register represent which address is used and set 1 to inform software.

Note: During I²C communication, the SCL clock will be released when writing '1' to PTRG (UI2C_PROTCTL [5]) in Slave mode.

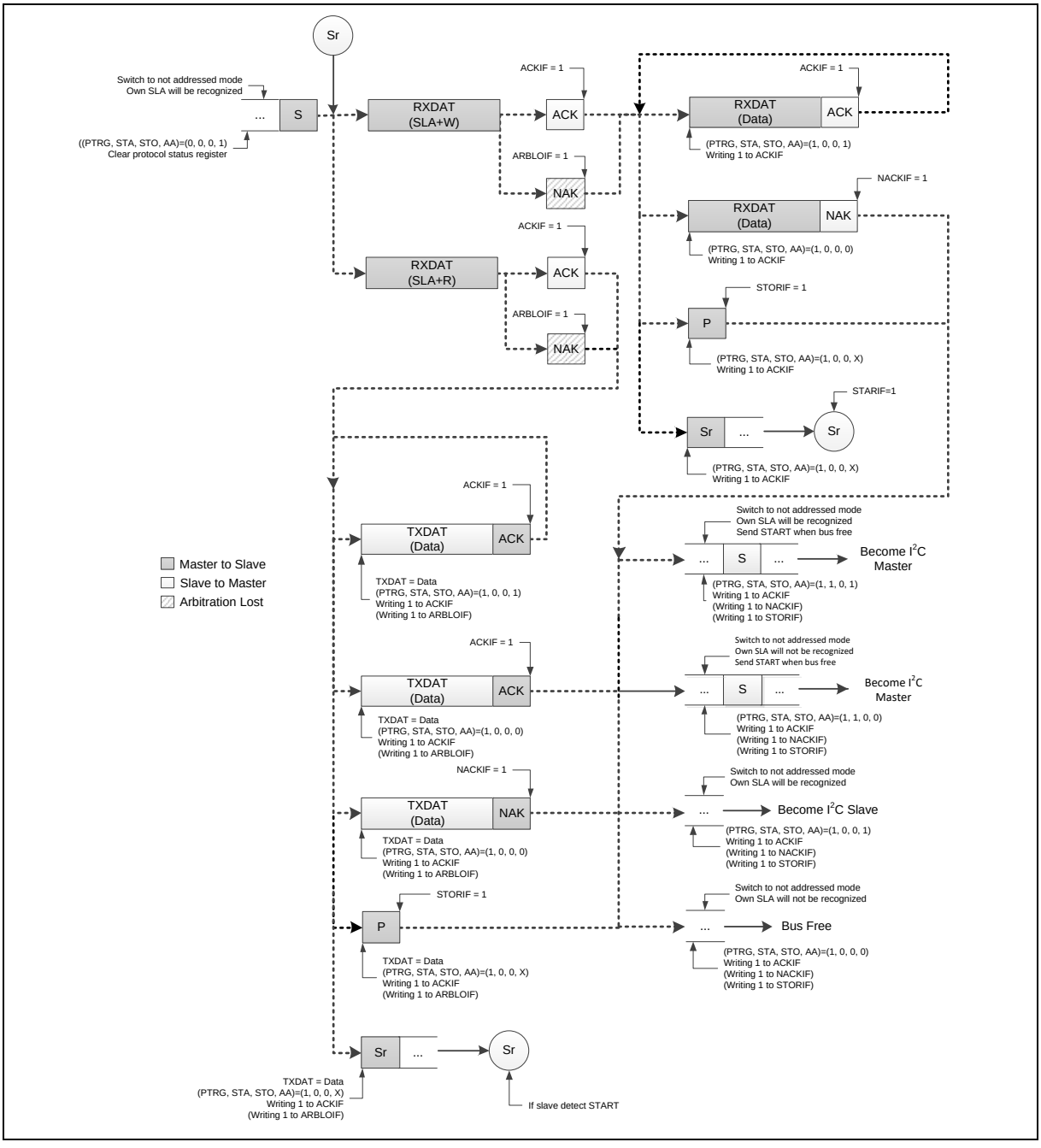


Figure 6.15-17 Save Mode Control Flow with 7-bit Address

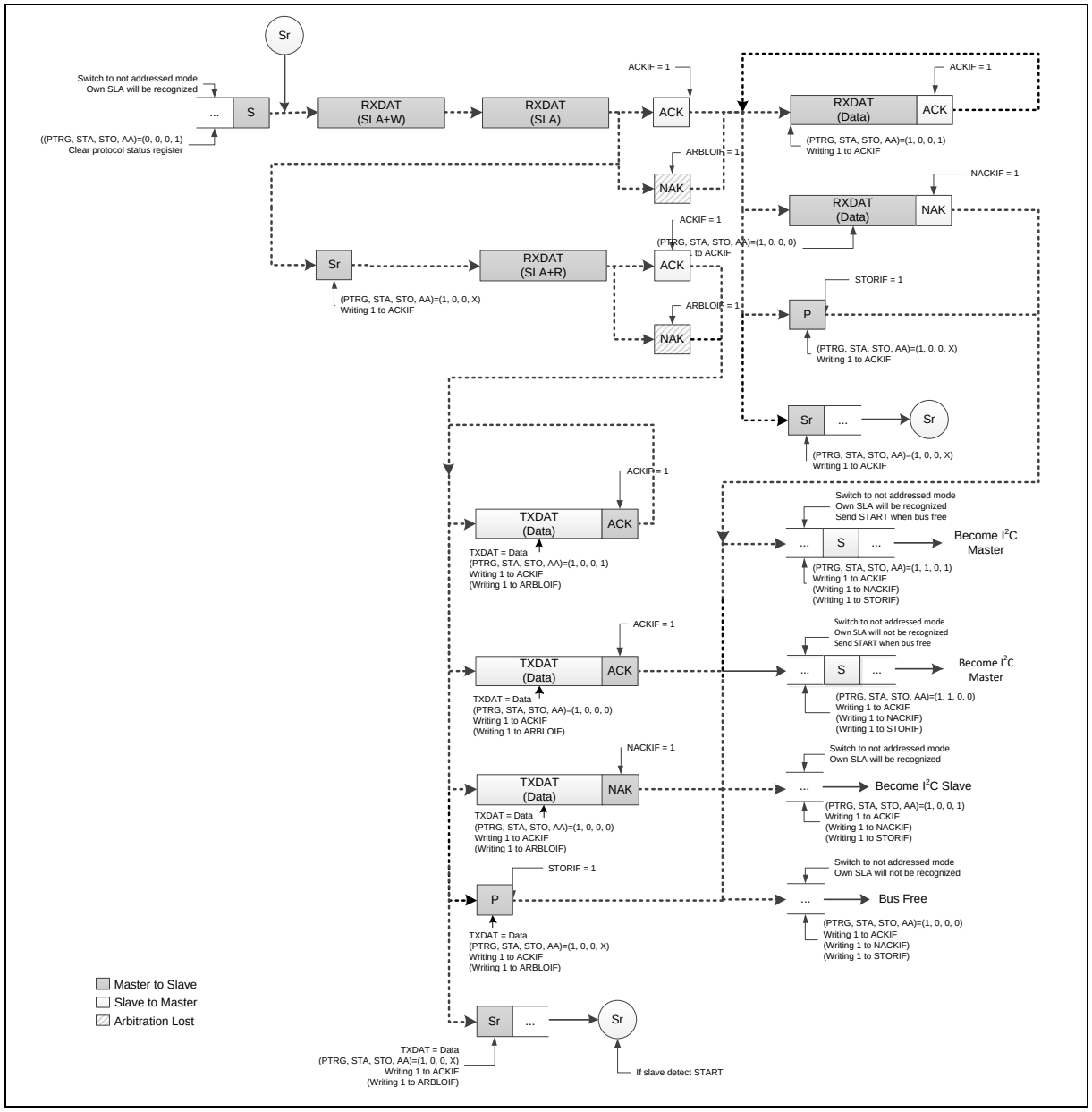


Figure 6.15-18 Save Mode Control Flow with 10-bit Address

If I^{2}C 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 I^{2}C signal or address from master. At this status, I^{2}C 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 I^{2}C 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.

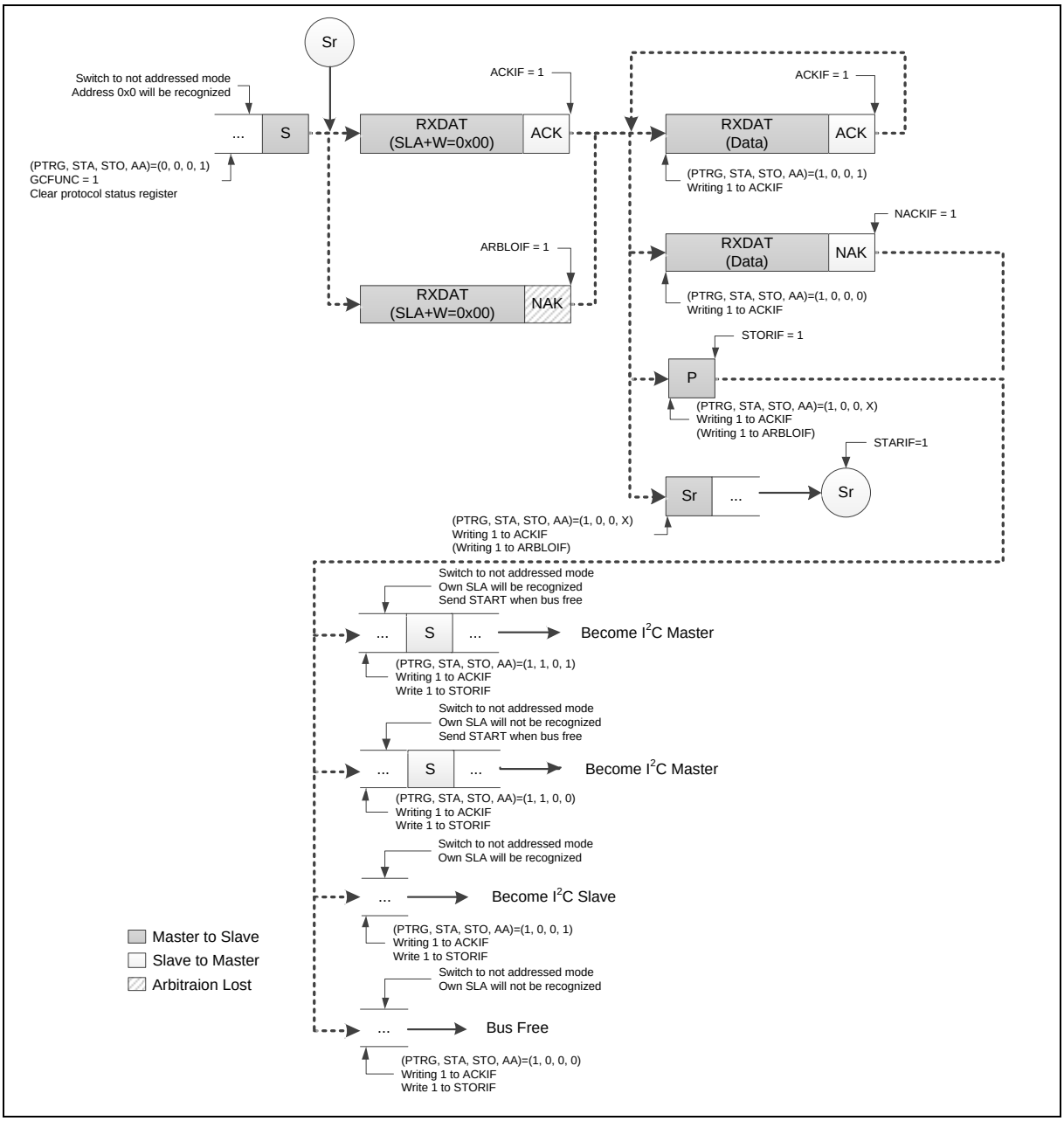


Figure 6.15-19 GC Mode with 7-bit Address

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, 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 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 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. And once hold time configuration 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 6.15-1 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 6.15-1 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.

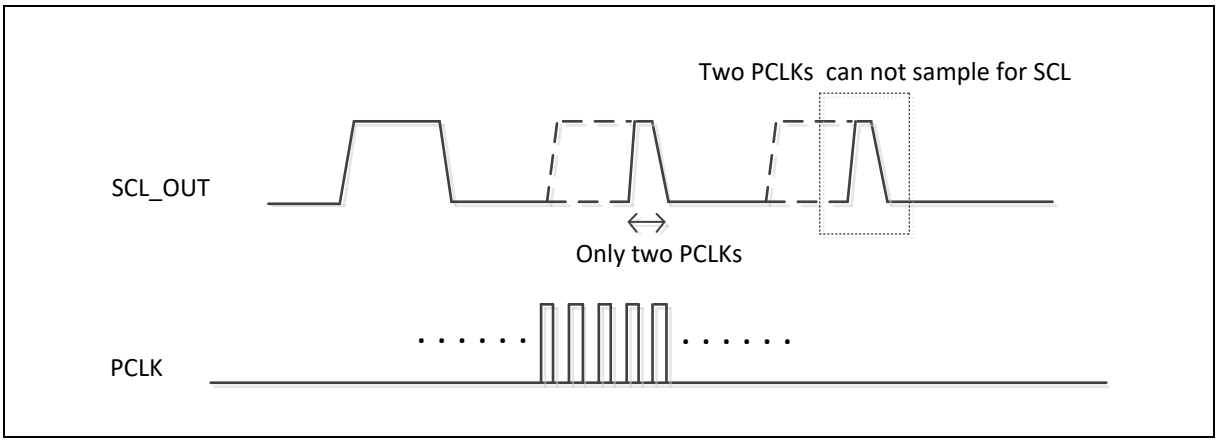


Figure 6.15-20 Setup Time Wrong Adjustment

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.

Note: Hold time adjust function can only work in master mode, when slave mode, the USCI-I²C HTCTL (UI2C_TMCTL[24:16]) should set as 0.

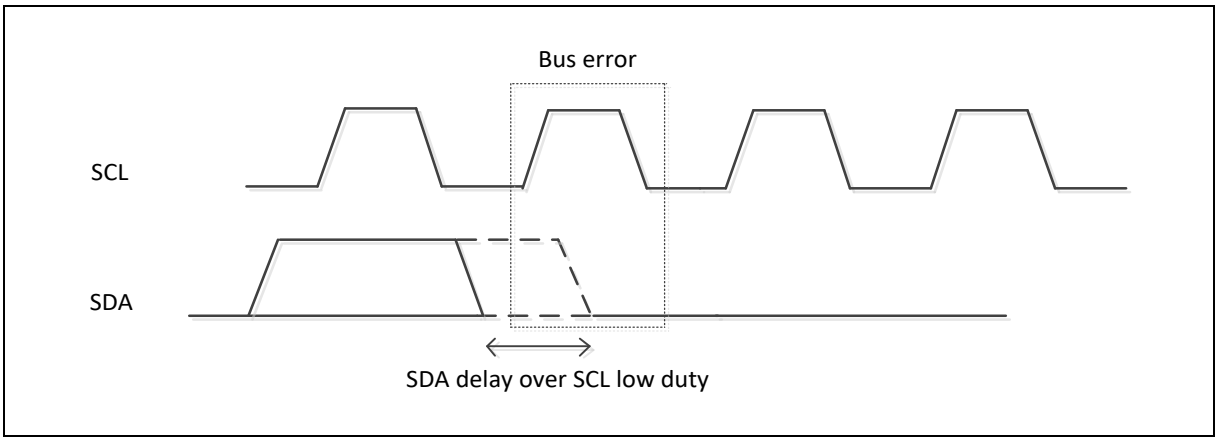


Figure 6.15-21 Hold Time Wrong Adjustment

I²C Time-out Function

There is a 10 bits 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 6.15-22 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.

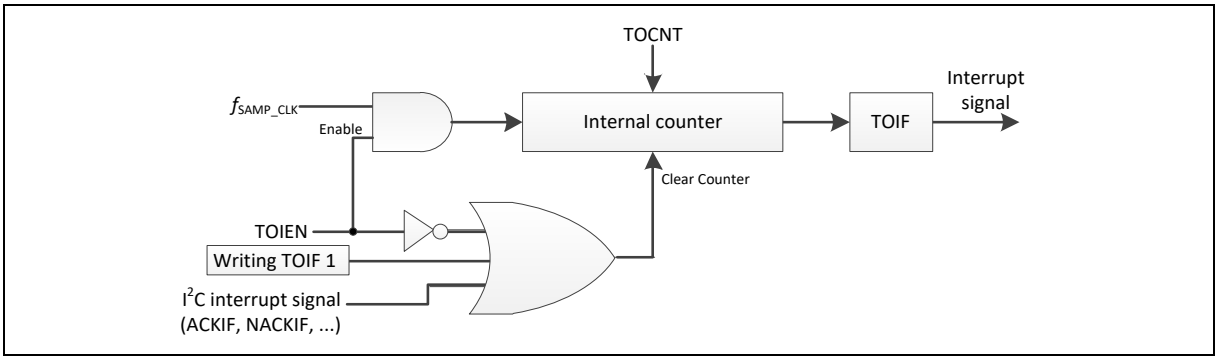


Figure 6.15-22 I²C Time-out Count Block Diagram

Wake-up Function

When chip enters Power-down mode and set WKEN (UI2C_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 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. 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 the address wakeup frame has done. The WKAKDONE bit indicates that the ACK bit cycle of address match frame is done in power-down. Remind 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. The multi-function configuration reference Basic Configuration.
2. Enable USCI0 APB clock. Enable clock configuration reference Basic Configuration.
3. Set USCI0RST=1 to reset USCI controller then set USCI0RST=0 let USCI controller to normal operation. The reset controller configuration reference Basic 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 USCI0 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_DEVADDR0 ~ UI2C_DEVADDR1”.

Random read operation is one of the methods of access EEPROM. The method allows the master to access any address of EEPROM space. Figure 6.15-23 shows the EEPROM random read operation.

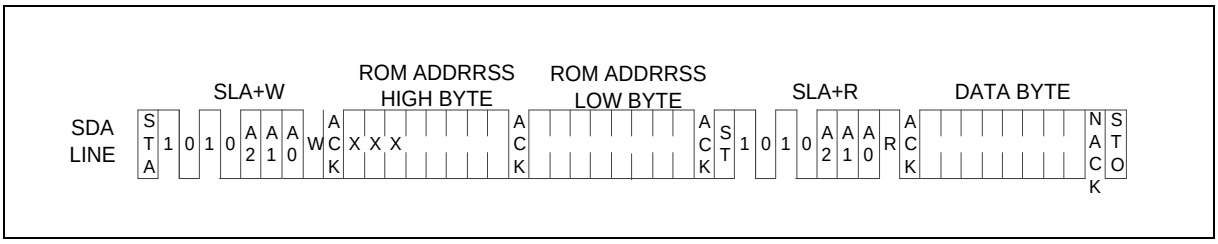


Figure 6.15-23 EEPROM Random Read

Figure 6.15-24 shows how to use I²C controller to implement the protocol of EEPROM random read.

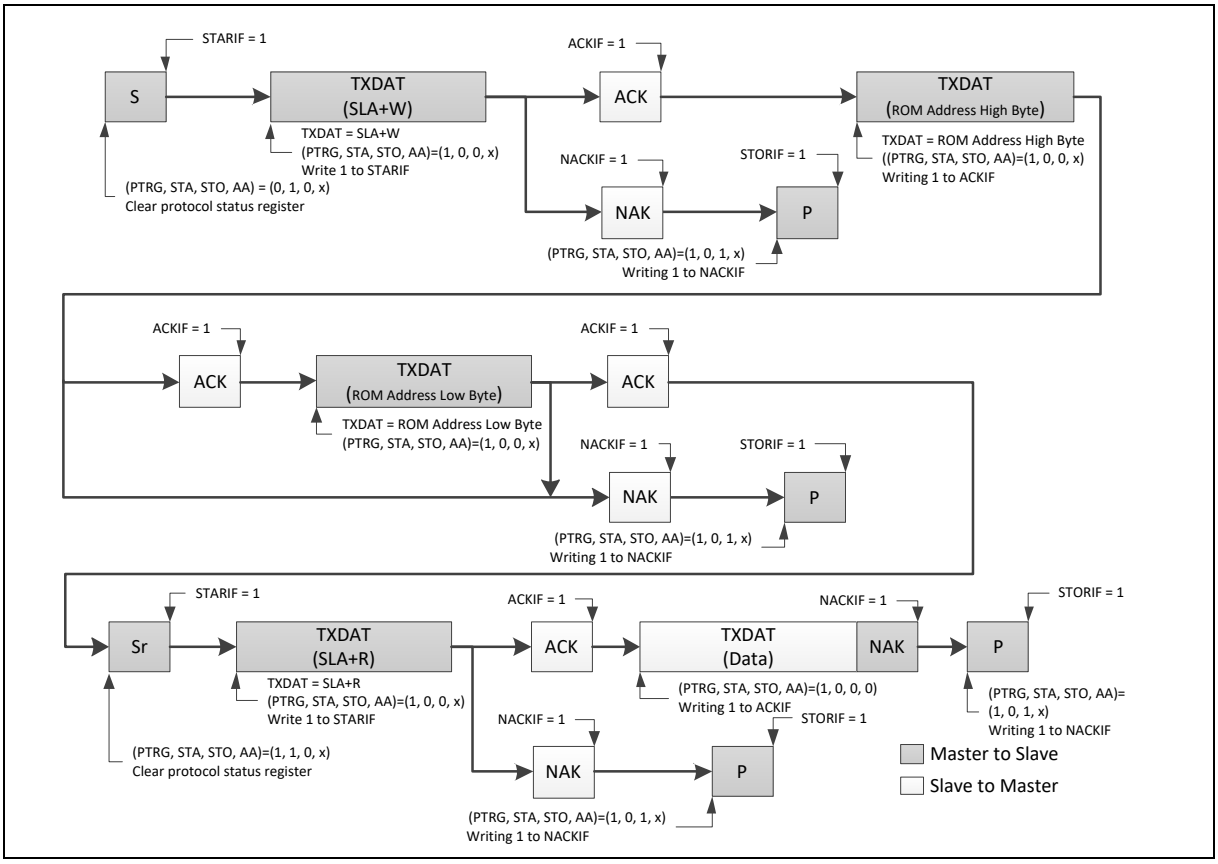


Figure 6.15-24 Protocol of EEPROM Random Read

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.

6.15.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UI2C_I2C Base Address: **UI2Cn_BA = 0x400D_0000 + (0x1000 * n)** **n= 0,1**				
UI2C_CTL	UI2Cn_BA+0x00	R/W	USCI Control Register	0x0000_0000
UI2C_BRGEN	UI2Cn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
UI2C_LINECTL	UI2Cn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
UI2C_TXDAT	UI2Cn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
UI2C_RXDAT	UI2Cn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
UI2C_DEVADDR0	UI2Cn_BA+0x44	R/W	USCI Device Address Register 0	0x0000_0000
UI2C_DEVADDR1	UI2Cn_BA+0x48	R/W	USCI Device Address Register 1	0x0000_0000
UI2C_ADDRMSK0	UI2Cn_BA+0x4C	R/W	USCI Device Address Mask Register 0	0x0000_0000
UI2C_ADDRMSK1	UI2Cn_BA+0x50	R/W	USCI Device Address Mask Register 1	0x0000_0000
UI2C_WKCTL	UI2Cn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
UI2C_WKSTS	UI2Cn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
UI2C_PROTCTL	UI2Cn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000
UI2C_PROTIEN	UI2Cn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
UI2C_PROTSTS	UI2Cn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000
UI2C_ADMAT	UI2Cn_BA+0x88	R/W	I^2C Slave Match Address Register	0x0000_0000
UI2C_TMCTL	UI2Cn_BA+0x8C	R/W	I^2C Timing Configure Control Register	0x0002_0000

6.15.7 Register Description

USCI Control Register (UI2C_CTL)

Register	Offset	R/W	Description	Reset Value
UI2C_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
UI2C_BRGEN	UI2Cn_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)).
[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).
[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 Disabled. 1 = Time measurement counter 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
UI2C_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 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].
[7:1]	**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
UI2C_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
UI2C_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
UI2C_DEVADDR0	UI2Cn_BA+0x44	R/W	USCI Device Address Register 0	0x0000_0000
UI2C_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 operating in 7-bit address mode. **Note 2:** When software set 10'h000, the address can not be used.

USCI Device Address Mask Register (UI2C_ADDRMSK) – for I²C Only

Register	Offset	R/W	Description	Reset Value
UI2C_ADDRMSK0	UI2Cn_BA+0x4C	R/W	USCI Device Address Mask Register 0	0x0000_0000
UI2C_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 support multiple address recognition with two 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.

USCI Wake-up Control Register (UI2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_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 data toggle. 1 = The chip is woken up according 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
UI2C_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
UI2C_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 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.
[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.

USCI Protocol Interrupt Enable Register – I²C (UI2C_PROTIEN)

Register	Offset	R/W	Description	Reset Value
UI2C_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
UI2C_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** 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** 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** 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** 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 clear when WKF is not be clear.
[15]	**SLAREAD**	**Slave Read Request Status** 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** 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_PROTIEN.ACKIEN = 1. 0 = An acknowledge has not been received. 1 = An acknowledge has been received. **Note:** It 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_PROTIEN.ERRIEN = 1. 0 = An I^2C error has not been detected. 1 = An I^2C error has been detected. **Note 1:** It 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_PROTIEN.ARBLOIEN = 1. 0 = An arbitration has not been lost. 1 = An arbitration has been lost. **Note:** It 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_PROTIEN.NACKIEN = 1. 0 = A non - acknowledge has not been received. 1 = A non - acknowledge has been received. **Note:** It 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_PROTIEN.STORIEN = 1. 0 = A stop condition has not yet been detected. 1 = A stop condition has been detected. **Note 1:** It 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_PROTIEN.STARIEN = 1. 0 = A start condition has not yet been detected. 1 = A start condition has been detected. **Note:** It is cleared by software writing 1 into this bit
[7]	**Reserved**	Reserved.
[6]	**ONBUSY**	**On Bus Busy** 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.
[5]	**TOIF**	**Time-out Interrupt Flag** 0 = A time-out interrupt status has not occurred. 1 = A time-out interrupt status has occurred. **Note:** It is cleared by software writing 1 into this bit
[4:0]	**Reserved**	Reserved.

USCI Slave Match Address Register (UI2C_ADMAT)

Register	Offset	R/W	Description	Reset Value
UI2C_ADMAT	UI2Cn_BA+0x88	R/W	I^2C Slave Match 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						**ADMAT1**	**ADMAT0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**ADMAT1**	**USCI Address 1 Match Status Register** 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**	**USCI Address 0 Match Status Register** 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.

USCI Timing Configure Control Register (UI2C_TMCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_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 adjust SDA transfer timing. which master will transfer SDA after SCL fallinng edge. The delay hold time is numbers of peripheral clock = HTCTL x f_{PCLK}. **Note:** Hold time adjust function can only work in master mode, when slave mode, this field should set as 0
[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}.

6.16 Controller Area Network (CAN)

6.16.1 Overview

The Bosch CAN (is named as C_CAN) consists of the CAN Core, Message RAM, Message Handler, Control Registers and Module Interface. The CAN Core performs communication according to the CAN protocol version 2.0 part A and B. The bit rate can be programmed to values up to 1 MBit/s. For the connection to the physical layer, additional transceiver hardware is required.

For communication on a CAN network, individual Message Objects are configured. The Message Objects and Identifier Masks for acceptance filtering of received messages are stored in the Message RAM. All functions concerning the handling of messages are implemented in the Message Handler. These functions include acceptance filtering, the transfer of messages between the CAN Core and the Message RAM, and the handling of transmission requests as well as the generation of the module interrupt.

The register set of the C_CAN can be accessed directly by the software through the module interface. These registers are used to control/configure the CAN Core and the Message Handler and to access the Message RAM.

6.16.2 Features

- Supports CAN protocol version 2.0 part A and B
- Bit rates up to 1 MBit/s
- 32 Message Objects
- Each Message Object has its own identifier mask
- Programmable FIFO mode (concatenation of Message Objects)
- Maskable interrupt
- Disabled Automatic Re-transmission mode for Time Triggered CAN applications
- Programmable loop-back mode for self-test operation
- 16-bit module interfaces to the AMBA APB bus
- Supports wake-up function

6.16.3 Block Diagram

The C_CAN interfaces with the AMBA APB bus. Figure 6.16-1 shows the block diagram of the C_CAN.

- CAN Core
  - CAN Protocol Controller and Rx/Tx Shift Register for serial/parallel conversion of messages.
- Message RAM
  - Stores Message Objects and Identifier Masks
- Registers
  - All registers used to control and to configure the C_CAN.
  - 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.
- Module Interface

- C_CAN interfaces to the AMBA APB 16-bit bus from CPU.

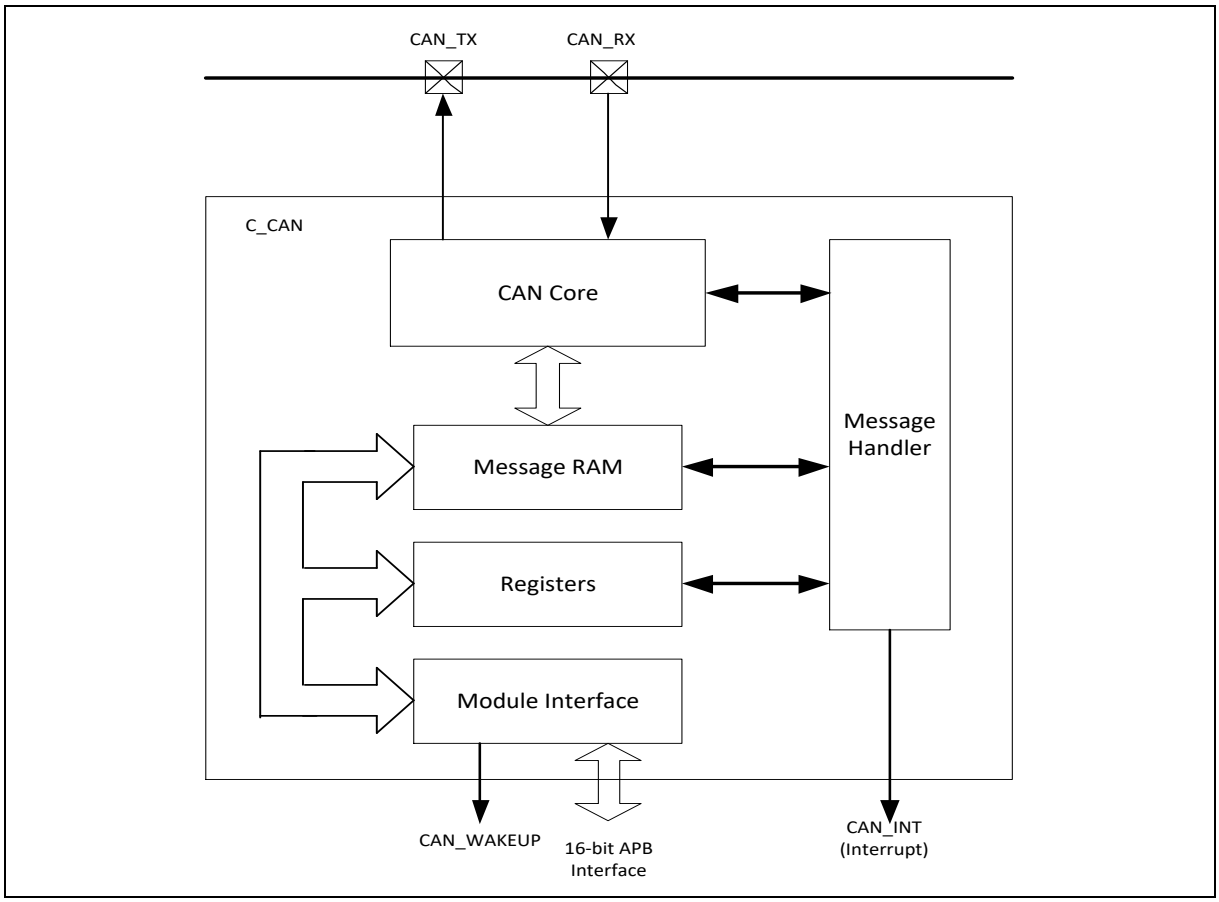


Figure 6.16-1 CAN Peripheral Block Diagram

6.16.4 Basic Configuration

6.16.4.1 CAN Basic Configuration

- Clock source Configuration
  - Enable CAN clock (CAN0CKEN (CLK_APBCLK0[24])).
- Reset Configuration
  - Reset CAN controller (CAN0RST (SYS_IPRST1[24])).
- Pin Configuration

Group	Pin Name	GPIO	MFP
CAN0	CAN0_RXD	GPA0, GPA1, GPA2, GPA3, GPA4, GPA5	MFP20
		GPB4, GPB5, GPB6, GPB7	MFP20
		GPC0, GPC1, GPC2, GPC3, GPC4, GPC5, GPC6, GPC7	MFP20
		GPD3, GPD7	MFP3
	CAN0_TXD	GPA0, GPA1, GPA2, GPA4, GPA5	MFP19
		GPB4, GPB5, GPB6, GPB7	MFP19

		GPC0, GPC1, GPC2, GPC3, GPC4, GPC5, GPC6, GPC7	MFP19
		GPD2, GPD6	MFP3

6.16.5 Functional Description

6.16.5.1 Software Initialization

The software initialization is started by setting the Init bit (CAN_CON[0]), either by a software or a hardware reset, or by going to bus-off state.

While the Init bit is set, all messages transfer to and from the CAN bus are stopped and the status of the CAN_TX output pin is recessive (HIGH). The Error Management Logic (EML) counters are unchanged. Setting the Init bit does not change any configuration register.

To initialize the CAN Controller, software has to set up the Bit Timing Register and each Message Object. If a Message Object is not required, the corresponding MsgVal bit (CAN_IFn_ARB2[15]) should be cleared. Otherwise, the entire Message Object has to be initialized.

Access to the Bit Timing Register and to the Baud Rate Prescaler Extension Register for configuring bit timing is enabled when both the Init and CCE (CAN_CON[6]) bits are set.

Resetting the Init bit (by software only) finishes the software initialization. Later, the Bit Stream Processor (BSP) (see Section 6.16.7.15: Configuring the Bit Timing) 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.

The initialization of the Message Objects is independent of Init and can be done on the fly, but the Message Objects should all be configured to particular identifiers or set to not valid before the BSP starts the message transfer.

To change the configuration of a Message Object during normal operation, the software has to start by resetting the corresponding MsgVal bit. When the configuration is completed, MsgVal bit is set again.

6.16.5.2 CAN Message Transfer

Once the C_CAN is initialized and Init bit (CAN_CON[0]) is reset to zero, the C_CAN Core synchronizes itself to the CAN bus and starts the message transfer.

Received messages are stored in their appropriate Message Objects if they pass the Message Handler's acceptance filtering. The whole message including all arbitration bits, DLC (CAN_IFn_MCON[3:0]) and eight data bytes (CAN_IFn_DAT_A1/2; CAN_IFn_DAT_B1/2) are stored in the Message Object. If the Identifier Mask is used, the arbitration bits which are masked to "don't care" may be overwritten in the Message Object.

Software can read or write each message any time through the Interface Registers and the Message Handler guarantees data consistency in case of concurrent accesses.

Messages to be transmitted are updated by the application software. If a permanent Message Object (arbitration and control bits are set during configuration) exists for the message, only the data bytes are updated and the TxRqst bit (CAN_IFn_MCON[8]) with NewDat bit (CAN_IFn_MCON[15]) are set to start the transmission. If several transmit messages are assigned to the same Message Object (when the number of Message Objects is not sufficient), the whole Message Object has to be configured before the transmission of this message is requested.

The transmission of any number of Message Objects may be requested at the same time. Message objects are transmitted subsequently according to their internal priority. Messages may be updated or set to not valid any time, even when their requested transmission is still pending. The old data will be discarded when a message is updated before its pending transmission has started.

Depending on the configuration of the Message Object, the transmission of a message may be requested autonomously by the reception of a remote frame with a matching identifier.

Disabled Automatic Retransmission

In accordance with the CAN Specification (see ISO11898, 6.3.3 Recovery Management), the C_CAN provides means for automatic retransmission of frames that have lost arbitration or have been disturbed by errors during transmission. The frame transmission service will not be confirmed to the user before the transmission is successfully completed. This means that, by default, automatic retransmission is enabled. It can be disabled to enable the C_CAN to work within a Time Triggered CAN (TTCAN, see ISO11898-1) environment.

The Disabled Automatic Retransmission mode is enabled by setting the Disable Automatic Retransmission (DAR bit (CAN_CON[5])) to one. In this operation mode, the programmer has to consider the different behavior of bits TxRqst (CAN_IFn_MCON[8]) and NewDat (CAN_IFn_MCON[15]) of the Message Buffers:

- When a transmission starts, bit TxRqst of the respective Message Buffer is cleared, while bit NewDat remains set.
- When the transmission completed successfully, bit NewDat is cleared.
- When a transmission fails (lost arbitration or error), bit NewDat remains set.
- To restart the transmission, the software should set the bit TxRqst again.

6.16.6 Test Mode

Test Mode is entered by setting the Test bit (CAN_CON[7]). In Test Mode, bits Tx1 (CAN_TEST[6]), Tx0 (CAN_TEST[5]), LBack (CAN_TEST[4]), Silent (CAN_TEST[3]) and Basic (CAN_TEST[2]) are writeable. Bit Rx (CAN_TEST[7]) monitors the state of the CAN_RX pin and therefore is only readable. All Test Register functions are disabled when the Test bit is cleared.

6.16.6.1 Silent Mode

The CAN Core can be set in Silent Mode by programming the Silent bit (CAN_TEST[3]) to one. In Silent Mode, the C_CAN is able to receive valid data frames and valid remote frames, but it sends only recessive bits on the CAN bus and it cannot start a transmission. If the CAN Core is required to send a dominant bit (ACK bit, Error Frames), the bit is rerouted internally so that the CAN Core monitors this dominant bit, although the CAN bus may remain in recessive state. The Silent Mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits. Figure 6.16-2 shows the connection of signals CAN_TX and CAN_RX to the CAN Core in Silent Mode.

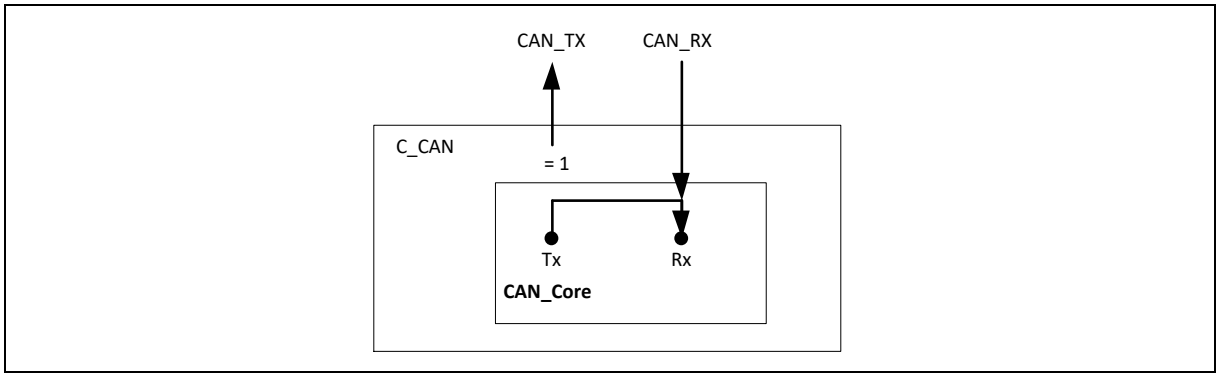


Figure 6.16-2 CAN Core in Silent Mode

6.16.6.2 Loop Back Mode

The CAN Core can be set in Loop Back Mode by programming the Test Register bit LBack (CAN_TEST[4]) to one. In Loop Back Mode, the CAN Core treats its own transmitted messages as received messages and stores them in a Receive Buffer (if they pass acceptance filtering). Figure 6.16-3 shows the connection of signals, CAN_TX and CAN_RX, to the CAN Core in Loop Back Mode.

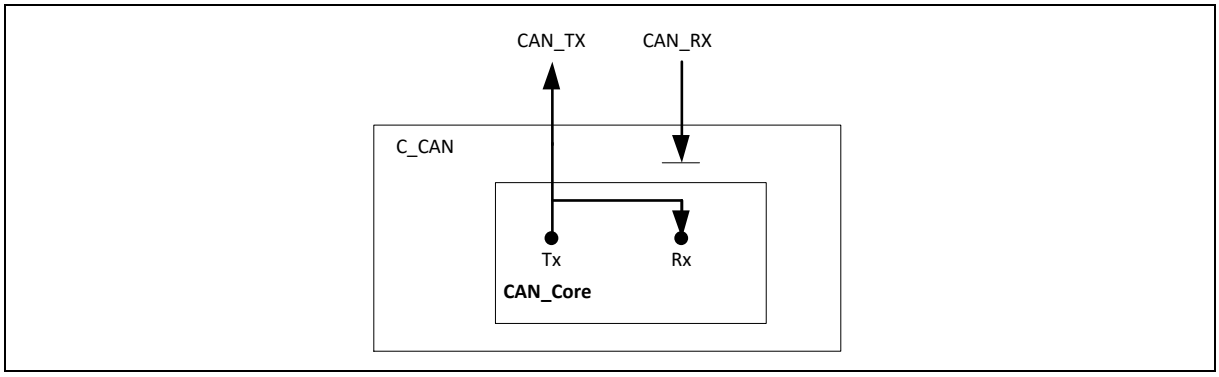


Figure 6.16-3 CAN Core in Loop Back Mode

This mode is provided for self-test functions. To be independent from external stimulation, the CAN Core ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/ remote frame) in Loop Back Mode. In this mode, the CAN Core performs an internal feedback from its Tx output to its Rx input. The actual value of the CAN_RX input pin is disregarded by the CAN Core. The transmitted messages can be monitored on the CAN_TX pin.

6.16.6.3 Loop Back Combined with Silent Mode

It is also possible to combine Loop Back Mode and Silent Mode by programming bits LBack (CAN_TEST[4]) and Silent (CAN_TEST[3]) to one at the same time. This mode can be used for a “Hot Selftest”, which means that C_CAN can be tested without affecting a running CAN system connected to the CAN_TX and CAN_RX pins. In this mode, the CAN_RX pin is disconnected from the CAN Core and the CAN_TX pin is held recessive. Figure 6.16-4 shows the connection of signals CAN_TX and CAN_RX to the CAN Core in case of the combination of Loop Back Mode with Silent Mode.

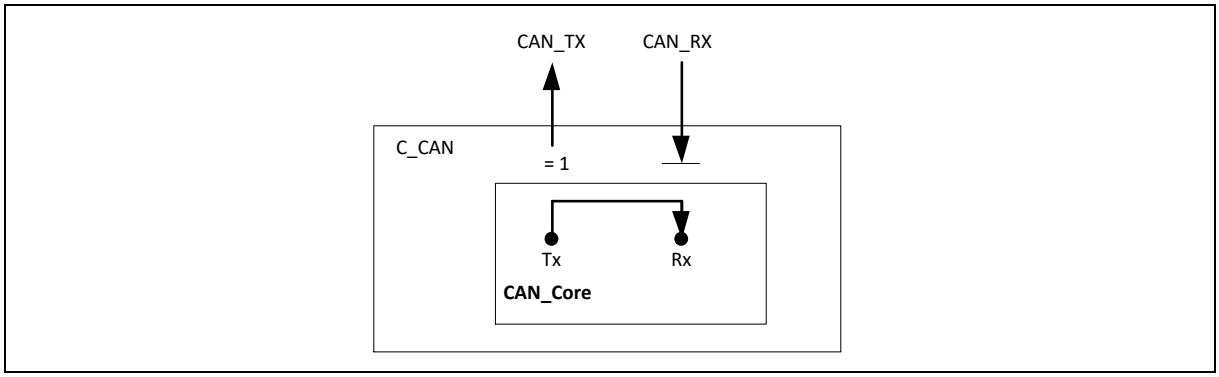


Figure 6.16-4 CAN Core in Loop Back Mode Combined with Silent Mode

6.16.6.4 Basic Mode

The CAN Core can be set in Basic Mode by programming the Basic bit (CAN_TEST[2]) to one. In this mode, the C_CAN runs without the Message RAM.

The IF1 Registers are used as Transmit Buffer. The transmission of the contents of the IF1 Registers is requested by writing the Busy bit (CAN_IFn_CREQ[15]) of the IF1 Command Request Register to one. The IF1 Registers are locked while the Busy bit is set. The Busy bit indicates that the transmission is pending.

As soon as the CAN bus is idle, the IF1 Registers are loaded into the shift register of the CAN Core and the transmission is started. When the transmission has been completed, the Busy bit is reset and the locked IF1 Registers are released.

A pending transmission can be aborted at any time by resetting the Busy bit in the IF1 Command Request Register while the IF1 Registers are locked. If the software has reset the Busy bit, a possible

retransmission in case of lost arbitration or in case of an error is disabled.

The IF2 Registers are used as a Receive Buffer. After the reception of a message the contents of the shift register is stored into the IF2 Registers, without any acceptance filtering.

Additionally, the actual contents of the shift register can be monitored during the message transfer. Each time a read Message Object is initiated by writing the Busy bit of the IF2 Command Request Register to one, the contents of the shift register are stored into the IF2 Registers.

In Basic Mode, the evaluation of all Message Object related control and status bits and the control bits of the IFn Command Mask Registers are turned off. The message number of the Command request registers is not evaluated. The NewDat (CAN_IFn_MCON[15]) and MsgLst (CAN_IFn_MCON[14]) bits retain their function, DLC3-0 indicates the received DLC (CAN_IFn_MCON[3:0]), and the other control bits are read as '0'.

6.16.6.5 Software Control of CAN_TX Pin

Four output functions are available for the CAN transmit pin, CAN_TX. In addition to its default function (serial data output), the CAN transmit pin can drive the CAN Sample Point signal to monitor CAN_Core's bit timing and it can drive constant dominant or recessive values. The latter two functions, combined with the readable CAN receive pin CAN_RX, can be used to check the physical layer of the CAN bus.

The output mode for the CAN_TX pin is selected by programming the Tx1 (CAN_TEST[6]) and Tx0 (CAN_TEST[5]) bits.

The three test functions of the CAN_TX pin interfere with all CAN protocol functions. CAN_TX must be left in its default function when CAN message transfer or any of the test modes (Loop Back Mode, Silent Mode or Basic Mode) are selected.

6.16.7 CAN Communications

6.16.7.1 Managing Message Objects

The configuration of the Message Objects in the Message RAM (with the exception of the bits MsgVal, NewDat, IntPnd and TxRqst) will not be affected by resetting the chip. All the Message Objects must be initialized by the application software or they must be "not valid" (MsgVal bit = '0') and the bit timing must be configured before the application software clears the Init bit (CAN_CON[0]).

The configuration of a Message Object is done by programming Mask, Arbitration, Control and Data fields of one of the two interface registers to the desired values. By writing to the corresponding IFn Command Request Register, the IFn Message Buffer Registers are loaded into the addressed Message Object in the Message RAM.

When the Init bit is cleared, the CAN Protocol Controller state machine of the CAN_Core and the state machine of the Message Handler control the internal data flow of the C_CAN. Received messages that pass the acceptance filtering are stored into the Message RAM, messages with pending transmission request are loaded into the CAN_Core's Shift Register and are transmitted through the CAN bus.

The application software reads received messages and updates messages to be transmitted through the IFn Interface Registers. Depending on the configuration, the application software is interrupted on certain CAN message and CAN error events.

6.16.7.2 Message Handler State Machine

The Message Handler controls the data transfer between the Rx/Tx Shift Register of the CAN Core, the Message RAM and the IFn Registers.

The Message Handler FSM controls the following functions:

- Data Transfer from IFn Registers to the Message RAM
- Data Transfer from Message RAM to the IFn Registers
- Data Transfer from Shift Register to the Message RAM
- Data Transfer from Message RAM to Shift Register

- Data Transfer from Shift Register to the Acceptance Filtering unit
- Scanning of Message RAM for a matching Message Object
- Handling of TxRqst flags
- Handling of interrupts.

6.16.7.3 Data Transfer from/to Message RAM

When the application software initiates a data transfer between the IFn Registers and Message RAM, the Message Handler sets the Busy bit (CAN_IFn_CREQ[15]) to '1'. After the transfer has completed, the Busy bit is again cleared (see Figure 6.16-5).

The respective Command Mask Register specifies whether a complete Message Object or only parts of it will be transferred. Due to the structure of the Message RAM, it is not possible to write single bits/bytes of one Message Object. It is always necessary to write a complete Message Object into the Message RAM. Therefore, the data transfer from the IFn Registers to the Message RAM requires a read-modify-write cycle. First, those parts of the Message Object that are not to be changed are read from the Message RAM and then the complete contents of the Message Buffer Registers are written into the Message Object.

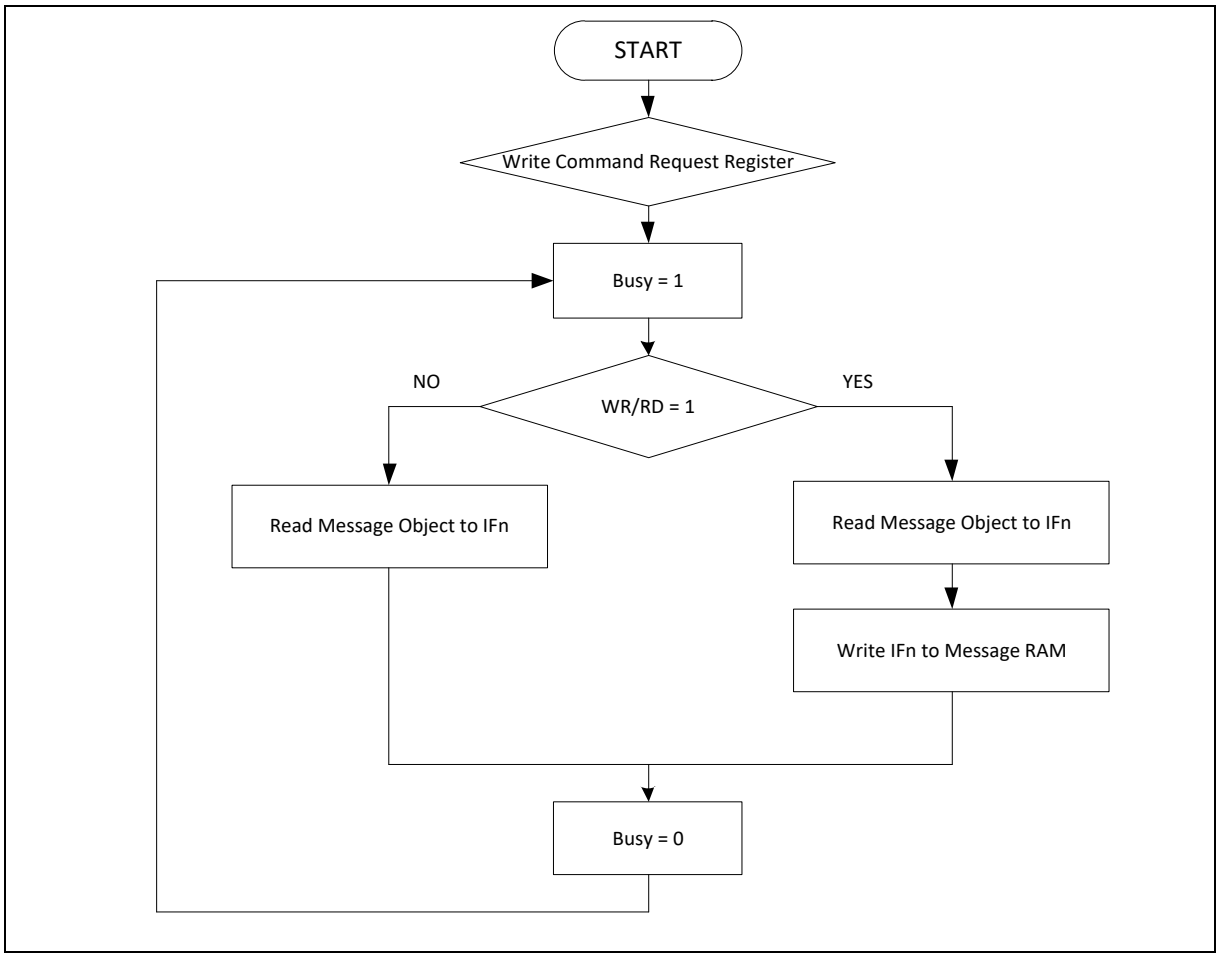


Figure 6.16-5 Data Transfer between IFn Registers and Message

After a partial write of a Message Object, the Message Buffer Registers that are not selected in the Command Mask Register will set the actual contents of the selected Message Object.

After a partial read of a Message Object, the Message Buffer Registers that are not selected in the Command Mask Register will be left unchanged.

6.16.7.4 Message Transmission

If the shift register of the CAN Core cell is ready for loading and if there is no data transfer between the IFn Registers and Message RAM, the MsgVal bit (CAN_IFn_ARB2[15]) and TxRqst bits (CAN_TXREQ1/2) are evaluated. The valid Message Object with the highest priority pending transmission request is loaded into the shift register by the Message Handler and the transmission is started. The NewDat (CAN_IFn_MCON[15]) bit of the Message Object is reset.

After a successful transmission and also if no new data was written to the Message Object (NewDat = '0') since the start of the transmission, the TxRqst bit of the Message Control register (CAN_IFn_MCON[8]) will be reset. If TxIE bit (CAN_IFn_MCON[11]) is set, IntPnd bit (CAN_IFn_MCON[13]) of the Interrupt Identifier register will be set after a successful transmission. If the C_CAN has lost the arbitration or if an error occurred during the transmission, the message will be retransmitted as soon as the CAN bus is free again. Meanwhile, if the transmission of a message with higher priority has been requested, the messages will be transmitted in the order of their priority.

6.16.7.5 Acceptance Filtering of Received Messages

When the arbitration and control field (Identifier + IDE + RTR + DLC) of an incoming message is completely shifted into the Rx/Tx Shift Register of the CAN Core, the Message Handler FSM starts the scanning of the Message RAM for a matching valid Message Object.

To scan the Message RAM for a matching Message Object, the Acceptance Filtering unit is loaded with the arbitration bits from the CAN Core shift register. The arbitration and mask fields (including MsgVal (CAN_IFn_ARB2[15]), UMask (CAN_IFn_MCON[12]), NewDat (CAN_IFn_MCON[15]) and EoB (CAN_IFn_MCON[7])) of Message Object 1 are then loaded into the Acceptance Filtering unit and compared with the arbitration field from the shift register. This is repeated with each following Message Object until a matching Message Object is found or until the end of the Message RAM is reached.

If a match occurs, the scan is stopped and the Message Handler FSM proceeds depending on the type of frame (Data Frame or Remote Frame) received.

Reception of Data Frame

The Message Handler FSM stores the message from the CAN Core shift register into the respective Message Object in the Message RAM. Not only the data bytes, but all arbitration bits and the Data Length Code are stored into the corresponding Message Object. This is done to keep the data bytes connected with the identifier even if arbitration mask registers are used.

The NewDat bit (CAN_IFn_MCON[15]) is set to indicate that new data (not yet seen by the software) has been received. The application software should reset NewDat bit when the Message Object has been read. If at the time of reception, the NewDat bit was already set, MsgLst (CAN_IFn_MCON[14]) is set to indicate that the previous data (supposedly not seen by the software) is lost. If the RxIE bit (CAN_IFn_MCON[10]) is set, the IntPnd bit (CAN_IFn_MCON[13]) is set, causing the Interrupt Register to point to this Message Object.

The TxRqst bit (CAN_IFn_MCON[8]) of this Message Object is reset to prevent the transmission of a Remote Frame, while the requested Data Frame has just been received.

Reception of Remote Frame

When a Remote Frame is received, three different configurations of the matching Message Object have to be considered:

1. Dir (CAN_IFn_ARB2[13]) = '1' (direction = transmit), RmtEn (CAN_IFn_MCON[9]) = '1' and UMask (CAN_IFn_MCON[12]) = '1' or '0'
2. At the reception of a matching Remote Frame, the TxRqst bit of this Message Object is set. The rest of the Message Object remains unchanged.
3. Dir = '1' (direction = transmit), RmtEn = '0' and UMask = '0'
4. At the reception of a matching Remote Frame, the TxRqst bit of this Message Object remains

unchanged; the Remote Frame is ignored.

5. Dir = '1' (direction = transmit), RmtEn = '0' and UMask = '1'
6. At the reception of a matching Remote Frame, the TxRqst bit of this Message Object is reset. The arbitration and control field (Identifier + IDE + RTR + DLC) from the shift register is stored in the Message Object of the Message RAM and the NewDat bit (CAN_IFn_MCON[15]) of this Message Object is set. The data field of the Message Object remains unchanged; the Remote Frame is treated similar to a received Data Frame.

6.16.7.6 Receive/Transmit Priority

The receive/transmit priority for the Message Objects is attached to the message number. Message Object 1 has the highest priority, while Message Object 32 has the lowest priority. If more than one transmission request is pending, they are serviced due to the priority of the corresponding Message Object

6.16.7.7 Configuring a Transmit Object

Table 6.16-1 shows how a Transmit Object should be initialized.

Ms	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

Table 6.16-1 Initialization of a Transmit Object

Note: appl. = application software.

The Arbitration Register values (ID28-0 (CAN_IFn_ARB1/2) and Xtd bit (CAN_IFn_ARB2[14])) are provided by the application. They define the identifier and type of the outgoing message. If an 11-bit Identifier ("Standard Frame") is used, it is programmed to ID28 - ID18. The ID17 - ID0 can then be disregarded.

If the TxIE bit (CAN_IFn_MCON[11]) is set, the IntPnd bit (CAN_IFn_MCON[13]) will be set after a successful transmission of the Message Object.

If the RmtEn bit (CAN_IFn_MCON[9]) is set, a matching received Remote Frame will cause the TxRqst bit (CAN_IFn_MCON[8]) to be set; the Remote Frame will autonomously be answered by a Data Frame.

The Data Register values (DLC3-0 (CAN_IFn_MCON[3:0]), Data(0)-(7)) are provided by the application, TxRqst and RmtEn may not be set before the data is valid.

The Mask Registers (Msk28-0, UMask, MXtd and MDir bits) may be used (UMask (CAN_IFn_MCON[12]) = '1') to allow groups of Remote Frames with similar identifiers to set the TxRqst bit. The Dir bit (CAN_IFn_ARB2[13]) should not be masked.

6.16.7.8 Updating a Transmit Object

The software may update the data bytes of a Transmit Object any time through the IFn Interface registers, neither MsgVal bit (CAN_IFn_ARB2[15]) nor TxRqst (CAN_IFn_MCON[8]) have to be reset before the update.

Even if only a part of the data bytes are to be updated, all four bytes of the corresponding IFn Data A Register or IFn Data B Register have to be valid before the contents of that register are transferred to the Message Object. Either the application software has to write all four bytes into the IFn Data Register or the Message Object is transferred to the IFn Data Register before the software writes the new data bytes.

When only the (eight) data bytes are updated, first 0x0087 is written to the Command Mask Register and then the number of the Message Object is written to the Command Request Register, concurrently

updating the data bytes and setting TxRqst.

To prevent the reset of TxRqst at the end of a transmission that may already be in progress while the data is updated, NewDat (CAN_IFn_MCON[15]) has to be set together with TxRqst.

When NewDat is set together with TxRqst, NewDat will be reset as soon as the new transmission has started.

6.16.7.9 Configuring a Receive Object

Table 6.16-2 shows how a Receive Object should be initialized.

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

Table 6.16-2 Initialization of a Receive Object

The Arbitration Registers values (ID28-0 (CAN_IFn_ARB1/2) and Xtd bit (CAN_IFn_ARB2[14])) are provided by the application. They define the identifier and type of accepted received messages. If an 11-bit Identifier (“Standard Frame”) is used, it is programmed to ID28 - ID18. Then ID17 - ID0 can be disregarded. When a Data Frame with an 11-bit Identifier is received, ID17 - ID0 will be set to ‘0’.

If the RxIE bit (CAN_IFn_MCON[10]) is set, the IntPnd bit (CAN_IFn_MCON[13]) will be set when a received Data Frame is accepted and stored in the Message Object.

The Data Length Code (DLC3-0 (CAN_IFn_MCON[3:0])) is provided by the application. When the Message Handler stores a Data Frame in the Message Object, it will store the received Data Length Code and eight data bytes. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by unspecified values.

The Mask Registers (Msk28-0, UMask, MXtd and MDir bits) may be used (UMask (CAN_IFn_MCON[12]) = ’1’) to allow groups of Data Frames with similar identifiers to be accepted. The Dir bit (CAN_IFn_ARB2[13]) should not be masked in typical applications.

6.16.7.10 Handling Received Messages

The application software may read a received message any time through the IFn Interface registers. The data consistency is guaranteed by the Message Handler state machine.

Typically, the software will write first 0x007F to the Command Mask Register and then the number of the Message Object to the Command Request Register. This combination will transfer the whole received message from the Message RAM into the Message Buffer Register. Additionally, the bits NewDat (CAN_IFn_MCON[15]) and IntPnd (CAN_IFn_MCON[13]) are cleared in the Message RAM (not in the Message Buffer).

If the Message Object uses masks for acceptance filtering, the arbitration bits show which of the matching messages have been received.

The actual value of NewDat shows whether a new message has been received since the last time this Message Object was read. The actual value of MsgLst (CAN_IFn_MCON[14]) shows whether more than one message has been received since the last time this Message Object was read. MsgLst will not be automatically reset.

By means of a Remote Frame, the software may request another CAN node to provide new data for a receive object. Setting the TxRqst bit (CAN_IFn_MCON[8]) of a receive object will cause the transmission of a Remote Frame with the receive object’s identifier. This Remote Frame triggers the other CAN node to start the transmission of the matching Data Frame. If the matching Data Frame is received before the Remote Frame could be transmitted, the TxRqst bit is automatically reset.

6.16.7.11 Configuring a FIFO Buffer

With the exception of the EoB bit (CAN_IFn_MCON[7]), the configuration of Receive Objects belonging to a FIFO Buffer is the same as the configuration of a (single) Receive Object, see Section 6.5.7.9: Configuring a Receive Object.

To concatenate two or more Message Objects into a FIFO Buffer, the identifiers and masks (if used) of these Message Objects have to be programmed to matching values. Due to the implicit priority of the Message Objects, the Message Object with the lowest number will be the first Message Object of the FIFO Buffer. The EoB bit of all Message Objects of a FIFO Buffer except the last have to be programmed to zero. The EoB bit of the last Message Object of a FIFO Buffer is set to one, configuring it as the End of the Block.

6.16.7.12 Receiving Messages with FIFO Buffers

Received messages with identifiers matching to a FIFO Buffer are stored into a Message Object of this FIFO Buffer starting with the Message Object with the lowest message number.

When a message is stored into a Message Object of a FIFO Buffer, the NewDat bit (CAN_IFn_MCON[15]) of this Message Object is set. By setting NewDat while EoB (CAN_IFn_MCON[7]) is zero, the Message Object is locked for further write access by the Message Handler until the application software has written the NewDat bit back to zero.

Messages are stored into a FIFO Buffer until the last Message Object of this FIFO Buffer is reached. If none of the preceding Message Objects is released by writing NewDat to zero, all further messages for this FIFO Buffer will be written into the last Message Object of the FIFO Buffer and therefore overwrite the previous messages.

6.16.7.13 Reading from a FIFO Buffer

When the application software transfers the contents of a Message Object to the IFn Message Buffer register by writing its number to the IFn Command Request Register, the corresponding Command Mask Register should be programmed in such a way that bits NewDat (CAN_IFn_MCON[15]) and IntPnd (CAN_IFn_MCON[13]) are reset to zero (TxRqst/NewDat (CAN_IFn_CMASK[2]) = ‘1’ and ClrIntPnd (CAN_IFn_CMASK[3]) = ‘1’). The values of these bits in the Message Control Register always reflect the status before resetting the bits.

To assure the correct function of a FIFO Buffer, the application software should read the Message Objects starting at the FIFO Object with the lowest message number.

Figure 6.16-6 shows how a set of Message Objects which are concatenated to a FIFO Buffer can be handled by the application software.

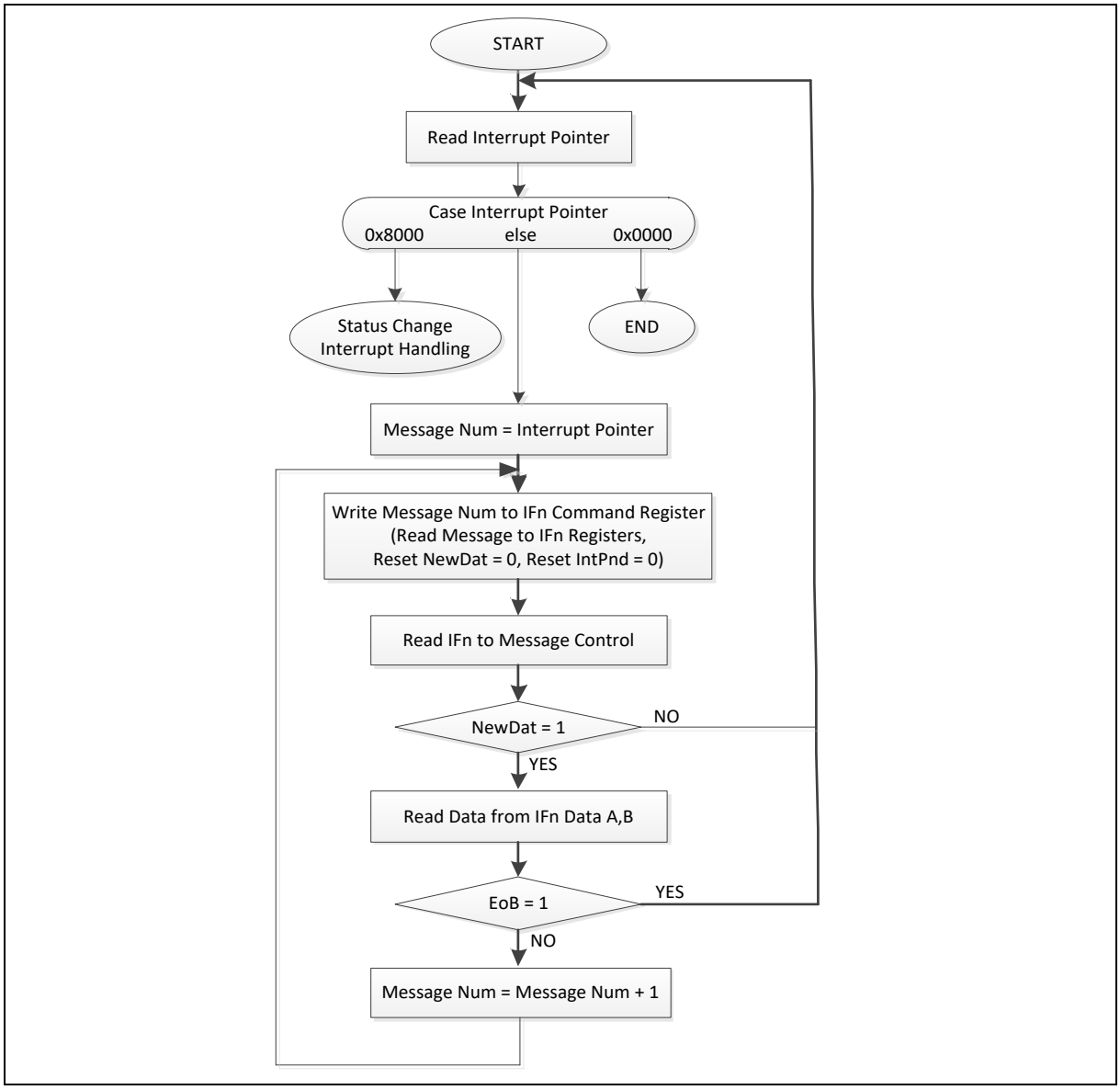


Figure 6.16-6 Application Software Handling of a FIFO Buffer

6.16.7.14 Handling Interrupts

If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the application software has cleared it.

The Status Interrupt has the highest priority. Among the message interrupts, interrupt priority of the Message Object decreases with increasing message number.

A message interrupt is cleared by clearing the IntPnd bit (CAN_IFn_MCON[13]) of the Message Object. The Status Interrupt is cleared by reading the Status Register.

The interrupt identifier, IntId, in the Interrupt Register, indicates the cause of the interrupt. When no interrupt is pending, the register will hold the value zero. If the value of the Interrupt Register is different from zero, then there is an interrupt pending and, if IE (CAN_CON[1]) is set, the CAN_INT interrupt signal is active. The interrupt remains active until the Interrupt Register is back to value zero (the cause of the interrupt is reset) or until IE is reset.

The value 0x8000 indicates that an interrupt is pending because the CAN Core has updated (not necessarily changed) the Status Register (Error Interrupt or Status Interrupt). This interrupt has the highest priority. The application software can update (reset) the status bits RxOk (CAN_STATUS[4]), TxOk (CAN_STATUS[3]) and LEC (CAN_STATUS[2:0]), but a write access of the software to the Status Register can never generate or reset an interrupt.

All other values indicate that the source of the interrupt is one of the Message Objects. IntId points to the pending message interrupt with the highest interrupt priority.

The application software controls whether a change of the Status Register may cause an interrupt (bits EIE (CAN_CON[3]) and SIE (CAN_CON[2])) and whether the interrupt line becomes active when the Interrupt Register is different from zero (bit IE in the CAN Control Register). The Interrupt Register will be updated even when IE is reset.

The application software has two possibilities to follow the source of a message interrupt. First, it can follow the IntId in the Interrupt Register and second it can poll the Interrupt Pending Register.

An interrupt service routine that is reading the message that is the source of the interrupt may read the message and reset the Message Object's IntPnd at the same time (bit ClrIntPnd (CAN_IFn_CMASK[3])). When IntPnd is cleared, the Interrupt Register will point to the next Message Object with a pending interrupt.

6.16.7.15 Configuring the Bit Timing

Even if minor errors in the configuration of the CAN bit timing do not result in immediate failure, the performance of a CAN network can be reduced significantly.

In many cases, the CAN bit synchronization will amend a faulty configuration of the CAN bit timing to such a degree that only occasionally an error frame is generated. However, in the case of arbitration, when two or more CAN nodes simultaneously try to transmit a frame, a misplaced sample point may cause one of the transmitters to become error passive.

The analysis of such sporadic errors requires a detailed knowledge of the CAN bit synchronization inside a CAN node and interaction of the CAN nodes on the CAN bus.

6.16.7.16 Bit Time and Bit Rate

CAN supports bit rates in the range of lower than 1 Kbit/s up to 1000 Kbit/s. Each member of the CAN network has its own clock generator, usually a quartz oscillator. The timing parameter of the bit time (i.e. the reciprocal of the bit rate) can be configured individually for each CAN node, creating a common bit rate even though the oscillator periods of the CAN nodes (fosc) may be different.

The frequencies of these oscillators are not absolutely stable, small variations are caused by changes in temperature or voltage and by deteriorating components. As long as the variations remain inside a specific oscillator tolerance range (df), the CAN nodes are able to compensate for the different bit rates by re-synchronizing to the bit stream.

According to the CAN specification, the bit time is divided into four segments (see Figure 6.16-7). The Synchronization Segment, the Propagation Time Segment, the Phase Buffer Segment 1 and the Phase Buffer Segment 2. Each segment consists of a specific, programmable number of time quanta (Table 6.16-3). The length of the time quantum (tq), which is the basic time unit of the bit time, is defined by the CAN controller's APB clock fAPB and the BRP bit (CAN_BTIME[5:0]) : tq = BRP / fAPB.

The Synchronization Segment, Sync_Seg, is that part of the bit time where edges of the CAN bus level are expected to occur. The distance between an edge that occurs outside of Sync_Seg, and the Sync_Seg is called the phase error of that edge. The Propagation Time Segment, Prop_Seg, is intended to compensate for the physical delay time within the CAN network. The Phase Buffer Segments Phase_Seg1 and Phase_Seg2 surround the Sample Point. The (Re-)Synchronization Jump Width (SJW) defines how far a re-synchronization may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.

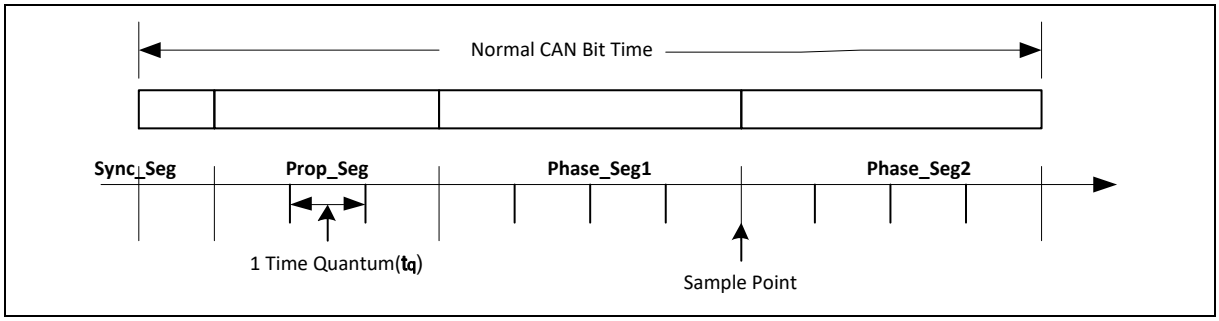


Figure 6.16-7 Bit Timing

Parameter	Range	Remark
BRP	[1..32]	Defines the length of the time quantum t_q
Sync_Seg	1 t_q	Fixed length, synchronization of bus input to APB clock
Prop_Seg	[1..8] t_q	Compensates for the physical delay time
Phase_Seg1	[1..8] t_q	Which may be lengthened temporarily by synchronization
Phase_Seg2	[1..8] t_q	Which may be shortened temporarily by synchronization
SJW	[1..4] t_q	Which may not be longer than either Phase Buffer Segment
This table describes the minimum programmable ranges required by the CAN protocol		

Table 6.16-3 CAN Bit Time Parameters

A given bit rate may be met by different bit time configurations, but for the proper function of the CAN network the physical delay time and the oscillator's tolerance range have to be considered.

6.16.7.17 Propagation Time Segment

This part of the bit time is used to compensate physical delay time within the network. These delay time consist of the signal propagation time on the bus and the internal delay time of the CAN nodes.

Any CAN node synchronized to the bit stream on the CAN bus will be out of phase with the transmitter of that bit stream, caused by the signal propagation time between the two nodes. The CAN protocol's non-destructive bitwise arbitration and the dominant acknowledge bit provided by receivers of CAN messages requires that a CAN node transmitting a bit stream must also be able to receive dominant bits transmitted by other CAN nodes that are synchronized to that bit stream. The example in Figure 6.16-8 shows the phase shift and propagation time between two CAN nodes.

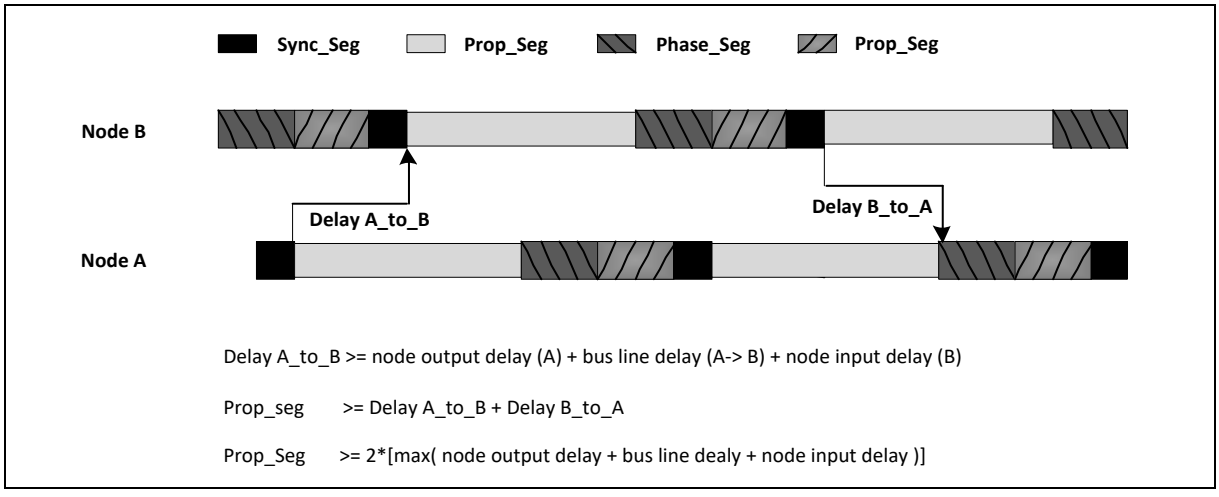


Figure 6.16-8 Propagation Time Segment

In this example, both nodes A and B are transmitters, performing an arbitration for the CAN bus. Node A has sent its Start of Frame bit less than one bit time earlier than node B, therefore node B has synchronized itself to the received edge from recessive to dominant. Since node B has received this edge delay (A_to_B) after it has been transmitted, B's bit timing segments are shifted with respect to A. Node B sends an identifier with higher priority and so it will win the arbitration at a specific identifier bit when it transmits a dominant bit while node A transmits a recessive bit. The dominant bit transmitted by node B will arrive at node A after the delay (B_to_A).

Due to oscillator tolerances, the actual position of node A's Sample Point can be anywhere inside the nominal range of node A's Phase Buffer Segments, so the bit transmitted by node B must arrive at node A before the start of Phase_Seg1. This condition defines the length of Prop_Seg.

If the edge from recessive to dominant transmitted by node B arrives at node A after the start of Phase_Seg1, it can happen that node A samples a recessive bit instead of a dominant bit, resulting in a bit error and the destruction of the current frame by an error flag.

The error occurs only when two nodes arbitrate for the CAN bus that have oscillators of opposite ends of the tolerance range and that are separated by a long bus line. This is an example of a minor error in the bit timing configuration (Prop_Seg is too short) that causes sporadic bus errors.

Some CAN implementations provide an optional 3 Sample Mode but the C_CAN does not. In this mode, the CAN bus input signal passes a digital low-pass filter, using three samples and a majority logic to determine the valid bit value. This results in an additional input delay of 1 tq, requiring a longer Prop_Seg.

6.16.7.18 Phase Buffer Segments and Synchronization

The Phase Buffer Segments (Phase_Seg1 and Phase_Seg2) and the Synchronization Jump Width (SJW) are used to compensate for the oscillator tolerance. The Phase Buffer Segments may be lengthened or shortened by synchronization.

Synchronizations occur on edges from recessive to dominant, their purpose is to control the distance between edges and Sample Points.

Edges are detected by sampling the actual bus level in each time quantum and comparing it with the bus level at the previous Sample Point. A synchronization may be done only if a recessive bit was sampled at the previous Sample PHardoint and if the bus level at the actual time quantum is dominant.

An edge is synchronous if it occurs inside of Sync_Seg, otherwise the distance between edge and the end of Sync_Seg is the edge phase error, measured in time quanta. If the edge occurs before Sync_Seg, the phase error is negative, else it is positive.

Two types of synchronization exist, Hard Synchronization and Re-synchronization.

A Hard Synchronization is done once at the start of a frame and inside a frame only when Re-synchronizations occur.

Hard Synchronization

After a hard synchronization, the bit time is restarted with the end of Sync_Seg, regardless of the edge phase error. Thus hard synchronization forces the edge, which has caused the hard synchronization to lie within the synchronization segment of the restarted bit time.

Bit Re-synchronization

Re-synchronization leads to a shortening or lengthening of the bit time such that the position of the sample point is shifted with regard to the edge.

When the phase error of the edge which causes Re-synchronization is positive, Phase_Seg1 is lengthened. If the magnitude of the phase error is less than SJW, Phase_Seg1 is lengthened by the magnitude of the phase error, else it is lengthened by SJW.

When the phase error of the edge, which causes Re-synchronization is negative, Phase_Seg2 is shortened. If the magnitude of the phase error is less than SJW, Phase_Seg2 is shortened by the magnitude of the phase error, else it is shortened by SJW.

When the magnitude of the phase error of the edge is less than or equal to the programmed value of SJW, the results of Hard Synchronization and Re-synchronization are the same. If the magnitude of the phase error is larger than SJW, the Re-synchronization cannot compensate the phase error completely, an error (phase error - SJW) remains.

Only one synchronization may be done between two Sample Points. The Synchronizations maintain a minimum distance between edges and Sample Points, giving the bus level time to stabilize and filtering out spikes that are shorter than (Prop_Seg + Phase_Seg1).

Apart from noise spikes, most synchronizations are caused by arbitration. All nodes synchronize “hard” on the edge transmitted by the “leading” transceiver that started transmitting first, but due to propagation delay time, they cannot become ideally synchronized. The “leading” transmitter does not necessarily win the arbitration, therefore the receivers have to synchronize themselves to different transmitters that subsequently “take the lead” and that are differently synchronized to the previously “leading” transmitter. The same happens at the acknowledge field, where the transmitter and some of the receivers will have to synchronize to that receiver that “takes the lead” in the transmission of the dominant acknowledge bit.

Synchronizations after the end of the arbitration will be caused by oscillator tolerance, when the differences in the oscillator’s clock periods of transmitter and receivers sum up during the time between synchronizations (at most ten bits). These summarized differences may not be longer than the SJW, limiting the oscillator’s tolerance range.

The examples in Figure 6.16-9 show how the Phase Buffer Segments are used to compensate for phase errors. There are three drawings of each two consecutive bit timings. The upper drawing shows the synchronization on a “late” edge, the lower drawing shows the synchronization on an “early” edge, and the middle drawing is the reference without synchronization.

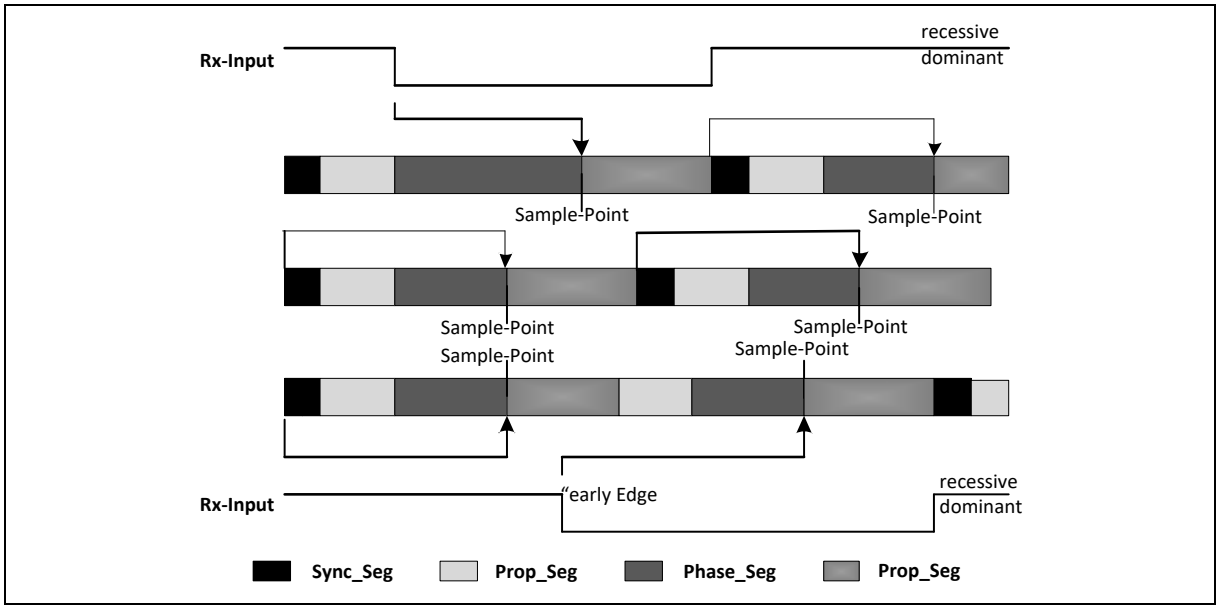


Figure 6.16-9 Synchronization on “late” and “early” Edges

In the first example an edge from recessive to dominant occurs at the end of Prop_Seg. The edge is “late” since it occurs after the Sync_Seg. Reacting to the “late” edge, Phase_Seg1 is lengthened so that the distance from the edge to the Sample Point is the same as it would have been from the Sync_Seg to the Sample Point if no edge had occurred. The phase error of this “late” edge is less than SJW, so it is fully compensated and the edge from dominant to recessive at the end of the bit, which is one nominal

bit time long, occurs in the Sync_Seg.

In the second example an edge from recessive to dominant occurs during Phase_Seg2. The edge is “early” since it occurs before a Sync_Seg. Reacting to the “early” edge, Phase_Seg2 is shortened and Sync_Seg is omitted, so that the distance from the edge to the Sample Point is the same as it would have been from an Sync_Seg to the Sample Point if no edge had occurred. As in the previous example, the magnitude of this “early” edge’s phase error is less than SJW, so it is fully compensated.

The Phase Buffer Segments are lengthened or shortened temporarily only; at the next bit time, the segments return to their nominal programmed values.

In these examples, the bit timing is seen from the point of view of the CAN implementation’s state machine, where the bit time starts and ends at the Sample Points. The state machine omits Sync_Seg when synchronising on an “early” edge because it cannot subsequently redefine that time quantum of Phase_Seg2 where the edge occurs to be the Sync_Seg.

The examples in Figure 6.16-11 show how short dominant noise spikes are filtered by synchronisations. In both examples the spike starts at the end of Prop_Seg and has the length of (Prop_Seg + Phase_Seg1).

In the first example, the Synchronization Jump Width is greater than or equal to the phase error of the spike’s edge from recessive to dominant. Therefore the Sample Point is shifted after the end of the spike; a recessive bus level is sampled.

In the second example, SJW is shorter than the phase error, so the Sample Point cannot be shifted far enough; the dominant spike is sampled as actual bus level.

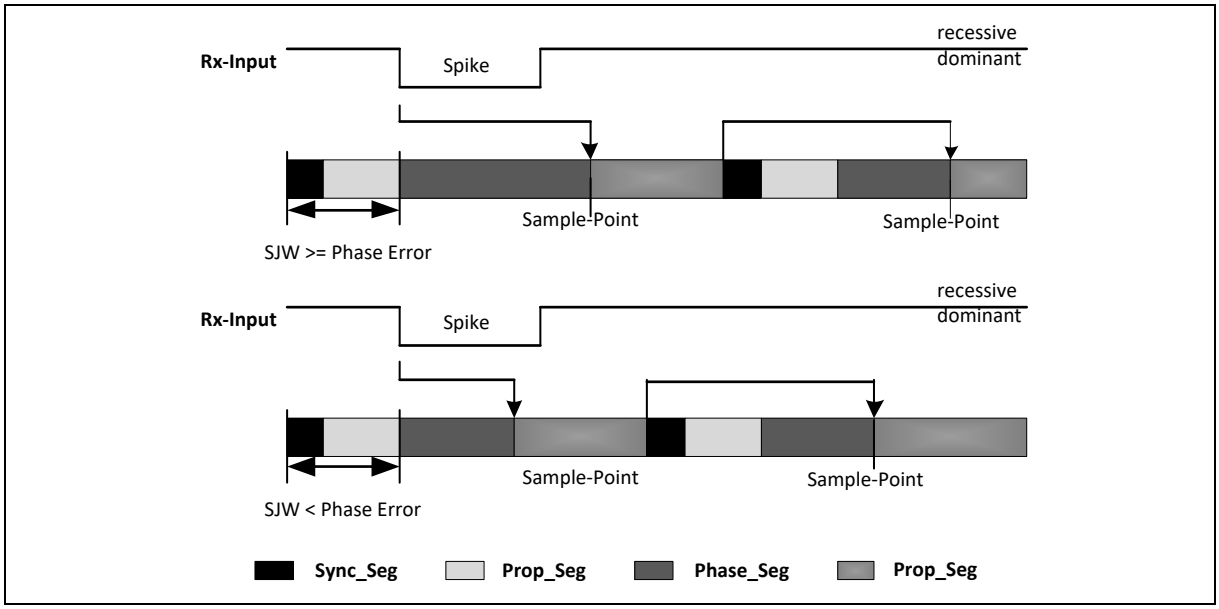


Figure 6.16-10 Filtering of Short Dominant Spikes

6.16.7.19 Oscillator Tolerance Range

The oscillator tolerance range is increased when the CAN protocol is developed from version 1.1 to version 1.2 (version 1.0 was never implemented in silicon). The option to synchronize on edges from dominant to recessive becomes obsolete, only edges from recessive to dominant are considered for synchronization. The protocol update to version 2.0 (A and B) has no influence on the oscillator tolerance.

The tolerance range d*f* for an oscillator frequency fosc around the nominal frequency fnom is:

(1 – d*f*). fnom <= fosc <= (1 + d*f*) . fnom

It depends on the proportions of Phase_Seg1, Phase_Seg2, SJW and the bit time. The maximum

tolerance df is the defined by two conditions (both shall be met):

$$I: df <= \frac{\min(\text{Phase_Seg1}, \text{Phase_Seg2})}{2 * (13 * \text{bit_time} - Phase_Seg2}$$

$$II: df <= \frac{\text{SJW}}{20 * \text{bit_tim}}$$

Note: These conditions base on the APB cock = *f* osc.

It has to be considered that SJW may not be larger than the smaller of the Phase Buffer Segments and that the Propagation Time Segment limits that part of the bit time that may be used for the Phase Buffer Segments.

The combination Prop_Seg = 1 and Phase_Seg1 = Phase_Seg2 = SJW = 4 allows the largest possible oscillator tolerance of 1.58%. This combination with a Propagation Time Segment of only 10% of the bit time is not suitable for short bit times; it can be used for bit rates of up to 125 Kbit/s (bit time = 8us) with a bus length of 40 m.

6.16.7.20 Configuring the CAN Protocol Controller

In most CAN implementations and also in the C_CAN, the bit timing configuration is programmed in two register bytes. The sum of Prop_Seg and Phase_Seg1 (as TSEG1 (CAN_BTIME[11:8])) is combined with Phase_Seg2 (as TSEG2 (CAN_BTIME[14:12])) in one byte, SJW (CAN_BTIME[7:6]) and BRP (CAN_BTIME[5:0]) are combined in the other byte.

In these bit timing registers, the four components TSEG1, TSEG2, SJW and BRP have to be programmed to a numerical value that is one less than its functional value. Therefore, instead of values in the range of [1..n], values in the range of [0..n-1] are programmed. That way, e.g. SJW (functional range of [1..4]) is represented by only two bits.

Therefore the length of the bit time is (programmed values) [TSEG1 + TSEG2 + 3] tq or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq.

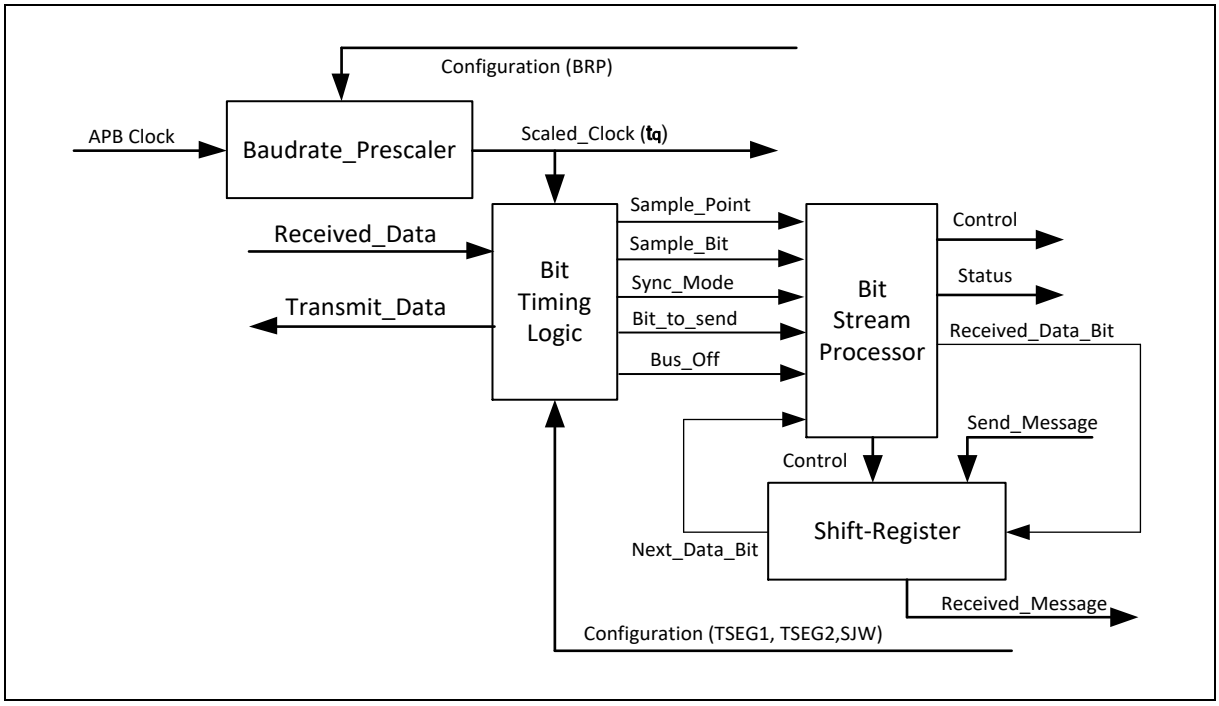


Figure 6.16-11 Structure of the CAN Core's CAN Protocol Controller

The data in the bit timing registers is the configuration input of the CAN protocol controller. The Baud Rate Prescaler (configured by BRP) defines the length of the time quantum, the basic time unit of the

bit time; the Bit Timing Logic (configured by TSEG1, TSEG2 and SJW) defines the number of time quanta in the bit time.

The processing of the bit time, the calculation of the position of the Sample Point, and occasional synchronizations are controlled by the BTL (Bit Timing Logic) state machine, which is evaluated once each time quantum. The rest of the CAN protocol controller, the BSP (Bit Stream Processor) state machine is evaluated once each bit time, at the Sample Point.

The Shift Register sends the messages serially and parallelizes received messages. It's loading and shifting is controlled by the BSP.

The BSP translates messages into frames and vice versa. It generates and discards the enclosing fixed format bits, inserts and extracts stuff bits, calculates and checks the CRC code, performs the error management, and decides which type of synchronization is to be used. It is evaluated at the Sample Point and processes the sampled bus input bit. The time that is needed to calculate the next bit to be sent after the Sample point (e.g. data bit, CRC (Cyclic Redundancy Check) bit, stuff bit, error flag or idle) is called the Information Processing Time (IPT).

The IPT is application specific but may not be longer than 2 tq; the IPT for the C_CAN is 0 tq. Its length is the lower limit of the programmed length of Phase_Seg2. In case of a synchronization, Phase_Seg2 may be shortened to a value less than IPT, which does not affect bus timing.

6.16.7.21 Calculating Bit Timing Parameters

Usually, the calculation of the bit timing configuration starts with a desired bit rate or bit time. The resulting bit time (1/bit rate) must be an integer multiple of the APB clock period.

The bit time may consist of 4 to 25 time quanta, the length of the time quantum tq is defined by the Baud Rate Prescaler with tq = (Baud Rate Prescaler)/fapb_clk. Several combinations may lead to the desired bit time, allowing iterations of the following steps.

First part of the bit time to be defined is the Prop_Seg. Its length depends on the delay time measured in the APB clock. A maximum bus length as well as a maximum node delay has to be defined for expandible CAN bus systems. The resulting time for Prop_Seg is converted into time quanta (rounded up to the nearest integer multiple of tq).

The Sync_Seg is 1 tq long (fixed), leaving (bit time – Prop_Seg – 1) tq for the two Phase Buffer Segments. If the number of remaining tq is even, the Phase Buffer Segments have the same length, Phase_Seg2 = Phase_Seg1, else Phase_Seg2 = Phase_Seg1 + 1.

The minimum nominal length of Phase_Seg2 has to be regarded as well. Phase_Seg2 may not be shorter than the IPT of the CAN controller, which, depending on the actual implementation, is in the range of [0..2] tq.

The length of the Synchronization Jump Width is set to its maximum value, which is the minimum of 4 and Phase_Seg1.

The oscillator tolerance range necessary for the resulting configuration is calculated by the formulas given in Section "Oscillator Tolerance Range".

If more than one configuration is possible, that configuration allowing the highest oscillator tolerance range should be chosen.

CAN nodes with different system clocks require different configurations to come to the same bit rate. The calculation of the propagation time in the CAN network, based on the nodes with the longest delay time, is done once for the whole network.

The oscillator tolerance range of the CAN systems is limited by that node with the lowest tolerance range.

The calculation may shows that bus length or bit rate have to be decreased or that the stability of the oscillator frequency has to be increased in order to find a protocol compliant configuration of the CAN bit timing. The resulting configuration is written into the Bit Timing Register: (Phase_Seg2-1) & (Phase_Seg1+Prop_Seg-1) & (SynchronisationJumpWidth-1) & (Prescaler-1)

Example for Bit Timing at High Baud Rate

In this example, the frequency of APB_CLK is 10 MHz, BRP (CAN_BTIME[5:0]) is 0, and the bit rate is 1 MBit/s.

t_q 100 ns = t_{APB_CLK}

delay of bus driver 50ns

delay of receiver circuit 30ns

delay of bus line (40m) 220ns

t_{Prop} 600ns = 6 • t_q

t_{SJW} 100ns = 1 • t_q

t_{TSeg1} 700ns = t_{Prop} + t_{SJW}

t_{TSeg2} 200ns = Information Processing Time + 1 • t_q

$t_{Sync\text{-}Seg}$ 100ns = 1 • t_q

bit time 1000ns = $t_{Sync\text{-}Seg}$ + t_{TSeg1} + t_{TSeg2}

tolerance for APB_CLK 0.39% = $\frac{Min(PB1, PB2)}{2 \times 13 \times (bit\ time - PB2))}$

= $\frac{0.1us}{2 \times 13 \times (1us - 0.2us))}$

In this example, the concatenated bit time parameters are $(2\text{-}1)_3$ & $(7\text{-}1)_4$ & $(1\text{-}1)_2$ & $(1\text{-}1)_6$, and the Bit Timing Register is programmed to 0x1600.

Note:

PB1/2: indicate the phase buffer segment 1/2

The subscript of $(2\text{-}1)_3$ indicates the number of bits in the corresponding bit of Bit Timing Register.

6.16.7.22 CAN Interface Reset State

After the hardware reset, the C_CAN registers hold the reset values which are given in the register description in 0.

Additionally the bus-off state is reset and the output CAN_TX is set to recessive (HIGH). The value 0x0001 (Init = '1') in the CAN Control Register enables the software initialization. The C_CAN does not influence the CAN bus until the application software resets the Init bit (CAN_CON[0]) to '0'.

The data stored in the Message RAM is not affected by a hardware reset. After powered on, the contents of the Message RAM are undefined.

Example for Bit Timing at Low Baud Rate

In this example, the frequency of APB_CLK is 2 MHz, BRP (CAN_BTIME[5:0]) is 1, and the bit rate is 100 Kbit/s.

t_q 1 us = 2 •t_{APB_CLK}

delay of bus driver 200ns

delay of receiver circuit 80ns

delay of bus line (40m) 220ns

t_{Prop} 1us = 1 • t_q

t_{SJW}	4us	= 4 • t_q
t_{TSeg1}	5us	= t_{Prop} + t_{SJW}
t_{TSeg2}	4us	= Information Processing Time + 3 • t_q
$t_{Sync-Seg}$	1us	= 1 • t_q
bit time	10us	= $t_{Sync-Seg}$ + t_{TSeg1} + t_{TSeg2}

tolerance for APB_CLK 1.58 $\% = \frac{Min(PB1, PB2)}{2 \times 13 \times (bit\ time - PB2))}$

$= \frac{4us}{2 \times (13 \times (10us - 4us))}$

In this example, the concatenated bit time parameters are $(4\text{-}1)_3$ & $(5\text{-}1)_4$ & $(4\text{-}1)_2$ & $(2\text{-}1)_6$, and the Bit Timing Register is programmed to 0x34C1.

6.16.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
CAN Base Address: **CAN_BA = 0x400A_0000**				
CAN_CON	CAN_BA+0x00	R/W	CAN Control Register	0x0000_0001
CAN_STATUS	CAN_BA+0x04	R/W	CAN Status Register	0x0000_0000
CAN_ERR	CAN_BA+0x08	R	CAN Error Counter Register	0x0000_0000
CAN_BTIME	CAN_BA+0x0C	R/W	Bit Timing Register	0x0000_2301
CAN_IIDR	CAN_BA+0x10	R	Interrupt Identifier Register	0x0000_0000
CAN_TEST	CAN_BA+0x14	R/W	Test Register (Register Map Note 1)	0x0000_0080
CAN_BRPE	CAN_BA+0x18	R/W	Baud Rate Prescaler Extension Register	0x0000_0000
CAN_IFn_CREQ **n = 1,2**	CAN_BA+0x20 + (0x60 *(n-1))	R/W	IFn (Register Map Note 2) Command Request Registers	0x0000_0001
CAN_IFn_CMASK **n=1,2**	CAN_BA+0x24 + (0x60 *(n-1))	R/W	IFn Command Mask Registers	0x0000_0000
CAN_IFn_MASK1 **n=1,2**	CAN_BA+0x28 + (0x60 *(n-1))	R/W	IFn Mask 1 Registers	0x0000_FFFF
CAN_IFn_MASK2 **n=1,2**	CAN_BA+0x2C + (0x60 *(n-1))	R/W	IFn Mask 2 Registers	0x0000_FFFF
CAN_IFn_ARB1 **n=1,2**	CAN_BA+0x30 + (0x60 *(n-1))	R/W	IFn Arbitration 1 Registers	0x0000_0000
CAN_IFn_ARB2 **n=1,2**	CAN_BA+0x34 + (0x60 *(n-1))	R/W	IFn Arbitration 2 Registers	0x0000_0000
CAN_IFn_MCON **n=1,2**	CAN_BA+0x38 + (0x60 *(n-1))	R/W	IFn Message Control Registers	0x0000_0000
CAN_IFn_DAT_A1 **n=1,2**	CAN_BA+0x3C + (0x60 *(n-1))	R/W	IFn Data A1 Registers (Register Map Note 3)	0x0000_0000
CAN_IFn_DAT_A2 **n=1,2**	CAN_BA+0x40 + (0x60 *(n-1))	R/W	IFn Data A2 Registers (Register Map Note 3)	0x0000_0000
CAN_IFn_DAT_B1 **n=1,2**	CAN_BA+0x44 + (0x60 *(n-1))	R/W	IFn Data B1 Registers (Register Map Note 3)	0x0000_0000
CAN_IFn_DAT_B2 **n=1,2**	CAN_BA+0x48 + (0x60 *(n-1))	R/W	IFn Data B2 Registers (Register Map Note 3)	0x0000_0000
CAN_TXREQ1	CAN_BA+0x100	R	Transmission Request Register 1	0x0000_0000
CAN_TXREQ2	CAN_BA+0x104	R	Transmission Request Register 2	0x0000_0000
CAN_NDAT1	CAN_BA+0x120	R	New Data Register 1	0x0000_0000

CAN_NDAT2	CAN_BA+0x124	R	New Data Register 2	0x0000_0000
CAN_IPND1	CAN_BA+0x140	R	Interrupt Pending Register 1	0x0000_0000
CAN_IPND2	CAN_BA+0x144	R	Interrupt Pending Register 2	0x0000_0000
CAN_MVLD1	CAN_BA+0x160	R	Message Valid Register 1	0x0000_0000
CAN_MVLD2	CAN_BA+0x164	R	Message Valid Register 2	0x0000_0000
CAN_WU_EN	CAN_BA+0x168	R/W	Wake-up Enable Control Register	0x0000_0000
CAN_WU_STATUS	CAN_BA+0x16C	R/W	Wake-up Status Register	0x0000_0000

Note:

1. 0x00 & 0br0000000, where r signifies the actual value of the CAN_RX
2. IFn: The two sets of Message Interface Registers – IF1 and IF2, have identical function
3. An/Bn: The two sets of data registers – A1, A2 and B1, B2.
4. CAN_BA, where x = 0 or 1.

CAN Register Map for Each Bit Function

Addr Offset	Register Name	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00h	CAN_CON	Reserved								Test	CCE	DAR	Res	EIE	SIE	IE	Init
04h	CAN_STATUS	Reserved								BOff	EWarn	EPass	RxOk	TxOk	LEC		
08h	CAN_ERR	RP	REC6-0							TEC7-0							
0Ch	CAN_BTIME	Res	TSeg2			TSeg1				SJW		BRP					
10h	CAN_IIDR	IntId15-8								IntId7-0							
14h	CAN_TEST	Reserved								Rx	Tx1	Tx0	LBack	Silent	Basic	Reserved	
18h	CAN_BRPE	Reserved												BRPE			
20h	CAN_IF1_CREQ	Busy	Reserved									Message Number					
24h	CAN_IF1_CMASK	Reserved								WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/	Data A	Data B
28h	CAN_IF1_MASK1	Msk15-0															
2Ch	CAN_IF1_MASK2	MXtd	MDir	Res	Msk28-16												
30h	CAN_IF1_ARB1	ID15-0															
34h	CAN_IF1_ARB2	MsgVal	Xtd	Dir	ID28-16												

Addr Offset	Register Name	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
38h	CAN_IF1_MCON	NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB	Reserved			DLC3-0			
3Ch	CAN_IF1_DAT_A1	Data(1)								Data(0)							
40h	CAN_IF1_DAT_A2	Data(3)								Data(2)							
44h	CAN_IF1_DAT_B1	Data(5)								Data(4)							
48h	CAN_IF1_DAT_B2	Data(7)								Data(6)							
80h	CAN_IF2_CREQ	Busy	Reserved									Message Number					
84h	CAN_IF2_CMASK	Reserved								WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/	Data A	Data B
88h	CAN_IF2_MASK1	Msk15-0															
8Ch	CAN_IF2_MASK2	MXtd	MDir	Res.	Msk28-16												
90h	CAN_IF2_ARB1	ID15-0															
94h	CAN_IF2_ARB2	MsgVal	Xtd	Dir	ID28-16												
98h	CAN_IF2_MCON	NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB	Reserved			DLC3-0			
9Ch	CAN_IF2_DAT_A1	Data(1)								Data(0)							

Addr Offset	Register Name	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A0h	CAN_IF2_DAT_A2	Data(3)								Data(2)							
A4h	CAN_IF2_DAT_B1	Data(5)								Data(4)							
A8h	CAN_IF2_DAT_B2	Data(7)								Data(6)							
100h	CAN_TXREQ1	TxRqst16-1															
104h	CAN_TXREQ2	TxRqst32-17															
120h	CAN_NDAT1	NewDat16-1															
124h	CAN_NDAT2	NewDat32-17															
140h	CAN_IPND1	IntPnd16-1															
144h	CAN_IPND2	IntPnd32-17															
160h	CAN_MVLD1	MsgVal16-1															
164h	CAN_MVLD2	MsgVal32-17															
168h	CAN_WU_EN	Reserved															WAKUP_EN
16Ch	CAN_WU_STATUS	Reserved															WAKUP_STS
170h	CAN_RAM_CEN	Reserved															RAM_CEN
Others	Reserved	Reserved															

Table 6.16-4 CAN Register Map for Each Bit Function

Note: Reserved bits are read as 0’ except for IFn Mask 2 Register where they are read as ’1’.

Res. = Reserved

6.16.9 Register Description

The C_CAN allocates an address space of 256 bytes. The registers are organized as 16-bit registers.

The two sets of interface registers (IF1 and IF2) control the software access to the Message RAM. They buffer the data to be transferred to and from the RAM, avoiding conflicts between software accesses and message reception/transmission.

CAN Control Register (CAN_CON)

Register	Offset	R/W	Description	Reset Value
CAN_CON	CAN_BA+0x00	R/W	CAN 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
Test	CCE	DAR	Reserved	EIE	SIE	IE	Init

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**Test**	**Test Mode Enable Bit** 0 = Normal Operation. 1 = Test Mode.
[6]	**CCE**	**Configuration Change Enable Bit** 0 = No write access to the Bit Timing Register. 1 = Write access to the Bit Timing Register (CAN_BTIME) allowed. (while Init bit (CAN_CON[0]) = 1).
[5]	**DAR**	**Automatic Re-transmission Disable Bit** 0 = Automatic Retransmission of disturbed messages Enabled. 1 = Automatic Retransmission Disabled.
[4]	**Reserved**	Reserved.
[3]	**EIE**	**Error Interrupt Enable Bit** 0 = Disabled - No Error Status Interrupt will be generated. 1 = Enabled - A change in the bits BOff (CAN_STATUS[7]) or EWarn (CAN_STATUS[6]) in the Status Register will generate an interrupt.
[2]	**SIE**	**Status Change Interrupt Enable Bit** 0 = Disabled - No Status Change Interrupt will be generated. 1 = Enabled - An interrupt will be generated when a message transfer is successfully completed or a CAN bus error is detected.
[1]	**IE**	**Module Interrupt Enable Bit** 0 = Funcrion interrupt Disabled.

		1 = Funcrion interrupt Enabled.
[0]	**Init**	**Init Initialization** 0 = Normal Operation. 1 = Initialization is started.

Note: The bus-off recovery sequence (see CAN Specification Rev. 2.0) cannot be shortened by setting or resetting the Init bit (CAN_CON[0]). If the device goes in the bus-off state, it will set Init of its own accord, stopping all bus activities. Once Init 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 operations. At the end of the bus-off recovery sequence, the Error Management Counters will be reset.

During the waiting time after resetting Init, each time a sequence of 11 recessive bits has been monitored, a Bit0Error code is written to the Status Register, enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the proceeding of the bus-off recovery sequence.

CAN Status Register (CAN_STATUS)

Register	Offset	R/W	Description	Reset Value
CAN_STATUS	CAN_BA+0x04	R/W	CAN 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
BOff	EWarn	EPass	RxOK	TxOK	LEC		

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**BOff**	**Bus-off Status (Read Only)** 0 = The CAN module is not in bus-off state. 1 = The CAN module is in bus-off state.
[6]	**EWarn**	**Error Warning Status (Read Only)** 0 = Both error counters are below the error warning limit of 96. 1 = At least one of the error counters in the EML has reached the error warning limit of 96.
[5]	**EPass**	**Error Passive (Read Only)** 0 = The CAN Core is error active. 1 = The CAN Core is in the error passive state as defined in the CAN Specification.
[4]	**RxOK**	**Received a Message Successfully** 0 = No message has been successfully received since this bit was last reset by the CPU. This bit is never reset by the CAN Core. 1 = A message has been successfully received since this bit was last reset by the CPU (independent of the result of acceptance filtering).
[3]	**TxOK**	**Transmitted a Message Successfully** 0 = Since this bit was reset by the CPU, no message has been successfully transmitted. This bit is never reset by the CAN Core. 1 = Since this bit was last reset by the CPU, a message has been successfully (error free and acknowledged by at least one other node) transmitted.
[2:0]	**LEC**	**Last Error Code (Type of the Last Error to Occur on the CAN Bus)** The LEC field holds a code, which 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. The unused code '7' may be written by the CPU to check for updates. Table 6.16-5 Last Error Codedescribes the error code.

Error Code	Meanings

0	No Error
1	Stuff Error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.
2	Form Error: A fixed format part of a received frame has the wrong format.
3	AckError: The message this CAN Core transmitted was not acknowledged by another node.
4	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.
5	Bit0Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), though 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 proceedings of the bus-off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed).
6	CRCError: The CRC check sum was incorrect in the message received, the CRC received for an incoming message does not match with the calculated CRC for the received data.
7	Unused: When the LEC shows the value '7', no CAN bus event was detected since the CPU wrote this value to the LEC.

Table 6.16-5 Last Error Code

Status Interrupts

A Status Interrupt is generated by bits BOff (CAN_STATUS[7]) and EWarn (CAN_STATUS[6]) (Error Interrupt) or by RxOk (CAN_STATUS[4]), TxOk (CAN_STATUS[3]) and LEC (CAN_STATUS[2:0]) (Status Change Interrupt) assumed that the corresponding enable bits in the CAN Control Register are set. A change of bit EPass (CAN_STATUS[5]) or a write to RxOk, TxOk or LEC will never generate a Status Interrupt.

Reading the Status Register will clear the Status Interrupt value (8000h) in the Interrupt Register, if it is pending.

CAN Error Counter Register (CAN_ERR)

Register	Offset	R/W	Description	Reset Value
CAN_ERR	CAN_BA+0x08	R	CAN Error 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
RP	REC						
7	6	5	4	3	2	1	0
TEC							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**RP**	**Receive Error Passive** 0 = The Receive Error Counter is below the error passive level. 1 = The Receive Error Counter has reached the error passive level as defined in the CAN Specification.
[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.

Bit Timing Register (CAN_BTIME)

Register	Offset	R/W	Description	Reset Value
CAN_BTIME	CAN_BA+0x0C	R/W	Bit Timing Register	0x0000_2301

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	TSeg2			TSeg1			
7	6	5	4	3	2	1	0
SJW		BRP					

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**TSeg2**	**Time Segment After Sample Point** 0x0-0x7: Valid values for TSeg2 are [0…7]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
[11:8]	**TSeg1**	**Time Segment Before the Sample Point Minus Sync_Seg** 0x01-0x0F: valid values for TSeg1 are [1…15]. The actual interpretation by the hardware of this value is such that one more than the value programmed is used.
[7:6]	**SJW**	**(Re)Synchronization Jump Width** 0x0-0x3: Valid programmed values are [0…3]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
[5:0]	**BRP**	**Baud Rate Prescaler** 0x01-0x3F: 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 Baud Rate Prescaler are [0…63]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

Note: With a module clock APB_CLK of 8 MHz, the reset value of 0x2301 configures the C_CAN for a bit rate of 500 Kbit/s. The registers are only writable if bits CCE (CAN_CON[6]) and Init (CAN_CON[0]) are set.

Interrupt Identify Register (CAN_IIDR)

Register	Offset	R/W	Description	Reset Value
CAN_IIDR	CAN_BA+0x10	R	Interrupt Identifier Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**IntId**	**Interrupt Identifier (Indicates the Source of the Interrupt)** If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the application software has cleared it. If IntId is different from 0x0000 and IE (CAN_CON[1]) is set, the IRQ interrupt signal to the EIC is active. The interrupt remains active until IntId is back to value 0x0000 (the cause of the interrupt is reset) or until IE is reset. The Status Interrupt has the highest priority. Among the message interrupts, the Message Object' s interrupt priority decreases with increasing message number. A message interrupt is cleared by clearing the Message Object's IntPnd bit (CAN_IFn_MCON[13]). The Status Interrupt is cleared by reading the Status Register.

IntId Value	Meanings
0x0000	No Interrupt is Pending
0x0001-0x0020	Number of Message Object which caused the interrupt.
0x0021-0x7FFF	Unused
0x8000	Status Interrupt
0x8001-0xFFFF	Unused

Table 6.16-6 Source of Interrupts

Test Register (CAN_TEST)

Register	Offset	R/W	Description	Reset Value
CAN_TEST	CAN_BA+0x14	R/W	Test Register (Register Map Note 1)	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
Rx	Tx		LBack	Silent	Basic	Reserved	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**Rx**	**Monitors the Actual Value of CAN_RX Pin (Read Only) *(1)** 0 = The CAN bus is dominant (CAN_RX = '0'). 1 = The CAN bus is recessive (CAN_RX = '1').
[6:5]	**Tx**	**Tx[1:0]: Control of CAN_TX Pin** 00 = Reset value, CAN_TX pin is controlled by the CAN Core. 01 = Sample Point can be monitored at CAN_TX pin. 10 = CAN_TX pin drives a dominant ('0') value. 11 = CAN_TX pin drives a recessive ('1') value.
[4]	**LBack**	**Loop Back Mode Enable Bit** 0 = Loop Back Mode Disabled. 1 = Loop Back Mode Enabled.
[3]	**Silent**	**Silent Mode** 0 = Normal operation. 1 = The module is in Silent Mode.
[2]	**Basic**	**Basic Mode** 0 = Basic Mode Disabled. 1= IF1 Registers used as Tx Buffer, IF2 Registers used as Rx Buffer.
[1:0]	**Reserved**	Reserved.

Reset value: 0000 0000 R000 0000 b (R:current value of RX pin)

Note: Write access to the Test Register is enabled by setting the Test bit (CAN_CON[7]). The different test functions may be combined, but Tx[1-0] "00" (CAN_TEST[6:5]) disturbs message transfer.

Baud Rate Prescaler Extension REGISTER (CAN_BRPE)

Register	Offset	R/W	Description	Reset Value
CAN_BRPE	CAN_BA+0x18	R/W	Baud Rate Prescaler Extension Register	0x0000_0000

31	30	29	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				**BRPE**			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**BRPE**	**BRPE: Baud Rate Prescaler Extension** 0x00-0x0F: By programming BRPE, the Baud Rate Prescaler can be extended to values up to 1023. The actual interpretation by the hardware is that one more than the value programmed by BRPE (MSBs) and BTIME (LSBs) is used.

Message Interface Register Sets

There are two sets of Interface Registers, which are used to control the CPU access to the Message RAM. The Interface Registers avoid conflict between the CPU accesses to the Message RAM and CAN message reception and transmission by buffering the data to be transferred. A complete Message Object or parts of the Message Object may be transferred between the Message RAM and the IFn Message Buffer registers in one single transfer.

The function of the two interface register sets is identical except for the Basic test mode. They can be used the way one set of registers is used for data transfer to the Message RAM while the other set of registers is used for the data transfer from the Message RAM, allowing both processes to be interrupted by each other. Table 6.16-7 provides an overview of the two Interface Register sets.

Each set of Interface Registers consists of Message Buffer Registers controlled by their own Command Registers. The Command Mask Register specifies the direction of the data transfer and which parts of a Message Object will be transferred. The Command Request Register is used to select a Message Object in the Message RAM as target or source for the transfer and to start the action specified in the Command Mask Register.

Address	IF1 Register Set	Address	IF2 Register Set
CAN_BA+0x20	IF1 Command Request	CAN_BA+0x80	IF2 Command Request
CAN_BA+0x24	IF1 Command Mask	CAN_BA+0x84	IF2 Command Mask
CAN_BA+0x28	IF1 Mask 1	CAN_BA+0x88	IF2 Mask 1
CAN_BA+0x2C	IF1 Mask 2	CAN_BA+0x8C	IF2 Mask 2
CAN_BA+0x30	IF1 Arbitration 1	CAN_BA+0x90	IF2 Arbitration 1
CAN_BA+0x34	IF1 Arbitration 2	CAN_BA+0x94	IF2 Arbitration 2
CAN_BA+0x38	IF1 Message Control	CAN_BA+0x98	IF2 Message Control
CAN_BA+0x3C	IF1 Data A 1	CAN_BA+0x9C	IF2 Data A 1
CAN_BA+0x40	IF1 Data A 2	CAN_BA+0xA0	IF2 Data A 2
CAN_BA+0x44	IF1 Data B 1	CAN_BA+0xA4	IF2 Data B 1
CAN_BA+0x48	IF1 Data B 2	CAN_BA+0xA8	IF2 Data B 2

Table 6.16-7 IF1 and IF2 Message Interface Register

IFn Command Request Register (CAN_IFn_CREQ)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_CREQ	CAN_BA+0x20 + (0x60 *(n-1))	R/W	IFn (Register Map Note 2) Command Request Registers	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
Busy	Reserved						
7	6	5	4	3	2	1	0
Reserved		Message Number					

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**Busy**	**Busy Flag** 0 = Read/write action has finished. 1 = Writing to the IFn Command Request Register is in progress. This bit can only be read by the software.
[14:6]	**Reserved**	Reserved.
[5:0]	**Message Number**	**Message Number** 0x01-0x20: Valid Message Number, the Message Object in the Message RAM is selected for data transfer. 0x00: Not a valid Message Number, interpreted as 0x20. 0x21-0x3F: Not a valid Message Number, interpreted as 0x01-0x1F.

A message transfer is started as soon as the application software has written the message number to the Command Request Register. With this write operation, the Busy bit (CAN_IFn_CREQ[15]) is automatically set to notify the CPU that a transfer is in progress. After a waiting time of 3 to 6 APB_CLK periods, the transfer between the Interface Register and the Message RAM is completed. The Busy bit is cleared.

Note: When a Message Number that is not valid is written into the Command Request Register, the Message Number will be transformed into a valid value and that Message Object will be transferred.

IFn Command Mask Register (CAN_IFn_CMASK)

The control bits of the IFn Command Mask Register specify the transfer direction and select which of the IFn Message Buffer Registers are source or target of the data transfer.

Register	Offset	R/W	Description	Reset Value
CAN_IFn_CMASK	CAN_BA+0x24 + (0x60 *(n-1))	R/W	IFn Command Mask 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							
7	6	5	4	3	2	1	0
WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/NewDat	DAT_A	DAT_B

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**WR/RD**	**Write / Read Mode** 0 = Read: Transfer data from the Message Object addressed by the Command Request Register into the selected Message Buffer Registers. 1 = Write: Transfer data from the selected Message Buffer Registers to the Message Object addressed by the Command Request Register.
[6]	**Mask**	**Access Mask Bits** Write Operation: 0 = Mask bits unchanged. 1 = Transfer Identifier Mask + MDir + MXtd to Message Object. Read Operation: 0 = Mask bits unchanged. 1 = Transfer Identifier Mask + MDir + MXtd to IFn Message Buffer Register.
[5]	**Arb**	**Access Arbitration Bits** Write Operation: 0 = Arbitration bits unchanged. 1 = Transfer Identifier + Dir (CAN_IFn_ARB2[13]) + Xtd (CAN_IFn_ARB2[14]) + MsgVal (CAN_IFn_ARB2[15]) to Message Object. Read Operation: 0 = Arbitration bits unchanged. 1 = Transfer Identifier + Dir + Xtd + MsgVal to IFn Message Buffer Register.
[4]	**Control**	**Control Access Control Bits** Write Operation: 0 = Control Bits unchanged. 1 = Transfer Control Bits to Message Object. Read Operation:

		0 = Control Bits unchanged. 1 = Transfer Control Bits to IFn Message Buffer Register.
[3]	**ClrIntPnd**	**Clear Interrupt Pending Bit** Write Operation: When writing to a Message Object, this bit is ignored. Read Operation: 0 = IntPnd bit (CAN_IFn_MCON[13]) remains unchanged. 1 = Clear IntPnd bit in the Message Object.
[2]	**TxRqst/NewDat**	**Access Transmission Request Bit When Write Operation** 0 = TxRqst bit unchanged. 1 = Set TxRqst bit. **Note:** If a transmission is requested by programming bit TxRqst/NewDat in the IFn Command Mask Register, bit TxRqst in the IFn Message Control Register will be ignored. Access New Data Bit when Read Operation. 0 = NewDat bit remains unchanged. 1 = Clear NewDat bit in the Message Object. **Note:** A read access to a Message Object can be combined with the reset of the control bits IntPnd and NewDat. The values of these bits transferred to the IFn Message Control Register always reflect the status before resetting these bits.
[1]	**DAT_A**	**Access Data Bytes [3:0]** Write Operation: 0 = Data Bytes [3:0] unchanged. 1 = Transfer Data Bytes [3:0] to Message Object. Read Operation: 0 = Data Bytes [3:0] unchanged. 1 = Transfer Data Bytes [3:0] to IFn Message Buffer Register.
[0]	**DAT_B**	**Access Data Bytes [7:4]** Write Operation: 0 = Data Bytes [7:4] unchanged. 1 = Transfer Data Bytes [7:4] to Message Object. Read Operation: 0 = Data Bytes [7:4] unchanged. 1 = Transfer Data Bytes [7:4] to IFn Message Buffer Register.

IFn Mask 1 Register (CAN_IFn_MASK1)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_MASK1	CAN_BA+0x28 + (0x60 *(n-1))	R/W	IFn Mask 1 Registers	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**
Msk							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Msk							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**Msk**	**Identifier Mask 15-0** 0 = The corresponding bit in the identifier of the message object cannot inhibit the match in the acceptance filtering. 1 = The corresponding identifier bit is used for acceptance filtering.

IFn Mask 2 Register (CAN_IFn_MASK2)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_MASK2	CAN_BA+0x2C + (0x60 *(n-1))	R/W	IFn Mask 2 Registers	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
MXtd	MDir	Reserved	Msk				
7	6	5	4	3	2	1	0
Msk							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**MXtd**	**Mask Extended Identifier** 0 = The extended identifier bit (IDE) has no effect on the acceptance filtering. 1 = The extended identifier bit (IDE) is used for acceptance filtering. **Note:** When 11-bit (“standard”) Identifiers are used for a Message Object, the identifiers of received Data Frames are written into bits ID28 to ID18 (CAN_IFn_ARB2[12:2]). For acceptance filtering, only these bits together with mask bits Msk28 to Msk18 (CAN_IFn_MASK2[12:2]) are considered.
[14]	**MDir**	**Mask Message Direction** 0 = The message direction bit (Dir (CAN_IFn_ARB2[13])) has no effect on the acceptance filtering. 1 = The message direction bit (Dir) is used for acceptance filtering.
[13]	**Reserved**	Reserved.
[12:0]	**Msk**	**Identifier Mask 28-16** 0 = The corresponding bit in the identifier of the message object cannot inhibit the match in the acceptance filtering. 1 = The corresponding identifier bit is used for acceptance filtering.

IFn Arbitration 1 Register (CAN_IFn_ARB1)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_ARB1	CAN_BA+0x30 + (0x60 *(n-1))	R/W	IFn Arbitration 1 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**
ID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ID							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**ID**	**Message Identifier 15-0** ID28 - ID0, 29-bit Identifier (“Extended Frame”). ID28 - ID18, 11-bit Identifier (“Standard Frame”)

IFn Arbitration 2 Register (CAN_IFn_ARB2)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_ARB2	CAN_BA+0x34 + (0x60 *(n-1))	R/W	IFn Arbitration 2 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
MsgVal	Xtd	Dir	ID				
7	6	5	4	3	2	1	0
ID							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**MsgVal**	**Message Valid** 0 = The Message Object is ignored by the Message Handler. 1 = The Message Object is configured and should be considered by the Message Handler. **Note:** The application software must reset the MsgVal bit of all unused Messages Objects during the initialization before it resets bit Init (CAN_CON[0]). This bit must also be reset before the identifier Id28-0 (CAN_IFn_ARB1/2), the control bits Xtd (CAN_IFn_ARB2[14]), Dir (CAN_IFn_ARB2[13]), or the Data Length Code DLC3-0 (CAN_IFn_MCON[3:0]) are modified, or if the Messages Object is no longer required.
[14]	**Xtd**	**Extended Identifier** 0 = The 11-bit (“standard”) Identifier will be used for this Message Object. 1 = The 29-bit (“extended”) Identifier will be used for this Message Object.
[13]	**Dir**	**Message Direction** 0 = Direction is receive. On TxRqst, a Remote Frame with the identifier of this Message Object is transmitted. On reception of a Data Frame with matching identifier, that message is stored in this Message Object. 1 = Direction is transmit. On TxRqst, the respective Message Object is transmitted as a Data Frame. On reception of a Remote Frame with matching identifier, the TxRqst bit (CAN_IFn_CMASK[2]) of this Message Object is set (if RmtEn (CAN_IFn_MCON[9]) = one).
[12:0]	**ID**	**Message Identifier 28-16** ID28 - ID0, 29-bit Identifier (“Extended Frame”). ID28 - ID18, 11-bit Identifier (“Standard Frame”)

IFn Message Control Register (CAN_IFn_MCON)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_MCON	CAN_BA+0x38 + (0x60 *(n-1))	R/W	IFn Message Control 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
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
7	6	5	4	3	2	1	0
EoB	Reserved			DLC			

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**NewDat**	**New Data** 0 = No new data has been written into the data portion of this Message Object by the Message Handler since last time this flag was cleared by the application software. 1 = The Message Handler or the application software has written new data into the data portion of this Message Object.
[14]	**MsgLst**	**Message Lost** 0 = No message lost since last time this bit was reset by the CPU. 1 = The Message Handler stored a new message into this object when NewDat was still set, the CPU has lost a message. **Note:** Only valid for Message Objects with direction = receive.
[13]	**IntPnd**	**Interrupt Pending** 0 = This message object is not the source of an interrupt. 1 = This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority.
[12]	**UMask**	**Use Acceptance Mask** 0 = Mask ignored. 1 = Use Mask (Msk28-0, MXtd, and MDir) for acceptance filtering. **Note:** If the UMask bit is set to one, the Message Object's mask bits have to be programmed during initialization of the Message Object before MsgVal bit (CAN_IFn_ARB2[15]) is set to one.
[11]	**TxIE**	**Transmit Interrupt Enable Bit** 0 = IntPnd (CAN_IFn_MCON[13]) will be left unchanged after the successful transmission of a frame. 1 = IntPnd will be set after a successful transmission of a frame.
[10]	**RxIE**	**Receive Interrupt Enable Bit** 0 = IntPnd (CAN_IFn_MCON[13]) will be left unchanged after a successful reception of a frame. 1 = IntPnd will be set after a successful reception of a frame.
[9]	**RmtEn**	**Remote Enable Bit** 0 = At the reception of a Remote Frame, TxRqst (CAN_IFn_MCON[8]) is left unchanged.

		1 = At the reception of a Remote Frame, TxRqst is set.
[8]	**TxRqst**	**Transmit Request** 0 = This Message Object is not waiting for transmission. 1 = The transmission of this Message Object is requested and is not yet done.
[7]	**EoB**	**End of Buffer** 0 = Message Object belongs to a FIFO Buffer and is not the last Message Object of that FIFO Buffer. 1 = Single Message Object or last Message Object of a FIFO Buffer. **Note:** This bit is used to concatenate two or more Message Objects (up to 32) to build a FIFO Buffer. For single Message Objects (not belonging to a FIFO Buffer), this bit must always be set to one.
[6:4]	**Reserved**	Reserved.
[3:0]	**DLC**	**Data Length Code** 0-8: Data Frame has 0-8 data bytes. 9-15: Data Frame has 8 data bytes **Note:** The Data Length Code of a Message Object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the Message Handler stores a data frame, it will write the DLC to the value given by the received message. Data(0): 1st data byte of a CAN Data Frame Data(1): 2nd data byte of a CAN Data Frame Data(2): 3rd data byte of a CAN Data Frame Data(3): 4th data byte of a CAN Data Frame Data(4): 5th data byte of a CAN Data Frame Data(5): 6th data byte of a CAN Data Frame Data(6): 7th data byte of a CAN Data Frame Data(7): 8th data byte of a CAN Data Frame **Note:** The Data(0) byte is the first data byte shifted into the shift register of the CAN Core during a reception while the Data(7) byte is the last. When the Message Handler stores a Data Frame, it will write all the eight data bytes into a Message Object. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by unspecified values.

IFn Data A1 Register (CAN_IFn_DAT_A1)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_DAT_A1	CAN_BA+0x3C + (0x60 *(n-1))	R/W	IFn Data A1 Registers (Register Map Note 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
Data(1)							
7	6	5	4	3	2	1	0
Data(0)							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**Data(1)**	**Data Byte 1** 2nd data byte of a CAN Data Frame
[7:0]	**Data(0)**	**Data Byte 0** 1st data byte of a CAN Data Frame

IFn Data A2 Register (CAN_IFn_DAT_A2)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_DAT_A2	CAN_BA+0x40 + (0x60 *(n-1))	R/W	IFn Data A2 Registers (Register Map Note 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
Data(3)							
7	6	5	4	3	2	1	0
Data(2)							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**Data(3)**	**Data Byte 3** 4th data byte of CAN Data Frame
[7:0]	**Data(2)**	**Data Byte 2** 3rd data byte of CAN Data Frame

IFn Data B1 Register (CAN_IFn_DAT_B1)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_DAT_B1	CAN_BA+0x44 + (0x60 *(n-1))	R/W	IFn Data B1 Registers (Register Map Note 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
Data(5)							
7	6	5	4	3	2	1	0
Data(4)							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**Data(5)**	**Data Byte 5** 6th data byte of CAN Data Frame
[7:0]	**Data(4)**	**Data Byte 4** 5th data byte of CAN Data Frame

IFn Data B2 Register (CAN_IFn_DAT_B2)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_DAT_B2	CAN_BA+0x48 + (0x60 *(n-1))	R/W	IFn Data B2 Registers (Register Map Note 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
Data(7)							
7	6	5	4	3	2	1	0
Data(6)							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**Data(7)**	**Data Byte 7** 8th data byte of CAN Data Frame.
[7:0]	**Data(6)**	**Data Byte 6** 7th data byte of CAN Data Frame.

In a CAN Data Frame, Data [0] is the first, Data [7] is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.

Message Object in the Message Memory

There are 32 Message Objects in the Message RAM. To avoid conflicts between application software access to the Message RAM and CAN message reception and transmission, the CPU cannot directly access the Message Objects, these accesses are handled through the IFn Interface Registers. Table 6.16-8 provides an overview of the structures of a Message Object.

Message Object												
UMask	Msk [28:0]	MXtd	MDir	EoB	NewDat		MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID [28:0]	Xtd	Dir	DLC [3:0]	Data(0)	Data(1)	Data(2)	Data(3)	Data(4)	Data(5)	Data(6)	Data(7)

Table 6.16-8 Structure of a Message Object in the Message Memory

The Arbitration Registers ID28-0 (CAN_IFn_ARB1/2), Xtd (CAN_IFn_ARB2[14]) and Dir (CAN_IFn_ARB2[13]) are used to define the identifier and type of outgoing messages and are used (together with the mask registers Msk28-0 (CAN_IFn_MASK1/2), MXtd (CAN_IFn_MASK2[15]) and MDir (CAN_IFn_MASK2[14])) for acceptance filtering of incoming messages. A received message is stored in the valid Message Object with matching identifier and Direction = receive (Data Frame) or Direction = transmit (Remote Frame). Extended frames can be stored only in Message Objects with Xtd = one, standard frames in Message Objects with Xtd = zero. If a received message (Data Frame or Remote Frame) matches with more than one valid Message Object, it is stored into that with the lowest message number.

Message Handler Registers

All Message Handler registers are read only. Their contents (TxRqst (CAN_IFn_MCON[8]), NewDat (CAN_IFn_MCON[15]), IntPnd (CAN_IFn_MCON[13]) and MsgVal (CAN_IFn_ARB2[15]) bits of each Message Object and the Interrupt Identifier) are status information provided by the Message Handler FSM.

Transmission Request Register 1 (CAN_TXREQ1)

These registers hold the TxRqst bits of the 32 Message Objects. By reading the TxRqst bits, the software can check which Message Object in a Transmission Request is pending. The TxRqst bit of a specific Message Object can be set/reset by the application software through the IFn Message Interface Registers or by the Message Handler after reception of a Remote Frame or after a successful transmission.

Register	Offset	R/W	Description	Reset Value
CAN_TXREQ1	CAN_BA+0x100	R	Transmission Request 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**
TxRqst16-9							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TxRqst8-1							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TxRqst16-1**	**Transmission Request Bits 16-1 (of All Message Objects) (Read Only)** 0 = This Message Object is not waiting for transmission. 1 = The transmission of this Message Object is requested and is not yet done.

Transmission Request Register 2 (CAN_TXREQ2)

Register	Offset	R/W	Description	Reset Value
CAN_TXREQ2	CAN_BA+0x104	R	Transmission Request 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
TxRqst32-25							
7	6	5	4	3	2	1	0
TxRqst24-17							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TxRqst32-17**	**Transmission Request Bits 32-17 (of All Message Objects) (Read Only)** 0 = This Message Object is not waiting for transmission. 1 = The transmission of this Message Object is requested and is not yet done.

New Data Register 1 (CAN_NDAT1)

These registers hold the NewDat bits of the 32 Message Objects. By reading out the NewDat bits, the software can check for which Message Object the data portion was updated. The NewDat bit of a specific Message Object can be set/reset by the software through the IFn Message Interface Registers or by the Message Handler after reception of a Data Frame or after a successful transmission.

Register	Offset	R/W	Description	Reset Value
CAN_NDAT1	CAN_BA+0x120	R	New Data 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**
NewData16-9							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NewData8-1							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**NewData16-1**	**New Data Bits 16-1 (of All Message Objects)** 0 = No new data has been written into the data portion of this Message Object by the Message Handler since the last time this flag was cleared by the application software. 1 = The Message Handler or the application software has written new data into the data portion of this Message Object.

New Data Register 2 (CAN_NDAT2)

Register	Offset	R/W	Description	Reset Value
CAN_NDAT2	CAN_BA+0x124	R	New Data 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
NewData32-25							
7	6	5	4	3	2	1	0
NewData24-17							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**NewData32-17**	**New Data Bits 32-17 (of All Message Objects)** 0 = No new data has been written into the data portion of this Message Object by the Message Handler since the last time this flag was cleared by the application software. 1 = The Message Handler or the application software has written new data into the data portion of this Message Object.

Interrupt Pending Register 1 (CAN_IPND1)

These registers contain the IntPnd bits of the 32 Message Objects. By reading the IntPnd bits, the software can check for which Message Object an interrupt is pending. The IntPnd bit of a specific Message Object can be set/reset by the application software through the IFn Message Interface Registers or by the Message Handler after reception or after a successful transmission of a frame. This will also affect the value of IntId in the Interrupt Register.

Register	Offset	R/W	Description	Reset Value
CAN_IPND1	CAN_BA+0x140	R	Interrupt Pending 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**
IntPnd16-9							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IntPnd8-1							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**IntPnd16-1**	**Interrupt Pending Bits 16-1 (of All Message Objects)** 0 = This message object is not the source of an interrupt. 1 = This message object is the source of an interrupt.

Interrupt Pending Register 2 (CAN_IPND2)

Register	Offset	R/W	Description	Reset Value
CAN_IPND2	CAN_BA+0x144	R	Interrupt Pending 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**
IntPnd32-25							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IntPnd24-17							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**IntPnd32-17**	**Interrupt Pending Bits 32-17 (of All Message Objects)** 0 = This message object is not the source of an interrupt. 1 = This message object is the source of an interrupt.

Message Valid Register 1 (CAN_MVLD1)

These registers hold the MsgVal bits of the 32 Message Objects. By reading the MsgVal bits, the application software can check which Message Object is valid. The MsgVal bit of a specific Message Object can be set/reset by the application software via the IFn Message Interface Registers.

Register	Offset	R/W	Description	Reset Value
CAN_MVLD1	CAN_BA+0x160	R	Message Valid 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**
MsgVal16- 9							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MsgVal8-1							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**MsgVal16-1**	**Message Valid Bits 16-1 (of All Message Objects) (Read Only)** 0 = This Message Object is ignored by the Message Handler. 1 = This Message Object is configured and should be considered by the Message Handler. **Note:** CAN_MVLD1[0] means Message object No.1 is valid or not. If CAN_MVLD1[0] is set, message object No.1 is configured.

Message Valid Register 2 (CAN_MVLD2)

Register	Offset	R/W	Description	Reset Value
CAN_MVLD2	CAN_BA+0x164	R	Message Valid 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**
MsgVal32-25							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MsgVal24-17							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**MsgVal32-17**	**Message Valid Bits 32-17 (of All Message Objects) (Read Only)** 0 = This Message Object is ignored by the Message Handler. 1 = This Message Object is configured and should be considered by the Message Handler. **Note:** CAN_MVLD2[15] means Message object No.32 is valid or not. If CAN_MVLD2[15] is set, message object No.32 is configured.

Wake-up Enable Control Register (CAN_WU_EN)

Register	Offset	R/W	Description	Reset Value
CAN_WU_EN	CAN_BA+0x168	R/W	Wake-up 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							**WAKUP_EN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WAKUP_EN**	**Wake-up Enable Bit** 0 = The wake-up function Disabled. 1 = The wake-up function Enabled. **Note:** User can wake up system when there is a falling edge in the CAN_Rx pin.

Wake-up Status Register (CAN_WU_STATUS)

Register	Offset	R/W	Description	Reset Value
CAN_WU_STATUS	CAN_BA+0x16C	R/W	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							**WAKUP_STS**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WAKUP_STS**	**Wake-up Status** 0 = No wake-up event occurred. 1 = Wake-up event occurred. **Note:** This bit can be cleared by writing '0' to it.

6.17 CRC Controller (CRC)

6.17.1 Overview

The Cyclic Redundancy Check (CRC) generator can perform CRC calculation with four common polynomials CRC-CCITT, CRC-8, CRC-16, and CRC-32 settings.

6.17.2 Features

- Supports four common polynomials CRC-CCITT, CRC-8, CRC-16, and CRC-32
  - CRC-CCITT: $X^{16} + X^{12} + X^5 + 1$
  - CRC-8: $X^8 + X^2 + X + 1$
  - CRC-16: $X^{16} + X^{15} + X^2 + 1$
  - CRC-32: $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- 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 using PDMA to write data to perform CRC operation

6.17.3 Block Diagram

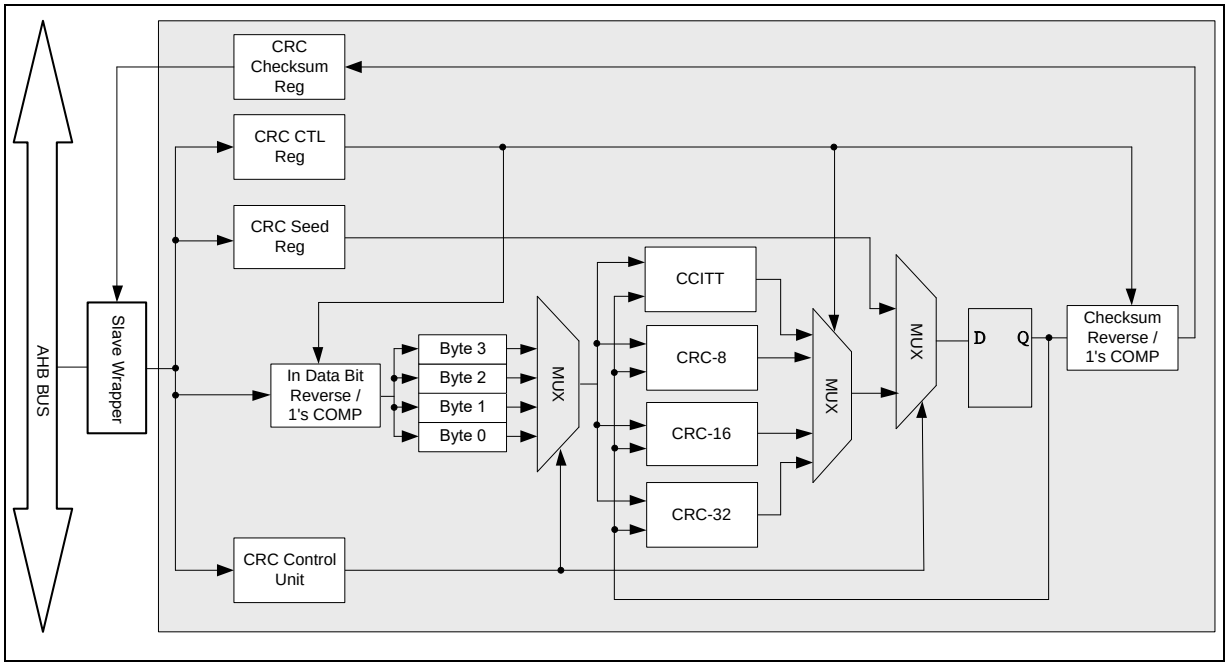


Figure 6.17-1 CRC Generator Block Diagram

6.17.4 Basic Configuration

- Clock Source Configuration

- Enable CRC peripheral clock in CRCCKEN (CLK_AHBCLK[7]).

- Reset Configuration
  - Reset CRC controller in CRCRST (SYS_IPRST0[7]).

6.17.5 Functional Description

CRC generator can perform CRC calculation with four common polynomial settings. The operation polynomial includes CRC-CCITT, CRC-8, CRC-16 and CRC-32; User can choose the CRC operation polynomial mode by setting CRCMODE[1:0] (CRC_CTL[31:30] CRC Polynomial Mode).

The following is a program sequence example.

1. Enable CRC generator by setting CRCEN (CRC_CTL[0] CRC Channel Enable Bit).
2. Initial setting for CRC calculation.
   1) Configure 1's complement for CRC checksum by setting CHKSFMT (CRC_CTL[27] Checksum 1's Complement).
   2) Configure bit order reverse for CRC checksum by setting CHKSREV (CRC_CTL[25] Checksum Bit Order Reverse). The funcitonal block is also shown in Figure 6.17-2 CHECKSUM Bit Order Reverse Functional Block
   3) Configure 1's complement for CRC write data by setting DATFMT (CRC_CTL[26] Write Data 1's Complement).
   4) 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 6.17-3.
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.

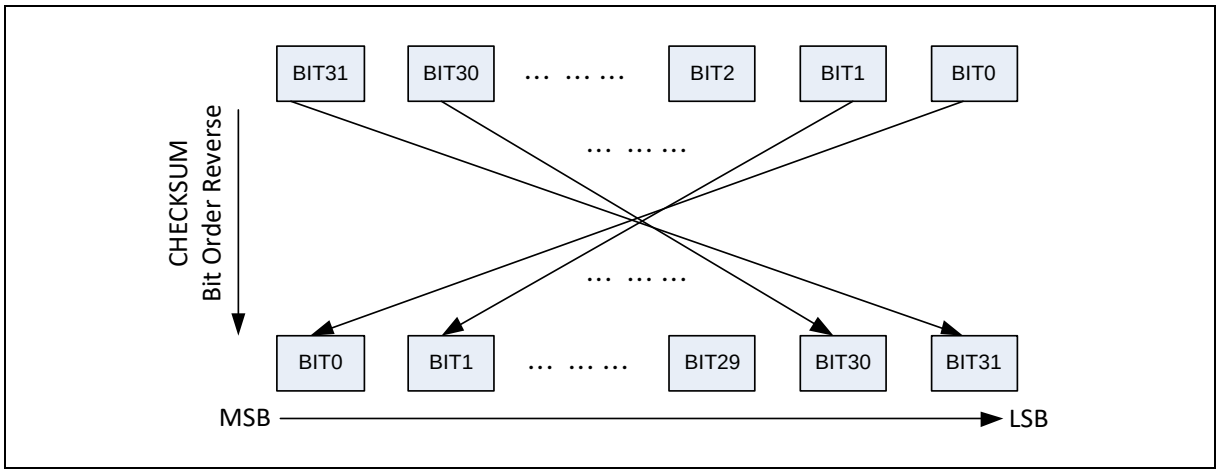


Figure 6.17-2 CHECKSUM Bit Order Reverse Functional Block

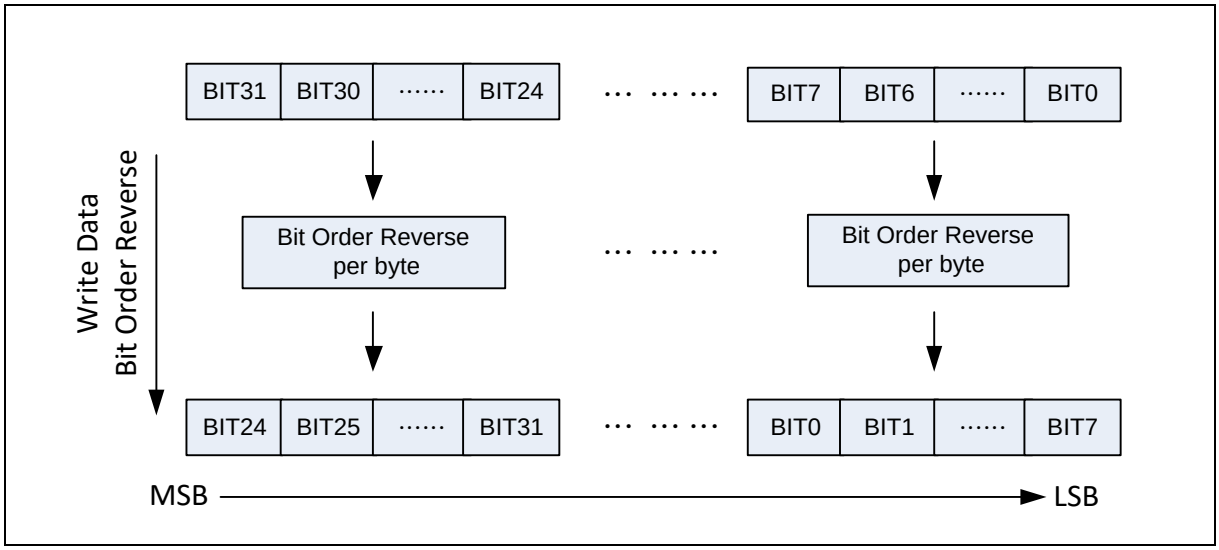


Figure 6.17-3 Write Data Bit Order Reverse Functional Block

6.17.6 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 = 0x4003_1000**				
CRC_CTL	CRC_BA+0x00	R/W	CRC Control Register	0x2000_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

6.17.7 Register Description

CRC Control Register (CRC_CTL)

Register	Offset	R/W	Description	Reset Value
CRC_CTL	CRC_BA+0x00	R/W	CRC Control Register	0x2000_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.** **00 = CRC-CCITT Polynomial mode.** **01 = CRC-8 Polynomial mode.** **10 = 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 mode or use PDMA function 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.

6.18 Hardware Divider (HDIV)

6.18.1 Overview

The hardware divider (HDIV) is useful to the high performance application. The hardware divider is a signed, integer divider with both quotient and remainder outputs.

6.18.2 Features

- Signed (two's complement) integer calculation
- 32-bit dividend with 16-bit divisor calculation capacity
- 32-bit quotient and 32-bit remainder outputs (16-bit remainder with sign extends to 32-bit)
- Divided by zero warning flag
- Write divisor to trigger calculation

6.18.3 Basic Configuration

Before using the hardware divider, the clock of hardware divider must be enabled. To enable hardware divider, it needs to set HDIV_EN on AHBCLK[4] to 1.

6.18.4 Functional Description

To use hardware divider, it needs to set dividend first. Then set divisor and the hardware divider will trigger calculation automatically after divisor written. The calculation results including the quotient and remainder could be obtained by reading DIVQUO and DIVREM register. User can read quotient and remainder after one cycle of writing the divisor.

DIV0 flag of DIVSTS will be set if divisor is 0.

The dividend is 32-bit signed integer and divisor is 16-bit signed integer. The quotient is 32-bit signed integer and the remainder is 16-bit signed integer.

Figure 6.18-1 shows the operation flow of hardware divider. To calculation X / Y, CPU needs to write X to DIVIDEND register, and then write Y to DIVISOR. CPU can read DIVQUO and DIVREM registers to get calculation results after DIVISOR been written.

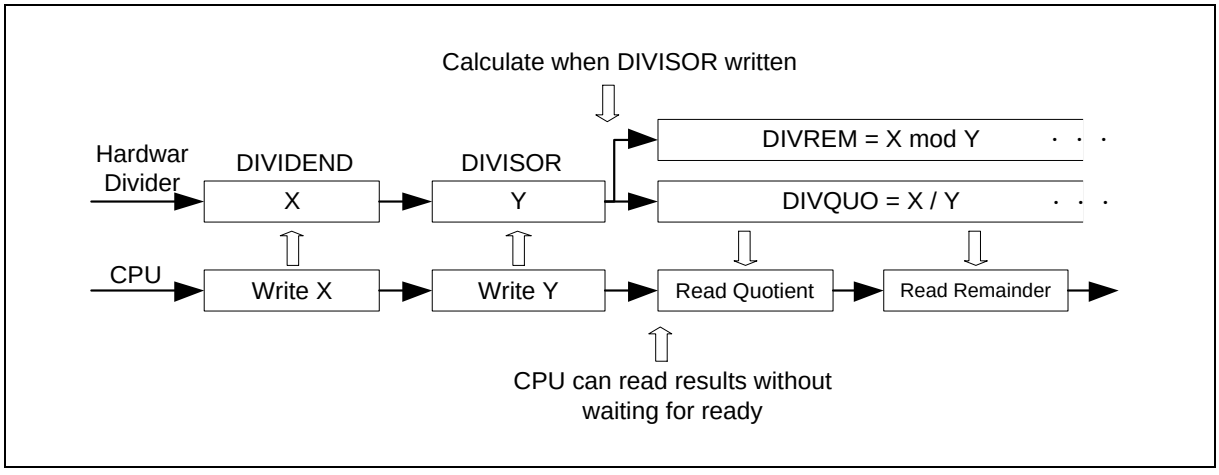


Figure 6.18-1 Hardware Divider Operation Flow

6.18.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
HDIV Base Address: **HDIV_BA = 0x4001_4000**				
DIVIDEND	HDIV_BA+0x00	R/W	Dividend Source Register	0x0000_0000
DIVISOR	HDIV_BA+0x04	R/W	Divisor Source Resister	0x0000_FFFF
DIVQUO	HDIV_BA+0x08	R/W	Quotient Result Resister	0x0000_0000
DIVREM	HDIV_BA+0x0C	R/W	Remainder Result Register	0x0000_0000
DIVSTS	HDIV_BA+0x10	R	Divider Status Register	0x0000_0001

6.18.6 Register Description

Dividend Source Register (DIVIDEND)

Register	Offset	R/W	Description	Reset Value
DIVIDEND	HDIV_BA+0x00	R/W	Dividend Source Register	0x0000_0000

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

Bits	Description	
[31:0]	**DIVIDEND**	**Dividend Source** This register is given the dividend of divider before calculation started.

Divisor Source Register (DIVISOR)

Register	Offset	R/W	Description	Reset Value
DIVISOR	HDIV_BA+0x04	R/W	Divisor Source Resister	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**
DIVISOR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVISOR							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DIVISOR**	**Divisor Source** This register is given the divisor of divider before calculation starts. **Note:** When this register is written, hardware divider will start calculation.

Quotient Result Register (DIVQUO)

Register	Offset	R/W	Description	Reset Value
DIVQUO	HDIV_BA+0x08	R/W	Quotient Result Resister	0x0000_0000

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

Bits	Description	
[31:0]	**QUOTIENT**	**Quotient Result** This register holds the quotient result of divider after calculation is complete.

Remainder Result Register (DIVREM)

Register	Offset	R/W	Description	Reset Value
DIVREM	HDIV_BA+0x0C	R/W	Remainder Result Register	0x0000_0000

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

Bits	Description	
[31:16]	**REMAINDER[31:16]**	**Sign Extension of REMAINDER[15:0]** The remainder of hardware divider is 16-bit sign integer (REMAINDER[15:0]) with sign extension (REMAINDER[31:16]) to 32-bit integer.
[15:0]	**REMAINDER[15:0]**	**Remainder Result** This register holds the remainder result of divider after calculation is complete.

Divider Status Register (DIVSTS)

Register	Offset	R/W	Description	Reset Value
DIVSTS	HDIV_BA+0x10	R	Divider Status 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						**DIV0**	**Reserved**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**DIV0**	**Divisor Zero Warning (Read Only)** 0 = The divisor is not 0. 1 = The divisor is 0. **Note:** The DIV0 flag is used to indicate divide-by-zero situation and updated whenever DIVISOR is written. This register is read only.
[0]	**Reserved**	Reserved.

6.19 Analog-to-Digital Converter (ADC)

6.19.1 Overview

The chip contains one 12-bit successive approximation analog-to-digital converter (SAR A/D converter) with twenty one input channels. The A/D converter supports four operation modes: Single, Burst, Single-cycle Scan and Continuous Scan mode. The A/D converter can be started by software, external pin, timer0~3 overflow pulse trigger and PWM trigger.

6.19.2 Features

- Analog input voltage range: 0 ~ AV_{DD} (voltage of V_{DD} pin).
- 12-bit resolution and 10-bit accuracy is guaranteed
- Up to 17 single-end analog input channels or 8 differential analog input channels
- Maximum ADC peripheral clock frequency is 16 MHz
- Up to 800 kSPS sampling rate
- Configurable ADC internal sampling time
- Four operation modes:
  - Single mode: A/D conversion is performed one time on a specified channel.
  - Burst mode: A/D converter samples and converts the specified single channel and sequentially stores the result in FIFO.
  - 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 (STADC)
  - Timer 0~3 overflow pulse trigger
  - PWM trigger with optional start delay period
- Each conversion result is 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.
- 4 internal channels, they are band-gap voltage (V_{BG}), temperature sensor (V_{TEMP}), internal reference voltage and DAC0 output
- Support PDMA transfer mode.

Note 1: ADC sampling rate = (ADC peripheral clock frequency) / (total ADC conversion cycle)

Note 2: If the internal channel for temperature sensor is active, the minimum time of sample stage should be T_{TEMP_ADC}. Please refer to section 8.5.7 in the relative Datasheet for detailed information.

Note 3: If the internal channel for band-gap voltage is active, the minimum time of sample stage should be T_{VBG_ADC}. Please refer to section 8.2 in the relative Datasheet for detailed information.

6.19.3 Block Diagram

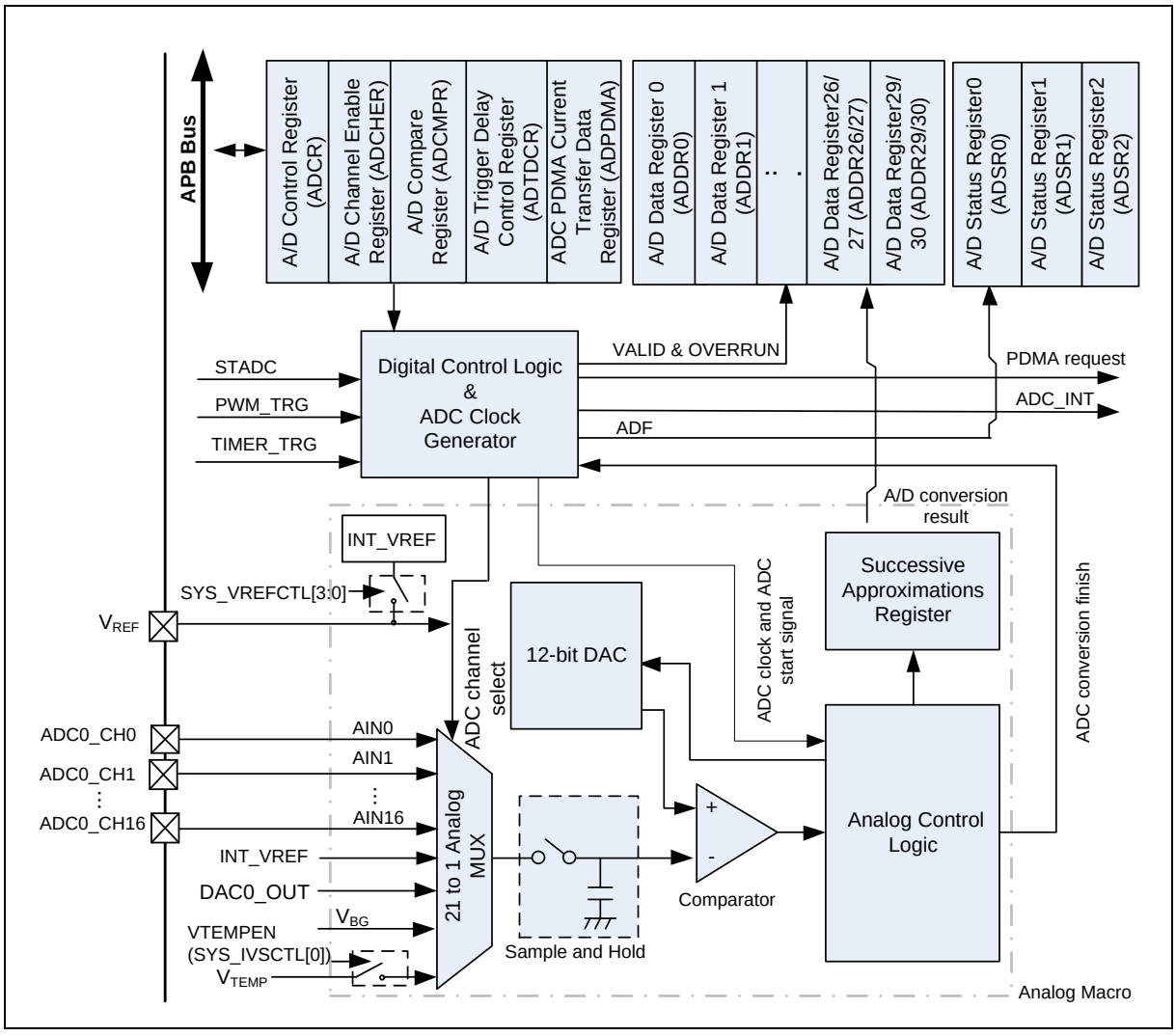


Figure 6.19-1 AD Controller Block Diagram

6.19.4 Basic Configuration

The ADC pin functions are configured in SYS_GPA_MFP0, SYS_GPA_MFP1, SYS_GPB_MFP1, SYS_GPC_MFP0, SYS_GPC_MFP1 registers. It is recommended to disable the digital input path of the analog input pins to avoid the leakage current. User can disable the digital input path by configuring PA_DINOFF, PB_DINOFF, PC_DINOFF registers.

The ADC peripheral clock can be enabled in ADCCKEN(APBCLK0[28]). The ADC peripheral clock source is selected by ADCSEL(CLKSEL2[21:20]). The clock prescalar is determined by ADCDIV(CLKDIV0[23:16]).

6.19.5 Functional Description

The A/D converter operates by successive approximation with 12-bit resolution. The ADC has four operation modes: Single, Burst, 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(ADC_ADCR[11]) bit to 0 in advance.

6.19.5.1 ADC peripheral Clock Generator

The maximum sampling rate is up to 800K SPS. The ADC has four clock sources selected by ADCSEL (CLKSEL2[21:20]), the ADC peripheral clock frequency is divided by an 8-bit pre-scalar with the following formula:

ADC peripheral clock frequency = (ADC peripheral clock source frequency) / (ADCDIV+1); where the 8-bit ADCDIV is located in register CLKDIV0[23:16].

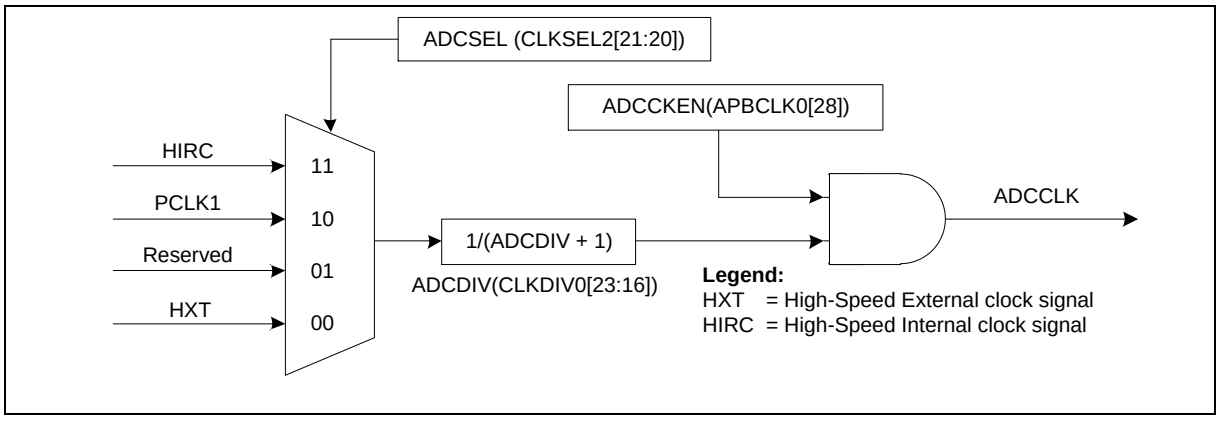


Figure 6.19-2 ADC Peripheral Clock Control

6.19.5.2 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 ADC_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 ADC data register corresponding to the channel.
3. The ADF bit of ADC_ADSR0 register will be set to 1. If the ADIE bit of ADC_ADCR register is set to 1, the ADC 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 ADC conversion done, ADC cannot finish the current conversion, 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 below.

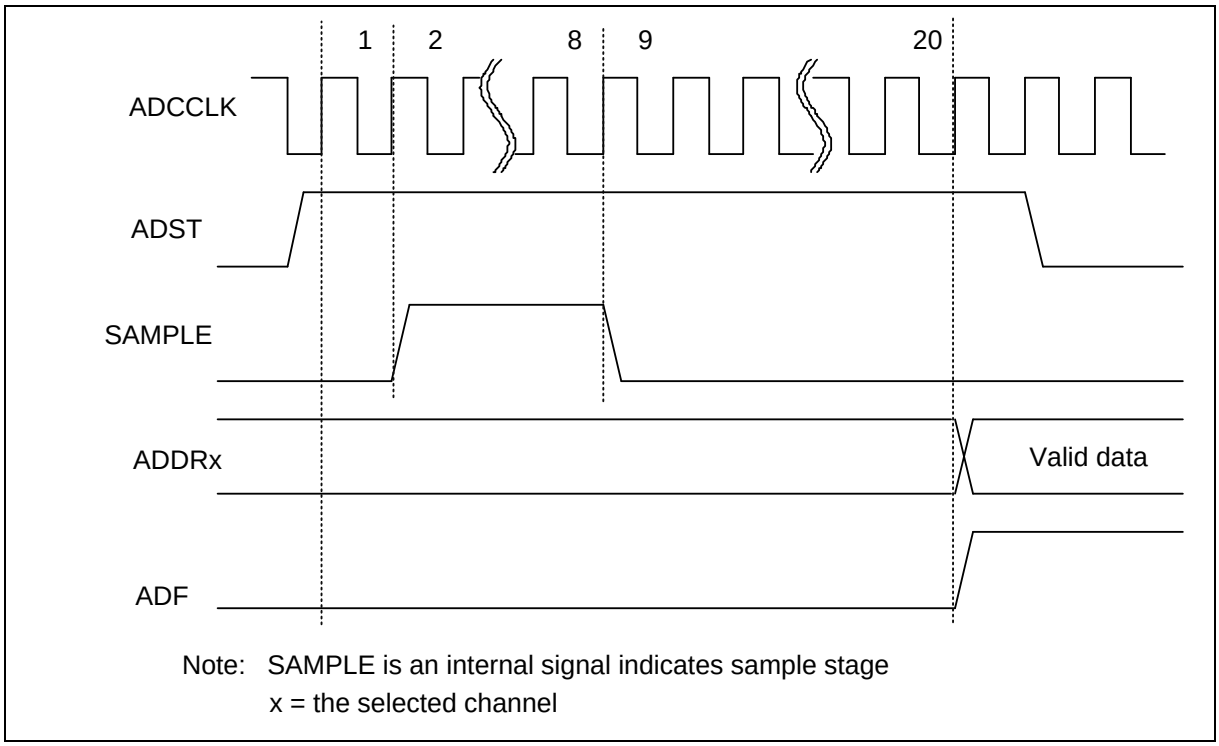


Figure 6.19-3 Single Mode Conversion Timing Diagram

6.19.5.3 Burst Mode

In Burst mode, A/D converter samples and converts the specified single channel and sequentially stores the result into FIFO (up to 16 samples). The operations are as follows:

1. When the ADST bit in ADC_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 the specified channel is completed, the result is sequentially transferred to FIFO and can be accessed only from the ADC data register 0.
3. When more than or equal to 8 samples in FIFO, the ADF bit in ADC_ADSR0 register is set to 1. If the ADIE bit of ADC_ADCR register is set to 1 at this time, an ADC interrupt is requested after finishing the A/D conversion.
4. Steps 2 to 3 are repeated as long as the ADST bit remains 1. When the ADST bit is cleared to 0, ADC cannot finish the current conversion and A/D converter enters idle state directly

An example timing diagram for Burst mode is shown below.

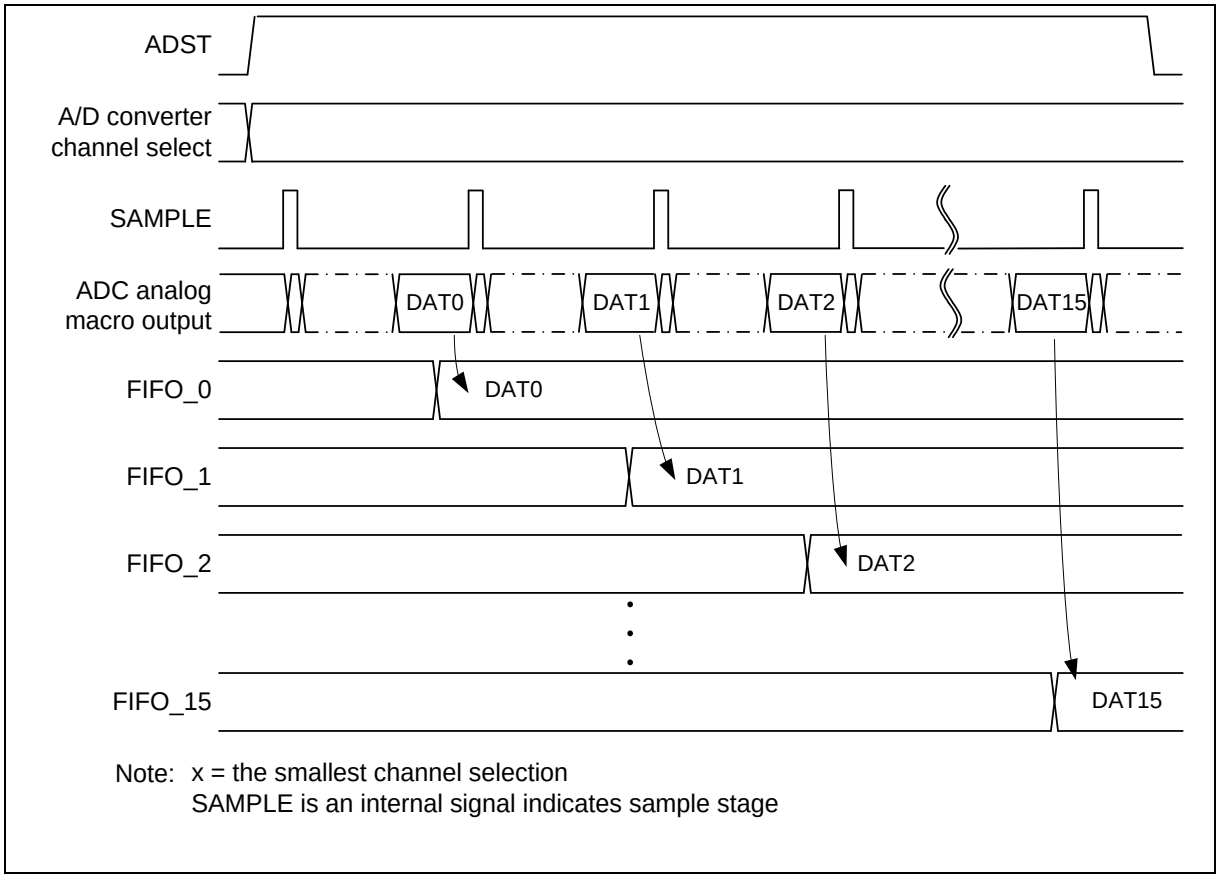


Figure 6.19-4 Burst Mode Conversion Timing Diagram

Note 1: If software enables more than one channel in Burst mode, only the channel with the smallest number is converted and other enabled channels will be ignored.

Note 2: User can obtain the conversion results by reading the ADC data register 0 (ADC_ADDR0) repeatedly until VALIDF (ADC_ADSR0[8]) turns to 0. For example, if there are 8 conversion results at FIFO, it needs to read ADC data register 0 (ADC_ADDR0) 8 times to get all of result.

6.19.5.4 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 ADC_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 ADC data register corresponding to each channel.
3. When the conversions of all the enabled channels are completed, the ADF bit in ADC_ADSR0 register is set to 1. If the ADC interrupt function is enabled, the ADC interrupt occurs.
4. After ADC 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 ADC channels conversion done, ADC 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 below.

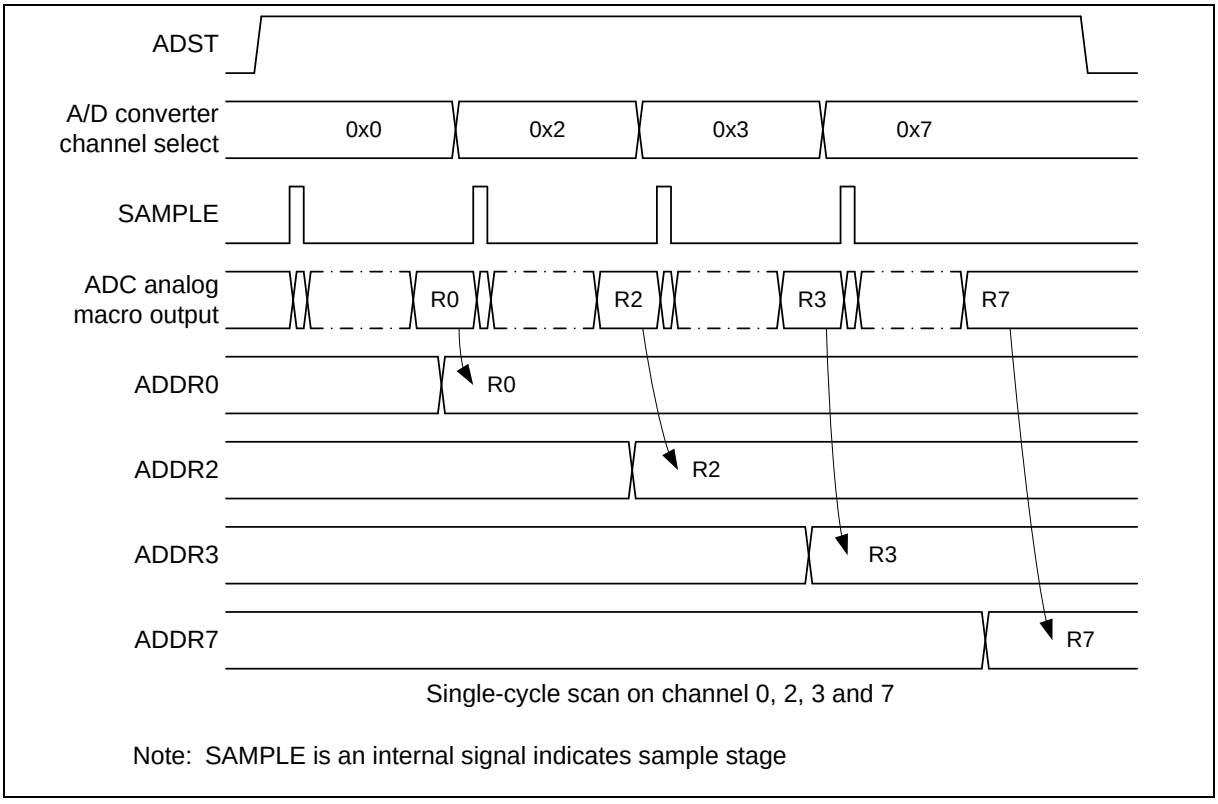


Figure 6.19-5 Single-Cycle Scan Mode on Enabled Channels Timing Diagram

6.19.5.5 Continuous Scan Mode

In Continuous Scan mode, A/D conversion is performed sequentially on the specified channels that enabled by CHEN bits in ADC_ADCHER register (maximum 21 channels for ADC). The operations are as follows:

1. When the ADST bit in ADC_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 ADC data register corresponding to each enabled channel.
3. When A/D converter completes the conversions of all enabled channels sequentially, the ADF bit in ADC_ADSR0 register will be set to 1. If the ADC interrupt function is enabled, the ADC 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, ADC 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 below.

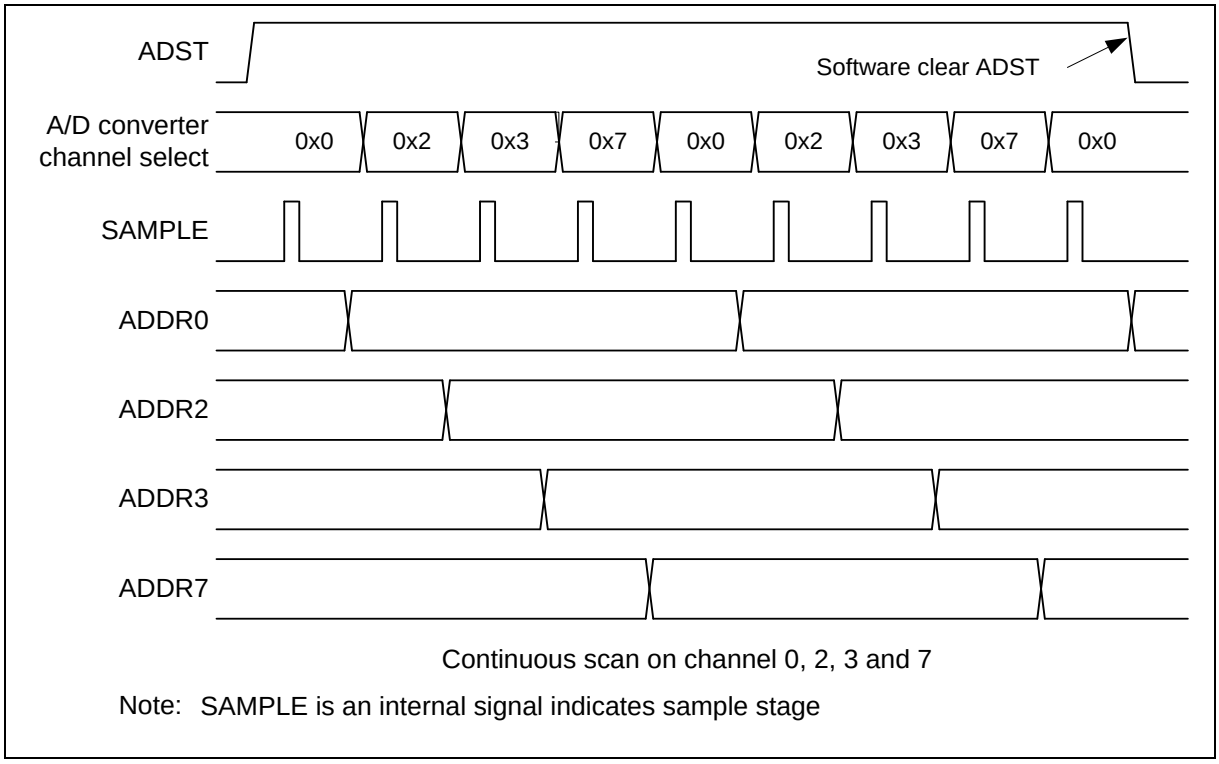


Figure 6.19-6 Continuous Scan Mode on Enabled Channels Timing Diagram

6.19.5.6 External Trigger Input

In Single-cycle Scan mode, A/D conversion can be triggered by external pin request. When the TRGEN bit of ADC_ADCR register is set to 1 to enable ADC external trigger function, setting the TRGS(ADC_ADCR[5:4]) bits to 2'b00 is to select external trigger input from the STADC pin. Software can set TRGCOND(ADC_ADCR[7:6]) to select trigger condition is falling/rising edge or low/high level. If level trigger condition is selected, the STADC 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 ADC must be at least 4 PCLKs after enabling ADC peripheral clock.

6.19.5.7 Timer trigger

There are 4 Timer trigger sources for ADC. When the TRGEN bit of ADC_ADCR register is set to high to enable ADC external hardware trigger function, setting the TRGS (ADC_ADCR[5:4]) bits to 2'b01 is to select external hardware trigger input source from Timer trigger. The detail trigger conditions of Timer are described descripted at TIMER0_CTL ~ TIMER3_CTL register.

6.19.5.8 PWM trigger

In Single-cycle Scan mode, A/D conversion can be triggered by PWM request. When the TRGEN bit of ADC_ADCR register is set to high to enable ADC external hardware trigger function, setting the TRGS(ADC_ADCR[5:4]) bits to 2'b11 is to select external hardware trigger input source from PWM trigger. When PWM trigger is enabled, setting PTDT (ADC_ADTDCR[7:0]) bits can insert a delay time between PWM trigger condition and ADC start conversion.

6.19.5.9 Conversion Result Monitor by Compare Mode Function

The ADC controller provides two compare registers, ADC_ADCMPRx(x=0,1), to monitor maximum two

specified channels. Software can select which channel to be monitored by setting CMPCH (ADC_ADCMPRx[7:3]). CMPCOND (ADC_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 (ADC_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 (ADC_ADCMPRx[27:16]). When the conversion of the channel specified by CMPCH(ADC_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 (CMPMATCNT+1) then CMPF0/1 (ADC_ADSR0[1]/[2]) bit will be set to 1, if CMPIE (ADC_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 below.

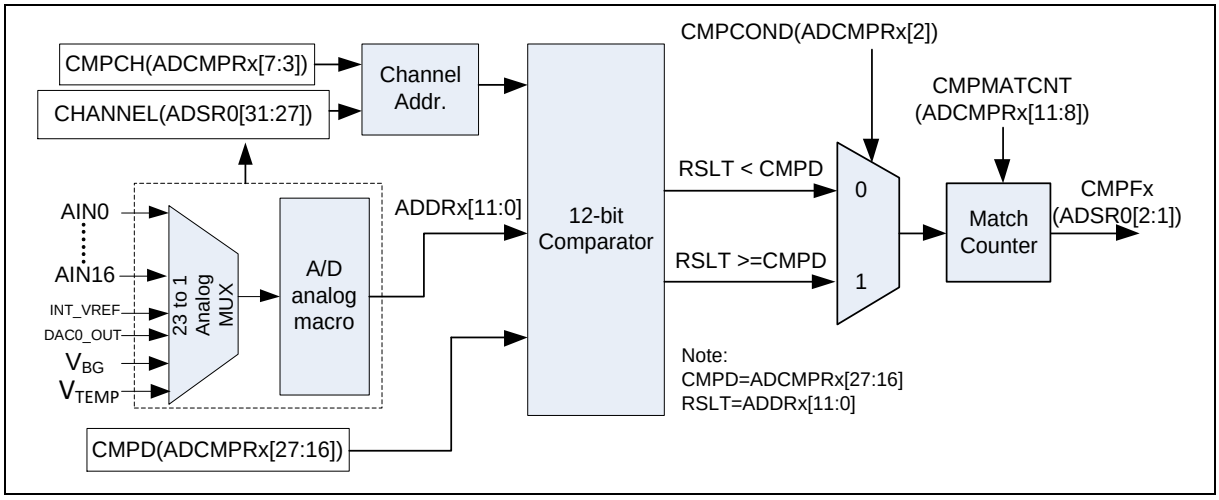


Figure 6.19-7 A/D Conversion Result Monitor Logic Diagram

6.19.5.10 Compare Window Mode

The ADC controller supports a compare window mode. User can set CMPWEN (ADC_ADCMPR0[15]) to enable this function. If user enables this function, CMPF0 (ADC_ADSR0[1]) will be set only when compared conditions of two conversion result monitor logic are matched and CMPF1 (ADC_ADSR0[2]) will always be zero. The range of compare window is between CMPD (ADC_ADCMPR0[27:16]) and CMPD (ADC_ADCMPR1[27:16]).

6.19.5.11 PDMA Transfer Mode

When A/D conversion is finished, the conversion result will be loaded into ADC_ADDRx(x=0~16, 26,27,29,30) register and VALID(ADC_ADDRx[17]) bit will be set to 1. If the PTEN(ADC_ADCR[9]) bit is set, ADC controller will generate a request to PDMA. User can use PDMA to transfer the conversion results to a user-specified memory space without CPU's intervention. The source address of PDMA operation is fixed at ADC_ADPDMA, no matter what channels was selected. When PDMA is transferring the conversion result, ADC will continue converting the next selected channel if the operation mode of ADC is burst mode, single scan mode or continuous scan mode. User can monitor current PDMA transfer data through reading ADC_ADPDMA register. If ADC completes the conversion of a selected channel and the last conversion result of the same channel has not been transferred by PDMA, OVERRUN(ADC_ADSR2[31:0]) bit of the corresponding channel will be set and the last ADC conversion result will be overwritten by the new ADC conversion result. PDMA will transfer the latest data of selected channels to the user-specified destination address.

6.19.5.12 Interrupt Sources

There are three interrupt sources of ADC interrupt. When an ADC operation mode finishes its conversion, the A/D conversion end flag, ADF(ADC_ADSR0[0]), will be set to 1. The

CMPF0(ADC_ADSR0[1]) and CMPF1(ADC_ADSR0[2]) are the compare flags of compare function. When the conversion result meets the settings of ADC_ADCMPR0/1 registers, the corresponding flag will be set to 1. When one of the flags, ADF, CMPF0 and CMPF1, is set to 1 and the corresponding interrupt enable bit, ADIE of ADC_ADCR register and CMPIE of ADC_ADCMPR0/1 registers, is set to 1, the ADC interrupt will be asserted. Software can clear these flags to revoke the interrupt request.

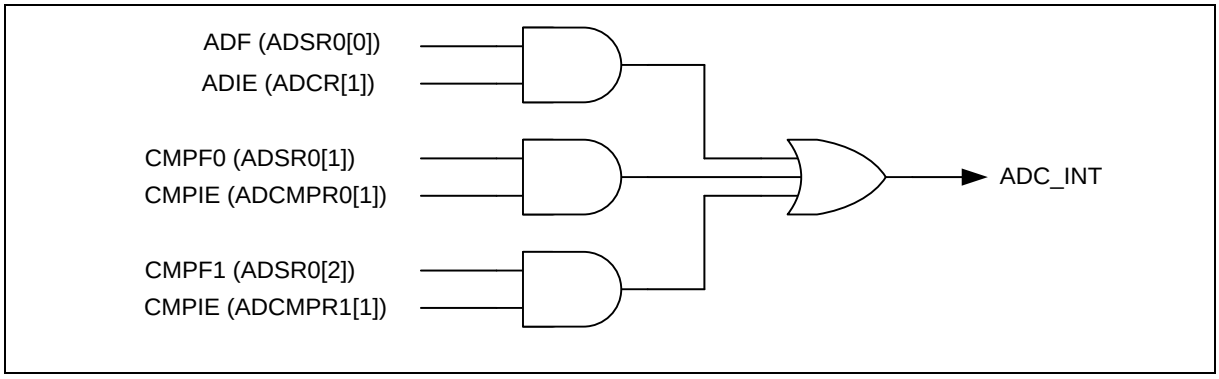


Figure 6.19-8 A/D Controller Interrupt

6.19.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
ADC Base Address: **ADC_BA = 0x4004_3000**				
ADC_ADDR0	ADC_BA+0x00	R	ADC Data Register 0	0x0000_0000
ADC_ADDR1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000
ADC_ADDR2	ADC_BA+0x08	R	ADC Data Register 2	0x0000_0000
ADC_ADDR3	ADC_BA+0x0C	R	ADC Data Register 3	0x0000_0000
ADC_ADDR4	ADC_BA+0x10	R	ADC Data Register 4	0x0000_0000
ADC_ADDR5	ADC_BA+0x14	R	ADC Data Register 5	0x0000_0000
ADC_ADDR6	ADC_BA+0x18	R	ADC Data Register 6	0x0000_0000
ADC_ADDR7	ADC_BA+0x1C	R	ADC Data Register 7	0x0000_0000
ADC_ADDR8	ADC_BA+0x20	R	ADC Data Register 8	0x0000_0000
ADC_ADDR9	ADC_BA+0x24	R	ADC Data Register 9	0x0000_0000
ADC_ADDR10	ADC_BA+0x28	R	ADC Data Register 10	0x0000_0000
ADC_ADDR11	ADC_BA+0x2C	R	ADC Data Register 11	0x0000_0000
ADC_ADDR12	ADC_BA+0x30	R	ADC Data Register 12	0x0000_0000
ADC_ADDR13	ADC_BA+0x34	R	ADC Data Register 13	0x0000_0000
ADC_ADDR14	ADC_BA+0x38	R	ADC Data Register 14	0x0000_0000
ADC_ADDR15	ADC_BA+0x3C	R	ADC Data Register 15	0x0000_0000
ADC_ADDR16	ADC_BA+0x40	R	ADC Data Register 16	0x0000_0000
ADC_ADDR26	ADC_BA+0x68	R	ADC Data Register 26	0x0000_0000
ADC_ADDR27	ADC_BA+0x6C	R	ADC Data Register 27	0x0000_0000
ADC_ADDR29	ADC_BA+0x74	R	ADC Data Register 29	0x0000_0000
ADC_ADDR30	ADC_BA+0x78	R	ADC Data Register 30	0x0000_0000
ADC_ADCR	ADC_BA+0x80	R/W	ADC Control Register	0x0005_0000
ADC_ADCHER	ADC_BA+0x84	R/W	ADC Channel Enable Register	0x0000_0000
ADC_ADCMPR0	ADC_BA+0x88	R/W	ADC Compare Register 0	0x0000_0000
ADC_ADCMPR1	ADC_BA+0x8C	R/W	ADC Compare Register 1	0x0000_0000
ADC_ADSR0	ADC_BA+0x90	R/W	ADC Status Register0	0x0000_0000
ADC_ADSR1	ADC_BA+0x94	R	ADC Status Register1	0x0000_0000

ADC_ADSR2	ADC_BA+0x98	R	ADC Status Register2	0x0000_0000
ADC_ADTDCR	ADC_BA+0x9C	R/W	ADC Trigger Delay Control Register	0x0000_0000
ADC_ADPDMA	ADC_BA+0x100	R	ADC PDMA Current Transfer Data Register	0x0000_0000

6.19.7 Register Description

ADC Data Registers (ADC_ADDRx x = 0~16, 26,27,29,30)

Register	Offset	R/W	Description	Reset Value
ADC_ADDR0	ADC_BA+0x00	R	ADC Data Register 0	0x0000_0000
ADC_ADDR1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000
ADC_ADDR2	ADC_BA+0x08	R	ADC Data Register 2	0x0000_0000
ADC_ADDR3	ADC_BA+0x0C	R	ADC Data Register 3	0x0000_0000
ADC_ADDR4	ADC_BA+0x10	R	ADC Data Register 4	0x0000_0000
ADC_ADDR5	ADC_BA+0x14	R	ADC Data Register 5	0x0000_0000
ADC_ADDR6	ADC_BA+0x18	R	ADC Data Register 6	0x0000_0000
ADC_ADDR7	ADC_BA+0x1C	R	ADC Data Register 7	0x0000_0000
ADC_ADDR8	ADC_BA+0x20	R	ADC Data Register 8	0x0000_0000
ADC_ADDR9	ADC_BA+0x24	R	ADC Data Register 9	0x0000_0000
ADC_ADDR10	ADC_BA+0x28	R	ADC Data Register 10	0x0000_0000
ADC_ADDR11	ADC_BA+0x2C	R	ADC Data Register 11	0x0000_0000
ADC_ADDR12	ADC_BA+0x30	R	ADC Data Register 12	0x0000_0000
ADC_ADDR13	ADC_BA+0x34	R	ADC Data Register 13	0x0000_0000
ADC_ADDR14	ADC_BA+0x38	R	ADC Data Register 14	0x0000_0000
ADC_ADDR15	ADC_BA+0x3C	R	ADC Data Register 15	0x0000_0000
ADC_ADDR16	ADC_BA+0x40	R	ADC Data Register 16	0x0000_0000
ADC_ADDR26	ADC_BA+0x68	R	ADC Data Register 26	0x0000_0000
ADC_ADDR27	ADC_BA+0x6C	R	ADC Data Register 27	0x0000_0000
ADC_ADDR29	ADC_BA+0x74	R	ADC Data Register 29	0x0000_0000
ADC_ADDR30	ADC_BA+0x78	R	ADC Data Register 30	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 overwrote. 1 = Data in RSLT bits is overwrote.
[15:0]	**RSLT**	**A/D Conversion Result (Read Only)** This field contains conversion result of ADC.

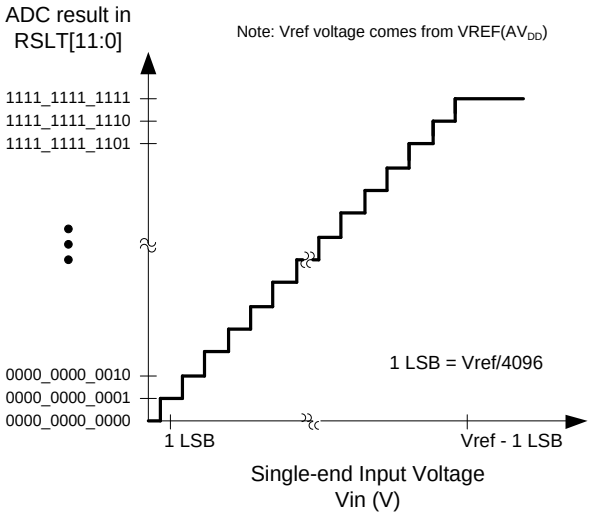


Figure 6.19-9 Conversion Result Mapping Diagram of ADC Single-end Input

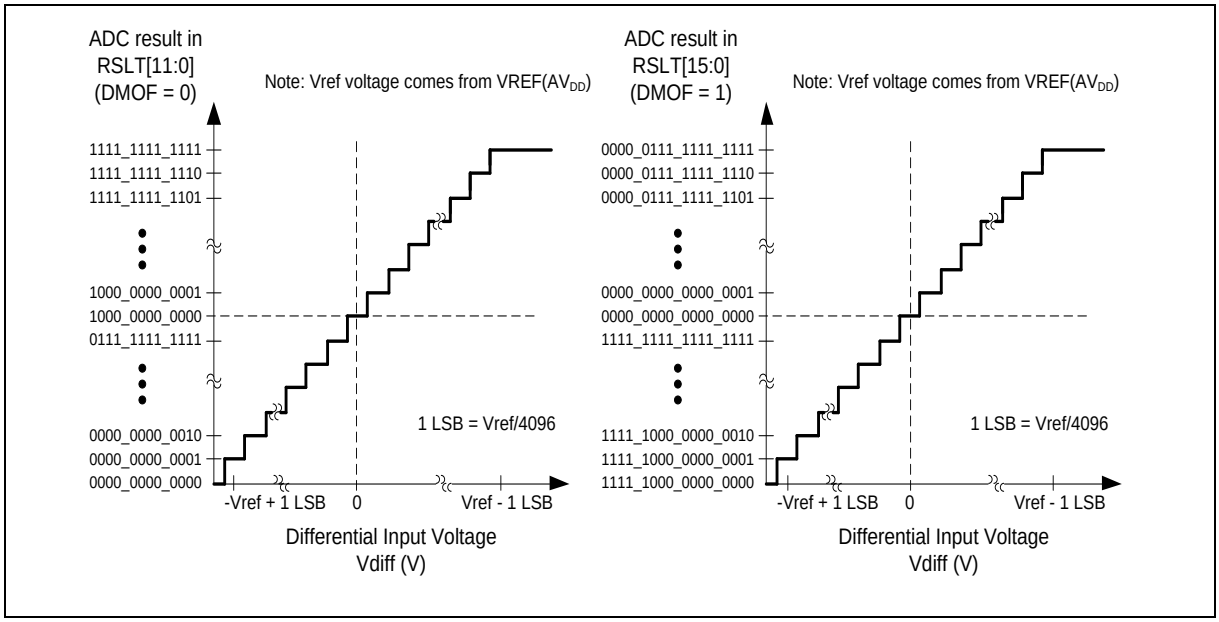


Figure 6.19-10 Conversion Result Mapping Diagram of ADC Differential Input

ADC Control Register (ADC_ADCR)

Register	Offset	R/W	Description	Reset Value
ADC_ADCR	ADC_BA+0x80	R/W	ADC Control Register	0x0005_0000

31	30	29	28	27	26	25	24
DMOF	Reserved						
23	22	21	20	19	18	17	16
Reserved					SMPTSEL		
15	14	13	12	11	10	9	8
Reserved				ADST	DIFFEN	PTEN	TRGEN
7	6	5	4	3	2	1	0
TRGCOND		TRGS		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.
[30:19]	**Reserved**	Reserved.
[18:16]	**SMPTSEL**	**ADC Internal Sampling Time Selection** Total ADC conversion cycle = sampling cycle + 12. 000 = 4 ADC clock for sampling; 16 ADC clock for complete conversion. 001 = 5 ADC clock for sampling; 17 ADC clock for complete conversion. 010 = 6 ADC clock for sampling; 18 ADC clock for complete conversion. 011 = 7 ADC clock for sampling; 19 ADC clock for complete conversion. 100 = 8 ADC clock for sampling; 20 ADC clock for complete conversion. 101 = 9 ADC clock for sampling; 21 ADC clock for complete conversion. 110 = 10 ADC clock for sampling; 22 ADC clock for complete conversion. 111 = 11 ADC clock for sampling; 23 ADC clock for complete conversion.
[15:12]	**Reserved**	Reserved.
[11]	**ADST**	**A/D Conversion Start** ADST bit can be set to 1 from four sources: software, external pin STADC, PWM trigger and Timer trigger. ADST bit will be cleared to 0 by hardware automatically at the ends of Single mode and Single-cycle Scan mode. In Continuous Scan mode and Burst 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.
[10]	**DIFFEN**	**Differential Input Mode Control** Differential input voltage (V_{diff}) = V_{plus} - V_{minus},. where V_{plus} is the analog input; V_{minus} is the inverted analog input. The V_{plus} of differential input paired channel x is from ADC0_CHy pin; V_{minus} is from ADC0_CHz pin, x=0,1..7,

		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 ADC_ ADCHER register. The conversion result will be placed to the corresponding data register of the enabled channel. **Note:** The relation between Vplus and Vminus is Vplus + Vminus = Vref
[9]	**PTEN**	**PDMA Transfer Enable Bit** When A/D conversion is completed, the converted data is loaded into ADDR0~16 ADDR26, ADDR27, ADDR29, ADDR30. Software can enable this bit to generate a PDMA data transfer request. 0 = PDMA data transfer Disabled. 1 = PDMA data transfer in ADDR0~16, ADDR26, ADDR27, ADDR29, ADDR30 Enabled. **Note:** When PTEN=1, software must set ADIE=0 to disable interrupt.
[8]	**TRGEN**	**External Trigger Enable Control** Enable or disable triggering of A/D conversion by external STADC pin, PWM trigger 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 STADC 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]	**TRGS**	**Hardware Trigger Source** 00 = A/D conversion is started by external STADC pin. 01 = Timer0 ~ Timer3 overflow pulse trigger. 10 = Reserved. 11 = A/D conversion is started by PWM trigger. **Note:** Software should clear TRGEN bit and ADST bit to 0 before changing TRGS bits.
[3:2]	**ADMD**	**A/D Converter Operation Mode Control** 00 = Single conversion. 01 = Burst conversion. 10 = Single-cycle Scan. 11 = Continuous Scan. **Note 1:** When changing the operation mode, software should clear ADST bit first. **Note 2:** In Burst mode, the A/D result data is always at ADC Data Register 0.
[1]	**ADIE**	**A/D Interrupt Enable Control** A/D conversion end interrupt request is generated if ADIE bit is set to 1. 0 = A/D interrupt function Disabled. 1 = A/D 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 (ADC_ADCHER)

Register	Offset	R/W	Description	Reset Value
ADC_ADCHER	ADC_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 ADC_ ADCHER[16:0] bits to enable the corresponding analog input channel 16 ~ 0. If DIFFEN bit is set to 1, only the even number channel needs to be enabled. Besides, set ADC_ ADCHER[26], ADC_ ADCHER[27], ADC_ ADCHER[29], ADC_ ADCHER[30] bits will enable internal channel for internal reference voltage, DAC0_OUT, band-gap voltage and temperature sensor respectively. Other bits are reserved. 0 = Channel Disabled. 1 = Channel Enabled. **Note 1**: If the internal channel for temperature sensor is active, the minimum time of sample stage should be T_{TEMP_ADC}. Please refer to section 8.5.7 in the relative Datasheet for detailed information. **Note 2**: If the internal channel for band-gap voltage is active, the minimum time of sample stage should be T_{VBG_ADC}. Please refer to section 8.2 in the relative Datasheet for detailed information.

ADC Compare Register 0/1 (ADC_ADCMPR0/1)

Register	Offset	R/W	Description	Reset Value
ADC_ADCMPR0	ADC_BA+0x88	R/W	ADC Compare Register 0	0x0000_0000
ADC_ADCMPR1	ADC_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			CMPMATCNT			
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]	**CMPMATCNT**	**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 (CMPMATCNT +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. 11010 = Internal reference voltage conversion result is selected to be compared 11011 = DAC0 output conversion result is selected to be compared 11101 = Band-gap voltage conversion result is selected to be compared. 11110 = Temperature sensor 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 (CMPMATCNT +1), the CMPFx bit will be set.
[1]	**CMPIE**	**Compare Interrupt Enable Control** If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMATCNT, 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 Control** Set this bit to 1 to enable ADC controller 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 (ADC_ADSR0)

Register	Offset	R/W	Description	Reset Value
ADC_ADSR0	ADC_BA+0x90	R/W	ADC Status Register0	0x0000_0000

31	30	29	28	27	26	25	24
CHANNEL					Reserved		
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.
[26:17]	**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 then 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. 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. 3. When more than or equal to 8 samples in FIFO in Burst mode.

ADC Status Register1 (ADC_ADSR1)

Register	Offset	R/W	Description	Reset Value
ADC_ADSR1	ADC_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[30:29], VALID[27:26], VALID[16:0] are the mirror of the VALID bits in ADDR30[17], ADDR29[17], ADDR27[17], ADDR26[17], ADDR16[17]~ ADDR0[17]. The other bits are reserved. **Note:** When ADC is in burst mode and any conversion result is valid, VALID[30:29], VALID[27:26], VALID[16:0] will be set to 1.

ADC Status Register2 (ADC_ADSR2)

Register	Offset	R/W	Description	Reset Value
ADC_ADSR2	ADC_BA+0x98	R	ADC Status Register2	0x0000_0000

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

Bits	Description	
[31:0]	**OVERRUN**	**Overrun Flag (Read Only)** OVERRUN[30:29], OVERRUN[27:26], OVERRUN[16:0] are the mirror of the OVERRUN bit in ADDR30[16], ADDR29[16], ADDR27[16], ADDR26[16], ADDR16[16] ~ ADDR0[16]. The other bits are reserved. **Note:** When ADC is in burst mode and the FIFO is overrun, OVERRUN[30:29], OVERRUN[27:26], OVERRUN[16:0] will be set to 1.

ADC Trigger Delay Control Register (ADC_ADTDCR)

Register	Offset	R/W	Description	Reset Value
ADC_ADTDCR	ADC_BA+0x9C	R/W	ADC Trigger Delay 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**
PTDT							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**PTDT**	**PWM Trigger Delay Time** Set this field will delay ADC start conversion time after PWM trigger. PWM trigger delay time is (4 * PTDT) * system clock

ADC PDMA Current Transfer Data Register (ADC_ADPDMA)

Register	Offset	R/W	Description	Reset Value
ADC_ADPDMA	ADC_BA+0x100	R	ADC PDMA 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 PDMA Current Transfer Data Register (Read Only)** When PDMA transferring, read this register can monitor current PDMA transfer data. Current PDMA transfer data could be the content of ADDR0 ~ ADDR16 and ADDR26, ADDR27, and ADDR29, ADDR30 registers.

6.20 Digital to Analog Converter (DAC)

6.20.1 Overview

The DAC module is a 5-bit, voltage output digital-to-analog converter. It 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.

6.20.2 Features

- Supports one 5-bit 100k SPS voltage type DAC
- Analog output voltage range: 0~AV_{DD} (voltage of V_{DD} pin)
- Reference voltage from internal reference voltage V_{REF} pin or AV_{DD}.
- DAC maximum conversion updating rate 100k SPS
- Rail to rail settle time 10us
- Supports software and timer0~3 trigger to start DAC conversion.
- Supports PDMA mode.

6.20.3 Block Diagram

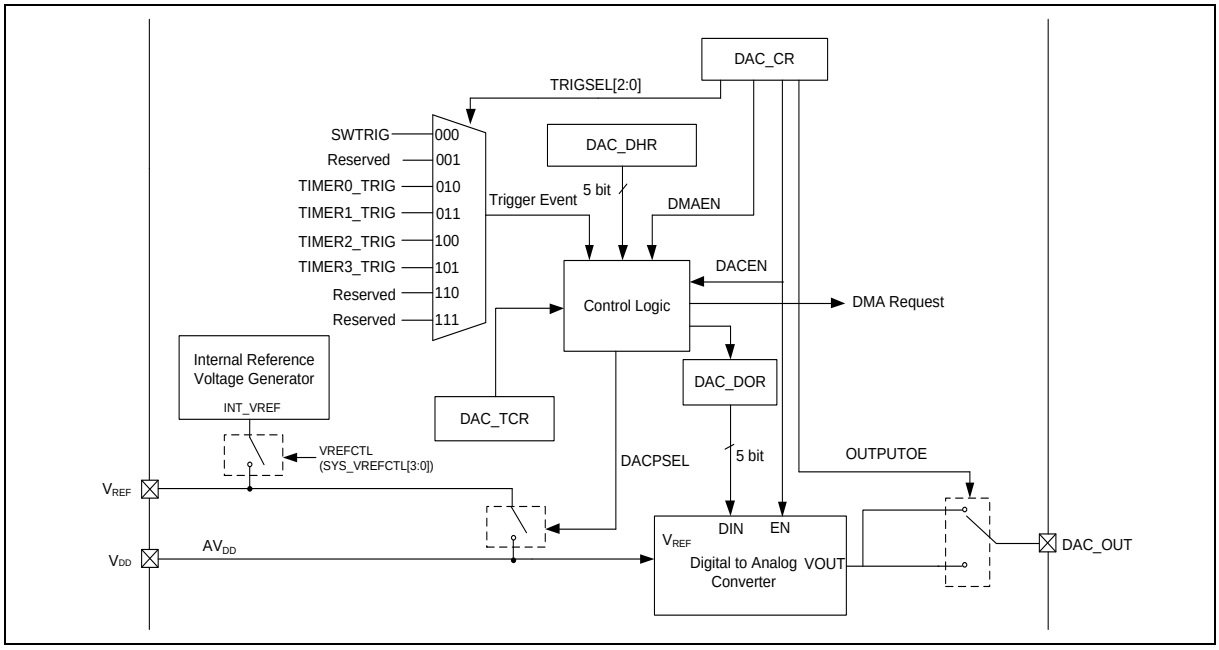


Figure 6.20-1 Digital-to-Analog Converter Block Diagram

6.20.4 Basic Configuration

The DAC output pin is configured in SYS_GPA_MFP0[7:0] Multi-function Register. The DAC Controller clock source is enabled by DACCKEN (CLK_APBCLK1[12]).

6.20.5 Functional Description

6.20.5.1 DAC Output

The DAC is a 5-bit voltage output digital-to-analog converter. The maximum DAC output voltage is limited to the selected reference voltage source.

6.20.5.2 DAC Reference Voltage

The DAC reference voltage is shared with ADC reference voltage and it is configured by VREFCTL(SYS_VREFCTL[3:0]) 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) or analog power AV_{DD}.

6.20.5.3 DAC Conversion

Any data transfer to the DAC channel is performed by loading the data into DAC_DAT register. Figure 6.20-2 shows the DAC conversion started by software write operation. When user writes the conversion data to data holding register DAC_DAT, the data is loaded into data output register DAC_DATOUT by hardware and DAC starts data conversion Figure 6.20-3 shows the DAC conversion started by hardware trigger (timer trigger event). The data stored in the DAC_DAT register is automatically transferred to the data output buffer DAC_DATOUT after occurring one PCLK (APB clock) the event.

When DAC data output register DAC_DATOUT is loaded with the DAC_DAT contents, the analog output voltage becomes available after specified conversion settling time. The conversion settling time is 10us when 5-bit input code transition from lowest code (0x00) to highest code (0x1F). 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 (DAC_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 48 MHz and DAC conversion settling time is 10us, the selected SETTLET value must be greater than 0x1E1. When the conversion is started, the conversion finish flag FINISH (DAC_STATUS[0]) is cleared to 0 by hardware and set to 1 after the time counter counts to SETTLET.

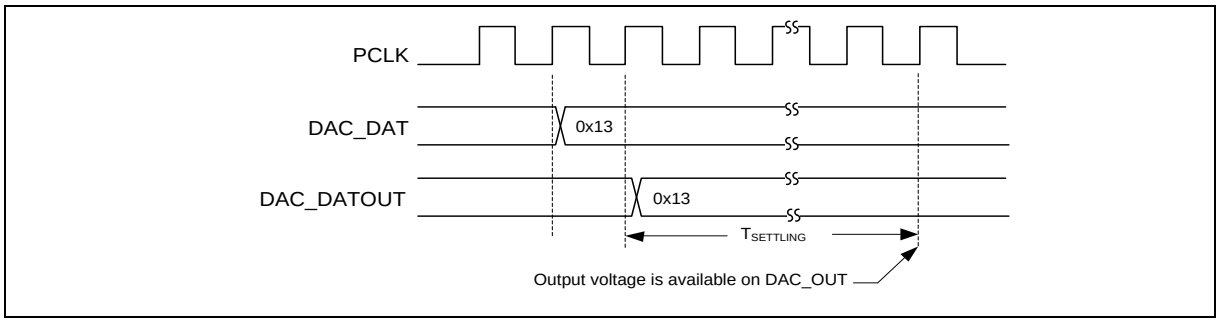


Figure 6.20-2 DAC Conversion Started by Software Write Trigger

6.20.5.4 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:

$$DAC_OUT = V_{REF} \times DATOUT[4:0]/32$$

6.20.5.5 DAC Trigger Selection

The DAC conversion can be started by writing DAC_DAT, software trigger or hardware trigger. When TRGEN (DAC_CTL[4]) is 0, the data conversion is started by writing DAC_DAT register. When TRGEN (DAC_CTL[4]) is 1, the data conversion is started by timer event or software trigger. If the software trigger is selected, the conversion starts once the SWTRG (DAC_SWTRG[0]) is set to 1. The SWTRG is cleared to 0 by hardware automatically when DAC_DATOUT has been loaded with DAC_DAT content. The TRGSEL (DAC_CTL[7:5]) determines which event is selected to start the conversion.

When DAC detects a rising edge on the selected trigger event input, the last data stored in DAC_DAT is transferred into the DAC_DATOUT[4:0] and DAC starts converting once DAC_DATOUT[4:0] is updated.

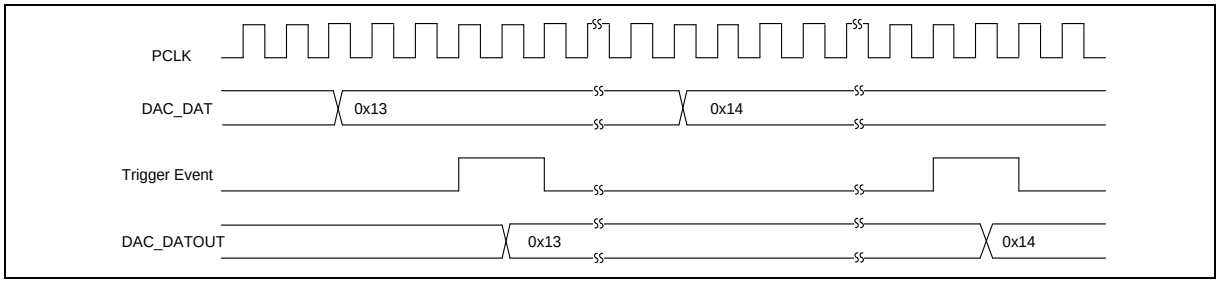


Figure 6.20-3 DAC Conversion Started by Hardware Trigger Event

6.20.5.6 PDMA Operation

DAC to PDMA request is generated when a hardware trigger event occurs while DMAEN (DAC_CTL[2]) is set. The content of DAC_DAT is transferred to the DAC_DATOUT[4:0] and DAC starts data conversion. The new transferred data by PDMA in DAC_DAT will be converted when next trigger event arrives. Figure 6.20-4 shows the DAC PDMA underrun condition, when the second DAC to PDMA request trigger event arrives before the first conversion finish, then no new PDMA request is issued and DMA underrun flag DMAUDR (DAC_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 (DAC_CTL[3]) is enabled. User has to change the trigger event frequency in timer and then start DAC conversion again.

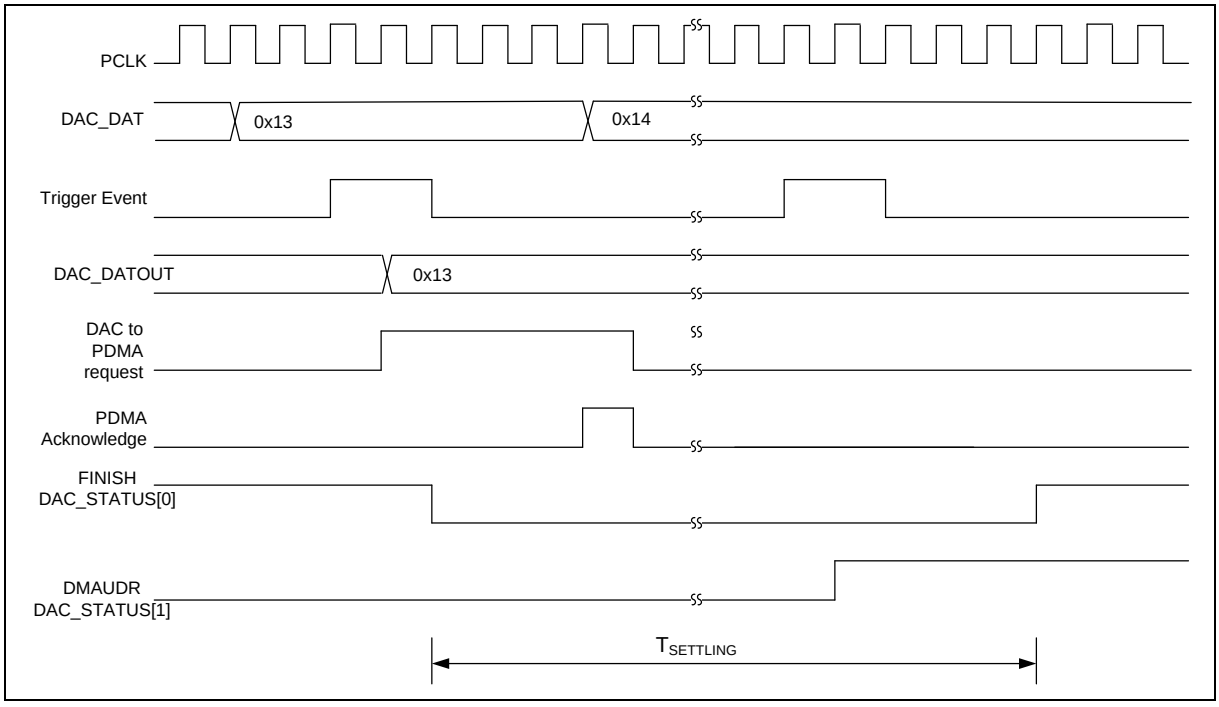


Figure 6.20-4 DAC PDMA Underrun Condition Example

DMA request can also be generated by software enable, user sets DMAEN (DAC_CTL[2]) to 1 and TRGEN (DAC_CTL[4]) to 0, DMA request is generated periodically according to the conversion time defined by SETTLET (DAC_TCTL[9:0]) value. DAC output is updated periodically. When user clears DMAEN (DAC_CTL[2]) to 0, DAC controller will stop issuing next new PDMA transfer request.

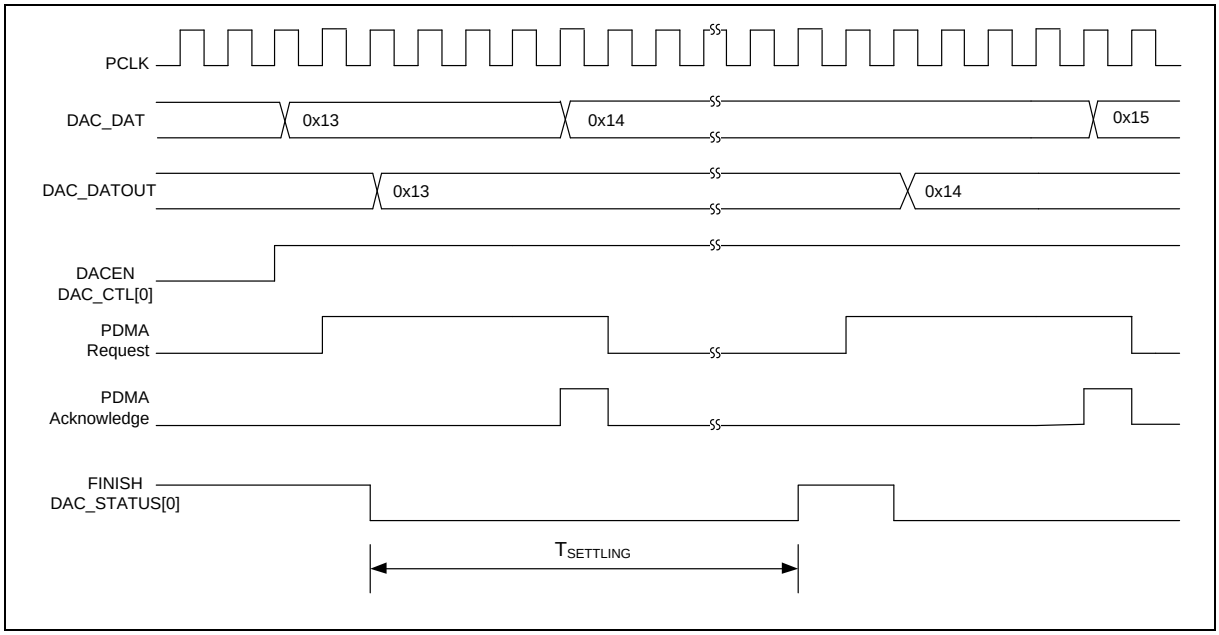


Figure 6.20-5 DAC Continuous Conversion with Software PDMA Mode

6.20.5.7 Interrupt Sources

There are two interrupt sources in DAC controller, one is DAC data conversion finish interrupt and the other is DMA under-run interrupt. When DAC conversion finish, the FINISH (DAC_STATUS[0]) is set to 1 and an interrupt occurs while DACIEN (DAC_CTL[1]) is enabled. If new DMA trigger event occurs during DAC data conversion period, the DMA under run flag DMAUDR (DAC_STATUS[1]) is generated and an interrupt occurs if DMAURIEN (DAC_CTL[3]) is enabled.

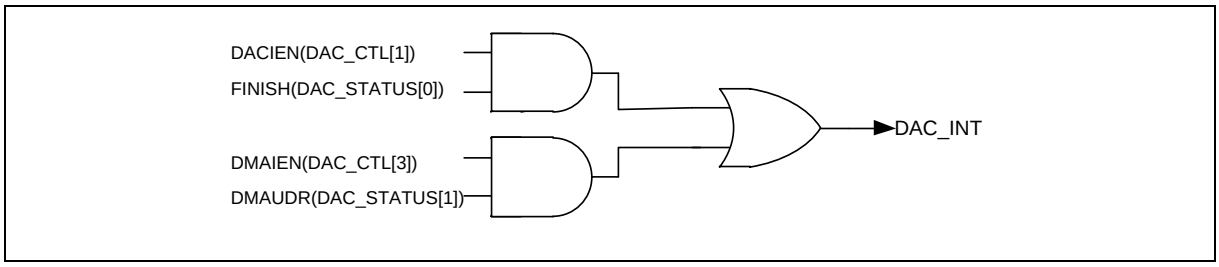


Figure 6.20-6 DAC Interrupt Source

6.20.6 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**				
DAC_CTL	DAC_BA+0x00	R/W	DAC Control Register	0x0000_0000
DAC_SWTRG	DAC_BA+0x04	R/W	DAC Software Trigger Control Register	0x0000_0000
DAC_DAT	DAC_BA+0x08	R/W	DAC Data Holding Register	0x0000_0000
DAC_DATOUT	DAC_BA+0x0C	R	DAC Data Output Register	0x0000_0000
DAC_STATUS	DAC_BA+0x10	R/W	DAC Status Register	0x0000_0000
DAC_TCTL	DAC_BA+0x14	R/W	DAC Timing Control Register	0x0000_0000

6.20.7 Register Description

DAC Control Register (DAC_CTL)

Register	Offset	R/W	Description	Reset Value
DAC_CTL	DAC_BA+0x00	R/W	DAC 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						**DACPSEL**	**OUTPUTOE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRGSEL			**TRGEN**	**DMAURIEN**	**DMAEN**	**DACIEN**	**DACEN**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**DACPSEL**	**DAC Reference Voltage Selection** 0 = Select AV_{DD} (voltage of V_{DD} pin) 1 = Select V_{REF}
[8]	**OUTPUTOE**	**DAC Output Enable** 1 = DAC otuput to PAD Enabled. 0 = DAC output to PAD disabled.
[7:5]	**TRGSEL**	**Trigger Source Selection** 000 = Software trigger. 001 = reserved 010 = Timer 0 trigger. 011 = Timer 1 trigger. 100 = Timer 2 trigger. 101 = Timer 3 trigger. 110 = reserved 111 = reserved
[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 underrun interrupt Disabled. 1 = DMA underrun interrupt Enabled.
[2]	**DMAEN**	**DMA Mode Enable Bit** 0 = DMA mode Disabled. 1 = DMA mode Enabled.

[1]	**DACIEN**	**DAC Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[0]	**DACEN**	**DAC Enable Bit** 0 = DAC Disabled. 1 = DAC Enabled.

DAC Software Trigger Control Register (DAC_SWTRG)

Register	Offset	R/W	Description	Reset Value
DAC_SWTRG	DAC_BA+0x04	R/W	DAC 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. 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.

DAC Data Holding Register (DAC_DAT)

Register	Offset	R/W	Description	Reset Value
DAC_DAT	DAC_BA+0x08	R/W	DAC 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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**DACDAT**				

Bits	Description	
[31:16]	**Reserved**	Reserved.
[4:0]	**DACDAT**	**DAC 5-bit Holding Data** These bits are written by user software which specifies 5-bit conversion data for DAC output.

DAC Data Output Register (DAC_DATOUT)

Register	Offset	R/W	Description	Reset Value
DAC_DATOUT	DAC_BA+0x0C	R	DAC 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**DATOUT**				

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:0]	**DATOUT**	**DAC 5-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.

DAC Status Register (DAC_STATUS)

Register	Offset	R/W	Description	Reset Value
DAC_STATUS	DAC_BA+0x10	R/W	DAC 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. This is read only bit.
[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. User writes 1 to clear this bit.
[0]	**FINISH**	**DAC Conversion Complete Finish Flag** 0 = DAC is in conversion state. 1 = DAC conversion finish. 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.

DAC Timing Control Register (DAC_TCTL)

Register	Offset	R/W	Description	Reset Value
DAC_TCTL	DAC_BA+0x14	R/W	DAC 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 setting time is 1 us, SETTLET value must be greater than 0x48.

6.21 Analog Comparator Controller (ACMP)

6.21.1 Overview

The chip provides two 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.

6.21.2 Features

- Analog input voltage range: 0 ~ AV_{DD} (voltage of V_{DD} pin)
- Up to two rail-to-rail analog comparators
- Supports hysteresis function
  - Support programmable hysteresis window: 0mV and 30mV
- Supports wake-up function
- Selectable input sources of positive input and negative input
- ACMP0 supports:
  - 3 multiplexed I/O pins at positive sources:
    - ACMP0_P0, Comparator Reference Voltage (CRV), and DAC0 output
  - 5 negative sources:
    - ACMP0_N0, ACMP0_N1, ACMP0_N2, ACMP0_N3
    - Comparator Reference Voltage (CRV)
- ACMP1 supports
  - 3 multiplexed I/O pins at positive sources:
    - ACMP1_P0, Comparator Reference Voltage (CRV), and DAC0 output
  - 5 negative sources:
    - ACMP1_N0, ACMP1_N1, ACMP1_N2, ACMP1_N3
    - Comparator Reference Voltage (CRV)
- Shares one ACMP interrupt vector for all comparators
- Interrupts generated when compare results change (Interrupt event condition is programmable)
- Supports triggers for break events and cycle-by-cycle control for PWM
- Supports window compare mode and window latch mode

6.21.3 Block Diagram

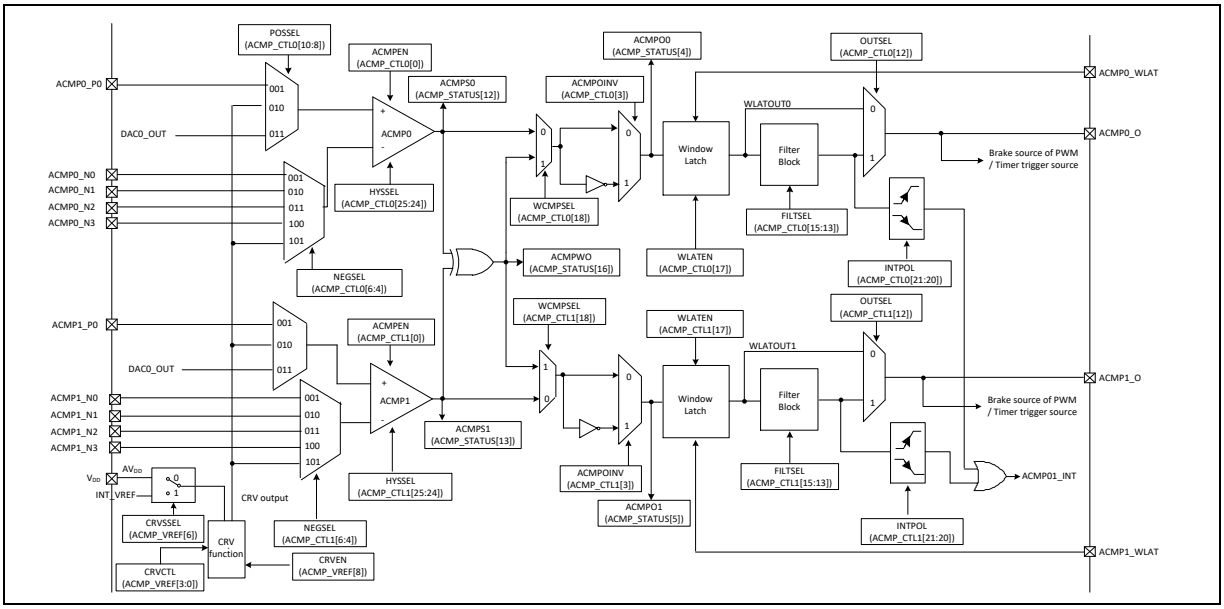


Figure 6.21-1 Analog Comparator Block Diagram

6.21.4 Basic Configuration

6.21.4.1 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]).
- Pin configuration

Group	Pin Name	GPIO	MFP
ACMP0	ACMP0_P0	PA.0	MFP1
	ACMP0_O	GPIO pins, Except PA.3, PD.0~PD.7	MFP21
	ACMP0_N0	PA.1	MFP1
	ACMP0_N1	PC.1	MFP1
	ACMP0_N2	PC.2	MFP1
	ACMP0_N3	PC.3	MFP1
	ACMP0_WLAT	PA.5, PB.7	MFP30

6.21.4.2 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]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
ACMP1	ACMP1_P0	PC.0	MFP1
	ACMP1_O	GPIO pins, Except PA.3, PD.0~PD.7	MFP22
	ACMP1_N0	PA.1	MFP1
	ACMP1_N1	PC.1	MFP1
	ACMP1_N2	PC.2	MFP1
	ACMP1_N3	PC.3	MFP1
	ACMP1_WLAT	PA.4, PB.6	MFP30

6.21.5 Functional Description

6.21.5.1 Hysteresis Function

The analog comparator provides the hysteresis function to make the comparator to have a stable output transition (referring to Figure 6.21-2). 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.

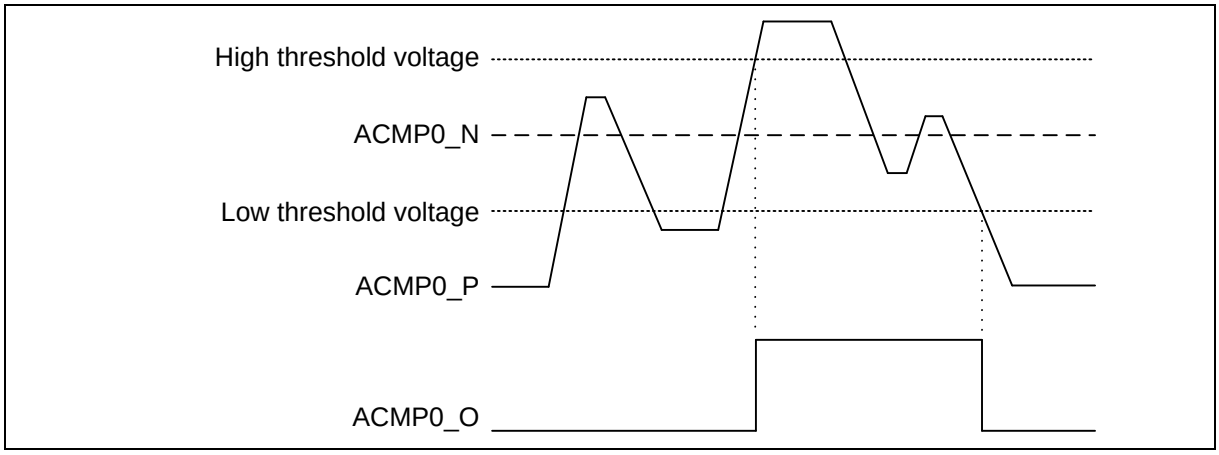


Figure 6.21-2 Comparator Hysteresis Function of ACMP0

6.21.5.2 Window Latch Mode

Figure 6.21-3 shows the comparator operation in window latch mode. Window latch mode can be enabled by setting WLATEN (ACMP_CTL0/1[17]) to 1. When window latch function enabled, ACMP0/1_WLAT pin is used to control the output WLATOUT0/1 .When ACMP0/1_WLAT pin is high, ACMPO0/1 passes through to WLATOUT0/1. When ACMP0/1_WLAT pin is low, WLATOUT0/1 will keep last state of WLATOUT0/1.

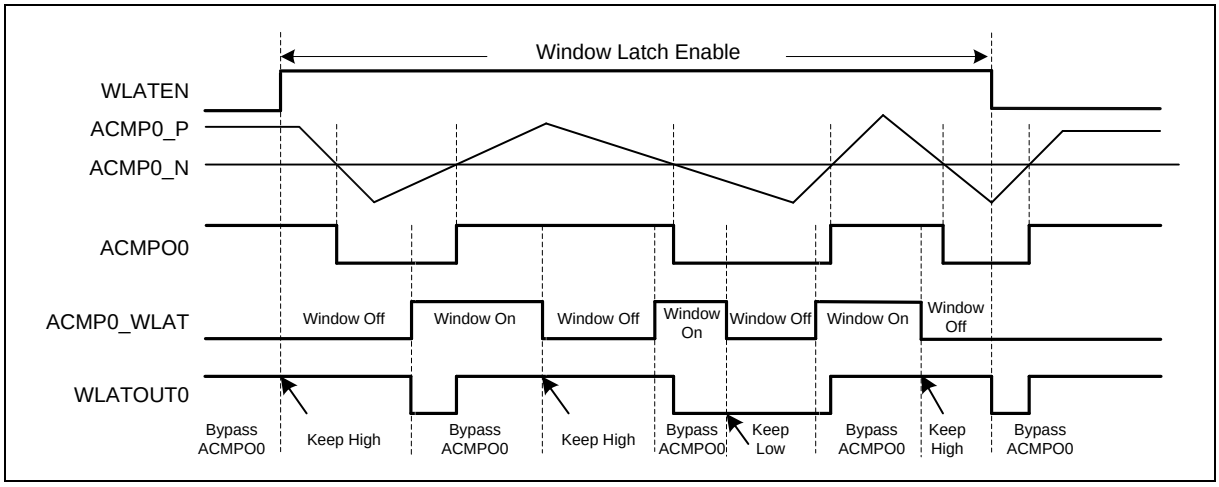


Figure 6.21-3 Window Latch Mode

6.21.5.3 Filter Function

The analog comparator provides filter function to avoid the un-stable state of comparator output.

By setting FILTSEL (ACMP_CTL0[15:13], ACMP_CTL1[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.

Figure 6.21-4 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 filted.

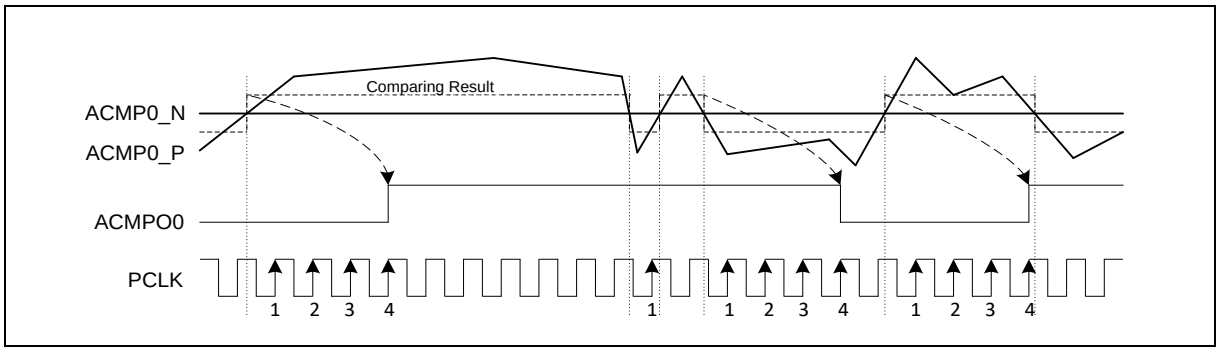


Figure 6.21-4 Filter Function Example

6.21.5.4 Interrupt 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 6.21-5, if ACMPIE of ACMP_CTL0/1 register is set to 1, the interrupt will be enabled. If the output state ACMPO0/1 is changed as the setting of INTPOL (ACMP_CTL0/1[9:8]), the comparator interrupt will be asserted and the corresponding flag, ACMPIF0 (ACMP_STATUS[0]) and ACMPIF1 (ACMP_STATUS[1]), will be set to 1. The interrupt flag can be cleared to 0 by writing 1.

If ACMP wakeup function is enabled and system wakeup from power down by ACMP with interrupt enabled (ACMPIE), the WKIF (ACMP_STATUS[8], ACMP_STATUS[9]) will be set that causes interrupt rising.

Figure 6.21-5 shows the interrups of ACMP is coming from ACMPIF or WKIF and enabled or disabled by ACMPIE.

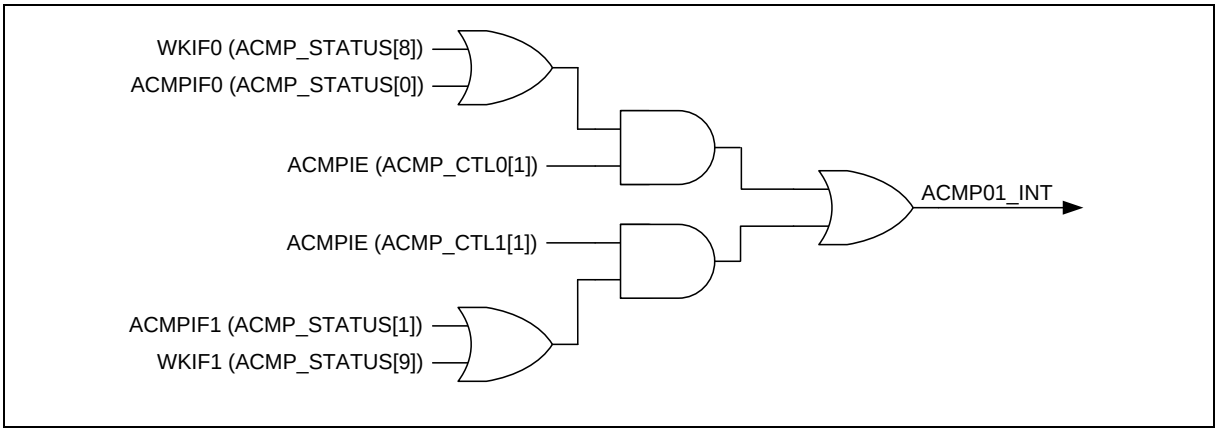


Figure 6.21-5 Comparator Controller Interrupt

6.21.5.5 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 CRVCTL (ACMP_VREF[3:0]). 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 6.21-6 shows the block diagram of Comparator Reference Voltage.

The resistor ladder will be disabled by hardware to reduce power consumption when NEGSEL (ACMP_CTL0[6:4], ACMP_CTL1[6:4]) is not selected to CRV module. The reference voltage of resistor ladder can be the voltage of AV_{DD} (voltage of V_{DD} pin) or the internal V_{REF}.

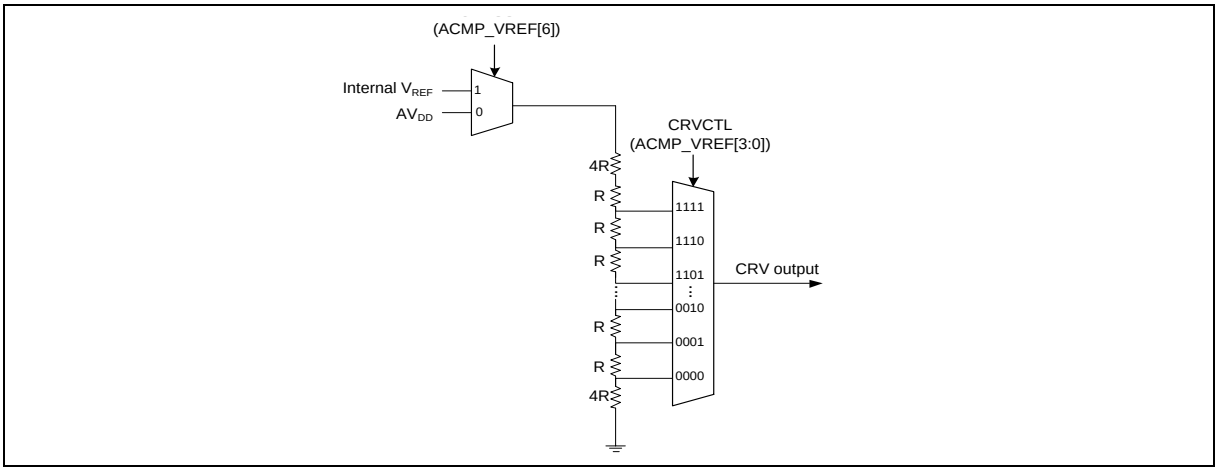


Figure 6.21-6 Comparator Reference Voltage Block Diagram

6.21.5.6 Window Compare Mode

The comparator provides 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 6.21-7 illustrates an example of window compare mode. In this example, once window compare

mode is selected, user can choose 1 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 is in the window. Otherwise, the voltage source is outside the window.

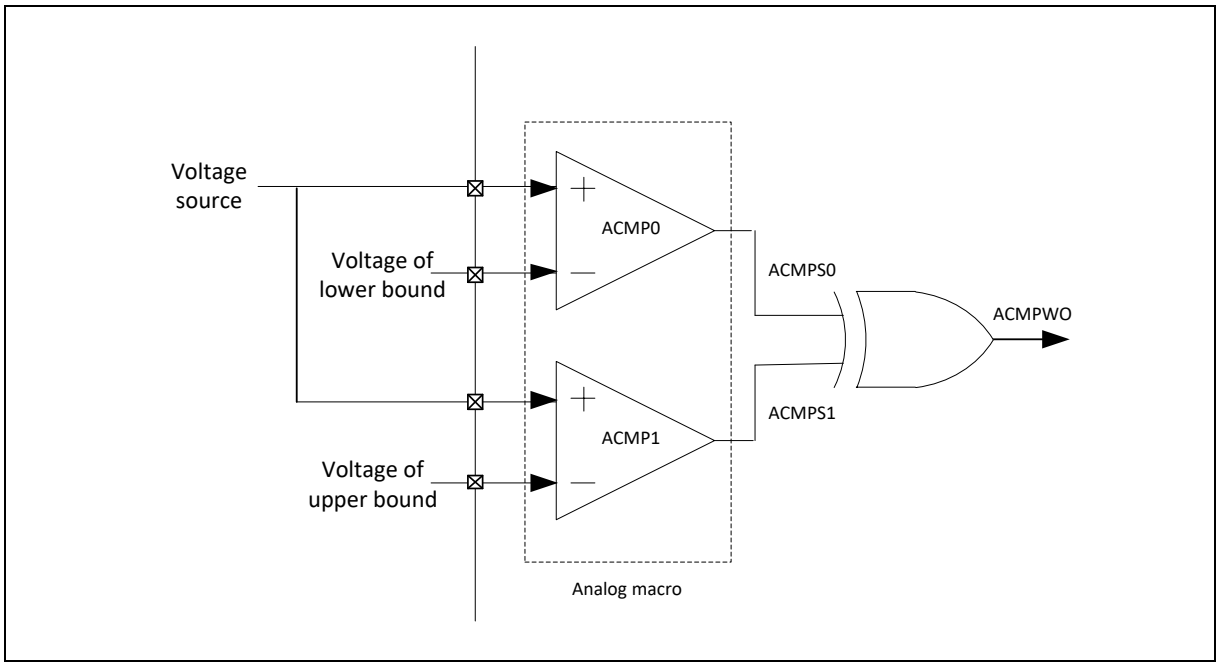


Figure 6.21-7 Example of Window Compare Mode

The comparator window output (ACMPWO) can be shown in ACMP_STATUS[16] and the truth table of window compare logic are shown in Table 6.21-1.

ACMPS0	ACMPS1	ACMPWO
0	0	0
0	1	1
1	1	0
1	0	1

Table 6.21-1 Truth Table of Window Compare Logic

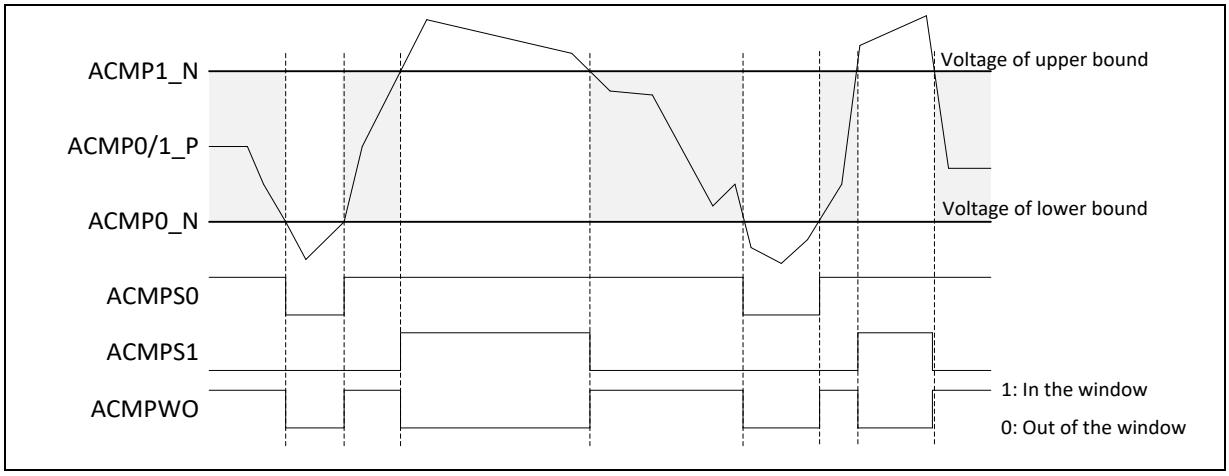


Figure 6.21-8 Example of Window Compare Mode

As shown in Figure 6.21-8, 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.

6.21.6 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**				
ACMP_CTL0	ACMP01_BA+0x00	R/W	Analog Comparator 0 Control Register	0x0000_0000
ACMP_CTL1	ACMP01_BA+0x04	R/W	Analog Comparator 1 Control Register	0x0000_0000
ACMP_STATUS	ACMP01_BA+0x08	R/W	Analog Comparator Status Register	0x0000_0000
ACMP_VREF	ACMP01_BA+0x0C	R/W	Analog Comparator Reference Voltage Control Register	0x0000_0000

6.21.7 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	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	HYSBYPASS	Reserved				HYSSEL	
23	22	21	20	19	18	17	16
Reserved		INTPOL		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]	**Reserved**	Reserved.
[30]	**HYSBYPASS**	**Hysteresis Adjust Function Selection** 0 = Enable adjust function 1 = Bypass adjust function
[29:26]	**Reserved**	Reserved.
[25:24]	**HYSSEL**	**Hysteresis Mode Selection** 00 = Hysteresis is 0mV. 11 = Hysteresis is 30mV.
[23:22]	**Reserved**	Reserved.
[21:20]	**INTPOL**	**Interrupt Condition Polarity Selection** ACMPIF0 will be set to 1 when comparator output edge condition is detected. 00 = Rising edge or falling edge. 01 = Rising edge. 10 = Falling edge. 11 = Reserved.
[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**	**Comparator 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**	**Comparator Output Select** 0 = Comparator 0 output to ACMP0_O pin is unfiltered comparator output. 1 = Comparator 0 output to ACMP0_O pin is from filter output.
[11]	**Reserved**	Reserved.
[10:8]	**POSSEL**	**Comparator Positive Input Selection** 000 = All positive input disabled 001 = Input from ACMP0_P0. 010 = Comparator Reference Voltage (CRV) 011 = DAC0 output Others = Reserved
[7]	**Reserved**	Reserved.
[6:4]	**NEGSEL**	**Comparator Negative Input Selection** 000 = All negative input disbaled 001 = ACMP0_N0 010 = ACMP0_N1 011 = ACMP0_N2 100 = ACMP0_N3 101 = Comparator Reference Voltage (CRV) Others = Reserved.
[3]	**ACMPOINV**	**Comparator Output Inverse** 0 = Comparator 0 output inverse Disabled. 1 = Comparator 0 output inverse Enabled.
[2]	**Reserved**	Reserved.
[1]	**ACMPIE**	**Comparator Interrupt Enable Bit** 0 = Comparator 0 interrupt Disabled. 1 = Comparator 0 interrupt Enabled. If WKEN (ACMP_CTL0[16]) is set to 1, the wake-up interrupt function will be enabled as well.
[0]	**ACMPEN**	**Comparator Enable Bit** 0 = Comparator 0 Disabled. 1 = Comparator 0 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	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						HYSSEL	
23	22	21	20	19	18	17	16
Reserved		INTPOL		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	
[31:26]	**Reserved**	Reserved.
[25:24]	**HYSSEL**	**Hysteresis Mode Selection** 00 = Hysteresis is 0mV. 11 = Hysteresis is 30mV.
[23:22]	**Reserved**	Reserved.
[21:20]	**INTPOL**	**Interrupt Condition Polarity Selection** ACMPIF0 will be set to 1 when comparator output edge condition is detected. 00 = Rising edge or falling edge. 01 = Rising edge. 10 = Falling edge. 11 = Reserved.
[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 Wakeup Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.
[15:13]	**FILTSEL**	**Comparator 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.

Bits	Description	
		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**	**Comparator Output Select** 0 = Comparator 1 output to ACMP1_O pin is unfiltered comparator output. 1 = Comparator 1 output to ACMP1_O pin is from filter output.
[11]	**Reserved**	Reserved.
[10:8]	**POSSEL**	**Comparator Positive Input Selection** 000 = All positive input disabled 001 = Input from ACMP1_P0. 010 = Comparator Reference Voltage (CRV) 011 = DAC0 output Others = Reserved
[7]	**Reserved**	Reserved.
[6:4]	**NEGSEL**	**Comparator Negative Input Selection** 000 = All negative input disbaled 001 = ACMP1_N0 010 = ACMP1_N1 011 = ACMP1_N2 100 = ACMP1_N3 101 = Comparator Reference Voltage (CRV) Others = Reserved.
[3]	**ACMPOINV**	**Comparator Output Inverse Control** 0 = Comparator 1 output inverse Disabled. 1 = Comparator 1 output inverse Enabled.
[2]	**Reserved**	Reserved.
[1]	**ACMPIE**	**Comparator Interrupt Enable Bit** 0 = Comparator 1 interrupt Disabled. 1 = Comparator 1 interrupt Enabled. If WKEN (ACMP_CTL1[16]) is set to 1, the wake-up interrupt function will be enabled as well.
[0]	**ACMPEN**	**Comparator Enable Bit** 0 = Comparator 1 Disabled. 1 = Comparator 1 Enabled.

Analog Comparator Status Register (ACMP_STATUS)

Register	Offset	R/W	Description	Reset Value
ACMP_STATUS	ACMP01_BA+0x08	R/W	Analog Comparator 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**	**Comparator 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**	**Comparator 1 Status** Synchronized to the PCLK to allow reading by software. Cleared when the comparator 1 is disabled, i.e. ACMPEN (ACMP_CTL1[0]) is cleared to 0.
[12]	**ACMPS0**	**Comparator 0 Status** Synchronized to the PCLK to allow reading by software. Cleared when the comparator 0 is disabled, i.e. ACMPEN (ACMP_CTL0[0]) is cleared to 0.
[11:10]	**Reserved**	Reserved.
[9]	**WKIF1**	**Comparator 1 Power-down Wake-up Interrupt Flag** 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**	**Comparator 0 Power-down Wake-up Interrupt Flag** 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**	**Comparator 1 Output** Synchronized to the PCLK to allow reading by software. Cleared when the comparator 1 is disabled, i.e. ACMPEN (ACMP_CTL1[0]) is cleared to 0.

Bits	Description	
[4]	**ACMPO0**	**Comparator 0 Output** Synchronized to the PCLK to allow reading by software. Cleared when the comparator 0 is disabled, i.e. ACMPEN (ACMP_CTL0[0]) is cleared to 0.
[3:2]	**Reserved**	Reserved.
[1]	**ACMPIF1**	**Comparator 1 Interrupt Flag** This bit is set by hardware when the edge condition defined by INTPOL (ACMP_CTL1[21:20]) is detected on comparator 1 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**	**Comparator 0 Interrupt Flag** This bit is set by hardware when the edge condition defined by INTPOL (ACMP_CTL0[21:20]) is detected on comparator 0 output. This will generate an interrupt if ACMPIE (ACMP_CTL0[1]) is set to 1. **Note:** Write 1 to clear this bit to 0.

ACMP Reference Voltage Control Register (ACMP_VREF)

Register	Offset	R/W	Description	Reset Value
ACMP_VREF	ACMP01_BA+0x0C	R/W	Analog Comparator 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						COMPEN	CRVEN
7	6	5	4	3	2	1	0
Reserved	CRVSSEL	Reserved		CRVCTL			

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**COMPEN**	**Comparator Bias Enable Bit** 0 = Comparator bias Disabled. 1 = Comparator bias Enabled.
[8]	**CRVEN**	**CRV Function Enable Bit** 0 = CRV function Disabled. 1 = CRV function Enabled.
[7]	**Reserved**	Reserved.
[6]	**CRVSSEL**	**CRV Source Voltage Selection** 0 = AV_{DD} (voltage of V_{DD} pin)is selected as CRV source voltage. 1 = Internal V_{REF} is selected as as CRV source voltage.
[5:4]	**Reserved**	Reserved.
[3:0]	**CRVCTL**	**Comparator Reference Voltage Setting** CRV = CRV source voltage * (1/6+CRVCTL/24).

6.22 Peripherals Interconnection

6.22.1 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 response.

6.22.2 Peripherals Interconnect Matrix Table

Source	Destination			
	ADC	DAC	PWM	Timer
LIRC	-	-	-	4
PWM	1	-	-	-
Timer	1	2	3	5

Table 6.22-1 Peripherals Interconnect Matrix Table

6.22.3 Functional Description

6.22.3.1 From PWM, TIMER to ADC

PWM Trigger ADC Conversion

PWM can be one of the ADC conversion trigger source.

Setting the ADC external hardware trigger input source from PWM trigger is described in section 6.19.5.8.

The detailed PWM trigger conditions are described in section 6.10.5.23.

Timer Trigger ADC Conversion

Timer0 ~ Timer3 can be one of the ADC 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 ADC external hardware trigger input source from timer trigger is described in section 6.19.5.7.

The detailed Timer trigger conditions are described in section 6.7.6.10.

6.22.3.2 From TIMER to DAC

Timer Trigger DAC Conversion

Timer0 ~ Timer3 can be one of the DAC 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 DAC to start the conversion.

Setting the DAC external hardware trigger input source from timer trigger is described in section 6.20.5.5

The detailed Timer trigger conditions are described in section 6.7.6.10.

6.22.3.3 From TIMER to PWM

Timer Generates Trigger Pulses as PWM External Clock Source

Timer0 ~ Timer3 can generate trigger pules as PWM external clock source.

When the 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 6.7.6.10.

The setting of PWM clock source is described in section 6.10.3.

6.22.3.4 From LIRC to Timer Capture Function

Measure the Time Interval of LIRC clock Speed

Set the timer capture source from LIRC clock and measure the time interval of the signal by using timer capture function. The results of time interval can be used to trim LIRC through software.

The detailed setting of time capture function is described in section 6.7.6.8.

6.22.3.5 From Timer0/2 to Timer1/3

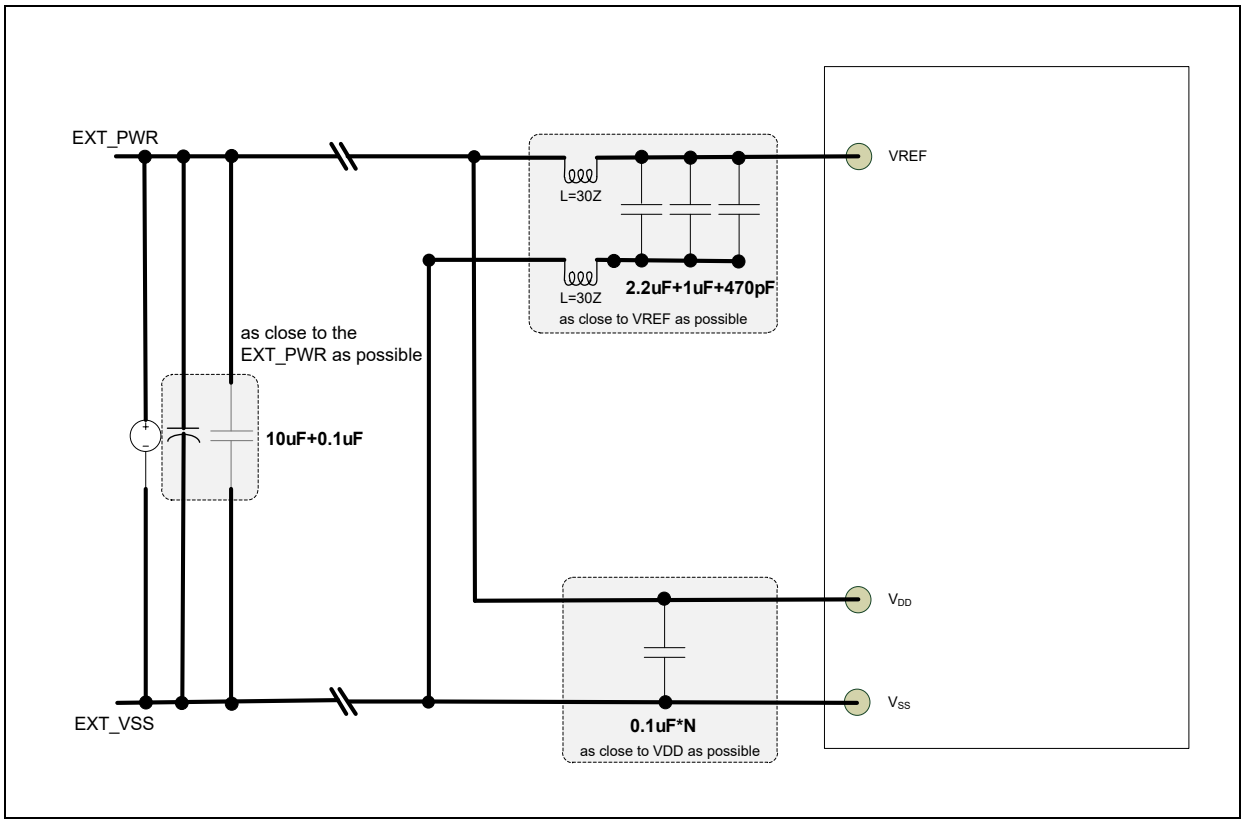
Inter-Timer Trigger Capture Mode

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.

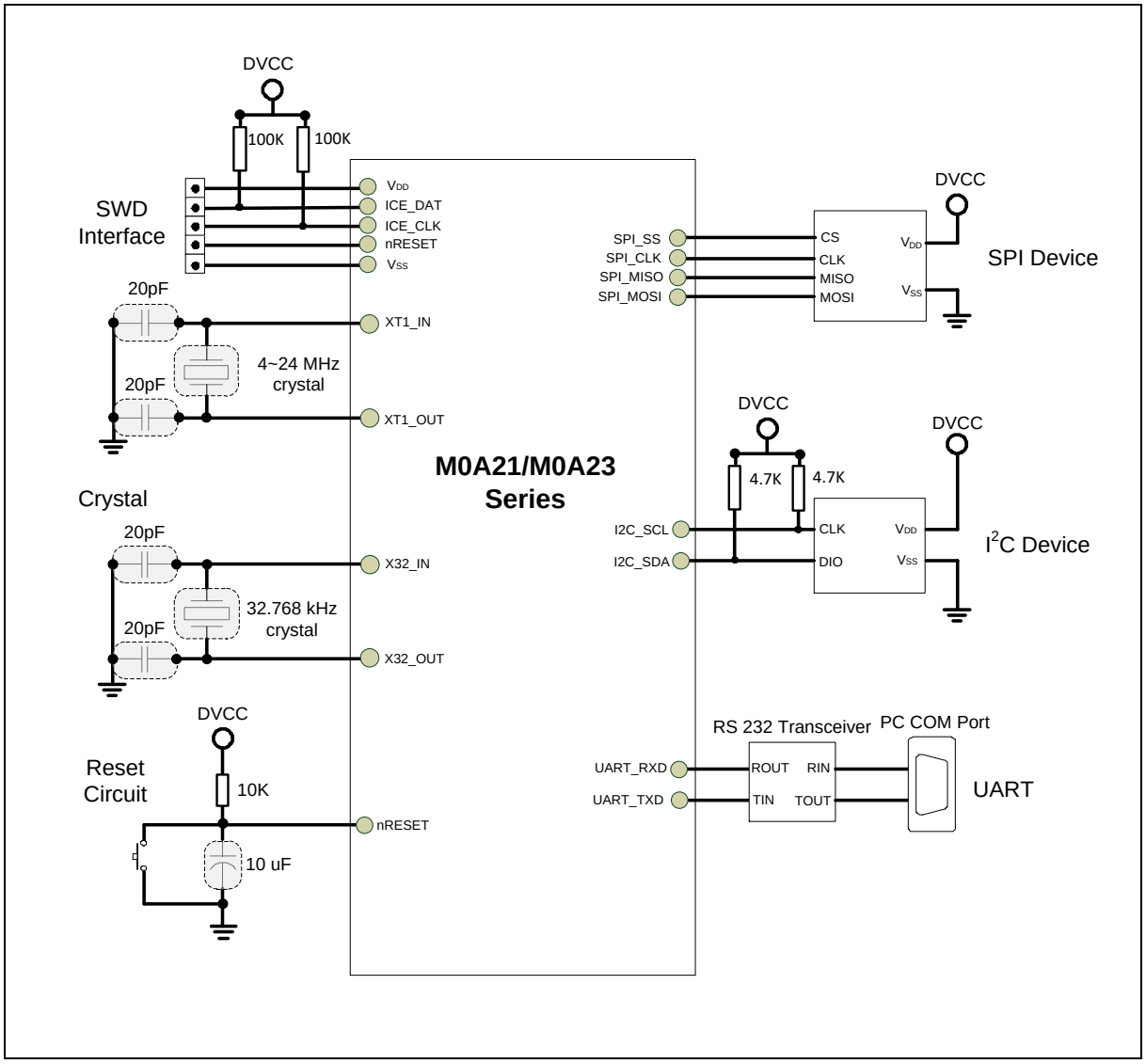
The detail of inter-timer trigger capture mode is described in section 6.7.6.11.

7 APPLICATION CIRCUIT

7.1 Power Supply Scheme



7.2 Peripheral Application Scheme



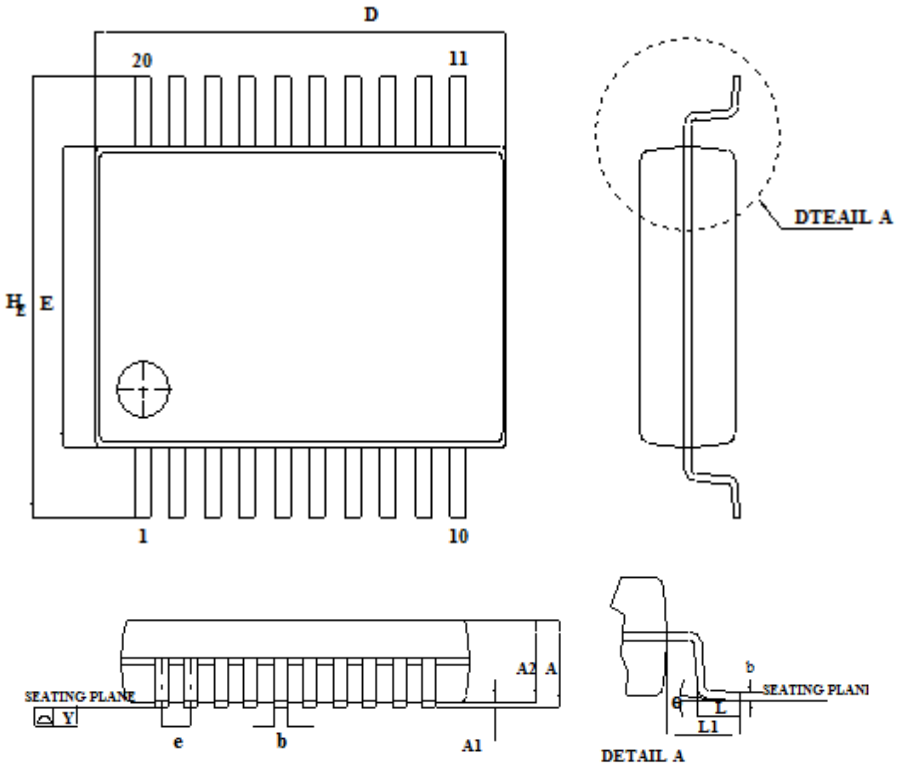
8 ELECTRICAL CHARACTERISTICS

Please refer to the relative Datasheet for detailed information about the M0A21/M0A23 electrical characteristics.

9 PACKAGE DIMENSIONS

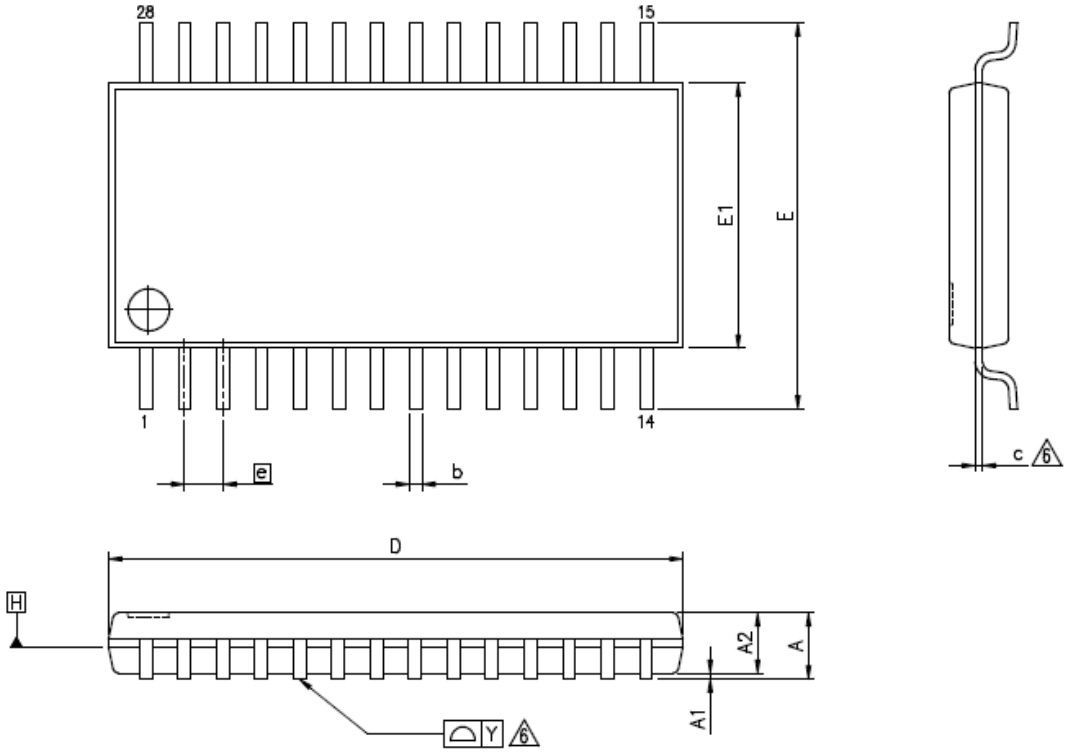
Package is Halogen-free, RoHS-compliant and TSCA-compliant.

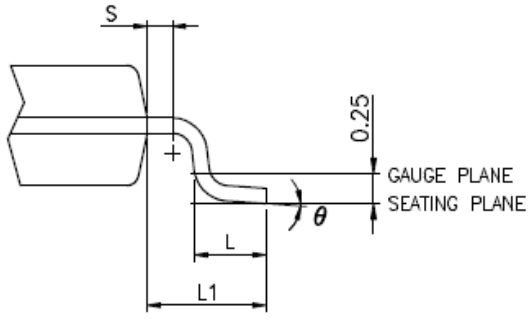
9.1 SSOP 20 (5.3x7.2x1.75 mm)



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN.	NOM	MAX.	MIN.	NOM	MAX.
A	—	—	2.00	—	—	0.079
A1	0.05	—	—	0.002	—	—
A2	1.65	1.75	1.85	0.065	0.069	0.073
b	0.22	—	0.38	0.009	—	0.015
c	0.09	—	0.25	0.004	—	0.010
D	6.90	7.20	7.50	0.272	0.283	0.295
E	5.00	5.30	5.60	0.197	0.209	0.220
H_E	7.40	7.80	8.20	0.291	0.307	0.323
e	—	0.65	—	—	0.0256	—
L	0.55	0.75	0.95	0.021	0.030	0.037
L1	—	1.25	—	—	0.050	—
Y	—	—	0.10	—	—	0.004
θ	0	—	8	0	—	8

9.2 TSSOP 28 (4.4x9.7x1.0 mm)





VARIATIONS (ALL DIMENSIONS SHOWN IN MM)

SYMBOLS	MIN.	NOM.	MAX.
A	–	–	1.20
A1	0.00	–	0.15
A2	0.80	1.00	1.05
b	0.19	–	0.30
c	0.09	–	0.20
D	9.60	9.70	9.80
E1	4.30	4.40	4.50
E	6.40 BSC		
e	0.65 BSC		
L1	1.00 REF		
L	0.45	0.60	0.75
S	0.20	–	–
θ	0°	–	8°
Y	0.10		

10 ABBREVIATIONS

10.1 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
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

QEI	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 10.1-1 List of Abbreviations

11 REVISION HISTORY

Date	Revision	Description
2020.11.23	1.00	• Initial version.
2022.02.23	1.01	• Fixed Figure 6.14-7 and Figure 6.14-8 overlapping.
2022.05.06	1.02	• Fixed the max frequency of HXT from 32 MHz to 24 MHz.
2023.03.08	1.03	• Fixed the the description of max ADC sampling rate to 800 kSPS in chapter 1, 2 and section 6.19.2, 6.19.5.1. • Changed the ADC note about using the internal channel for temperature sensor voltage and band-gap voltage in section 6.19.2 and 6.19.7. • Added "Package is Halogen-free, RoHS-compliant and TSCA-compliant.” section 3.1 and chapter 9.
2024.01.23	1.04	• Fixed PA.3 type to input only in section 4.2.1, 4.2.2 and 4.2.3. • Removed non-supported register VTOR and SHPR1 in section 6.2.15. • Added PA.3 to ineffective for pull-up selection in section 6.5.7.
2025.08.11	1.05	• Updated function pin tables in section 4.1.2 and 4.1.4.

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.