Arm® Cortex®-M

32-bit Microcontroller

NuMicro® Family
NUC029xEE Series
Technical Reference Manual

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

www.nuvoton.com

TABLE OF CONTENTS

LIST OF FIGURES 7
LIST OF TABLES 11
1 GENERAL DESCRIPTION 12
2 FEATURES 13
3 ABBREVIATIONS 16
4 PARTS INFORMATION LIST AND PIN CONFIGURATION 18
4.1 NuMicro® NUC029 Series Selection Code 18
4.2 NuMicro® NUC029 Series Selection Guide 19
4.3 Pin Configuration 20
4.3.1 NuMicro® NUC029LEE/NUC029SEE Pin Diagram 20
4.4 Pin Description 22
4.4.1 NuMicro® NUC029LEE/NUC029SEE Pin Description 22
5 BLOCK DIAGRAM 27
5.1 NuMicro® NUC029LEE/NUC029SEE Block Diagram 27
6 FUNCTIONAL DESCRIPTION 28
6.1 Arm® Cortex®-M0 Core 28
6.2 System Manager 30
6.2.1 Overview 30
6.2.2 System Reset 30
6.2.3 System Power Distribution 31
6.2.4 System Memory Map 32
6.2.5 Register Lock 33
6.2.6 Auto Trim 34
6.2.7 Register Map 36
6.2.8 Register Description 37
6.2.9 System Timer (SysTick) 74
6.2.10 Nested Vectored Interrupt Controller (NVIC) 79
6.2.11 System Control 105
6.3 Clock Controller 113
6.3.1 Overview 113
6.3.2 System Clock and SysTick Clock 116
6.3.3 Power-down Mode Clock 117
6.3.4 Frequency Divider Output 117
6.3.5 Register Map 119

6.3.6 Register Description 120
6.4 Flash Memory Controller (FMC) 141
6.4.1 Overview 141
6.4.2 Features 141
6.4.3 Block Diagram 142
6.4.4 Functional Description 143
6.4.5 Register Map 153
6.4.6 Register Description 154
6.5 External Bus Interface (EBI) 163
6.5.1 Overview 163
6.5.2 Features 163
6.5.3 Block Diagram 164
6.5.4 Functional Description 164
6.5.5 Register Map 170
6.5.6 Register Description 171
6.6 General Purpose I/O (GPIO) 175
6.6.1 Overview 175
6.6.2 Features 175
6.6.3 Basic Configuration 176
6.6.4 Functional Description 176
6.6.5 Register Map 179
6.6.6 Register Description 182
6.7 PDMA Controller (PDMA) 194
6.7.1 Overview 194
6.7.2 Features 194
6.7.3 Block Diagram 195
6.7.4 Basic Configuration 196
6.7.5 Functional Description 196
6.7.6 Register Map 199
6.7.7 Register Description 201
6.8 Timer Controller (TIMER) 232
6.8.1 Overview 232
6.8.2 Features 232
6.8.3 Block Diagram 233
6.8.4 Basic Configuration 235
6.8.5 Functional Description 235

6.8.6 Register Map 238
6.8.7 Register Description 240
6.9 PWM Generator and Capture Timer (PWM) 249
6.9.1 Overview 249
6.9.2 Features 250
6.9.3 Block Diagram 251
6.9.4 Basic Configuration 254
6.9.5 Functional Description 254
6.9.6 Register Map 264
6.9.7 Register Description 266
6.10 Watchdog Timer (WDT) 293
6.10.1 Overview 293
6.10.2 Features 293
6.10.3 Block Diagram 294
6.10.4 Basic Configuration 295
6.10.5 Functional Description 295
6.10.6 Register Map 297
6.10.7 Register Description 298
6.11 Window Watchdog Timer (WWDT) 301
6.11.1 Overview 301
6.11.2 Features 301
6.11.3 Block Diagram 301
6.11.4 Basic Configuration 302
6.11.5 Functional Description 302
6.11.6 Register Map 305
6.11.7 Register Description 306
6.12 Real Time Clock (RTC) 311
6.12.1 Overview 311
6.12.2 Features 311
6.12.3 Block Diagram 312
6.12.4 Basic Configuration 312
6.12.5 Functional Description 312
6.12.6 Register Map 316
6.12.7 Register Description 318
6.13 UART Interface Controller (UART) 334
6.13.1 Overview 334

6.13.2 Features 334
6.13.3 Block Diagram 335
6.13.4 Basic Configuration 336
6.13.5 Functional Description 337
6.13.6 Register Map 358
6.13.7 Register Description 360
6.14 I²C Serial Interface Controller (I²C) 387
6.14.1 Overview 387
6.14.2 Features 387
6.14.3 Basic Configuration 388
6.14.4 Block Diagram 388
6.14.5 Functional Description 388
6.14.6 Example for Random Read on EEPROM 400
6.14.7 Register Map 403
6.14.8 Register Description 404
6.15 Serial Peripheral Interface (SPI) 414
6.15.1 Overview 414
6.15.2 Features 414
6.15.3 Block Diagram 415
6.15.4 Basic Configuration 415
6.15.5 Functional Description 415
6.15.6 Timing Diagram 423
6.15.7 Programming Examples 426
6.15.8 Register Map 428
6.15.9 Register Description 429
6.16 USB Device Controller (USBD) 445
6.16.1 Overview 445
6.16.2 Features 445
6.16.3 Block Diagram 446
6.16.4 Basic Configuration 446
6.16.5 Functional Description 446
6.16.6 Register Map 450
6.16.7 Register Description 452
6.17 Analog-to-Digital Converter (ADC) 469
6.17.1 Overview 469
6.17.2 Features 469

6.17.3 Block Diagram 470
6.17.4 Basic Configuration 470
6.17.5 Functional Description 470
6.17.6 Register Map 478
6.17.7 Register Description 479
7 ELECTRICAL CHARACTERISTICS 490
8 PACKAGE DIMENSIONS 491
8.1 64-pin LQFP (7x7x1.4 mm footprint 2.0 mm) 491
8.2 48-pin LQFP (7x7x1.4 mm footprint 2.0 mm) 492
9 REVISION HISTORY 493

LIST OF FIGURES

Figure 4.1-1 NuMicro® NUC029 Series Selection Code 18
Figure 4.3-1 NuMicro® NUC029SEE LQFP 64-pin Diagram 20
Figure 4.3-2 NuMicro® NUC029LEE LQFP 48-pin Diagram 21
Figure 5.1-1 NuMicro® NUC029LEE/NUC029SEE Block Diagram 27
Figure 6.1-1 Functional Controller Diagram 28
Figure 6.2-1 NuMicro® NUC029LEE/NUC029SEE Power Distribution Diagram 31
Figure 6.3-1 Clock Generator Block Diagram 114
Figure 6.3-2 Clock Generator Global View Diagram 115
Figure 6.3-3 System Clock Block Diagram 116
Figure 6.3-4 SysTick Clock Control Block Diagram 116
Figure 6.3-5 Clock Source of Frequency Divider 117
Figure 6.3-6 Frequency Divider Block Diagram 118
Figure 6.4-1 Flash Memory Control Block Diagram 142
Figure 6.4-2 Flash Memory Organization 144
Figure 6.4-3 Program Executing Range for Booting from APROM and LDROM 148
Figure 6.4-4 Executable Range of Code with IAP Function Enabled 149
Figure 6.4-5 Example Flow of Boot Selection by BS Bit 150
Figure 6.4-6 ISP Flow Example 151
Figure 6.5-1 EBI Block Diagram 164
Figure 6.5-2 Connection of 16-bit EBI Data Width with 16-bit Device 165
Figure 6.5-3 Connection of 8-bit EBI Data Width with 8-bit Device 165
Figure 6.5-4 Timing Control Waveform for 16-bit Data Width 167
Figure 6.5-5 Timing Control Waveform for 8-bit Data Width 168
Figure 6.5-6 Timing Control Waveform for Insert Idle Cycle 169
Figure 6.6-1 Push-Pull Output 176
Figure 6.6-2 Open-Drain Output 177
Figure 6.6-3 Quasi-bidirectional I/O Mode 177
Figure 6.7-1 DMA Controller Block Diagram 195
Figure 6.7-2 CRC Generator Block Diagram 196
Figure 6.8-1 Timer Controller Block Diagram 233
Figure 6.8-2 Clock Source of Timer Controller 234
Figure 6.8-3 Continuous Counting Mode 236
Figure 6.9-1 PWM Generator 0 Clock Source Control 251
Figure 6.9-2 PWM Generator 0 Architecture Diagram 251
Figure 6.9-3 PWM Generator 2 Clock Source Control 252
Figure 6.9-4 PWM Generator 2 Architecture Diagram 252

Figure 6.9-5 PWM Generator 4 Clock Source Control 253
Figure 6.9-6 PWM Generator 4 Architecture Diagram 253
Figure 6.9-7 Legend of Internal Comparator Output of PWM-Timer 254
Figure 6.9-8 PWM-Timer Operation Timing 255
Figure 6.9-9 PWM Edge-aligned Interrupt Generate Timing Waveform 255
Figure 6.9-10 Center-aligned Type Output Waveform 256
Figure 6.9-11 PWM Center-aligned Interrupt Generate Timing Waveform 257
Figure 6.9-12 PWM Double Buffering Illustration 258
Figure 6.9-13 PWM Controller Output Duty Ratio 258
Figure 6.9-14 Paired-PWM Output with Dead-zone Generation Operation 259
Figure 6.9-15 PWM trigger ADC to conversion in Center-aligned type Timing Waveform 259
Figure 6.9-16 Capture Operation Timing 260
Figure 6.9-17 PWM Group A PWM-Timer Interrupt Architecture Diagram 261
Figure 6.9-18 PWM Group B PWM-Timer Interrupt Architecture Diagram 261
Figure 6.10-1 Watchdog Timer Clock Control 294
Figure 6.10-2 Watchdog Timer Block Diagram 294
Figure 6.10-3 Watchdog Timer Time-out Interval and Reset Period Timing 296
Figure 6.11-1 Window Watchdog Timer Clock Control 301
Figure 6.11-2 Window Watchdog Timer Block Diagram 302
Figure 6.11-3 Window Watchdog Timer Reset and Reload Behavior 303
Figure 6.12-1 RTC Block Diagram 312
Figure 6.13-1 UART Clock Control Diagram 335
Figure 6.13-2 UART Block Diagram 336
Figure 6.13-3 Transmit Delay Time Operation 339
Figure 6.13-4 Auto Flow Control Block Diagram 343
Figure 6.13-5 UART CTS Auto Flow Control Enabled 343
Figure 6.13-6 UART RTS Auto Flow Control Enabled 344
Figure 6.13-7 UART RTS Flow with Software Control 344
Figure 6.13-8 IrDA Control Block Diagram 345
Figure 6.13-9 IrDA TX/RX Timing Diagram 346
Figure 6.13-10 Structure of LIN Frame 346
Figure 6.13-11 Structure of LIN Byte 347
Figure 6.13-12 Break Detection in LIN Mode 349
Figure 6.13-13 LIN Frame ID and Parity Format 349
Figure 6.13-14 LIN Sync Field Measurement 352
Figure 6.13-15 UA_BAUD Update Sequence in Automatic Resynchronization Mode when LINS_DUM_EN (UA_LIN_CTL[3]) = 1 353
Figure 6.13-16 UA_BAUD Update Sequence in Automatic Resynchronization Mode when

LINS_DUM_EN (UA_LIN_CTL[3])= 0 353
Figure 6.13-17 RS-485 RTS Driving Level in Auto Direction Mode 356
Figure 6.13-18 RS-485 RTS Driving Level with Software Control 356
Figure 6.13-19 Structure of RS-485 Frame 357
Figure 6.14-1 I2C Controller Block Diagram 388
Figure 6.14-2 I2C Bus Timing 389
Figure 6.14-3 I2C Protocol 389
Figure 6.14-4 START and STOP Conditions 390
Figure 6.14-5 Bit Transfer on the I2C Bus 391
Figure 6.14-6 Acknowledge on the I2C Bus 391
Figure 6.14-7 Master Transmits Data to Slave 391
Figure 6.14-8 Master Reads Data from Slave 392
Figure 6.14-9 Control I2C Bus according to Current I2C Status 392
Figure 6.14-10 Master Transmitter Mode Control Flow 393
Figure 6.14-11 Master Receiver Mode Control Flow 394
Figure 6.14-12 Save Mode Control Flow 395
Figure 6.14-13 GC Mode 396
Figure 6.14-14 Arbitration Lost 397
Figure 6.14-15 I2C Data Shifting Direction 398
Figure 6.14-16 I2C Time-out Count Block Diagram 400
Figure 6.14-17 EEPROM Random Read 401
Figure 6.14-18 Protocol of EEPROM Random Read 402
Figure 6.15-1 SPI Block Diagram 415
Figure 6.15-2 SPI Master Mode Application Block Diagram 416
Figure 6.15-3 SPI Slave Mode Application Block Diagram 416
Figure 6.15-4 32-Bit in One Transaction 417
Figure 6.15-5 Variable Bus Clock Frequency 419
Figure 6.15-6 Byte Reorder Function 419
Figure 6.15-7 Timing Waveform for Byte Suspend 420
Figure 6.15-8 Bit Sequence of Dual Output Mode 421
Figure 6.15-9 Bit Sequence of Dual Input Mode 421
Figure 6.15-10 FIFO Mode Block Diagram 422
Figure 6.15-11 SPI Timing in Master Mode 424
Figure 6.15-12 SPI Timing in Master Mode (Alternate Phase of SPI Bus Clock) 424
Figure 6.15-13 SPI Timing in Slave Mode 425
Figure 6.15-14 SPI Timing in Slave Mode (Alternate Phase of SPI Bus Clock) 425
Figure 6.16-1 USB Device Block Diagram 446

Figure 6.16-2 Wake-up Interrupt Operation Flow 447
Figure 6.16-3 Endpoint SRAM Structure 448
Figure 6.16-4 Setup Transaction Followed by Data IN Transaction 449
Figure 6.16-5 Data Out Transfer 449
Figure 6.17-1 ADC Controller Block Diagram 470
Figure 6.17-2 ADC Clock Control 471
Figure 6.17-3 Single Mode Conversion Timing Diagram 472
Figure 6.17-4 Single-Cycle Scan on Enabled Channels Timing Diagram 473
Figure 6.17-5 Continuous Scan on Enabled Channels Timing Diagram 474
Figure 6.17-6 V_{BG} for Measuring AV_{DD} Application Block Diagram 475
Figure 6.17-7 A/D Conversion Result Monitor Logics Diagram 476
Figure 6.17-8 A/D Controller Interrupt 477
Figure 6.17-9 ADC Single-end Input Conversion Voltage and Conversion Result Mapping 480
Figure 6.17-10 ADC Differential Input Conversion Voltage and Conversion Result Mapping 481

LIST OF TABLES

Table 4.1-1 List of Abbreviations 17
Table 6.2-1 Address Space Assignments for On-Chip Controllers 33
Table 6.2-2 Exception Model 80
Table 6.2-3 System Interrupt Map 81
Table 6.2-4 Vector Table Format 81
Table 6.3-1 Chip Idle/Power-down Mode Control Table 123
Table 6.4-1 Memory Address Map 143
Table 6.4-2 ISP Command List 152
Table 6.10-1 Watchdog Timer Time-out Interval Period Selection 296
Table 6.11-1 Window Watchdog Timer Prescale Value Selection 302
Table 6.11-2 WINCMP Setting Limitation 304
Table 6.13-1 UART Interface Controller Pin 337
Table 6.13-2 UART Baud Rate Equation 337
Table 6.13-3 UART Controller Baud Rate Parameter Setting Table 338
Table 6.13-4 UART Controller Baud Rate Register (UA_BAUD) Setting Table 339
Table 6.13-5 UART Controller Interrupt Source and Flag List 341
Table 6.13-6 Controller Interrupt Source and Flag in DMA Mode List 341
Table 6.13-7 UART Line Control of Word and Stop Length Setting 342
Table 6.13-8 UART Line Control of Parity Bit Setting 342
Table 6.13-9 LIN Header Selection in Master Mode 347
Table 6.14-1 I^2C Status Code Description 399

1 GENERAL DESCRIPTION

The NuMicro® NUC029LEE/NUC029SEE of NUC029 series is embedded with the Arm® Cortex®-M0 core running up to 72 MHz and features 128 Kbytes Flash, 16 Kbytes SRAM, and 8 Kbytes loader ROM for the ISP. It is also equipped with plenty of peripheral devices, such as Timers, Watchdog Timer, Window Watchdog Timer, RTC, PDMA with CRC calculation unit, UART, SPI, I^2C, PWM Timer, GPIO, LIN, CAN, USB 2.0 FS Device, 12-bit ADC, Low Voltage Reset Controller and Brown-out Detector.

2 FEATURES

- Arm® Cortex®-M0 core
  - Runs up to 72 MHz
  - One 24-bit system timer
  - Supports low power sleep mode
  - Single-cycle 32-bit hardware multiplier
  - NVIC for the 32 interrupt inputs, each with 4-levels of priority
  - Serial Wire Debug supports with 2 watchpoints/4 breakpoints
- Built-in LDO for wide operating voltage ranges from 2.5 V to 5.5 V
- Flash Memory
  - 128 Kbytes Flash for program code
  - 8 Kbytes Flash for ISP loader
  - Supports In-System-Program (ISP) and In-Application-Program (IAP) application code update
  - 512 byte page erase for flash
  - Configurable Data Flash address and size for 128 Kbytes system
  - Supports 2-wired ICP update through SWD/ICE interface
- SRAM Memory
  - 16 Kbytes embedded SRAM
  - Supports PDMA mode
- PDMA (Peripheral DMA)
  - Supports 9 channels PDMA for automatic data transfer between SRAM and peripherals
  - Supports CRC calculation with four common polynomials, CRC-CCITT, CRC-8, CRC-16 and CRC-32
- Clock Control
  - Flexible selection for different applications
  - Built-in 22.1184 MHz high speed oscillator for system operation
    - Trimmed to ±1 % at +25 ℃ and V_{DD} = 5 V
    - Trimmed to ±3 % at -40 ℃ ~ +105 ℃ and V_{DD} = 2.5 V ~ 5.5 V
  - Built-in 48 MHz internal high speed RC oscillator (HIRC) for USB device operation (Frequency variation < 2% at -40oC ~ +105oC)
    - Dynamically calibrating the HIRC OSC to 48 MHz ±0.25% from -40℃ to 105℃ by external 32.768K crystal oscillator (LXT) or internal USB synchronous mode
  - Built-in 10 kHz low speed oscillator for Watchdog Timer and Wake-up operation
  - Supports one PLL, up to 72 MHz, for high performance system operation
  - External 4~24 MHz high speed crystal input for USB and precise timing operation
  - External 32.768 kHz low speed crystal input for RTC function and low power system operation
- GPIO
  - Four I/O modes:
    - Quasi-bidirectional
    - Push-pull output
    - Open-drain output
    - Input only with high impendence
  - TTL/Schmitt trigger input selectable
  - I/O pin configured as interrupt source with edge/level setting
- Timer
  - Supports 4 sets of 32-bit timers with 24-bit up-timer and one 8-bit prescale counter
  - Independent clock source for each timer
  - Provides one-shot, periodic, toggle and continuous counting operation modes
  - Supports event counting function
  - Supports input capture function
- Watchdog Timer
  - Multiple clock sources
  - 8 selectable time-out period from 1.6 ms ~ 26.0 sec (depending on clock source)

- Wake-up from Power-down or Idle mode
- Interrupt or reset selectable on watchdog time-out
- Supports 4 selectable Watchdog Timer reset delay period(1026, 130, 18 or 3 WDT_CLK)

- Window Watchdog Timer
  - 6-bit down counter with 11-bit prescale for wide range window selected
- RTC
  - Supports software compensation by setting frequency compensate register (FCR)
  - Supports RTC counter (second, minute, hour) and calendar counter (day, month, year)
  - Supports Alarm registers (second, minute, hour, day, month, year)
  - Selectable 12-hour or 24-hour mode
  - Automatic leap year recognition
  - Supports periodic time tick interrupt with 8 period options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second
  - Supports battery power pin (V_{BAT})
  - Supports wake-up function
- PWM/Capture
  - Up to three built-in 16-bit PWM generators providing six PWM outputs or three complementary paired PWM outputs
  - Each PWM generator equipped with one clock source selector, one clock divider, one 8-bit prescaler and one Dead-Zone generator for complementary paired PWM
  - Supports One-shot or Auto-reload mode
  - Up to six 16-bit digital capture timers (shared with PWM timers) providing six rising/falling capture inputs
  - Supports Capture interrupt
- UART
  - Up to three UART controllers
  - UART ports with flow control (TXD, RXD, nCTS and nRTS)
  - UART0 with 64-byte FIFO is for high speed
  - UART1/2(optional) with 16-byte FIFO for standard device
  - Supports IrDA (SIR) and LIN function
  - Supports RS-485 9-bit mode and direction control
  - Programmable baud-rate generator up to 1/16 system clock
  - Supports CTS wake-up function (UART0 and UART1 support)
  - Supports PDMA mode
- SPI
  - Up to two sets of SPI controllers
  - The maximum SPI clock rate of Master can up to 36 MHz (chip working at 5V)
  - The maximum SPI clock rate of Slave can up to 18 MHz (chip working at 5V)
  - Supports SPI Master/Slave mode
  - Full duplex synchronous serial data transfer
  - Variable length of transfer data from 8 to 32 bits
  - MSB or LSB first data transfer
  - Rx and Tx on both rising or falling edge of serial clock independently
  - Two slave/device select lines in Master mode, and one slave/device select line in Slave mode
  - Supports Byte Suspend mode in 32-bit transmission
  - Supports PDMA mode
  - Supports three wire, no slave select signal, bi-direction interface
- I^2C
  - Up to two sets of I^2C devices
  - Master/Slave mode
  - Bidirectional data transfer between masters and slaves
  - Multi-master bus (no central master)
  - Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
  - Serial clock synchronization allowing devices with different bit rates to communicate via one

serial bus
  - Serial clock synchronization used as a handshake mechanism to suspend and resume serial transfer
  - Programmable clocks allowing for versatile rate control
  - Supports multiple address recognition (four slave address with mask option)
  - Supports wake-up function
- USB 2.0 Full-Speed Device
  - One set of USB 2.0 FS Device 12 Mbps
  - On-chip USB Transceiver
  - Provides 1 interrupt source with 4 interrupt events
  - Supports Control, Bulk In/Out, Interrupt and Isochronous transfers
  - Auto suspend function when no bus signaling for 3 ms
  - Provides 8 programmable endpoints
  - Includes 512 Bytes internal SRAM as USB buffer
  - Provides remote wake-up capability
  - Supports Crystal-less function
- ADC
  - 12-bit SAR ADC with 1 MSPS(chip working at 5V)
  - Up to 12-ch single-end input or 5-ch differential input
  - Single scan/single cycle scan/continuous scan
  - Each channel with individual result register
  - Scan on enabled channels
  - Threshold voltage detection
  - Conversion started by software programming, external input or PWM Center-aligned trigger
  - Supports PDMA mode
- EBI (External bus interface)
  - Accessible space: 64 Kbytes in 8-bit mode or 128 Kbytes in 16-bit mode
  - Supports 8-/16-bit data width
  - Supports byte write in 16-bit data width mode
- 96-bit unique ID (UID)
- 128-bit unique customer ID(UCID)
- One built-in temperature sensor with 1℃ resolution
- Brown-out Detector
  - With 4 levels: 4.4 V/3.7 V/2.7 V/2.2 V
  - Supports Brown-out Interrupt and Reset option
- Low Voltage Reset
  - Threshold voltage level: 2.0 V
- Operating Temperature: -40℃ ~ 105℃
- Packages:
  - All Green package (RoHS)
  - LQFP 64-pin / 48-pin

3 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
EBI	External Bus Interface
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	22.1184 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
SDIO	Secure Digital Input/Output
SPI	Serial Peripheral Interface

SPS	Samples per Second
TDES	Triple Data Encryption Standard
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
UCID	Unique Customer ID
USB	Universal Serial Bus
WDT	Watchdog Timer
WWDT	Window Watchdog Timer

Table 4.1-1 List of Abbreviations

4 PARTS INFORMATION LIST AND PIN CONFIGURATION

4.1 NuMicro® NUC029 Series Selection Code

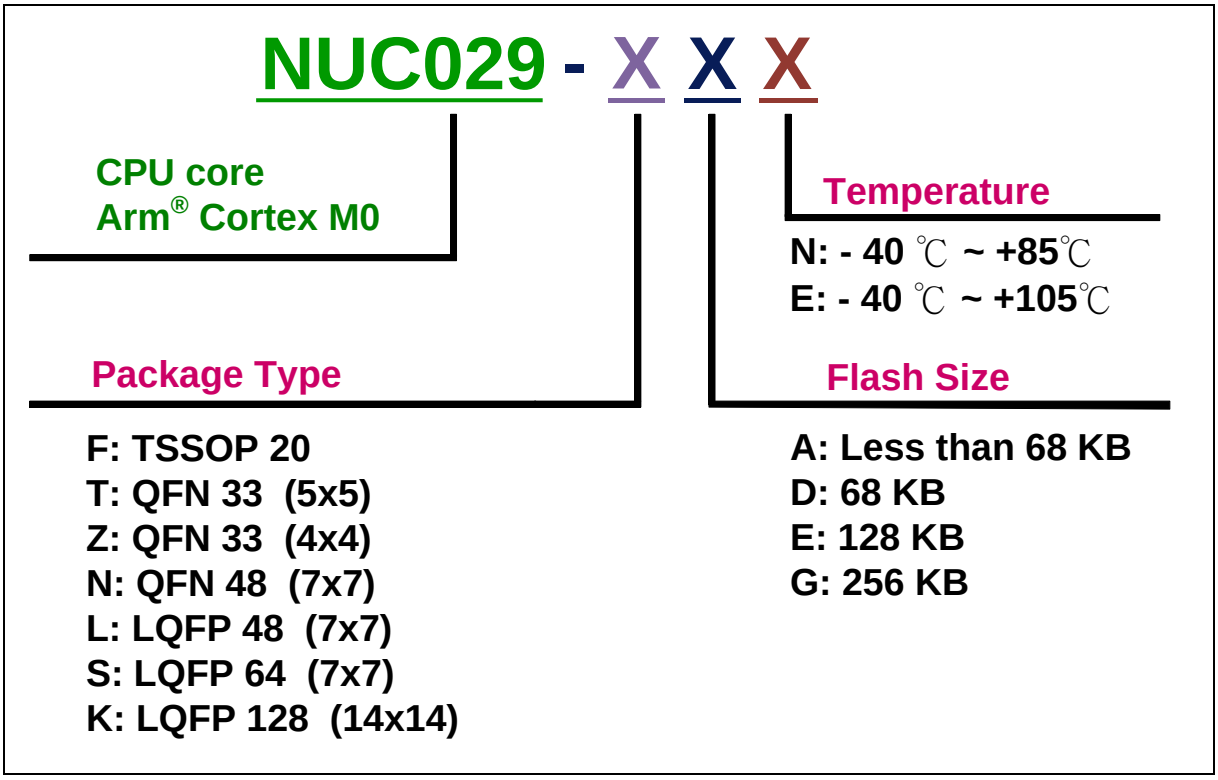


Figure 4.1-1 NuMicro® NUC029 Series Selection Code

4.2 NuMicro® NUC029 Series Selection Guide

Part Number	APROM (KB)	RAM (KB)	Data Flash (KB)	SPROM(KB)	ISP ROM (KB)	I/O	Timer (32-Bit)	Connectivity						I²S	PWM (16-Bit)	ADC	ACMP	PDMA	Smart Card	RTC	EBI	PLL	ISP/ICP/IAP	Package	Operating Temperature Range (℃)
								UART	SPI	I²C	USCI[4]	USB	LIN												
NUC029FAE	16	2	Conf	-	2	17	2	1	1	1	-	-	-	-	3	4[1]	2[3]	-	-	-	-	-	√	TSSOP20	-40 to +105
NUC029TAN	32	4	4	-	4	24	4	2	1	2	-	-	-	-	5	5	3[2]	-	-	-	-	√	√	QFN33(4*4)	-40 to +85
NUC029ZAN	64	4	4	-	4	24	4	2	1	2	-	-	-	-	5	5	3[2]	-	-	-	-	√	√	QFN33(5*5)	-40 to +85
NUC029LAN	64	4	4	-	4	40	4	2	2	2	-	-	-	-	8	8	4	-	-	-	√	√	√	LQFP48	-40 to +85
NUC029LDE	68	8	Conf	-	4	42	4	4	1	2	-	-	3	-	12	8	-	-	-	-	-	√	√	LQFP48	-40 to +105
NUC029SDE	68	8	Conf	-	4	56	4	4	1	2	-	-	3	-	12	8	-	-	-	-	-	√	√	LQFP64	-40 to +105
NUC029LEE	128	16	Conf	-	8	31	4	2	1	2	-	1	2	-	4	10	-	9	-	√	-	√	√	LQFP48	-40 to +105
NUC029SEE	128	16	Conf	-	8	45	4	3	2	2	-	1	3	-	6	12	-	9	-	√	√	√	√	LQFP64	-40 to +105
NUC029LGE	256	20	Conf	2	4	35	4	3	2	2	3	1	-	2	10	9	2	5	-	√	√	√	√	LQFP48	-40 to +105
NUC029SGE	256	20	Conf	2	4	49	4	3	2	2	3	1	-	2	12	15	2	5	-	√	√	√	√	LQFP64	-40 to +105
NUC029KGE	256	20	Conf	2	4	86	4	3	2	2	3	1	-	2	12	20	2	5	2	√	√	√	√	LQFP128	-40 to +105

[1] NUC029FAE is 10-bit ADC. All the others are 12-bit ADC.

[2] For NUC029TAN/NUC029ZAN, ACMP3 only has positive and negative input.

[3] For NUC029FAE , ACMP0 only has positive and negative input. And ACMP1 only has positive input.

[4] USCI can be configured as UART, SPI or I²C

4.3 Pin Configuration

4.3.1 NuMicro® NUC029LEE/NUC029SEE Pin Diagram

*4.3.1.1 NuMicro® NUC029SEE LQFP 64 pin (7 mm * 7mm)*

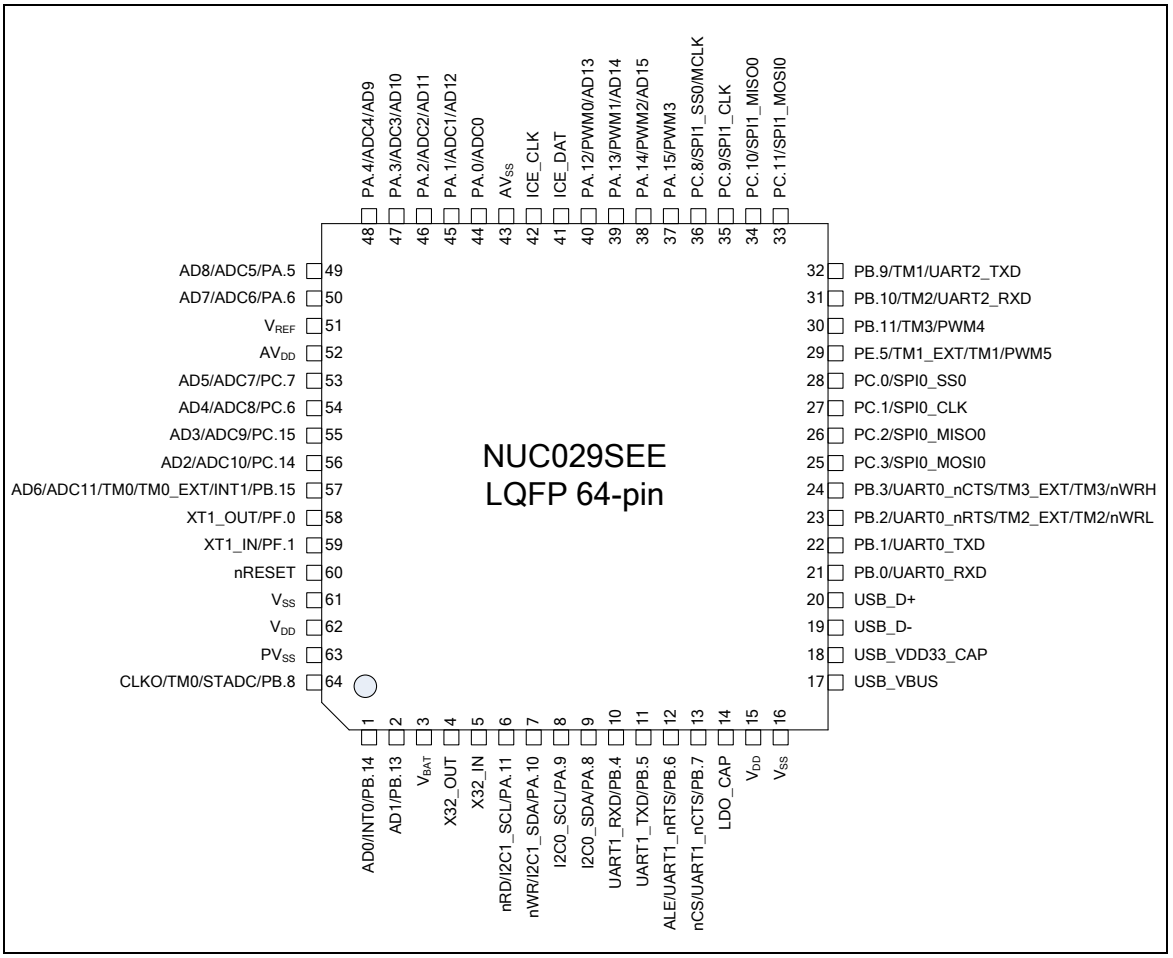


Figure 4.3-1 NuMicro® NUC029SEE LQFP 64-pin Diagram

4.3.1.2 NuMicro® NUC029LEE LQFP 48 pin (7 mm * 7mm)

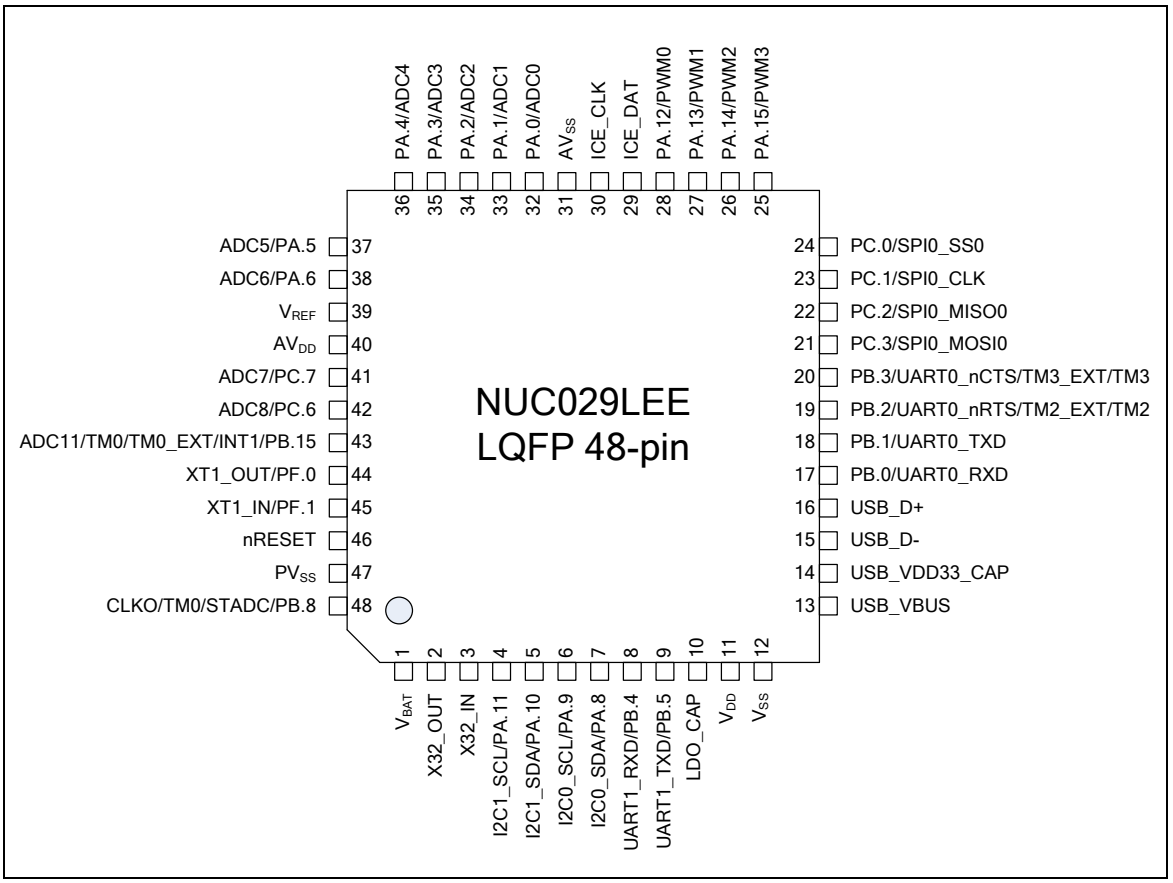


Figure 4.3-2 NuMicro® NUC029LEE LQFP 48-pin Diagram

4.4 Pin Description

4.4.1 NuMicro® NUC029LEE/NUC029SEE Pin Description

Pin No.		Pin Name	Pin Type	Description
LQFP 64-pin	LQFP 48-pin			
1		PB.14	I/O	General purpose digital I/O pin.
		INT0	I	External interrupt0 input pin.
		AD0	I/O	EBI Address/Data bus bit0
2		PB.13	I/O	General purpose digital I/O pin.
		AD1	I/O	EBI Address/Data bus bit1
3	1	V_{BAT}	P	Power supply by batteries for RTC.
4	2	X32_OUT	O	External 32.768 kHz (low speed) crystal output pin.
5	3	X32_IN	I	External 32.768 kHz (low speed) crystal input pin.
6	4	PA.11	I/O	General purpose digital I/O pin.
		I2C1_SCL	I/O	I^2C1 clock pin.
		nRD	O	EBI read enable output pin
7	5	PA.10	I/O	General purpose digital I/O pin.
		I2C1_SDA	I/O	I^2C1 data input/output pin.
		nWR	O	EBI write enable output pin
8	6	PA.9	I/O	General purpose digital I/O pin.
		I2C0_SCL	I/O	I^2C0 clock pin.
9	7	PA.8	I/O	General purpose digital I/O pin.
		I2C0_SDA	I/O	I^2C0 data input/output pin.
10	8	PB.4	I/O	General purpose digital I/O pin.
		UART1_RXD	I	Data receiver input pin for UART1.
11	9	PB.5	I/O	General purpose digital I/O pin.
		UART1_TXD	O	Data transmitter output pin for UART1.
12		PB.6	I/O	General purpose digital I/O pin.
		UART1_nRTS	O	Request to Send output pin for UART1.
		ALE	O	EBI address latch enable output pin
13		PB.7	I/O	General purpose digital I/O pin.
		UART1_nCTS	I	Clear to Send input pin for UART1.
		nCS	O	EBI chip select enable output pin
14	10	LDO_CAP	P	LDO output pin.

Pin No.		Pin Name	Pin Type	Description
LQFP 64-pin	LQFP 48-pin			
15	11	V_{DD}	**P**	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
16	12	V_{SS}	**P**	Ground pin for digital circuit.
17	13	USB_VBUS	**USB**	Power supply from USB host or HUB.
18	14	USB_VDD33_CAP	**USB**	Internal power regulator output 3.3V decoupling pin.
19	15	USB_D-	**USB**	USB differential signal D-.
20	16	USB_D+	**USB**	USB differential signal D+.
21	17	PB.0	**I/O**	General purpose digital I/O pin.
		UART0_RXD	**I**	Data receiver input pin for UART0.
22	18	PB.1	**I/O**	General purpose digital I/O pin.
		UART0_TXD	**O**	Data transmitter output pin for UART0.
23	19	PB.2	**I/O**	General purpose digital I/O pin.
		UART0_nRTS	**O**	Request to Send output pin for UART0.
		TM2_EXT	**I**	Timer2 external capture input pin.
		TM2	**O**	Timer2 toggle output pin.
		nWRL	**O**	EBI low byte write enable output pin
24	20	PB.3	**I/O**	General purpose digital I/O pin.
		UART0_nCTS	**I**	Clear to Send input pin for UART0.
		TM3_EXT	**I**	Timer3 external capture input pin.
		TM3	**O**	Timer3 toggle output pin.
		nWRH	**O**	EBI high byte write enable output pin
25	21	PC.3	**I/O**	General purpose digital I/O pin.
		SPI0_MOSI0	**I/O**	1st SPI0 MOSI (Master Out, Slave In) pin.
26	22	PC.2	**I/O**	General purpose digital I/O pin.
		SPI0_MISO0	**I/O**	1st SPI0 MISO (Master In, Slave Out) pin.
27	23	PC.1	**I/O**	General purpose digital I/O pin.
		SPI0_CLK	**I/O**	SPI0 serial clock pin.
28	24	PC.0	**I/O**	General purpose digital I/O pin.
		SPI0_SS0	**I/O**	1st SPI0 slave select pin.
29		PE.5	**I/O**	General purpose digital I/O pin.
		PWM5	**I/O**	PWM5 output/Capture input.
		TM1_EXT	**I**	Timer1 external capture input pin.
		TM1	**O**	Timer1 toggle output pin.

Pin No.		Pin Name	Pin Type	Description
LQFP 64-pin	LQFP 48-pin			
30		PB.11	I/O	General purpose digital I/O pin.
		TM3	I/O	Timer3 event counter input / toggle output.
		PWM4	I/O	PWM4 output/Capture input.
31		PB.10	I/O	General purpose digital I/O pin.
		TM2	I/O	Timer2 event counter input / toggle output.
		UART2_RXD	I	Data receiver input pin for UART2.
32		PB.9	I/O	General purpose digital I/O pin.
		TM1	I/O	Timer1 event counter input / toggle output.
		UART2_TXD	O	Data transmitter output pin for UART2.
33		PC.11	I/O	General purpose digital I/O pin.
		SPI1_MOSI0	I/O	1st SPI1 MOSI (Master Out, Slave In) pin.
34		PC.10	I/O	General purpose digital I/O pin.
		SPI1_MISO0	I/O	1st SPI1 MISO (Master In, Slave Out) pin.
35		PC.9	I/O	General purpose digital I/O pin.
		SPI1_CLK	I/O	SPI1 serial clock pin.
36		PC.8	I/O	General purpose digital I/O pin.
		SPI1_SS0	I/O	1st SPI1 slave select pin.
		MCLK	O	EBI clock output
37	25	PA.15	I/O	General purpose digital I/O pin.
		PWM3	I/O	PWM3 output/Capture input.
38	26	PA.14	I/O	General purpose digital I/O pin.
		PWM2	I/O	PWM2 output/Capture input.
		AD15	I/O	EBI Address/Data bus bit15
39	27	PA.13	I/O	General purpose digital I/O pin.
		PWM1	I/O	PWM1 output/Capture input.
		AD14	I/O	EBI Address/Data bus bit14
40	28	PA.12	I/O	General purpose digital I/O pin.
		PWM0	I/O	PWM0 output/Capture input.
		AD13	I/O	EBI Address/Data bus bit13
41	29	ICE_DAT	I/O	Serial wire debugger data pin. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_DAT pin.
42	30	ICE_CLK	I	Serial wire debugger clock pin. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_CLK pin.

Pin No.		Pin Name	Pin Type	Description
LQFP 64-pin	LQFP 48-pin			
43	31	AV_{SS}	**AP**	Ground pin for analog circuit.
44	32	PA.0	**I/O**	General purpose digital I/O pin.
		ADC0	**AI**	ADC0 analog input.
45	33	PA.1	**I/O**	General purpose digital I/O pin.
		ADC1	**AI**	ADC1 analog input.
		AD12	**I/O**	EBI Address/Data bus bit12
46	34	PA.2	**I/O**	General purpose digital I/O pin.
		ADC2	**AI**	ADC2 analog input.
		AD11	**I/O**	EBI Address/Data bus bit11
47	35	PA.3	**I/O**	General purpose digital I/O pin.
		ADC3	**AI**	ADC3 analog input.
		AD10	**I/O**	EBI Address/Data bus bit10
48	36	PA.4	**I/O**	General purpose digital I/O pin.
		ADC4	**AI**	ADC4 analog input.
		AD9	**I/O**	EBI Address/Data bus bit9
49	37	PA.5	**I/O**	General purpose digital I/O pin.
		ADC5	**AI**	ADC5 analog input.
		AD8	**I/O**	EBI Address/Data bus bit8
50	38	PA.6	**I/O**	General purpose digital I/O pin.
		ADC6	**AI**	ADC6 analog input.
		AD7	**I/O**	EBI Address/Data bus bit7
51	39	V_{REF}	**AP**	Voltage reference input for ADC.
52	40	AV_{DD}	**AP**	Power supply for internal analog circuit.
53	41	PC.7	**I/O**	General purpose digital I/O pin.
		ADC7	**AI**	ADC7 analog input.
		AD5	**I/O**	EBI Address/Data bus bit5
54	42	PC.6	**I/O**	General purpose digital I/O pin.
		ADC8	**AI**	ADC8 analog input.
		AD4	**I/O**	EBI Address/Data bus bit4
55		PC.15	**I/O**	General purpose digital I/O pin.
		ADC9	**AI**	ADC9 analog input.
		AD3	**I/O**	EBI Address/Data bus bit3
56		PC.14	**I/O**	General purpose digital I/O pin.

Pin No.		Pin Name	Pin Type	Description
LQFP 64-pin	LQFP 48-pin			
		ADC10	AI	ADC10 analog input.
		AD2	I/O	EBI Address/Data bus bit2
57	43	PB.15	I/O	General purpose digital I/O pin.
		INT1	I	External interrupt1 input pin.
		TM0_EXT	I	Timer 0 external capture input pin.
		TM0	I/O	Timer0 event counter input / toggle output.
		ADC11	AI	ADC11 analog input.
		AD6	I/O	EBI Address/Data bus bit6
58	44	PF.0	I/O	General purpose digital I/O pin.
		XT1_OUT	O	External 4~24 MHz (high speed) crystal output pin.
59	45	PF.1	I/O	General purpose digital I/O pin.
		XT1_IN	I	External 4~24 MHz (high speed) crystal input pin.
60	46	nRESET	I	External reset input: active LOW, with an internal pull-up. Set this pin low reset chip to initial state. **Note:** It is recommended to use 10 kΩ pull-up resistor and 10 uF capacitor on nRESET pin.
61		V_{SS}	P	Ground pin for digital circuit.
62		V_{DD}	P	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
63	47	PV_{SS}	P	PLL ground.
64	48	PB.8	I/O	General purpose digital I/O pin.
		STADC	I	ADC external trigger input.
		TM0	I/O	Timer0 event counter input / toggle output.
		CLKO	O	Frequency divider clock output pin.

Note: Pin Type I = Digital Input, O = Digital Output; AI = Analog Input; P = Power Pin; AP = Analog Power

5 BLOCK DIAGRAM

5.1 NuMicro® NUC029LEE/NUC029SEE Block Diagram

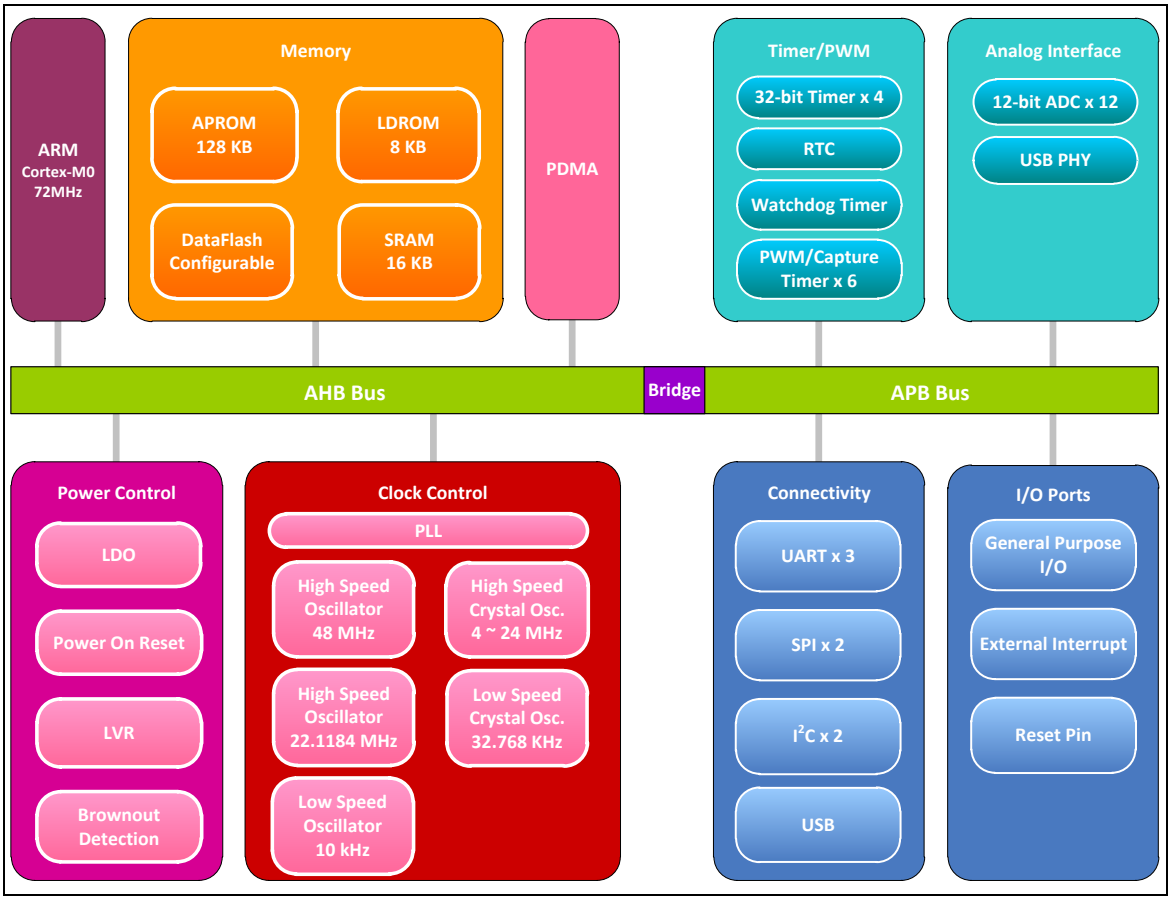


Figure 5.1-1 NuMicro® NUC029LEE/NUC029SEE Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 Arm® Cortex®-M0 Core

The Cortex®-M0 processor 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 processor 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 controller of processor.

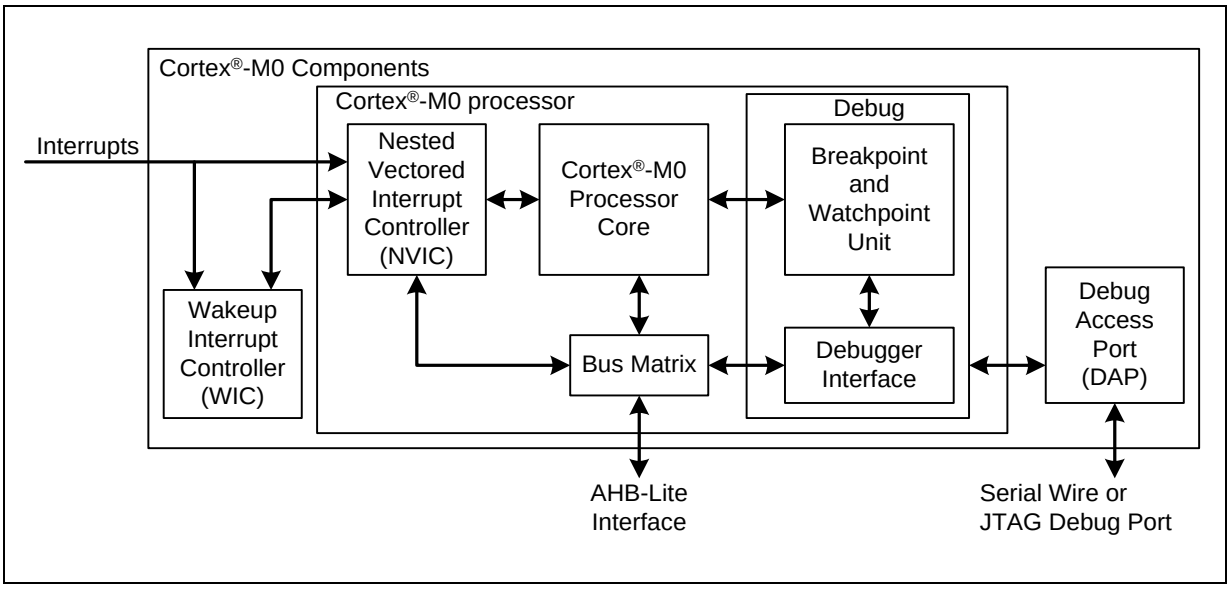


Figure 6.1-1 Functional Controller Diagram

The implemented device provides the following components and features:

- A low gate count processor:
  - ARMv6-M Thumb® instruction set
  - Thumb-2 technology
  - ARMv6-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 Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or the 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 Resets
- System Memory Map
- System management registers for Part Number ID, chip reset and on-chip controllers reset , multi-functional pin control
- System Timer (SysTick)
- Nested Vectored Interrupt Controller (NVIC)
- System Control registers

6.2.2 System Reset

The system reset can be issued by one of the following listed events. For these reset event flags can be read by RSTSRC register.

- Power-on Reset
- Low level on the nRESET pin
- Watchdog Time-out Reset
- Low Voltage Reset
- Brown-out Detector Reset
- CPU Reset
- System Reset

System Reset and Power-on Reset all reset the whole chip including all peripherals. The difference between System Reset and Power-on Reset is external crystal circuit and BS(ISPCON[1]) bit. System Reset does not reset external crystal circuit and BS(ISPCON[1]) bit, but Power-on Reset does.

6.2.3 System Power Distribution

In this chip, the power distribution is divided into three segments.

- Analog power from AV_{DD} and AV_{SS} provides the power for analog components operation.
- Digital power from V_{DD} and V_{SS} supplies the power to the internal regulator which provides a fixed 1.8 V power for digital operation and I/O pins.
- USB transceiver power from V_{BUS} offers the power for operating the USB transceiver.
- Battery power from V_{BAT} supplies the RTC and external 32.768 kHz crystal.

The outputs of internal voltage regulators, LDO and V_{DD33}, require an external capacitor which should be located close to the corresponding pin. Analog power (AV_{DD}) should be the same voltage level with the digital power (V_{DD}). Figure 6.2-1 shows the NuMicro® NUC029LEE/NUC029SEE power distribution.

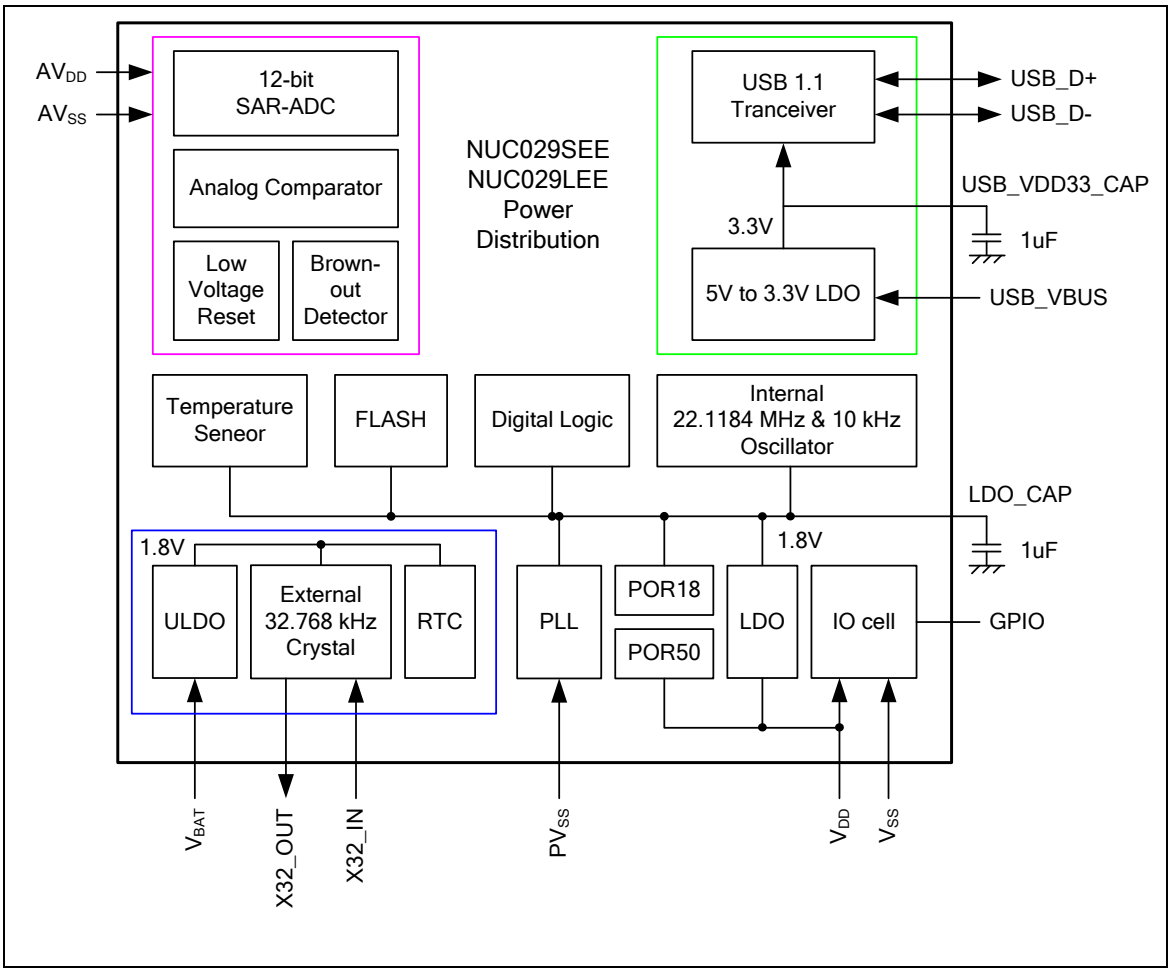


Figure 6.2-1 NuMicro® NUC029LEE/NUC029SEE Power Distribution Diagram

6.2.4 System Memory Map

The NuMicro® NUC029LEE/NUC029SEE provides 4G-byte addressing space. The memory locations assigned to each on-chip controllers are shown in the following table. The detailed register definition, memory space, and programming detailed will be described in the following sections for each on-chip peripheral. The NuMicro® NUC029LEE/NUC029SEE only supports little-endian data format.

Address Space	Token	Controllers
Flash and SRAM Memory Space		
0x0000_0000 – 0x0001_FFFF	FLASH_BA	FLASH Memory Space (128 KB)
0x2000_0000 – 0x2000_3FFF	SRAM_BA	SRAM Memory Space (16 KB)
AHB Controllers Space (0x5000_0000 – 0x501F_FFFF)		
0x5000_0000 – 0x5000_01FF	GCR_BA	System Global Control Registers
0x5000_0200 – 0x5000_02FF	CLK_BA	Clock Control Registers
0x5000_0300 – 0x5000_03FF	INT_BA	Interrupt Multiplexer Control Registers
0x5000_4000 – 0x5000_7FFF	GPIO_BA	GPIO Control Registers
0x5000_8000 – 0x5000_BFFF	PDMA_BA	Peripheral DMA Control Registers
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash Memory Control Registers
0x5001_0000 – 0x5001_03FF	EBI_BA	External Bus Interface Control Registers
APB1 Controllers Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4000_4000 – 0x4000_7FFF	WDT_BA	Watchdog Timer Control Registers
0x4000_8000 – 0x4000_BFFF	RTC_BA	Real Time Clock (RTC) Control Register
0x4001_0000 – 0x4001_3FFF	TMR01_BA	Timer0/Timer1 Control Registers
0x4002_0000 – 0x4002_3FFF	I2C0_BA	I^2C0 Interface Control Registers
0x4003_0000 – 0x4003_3FFF	SPI0_BA	SPI0 with master/slave function Control Registers
0x4003_4000 – 0x4003_7FFF	SPI1_BA	SPI1 with master/slave function Control Registers
0x4004_0000 – 0x4004_3FFF	PWMA_BA	PWM0/1/2/3 Control Registers
0x4005_0000 – 0x4005_3FFF	UART0_BA	UART0 Control Registers
0x4006_0000 – 0x4006_3FFF	USBD_BA	USB 2.0 FS device Controller Registers
0x400E_0000 – 0x400E_FFFF	ADC_BA	Analog-Digital-Converter (ADC) Control Registers
APB2 Controllers Space (0x4010_0000 ~ 0x401F_FFFF)		
0x4011_0000 – 0x4011_3FFF	TMR23_BA	Timer2/Timer3 Control Registers
0x4012_0000 – 0x4012_3FFF	I2C1_BA	I^2C1 Interface Control Registers
0x4014_0000 – 0x4014_3FFF	PWMB_BA	PWM4/5 Control Registers
0x4015_0000 – 0x4015_3FFF	UART1_BA	UART1 Control Registers
0x4015_4000 – 0x4015_7FFF	UART2_BA	UART2 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 6.2-1 Address Space Assignments for On-Chip Controllers

6.2.5 Register Lock

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 REGWRPROT address at 0x5000_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 0x5000_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 “0x5000_0100” to enable register protection.

The protected registers are listed as following table.

Register	Bit	Description
IPRSTC1	[3] EBI_RST	EBI Controller Reset (Write-protection Bit)
IPRSTC1	[2] PDMA_RST	PDMA Controller Reset (Write Protect)
IPRSTC1	[1] CPU_RST	CPU Kernel One-Shot Reset (Write Protect)
IPRSTC1	[0] CHIP_RST	CHIP One-Shot Reset (Write Protect)
BODCR	[7] LVR_EN	Low Voltage Reset Enable Bit (Write Protect)
BODCR	[5] BOD_LPM	Brown-Out Detector Low Power Mode (Write Protect)
BODCR	[3] BOD_RSTEN	Brown-Out Reset Enable Bit (Write Protect)
BODCR	[2:1] BOD_VL	Brown-Out Detector Threshold Voltage Selection (Write Protect)
BODCR	[0] BOD_EN	Brown-Out Detector Enable Bit (Write Protect)
PORCR	[15:0] POR_DIS_CODE	Power-On-Reset Enable Bit (Write Protect)
REGWRPROT	[7:0] REGWRPROT	Register Write-Protection Code (Write Only)
REGWRPROT	[0] REGPROTDIS	Register Write-Protection Disable Index (Read Only)
NMI_SEL	[8] NMI_EN	NMI Interrupt Enable Bit (Write Protect)
PWRCON	[8] PD_WAIT_CPU	Power-Down Entry Condition Control (Write Protect)
PWRCON	[7] PWR_DOWN_EN	System Power-Down Enable Bit (Write Protect)
PWRCON	[5] PD_WU_INT_EN	Power-Down Mode Wake-Up Interrupt Enable Bit (Write Protect)
PWRCON	[4] PD_WU_DLY	Wake-Up Delay Counter Enable Bit (Write Protect)
PWRCON	[3] OSC10K_EN	10 KHz Internal Low Speed RC Oscillator (LIRC) Enable Bit (Write

		Protect)
PWRCON	[2] OSC22M_EN	22.1184 MHz Internal High Speed RC Oscillator (HIRC) Enable Bit (Write Protect)
PWRCON	[1] XTL32K_EN	32.768 KHz External Low Speed Crystal Oscillator (LXT) Enable Bit (Write Protect)
PWRCON	[0] XTL12M_EN	4~24 MHz External High Speed Crystal Oscillator (HXT) Enable Bit (Write Protect)
APBCLK	[0] WDT_EN	Watchdog Timer Clock Enable Bit (Write Protect)
CLKSEL0	[5:3] STCLK_S	Cortex®-M0 SysTick Clock Source Select (Write Protect)
CLKSEL0	[2:0] HCLK_S	HCLK Clock Source Select (Write Protect)
CLKSEL1	[1:0] WDT_S	Watchdog Timer Clock Source Select (Write Protect)
ISPCON	[6] ISPFF	ISP Fail Flag (Write Protect)
ISPCON	[5] LDUEN	LDROM Update Enable Bit (Write Protect)
ISPCON	[4] CFGUEN	Enable Config Update By ISP (Write Protect)
ISPCON	[3] APUEN	APROM Update Enable Bit (Write Protect)
ISPCON	[1] BS	Boot Select (Write Protect)
ISPCON	[0] ISPEN	ISP Enable Bit (Write Protect)
ISPTRG	[0] ISPGO	ISP Start Trigger (Write-Protection Bit)
FATCON	[4] FOMSEL0	Chip Frequency Optimization Mode Select 0 (Write-Protection Bit)
ISPSTA	[6] ISPFF	ISP Fail Flag (Write-Protection Bit)
TCSR0	[31] DBGACK_TMR	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
TCSR1	[31] DBGACK_TMR	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
TCSR2	[31] DBGACK_TMR	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
TCSR3	[31] DBGACK_TMR	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
WTCR	[31] DBGACK_WDT	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
WTCR	[10:8] WTIS	Watchdog Timer Time-Out Interval Selection (Write Protect)
WTCR	[7] WTE	Watchdog Timer Enable Bit (Write Protect)
WTCR	[6] WTIE	Watchdog Timer Time-Out Interrupt Enable Bit (Write Protect)
WTCR	[4] WTWKE	Watchdog Timer Time-Out Wake-Up Function Control (Write Protect)
WTCR	[1] WTRE	Watchdog Timer Reset Enable Bit (Write Protect)
WTCRALT	[1:0] WTRDSEL	Watchdog Timer Reset Delay Selection (Write Protect)

6.2.6 Auto Trim

This chip supports auto-trim function: the HIRC trim (48 MHz and 22.1184 MHz RC oscillator), according to the accurate LXT (32.768 kHz crystal oscillator) or internal USB synchronous mode, automatically gets accurate HIRC output frequency, 0.25 % deviation within all temperature ranges.

For instance, the system needs an accurate 22.1184 MHz clock. In such case, if users do not want to use PLL as the system clock source, they need to solder 32.768 kHz crystal in system, and set

FREQSEL (SYS_IRCTCTL[1:0] trim frequency selection) to “01”, and the auto-trim function will be enabled. Interrupt status bit FREQ_LOCK (SYS_IRCTSTS[0] HIRC frequency lock status) “1” indicates the HIRC output frequency is accurate within 0.25% deviation. To get better results, it is recommended to set both TRIM_LOOP (SYS_IRCTCTL[5:4]) Trim Calculation Loop and TRIM_RETRY_CNT (SYS_IRCTCTL[7:6] Trim Value Update Limitation Count) to “11”.

Another example is that the system needs an accurate 48 MHz clock for USB application. In such case, if neither using use PLL as the system clock source, user has to set FREQSEL (SYS_HIRCTCTL1[1:0] trim frequency selection) to “01”, and the auto-trim function will be enabled. Status bit FREQLOCK (SYS_HIRCTISTS[8] HIRC Frequency Lock Status) “1” indicates the HIRC48 output frequency is accurate within 0.25% deviation.

6.2.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
GCR Base Address: **GCR_BA = 0x5000_0000**				
PDID	GCR_BA+0x00	R	Part Device Identification Number Register	0x2014_0018[1]
RSTSRC	GCR_BA+0x04	R/W	System Reset Source Register	0x0000_00XX
IPRSTC1	GCR_BA+0x08	R/W	Peripheral Reset Control Register 1	0x0000_0000
IPRSTC2	GCR_BA+0x0C	R/W	Peripheral Reset Control Register 2	0x0000_0000
BODCR	GCR_BA+0x18	R/W	Brown-out Detector Control Register	0x0000_008X
TEMPCR	GCR_BA+0x1C	R/W	Temperature Sensor Control Register	0x0000_0000
PORCR	GCR_BA+0x24	R/W	Power-on-reset Controller Register	0x0000_XXXX
GPA_MFP	GCR_BA+0x30	R/W	GPIOA Multiple Function and Input Type Control Register	0x0000_0000
GPB_MFP	GCR_BA+0x34	R/W	GPIOB Multiple Function and Input Type Control Register	0x0000_0000
GPC_MFP	GCR_BA+0x38	R/W	GPIOC Multiple Function and Input Type Control Register	0x0000_0000
GPE_MFP	GCR_BA+0x40	R/W	GPIOE Multiple Function and Input Type Control Register	0x0000_0000
GPF_MFP	GCR_BA+0x44	R/W	GPIOF Multiple Function and Input Type Control Register	0x0000_000X
ALT_MFP	GCR_BA+0x50	R/W	Alternative Multiple Function Pin Control Register	0x0000_0000
ALT_MFP2	GCR_BA+0x5C	R/W	Alternative Multiple Function Pin Control Register 2	0x0000_0000
IRCTCTL	GCR_BA+0x80	R/W	IRC Trim Control Register	0x0000_0000
IRCTIEN	GCR_BA+0x84	R/W	IRC Trim Interrupt Enable Register	0x0000_0000
IRCTSTS	GCR_BA+0x88	R/W	IRC Trim Interrupt Status Register	0x0000_0000
HIRCTCTL	GCR_BA+0x90	R/W	HIRC Trim Control Register	0x0008_0000
HIRCTIEN	GCR_BA+0x94	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000
HIRCTSTS	GCR_BA+0x98	R/W	HIRC Trim Interrupt Status Register	0x0000_0000
REGWRPROT	GCR_BA+0x100	R/W	Register Write Protection Register	0x0000_0000

Note: [1] It depends on the part number.

6.2.8 Register Description

Part Device ID Code Register (PDID)

Register	Offset	R/W	Description	Reset Value
PDID	GCR_BA+0x00	R	Part Device Identification Number Register	0x2014_0018[1]

[1] Each 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** This register reflects device part number code. Software can read this register to identify which device is used.

System Reset Source Register (RSTSRC)

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

Register	Offset	R/W	Description	Reset Value
RSTSRC	GCR_BA+0x04	R/W	System Reset Source 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**
RSTS_CPU	**Reserved**	**RSTS_SYS**	**RSTS_BOD**	**RSTS_LVR**	**RSTS_WDT**	**RSTS_RESET**	**RSTS_POR**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**RSTS_CPU**	**CPU Reset Flag** The RSTS_CPU flag Is set by hardware if software writes CPU_RST (IPRSTC1[1]) 1 To reset Cortex®-M0 kernel and flash memory controller (FMC). 0 = No reset from CPU. 1 = Cortex®-M0 CPU kernel and FMC are reset by software setting CPU_RST(IPRSTC1[1]) to 1. **Note:** Write 1 to clear this bit to 0.
[6]	**Reserved**	Reserved.
[5]	**RSTS_SYS**	**SYS Reset Flag** The RSTS_SYS flag Is set by the "Reset Signal" from the Cortex®-M0 kernel 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 bit SYSRESETREQ (AIRCR[2], Application Interrupt and Reset Control Register, address = 0xE000ED0C) in system control registers of Cortex®-M0 kernel. **Note:** Write 1 to clear this bit to 0.
[4]	**RSTS_BOD**	**Brown-Out Detector Reset Flag** The RSTS_BOD 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]	**RSTS_LVR**	**Low Voltage Reset Flag** The RSTS_LVR 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 = The LVR controller had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[2]	**RSTS_WDT**	**Watchdog Timer Reset Flag** The RSTS_WDT 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. **Note1:** Write 1 to clear this bit to 0. **Note2:** Watchdog Timer register WTRF(WTCR[2]) bit is set if the system has been reset by WDT time-out reset. Window Watchdog Timer register WWDTRF(WWDTSR) bit is set if the system has been reset by WWDT time-out reset.
[1]	**RSTS_RESET**	**Reset Pin Reset Flag** The RSTS_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 = The Pin nRESET had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[0]	**RSTS_POR**	**Power-On Reset Flag** The RSTS_POR Flag is set by the “Reset Signal” from the Power-On Reset (POR) vontroller or bit CHIP_RST (IPRSTC1[0]) to indicate the previous reset source 0 = No reset from POR or CHIP_RST (IPRSTC1[0]). 1 = Power-on Reset (POR) or CHIP_RST (IPRSTC1[0]) had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.

Peripheral Reset Control Register 1 (IPRSTC1)

Register	Offset	R/W	Description	Reset Value
IPRSTC1	GCR_BA+0x08	R/W	Peripheral Reset Control Register 1	0x0000_0000

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

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**EBI_RST**	**EBI Controller Reset (Write-protection Bit)** Set this bit to 1 will generate a reset signal to the EBI. User need to set this bit to 0 to release from the reset state. This bit is the protected bit, It means programming this bit needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100 1 = EBI controller reset 0 = EBI controller normal operation
[2]	**PDMA_RST**	**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. **Note1:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100. **Note2:** Setting PDMA_RST bit to 1 will generate asynchronous reset signal to PDMA module. Users need to set PDMA_RST to 0 to release PDMA module from reset state.
[1]	**CPU_RST**	**CPU Kernel One-Shot Reset (Write Protect)** Setting this bit will only reset the CPU kernel and Flash Memory Controller(FMC), and this bit will automatically return 0 after the two clock cycles 0 = CPU normal operation. 1 = CPU one-shot reset. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[0]	**CHIP_RST**	**CHIP One-Shot Reset (Write Protect)** Setting this bit will reset the whole chip, including CPU kernel and all peripherals, and this bit will automatically return to 0 after the 2 clock cycles.

		The CHIP_RST is the same as the POR reset, all the chip controllers are reset and the chip setting from flash are also reload. For the difference between CHIP_RST and SYSRESETREQ, please refer to section 5.2.2 0 = CHIP normal operation. 1 = CHIP one-shot reset. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Peripheral Reset Control Register 2 (IPRSTC2)

Setting these bits to 1 will generate asynchronous reset signals to the corresponding module. User needs to set these bits to 0 to release the corresponding module from reset state.

Register	Offset	R/W	Description	Reset Value
IPRSTC2	GCR_BA+0x0C	R/W	Peripheral Reset Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			ADC_RST	USBD_RST	Reserved		
23	22	21	20	19	18	17	16
Reserved		PWM45_RST	PWM03_RST	Reserved	UART2_RST	UART1_RST	UART0_RST
15	14	13	12	11	10	9	8
Reserved		SPI1_RST	SPI0_RST	Reserved		I2C1_RST	I2C0_RST
7	6	5	4	3	2	1	0
Reserved		TMR3_RST	TMR2_RST	TMR1_RST	TMR0_RST	GPIO_RST	Reserved

Bits	Description	
[31:39]	**Reserved**	Reserved.
[28]	**ADC_RST**	**ADC Controller Reset** 0 = ADC controller normal operation. 1 = ADC controller reset.
[27]	**USBD_RST**	**USB Device Controller Reset** 0 = USB device controller normal operation. 1 = USB device controller reset.
[26:22]	**Reserved**	Reserved.
[21]	**PWM45_RST**	**PWM45 Controller Reset** 0 = PWM45 controller normal operation. 1 = PWM45 controller reset.
[20]	**PWM03_RST**	**PWM03 Controller Reset** 0 = PWM03 controller normal operation. 1 = PWM03 controller reset.
[19]	**Reserved**	Reserved.
[18]	**UART2_RST**	**UART2 Controller Reset** 0 = UART2 controller normal operation. 1 = UART2 controller reset.
[17]	**UART1_RST**	**UART1 Controller Reset** 0 = UART1 controller normal operation. 1 = UART1 controller reset.
[16]	**UART0_RST**	**UART0 Controller Reset** 0 = UART0 controller normal operation.

		1 = UART0 controller reset.
[15:14]	**Reserved**	Reserved.
[13]	**SPI1_ RST**	**SPI1 Controller Reset** 0 = SPI1 controller normal operation. 1 = SPI1 controller reset.
[12]	**SPI0_ RST**	**SPI0 Controller Reset** 0 = SPI0 controller normal operation. 1 = SPI0 controller reset.
[11:10]	**Reserved**	Reserved.
[9]	**I2C1_RST**	**I^{2}C1 Controller Reset** 0 = I^{2}C1 controller normal operation. 1 = I^{2}C1 controller reset.
[8]	**I2C0_RST**	**I^{2}C0 Controller Reset** 0 = I^{2}C0 controller normal operation. 1 = I^{2}C0 controller reset.
[7:6]	**Reserved**	Reserved.
[5]	**TMR3_RST**	**Timer3 Controller Reset** 0 = Timer3 controller normal operation. 1 = Timer3 controller reset.
[4]	**TMR2_RST**	**Timer2 Controller Reset** 0 = Timer2 controller normal operation. 1 = Timer2 controller reset.
[3]	**TMR1_RST**	**Timer1 Controller Reset** 0 = Timer1 controller normal operation. 1 = Timer1 controller reset.
[2]	**TMR0_RST**	**Timer0 Controller Reset** 0 = Timer0 controller normal operation. 1 = Timer0 controller reset.
[1]	**GPIO_RST**	**GPIO Controller Reset** 0 = GPIO controller normal operation. 1 = GPIO controller reset.
[0]	**Reserved**	Reserved.

Brown-out Detector Control Register (BODCR)

Partial of the BODCR control registers values are initiated by the flash configuration and partial bits are write-protected bit. Programming write-protected bits needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Register	Offset	R/W	Description	Reset Value
BODCR	GCR_BA+0x18	R/W	Brown-out Detector Control Register	0x0000_008X

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**LVR_EN**	**Low Voltage Reset Enable Bit (Write Protect)** The LVR function reset 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 – After enabling the bit, the LVR function will be active with 100us delay for LVR output stable (default). **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[6]	**BOD_OUT**	**Brown-Out Detector Output Status** 0 = Brown-out Detector output status is 0. It means the detected voltage is higher than BOD_VL setting or BOD_EN is 0. 1 = Brown-out Detector output status is 1. It means the detected voltage is lower than BOD_VL setting. If the BOD_EN is 0, BOD function disabled, this bit always responds to 0.
[5]	**BOD_LPM**	**Brown-Out Detector Low Power Mode (Write Protect)** 0 = BOD operated in Normal mode (default). 1 = BOD Low Power mode Enabled. **Note1:** The BOD consumes about 100 uA in Normal mode, and the low power mode can reduce the current to about 1/10 but slow the BOD response. **Note2:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[4]	**BOD_INTF**	**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 BOD_VL setting. 1 = When Brown-out Detector detects the V_{DD} is dropped down through the voltage of

		BOD_VL setting or the V_{DD} is raised up through the voltage of BOD_VL setting, this bit is set to 1 and the Brown-out interrupt is requested if Brown-out interrupt is enabled. **Note:** Write 1 to clear this bit to 0.
[3]	**BOD_RSTEN**	**Brown-Out Reset Enable Bit (Write Protect)** 0 = Brown-out "INTERRUPT" function Enabled. 1 = Brown-out "RESET" function Enabled. While the Brown-out Detector function is enabled (BOD_EN high) and BOD reset function is enabled (BOD_RSTEN high), BOD will assert a signal to reset chip when the detected voltage is lower than the threshold (BOD_OUT high). **Note1:** While the BOD function is enabled (BOD_EN high) and BOD interrupt function is enabled (BOD_RSTEN low), BOD will assert an interrupt if BOD_OUT is high. BOD interrupt will keep till to the BOD_EN set to 0. BOD interrupt can be blocked by disabling the NVIC BOD interrupt or disabling BOD function (set BOD_EN low). **Note2:** The default value is set by flash controller user configuration register CBORST(CONFIG0[20]) bit. **Note3:** This bit is the protected bit. It means programming this needs to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[2:1]	**BOD_VL**	**Brown-Out Detector Threshold Voltage Selection (Write Protect)** The default value is set by flash momory controller user configuration register CBOV(CONFIG0[22:21]) bit . 00 = Brown-out voltage is 2.2V. 01 = Brown-out voltage is 2.7V. 10 = Brown-out voltage is 3.7V. 11 = Brown-out voltage is 4.4V. **Note:** This bit is the protected bit. It means programming this needs to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[0]	**BOD_EN**	**Brown-Out Detector Enable Bit (Write Protect)** The default value is set by flash memory controller user configuration register CBODEN(CONFIG0[23]) bit. 0 = Brown-out Detector function Disabled. 1 = Brown-out Detector function Enabled. **Note:** This bit is the protected bit. It means programming this needs to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Temperature Sensor Control Register (TEMPCR)

Register	Offset	R/W	Description	Reset Value
TEMPCR	GCR_BA+0x1C	R/W	Temperature Sensor 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							**VTEMP_EN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**VTEMP_EN**	**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 can be obtained from ADC conversion result by ADC channel selecting channel 7 and alternative multiplexer channel selecting temperature sensor. Please refer to the ADC function chapter for detail ADC conversion functional description.

Power-on-Reset Control Register (PORCR)

Register	Offset	R/W	Description	Reset Value
PORCR	GCR_BA+0x24	R/W	Power-on-reset Controller Register	0x0000_XXXX

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**POR_DIS_CODE**	**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 Timer reset, Window Watchdog Timer reset, LVR reset, BOD reset, ICE reset command and the software-chip reset function **Note:** This bit is the protected bit. It means programming this needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

GPIOA Multiple Function Pin and Input Type Control Register (GPA_MFP)

Register	Offset	R/W	Description	Reset Value
GPA_MFP	GCR_BA+0x30	R/W	GPIOA Multiple Function and Input Type Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**GPA_TYPEn**	**Trigger Function Selection** 0 = GPIOA[15:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOA[15:0] I/O input Schmitt Trigger function Enabled.
[15]	**GPA_MFP15**	**PA.15 Pin Function Selection** 0 = GPIOA function is selected. 1 = PWM3 function is selected.
[14]	**GPA_MFP14**	**PA.14 Pin Function Selection** Bits EBI_HB_EN[7] (ALT_MFP[23]), EBI_EN (ALT_MFP[11]) and GPA_MFP[14] determine the PA.14 function. (EBI_HB_EN, EBI_EN, GPA_MFP14) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = PWM2 function is selected. (1, 1, 1) = AD15 function is selected.
[13]	**GPA_MFP13**	**PA.13 Pin Function Selection** Bits EBI_HB_EN[6] (ALT_MFP[22]), EBI_EN (ALT_MFP[11]) and GPA_MFP[13] determine the PA.13 function. (EBI_HB_EN, EBI_EN, GPA_MFP13) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = PWM1 function is selected. (1, 1, 1) = AD14 function is selected.
[12]	**GPA_MFP12**	**PA.12 Pin Function Selection** Bits EBI_HB_EN[5] (ALT_MFP[21]), EBI_EN (ALT_MFP[11]) and GPA_MFP[12] determine the PA.12 function. (EBI_HB_EN, EBI_EN, GPA_MFP12) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = PWM0 function is selected. (1, 1, 1) = AD13 function is selected.

[11]	**GPA_MFP11**	**PA.11 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]) and GPA_MFP[11] determine the PA.11 function. (EBI_EN, GPA_MFP11) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = I2C1_SCL function is selected. (1, 1) = nRD(EBI) function is selected.
[10]	**GPA_MFP10**	**PA.10 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]) and GPA_MFP[10] determine the PA.10 function. (EBI_EN, GPA_MFP10) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = I2C1_SDA function is selected. (1, 1) = nWR(EBI) function is selected.
[9]	**GPA_MFP9**	**PA.9 Pin Function Selection** Bit GPA_MFP[9] determines the PA.9 function. 0 = GPIO function is selected. 1 = I2C0_SCL function is selected.
[8]	**GPA_MFP8**	**PA.8 Pin Function Selection** Bit GPA_MFP[8] determines the PA.9 function. 0 = GPIO function is selected to the pin PA.8. 1 = I2C0_SDA function is selected to the pin PA.8.
[7]	**GPA_MFP7**	Reserved.
[6]	**GPA_MFP6**	**PA.6 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]) and GPA_MFP[6] determine the PA.6 function. (EBI_EN, GPA_MFP6) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = ADC6 function is selected. (1, 1) = AD7 function is selected.
[5]	**GPA_MFP5**	**PA.5 Pin Function Selection** Bits EBI_HB_EN[0] (ALT_MFP[16]), EBI_EN (ALT_MFP[11]) and GPA_MFP[5] determine the PA.5 function. (EBI_HB_EN, EBI_EN, GPA_MFP5) value and function mapping is as following list, (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC5 function is selected. (1, 1, 1) = AD8 function is selected.
[4]	**GPA_MFP4**	**PA.4 Pin Function Selection** Bits EBI_HB_EN[1] (ALT_MFP[17]), EBI_EN (ALT_MFP[11]) and GPA_MFP[4] determine the PA.4 function. (EBI_HB_EN, EBI_EN, GPA_MFP4) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC4 function is selected. (1, 1, 1) = AD9 function is selected.

[3]	**GPA_MFP3**	**PA.3 Pin Function Selection** Bits EBI_HB_EN[2] (ALT_MFP[18]), EBI_EN (ALT_MFP[11]) and GPA_MFP[3] determine the PA.3 function. (EBI_HB_EN, EBI_EN, GPA_MFP3) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC3 function is selected. (1, 1, 1) = AD10 function is selected.
[2]	**GPA_MFP2**	**PA.2 Pin Function Selection** Bits EBI_HB_EN[3] (ALT_MFP[19]), EBI_EN (ALT_MFP[11]) and GPA_MFP[2] determine the PA.2 function. (EBI_HB_EN, EBI_EN, GPA_MFP2) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC2 function is selected. (1, 1, 1) = AD11 function is selected.
[1]	**GPA_MFP1**	**PA.1 Pin Function Selection** Bit EBI_HB_EN[4] (ALT_MFP[20]), EBI_EN (ALT_MFP[11]) and GPA_MFP[1] determine the PA.1 function. (EBI_HB_EN, EBI_EN, GPA_MFP1) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC1 function is selected. (1, 1, 1) = AD12 function is selected.
[0]	**GPA_MFP0**	**PA.0 Pin Function Selection** 0 = GPIO function is selected. 1 = ADC0 function is selected.

GPIOB Multiple Function Pin and Input Type Control Register (GPB_MFP)

Register	Offset	R/W	Description	Reset Value
GPB_MFP	GCR_BA+0x34	R/W	GPIOB Multiple Function and Input Type Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**GPB_TYPEn**	**Trigger Function Selection** 0 = GPIOB[15:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOB[15:0] I/O input Schmitt Trigger function Enabled.
[15]	**GPB_MFP15**	**PB.15 Pin Function Selection** Bits PB14_15_EBI (ALT_MFP2[1]), PB15_T0EX (ALT_MFP[24]), PB15_TM0 (ALT_MFP2[2]) and GPB_MFP[15] determine the PB.15 function. (PB14_15_EBI, PB15_T0EX, PB15_TM0, GPB_MFP15) value and function mapping is as following list. (0, 0, 0, 0) = GPIO function is selected. (0, 0, 0, 1) = INT1 function is selected. (0 ,0, 1, 1) = TM0 function is selected. (0, 1, 0, 0) = ADC11 function is selected. (0, 1, 0, 1) = TM0_EXT function is selected. (1, 0, 0, 1) = AD6 function is selected.
[14]	**GPB_MFP14**	**PB.14 Pin Function Selection** Bits PB14_15_EBI (ALT_MFP2[1]) and GPB_MFP[14] determine the PB.14 function. (PB14_15_EBI, GPB_MFP14) value and function mapping is as following list (0, 0) = GPIO function is selected. (0, 1) = INT0 function is selected. (1, 1) = AD0 function is selected.
[13]	**GPB_MFP13**	**PB.13 Pin Function Selection** 0 = GPIO function is selected to the pin PB.13. 1 = AD1 function is selected.
[12]	**GPB_MFP12**	Reserved
[11]	**GPB_MFP11**	**PB.11 Pin Function Selection** Bits PB11_PWM4 (ALT_MFP[4]) and GPB_MFP[11] determine the PB.11 function. (PB11_PWM4, GPB_MFP11) value and function mapping is as following list.

		(0, 0) = GPIO function is selected. (0, 1) = TM3 function is selected. (1, 1) = PWM4 function is selected.
[10]	**GPB_MFP10**	**PB.10 Pin Function Selection** Bits PB10_S01 (ALT_MFP[0]) and GPB_MFP[10] determine the PB.10 function. (PB10_S01, GPB_MFP10) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = TM2 function is selected. (1, 1) = UART2_RXD function is selected.
[9]	**GPB_MFP9**	**PB.9 Pin Function Selection** Bits PB9_S11 (ALT_MFP[1]) and GPB_MFP[9] determine the PB.9 function. (PB9_S11, GPB_MFP9) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = TM1 function is selected. (1, 1) = UART2_TXD function is selected.
[8]	**GPB_MFP8**	**PB.8 Pin Function Selection** Bits PB8_CLKO (ALT_MFP[29]) and GPB_MFP[8] determine the PB.8 function. (PB8_CLKO, GPB_MFP8) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = TM0 function is selected to the pin PB.8. (1, 0) = STADC function is selected to the pin PB.8. (1, 1) = CLKO function is selected to the pin PB.8.
[7]	**GPB_MFP7**	**PB.7 Pin Function Selection** Bit EBI_EN (ALT_MFP[11]), GPB_MFP[7] determines the PB.7 function. (EBI_EN, GPB_MFP7) value and function mapping is as following list. (0, 0) = GPIO function is selected to the pin PB.7. (0, 1) = UART1_nCTS function is selected to the pin PB.7. (1, 1) = nCS(EBI) function is selected to the pin PB.7.
[6]	**GPB_MFP6**	**PB.6 Pin Function Selection** Bit EBI_EN (ALT_MFP[11]), GPB_MFP[6] determines the PB.6 function. (EBI_EN, GPB_MFP6) value and function mapping is as following list. (0, 0) = GPIO function is selected to the pin PB.6. (0, 1) = UART1_nRTS function is selected to the pin PB.6. (1, 1) = ALE(EBI) function is selected to the pin PB.6.
[5]	**GPB_MFP5**	**PB 5 Pin Function Selection** Bit GPB_MFP[5] determines the PB.5 function. 0 = GPIO function is selected to the pin PB.5. 1 = UART1_TXD function is selected to the pin PB.5.
[4]	**GPB_MFP4**	**PB.4 Pin Function Selection** Bit GPB_MFP[4] determines the PB.4 function. 0 = GPIO function is selected to the pin PB.4. 1 = UART1_RXD function is selected to the pin PB.4.
[3]	**GPB_MFP3**	**PB.3 Pin Function Selection** Bits EBI_nWRH_EN (ALT_MFP[14]), EBI_EN (ALT_MFP[11]), PB3_TM3 (ALT_MFP2[5]), PB3_T3EX (ALT_MFP[27]) and GPB_MFP[3] determine the PB.3 function. (EBI_nWRH_EN, EBI_EN, PB3_TM3, PB3_T3EX, GPB_MFP3) value and function mapping is as following list.

		(0, 0, 0, 0, 0) = GPIO function is selected. (0, 0, 0, 0, 1) = UART0_nCTS function is selected. (0, 0, 0, 1, 1) = TM3_EXT function is selected. (0, 0, 1, 0, 1) = TM3 function is selected. (1, 1, 0, 0, 1) = nWRH(EBI) function is selected.
[2]	**GPB_MFP2**	**PB.2 Pin Function Selection** Bits EBI_nWRL_EN (ALT_MFP[13]), EBI_EN (ALT_MFP[11]), PB2_TM2 (ALT_MFP2[4]), PB2_T2EX (ALT_MFP[26]) and GPB_MFP[2] determine the PB.2 function. (EBI_nWRL_EN, EBI_EN, PB2_TM2, PB2_T2EX, GPB_MFP2) value and function mapping is as following list. (0, 0, 0 , 0, 0) = GPIO function is selected. (0, 0, 0, 0, 1) = UART0_nRTS function is selected. (0, 0, 0, 1, 1) = TM2_EXT function is selected. (0, 0, 1, 0, 1) = TM2 function is selected. (1, 1, 0, 0, 1) = nWRL(EBI) function is selected.
[1]	**GPB_MFP1**	**PB.1 Pin Function Selection** Bit GPB_MFP[1] determines the PB.1 function. 0 = GPIO function is selected to the pin PB.1. 1 = UART0_TXD function is selected to the pin PB.1.
[0]	**GPB_MFP0**	**PB.0 Pin Function Selection** Bit GPB_MFP[0] determines the PB.0 function. 0 = GPIO function is selected to the pin PB.0. 1 = UART0_RXD function is selected to the pin PB.0.

GPIOC Multiple Function Pin and input Type Control Register (GPC_MFP)

Register	Offset	R/W	Description	Reset Value
GPC_MFP	GCR_BA+0x38	R/W	GPIOC Multiple Function and Input Type Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**GPC_TYPEn**	**Trigger Function Selection** 0 = GPIOC[15:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOC[15:0] I/O input Schmitt Trigger function Enabled.
[15]	**GPC_MFP15**	**PC.15 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]) and GPC_MFP[15] determine the PC.15 function. (EBI_EN, GPC_MFP15) value and function mapping is as following list (0, 0) = GPIO function is selected. (0, 1) = ADC9 function is selected. (1, 1) = AD3 function is selected.
[14]	**GPC_MFP14**	**PC.14 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]) and GPC_MFP[14] determine the PC.14 function. (EBI_EN, GPC_MFP14) value and function mapping is as following list (0, 0) = GPIO function is selected. (0, 1) = ADC10 function is selected. (1, 1) = AD2 function is selected.
[13]	**GPC_MFP13**	Reserved
[12]	**GPC_MFP12**	Reserved
[11]	**GPC_MFP11**	**PC.11 Pin Function Selection** Bit GPC_MFP[11] determines the PC.11 function. 0 = GPIO function is selected to the pin PC.11. 1 = SPI1_MOSI0 function is selected to the pin PC.11.
[10]	**GPC_MFP10**	**PC.10 Pin Function Selection** Bit GPC_MFP[10] determines the PC.10 function. 0 = GPIO function is selected to the pin PC.10. 1 = SPI1_MISO0 function is selected to the pin PC.10.

[9]	**GPC_MFP9**	**PC.9 Pin Function Selection** Bit GPC_MFP[9] determines the PC.9 function. 0 = GPIO function is selected to the pin PC.9. 1 = SPI1_CLK function is selected to the pin PC.9.
[8]	**GPC_MFP8**	**PC.8 Pin Function Selection** Bits EBI_MCLK_EN (ALT_MFP[12]), EBI_EN (ALT_MFP[11]), GPC_MFP[8] determine the PC.8 function. (EBI_MCLK_EN, EBI_EN, GPC_MFP8) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected to the pin PC.8. (0, 0 ,1) = SPI1_SS0 function is selected to the pin PC.8. (1, 1, 1) = MCLK(EBI) function is selected to the pin PC.8.
[7]	**GPC_MFP7**	**PC.7 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]) and GPC_MFP[7] determine the PC.7 function. (EBI_EN, GPC_MFP7) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC7 function is selected. (1, 0, 1) = AD5 function is selected.
[6]	**GPC_MFP6**	**PC.6 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]) and GPC_MFP[6] determine the PC.6 function. (EBI_EN, GPB_MFP6) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = ADC9 function is selected. (1, 1) = AD4 function is selected.
[5]	**GPC_MFP5**	Reserved
[4]	**GPC_MFP4**	Reserved
[3]	**GPC_MFP3**	**PC.3 Pin Function Selection** 0 = GPIO function is selected. 1 = SPI0_MOSI0 function is selected.
[2]	**GPC_MFP2**	**PC.2 Pin Function Selection** 0 = GPIO function is selected. 1 = SPI0_MISO0 function is selected.
[1]	**GPC_MFP1**	**PC.1 Pin Function Selection** 0 = GPIO function is selected. 1 = SPI0_CLK function is selected.
[0]	**GPC_MFP0**	**PC.0 Pin Function Selection** 0 = GPIO function is selected. 1 = SPI0_SS0 function is selected.

GPIOE Multiple Function Pin and Input Type Control Register (GPE_MFP)

Register	Offset	R/W	Description	Reset Value
GPE_MFP	GCR_BA+0x40	R/W	GPIOE Multiple Function and Input Type Control Register	0x0000_0000

31	30	29	28	27	26	25	24
GPE_TYPE							
23	22	21	20	19	18	17	16
GPE_TYPE							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		GPE_MFP5	Reserved				

Bits	Description	
[31:16]	**GPE_TYPEn**	**Trigger Function Selection** 0 = GPIOE[15:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOE[15:0] I/O input Schmitt Trigger function Enabled.
[15:6]	**Reserved**	Reserved.
[5]	**GPE_MFP5**	**PE.5 Pin Function Selection** Bits PE5_T1EX (ALT_MFP[25]), PE5_TM1 (ALT_MFP2[3]) and GPE_MFP5 determine the PE.5 function. (PE5_T1EX, PE5_TM1, GPE_MFP5) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = PWM5 function is selected. (1, 0, 1) = TM1_EXT function is selected. (0, 1, 1) = TM1 function is selected.
[4:0]	**Reserved**	Reserved.

GPIOF Multiple Function Pin and Input Type Control Register (GPF_MFP)

Register	Offset	R/W	Description	Reset Value
GPF_MFP	GCR_BA+0x44	R/W	GPIOF Multiple Function and Input Type Control Register	0x0000_000X

Note: The default value of GPF_MFP[3]/GPF_MFP[2] is 1. The default value of GPF_MFP[1]/GPF_MFP[0] is decided by user configuration CGPFMFP(CONFIG0[27]).

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				GPF_TYPE			
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						GPF_MFP1	GPF_MFP0

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**GPF_TYPEn**	**Trigger Function Selection** 0 = GPIOF[3:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOF[3:0] I/O input Schmitt Trigger function Enabled.
[15:2]	**Reserved**	Reserved.
[1]	**GPF_MFP1**	**PF.1 Pin Function Selection** Bit GPF_MFP[1] determines the PF.1 function. 0 = GPIO function is selected to the pin PF.1. 1 = XT1_IN function is selected to the pin PF.1. **Note:** This bit is read only and is decided by user configuration CGPFMFP (CONFIG0[27]).
[0]	**GPF_MFP0**	**PF.0 Pin Function Selection** Bit GPF_MFP[0] determines the PF.0 function 0 = GPIO function is selected to the pin PF.0. 1 = XT1_OUT function is selected to the pin PF.0. **Note:** This bit is read only and is decided by user configuration CGPFMFP (CONFIG0[27]).

Alternative Multiple Function Pin Control Register (ALT_MFP)

Register	Offset	R/W	Description	Reset Value
ALT_MFP	GCR_BA+0x50	R/W	Alternative Multiple Function Pin Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		PB8_CLKO	Reserved	PB3_T3EX	PB2_T2EX	PE5_T1EX	PB15_T0EX
23	22	21	20	19	18	17	16
EBI_HB_EN							
15	14	13	12	11	10	9	8
Reserved	EBI_nWRH_EN	EBI_nWRL_EN	EBI_MCLK_EN	EBI_EN	Reserved		
7	6	5	4	3	2	1	0
Reserved			PB11_PWM4	Reserved		PB9_S11	PB10_S01

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**PB8_CLKO**	**PB.8 Pin Alternative Function Selection** Bits PB8_CLKO (ALT_MFP[29]) and GPB_MFP[8] determine the PB.8 function. (PB8_CLKO, GPB_MFP8) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = TM0 function is selected to the pin PB.8. (1, 0) = STADC function is selected to the pin PB.8. (1, 1) = CLKO function is selected to the pin PB.8.
[28]	**Reserved**	Reserved.
[27]	**PB3_T3EX**	**PB.3 Pin Alternative Function Selection** Bits EBI_nWRH_EN (ALT_MFP[14]), EBI_EN (ALT_MFP[11]), PB3_TM3 (ALT_MFP2[5]), PB3_T3EX (ALT_MFP[27]) and GPB_MFP[3] determine the PB.3 function. (EBI_nWRH_EN, EBI_EN, PB3_TM3, PB3_T3EX, GPB_MFP3) value and function mapping is as following list. (0, 0, 0, 0, 0) = GPIO function is selected. (0, 0, 0, 0, 1) = UART0_nCTS function is selected. (0, 0, 0, 1, 1) = TM3_EXT function is selected. (0, 0, 1, 0, 1) = TM3 function is selected. (1, 1, 0, 0, 1) = nWRH(EBI) function is selected.

[26]	**PB2_T2EX**	**PB.2 Pin Alternative Function Selection** Bits EBI_nWRL_EN (ALT_MFP[13]), EBI_EN (ALT_MFP[11]), PB2_TM2 (ALT_MFP2[4]), PB2_T2EX (ALT_MFP[26]) and GPB_MFP[2] determine the PB.2 function. (EBI_nWRL_EN, EBI_EN, PB2_TM2, PB2_T2EX, GPB_MFP2) value and function mapping is as following list. (0, 0, 0 , 0, 0) = GPIO function is selected. (0, 0, 0, 0, 1) = UART0_nRTS function is selected. (0, 0, 0, 1, 1) = TM2_EXT function is selected. (0, 0, 1, 0, 1) = TM2 function is selected. (1, 1, 0, 0, 1) = nWRL(EBI) function is selected.
[25]	**PE5_T1EX**	**PE.5 Pin Alternative Function Selection** Bits PE5_T1EX (ALT_MFP[25]), PE5_TM1 (ALT_MFP2[3]) and GPE_MFP5 determine the PE.5 function. (PE5_T1EX, PE5_TM1, GPE_MFP5) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = PWM5 function is selected. (1, 0, 1) = TM1_EXT function is selected. (0, 1, 1) = TM1 function is selected.
[24]	**PB15_T0EX**	**PB.15 Pin Alternative Function Selection** Bits PB14_15_EBI (ALT_MFP2[1]), PB15_T0EX (ALT_MFP[24]), PB15_TM0 (ALT_MFP2[2]) and GPB_MFP[15] determine the PB.15 function. (PB14_15_EBI, PB15_T0EX, PB15_TM0, GPB_MFP15) value and function mapping is as following list. (0, 0, 0, 0) = GPIO function is selected. (0, 0, 0, 1) = INT1 function is selected. (0 ,0, 1, 1) = TM0 function is selected. (0, 1, 0, 0) = ADC11 function is selected. (0, 1, 0, 1) = TM0_EXT function is selected. (1, 0, 0, 1) = AD6 function is selected.
[23]	**EBI_HB_EN[7]**	Bits EBI_HB_EN[7] (ALT_MFP[23]), EBI_EN (ALT_MFP[11]) and GPA_MFP[14] determine the PA.14 function. (EBI_HB_EN, EBI_EN, GPA_MFP14) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = PWM2 function is selected. (1, 1, 1) = AD15 function is selected.
[22]	**EBI_HB_EN[6]**	Bits EBI_HB_EN[6] (ALT_MFP[22]), EBI_EN (ALT_MFP[11]) and GPA_MFP[13] determine the PA.13 function. (EBI_HB_EN, EBI_EN, GPA_MFP13) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = PWM1 function is selected. (1, 1, 1) = AD14 function is selected.
[21]	**EBI_HB_EN[5]**	Bits EBI_HB_EN[5] (ALT_MFP[21]), EBI_EN (ALT_MFP[11]) and GPA_MFP[12] determine the PA.12 function. (EBI_HB_EN, EBI_EN, GPA_MFP12) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = PWM0 function is selected. (1, 1, 1) = AD13 function is selected.

[20]	**EBI_HB_EN[4]**	Bit EBI_HB_EN[4] (ALT_MFP[20]), EBI_EN (ALT_MFP[11]) and GPA_MFP[1] determine the PA.1 function. (EBI_HB_EN, EBI_EN, GPA_MFP1) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC1 function is selected. (1, 1, 1) = AD12 function is selected.
[19]	**EBI_HB_EN[3]**	Bits EBI_HB_EN[3] (ALT_MFP[19]), EBI_EN (ALT_MFP[11]) and GPA_MFP[2] determine the PA.2 function. (EBI_HB_EN, EBI_EN, GPA_MFP2) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC2 function is selected. (1, 1, 1) = AD11 function is selected.
[18]	**EBI_HB_EN[2]**	Bits EBI_HB_EN[2] (ALT_MFP[18]), EBI_EN (ALT_MFP[11]) and GPA_MFP[3] determine the PA.3 function. (EBI_HB_EN, EBI_EN, GPA_MFP3) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC3 function is selected. (1, 1, 1) = AD10 function is selected.
[17]	**EBI_HB_EN[1]**	Bits EBI_HB_EN[1] (ALT_MFP[17]), EBI_EN (ALT_MFP[11]) and GPA_MFP[4] determine the PA.4 function. (EBI_HB_EN, EBI_EN, GPA_MFP4) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC4 function is selected. (1, 1, 1) = AD9 function is selected.
[16]	**EBI_HB_EN[0]**	Bits EBI_HB_EN[0] (ALT_MFP[16]), EBI_EN (ALT_MFP[11]) and GPA_MFP[5] determine the PA.5 function. (EBI_HB_EN, EBI_EN, GPA_MFP5) value and function mapping is as following list, (0, 0, 0) = GPIO function is selected. (0, 0, 1) = ADC5 function is selected. (1, 1, 1) = AD8 function is selected.
[15]	**Reserved**	Reserved
[14]	**EBI_nWRH_EN**	Bits EBI_nWRH_EN (ALT_MFP[14]), EBI_EN (ALT_MFP[11]), PB3_TM3 (ALT_MFP2[5]), PB3_T3EX (ALT_MFP[27]) and GPB_MFP[3] determine the PB.3 function. (EBI_nWRH_EN, EBI_EN, PB3_TM3, PB3_T3EX, GPB_MFP3) value and function mapping is as following list. (0, 0, 0, 0, 0) = GPIO function is selected. (0, 0, 0, 0, 1) = UART0_nCTS function is selected. (0, 0, 0, 1, 1) = TM3_EXT function is selected. (0, 0, 1, 0, 1) = TM3 function is selected. (1, 1, 0, 0, 1) = nWRH(EBI) function is selected.
[13]	**EBI_nWRL_EN**	Bits EBI_nWRL_EN (ALT_MFP[13]), EBI_EN (ALT_MFP[11]), PB2_TM2 (ALT_MFP2[4]), PB2_T2EX (ALT_MFP[26]) and GPB_MFP[2] determine the PB.2 function. (EBI_nWRL_EN, EBI_EN, PB2_TM2, PB2_T2EX, GPB_MFP2) value and function mapping is as following list. (0, 0, 0 , 0, 0) = GPIO function is selected. (0, 0, 0, 0, 1) = UART0_nRTS function is selected. (0, 0, 0, 1, 1) = TM2_EXT function is selected. (0, 0, 1, 0, 1) = TM2 function is selected. (1, 1, 0, 0, 1) = nWRL(EBI) function is selected.

[12]	**EBI_MCLK_EN**	Bits EBI_MCLK_EN (ALT_MFP[12]), EBI_EN (ALT_MFP[11]), GPC_MFP[8] determine the PC.8 function. (EBI_MCLK_EN, EBI_EN, GPC_MFP8) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected to the pin PC.8. (0, 0 ,1) = SPI1_SS0 function is selected to the pin PC.8. (1, 1, 1) = MCLK(EBI) function is selected to the pin PC.8.
[11]	**EBI_EN**	EBI_EN is use to switch GPIO function to EBI function (AD[15:0], ALE, RE, WE, CS, MCLK), it need additional registers EBI_EN[7:0] and EBI_MCLK_EN for some GPIO to switch to EBI function(AD[15:8], MCLK)
[10:5]	**Reserved**	Reserved
[4]	**PB11_PWM4**	**PB.11 Pin Alternative Function Selection** Bits PB11_PWM4 (ALT_MFP[4]) and GPB_MFP[11] determine the PB.11 function. (PB11_PWM4, GPB_MFP11) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = TM3 function is selected. (1, 1) = PWM4 function is selected.
[3]	**PB14_S31**	**PB.14 Pin Alternative Function Selection** Bits PB14_15_EBI (ALT_MFP2[1]) and GPB_MFP[14] determine the PB.14 function. (PB14_15_EBI, GPB_MFP14) value and function mapping is as following list (0, 0) = GPIO function is selected. (0, 1) = INT0 function is selected. (1, 1) = AD0 function is selected.
[2]	**Reserved**	Reserved
[1]	**PB9_S11**	**PB.9 Pin Alternative Function Selection** Bits PB9_S11 (ALT_MFP[1]) and GPB_MFP[9] determine the PB.9 function. (PB9_S11, GPB_MFP9) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = TM1 function is selected. (1, 1) = UART2_TXD function is selected.
[0]	**PB10_S01**	**PB.10 Pin Alternative Function Selection** Bits PB10_S01 (ALT_MFP[0]) and GPB_MFP[10] determine the PB.10 function. (PB10_S01, GPB_MFP10) value and function mapping is as following list. (0, 0) = GPIO function is selected. (0, 1) = TM2 function is selected. (1, 1) = UART2_RXD function is selected.

Alternative Multiple Function Pin Control Register 2 (ALT_MFP2)

Register	Offset	R/W	Description	Reset Value
ALT_MFP2	GCR_BA+0x5C	R/W	Alternative Multiple Function Pin Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		PB3_TM3	PB2_TM2	PE5_TM1	PB15_TM0	PB14_15_EBI	Reserved

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**PB3_TM3**	**PB.3 Pin Alternative Function Selection** Bits EBI_nWRH_EN (ALT_MFP[14]), EBI_EN (ALT_MFP[11]), PB3_TM3 (ALT_MFP2[5]), PB3_T3EX (ALT_MFP[27]) and GPB_MFP[3] determine the PB.3 function. (EBI_nWRH_EN, EBI_EN, PB3_TM3, PB3_T3EX, GPB_MFP3) value and function mapping is as following list. (0, 0, 0, 0, 0) = GPIO function is selected. (0, 0, 0, 0, 1) = UART0_nCTS function is selected. (0, 0, 0, 1, 1) = TM3_EXT function is selected. (0, 0, 1, 0, 1) = TM3 function is selected. (1, 1, 0, 0, 1) = nWRH(EBI) function is selected.
[4]	**PB2_TM2**	**PB.2 Pin Alternative Function Selection** Bits EBI_nWRL_EN (ALT_MFP[13]), EBI_EN (ALT_MFP[11]), PB2_TM2 (ALT_MFP2[4]), PB2_T2EX (ALT_MFP[26]) and GPB_MFP[2] determine the PB.2 function. (EBI_nWRL_EN, EBI_EN, PB2_TM2, PB2_T2EX, GPB_MFP2) value and function mapping is as following list. (0, 0, 0 , 0, 0) = GPIO function is selected. (0, 0, 0, 0, 1) = UART0_nRTS function is selected. (0, 0, 0, 1, 1) = TM2_EXT function is selected. (0, 0, 1, 0, 1) = TM2 function is selected. (1, 1, 0, 0, 1) = nWRL(EBI) function is selected.
[3]	**PE5_TM1**	**PE.5 Pin Alternative Function Selection** Bits PE5_T1EX (ALT_MFP[25]), PE5_TM1 (ALT_MFP2[3]) and GPE_MFP5 determine the PE.5 function. (PE5_T1EX, PE5_TM1, GPE_MFP5) value and function mapping is as following list. (0, 0, 0) = GPIO function is selected. (0, 0, 1) = PWM5 function is selected. (1, 0, 1) = TM1_EXT function is selected. (0, 1, 1) = TM1 function is selected.

[2]	**PB15_TM0**	**PB.15 Pin Alternative Function Selection** Bits PB14_15_EBI (ALT_MFP2[1]), PB15_T0EX (ALT_MFP[24]), PB15_TM0 (ALT_MFP2[2]) and GPB_MFP[15] determine the PB.15 function. (PB14_15_EBI, PB15_T0EX, PB15_TM0, GPB_MFP15) value and function mapping is as following list. (0, 0, 0, 0) = GPIO function is selected. (0, 0, 0, 1) = INT1 function is selected. (0 ,0, 1, 1) = TM0 function is selected. (0, 1, 0, 0) = ADC11 function is selected. (0, 1, 0, 1) = TM0_EXT function is selected. (1, 0, 0, 1) = AD6 function is selected.
[1]	**PB14_15_EBI**	**PB .14 and PB.15 Pin Alternative Function Selection** Bits PB14_15_EBI (ALT_MFP2[1]), PB15_T0EX (ALT_MFP[24]), PB15_TM0 (ALT_MFP2[2]) and GPB_MFP[15] determine the PB.15 function. (PB14_15_EBI, PB15_T0EX, PB15_TM0, GPB_MFP15) value and function mapping is as following list. (0, 0, 0, 0) = GPIO function is selected. (0, 0, 0, 1) = INT1 function is selected. (0 ,0, 1, 1) = TM0 function is selected. (0, 1, 0, 0) = ADC11 function is selected. (0, 1, 0, 1) = TM0_EXT function is selected. (1, 0, 0, 1) = AD6 function is selected. Bits PB14_15_EBI (ALT_MFP2[1]) and GPB_MFP[14] determine the PB.14 function. (PB14_15_EBI, GPB_MFP14) value and function mapping is as following list (0, 0) = GPIO function is selected. (0, 1) = INT0 function is selected. (1, 1) = AD0 function is selected.
[0]	**Reserved**	Reserved.

IRC Trim Control Register (SYS_IRCTCTL)

Register	Offset	R/W	Description	Reset Value
IRCTCTL	GCR_BA+0x80	R/W	IRC Trim 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							CLKERR_STOP_EN
7	6	5	4	3	2	1	0
TRIM_RETRY_CNT		TRIM_LOOP		Reserved		TRIM_SEL	

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CLKERR_STOP_EN**	**Clock Error Stop Enable Bit** 0 = The trim operation is kept going if clock is inaccuracy. 1 = The trim operation is stopped if clock is inaccuracy.
[7:6]	**TRIM_RETRY_CNT**	**Trim Value Update Limitation Count** The field defines that how many times of HIRC trim value is updated by auto trim circuit before the HIRC frequency 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 TRIM_SEL will be cleared to 00. 00 = Trim retry count limitation is 64. 01 = Trim retry count limitation is 128. 10 = Trim retry count limitation is 256. 11 = Trim retry count limitation is 512.
[5:4]	**TRIM_LOOP**	**Trim Calculation Loop** This field defines that trim value calculation is based on how many 32.768 kHz clocks in. For example, if TRIM_LOOP is set as 00, auto trim circuit will calculate trim value based on the average frequency difference in 4 32.768 kHz clock. 00 = Trim value calculation is based on average difference in 4 clocks. 01 = Trim value calculation is based on average difference in 8 clocks. 10 = Trim value calculation is based on average difference in 16 clocks. 11 = Trim value calculation is based on average difference in 32 clocks.
[3:2]	**Reserved**	Reserved.
[1:0]	**TRIM_SEL**	**Trim Frequency Selection** This field indicates the target frequency of internal 22.1184 MHz high speed oscillator will trim to precise 22.1184MHz or 24MHz automatically. If no any target frequency is selected (TRIM_SEL is 00), the HIRC auto trim function is

		disabled. During auto trim operation, if clock error detected because of CLKERR_STOP_EN is set to 1 or trim retry limitation counts reached, this field will be cleared to 00 automatically. 00 = HIRC auto trim function Disabled. 01 = HIRC auto trim function Enabled and HIRC trimmed to 22.1184 MHz. 10 = HIRC auto trim function Enabled and HIRC trimmed to 24 MHz. 11 = Reserved.

IRC Trim Interrupt Enable Register (SYS_IRCTIEN)

Register	Offset	R/W	Description	Reset Value
IRCTIEN	GCR_BA+0x84	R/W	IRC 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					**CLKERR_IEN**	**TRIM_FAIL_IEN**	**Reserved**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CLKERR_IEN**	**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 CLKERR_INT (IRCTRIMINT[2]) is set during auto trim operation. An interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = CLKERR_INT (IRCTRIMINT[2]) status to trigger an interrupt to CPU Disabled. 1 = CLKERR_INT (IRCTRIMINT[2]) status to trigger an interrupt to CPU Enabled.
[1]	**TRIM_FAIL_IEN**	**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 TRIM_SEL (IRCTCTL[1:0]). If this bit is high and TRIM_FAIL_INT (IRCTRIMINT[1]) is set during auto trim operation. An interrupt will be triggered to notify that HIRC trim value update limitation count was reached. 0 = TRIM_FAIL_INT (IRCTRIMINT[1]) status to trigger an interrupt to CPU Disabled. 1 = TRIM_FAIL_INT (IRCTRIMINT[1]) status to trigger an interrupt to CPU Enabled.
[0]	**Reserved**	Reserved.

IRC Trim Interrupt Status Register (SYS_IRCTSTS)

Register	Offset	R/W	Description	Reset Value
IRCTSTS	GCR_BA+0x88	R/W	IRC 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					**CLKERR_INT**	**TRIM_FAIL_INT**	**FREQ_LOCK**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CLKERR_INT**	**Clock Error Interrupt Status** When the frequency of external 32.768 kHz low speed crystal or internal 22.1184 MHz high speed oscillator 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 TRIM_SEL (IRCTCTL[1:0]) will be cleared to 00 by hardware automatically if CLKERR_STOP_EN (IRCTCTL[8]) is set to 1. If this bit is set and CLKERR_IEN (IRCTIEN [2]) is high, an interrupt will be triggered to notify the clock frequency is inaccuracy. Write 1 to clear this to 0. 0 = Clock frequency is accurate. 1 = Clock frequency is inaccurate.
[1]	**TRIM_FAIL_INT**	**Trim Failure Interrupt Status** This bit indicates that internal 22.1184 MHz high speed oscillator trim value update limitation count reached and the internal 22.1184 MHz high speed oscillator clock frequency still doesn't be locked. Once this bit is set, the auto trim operation stopped and TRIM_SEL (IRCTCTL[1:0]) will be cleared to 00 by hardware automatically. If this bit is set and TRIM_FAIL_IEN (IRCTIEN[1]) is high, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. Write 1 to clear this to 0. 0 = Trim value update limitation count did not reach. 1 = Trim value update limitation count reached and internal 22.1184 MHz high speed oscillator frequency was still not locked.
[0]	**FREQ_LOCK**	**HIRC Frequency Lock Status** This bit indicates the internal 22.1184 MHz high speed oscillator frequency is locked. This is a status bit and doesn't trigger any interrupt.

HIRC Trim Control Register (SYS_HIRCTCTL)

Register	Offset	R/W	Description	Reset Value
HIRCTCTL	GCR_BA+0x90	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						**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 1 to 31, 0 is reserved. **Note**: This field is effective only when the BOUNDEN(SYS_HIRCTCTL[9]) is enable.
[15:10]	**Reserved**	Reserved.
[9]	**BOUNDEN**	**Boundary Enable** 0 = Boundary function is disable. 1 = Boundary function is enable.
[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_HIRCTIEN)

Register	Offset	R/W	Description	Reset Value
HIRCTIEN	GCR_BA+0x94	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_HIRCTSTS[2]) is set during auto trim operation, an interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = Disable CLKERRIF(SYS_HIRCTSTS[2]) status to trigger an interrupt to CPU. 1 = Enable CLKERRIF(SYS_HIRCTSTS[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_HIRCTCTL[1:0]). If this bit is high and TFAILIF(SYS_HIRCTSTS[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_HIRCTSTS[1]) status to trigger an interrupt to CPU. 1 = Enable TFAILIF(SYS_HIRCTSTS[1]) status to trigger an interrupt to CPU.
[0]	**Reserved**	Reserved.

HIRC Trim Interrupt Status Register (SYS_HIRCTSTS)

Register	Offset	R/W	Description	Reset Value
HIRCTSTS	GCR_BA+0x98	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 reference clock or 48MHz 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_HIRCTCTL[1:0]) will be cleared to 00 by hardware automatically if CESTOPEN(SYS_HIRCTCTL[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_HIRCTCTL[1:0]) will be cleared to 00 by hardware automatically. If this bit is set and TFAILIEN(SYS_HIRCTIEN[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 Write Protection Register (REGWRPROT)

This register is write for disable/enable register protection and read for the REGPROTDIS status

Register	Offset	R/W	Description	Reset Value
REGWRPROT	GCR_BA+0x100	R/W	Register Write Protection Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[7:0]	**REGWRPROT**	**Register Write-Protection 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 REGPROTDIS bit will be set to 1 and write-protection registers can be normal write.
[0]	**REGPROTDIS**	**Register Write-Protection Disable Index (Read Only)** 0 = Write-protection is enabled for writing protected registers. Any write to the protected register is ignored. 1 = Write-protection is disabled for writing protected registers. The Protected registers are: **IPRSTC1**: address 0x5000_0008 **BODCR**: address 0x5000_0018 **PORCR**: address 0x5000_0024 **PWRCON**: address 0x5000_0200 (bit[6] is not protected for power wake-up interrupt clear) **APBCLK bit[0]**: address 0x5000_0208 (bit[0] is Watchdog Timer clock enable) **CLKSEL0**: address 0x5000_0210 (for HCLK and CPU STCLK clock source selection) **CLKSEL1 bit[1:0]**: address 0x5000_0214 (for Watchdog Timer clock source selection) **NMI_SEL bit[8]**: address 0x5000_0380 (for NMI_EN interrupt enable) **ISPCON**: address 0x5000_C000 (Flash ISP Control register) **ISPTRG**: address 0x5000_C010 (ISP Trigger Control register) **WTCR**: address 0x4000_4000 **FATCON**: address 0x5000_C018 **Note:** The bits which are write-protected will be noted as” **(Write Protect)**” beside the description.

6.2.9 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_CVR) to 0, and reload (wrap) to the value in the SysTick Reload Value Register (SYST_RVR) on the next clock cycle, then decrement on subsequent clocks. When the counter transitions to 0, the COUNTFLAG status bit is set. The COUNTFLAG bit clears on reads.

The SYST_CVR value is UNKNOWN on reset. Software should write to the register to clear it to 0 before enabling the feature. This ensures the timer will count from the SYST_RVR value rather than an arbitrary value when it is enabled.

If the SYST_RVR is 0, the timer will be maintained with a current value of 0 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”.

6.2.9.1 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_CSR	SCS_BA+0x10	R/W	SysTick Control and Status Register	0x0000_0000
SYST_RVR	SCS_BA+0x14	R/W	SysTick Reload Value Register	0xXXXX_XXXX
SYST_CVR	SCS_BA+0x18	R/W	SysTick Current Value Register	0xXXXX_XXXX

6.2.9.2 System Timer Control Register Description

SysTick Control and Status (SYST_CSR)

Register	Offset	R/W	Description	Reset Value
SYST_CSR	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**	**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** If CLKSRC(SYST_CSR[2]) = 1, SysTick clock source is from HCLK. If CLKSRC(SYST_CSR[2]) = 0, SysTick clock source is defined by STCLK_S(CLKSEL0[5:3]). 0 = Clock source is (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 0 has occurred. 1 = Counting down to 0 will cause the SysTick exception to be pended. Clearing the SysTick Current Value register by a 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_RVR)

Register	Offset	R/W	Description	Reset Value
SYST_RVR	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**	Value to load into the Current Value register when the counter reaches 0.

SysTick Current Value Register (SYST_CVR)

Register	Offset	R/W	Description	Reset Value
SYST_CVR	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.

6.2.10 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 kernel 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”.

6.2.10.1 Exception Model and System Interrupt Map

The following table lists the exception model supported by NuMicro® NUC029LEE/NUC029SEE. 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 6.2-2 Exception Model

Vector Number	Interrupt Number (Bit In Interrupt Registers)	Interrupt Name	Source Module	Interrupt Description
1 ~ 15	-	-	-	System exceptions
16	0	**BOD_INT**	Brown-out	Brown-out low voltage detected interrupt
17	1	**WDT_INT**	WDT	Watchdog Timer interrupt
18	2	**EINT0**	GPIO	External signal interrupt from PB.14 pin
19	3	**EINT1**	GPIO	External signal interrupt from PB.15 pin
20	4	**GPAB_INT**	GPIO	External signal interrupt from PA[6:0]/PA[15:8]/PB[11:0]/PB[15:13]
21	5	**GPCEF_INT**	GPIO	External interrupt from PC[3:0]/PC[11:6]/PC[15:14]/PE[5]/PF[1:0]
22	6	**PWMA_INT**	PWM0~3	PWM0, PWM1, PWM2 and PWM3 interrupt
23	7	**PWMB_INT**	PWM4~7	PWM4 and PWM5 interrupt
24	8	**TMR0_INT**	TMR0	Timer 0 interrupt
25	9	**TMR1_INT**	TMR1	Timer 1 interrupt
26	10	**TMR2_INT**	TMR2	Timer 2 interrupt
27	11	**TMR3_INT**	TMR3	Timer 3 interrupt
28	12	**UART02_INT**	UART0/2	UART0 and UART2 interrupt

29	13	**UART1_INT**	UART1	UART1 interrupt
30	14	**SPI0_INT**	SPI0	SPI0 interrupt
31	15	**SPI1_INT**	SPI1	SPI1 interrupt
32	16	**-**	-	Reserved
33	17	**-**	-	Reserved
34	18	**I2C0_INT**	I^2C0	I^2C0 interrupt
35	19	**I2C1_INT**	I^2C1	I^2C1 interrupt
36	20	**-**	-	Reserved
37	21	**-**	-	Reserved
38	22	**-**	-	Reserved
39	23	**USB_INT**	USBD	USB 2.0 FS Device interrupt
40	24	**-**	-	Reserved
41	25	**-**	-	Reserved
42	26	**PDMA_INT**	PDMA	PDMA interrupt
43	27	**-**	-	Reserved
44	28	**PWRWU_INT**	CLKC	Clock controller interrupt for chip wake-up from Power-down state
45	29	**ADC_INT**	ADC	ADC interrupt
46	30	**IRC_INT**	IRC	IRC TRIM interrupt
47	31	**RTC_INT**	RTC	Real Time Clock interrupt

Table 6.2-3 System Interrupt Map

6.2.10.2 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 6.2-4 Vector Table Format

6.2.10.3 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 activate. If an interrupt is Active when it is disabled, it remains in its Active state until cleared by reset or an exception return. Clearing the enable bit prevents new activations of the associated interrupt.

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

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

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

6.2.10.4 NVIC Control Register Map

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

Register	Offset	R/W	Description	Reset Value
NVIC Base Address: **SCS_BA = 0xE000_E000**				
NVIC_ISER	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-enable Control Register	0x0000_0000
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-enable Control Register	0x0000_0000
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ31 Set-pending Control Register	0x0000_0000
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ31 Clear-pending Control Register	0x0000_0000
NVIC_IPR0	SCS_BA+0x400	R/W	IRQ0 ~ IRQ3 Priority Control Register	0x0000_0000
NVIC_IPR1	SCS_BA+0x404	R/W	IRQ4 ~ IRQ7 Priority Control Register	0x0000_0000
NVIC_IPR2	SCS_BA+0x408	R/W	IRQ8 ~ IRQ11 Priority Control Register	0x0000_0000
NVIC_IPR3	SCS_BA+0x40C	R/W	IRQ12 ~ IRQ15 Priority Control Register	0x0000_0000
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ19 Priority Control Register	0x0000_0000
NVIC_IPR5	SCS_BA+0x414	R/W	IRQ20 ~ IRQ23 Priority Control Register	0x0000_0000
NVIC_IPR6	SCS_BA+0x418	R/W	IRQ24 ~ IRQ27 Priority Control Register	0x0000_0000
NVIC_IPR7	SCS_BA+0x41C	R/W	IRQ28 ~ IRQ31 Priority Control Register	0x0000_0000

6.2.10.5NVIC Control Register Description

IRQ0 ~ IRQ31 Set-Enable Control Register (NVIC_ISER)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER	SCS_BA+0x100	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 Enable Register** Enable one or more interrupts. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Write Operation: 0 = No effect. 1 = Write 1 to enable associated interrupt. Read Operation: 0 = Associated interrupt status is Disabled. 1 = Associated interrupt status is Enabled. Read value indicates the current enable status.

IRQ0 ~ IRQ31 Clear-Enable Control Register (NVIC_ICER)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-enable Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**CLRENA**	**Interrupt Disable Bits** Disable one or more interrupts. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Write Operation: 0 = No effect. 1 = Write 1 to disable associated interrupt. Read Operation: 0 = Associated interrupt status is Disabled. 1 = Associated interrupt status is Enabled. Read value indicates the current enable status.

IRQ0 ~ IRQ31 Set-Pending Control Register (NVIC_ISPR)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR	SCS_BA+0x200	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	**Set Interrupt Pending Register** Write Operation: 0 = No effect. 1 = Write 1 to set pending state. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Read Operation: 0 = Associated interrupt in not in pending status. 1 = Associated interrupt is in pending status. Read value indicates the current pending status.

IRQ0 ~ IRQ31 Clear-Pending Control Register (NVIC_ICPR)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**CLRPEND**	**Clear Interrupt Pending Register** Write Operation: 0 = No effect. 1 = Write 1 to clear pending state. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Read Operation: 0 = Associated interrupt in not in pending status. 1 = Associated interrupt is in pending status. Read value indicates the current pending status.

IRQ0 ~ IRQ3 Priority Register (NVIC_IPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR0	SCS_BA+0x400	R/W	IRQ0 ~ IRQ3 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_3		Reserved					
23	22	21	20	19	18	17	16
PRI_2		Reserved					
15	14	13	12	11	10	9	8
PRI_1		Reserved					
7	6	5	4	3	2	1	0
PRI_0		Reserved					

Bits	Description	
[31:30]	**PRI_3**	**Priority Of IRQ3** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_2**	**Priority Of IRQ2** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_1**	**Priority Of IRQ1** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_0**	**Priority Of IRQ0** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ4 ~ IRQ7 Priority Register (NVIC_IPR1)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR1	SCS_BA+0x404	R/W	IRQ4 ~ IRQ7 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_7		Reserved					
23	22	21	20	19	18	17	16
PRI_6		Reserved					
15	14	13	12	11	10	9	8
PRI_5		Reserved					
7	6	5	4	3	2	1	0
PRI_4		Reserved					

Bits	Description	
[31:30]	**PRI_7**	**Priority Of IRQ7** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_6**	**Priority Of IRQ6** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_5**	**Priority Of IRQ5** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_4**	**Priority Of IRQ4** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ8 ~ IRQ11 Priority Register (NVIC_IPR2)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR2	SCS_BA+0x408	R/W	IRQ8 ~ IRQ11 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11		Reserved					
23	22	21	20	19	18	17	16
PRI_10		Reserved					
15	14	13	12	11	10	9	8
PRI_9		Reserved					
7	6	5	4	3	2	1	0
PRI_8		Reserved					

Bits	Description	
[31:30]	**PRI_11**	**Priority Of IRQ11** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_10**	**Priority Of IRQ10** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_9**	**Priority Of IRQ9** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_8**	**Priority Of IRQ8** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ12 ~ IRQ15 Priority Register (NVIC_IPR3)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR3	SCS_BA+0x40C	R/W	IRQ12 ~ IRQ15 Priority Control Register	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
PRI_13		Reserved					
7	6	5	4	3	2	1	0
PRI_12		Reserved					

Bits	Description	
[31:30]	**PRI_15**	**Priority Of IRQ15** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_14**	**Priority Of IRQ14** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_13**	**Priority Of IRQ13** "0" denotes the highest priority and "3" denotes the lowest priority
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_12**	**Priority Of IRQ12** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ16 ~ IRQ19 Priority Register (NVIC_IPR4)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ19 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_19		Reserved					
23	22	21	20	19	18	17	16
PRI_18		Reserved					
15	14	13	12	11	10	9	8
PRI_17		Reserved					
7	6	5	4	3	2	1	0
PRI_16		Reserved					

Bits	Description	
[31:30]	**PRI_19**	**Priority Of IRQ19** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_18**	**Priority Of IRQ18** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_17**	**Priority Of IRQ17** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_16**	**Priority Of IRQ16** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ20 ~ IRQ23 Priority Register (NVIC_IPR5)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR5	SCS_BA+0x414	R/W	IRQ20 ~ IRQ23 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_23		Reserved					
23	22	21	20	19	18	17	16
PRI_22		Reserved					
15	14	13	12	11	10	9	8
PRI_21		Reserved					
7	6	5	4	3	2	1	0
PRI_20		Reserved					

Bits	Description	
[31:30]	**PRI_23**	**Priority Of IRQ23** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_22**	**Priority Of IRQ22** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_21**	**Priority Of IRQ21** "0" denotes the highest priority and "3" denotes the lowest priority
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_20**	**Priority Of IRQ20** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ24 ~ IRQ27 Priority Register (NVIC_IPR6)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR6	SCS_BA+0x418	R/W	IRQ24 ~ IRQ27 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_27		Reserved					
23	22	21	20	19	18	17	16
PRI_26		Reserved					
15	14	13	12	11	10	9	8
PRI_25		Reserved					
7	6	5	4	3	2	1	0
PRI_24		Reserved					

Bits	Description	
[31:30]	**PRI_27**	**Priority Of IRQ27** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_26**	**Priority Of IRQ26** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_25**	**Priority Of IRQ25** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_24**	**Priority Of IRQ24** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ28 ~ IRQ31 Priority Register (NVIC_IPR7)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR7	SCS_BA+0x41C	R/W	IRQ28 ~ IRQ31 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_31		Reserved					
23	22	21	20	19	18	17	16
PRI_30		Reserved					
15	14	13	12	11	10	9	8
PRI_29		Reserved					
7	6	5	4	3	2	1	0
PRI_28		Reserved					

Bits	Description	
[31:30]	**PRI_31**	**Priority Of IRQ31** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_30**	**Priority Of IRQ30** "0" denotes the highest priority and "3" denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_29**	**Priority Of IRQ29** "0" denotes the highest priority and "3" denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_28**	**Priority Of IRQ28** "0" denotes the highest priority and "3" denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

6.2.10.6 Interrupt Source Register Map

Besides the interrupt control registers associated with the NVIC, the NuMicro® NUC029LEE/NUC029SEE also implement some specific control registers to facilitate the interrupt functions, including “interrupt source identification”, ”NMI source selection” and “interrupt test mode”, which are described below.

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

Register	Offset	R/W	Description	Reset Value
INT Base Address: **INT_BA = 0x5000_0300**				
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) Interrupt Source Identity	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) Interrupt Source Identity	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) Interrupt Source Identity	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) Interrupt Source Identity	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/B) Interrupt Source Identity	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (GPC/E/F) Interrupt Source Identity	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWMA) Interrupt Source Identity	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (PWMB) Interrupt Source Identity	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) Interrupt Source Identity	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) Interrupt Source Identity	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) Interrupt Source Identity	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) Interrupt Source Identity	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0/2) Interrupt Source Identity	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (UART1) Interrupt Source Identity	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) Interrupt Source Identity	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (SPI1) Interrupt Source Identity	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	Reserved	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	Reserved	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I^2C0) Interrupt Source Identity	0xXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I^2C1) Interrupt Source Identity	0xXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	R	Reserved	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	R	Reserved	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	Reserved	0xXXXX_XXXX

IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USBD) Interrupt Source Identity	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	R	Reserved	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	Reserved	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (PDMA) Interrupt Source Identity	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	Reserved	0xXXXX_XXXX
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) Interrupt Source Identity	0xXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) Interrupt Source Identity	0xXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	R	IRQ30 (IRC) Interrupt Source Identity	0xXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	R	IRQ31 (RTC) Interrupt Source Identity	0xXXXX_XXXX
NMI_SEL	INT_BA+0x80	R/W	NMI Source Interrupt Select Control Register	0x0000_0000
MCU_IRQ	INT_BA+0x84	R/W	MCU Interrupt Request Source Register	0x0000_0000
MCU_IRQCR	INT_BA+0x88	R/W	MCU Interrupt Request Control Register	0x0000_0000

6.2.10.7 Interrupt Source Register Description

Interrupt Source Identity Register (IRQn_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) Interrupt Source Identity	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) Interrupt Source Identity	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) Interrupt Source Identity	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) Interrupt Source Identity	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/B) Interrupt Source Identity	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (GPC/E/F) Interrupt Source Identity	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWMA) Interrupt Source Identity	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (PWMB) Interrupt Source Identity	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) Interrupt Source Identity	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) Interrupt Source Identity	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) Interrupt Source Identity	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) Interrupt Source Identity	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0/2) Interrupt Source Identity	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (UART1) Interrupt Source Identity	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) Interrupt Source Identity	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (SPI1) Interrupt Source Identity	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	Reserved	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	Reserved	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I^2C0) Interrupt Source Identity	0xXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I^2C1) Interrupt Source Identity	0xXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	R	Reserved	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	R	Reserved	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	Reserved	0xXXXX_XXXX
IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USBD) Interrupt Source Identity	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	R	Reserved	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	Reserved	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (PDMA) Interrupt Source Identity	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	Reserved	0xXXXX_XXXX

IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) Interrupt Source Identity	0xXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) Interrupt Source Identity	0xXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	R	IRQ30 (IRC) Interrupt Source Identity	0xXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	R	IRQ31 (RTC) Interrupt Source Identity	0xXXXX_XXXX

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

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**INT_SRC**	**Interrupt Source** Define the interrupt sources for interrupt event.

Bits	Address	INT-Num	Description
[2:0]	INT_BA+0x00	0	Bit2: 0 Bit1: 0 Bit0: BOD_INT
[2:0]	INT_BA+0x04	1	Bit2: 0 Bit1: WWDT_INT Bit0: WDT_INT
[2:0]	INT_BA+0x08	2	Bit2: 0 Bit1: 0 Bit0: EINT0 – external interrupt 0 from PB.14
[2:0]	INT_BA+0x0C	3	Bit2: 0 Bit1: 0 Bit0: EINT1 – external interrupt 1 from PB.15
[2:0]	INT_BA+0x10	4	Bit2: 0 Bit1: GPB_INT Bit0: GPA_INT
[3:0]	INT_BA+0x14	5	Bit3: GPF_INT Bit2: GPE_INT Bit1: 0

			Bit0: GPC_INT
[3:0]	INT_BA+0x18	6	Bit3: PWM3_INT Bit2: PWM2_INT Bit1: PWM1_INT Bit0: PWM0_INT
[3:0]	INT_BA+0x1C	7	Bit3: 0 Bit2: 0 Bit1: PWM5_INT Bit0: PWM4_INT
[2:0]	INT_BA+0x20	8	Bit2: 0 Bit1: 0 Bit0: TMR0_INT
[2:0]	INT_BA+0x24	9	Bit2: 0 Bit1: 0 Bit0: TMR1_INT
[2:0]	INT_BA+0x28	10	Bit2: 0 Bit1: 0 Bit0: TMR2_INT
[2:0]	INT_BA+0x2C	11	Bit2: 0 Bit1: 0 Bit0: TMR3_INT
[2:0]	INT_BA+0x30	12	Bit2: 0 Bit1: UART2_INT Bit0: UART0_INT
[2:0]	INT_BA+0x34	13	Bit2: 0 Bit1: 0 Bit0: UART1_INT
[2:0]	INT_BA+0x38	14	Bit2: 0 Bit1: 0 Bit0: SPI0_INT
[2:0]	INT_BA+0x3C	15	Bit2: 0 Bit1: 0 Bit0: SPI1_INT
[2:0]	INT_BA+0x48	18	Bit2: 0 Bit1: 0 Bit0: I2C0_INT
[2:0]	INT_BA+0x4C	19	Bit2: 0 Bit1: 0 Bit0: I2C1_INT
[2:0]	INT_BA+0x5C	23	Bit2: 0 Bit1: 0 Bit0: USB_INT
[2:0]	INT_BA+0x68	26	Bit2: 0 Bit1: 0

			Bit0: PDMA_INT
[2:0]	INT_BA+0x70	28	Bit2: 0 Bit1: 0 Bit0: PWRWU_INT
[2:0]	INT_BA+0x74	29	Bit2: 0 Bit1: 0 Bit0: ADC_INT
[2:0]	INT_BA+0x78	30	Bit2: 0 Bit1: 0 Bit0: IRC_INT
[2:0]	INT_BA+0x7C	31	Bit2: 0 Bit1: 0 Bit0: RTC_INT

NMI Source Interrupt Select Control Register (NMI_SEL)

Register	Offset	R/W	Description	Reset Value
NMI_SEL	INT_BA+0x80	R/W	NMI Source Interrupt 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							**NMI_EN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**NMI_SEL**				

Bits	Description	
[31:8]	**Reserved**	Reserved.
[8]	**NMI_EN**	**NMI Interrupt Enable Bit (Write Protect)** 0 = NMI interrupt Disabled. 1 = NMI interrupt Enabled. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[7:5]	**Reserved**	Reserved.
[4:0]	**NMI_SEL**	**NMI Interrupt Source Selection** The NMI interrupt to Cortex®-M0 can be selected from one of the peripheral interrupt by setting NMI_SEL.

MCU Interrupt Request Source Register (MCU_IRQ)

Register	Offset	R/W	Description	Reset Value
MCU_IRQ	INT_BA+0x84	R/W	MCU Interrupt Request Source Register	0x0000_0000

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

Bits	Description	
[31:0]	**MCU_IRQ**	**MCU IRQ Source Register** The MCU_IRQ collects all the interrupts from the peripherals and generates the synchronous interrupt to Cortex®-M0. There are two modes to generate interrupt to Cortex®-M0, the normal mode and test mode. The MCU_IRQ collects all interrupts from each peripheral and synchronizes them and interrupts the Cortex®-M0. When the MCU_IRQ[n] is 0: Set MCU_IRQ[n] 1 will generate an interrupt to Cortex®-M0 NVIC[n]. When the MCU_IRQ[n] is 1 (mean an interrupt is assert), setting 1 to the MCU_IRQ[n] 1 will clear the interrupt and setting MCU_IRQ[n] 0: has no effect

MCU Interrupt Request Control Register (MCU_IRQCR)

Register	Offset	R/W	Description	Reset Value
MCU_IRQCR	INT_BA+0x88	R/W	MCU Interrupt Request 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							**FAST_IRQ**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**FAST_IRQ**	**Fast IRQ Latency Enable Bit** 0 = MCU IRQ latency is fixed at 13 clock cycles of HCLK, MCU will enter IRQ handler after this fixed latency when interrupt happened. 1 = MCU IRQ latency will not fixed, MCU will enter IRQ handler as soon as possible when interrupt happened.

6.2.11 System Control

The Cortex®-M0 status and operating 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”.

6.2.11.1 System Control Register Map

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

Register	Offset	R/W	Description	Reset Value
SCS Base Address: **SCS_BA = 0xE000_E000**				
CPUID	SCS_BA+0xD00	R	CPUID Register	0x410C_C200
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

6.2.11.2 System Control Register Description

CPUID Register (CPUID)

Register	Offset	R/W	Description	Reset Value
CPUID	SCS_BA+0xD00	R	CPUID Register	0x410C_C200

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

Bits	Description	
[31:24]	**IMPLEMENTER**	**Implementer Code Assigned By ARM** Implementer code assigned by ARM. (ARM = 0x41).
[23:20]	**Reserved**	Reserved.
[19:16]	**PART**	**Architecture Of The Processor** Read as 0xC for ARMv6-M parts
[15:4]	**PARTNO**	**Part Number Of The Processor** Read as 0xC20.
[3:0]	**REVISION**	**Revision Number** Read as 0x0

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				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 not pending. 1 = NMI exception pending. 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. This is a write only bit. When you want to clear PENDSV bit, you must “write 0 to PENDSVSET and write 1 to PENDSVCLR” 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. This is a write only bit. When you want to clear PENDST bit, you must “write 0 to PENDSTSET and write 1 to PENDSTCLR” at the same time.
[24]	**Reserved**	Reserved.
[23]	**ISRPREEMPT**	**If Set, A Pending Exception Will Be Serviced On Exit From The Debug Halt State** This bit is read only.
[22]	**ISRPENDING**	**Interrupt Pending Flag, Excluding NMI And Faults:** 0 = Interrupt not pending. 1 = Interrupt pending. This bit is read only.
[21:18]	**Reserved**	Reserved.
[17:12]	**VECTPENDING**	**Indicates The Exception Number Of The Highest Priority Pending Enabled Exception:** 0 = No pending exceptions. Non-zero = Exception number of the highest priority pending enabled exception.
[11:6]	**Reserved**	Reserved.
[5:0]	**VECTACTIVE**	**Contains The Active Exception Number** 0 = Thread mode. Non-zero = 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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**SYSRESETREQ**	**VECTCLKACTIVE**	**Reserved**

Bits	Description	
[31:16]	**VECTORKEY**	**Register Access Key** Write Operation: When writing to this register, the VECTORKEY field need to be set to 0x05FA, otherwise the write operation would be ignored. The VECTORKEY filed is used to prevent accidental write to this register from resetting the system or clearing of the exception status. Read Operation: Read as 0xFA05.
[15:3]	**Reserved**	Reserved.
[2]	**SYSRESETREQ**	**System Reset Request** Writing this bit 1 will cause a reset signal to be asserted to the chip to indicate a reset is requested. The bit is a write only bit and self-clears as part of the reset sequence.
[1]	**VECTCLRACTIVE**	**Exception Active Status Clear Bit** Reserved for debug use. When writing to the register, user must write 0 to this bit, otherwise behavior is unpredictable.
[0]	**Reserved**	Reserved.

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 Bit** 0 = Only enabled interrupts or events can wake-up the processor, 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** Controls whether the processor uses sleep or deep sleep as its low power mode: 0 = Sleep mode. 1 = Deep Sleep mode.
[1]	**SLEEPONEXIT**	**Sleep-On-Exit Enable Bit** This bit indicates 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 when returning from ISR to Thread mode. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application..
[0]	**Reserved**	Reserved.

System Handler Priority Register 2 (SHPR2)

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

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

Bits	Description	
[31:30]	**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
[21:0]	**Reserved**	Reserved.

6.3 Clock Controller

6.3.1 Overview

The clock controller generates the 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 clock divider. The chip enters Power-down mode when Cortex®-M0 core executes the WFI instruction only if the PWR_DOWN_EN (PWRCON[7]) bit and PD_WAIT_CPU (PWRCON[8]) bit are both set to 1. After that, chip enters Power-down mode and wait for wake-up interrupt source triggered to leave Power-down mode. In the Power-down mode, the clock controller turns off the 4~24 MHz external high speed crystal oscillator and 22.1184/48 MHz internal high speed RC oscillator to reduce the overall system power consumption. The following figures show the clock generator and the overview of the clock source control.

The clock generator consists of 6 clock sources as listed below:

- 32.768 kHz external low speed crystal oscillator (LXT)
- 4~24 MHz external high speed crystal oscillator (HXT)
- Programmable PLL output clock frequency (PLL source can be selected from external 4~24 MHz external high speed crystal oscillator (HXT) or 22.1184 MHz internal high speed RC oscillator (HIRC)) (PLL FOUT)
- 22.1184 MHz internal high speed RC oscillator (HIRC)
- 48 MHz internal high speed RC oscillator (HIRC48)
- 10 kHz internal low speed RC oscillator (LIRC)

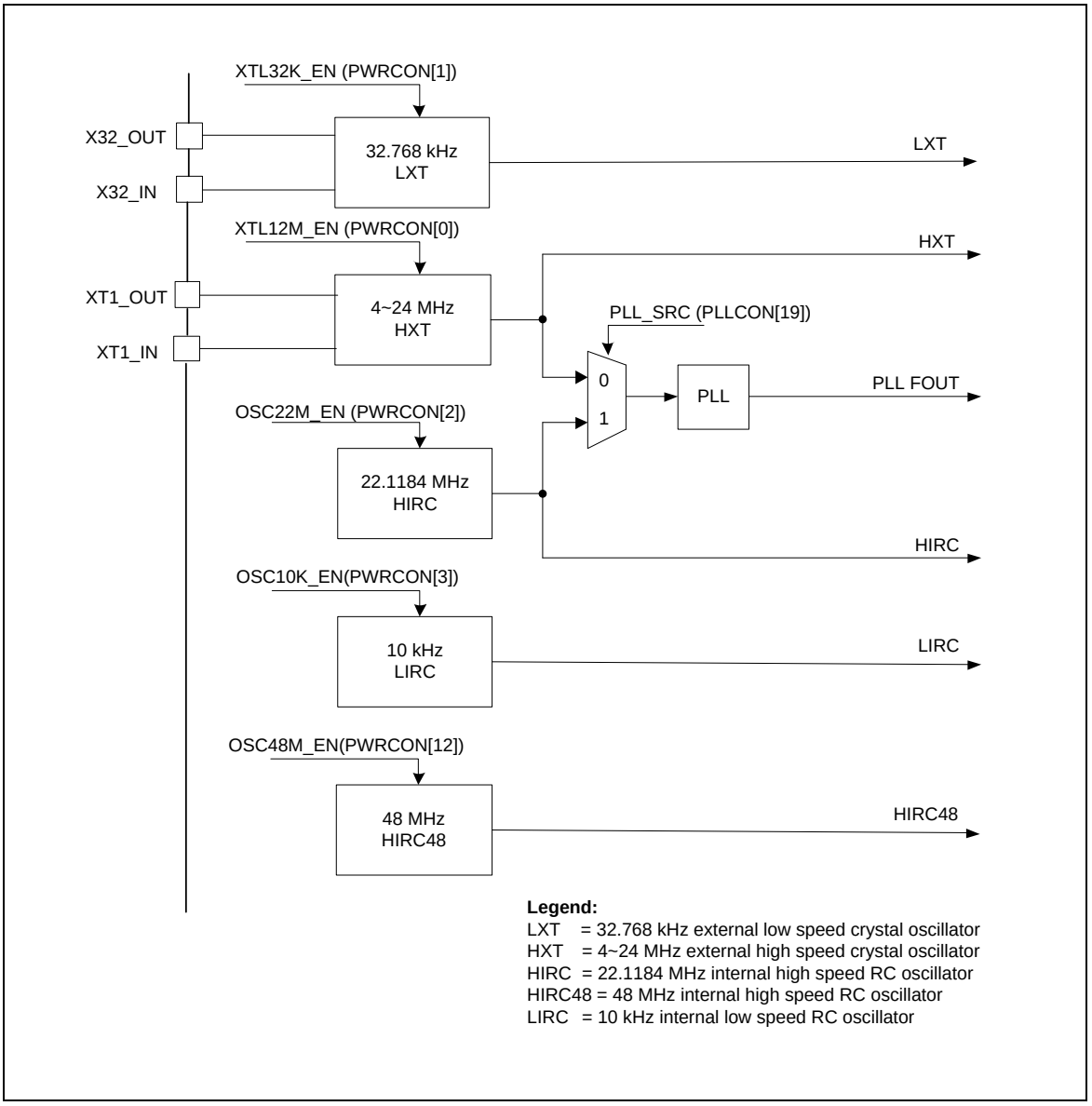


Figure 6.3-1 Clock Generator Block Diagram

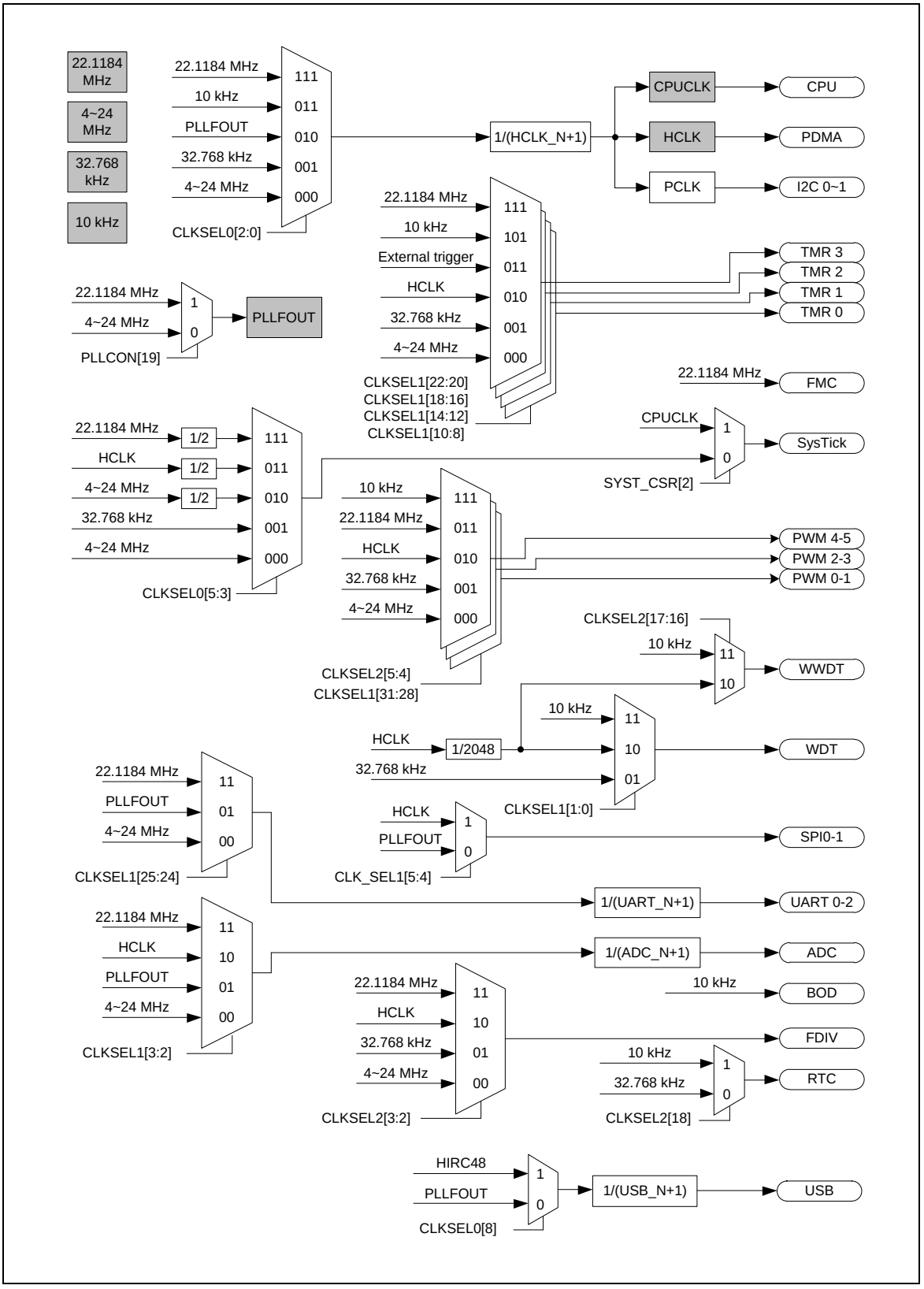
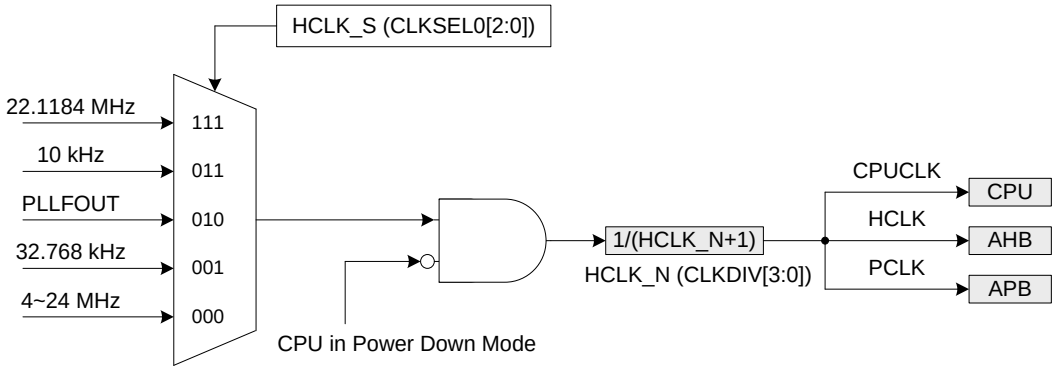


Figure 6.3-2 Clock Generator Global View Diagram

6.3.2 System Clock and SysTick Clock

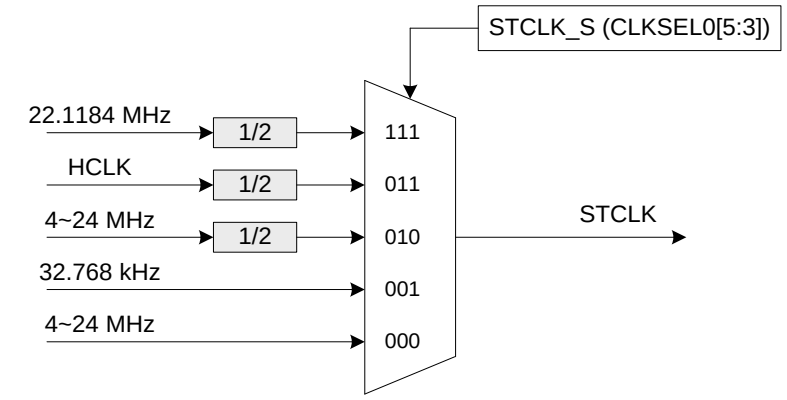
The system clock has 5 clock sources which were generated from clock generator block. The clock source switch depends on the register HCLK_S (CLKSEL0[2:0]). The block diagram is shown in Figure 6.3-3.



Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 6.3-3 System Clock Block Diagram

The clock source of SysTick in Cortex®-M0 core can use CPU clock or external clock (SYST_CSR[2]). If using external clock, the SysTick clock (STCLK) has 5 clock sources. The clock source switch depends on the setting of the register STCLK_S (CLKSEL0[5:3]). The block diagram is shown in Figure 6.3-4.



Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 6.3-4 SysTick Clock Control Block Diagram

6.3.3 Power-down Mode Clock

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

The clocks still kept active are listed below:

- Clock Generator
  - 10 kHz internal low speed RC oscillator clock
  - 32.768 kHz external low speed crystal oscillator clock
- RTC/WDT/Timer/PWM Peripherals Clock (when 32.768 kHz external low speed crystal oscillator or 10 kHz intertnal low speed RC oscillator is adopted as clock source)

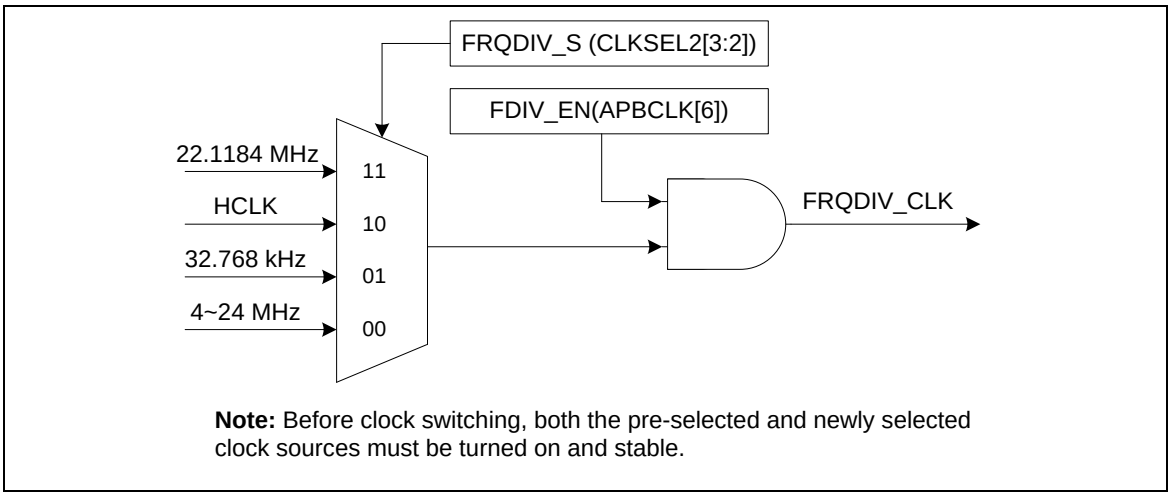
6.3.4 Frequency Divider Output

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 Fin 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 FSEL (FRQDIV[3:0]).

When writing 1 to DIVIDER_EN (FRQDIV[4]), the chained counter starts to count. When writing 0 to DIVIDER_EN (FRQDIV[4]), the chained counter continuously runs till divided clock reaches low state and stay in low state.

If DIVIDER1(FRQDIV[5]) is set to 1, the frequency divider clock (FRQDIV_CLK) will bypass power-of-2 frequency divider. The frequency divider clock will be output to CLKO pin directly.



Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 6.3-5 Clock Source of Frequency Divider

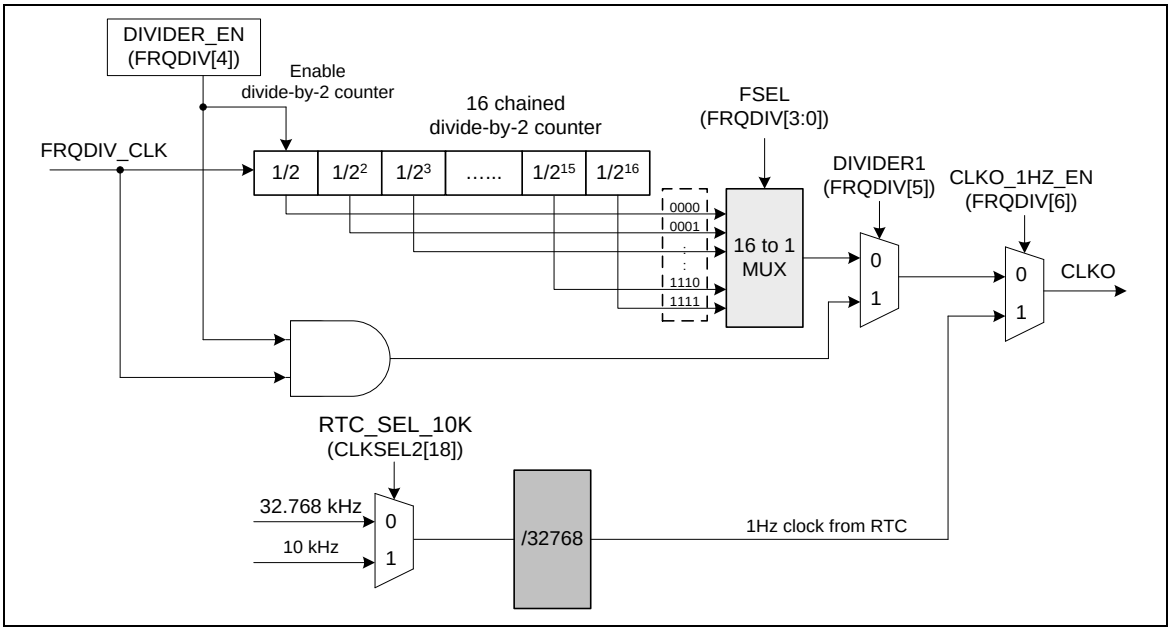


Figure 6.3-6 Frequency Divider Block Diagram

6.3.5 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 = 0x5000_0200**				
PWRCON	CLK_BA+0x00	R/W	System Power-down Control Register	0x0000_001X
AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_0005
APBCLK	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register	0x0000_000X
CLKSTATUS	CLK_BA+0x0C	R/W	Clock Status Monitor Register	0x0000_00XX
CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0000_003X
CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0xFFFF_FFFF
CLKSEL2	CLK_BA+0x1C	R/W	Clock Source Select Control Register 2	0x0002_00FF
CLKDIV	CLK_BA+0x18	R/W	Clock Divider Number Register	0x0000_0000
PLLCON	CLK_BA+0x20	R/W	PLL Control Register	0x0005_C22E
FRQDIV	CLK_BA+0x24	R/W	Frequency Divider Control Register	0x0000_0000

6.3.6 Register Description

System Power-down Control Register (PWRCON)

Except the BIT[6], all the other bits are protected, programming these bits need to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100

Register	Offset	R/W	Description	Reset Value
PWRCON	CLK_BA+0x00	R/W	System Power-down Control Register	0x0000_001X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			OSC48M_EN	Reserved			PD_WAIT_CPU
7	6	5	4	3	2	1	0
PWR_DOWN_EN	PD_WU_STS	PD_WU_INT_EN	PD_WU_DLY	OSC10K_EN	OSC22M_EN	XTL32K_EN	XTL12M_EN

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**OSC48M_EN**	**48 MHz Internal High Speed RC Oscillator (HIRC48) Enable Bit (Write Protect)** 0 = 48 MHz internal high speed RC oscillator (HIRC48) Disabled. 1 = 48 MHz internal high speed RC oscillator (HIRC48) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[11:9]	**Reserved**	Reserved.
[8]	**PD_WAIT_CPU**	**Power-Down Entry Condition Control (Write Protect)** 0 = Chip enters Power-down mode when the PWR_DOWN_EN bit is set to 1. 1 = Chip enters Power- down mode when the both PD_WAIT_CPU and PWR_DOWN_EN bits are set to 1 and CPU run WFI instruction. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[7]	**PWR_DOWN_EN**	**System Power-Down Enable Bit (Write Protect)** When this bit is set to 1, Power-down mode is enabled and chip Power-down behavior will depends on the PD_WAIT_CPU bit (a) If the PD_WAIT_CPU is 0, the chip enters Power-down mode immediately after the PWR_DOWN_EN bit set. (b) if the PD_WAIT_CPU is 1, the chip keeps active till the CPU sleep mode is also active and then the chip enters Power-down mode (recommend) When chip wakes up from Power-down mode, this bit is cleared by hardware. User needs to set this bit again for next Power-down. In Power-down mode, 4~24 MHz external high speed crystal oscillator and the 22.1184

		MHz internal high speed RC oscillator will be disabled in this mode, but the 32.768 kHz external low speed crystal oscillator and 10 kHz internal low speed oscillator are not controlled by Power-down mode. In Power- down mode, the PLL and system clock are disabled, and ignored the clock source selection. The clocks of peripheral are not controlled by Power-down mode, if the peripheral clock source is from 32.768 kHz external low speed crystal oscillator or the internal 10 kHz low speed oscillator. 0 = Chip operating normally or chip in Idle mode because of WFI command. 1 = Chip enters Power-down mode instantly or waits CPU sleep command WFI. **Note:** This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[6]	**PD_WU_STS**	**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 the GPIO, USB, UART, WDT, I^2C, TIMER, BOD or RTC wake-up occurred Write 1 to clear the bit to 0. **Note:** This bit is working only if PD_WU_INT_EN (PWRCON[5]) set to 1.
[5]	**PD_WU_INT_EN**	**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. **Note1:** The interrupt will occur when both PD_WU_STS and PD_WU_INT_EN are high. **Note2:** This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[4]	**PD_WU_DLY**	**Wake-Up Delay Counter Enable Bit (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 work at external 4~24 MHz high speed crystal, and 256 clock cycles when chip work at internal 22.1184 MHz high speed oscillator. 0 = Clock cycles delay Disabled. 1 = Clock cycles delay Enabled. **Note:** This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[3]	**OSC10K_EN**	**10 KHz Internal Low Speed RC Oscillator (LIRC) Enable Bit (Write Protect)** 0 = 10 kHz internal low speed RC oscillator (LIRC) Disabled. 1 = 10 kHz internal low speed RC oscillator (LIRC) Enabled. **Note:** This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[2]	**OSC22M_EN**	**22.1184 MHz Internal High Speed RC Oscillator (HIRC) Enable Bit (Write Protect)** 0 = 22.1184 MHz internal high speed RC oscillator (HIRC) Disabled. 1 = 22.1184 MHz internal high speed RC oscillator (HIRC) Enabled. **Note:** This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[1]	**XTL32K_EN**	**32.768 KHz External Low Speed Crystal Oscillator (LXT) Enable Bit (Write Protect)** 0 = 32.768 kHz external low speed crystal oscillator (LXT) Disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) Enabled (Normal operation). **Note:** This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register

		REGWRPROT at address GCR_BA+0x100.
[0]	**XTL12M_EN**	**4~24 MHz External High Speed Crystal Oscillator (HXT) Enable Bit (Write Protect)** The bit default value is set by flash controller user configuration register CONFIG0 [26:24]. When the default clock source is from 4~24 MHz external high speed crystal oscillator, this bit is set to 1 automatically. 0 = 4 ~ 24 MHz external high speed crystal oscillators (HXT) Disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT) Enabled. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Register Or Instruction Mode	SLEEPDEEP (SCR[2])	PD_WAIT_CPU (PWRCON[8])	PWR_DOWN_EN (PWRCON[7])	CPU Run WFI Instruction	Clock Disable
Normal operation	0	0	0	NO	All clocks disabled by control register
Idle mode (CPU entering Sleep mode)	0	x	0	YES	Only CPU clock disabled
Power-down mode (CPU entering Deep Sleep mode)	1	1	1	YES	Most clocks are disabled except 10 kHz and 32.768 kHz, only RTC/WDT/Timer/PWM peripheral clock still enable if their clock source are selected as 10 kHz or 32.768 kHz.

Table 6.3-1 Chip Idle/Power-down Mode Control Table

When chip enters Power-down mode, user can wake-up chip using some interrupt sources. The related interrupt sources and NVIC IRQ enable bits (NVIC_ISER) should be enabled before setting the PWR_DOWN_EN bit in PWRCON[7] to ensure chip can enter Power-down and wake-up successfully.

AHB Devices Clock Enable Control Register (AHBCLK)

These bits for this register are used to enable/disable clock for system clock PDMA clock.

Register	Offset	R/W	Description	Reset Value
AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_0005

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

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**EBI_EN**	**EBI Controller Clock Enable Control** 1 = EBI engine clock Enabled. 0 = EBI engine clock Disabled.
[2]	**ISP_EN**	**Flash ISP Controller Clock Enable Bit** 0 = Flash ISP peripherial clock Disabled. 1 = Flash ISP peripherial clock Enabled.
[1]	**PDMA_EN**	**PDMA Controller Clock Enable Bit** 0 = PDMA peripherial clock Disabled. 1 = PDMA peripherial clock Enabled.
[0]	**Reserved**	Reserved.

APB Devices Clock Enable Register (APBCLK)

These bits of this register are used to enable/disable clock for peripheral controller clocks.

Register	Offset	R/W	Description	Reset Value
APBCLK	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register	0x0000_000X

31	30	29	28	27	26	25	24
Reserved			ADC_EN	USBD_EN	Reserved		
23	22	21	20	19	18	17	16
Reserved	PWM45_EN	PWM23_EN	PWM01_EN	Reserved	UART2_EN	UART1_EN	UART0_EN
15	14	13	12	11	10	9	8
Reserved		SPI1_EN	SPI0_EN	Reserved		I2C1_EN	I2C0_EN
7	6	5	4	3	2	1	0
Reserved	FDIV_EN	TMR3_EN	TMR2_EN	TMR1_EN	TMR0_EN	RTC_EN	WDT_EN

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**ADC_EN**	**Analog-Digital-Converter (ADC) Clock Enable Bit** 0 = ADC clock Disabled. 1 = ADC clock Enabled.
[27]	**USBD_EN**	**USB 2.0 FS Device Controller Clock Enable Bit** 0 = USB clock Disabled. 1 = USB clock Enabled.
[26:23]	**Reserved**	Reserved.
[22]	**PWM45_EN**	**PWM_45 Clock Enable Bit** 0 = PWM45 clock Disabled. 1 = PWM45 clock Enabled.
[21]	**PWM23_EN**	**PWM_23 Clock Enable Bit** 0 = PWM23 clock Disabled. 1 = PWM23 clock Enabled.
[20]	**PWM01_EN**	**PWM_01 Clock Enable Bit** 0 = PWM01 clock Disabled. 1 = PWM01 clock Enabled.
[19]	**Reserved**	Reserved.
[18]	**UART2_EN**	**UART2 Clock Enable Bit** 0 = UART2 clock Disabled. 1 = UART2 clock Enabled.
[17]	**UART1_EN**	**UART1 Clock Enable Bit** 0 = UART1 clock Disabled. 1 = UART1 clock Enabled.

[16]	**UART0_EN**	**UART0 Clock Enable Bit** 0 = UART0 clock Disabled. 1 = UART0 clock Enabled.
[15:14]	**Reserved**	Reserved.
[13]	**SPI1_EN**	**SPI1 Clock Enable Bit** 0 = SPI1 clock Disabled. 1 = SPI1 clock Enabled.
[12]	**SPI0_EN**	**SPI0 Clock Enable Bit** 0 = SPI0 clock Disabled. 1 = SPI0 clock Enabled.
[11:10]	**Reserved**	Reserved.
[9]	**I2C1_EN**	**I^2C1 Clock Enable Bit** 0 = I^2C1 clock Disabled. 1 = I^2C1 clock Enabled.
[8]	**I2C0_EN**	**I^2C0 Clock Enable Bit** 0 = I^2C0 clock Disabled. 1 = I^2C0 clock Enabled.
[7]	**Reserved**	Reserved.
[6]	**FDIV_EN**	**Frequency Divider Output Clock Enable Bit** 0 = FDIV clock Disabled. 1 = FDIV clock Enabled.
[5]	**TMR3_EN**	**Timer3 Clock Enable Bit** 0 = Timer3 clock Disabled. 1 = Timer3 clock Enabled.
[4]	**TMR2_EN**	**Timer2 Clock Enable Bit** 0 = Timer2 clock Disabled. 1 = Timer2 clock Enabled.
[3]	**TMR1_EN**	**Timer1 Clock Enable Bit** 0 = Timer1 clock Disabled. 1 = Timer1 clock Enabled.
[2]	**TMR0_EN**	**Timer0 Clock Enable Bit** 0 = Timer0 clock Disabled. 1 = Timer0 clock Enabled.
[1]	**RTC_EN**	**Real-Time-Clock APB Interface Clock Enable Bit** This bit is used to control the RTC APB clock only, The RTC peripheral clock source is selected from RTC_SEL_10K(CLKSEL2[18]). It can be selected to the 32.768 kHz external low speed crystal oscillator or 10 kHz internal low speed RC oscillator. 0 = RTC clock Disabled. 1 = RTC clock Enabled.
[0]	**WDT_EN**	**Watchdog Timer Clock Enable Bit (Write Protect)** 0 = Watchdog Timer clock Disabled. 1 = Watchdog Timer clock Enabled. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Clock status Register (CLKSTATUS)

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

Register	Offset	R/W	Description	Reset Value
CLKSTATUS	CLK_BA+0x0C	R/W	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
CLK_SW_FAIL	Reserved	OSC48M_STB	OSC22M_STB	OSC10K_STB	PLL_STB	XTL32K_STB	XTL12M_STB

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**CLK_SW_FAIL**	**Clock Switching Fail Flag (Read Only)** 0 = Clock switching success. 1 = Clock switching failure. This bit is an index that if current system clock source is match as user defined at HCLK_S (CLKSEL[2:0]). When user switch system clock, the system clock source will keep old clock until the new clock is stable. During the period that waiting new clock stable, this bit will be an index shows system clock source is not match as user wanted.
[6]	**Reserved**	Reserved.
[5]	**OSC48M_STB**	**48 MHz Internal High Speed RC Oscillator (HIRC48) Clock Source Stable Flag (Read Only)** 0 = 48MHz internal high speed RC oscillator (HIRC48) clock is not stable or disabled. 1 = 48MHz internal high speed RC oscillator (HIRC48) clock is stable and enabled.
[4]	**OSC22M_STB**	**22.1184 MHz Internal High Speed RC Oscillator (HIRC) Clock Source Stable Flag (Read Only)** 0 = 22.1184 MHz internal high speed RC oscillator (HIRC) clock is not stable or disabled. 1 = 22.1184 MHz internal high speed RC oscillator (HIRC) clock is stable and enabled.
[3]	**OSC10K_STB**	**Internal 10 KHz Low Speed Oscillator (LIRC) Clock Source Stable Flag (Read Only)** 0 = 10 kHz internal low speed RC oscillator (LIRC) clock is not stable or disabled. 1 = 10 kHz internal low speed RC oscillator (LIRC) clock is stable and enabled.
[2]	**PLL_STB**	**Internal PLL Clock Source Stable Flag (Read Only)** 0 = Internal PLL clock is not stable or disabled. 1 = Internal PLL clock is stable in normal mode.
[1]	**XTL32K_STB**	**32.768 KHz External Low Speed Crystallator Oscillator (LXT) Clock Source Stable**

		Flag (Read Only) 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock is not stable or disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) clock is stable and enabled.
[0]	**XTL12M_STB**	**4~24 MHz External High Speed Crystal Oscillator (HXT) Clock Source Stable Flag (Read Only)** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock is not stable or disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock is stable and enabled.

Clock Source Select Control Register 0 (CLKSEL0)

Register	Offset	R/W	Description	Reset Value
CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0000_003X

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**USB_S**	**USB Clock Source Selection** 0 = Clock source from PLL clock. 1 = Clock source from 48 MHz high speed RC oscillator clock.
[7:6]	**Reserved**	Reserved.
[5:3]	**STCLK_S**	**Cortex®-M0 SysTick Clock Source Select (Write Protect)** If CLKSRC(SYST_CSR[2]) = 1, SysTick clock source is from HCLK. If CLKSRC(SYST_CSR[2]) = 0, SysTick clock source is defined by STCLK_S(CLKSEL0[5:3]). 000 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 001 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 010 = Clock source from 4~24 MHz external high speed crystal oscillator clock/2. 011 = Clock source from HCLK/2. 111 = Clock source from 22.1184 MHz internal high speed RC oscillator clock/2. **Note1:** These bits are protected bit. It means programming this bit needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100. **Note2:** if SysTick clock source is not from HCLK (i.e. SYST_CSR[2] = 0), SysTick clock source must less than or equal to HCLK/2.
[2:0]	**HCLK_S**	**HCLK Clock Source Select (Write Protect)** 1. Before clock switching, the related clock sources (both pre-select and new-select) must be enabled 2. The 3-bit default value is reloaded from the value of CFOSC (CONFIG0[26:24]) in user configuration register of Flash controller by any reset. Therefore the default value is either 000b or 111b. 3. These bits are protected bit, It means programming this bit needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100. 000 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 001 = Clock source from 32.768 kHz external low speed crystal oscillator clock.

		010 = Clock source from PLL clock. 011 = Clock source from 10 kHz internal low speed RC oscillator clock. 111 = Clock source from 22.1184 MHz internalhigh speed RC oscillator clock. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Clock Source Select Control Register 1(CLKSEL1)

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

Register	Offset	R/W	Description	Reset Value
CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0xFFFF_FFFF

31	30	29	28	27	26	25	24
PWM23_S		PWM01_S		Reserved		UART_S	
23	22	21	20	19	18	17	16
Reserved	TMR3_S			Reserved	TMR2_S		
15	14	13	12	11	10	9	8
Reserved	TMR1_S			Reserved	TMR0_S		
7	6	5	4	3	2	1	0
Reserved		SPI1_S	SPI0_S	ADC_S		WDT_S	

Bits	Description	
[31:30]	**PWM23_S**	**PWM2 And PWM3 Clock Source Selection** PWM2 and PWM3 used the same peripheral clock source; both of them used the same prescaler. The peripheral clock source of PWM2 and PWM3 is defined by PWM23_S (CLKSEL1[31:30]) and PWM23_S_E (CLKSEL2[9]). If PWM23_S_E = 0, the perpherial clock source of PWM2 and PWM3 defined by PWM23_S list below:. 00 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 01 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from 22.1184 MHz internal high speed RC oscillator clock. If PWM23_S_E = 1, the peripheral clock source of PWM2 and PWM3 defined by PWM23_S list below:. 00 = Reserved. 01 = Reserved. 10 = Reserved. 11 = Clock source from 10 kHz internal low speed RC oscillator clock.
[29:28]	**PWM01_S**	**PWM0 And PWM1 Clock Source Selection** PWM0 and PWM1 used the same peripheral clock source; both of them used the same prescaler. The peripheral clock source of PWM0 and PWM1 is defined by PWM01_S (CLKSEL1[29:28]) and PWM01_S_E (CLKSEL2[8]). If PWM01_S_E = 0, the peripheral clock source of PWM0 and PWM1 defined by PWM01_S list below:. 00 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 01 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from 22.1184 MHz internal high speed RC oscillator clock. If PWM01_S_E = 1, the peripheral clock source of PWM0 and PWM1 defined by PWM01_S list below:. 00 = Reserved.

		01 = Reserved. 10 = Reserved. 11 = Clock source from 10 kHz internal low speed RC oscillator clock.
[27:26]	**Reserved**	Reserved.
[25:24]	**UART_S**	**UART Clock Source Selection** 00 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 01 = Clock source from PLL clock. 11 = Clock source from 22.1184 MHz internal high speed RC oscillator clock.
[23]	**Reserved**	Reserved.
[22:20]	**TMR3_S**	**TIMER3 Clock Source Selection** 000 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 001 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 010 = Clock source from HCLK. 011 = Clock source from external trigger. 101 = Clock source from 10 kHz internal low speed RC oscillator clock. 111 = Clock source from 22.1184 MHz internal high speed RC oscillator clock. Others = reserved.
[19]	**Reserved**	Reserved.
[18:16]	**TMR2_S**	**TIMER2 Clock Source Selection** 000 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 001 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 010 = Clock source from HCLK. 011 = Clock source from external trigger. 101 = Clock source from internal 10 kHz low speed RC oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed RC oscillator clock. Others = reserved.
[15]	**Reserved**	Reserved.
[14:12]	**TMR1_S**	**TIMER1 Clock Source Selection** 000 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 001 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 010 = Clock source from HCLK. 011 = Clock source from external trigger. 101 = Clock source from 10 kHz internal low speed RC oscillator clock. 111 = Clock source from 22.1184 MHz internal high speed RC oscillator clock. Others = reserved.
[11]	**Reserved**	Reserved.
[10:8]	**TMR0_S**	**TIMER0 Clock Source Selection** 000 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 001 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 010 = Clock source from HCLK. 011 = Clock source from external trigger. 101 = Clock source from 10 kHz internal low speed RC oscillator clock. 111 = Clock source from 22.1184 MHz internal high speed RC oscillator clock. Others = reserved.
[7:6]	**Reserved**	Reserved.

[5]	**SPI1_S**	**SPI1 Clock Source Selection** 0 = Clock source from PLL clock. 1 = Clock source from HCLK.
[4]	**SPI0_S**	**SPI0 Clock Source Selection** 0 = Clock source from PLL clock. 1 = Clock source from HCLK.
[3:2]	**ADC_S**	**ADC Clock Source Select** 00 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 01 = Clock source from PLL clock. 10 = Clock source from HCLK. 11 = Clock source from 22.1184 MHz internal high speed RC oscillator clock.
[1:0]	**WDT_S**	**Watchdog Timer Clock Source Select (Write Protect)** 00 = Reserved. 01 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 10 = Clock source from HCLK/2048 clock. 11 = Clock source from 10 kHz internal low speed RC oscillator clock. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Clock Source Select Control Register 2 (CLKSEL2)

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

Register	Offset	R/W	Description	Reset Value
CLKSEL2	CLK_BA+0x1C	R/W	Clock Source Select Control Register 2	0x0002_00FF

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**RTC_SEL_10K**	**WWDT_S**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**PWM45_S_E**	**PWM23_S_E**	**PWM01_S_E**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**PWM45_S**		**FRQDIV_S**		**Reserved**	

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**RTC_SEL_10K**	**RTC Clock Source Selection** 0 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 1 = Clock source from 10 kHz internal low speed RC oscillator clock.
[17:16]	**WWDT_S**	**Window Watchdog Timer Clock Source Selection** 10 = Clock source from HCLK/2048 clock. 11 = Clock source from 10 kHz internal low speed RC oscillator clock.
[15:11]	**Reserved**	Reserved.
[10]	**PWM45_S_E**	**PWM4 And PWM5 Clock Source Selection Extend** PWM4 and PWM5 used the same peripheral clock source; both of them used the same prescaler. The peripheral clock source of PWM4 and PWM5 is defined by PWM45_S (CLKSEL2[5:4]) and PWM45_S_E (CLKSEL2[10]). If PWM45_S_E = 0, the peripheral clock source of PWM4 and PWM5 defined by PWM45_S list below:. 00 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 01 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from 22.1184 MHz internal high speed RC oscillator clock. If PWM45_S_E = 1, the peripheral clock source of PWM4 and PWM5 defined by PWM45_S list below:. 00 = Reserved. 01 = Reserved. 10 = Reserved. 11 = Clock source from internal 10 kHz low speed oscillator clock.
[9]	**PWM23_S_E**	**PWM2 And PWM3 Clock Source Selection Extend** PWM2 and PWM3 used the same peripheral clock source; both of them used the same prescaler. The perpherial clock source of PWM2 and PWM3 is defined by PWM23_S

		(CLKSEL1[31:30]) and PWM23_S_E (CLKSEL2[9]). If PWM23_S_E = 0, the peripheral clock source of PWM2 and PWM3 defined by PWM23_S list below:. 00 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 01 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from 22.1184 MHz internal high speed RC oscillator clock. If PWM23_S_E = 1, the peripheral clock source of PWM2 and PWM3 defined by PWM23_S list below:. 00 = Reserved. 01 = Reserved. 10 = Reserved. 11 = Clock source from 10 kHz internal low speed RC oscillator clock.
[8]	**PWM01_S_E**	**PWM0 And PWM1 Clock Source Selection Extend** PWM0 and PWM1 used the same peripheral clock source; both of them used the same prescaler. The peripheral clock source of PWM0 and PWM1 is defined by PWM01_S (CLKSEL1[29:28]) and PWM01_S_E (CLKSEL2[8]). If PWM01_S_E = 0, the peripheral clock source of PWM0 and PWM1 defined by PWM01_S list below:. 00 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 01 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from 22.1184 MHz internal high speed RC oscillator clock. If PWM01_S_E = 1, the peripheral clock source of PWM0 and PWM1 defined by PWM01_S list below:. 00 = Reserved. 01 = Reserved. 10 = Reserved. 11 = Clock source from 10 kHz internal low speed RC oscillator clock.
[7:6]	**Reserved**	Reserved.
[5:4]	**PWM45_S**	**PWM4 And PWM5 Clock Source Selection** PWM4 and PWM5 used the same peripheral clock source; both of them used the same prescaler. The peripheral clock source of PWM4 and PWM5 is defined by PWM45_S (CLKSEL2[5:4]) and PWM45_S_E (CLKSEL2[10]). If PWM45_S_E = 0, the peripheral clock source of PWM4 and PWM5 defined by PWM45_S list below:. 00 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 01 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from 22.1184 MHz internal high speed RC oscillator clock. If PWM45_S_E = 1, the peripheral clock source of PWM4 and PWM5 defined by PWM45_S list below:. 00 = Reserved. 01 = Reserved. 10 = Reserved. 11 = Clock source from 10 kHz internal low speed RC oscillator clock.
[3:2]	**FRQDIV_S**	**Clock Divider Clock Source Selection** 00 = Clock source from 4~24 MHz external high speed crystal oscillator clock. 01 = Clock source from 32.768 kHz external low speed crystal oscillator clock. 10 = Clock source from HCLK.

		11 = Clock source from 22.1184 MHz internal high speed RC oscillator clock.
[1:0]	**Reserved**	Reserved.

Clock Divider Register (CLKDIV)

Register	Offset	R/W	Description	Reset Value
CLKDIV	CLK_BA+0x18	R/W	Clock Divider Number Register	0x0000_0000

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

Bits	Description	
[15:12]	**Reserved**	Reserved.
[23:16]	**ADC_N**	**ADC Clock Divide Number From ADC Clock Source** ADC clock frequency = (ADC clock source frequency) / (ADC_N + 1).
[15:12]	**Reserved**	Reserved.
[11:8]	**UART_N**	**UART Clock Divide Number From UART Clock Source** UART clock frequency = (UART clock source frequency) / (UART_N + 1).
[7:4]	**USB_N**	**USB Clock Divide Number From PLL Clock** USB clock frequency = (PLL frequency) / (USB_N + 1).
[3:0]	**HCLK_N**	**HCLK Clock Divide Number From HCLK Clock Source** HCLK clock frequency = (HCLK clock source frequency) / (HCLK_N + 1).

PLL Control Register (PLLCON)

The PLL reference clock input is from the 4~24 MHz external high speed crystal oscillator clock input or from the 22.1184 MHz internal high speed RC oscillator. These registers are used to control the PLL output frequency and PLL operating mode.

Register	Offset	R/W	Description	Reset Value
PLLCON	CLK_BA+0x20	R/W	PLL Control Register	0x0005_C22E

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19]	**PLL_SRC**	**PLL Source Clock Selection** 0 = PLL source clock from 4~24 MHz external high speed crystal oscillator. 1 = PLL source clock from 22.1184 MHz internal high speed RC oscillator.
[18]	**OE**	**PLL OE (FOUT Enable) Pin Control** 0 = PLL FOUT Enabled. 1 = PLL FOUT is fixed low.
[17]	**BP**	**PLL Bypass Control** 0 = PLL is in Normal mode (default). 1 = PLL clock output is same as PLL source clock input.
[16]	**PD**	**Power-Down Mode** If the PWR_DOWN_EN bit is set to 1 in PWRCON register, the PLL will enter Power-down mode too. 0 = PLL is in Normal mode. 1 = PLL is in Power-down mode (default).
[15:14]	**OUT_DV**	**PLL Output Divider Control Bits** Refer to the formulas below the table.
[13:9]	**IN_DV**	**PLL Input Divider Control Bits** Refer to the formulas below the table.
[8:0]	**FB_DV**	**PLL Feedback Divider Control Bits** Refer to the formulas below the table.

Output Clock Frequency Setting

$$FOUT = FIN \times \frac{NF}{NR} \times \frac{1}{NO}$$

Constraint:

1. $3.2MHz < FIN < 150MHz$
2. $800KHz < \frac{FIN}{2 * NR} < 7.5MHz$
3. $100MHz < FCO = FIN * \frac{NF}{NR} < 200MHz$

   $120MHz <$ FCO is preferred

Symbol	Description
FOUT	Output Clock Frequency
FIN	Input (Reference) Clock Frequency
NR	Input Divider (IN_DV + 2)
NF	Feedback Divider (FB_DV + 2)
NO	OUT_DV = “00” : NO = 1 OUT_DV = “01” : NO = 2 OUT_DV = “10” : NO = 2 OUT_DV = “11” : NO = 4

Default Frequency Setting

The default value: 0xC22E
FIN = 12 MHz
NR = (1+2) = 3
NF = (46+2) = 48
NO = 4
FOUT = 12/4 x 48 x 1/3 = 48 MHz

Frequency Divider Control Register (FRQDIV)

Register	Offset	R/W	Description	Reset Value
FRQDIV	CLK_BA+0x24	R/W	Frequency Divider 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	**CLKO_1HZ_EN**	**DIVIDER1**	**DIVIDER_EN**	**FSEL**			

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**CLKO_1HZ_EN**	**Clock Output 1Hz Enable Bit** 0 = 1 Hz clock output for 32.768 kHz external low speed crystal oscillator clock frequency compensation Disabled. 1 = 1 Hz clock output for 32.768 kHz external low speed crystal oscillator clock frequency compensation Enabled.
[5]	**DIVIDER1**	**Frequency Divider One Enable Bit** 0 = Frequency divider will output clock with source frequency divided by FSEL. 1 = Frequency divider will output clock with source frequency.
[4]	**DIVIDER_EN**	**Frequency Divider Enable Bit** 0 = Frequency Divider function Disabled. 1 = Frequency Divider function Enabled.
[3:0]	**FSEL**	**Divider Output Frequency Selection Bits** 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 FSEL[3:0].

6.4 Flash Memory Controller (FMC)

6.4.1 Overview

The NuMicro® NUC029LEE/NUC029SEE has 128K bytes on-chip embedded Flash for application program memory (APROM) that can be updated through ISP procedure. The In-System-Programming (ISP) function enables user to update program memory when chip is soldered on PCB. After chip is powered on, Cortex®-M0 CPU fetches code from APROM or LDROM decided by boot select (CBS) in CONFIG0. By the way, the NuMicro® NUC029LEE/NUC029SEE also provides additional Data Flash for user to store some application dependent data. The Data Flash is shared with original 128 KB program memory and its start address is configurable in CONFIG1

6.4.2 Features

- Runs up to 50 MHz with zero wait cycle for continuous address read access and runs up to 72 MHz with one wait cycle for continuous address read.
- All embedded flash memory supports 512 bytes page erase
- 128 KB application program memory (APROM)
- 8 KB In-System-Programming (ISP) loader program memory (LDROM)
- Configurable Data Flash size with 512 bytes page erase unit
- Supports In-Application-Programming (IAP) to switch code between APROM and LDROM without reset
- In-System-Programming (ISP) to update on-chip Flash

6.4.3 Block Diagram

The flash memory controller consists of AHB slave interface, ISP control logic, writer interface and flash macro interface timing control logic. The block diagram of flash memory controller is shown as follows:

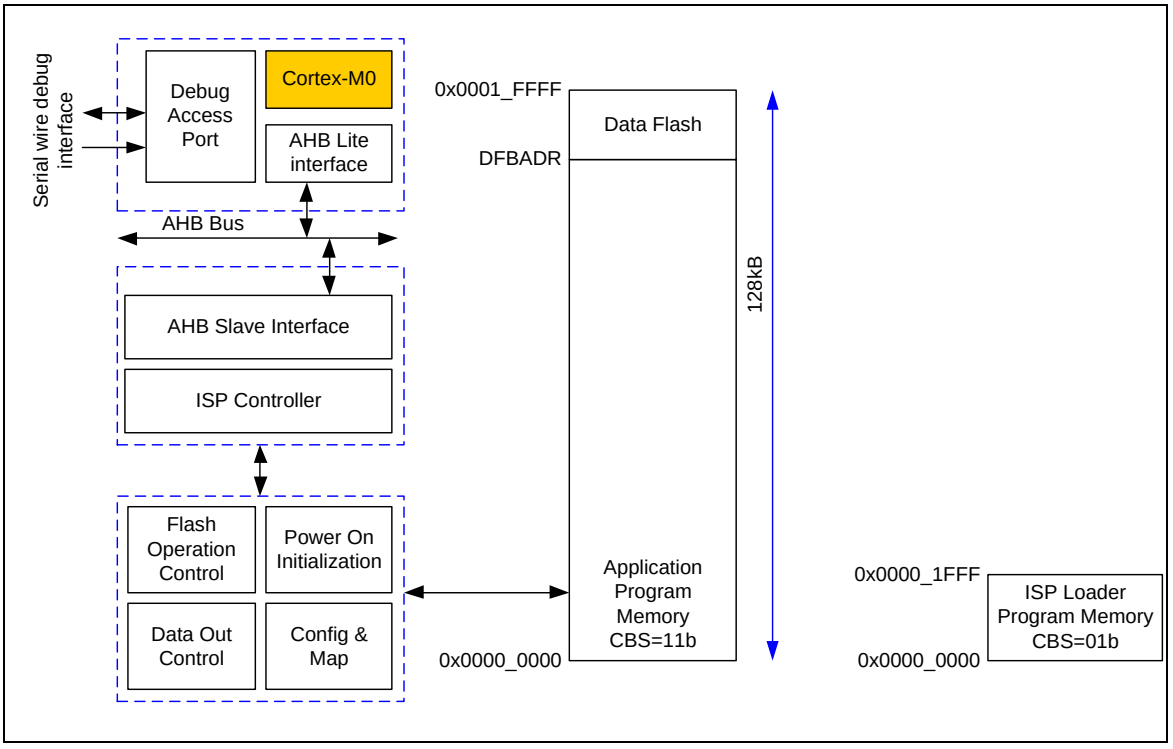


Figure 6.4-1 Flash Memory Control Block Diagram

6.4.4 Functional Description

6.4.4.1 Flash Memory Organization

The NuMicro® NUC029LEE/NUC029SEE flash memory consists of program memory (APROM), Data Flash, ISP loader program memory (LDROM), and user configuration.

Program memory is main memory for user applications and called APROM. User can write their application to APROM and set system to boot from APROM.

ISP loader program memory is designed for a loader to implement In-System-Programming function. LDROM is independent to APROM and system can also be set to boot from LDROM. Therefore, user can use LDROM to avoid system boot fail when code of APROM was corrupted.

Data Flash is used for user to store data. It can be read by ISP read or memory read and programmed through ISP procedure. The size of each erase unit is 512 bytes. The Data Flash and application program share the same 128 KB memory, if DFEN (Data Flash Enable) bit in CONFIG0 is enabled, the Data Flash base address is defined by DFBADR and its size is (0x20000 - DFBADR), At the same time, the APROM size will be (128 KB – Data Flash size).

User configuration provides several bytes to control system logic, such as flash security lock, boot select, Brown-out voltage level, Data Flash base address, etc…. User configuration works like a fuse for power on setting and loaded from flash memory to its corresponding control registers during chip powered on.

In NuMicro® Family, the flash memory organization is different to system memory map. Flash memory organization is used when user using ISP command to read, program or erase flash memory. System memory map is used when CPU access flash memory to fetch code or data. For example, When system is set to boot from LDROM by CBS = 01b, CPU will be able to fetch code of LDROM from 0x0 ~ 0x1FFF. However, if user want to read LDROM by ISP, they still need to read the address of LDROM as 0x0010_0000 ~ 0x0010_1FFF.

Table 6.4-1 and Figure 6.4-2 show the address mapping information of APROM, LDROM, Data Flash and user configuration.

Block Name	Device Type	Size		Start Address	End Address
APROM	128 KB	Data Flash Enable	128 KB - Data Flash Size	0x0000_0000	0x20000 – Data Flash Size - 1
		Data Flash Disable	128 KB	0x0000_0000	0x0001_FFFF
Data Flash	128 KB	Data Flash Enable	0x20000-DFBADR	DFBADR	0x0001_FFFF
		Data Flash Disable	0 KB	N/A	N/A
LDROM	128 KB	8 KB		0x0010_0000	0x0010_1FFF
User Configuration	128 KB	2 words		0x0030_0000	0x0030_0004

Table 6.4-1 Memory Address Map

The Flash memory organization is shown as Figure 6.4-2:

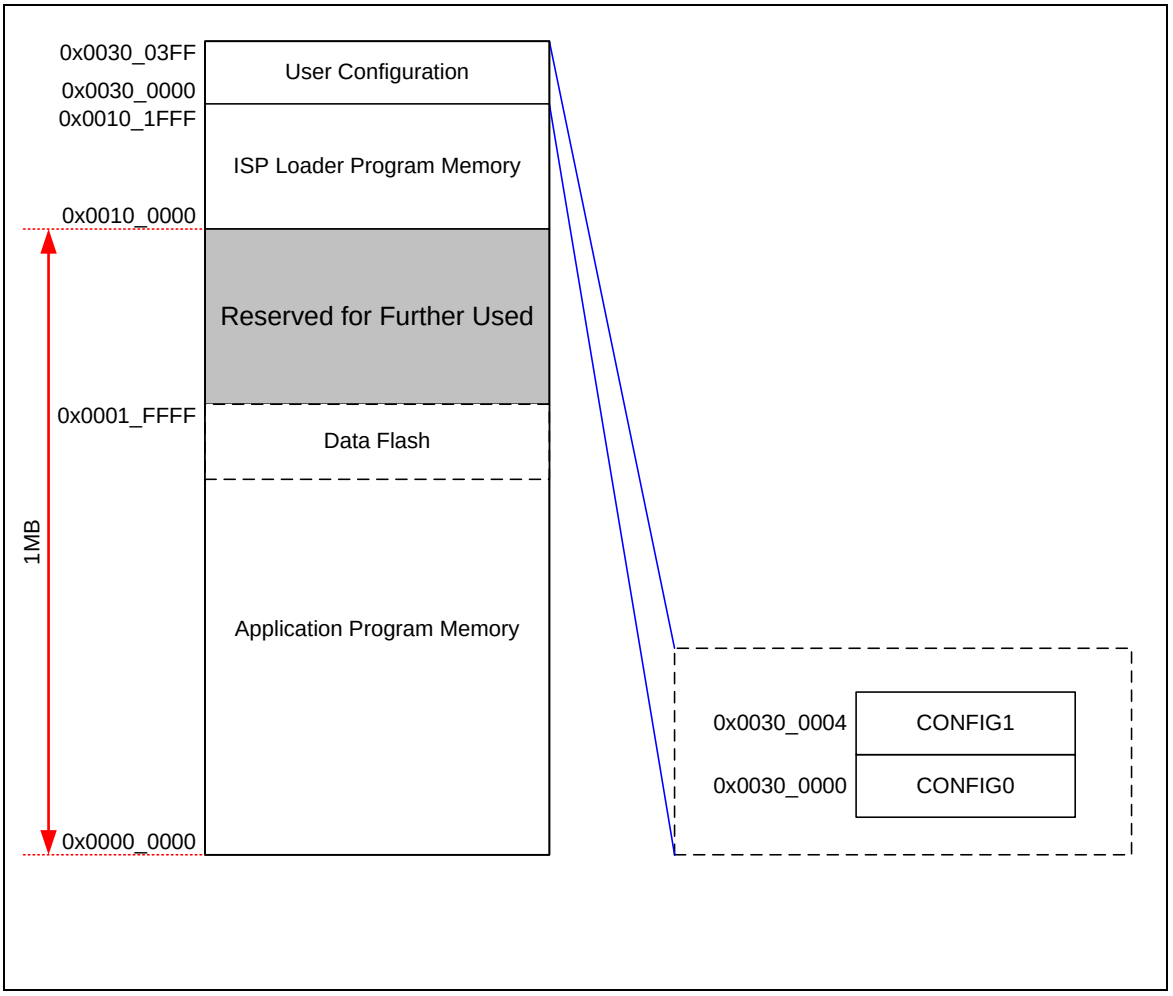


Figure 6.4-2 Flash Memory Organization

6.4.4.2 User Configuration

User configuration is internal programmable configuration area for boot options. The user configuration is located at 0x300000 of Flash Memory Organization and they are two 32 bits words. Any change on user configuration will take effect after system reboot.

CONFIG0 (Address = 0x0030_0000)

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

CONFIG0	Address = 0x0030_0000	
Bits	**Description**	
[31]	**CWDTEN**	**Watchdog Enable Bit** 0 = Watchdog Timer Enabled and force Watchdog Timer clock source as OSC10K after chip powered on. 1 = Watchdog Timer Disabled after chip powered on.
[30]	**CWDTPDEN**	**Watchdog Clock Power-down Enable Bit** 0 = OSC10K Watchdog Timer clock source is forced to be always enabled. 1 = OSC10K Watchdog Timer clock source is controlled by OSC10K_EN (PWRCON[3]) when chip enters Power-down. **Note:** This bit only works at CWDTEN is set to 0
[29:28]	**Reserved**	Reserved
[27]	**CGPFMFP**	**GPF Multi-function Selection** 0 = XT1_IN and XT1_OUT pin is configured as GPIO function. 1 = XT1_IN and XT1_OUT pin is used as external 4~24MHz crystal oscillator pin. **Note:** XT1_IN, XT1_OUT multi-function can only be changed by CGPFMFP.
[26:24]	**CFOSC**	**CPU Clock Source Selection after Reset** 000 = External 4~24 MHz high speed crystal oscillator clock. 111 = Internal RC 22.1184 MHz high speed oscillator clock. Others = Reserved. The value of CFOSC will be load to HCLK_S (CLKSEL0[2:0]) in system register after any reset occurs.
[23]	**CBODEN**	**Brown-out Detector Enable Bit** 0= Brown-out detect Enabled after powered on. 1= Brown-out detect Disabled after powered on.

[22:21]	**CBOV**	**Brown-out Voltage Selection** 00 = 2.2 V 01 = 2.7 V 10 = 3.7 V 11 = 4.4 V
[20]	**CBORST**	**Brown-out Reset Enable Bit** 0 = Brown-out reset Enabled after powered on. 1 = Brown-out reset Disabled after powered on.
[19:11]	**Reserved**	Reserved
[10]	**CIOINI**	**I/O Initial State Select** 0 = All GPIO default to be input tri-state mode after powered on 1 = All GPIO default to be Quasi-bidirectional mode after chip is powered on **Note:** It is recommended to use 100 kΩ pull-up resistor on both ICE_DAT and ICE_CLK pin.
[9:8]	**Reserved**	Reserved
[7:6]	**CBS**	**Chip Boot Selection** 00 = Boot from LDROM with IAP function 01 = Boot from LDROM without IAP function 10 = Boot from APROM with IAP function 11 = Boot from APROM without IAP function IAP function means APROM and LDROM can be executed and access by CPU without reset. When IAP function enabled, APROM base address is 0x0 and LDROM base address is 0x100000.
[5:2]	**Reserved**	Reserved
[1]	**LOCK**	**Security Lock** 0 = Flash data is locked 1 = Flash data is not locked When flash data is locked, only device ID, CONFIG0 and CONFIG1 can be read by writer and ICP through serial debug interface. Others data is locked as 0xFFFFFFFF. ISP can read data anywhere regardless of LOCK bit value. User need to erase whole chip by ICP/Writer tool or erase user configuration by ISP to unlock.
[0]	**DFEN**	**Data Flash Enable Bit** 0 = Data Flash Enabled. 1 = Data Flash Disabled. **Note:** This bit only for 128 KB APROM Device

Brown-out detection function is for monitoring the voltage on V_{DD} pin. If V_{DD} voltage falls below level setting of CBOV, the BOD event will be triggered when BOD enabled. User can decide to use BOD reset by enable CBORST or just enable BOD interrupt by NVIC when BOD detected. Because BOD reset is issued whenever V_{DD} voltage falls below the level setting of CBOV, user must make sure the CBOV setting to avoid BOD reset right after BOD reset enabled. For example, if the V_{DD} is 3.3V, CBOV could only be 00'b or 01'b. Otherwise, the system will be halted in BOD reset state when BOD reset is enabled and CBOV is 10'b or 11'b.

CONFIG1 (Address = 0x0030_0004)

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

Config	Address = 0x0030_0004	
Bits	Description	
[31:20]	**Reserved**	Reserved (It is mandatory to program 0x00 to these Reserved bits)
[19:0]	**DFBADR**	**Data Flash Base Address** 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 as 0. This configuration is only valid for 128 KB flash device.

6.4.4.3 Boot Selection

The NuMicro® NUC029LEE/NUC029SEE provides In-System-Programming (ISP) feature to enable user to update program memory by a stand-alone ISP firmware. A dedicated 8 KB program memory (LDROM) is used to store ISP firmware. User can select to start program fetch from APROM or LDROM by CBS[1] in CONFIG0.

In addition to setting boot from APROM or LDROM, CBS in CONFIG0 is also used to control system memory map after booting. When CBS[0] = 1 and set CBS[1] = 1 to boot from APROM, the application in APROM will not be able to access LDROM by memory read. In other words, when CBS[0] = 1 and CBS[1] = 0 are set to boot from LDROM, the software executed in LDROM will not be able to access APROM by memory read. Figure 6.4-3 shows the memory map when booting from APROM and LDROM.

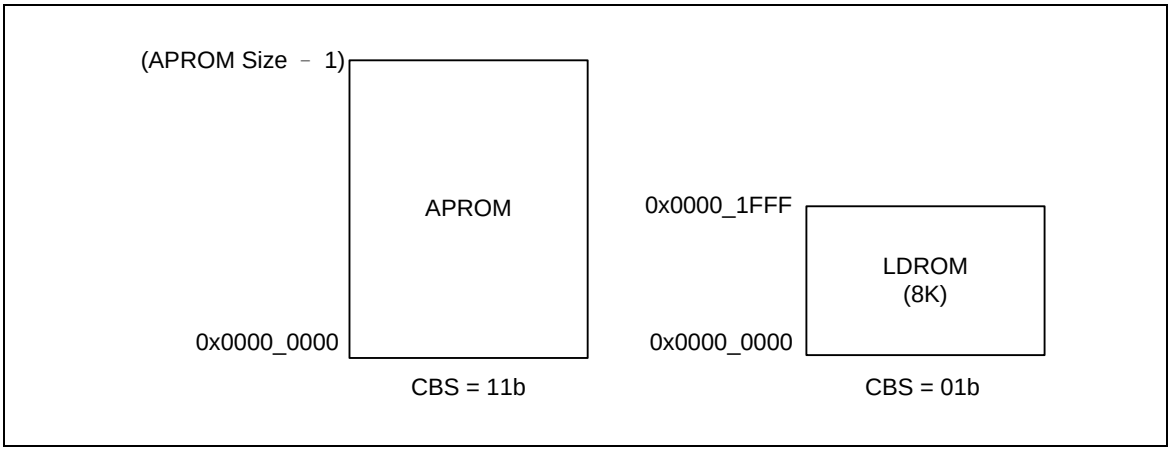


Figure 6.4-3 Program Executing Range for Booting from APROM and LDROM

For the application that software needs to execute code in APROM and call the functions in LDROM or to execute code in LDROM and call the APROM function without changing boot mode, CBS[0] needs to be set as 0 and this is called In-Application-Programming(IAP).

6.4.4.4 In-Application-Programming (IAP)

The NuMicro® NUC029LEE/NUC029SEE provides In-application-programming (IAP) function for user to switch the code executing between APROM and LDROM without a reset. User can enable the IAP function by re-booting chip and setting the chip boot selection bits in CONFIG0 (CBS[1:0]) as 10b or 00b.

In the case that the chip boots from APROM with the IAP function enabled (CBS[1:0] = 10b), the executable range of code includes all of APROM and LDROM. The address space of APROM is kept as the original size but the address space of the 8 KB LDROM is mapped to 0x0010_0000~ 0x0010_1FFF.

In the case that the chip boots from LDROM with the IAP function enabled (CBS[1:0] = 00b), the executable range of code includes all of LDROM and almost all of APROM except for its first page. User cannot access the first page of APROM by CPU because the first page of executable code range becomes the mirror of the first page of LDROM as set by default. Meanwhile, the address space of 8 KB LDROM is mapped to 0x0010_0000~0x0010_1FFF.

Please refer to Figure 6.4-4 for the address map while IAP is activating.

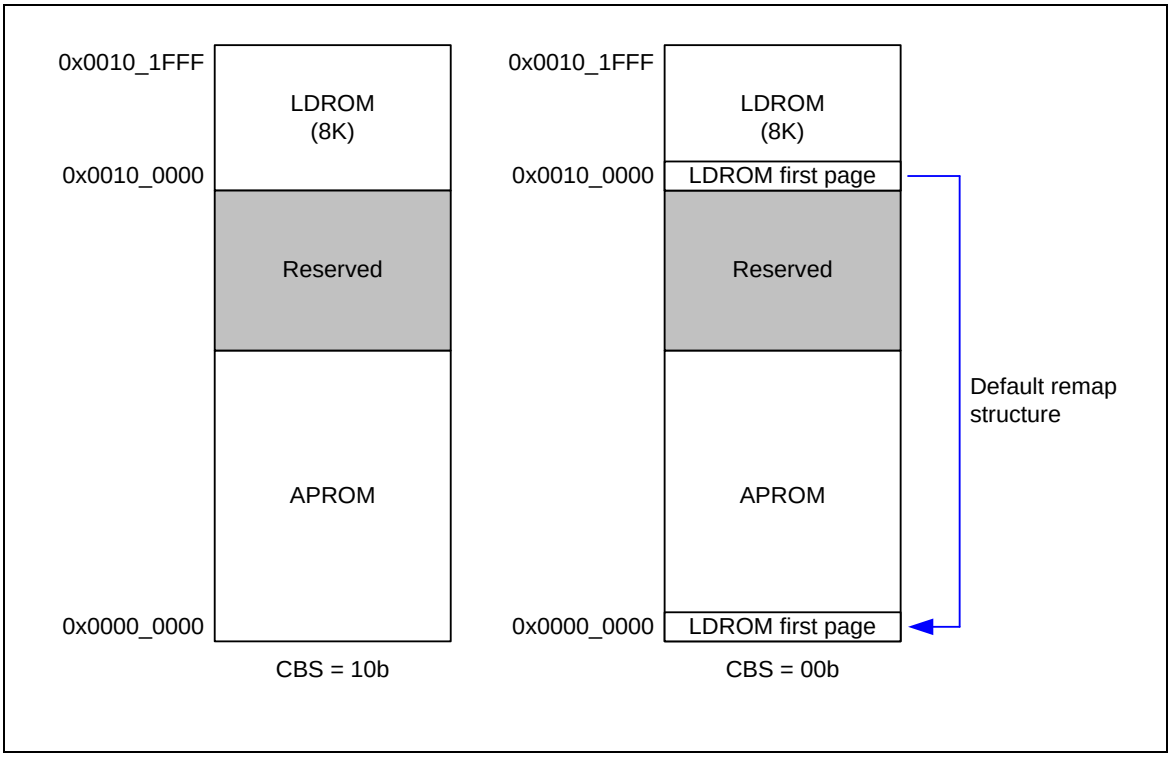


Figure 6.4-4 Executable Range of Code with IAP Function Enabled

When chip boots with the IAP function enabled, any other page within the executable range of code can be mirrored to the first page of executable code (0x0000_0000~0x0000_01FF) any time. User can change the remap address of the first executing page by filling the target remap address to ISPADR and then go through ISP procedure with the Vector Page Re-map command. After changing the remap address, user can check if the change is successful by reading the VECMAP field in the ISPSTA register.

6.4.4.5 In-System-Programming (ISP)

The NuMicro® NUC029LEE/NUC029SEE supports ISP mode which allows a device to be reprogrammed under software control and avoids system fail risk when download or programming fail. Furthermore, the capability to update the application firmware makes a wide range of applications possible.

ISP provides the ability to update system firmware on board. Various peripheral interfaces let ISP loader in LDROM to update application program code easily. The most common method to perform ISP is via UART along with the ISP loader in LDROM. General speaking, PC transfers the new APROM code through serial port. Then ISP loader receives it and re-programs into APROM through ISP commands.

6.4.4.6 ISP Procedure

The NuMicro® NUC029LEE/NUC029SEE supports booting from APROM or LDROM initially defined by user configuration. The change of user configuration needs to reboot system to make it take effect. If user wants to switch between APROM or LDROM mode without changing user configuration, he needs to control BS bit of ISPCON control register, then reset CPU by IPRSTC1 control register. The boot switching flow by BS bit is shown in the Figure 6.4-5.

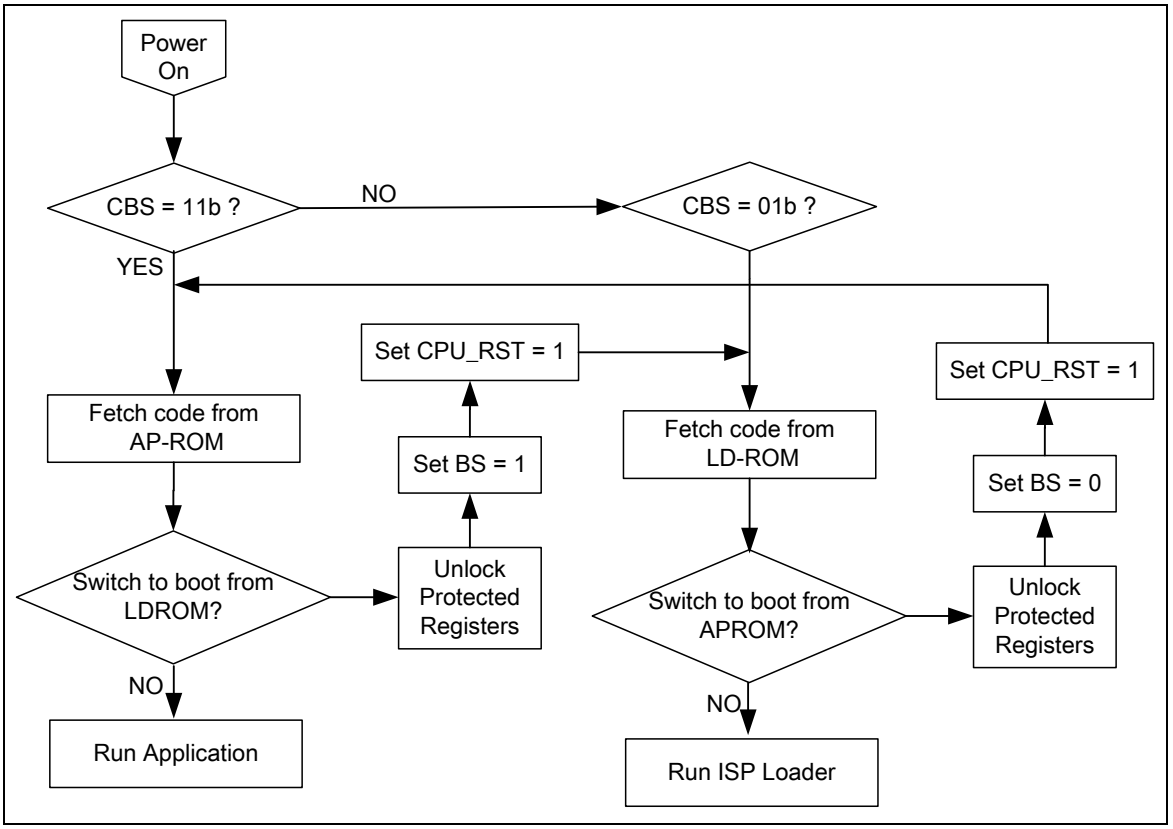


Figure 6.4-5 Example Flow of Boot Selection by BS Bit

Updating APROM by software in LDROM or updating LDROM by software in APROM can avoid a system failure when update fails.

The ISP controller supports to read, erase and program embedded flash memory. Several control bits of ISP controller are write-protected, thus it is necessary to unlock before we can set them. To unlock the protected register bits, software needs to write 0x59, 0x16 and 0x88 sequentially to REGWRPROT. If register is unlocked successfully, the value of REGWRPROT will be 1. The unlock sequence must not be interrupted by other access; otherwise it may fail to unlock.

After unlocking the protected register bits, user needs to set the ISPCON control register to decide to update LDROM, User Configuration, APROM and enable ISP controller.

Once the ISPCON register is set properly, user can set ISPCMD for erase, read or programming. Set ISPADR for target flash memory based on flash memory origination. ISPDAT can be used to set the data to program or used to return the read data according to ISPCMD.

Finally, set ISPGO bit of ISPTRG control register to perform the relative ISP function. The ISPGO bit is self-cleared when ISP function has been done. To make sure ISP function has been finished before CPU goes ahead, ISB instruction is used right after ISPGO setting.

Several error conditions are 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 flag can only be cleared by software. The next ISP procedure can be started even ISPFF bit is kept as 1. Therefore, it is recommended to check the ISPFF bit and clear it after each ISP operation if it is set to 1.

When the ISPGO 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

bit. User should add ISB instruction next to the instruction in which ISPGO bit is set 1 to ensure correct execution of the instructions following ISP operation.

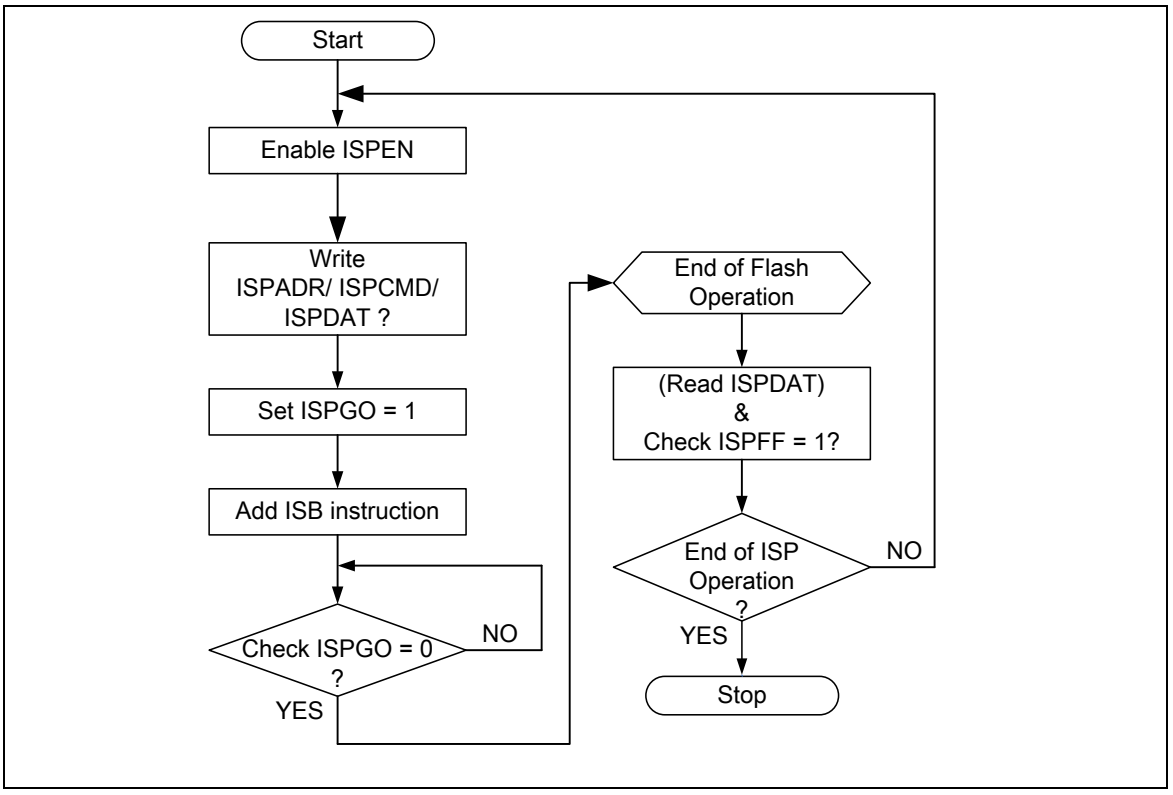


Figure 6.4-6 ISP Flow Example

ISP Command	ISPCMD	ISPADR	ISPDAT
FLASH Page Erase	0x22	Valid address of flash memory origination. It must be 512 bytes page alignment.	N/A
FLASH Program	0x21	Valid address of flash memory origination	Programming Data
FLASH Read	0x00	Valid address of flash memory origination	Return Data
Read Unique ID	0x04	0x0000_0000	Unique ID Word 0
		0x0000_0004	Unique ID Word 1
		0x0000_0008	Unique ID Word 2
Vector Page Re-Map	0x2E	Page in APROM or LDROM It must be 512 bytes page alignment	N/A

Table 6.4-2 ISP Command List

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 = 0x5000_C000**				
ISPCON	FMC_BA+0x00	R/W	ISP Control Register	0x0000_0000
ISPADR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000
ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000
ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000
ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000
DFBADR	FMC_BA+0x14	R	Data Flash Base Address	0x000X_XXXX
FATCON	FMC_BA+0x18	R/W	Flash Access Time Control Register	0x0000_0000
ISPSTA	FMC_BA+0x40	R/W	ISP Status Register	0x0000_0000

6.4.6 Register Description

ISP Control Register (ISPCON)

Register	Offset	R/W	Description	Reset Value
ISPCON	FMC_BA+0x00	R/W	ISP 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	**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: (1) APROM writes to itself if APUEN is set to 0 (2) LDROM writes to itself if LDUEN is set to 0 (3) CONFIG is erased/programmed if CFGUEN is set to 0 (4) Destination address is illegal, such as over an available range Write 1 to clear to this bit to 0.
[5]	**LDUEN**	**LDROM Update Enable Bit (Write Protect)** 0 = LDROM cannot be updated. 1 = LDROM can be updated when chip runs in APROM.
[4]	**CFGUEN**	**Enable Config Update By ISP (Write Protect)** 0 = ISP update config-bit Disabled. 1 = ISP update config-bit Enabled.
[3]	**APUEN**	**APROM Update Enable Bit (Write Protect)** 0 = APROM cannot be updated when chip runs in APROM. 1 = APROM can be updated when chip runs in APROM.
[2]	**Reserved**	Reserved.
[1]	**BS**	**Boot Select (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 in CONFIG0 after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened 0 = Boot from APROM. 1 = Boot from LDROM.

[0]	**ISPEN**	**ISP Enable Bit (Write Protect)** ISP function enable bit. Set this bit to enable ISP function. 0 = ISP function Disabled. 1 = ISP function Enabled.

ISP Address Register (ISPADR)

Register	Offset	R/W	Description	Reset Value
ISPADR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**ISPADR**	**ISP Address** The NuMicro® NUC029LEE/NUC029SEE has a maximum of 32Kx32 (128 KB) embedded Flash, which supports word program only. ISPADR[1:0] must be kept 00b for ISP operation.

ISP Data Register (ISPDAT)

Register	Offset	R/W	Description	Reset Value
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

ISP Command Register (ISPCMD)

Register	Offset	R/W	Description	Reset Value
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		**ISPCMD**					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**ISPCMD**	**ISP Command** ISP command table is shown below: 0x00 = Read. 0x04 = Read Unique ID. 0x0B = Read Company ID (0xDA). 0x21 = Program. 0x22 = Page Erase. 0x2E = Set Vector Page Re-Map.

ISP Trigger Control Register (ISPTRG)

Register	Offset	R/W	Description	Reset Value
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-Protection Bit)** 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 finished. 1 = ISP progressed. This bit is the protected bit, It means programming this bit needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100

Data Flash Base Address Register (DFBADR)

Register	Offset	R/W	Description	Reset Value
DFBADR	FMC_BA+0x14	R	Data Flash Base Address	0x000X_XXXX

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

Bits	Description	
[31:0]	DFBADR	**Data Flash Base Address** This register indicates Data Flash start address. It is read only. Tthe Data Flash size is defined by user configuration, register content is loaded from CONFIG1 when chip is powered on.

Flash Access Time Control Register (FATCON)

Register	Offset	R/W	Description	Reset Value
FATCON	FMC_BA+0x18	R/W	Flash Access Time Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	FOMSEL1	Reserved	FOMSEL0	Reserved			

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**FOMSEL1**	**Chip Frequency Optimization Mode Select1 (Write-protection Bit)**
[5]	**Reserved**	Reserved.
[4]	**FOMSEL0**	**Chip Frequency Optimization Mode Select 0 (Write-Protection Bit)** When CPU frequency is lower than 72 MHz, user can modify flash access delay cycle by FOMSEL1 and FOMSEL0 to improve system performance. 00 = CPU runs at 50MHz with zero wait cycle for continuous address read access. 01 = CPU runs at 25MHz with zero wait cycle for random address read access. 10 = CPU runs at 50MHz with zero wait cycle for continuous address read access. 11 = CPU runs at 72MHz with one wait cycle for continuous address read access. Where 00 means FOMSEL1 = 0, FOMSEL0 = 0; 01 means FOMSEL1 = 0, FOMSEL0 = 1 and etc.
[3:0]	**Reserved**	Reserved.

ISP Status Register (ISPSTA)

Register	Offset	R/W	Description	Reset Value
ISPSTA	FMC_BA+0x40	R/W	ISP Status Register	0x0000_0000

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

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20:9]	**VECMAP**	**Vector Page Mapping Address (Read Only)** The current flash address space 0x0000_0000~0x0000_01FF is mapping to address {VECMAP[11:0], 9'h000} ~ {VECMAP[11:0], 9'h1FF}
[8:7]	**Reserved**	Reserved.
[6]	**ISPFF**	**ISP Fail Flag (Write-Protection Bit)** This bit is set by hardware when a triggered ISP meets any of the following conditions: (1) APROM writes to itself (2) LDROM writes to itself (3) CONFIG is erased/programmed if CFGUEN is set to 0 (4) Destination address is illegal, such as over an available range Write 1 to clear this bit. **Note:** The function of this bit is the same as ISPCON bit6
[5:3]	**Reserved**	Reserved.
[2:1]	**CBS**	**Chip Boot Selection (Read Only)** This is a mirror of CBS in CONFIG0.
[0]	**ISPGO**	**ISP Start Trigger (Read Only)** 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 finished. 1 = ISP operation progressed. **Note:** This bit is the same as ISPTRG bit0

6.5 External Bus Interface (EBI)

6.5.1 Overview

The NuMicro® NUC029LEE/NUC029SEE LQFP-64 package equips an external bus interface (EBI) for access external device.

To save the connections between external device and this chip, EBI supports address bus and data bus multiplex mode. And, address latch enable (ALE) signal is used to differentiate the address and data cycle.

6.5.2 Features

External Bus Interface has the following functions:

- Supports external devices with max. 64 KB size (8-bit data width)/128 KB (16-bit data width)
- Supports variable external bus base clock (MCLK) which based on HCLK
- Supports 8-bit or 16-bit data width
- Supports variable data access time (tACC), address latch enable time (tALE) and address hold time (tAHD)
- Supports address bus and data bus multiplex mode to save the address pins
- Supports configurable idle cycle for different access condition: Write command finish (W2X), Read-to-Read (R2R)

6.5.3 Block Diagram

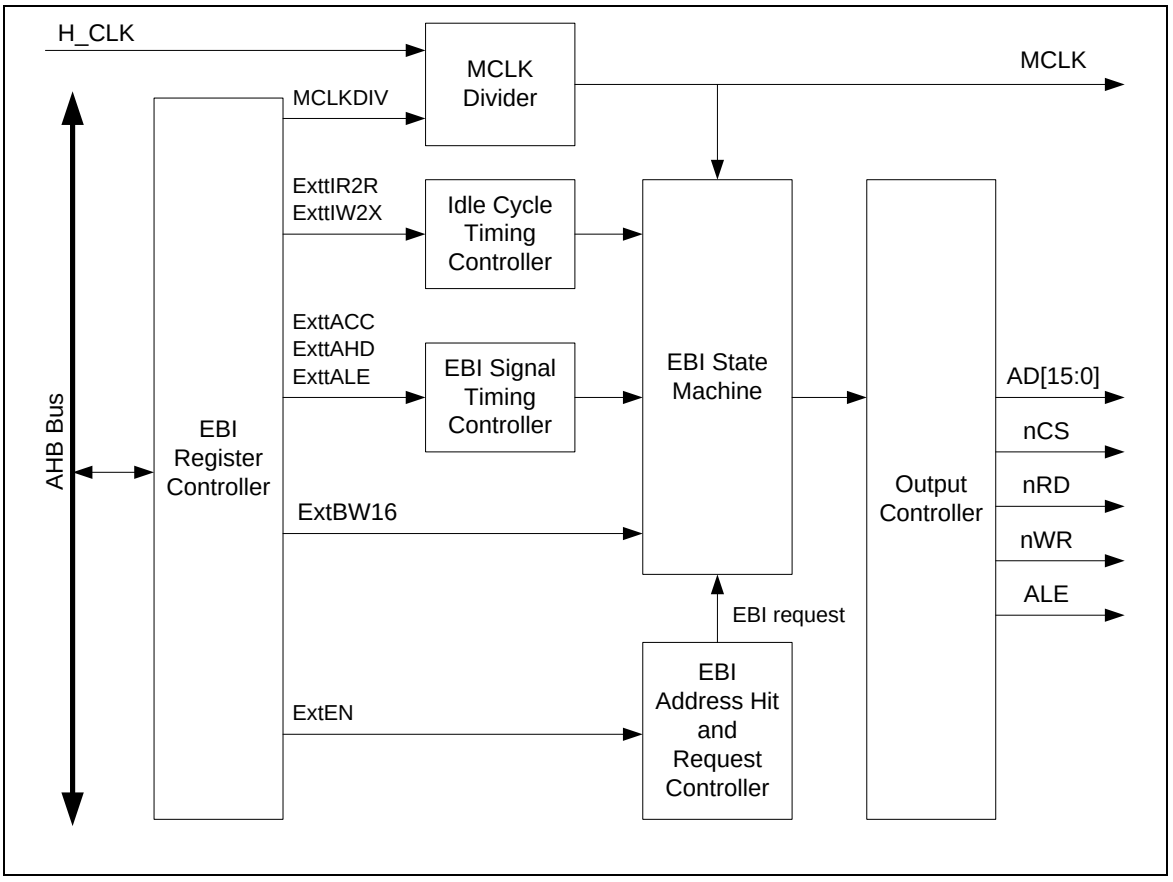


Figure 6.5-1 EBI Block Diagram

6.5.4 Functional Description

6.5.4.1 EBI Area and Address Hit

The EBI mapping address is located at 0x6000_0000 ~ 0x6001_FFFF and the maximum available memory space is 128 Kbytes. When the system request address hits EBI's memory space, the corresponding EBI chip select signal (nCS) is assert and EBI state machine operates.

For an 8-bit device (64 Kbytes), EBI mapped this 64 Kbytes device to 0x6000_0000 ~ 0x6000_FFFF and 0x6001_0000 ~ 0x6001_FFFF simultaneously.

For a 16-bit device (128 Kbytes), EBI mapped this 128 Kbytes device to 0x6000_0000 ~ 0x6001_FFFF.

6.5.4.2 EBI Data Width Connection

The EBI controller supports to connect the external device whose address bus and data bus are multiplexed. For the external device with separated address and data bus, the connection to device needs additional logic (latch device) to latch the address. In this case, pin ALE is connected to the latch device to latch the address value. Pins AD0~AD15 for 16-bit data width / Pins AD0~AD7 for 8-bit data width are the input pins of the latch device, and the output pins of the latch device are connected to the Addr[15:0] of external device.

For 16-bit device, the AD [15:0] shared by address (Addr [15:0]) and 16-bit data (Data [15:0]). For

8-bit device, only AD [7:0] shared by address (Addr [7:0]) and 8-bit data (Data [7:0]), AD [15:8] is dedicated for address (Addr [15:8]) and could be connected to 8-bit device directly.

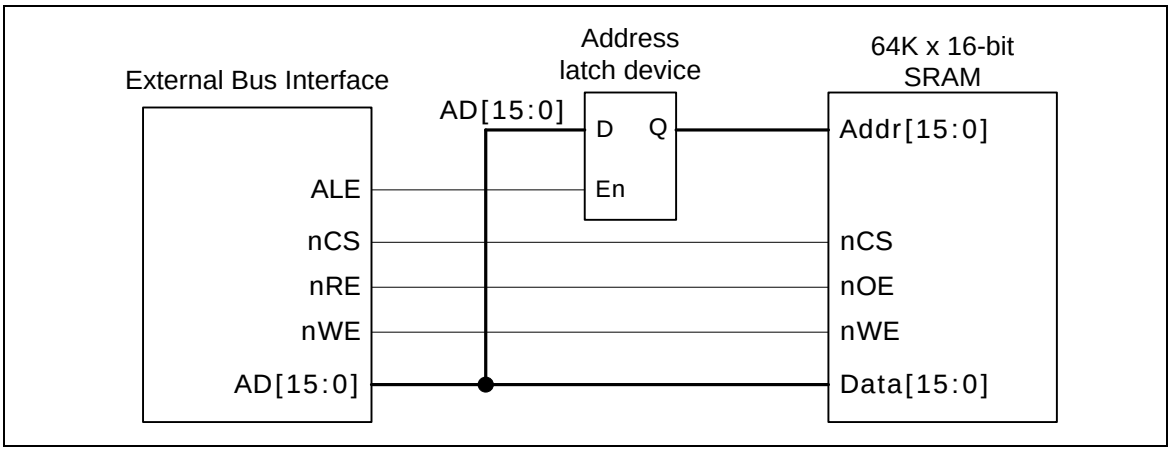


Figure 6.5-2 Connection of 16-bit EBI Data Width with 16-bit Device

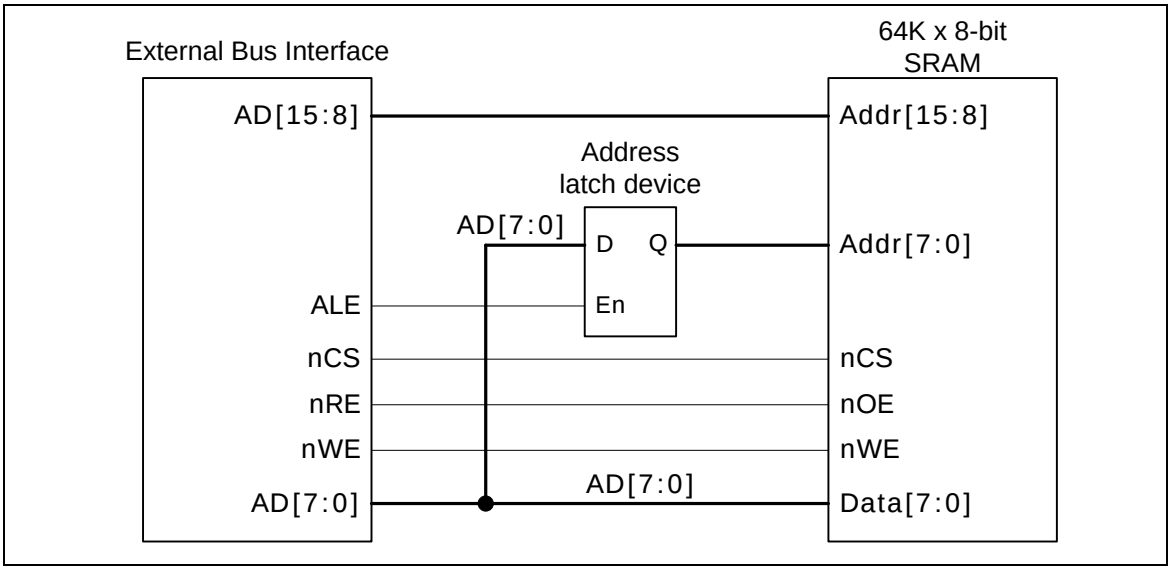


Figure 6.5-3 Connection of 8-bit EBI Data Width with 8-bit Device

When the system access data width is larger than EBI data width (8-bit / 16 bit data width), the EBI controller will finish a system access command by operating EBI access more than once. For example, if system requests a 32-bit data through EBI device, the EBI controller will operate accessing four times when setting EBI data width with 8-bit data width.

6.5.4.3 EBI Operating Control

MCLK Control

In the chip, all EBI signals will be synchronized by MCLK when EBI is operating. When the chip connects to the external device with slower operating frequency, the MCLK can divide most to 32 from HCLK by setting MCLKDIV[2:0] (EBICON [10:8] External Output Clock Divider). Therefore, the EBI controller is suitable for a wide frequency range of EBI device. If MCLK frequency is

setting as HCLK/1, EBI signals are synchronized by positive edge of MCLK, else by negative edge of MCLK.

Operation and Access Timing Control

In the start of EBI access, chip select signal (nCS) asserts to low and waits one MCLK for address setup time (tASU) for address stable. Then ALE signal asserts to high after address is stable and keeps for a period of time (tALE) for latch address. After latch address, ALE signal asserts to low and wait one MCLK for address latch hold time (tLHD) and another one MCLK cycle (tA2D) that is inserted behind address latch hold time (tLHD) to be the bus turn-around time for address change to data. Then nRD signal asserts to low when read access or nWR signal asserts to low when write access. Then nRD or nWR signal asserts to high after keeps access time (tACC) for reading output stable or writing finish. After that, EBI signals keep for data access hold time (tAHD) and chip select signal (nCS) asserts to high then address is released by current access control.

The EBI controller provides a flexible timing control for different external devices. In EBI timing control, tASU, tLHD and tA2D are all fixed to 1 MCLK cycle, tAHD can modulate to 1~8 MCLK cycles by setting ExttAHD[2:0] (EXTIME [10:8] EBI Data Access Hold Time), tACC can modulate to 1~32 MCLK cycles by setting ExttACC[4:0] (EXTIME [7:3] EBI Data Access Time), and tALE can modulate to 1~8 MCLK cycles by setting ExttALE [2:0] (EBICON [18:16] Expand Time of ALE).

Parameter	Value	Unit	Description
tASU	1	MCLK	Address Latch Setup Time.
tALE	1 ~ 8	MCLK	ALE High Period. Controlled by ExttALE[2:0] of EBICON[18:16].
tLHD	1	MCLK	Address Latch Hold Time.
tA2D	1	MCLK	Address To Data Delay (Bus Turn-Around Time).
tACC	1 ~ 32	MCLK	Data Access Time. Controlled by ExttACC[4:0] of EXTIME[7:3].
tAHD	1 ~ 8	MCLK	Data Access Hold Time. Controlled by ExttAHD[2:0] of EXTIME[10:8].
IDLE	0 ~ 15	MCLK	Idle Cycle. Controlled by ExtIR2R[3:0] of EXTIME[27:24] and ExtIW2X[3:0] of EXTIME[15:12].

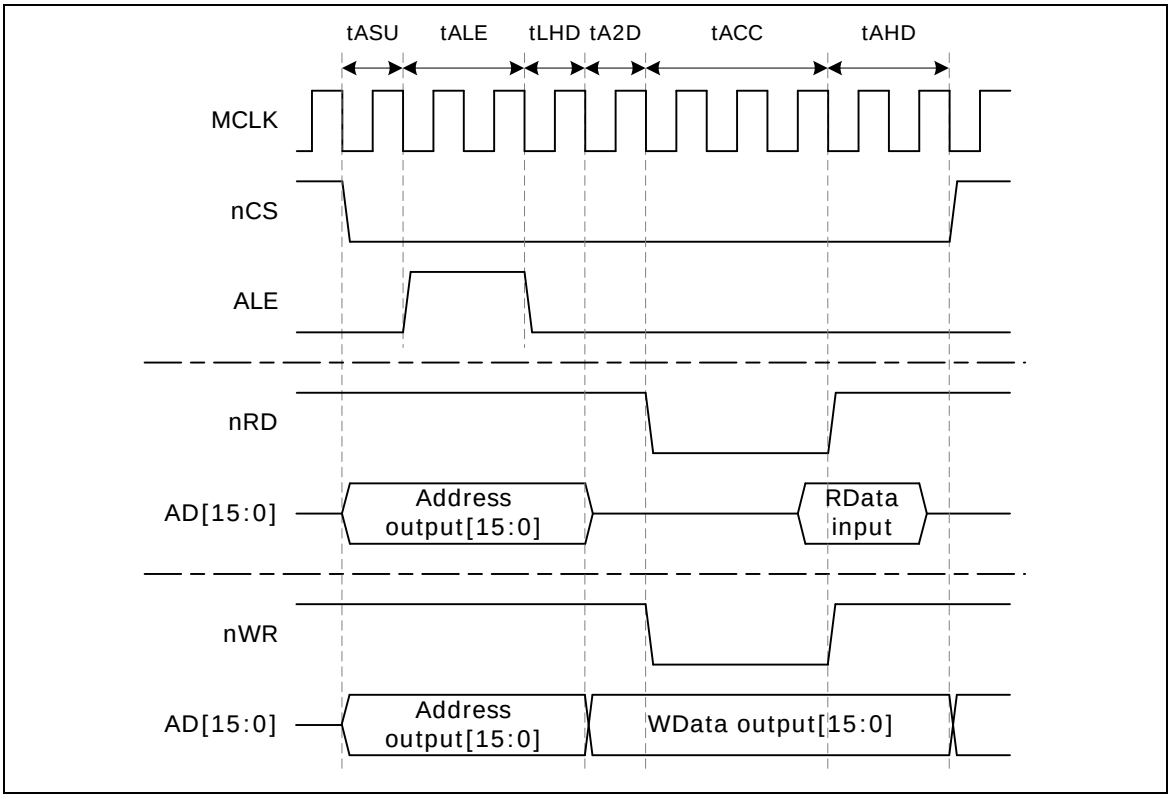


Figure 6.5-4 Timing Control Waveform for 16-bit Data Width

The figure above shows an example of setting 16-bit data width for EBI application. In this example, AD0~AD15 are used to be address[15:0] and data[15:0]. When ALE signal asserts to high, AD0~AD15 are the address output. After address is latched (tLHD), ALE signal asserts to low and the AD bus changes to high impedance to wait device output data in read access operation, or it is used to be write data output.

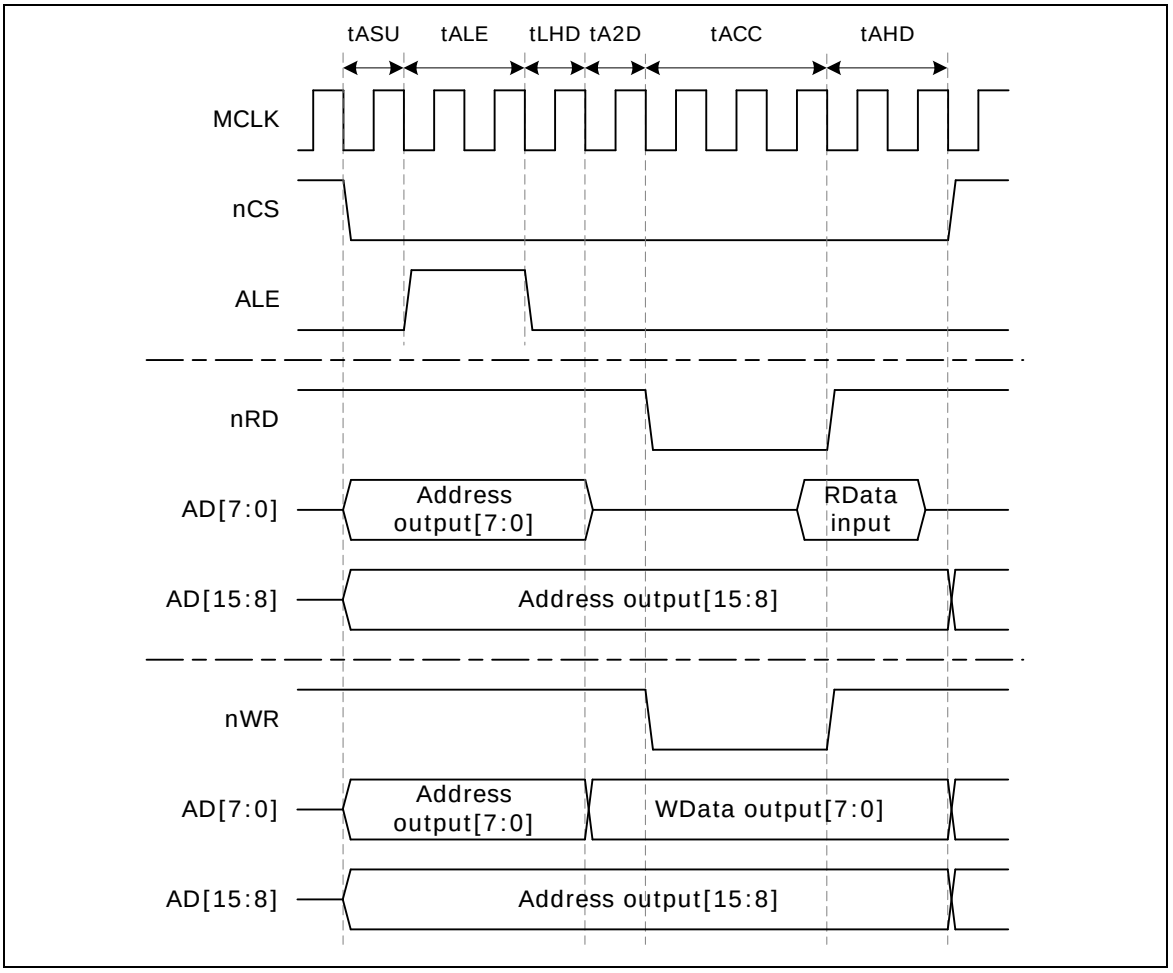


Figure 6.5-5 Timing Control Waveform for 8-bit Data Width

The figure above shows an example of setting 8-bit data width for EBI application. The difference between 8-bit and 16-bit data width is AD8~AD15. In 8-bit data width setting, and AD8~AD15 are always the Address[15:8] output so that the external latch needs only 8-bit width.

Insert Idle Cycle

When EBI is accessing continuously, bus conflict may occur if the device access time is much longer compared with system clock frequency. The EBI controller supplies additional idle cycle to solve this problem. During idle cycle period, all control signals of EBI bus are inactive. The following figure shows idle cycle.

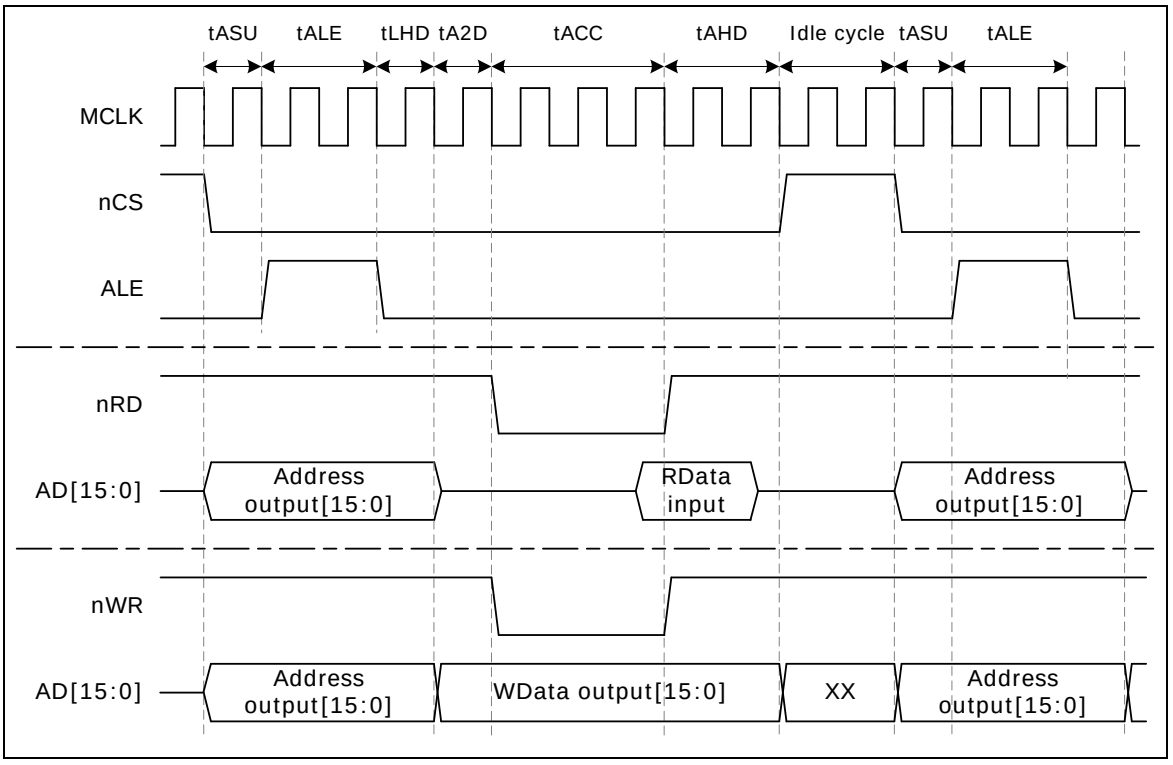


Figure 6.5-6 Timing Control Waveform for Insert Idle Cycle

There are two conditions that EBI can insert idle cycle by timing control:

1. After write access
2. After read access and before the next read access

By setting ExtIW2X[3:0] of EXTIME [15:12] and ExtIR2R[3:0] of the register EXTIME[27:24], the time of idle cycle can be specified from 0~15 MCLK.

6.5.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
EBI Base Address: **EBI_BA = 0x5001_0000**				
EBICON	EBI_BA+0x00	R/W	External Bus Interface General Control Register	0x0000_0000
EXTIME	EBI_BA+0x04	R/W	External Bus Interface Timing Control Register	0x0000_0000
EBICON2	EBI_BA+0x08	R/W	External Bus Interface General Control Register 2	0x0000_0000

6.5.6 Register Description

External Bus Interface General Control Register (EBICON)

Register	Offset	R/W	Description	Reset Value
EBICON	EBI_BA+0x00	R/W	External Bus Interface General Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**ExttALE**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**MCLKDIV**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**ExtBW16**	**ExtEN**

Bits	Description	
[31:19]	**Reserved**	Reserved
[18:16]	**ExttALE**	**Expand Time of ALE** This field is used for control the ALE pulse width (tALE) for latch the address tALE = (ExttALE+1)*MCLK
[15:11]	**Reserved**	Reserved
[10:8]	**MCLKDIV**	**External Output Clock Divider** The frequency of EBI output clock (MCLK) is controlled by MCLKDIV as follows table: **MCLKDIV** / **Output Clock (MCLK)** 000 / HCLK/1 001 / HCLK/2 010 / HCLK/4 011 / HCLK/8 100 / HCLK/16 101 / HCLK/32 Others / default **Note:** Default value of output clock is HCLK/1
[7:2]	**Reserved**	Reserved
[1]	**ExtBW16**	**EBI Data Width 16-bit/8-bit** This bit defines if the data bus is 8-bit or 16-bit. 1 = EBI data width is 16-bit

		0 = EBI data width is 8-bit
[0]	**ExtEN**	**EBI Enable** This bit is the functional enable bit for EBI. 1 = EBI function Enabled 0 = EBI function Disabled

External Bus Interface Timing Control Register (EXTIME)

Register	Offset	R/W	Description	Reset Value
EXTIME	EBI_BA+0x04	R/W	External Bus Interface Timing Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				ExtIR2R			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ExtIW2X				Reserved	ExttAHD		
7	6	5	4	3	2	1	0
ExttACC					Reserved		

Bits	Description	
[31:28]	**Reserved**	Reserved
[27:24]	**ExtIR2R**	**Idle State Cycle Between Read-Read** When read action is finished and the next action is going to read, idle state is inserted and nCS signal return to high if ExtIR2R is not zero. Idle state cycle = (ExtIR2R*MCLK)
[23:16]	**Reserved**	Reserved
[15:12]	**ExtIW2X**	**Idle State Cycle After Write** When write action is finished, idle state is inserted and nCS signal return to high if ExtIW2X is not zero. Idle state cycle = (ExtIW2X*MCLK)
[11]	**Reserved**	Reserved
[10:8]	**ExttAHD**	**EBI Data Access Hold Time** ExttAHD defines data access hold time (tAHD). tAHD = (ExttAHD +1) * MCLK
[7:3]	**ExttACC**	**EBI Data Access Time** ExttACC defines data access time (tACC). tACC = (ExttACC +1) * MCLK
[2:0]	**Reserved**	Reserved

External Bus Interface Control Register 2 (EBICON2)

Register	Offset	R/W	Description	Reset Value
EBICON2	EBI_BA+0x08	R/W	External Bus Interface General Control Register 2	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved
[2]	**WAHD_OFF**	**Access Hold Time Disable Control When Write** 0 = tAHD is controlled by ExttAHD when write through EBI 1 = No tAHD when write through EBI
[1]	**RAHD_OFF**	**Access Hold Time Disable Control When Read** 0 = tAHD is controlled by ExttAHD when read through EBI 1 = No tAHD when read through EBI
[0]	**WBUFF_EN**	**EBI Write Buffer Enable** 0 = EBI write buffer disable 1 = EBI write buffer enable

6.6 General Purpose I/O (GPIO)

6.6.1 Overview

The NuMicro® NUC029LEE/NUC029SEE has up to 45 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 45 pins are arranged in 5 ports named as GPIOA, GPIOB, GPIOC, GPIOE and GPIOF. The GPIOA/B/C/E port has the maximum of 15 pins and GPIOF port has the maximum of 2 pins. Each of the 45 pins is independent and has the corresponding register bits to control the pin mode function and data.

The I/O type of each of I/O pins can be configured by software individually as input, output, open-drain or Quasi-bidirectional mode. After reset, the I/O mode of all pins are depending on Config0[10] setting. In Quasi-bidirectional mode, I/O pin has a very weak individual pull-up resistor which is about 110~300 KΩ for V_{DD} is from 5.0 V to 2.5 V.

6.6.2 Features

- Four I/O modes:
  - Quasi-bidirectional
  - Push-Pull output
  - Open-Drain output
  - Input only with high impendence
- TTL/Schmitt trigger input selectable by GPx_TYPE[15:0] in GPx_MFP[31:16]
- I/O pin configured as interrupt source with edge/level setting
- Configurable default I/O mode of all pins after reset by Config0[10] setting
  - If Config[10] is 0, all GPIO pins in input tri-state mode after chip reset
  - If Config[10] is 1, all GPIO pins in Quasi-bidirectional mode after chip reset
- I/O pin internal pull-up resistor enabled only in Quasi-bidirectional I/O mode
- Enabling the pin interrupt function will also enable the pin wake-up function.

6.6.3 Basic Configuration

The GPIO pin functions are configured in GPA_MFP, GPB_MFP, GPC_MFP, GPE_MFP, ALT_MFP, ALT_MFP1 and ALT_MFP2 registers.

6.6.4 Functional Description

6.6.4.1 Input Mode Explanation

Set GPIOx_PMD (PMDn[1:0]) to 00b as the GPIOx port [n] pin is in Input mode and the I/O pin is in tri-state (high impedance) without output drive capability. The GPIOx_PIN value reflects the status of the corresponding port pins.

6.6.4.2 Push-pull Output Mode Explanation

Set GPIOx_PMD (PMDn[1:0]) to 01b as the GPIOx port [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 bit [n] of GPIOx_DOUT is driven on the pin.

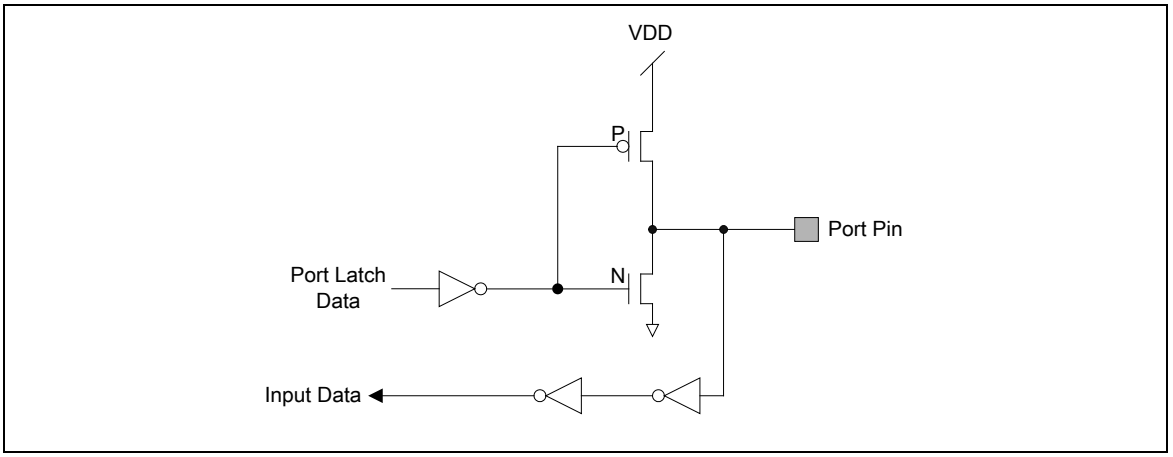


Figure 6.6-1 Push-Pull Output

6.6.4.3 Open-drain Output Mode Explanation

Set GPIOx_PMD (PMDn[1:0]) to 10b as the GPIOx port [n] pin is in Open-drain mode and the digital output function of I/O pin supports only sink current capability, an additional pull-up resistor is needed for driving high state. If the bit value in the corresponding bit [n] of GPIOx_DOUT is 0, the pin drive a “low” output on the pin. If the bit value in the corresponding bit [n] of GPIOx_DOUT is 1, the pin output drives high that is controlled by external pull-up resistor.

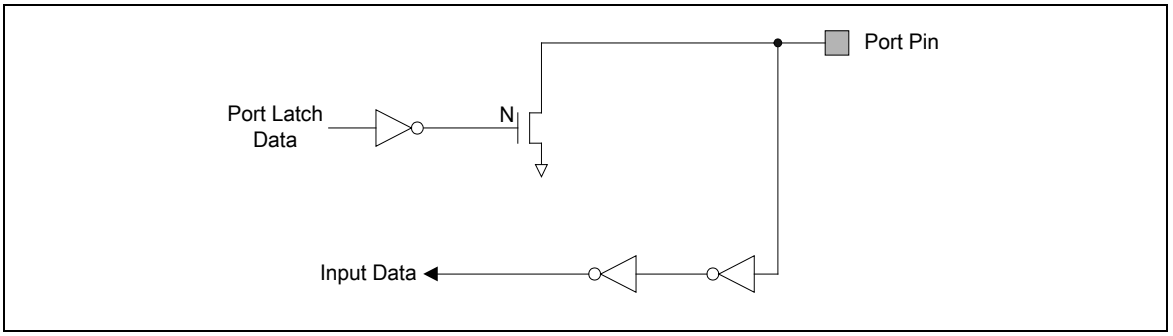


Figure 6.6-2 Open-Drain Output

6.6.4.4 Quasi-bidirectional Mode Explanation

Set GPIOx_PMD (PMDn[1:0]) to 11b as the GPIOx port [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 of uA. Before the digital input function is performed the corresponding bit in GPIOx_DOUT 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 bit [n] of GPIOx_DOUT is 0, the pin drive a “low” output on the pin. If the bit value in the corresponding bit [n] of GPIOx_DOUT is 1, the pin will check the pin value. If pin value is high, no action takes. If pin state is low, then pin will drive strong high with 2 clock cycles on the pin and then disable the strong output drive and then the pin status is control by internal pull-up resistor. Note that the source current capability in Quasi-bidirectional mode is only about 200 uA to 30 uA for V_{DD} is form 5.0 V to 2.5 V.

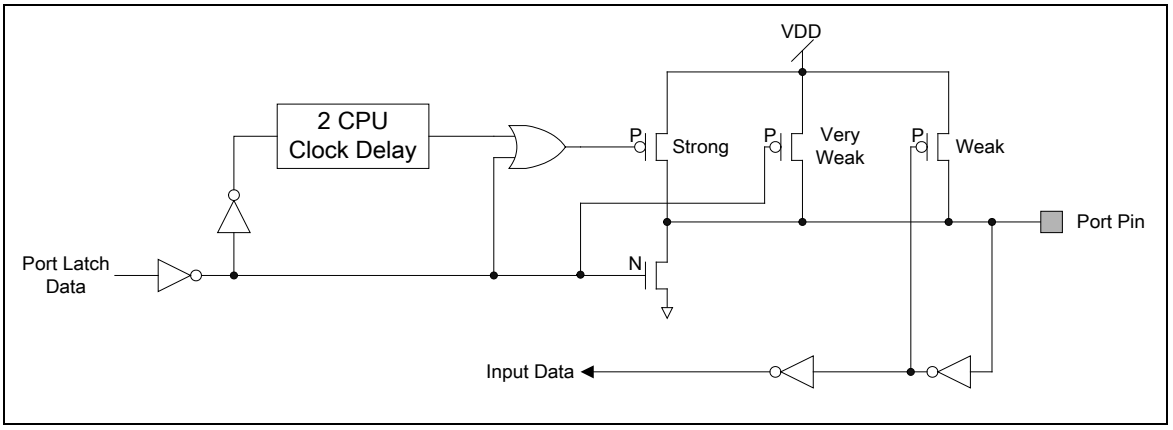


Figure 6.6-3 Quasi-bidirectional I/O Mode

6.6.4.5 GPIO Interrupt and Wake-up Function

Each GPIO pin can be set as chip interrupt source by setting correlative GPIOx_IEN bit and GPIOx_IMD. There are four types of interrupt condition can be selected: low level trigger, high level trigger, falling edge trigger and rising edge trigger. For edge trigger condition, user can enable input signal de-bounce function to prevent unexpected interrupt happened which caused by noise. The de-bounce clock source and sampling cycle can be set through DEBOUNCE register.

The GPIO can also be the chip wake-up source when chip enters Idle mode or Power-down mode. The setting of wake-up trigger condition is the same as GPIO interrupt trigger, but there is one thing need to be noticed if using GPIO as chip wake-up source

- **To ensure the I/O status before enter into Power-down mode**

When using toggle GPIO to wake-up system, user must make sure the I/O status before entering Idle mode or Power-down mode according to the relative wake-up settings.

For example, if configuring the wake-up event occurred by I/O rising edge/high level trigger, user must make sure the I/O status of specified pin is at low level before entering to Idle/Power-down mode; and if configure I/O falling edge/low level trigger to trigger a wake-up event, user must make sure the I/O status of specified pin is at high level before entering to Power-down mode.

6.6.5 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 = 0x5000_4000**				
GPIOA_PMD	GPIO_BA+0x000	R/W	GPIO Port A Pin I/O Mode Control Register	0xXXXX_XXXX
GPIOA_OFFD	GPIO_BA+0x004	R/W	GPIO Port A Pin Digital Input Path Disable Register	0x0000_0000
GPIOA_DOUT	GPIO_BA+0x008	R/W	GPIO Port A Data Output Value Register	0x0000_FFFF
GPIOA_DMASK	GPIO_BA+0x00C	R/W	GPIO Port A Data Output Write Mask Register	0x0000_0000
GPIOA_PIN	GPIO_BA+0x010	R	GPIO Port A Pin Value Register	0x0000_XXXX
GPIOA_DBEN	GPIO_BA+0x014	R/W	GPIO Port A De-bounce Enable Register	0x0000_0000
GPIOA_IMD	GPIO_BA+0x018	R/W	GPIO Port A Interrupt Mode Control Register	0x0000_0000
GPIOA_IEN	GPIO_BA+0x01C	R/W	GPIO Port A Interrupt Enable Register	0x0000_0000
GPIOA_ISRC	GPIO_BA+0x020	R/W	GPIO Port A Interrupt Source Flag Register	0x0000_0000
GPIOB_PMD	GPIO_BA+0x040	R/W	GPIO Port B Pin I/O Mode Control Register	0xXXXX_XXXX
GPIOB_OFFD	GPIO_BA+0x044	R/W	GPIO Port B Pin Digital Input Path Disable Control Register	0x0000_0000
GPIOB_DOUT	GPIO_BA+0x048	R/W	GPIO Port B Data Output Value Register	0x0000_FFFF
GPIOB_DMASK	GPIO_BA+0x04C	R/W	GPIO Port B Data Output Write Mask Register	0x0000_0000
GPIOB_PIN	GPIO_BA+0x050	R	GPIO Port B Pin Value Register	0x0000_XXXX
GPIOB_DBEN	GPIO_BA+0x054	R/W	GPIO Port B De-bounce Enable Register	0x0000_0000
GPIOB_IMD	GPIO_BA+0x058	R/W	GPIO Port B Interrupt Mode Control Register	0x0000_0000
GPIOB_IEN	GPIO_BA+0x05C	R/W	GPIO Port B Interrupt Enable Register	0x0000_0000
GPIOB_ISRC	GPIO_BA+0x060	R/W	GPIO Port B Interrupt Source Flag Register	0x0000_0000
GPIOC_PMD	GPIO_BA+0x080	R/W	GPIO Port C Pin I/O Mode Control Register	0xXXXX_XXXX
GPIOC_OFFD	GPIO_BA+0x084	R/W	GPIO Port C Pin Digital Input Path Disable Control Register	0x0000_0000
GPIOC_DOUT	GPIO_BA+0x088	R/W	GPIO Port C Data Output Value Register	0x0000_FFFF
GPIOC_DMASK	GPIO_BA+0x08C	R/W	GPIO Port C Data Output Write Mask Register	0x0000_0000
GPIOC_PIN	GPIO_BA+0x090	R	GPIO Port C Pin Value Register	0x0000_XXXX
GPIOC_DBEN	GPIO_BA+0x094	R/W	GPIO Port C De-bounce Enable Register	0x0000_0000
GPIOC_IMD	GPIO_BA+0x098	R/W	GPIO Port C Interrupt Mode Control Register	0x0000_0000
GPIOC_IEN	GPIO_BA+0x09C	R/W	GPIO Port C Interrupt Enable Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
GPIOC_ISRC	GPIO_BA+0x0A0	R/W	GPIO Port C Interrupt Source Flag Register	0x0000_0000
GPIOD_PMD	GPIO_BA+0x0C0	R/W	GPIO Port D Pin I/O Mode Control Register	0xXXXX_XXXX
GPIOD_OFFD	GPIO_BA+0x0C4	R/W	GPIO Port D Pin Digital Input Path Disable Control Register	0x0000_0000
GPIOD_DOUT	GPIO_BA+0x0C8	R/W	GPIO Port D Data Output Value Register	0x0000_FFFF
GPIOD_DMASK	GPIO_BA+0x0CC	R/W	GPIO Port D Data Output Write Mask Register	0x0000_0000
GPIOE_PMD	GPIO_BA+0x100	R/W	GPIO Port E Pin I/O Mode Control Register	0xXXXX_XXXX
GPIOE_OFFD	GPIO_BA+0x104	R/W	GPIO Port E Pin Digital Input Path Disable Control Register	0x0000_0000
GPIOE_DOUT	GPIO_BA+0x108	R/W	GPIO Port E Data Output Value Register	0x0000_FFFF
GPIOE_DMASK	GPIO_BA+0x10C	R/W	GPIO Port E Data Output Write Mask Register	0x0000_0000
GPIOE_PIN	GPIO_BA+0x110	R	GPIO Port E Pin Value Register	0x0000_XXXX
GPIOE_DBEN	GPIO_BA+0x114	R/W	GPIO Port E De-bounce Enable Register	0x0000_0000
GPIOE_IMD	GPIO_BA+0x118	R/W	GPIO Port E Interrupt Mode Control Register	0x0000_0000
GPIOE_IEN	GPIO_BA+0x11C	R/W	GPIO Port E Interrupt Enable Register	0x0000_0000
GPIOE_ISRC	GPIO_BA+0x120	R/W	GPIO Port E Interrupt Source Flag Register	0x0000_0000
GPIOF_PMD	GPIO_BA+0x140	R/W	GPIO Port F Pin I/O Mode Control Register	0x0000_00XX
GPIOF_OFFD	GPIO_BA+0x144	R/W	GPIO Port F Pin Digital Input Path Disable Control Register	0x0000_0000
GPIOF_DOUT	GPIO_BA+0x148	R/W	GPIO Port F Data Output Value Register	0x0000_000F
GPIOF_DMASK	GPIO_BA+0x14C	R/W	GPIO Port F Data Output Write Mask Register	0x0000_0000
GPIOF_PIN	GPIO_BA+0x150	R	GPIO Port F Pin Value Register	0x0000_000X
GPIOF_DBEN	GPIO_BA+0x154	R/W	GPIO Port F De-bounce Enable Register	0x0000_0000
GPIOF_IMD	GPIO_BA+0x158	R/W	GPIO Port F Interrupt Mode Control Register	0x0000_0000
GPIOF_IEN	GPIO_BA+0x15C	R/W	GPIO Port F Interrupt Enable Register	0x0000_0000
GPIOF_ISRC	GPIO_BA+0x160	R/W	GPIO Port F Interrupt Source Flag Register	0x0000_0000
DBNCECON	GPIO_BA+0x180	R/W	External Interrupt De-bounce Control Register	0x0000_0020
PAn_PDIO n=0,1..15	GPIO_BA+0x200 + 0x04 * n	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO n=0,1..15	GPIO_BA+0x240 + 0x04 * n	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO n=0,1..15	GPIO_BA+0x280 + 0x04 * n	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PEn_PDIO n=0,1..15	GPIO_BA+0x300 + 0x04 * n	R/W	GPIO PE.n Pin Data Input/Output Register	0x0000_000X

Register	Offset	R/W	Description	Reset Value
PFn_PDIO n=0,1..3	GPIO_BA+0x340 + 0x04 * n	R/W	GPIO PF.n Pin Data Input/Output Register	0x0000_000X

6.6.6 Register Description

GPIO Port [A/B/C/E/F] Pin I/O Mode Control Register (GPIOx_PMD)

Register	Offset	R/W	Description	Reset Value
GPIOA_PMD	GPIO_BA+0x000	R/W	GPIO Port A Pin I/O Mode Control Register	0xXXXX_XXXX
GPIOB_PMD	GPIO_BA+0x040	R/W	GPIO Port B Pin I/O Mode Control Register	0xXXXX_XXXX
GPIOC_PMD	GPIO_BA+0x080	R/W	GPIO Port C Pin I/O Mode Control Register	0xXXXX_XXXX
GPIOE_PMD	GPIO_BA+0x100	R/W	GPIO Port E Pin I/O Mode Control Register	0xXXXX_XXXX
GPIOF_PMD	GPIO_BA+0x140	R/W	GPIO Port F Pin I/O Mode Control Register	0x0000_00XX

31	30	29	28	27	26	25	24
PMD15		PMD14		PMD13		PMD12	
23	22	21	20	19	18	17	16
PMD11		PMD10		PMD9		PMD8	
15	14	13	12	11	10	9	8
PMD7		PMD6		PMD5		PMD4	
7	6	5	4	3	2	1	0
PMD3		PMD2		PMD1		PMD0	

Bits	Description	
[2n+1:2n] n=0,1..15	**PMDn**	**GPIOx I/O Pin[N] Mode Control** Determine each I/O mode of GPIOx pins. 00 = GPIO port [n] pin is in Input mode. 01 = GPIO port [n] pin is in Push-pull Output mode. 10 = GPIO port [n] pin is in Open-drain Output mode. 11 = GPIO port [n] pin is in Quasi-bidirectional mode. **Note1:** The initial value of this field is defined by CIOINI (CONFIG0[10]). If CIOINI is set to 1, the default value is 0xFFFF_FFFF and all pins will be Quasi-bidirectional mode after chip is powered on. If CIOINI is cleared to 0, the default value is 0x0000_0000 and all pins will be input only mode after chip is powered on. **Note2:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note3:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.

GPIO Port [A/B/C/E/F] Pin Digital Input Path Disable Register (GPIOx_OFFD)

Register	Offset	R/W	Description	Reset Value
GPIOA_OFFD	GPIO_BA+0x004	R/W	GPIO Port A Pin Digital Input Path Disable Register	0x0000_0000
GPIOB_OFFD	GPIO_BA+0x044	R/W	GPIO Port B Pin Digital Input Path Disable Register	0x0000_0000
GPIOC_OFFD	GPIO_BA+0x084	R/W	GPIO Port C Pin Digital Input Path Disable Register	0x0000_0000
GPIOE_OFFD	GPIO_BA+0x104	R/W	GPIO Port E Pin Digital Input Path Disable Register	0x0000_0000
GPIOF_OFFD	GPIO_BA+0x144	R/W	GPIO Port F Pin Digital Input Path Disable Register	0x0000_0000

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

Bits	Description	
[31:16]	**OFFD**	**GPIOx Pin[N] Digital Input Path Disable Bit** Each of these bits is used to control if the digital input path of corresponding GPIO pin is disabled. If input is analog signal, users can disable GPIO digital input path to avoid current leakage. 0 = I/O digital input path Enabled. 1 = I/O digital input path Disabled (digital input tied to low). **Note1:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note2:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.
[15:0]	**Reserved**	Reserved.

GPIO Port [A/B/C/E/F] Data Output Value Register (GPIOx_DOUT)

Register	Offset	R/W	Description	Reset Value
GPIOA_DOUT	GPIO_BA+0x008	R/W	GPIO Port A Data Output Value Register	0x0000_FFFF
GPIOB_DOUT	GPIO_BA+0x048	R/W	GPIO Port B Data Output Value Register	0x0000_FFFF
GPIOC_DOUT	GPIO_BA+0x088	R/W	GPIO Port C Data Output Value Register	0x0000_FFFF
GPIOE_DOUT	GPIO_BA+0x108	R/W	GPIO Port E Data Output Value Register	0x0000_FFFF
GPIOF_DOUT	GPIO_BA+0x148	R/W	GPIO Port F Data Output Value 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**
DOUT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DOUT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n = 0,1..15	**DOUT[n]**	**GPIOx Pin[N] Output Value** Each of these bits controls the status of a GPIO pin when the GPIO pin is configured as Push-pull output, open-drain output or quasi-bidirectional mode. 0 = GPIO port [A/B/C/E/F] Pin[n] will drive Low if the GPIO pin is configured as Push-pull output, Open-drain output or Quasi-bidirectional mode. 1 = GPIO port [A/B/C/E/F] Pin[n] will drive High if the GPIO pin is configured as Push-pull output or Quasi-bidirectional mode. **Note1:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note2:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.

GPIO Port [A/B/C/E/F] Data Output Write Mask Register (GPIOx _DMASK)

Register	Offset	R/W	Description	Reset Value
GPIOA_DMASK	GPIO_BA+0x00C	R/W	GPIO Port A Data Output Write Mask Register	0x0000_0000
GPIOB_DMASK	GPIO_BA+0x04C	R/W	GPIO Port B Data Output Write Mask Register	0x0000_0000
GPIOC_DMASK	GPIO_BA+0x08C	R/W	GPIO Port C Data Output Write Mask Register	0x0000_0000
GPIOE_DMASK	GPIO_BA+0x10C	R/W	GPIO Port E Data Output Write Mask Register	0x0000_0000
GPIOF_DMASK	GPIO_BA+0x14C	R/W	GPIO Port F Data Output Write Mask Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n = 0,1..15	**DMASK[n]**	**Port [A/B/C/E/F] Data Output Write Mask** These bits are used to protect the corresponding register of GPIOx_DOUT[n] bit. When the DMASK[n] bit is set to 1, the corresponding GPIOx_DOUT[n] bit is protected. If the write signal is masked, write data to the protect bit is ignored 0 = Corresponding GPIOx_DOUT[n] bit can be updated. 1 = Corresponding GPIOx_DOUT[n] bit protected. **Note1:** This function only protects the corresponding GPIOx_DOUT[n] bit, and will not protect the corresponding bit control register (PAn_PDIO, PBn_PDIO, PCn_PDIO, PEn_PDIO and PFn_PDIO). **Note2:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note3:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.

GPIO Port [A/B/C/E/F] Pin Value (GPIOx _PIN)

Register	Offset	R/W	Description	Reset Value
GPIOA_PIN	GPIO_BA+0x010	R	GPIO Port A Pin Value	0x0000_XXXX
GPIOB_PIN	GPIO_BA+0x050	R	GPIO Port B Pin Value	0x0000_XXXX
GPIOC_PIN	GPIO_BA+0x090	R	GPIO Port C Pin Value	0x0000_XXXX
GPIOE_PIN	GPIO_BA+0x110	R	GPIO Port E Pin Value	0x0000_XXXX
GPIOF_PIN	GPIO_BA+0x150	R	GPIO Port F Pin Value	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**
PIN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PIN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n = 0,1..15	**PIN[n]**	**Port [A/B/C/E/F] Pin Values** Each bit of the register reflects the actual status of the respective GPIO pin. If the bit is 1, it indicates the corresponding pin status is high, else the pin status is low **Note1:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note2:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.

GPIO Port [A/B/C/E/F] De-bounce Enable Register (GPIOx _DBEN)

Register	Offset	R/W	Description	Reset Value
GPIOA_DBEN	GPIO_BA+0x014	R/W	GPIO Port A De-bounce Enable Register	0x0000_0000
GPIOB_DBEN	GPIO_BA+0x054	R/W	GPIO Port B De-bounce Enable Register	0x0000_0000
GPIOC_DBEN	GPIO_BA+0x094	R/W	GPIO Port C De-bounce Enable Register	0x0000_0000
GPIOE_DBEN	GPIO_BA+0x114	R/W	GPIO Port E De-bounce Enable Register	0x0000_0000
GPIOF_DBEN	GPIO_BA+0x154	R/W	GPIO Port F De-bounce 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**
DBEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBEN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n = 0,1..15	**DBEN[n]**	**Port [A/B/C/E/F] Input Signal De-Bounce Enable Bit** DBEN[n] 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 DBNCECON[4], one de-bounce sample cycle period is controlled by DBNCECON[3:0] 0 = Bit[n] de-bounce function Disabled. 1 = Bit[n] de-bounce function Enabled. The de-bounce function is valid only for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored. **Note1:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note2:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.

GPIO Port [A/B/C/E/F] Interrupt Mode Control Registrt (GPIOx _IMD)

Register	Offset	R/W	Description	Reset Value
GPIOA_IMD	GPIO_BA+0x018	R/W	GPIO Port A Interrupt Mode Control Register	0x0000_0000
GPIOB_IMD	GPIO_BA+0x058	R/W	GPIO Port B Interrupt Mode Control Register	0x0000_0000
GPIOC_IMD	GPIO_BA+0x098	R/W	GPIO Port C Interrupt Mode Control Register	0x0000_0000
GPIOE_IMD	GPIO_BA+0x118	R/W	GPIO Port E Interrupt Mode Control Register	0x0000_0000
GPIOF_IMD	GPIO_BA+0x158	R/W	GPIO Port F Interrupt Mode Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n = 0,1..15	**IMD[n]**	**Port [A/B/C/E/F] Edge Or Level Detection Interrupt Control** IMD[n] is used to control the 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 GPIOx_IEN. If both levels to trigger interrupt are set, the setting is ignored 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 ignored. **Note1:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note2:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.

GPIO Port [A/B/C/E/F] Interrupt Enable Register (GPIOx _IEN)

Register	Offset	R/W	Description	Reset Value
GPIOA_IEN	GPIO_BA+0x01C	R/W	GPIO Port A Interrupt Enable Register	0x0000_0000
GPIOB_IEN	GPIO_BA+0x05C	R/W	GPIO Port B Interrupt Enable Register	0x0000_0000
GPIOC_IEN	GPIO_BA+0x09C	R/W	GPIO Port C Interrupt Enable Register	0x0000_0000
GPIOE_IEN	GPIO_BA+0x11C	R/W	GPIO Port E Interrupt Enable Register	0x0000_0000
GPIOF_IEN	GPIO_BA+0x15C	R/W	GPIO Port F Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[n+16] n = 0,1..15	**IR_EN[n]**	**Port [A/B/C/E/F] Interrupt Enabled By Input Rising Edge Or Input Level High** IR_EN[n] used to enable the interrupt for each of the corresponding input GPIO_PIN[n]. Set bit to 1 also enable the pin wake-up function When setting the IR_EN[n] bit to 1: If the interrupt is level trigger, the input PIN[n] state at level “high” will generate the interrupt. If the interrupt is edge trigger, the input PIN[n] state change from “low-to-high” will generate the interrupt. 0 = PIN[n] level-high or low-to-high interrupt Disabled. 1 = PIN[n] level-high or low-to-high interrupt Enabled. **Note1:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note2:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.
[n] n = 0,1..15	**IF_EN[n]**	**Port [A/B/C/E/F] Interrupt Enabled By Input Falling Edge Or Input Level Low** IF_EN[n] is used to enable the interrupt for each of the corresponding input GPIO_PIN[n]. Set bit to 1 also enable the pin wake-up function When setting the IF_EN[n] bit to 1: If the interrupt is level trigger, the input PIN[n] state at level “low” will generate the interrupt. If the interrupt is edge trigger, the input PIN[n] state change from “high-to-low” will generate the interrupt. 0 = PIN[n] state low-level or high-to-low change interrupt Disabled. 1 = PIN[n] state low-level or high-to-low change interrupt Enabled. **Note1:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note2:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.

GPIO Port [A/B/C/E/F] Interrupt Source Flag Register (GPIOx _ISRC)

Register	Offset	R/W	Description	Reset Value
GPIOA_ISRC	GPIO_BA+0x020	R/W	GPIO Port A Interrupt Source Flag Register	0x0000_0000
GPIOB_ISRC	GPIO_BA+0x060	R/W	GPIO Port B Interrupt Source Flag Register	0x0000_0000
GPIOC_ISRC	GPIO_BA+0x0A0	R/W	GPIO Port C Interrupt Source Flag Register	0x0000_0000
GPIOE_ISRC	GPIO_BA+0x120	R/W	GPIO Port E Interrupt Source Flag Register	0x0000_0000
GPIOF_ISRC	GPIO_BA+0x160	R/W	GPIO Port F Interrupt Source 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**
ISRC							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISRC							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n = 0,1..15	**ISRC[n]**	**Port [A/B/C/E/F] Interrupt Source Flag** Read : 0 = No interrupt at GPIOx[n]. 1 = GPIOx[n] generates an interrupt. Write : 0= No action. 1= Clear the corresponding pending interrupt. **Note1:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note2:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.

Interrupt De-bounce Cycle Control Register (DBNCECON)

Register	Offset	R/W	Description	Reset Value
DBNCECON	GPIO_BA+0x180	R/W	External Interrupt De-bounce Control Register	0x0000_0020

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

Bits	Description	
[5]	ICLK_ON	**Interrupt Clock On Mode** 0 = Edge detection circuit is active only if I/O pin corresponding GPIOx_IEN bit is set to 1. 1 = All I/O pins edge detection circuit is always active after reset. It is recommended to disable this bit to save system power if no special application concern.
[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 internal 10 kHz low speed oscillator.
[3:0]	DBCLKSEL	**De-Bounce Sampling Cycle Selection**

DBCLKSEL	Description
0	Sample interrupt input once per 1 clocks
1	Sample interrupt input once per 2 clocks
2	Sample interrupt input once per 4 clocks
3	Sample interrupt input once per 8 clocks
4	Sample interrupt input once per 16 clocks
5	Sample interrupt input once per 32 clocks
6	Sample interrupt input once per 64 clocks
7	Sample interrupt input once per 128 clocks
8	Sample interrupt input once per 256 clocks
9	Sample interrupt input once per 2*256 clocks
10	Sample interrupt input once per 4*256clocks
11	Sample interrupt input once per 8*256 clocks
12	Sample interrupt input once per 16*256 clocks

		13	Sample interrupt input once per 32*256 clocks	
		14	Sample interrupt input once per 64*256 clocks	
		15	Sample interrupt input once per 128*256 clocks	

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

Register	Offset	R/W	Description	Reset Value
PAn_PDIO **n=0,1..6,8..15**	GPIO_BA+0x200 + 0x04 * n	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO **n=0,1..11,13..15**	GPIO_BA+0x240 + 0x04 * n	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO **n=0,1..3,6..11,14,15**	GPIO_BA+0x280 + 0x04 * n	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PEn_PDIO **n=5**	GPIO_BA+0x300 + 0x04 * n	R/W	GPIO PE.n Pin Data Input/Output Register	0x0000_000X
PFn_PDIO **n=0,1**	GPIO_BA+0x340 + 0x04 * n	R/W	GPIO PF.n Pin Data Input/Output Register	0x0000_000X

Note: x = A/B/C/E/F and n = 0~15

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

Bits	Description	
[0]	**Pxn_PDIO**	**GPIO Px.N Pin Data Input/Output** Write 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 GPIOA_DOUT[0], read PA0_PDIO will return the value of GPIOA_PIN[0] **Note1:** The write operation will not be affected by register GPIOx_DMASK **Note2:** Max. n = 15 for GPIOA/GPIOB/GPIOC; n = 5 for GPIOE; Max. n = 1 for GPIOF. **Note3:** The PA.7, PB.12, PC.4, PC.5, PC.12, PC.13 pin is ignored.

6.7 PDMA Controller (PDMA)

6.7.1 Overview

The NuMicro® NUC029LEE/NUC029SEE DMA contains nine-channel peripheral direct memory access (PDMA) controller and a cyclic redundancy check (CRC) generator.

The PDMA that transfers data to and from memory or transfer data to and from APB devices. For PDMA channel (PDMA CH0~CH8), there is one-word buffer as transfer buffer between the Peripherals APB devices and Memory. Software can stop the PDMA operation by disable PDMA PDMACEN (PDMA_CSRx[0]). The CPU can recognize the completion of a PDMA operation by software polling or when it receives an internal PDMA interrupt. The PDMA controller can increase source or destination address or fixed them as well.

The DMA controller contains a cyclic redundancy check (CRC) generator that can perform CRC calculation with programmable polynomial settings. The CRC engine supports CPU PIO mode and DMA transfer mode.

6.7.2 Features

- Supports nine PDMA channels and one CRC channel. Each PDMA channel can support a unidirectional transfer
- AMBA AHB master/slave interface compatible, for data transfer and register read/write
- Hardware round robin priority scheme. DMA channel 0 has the highest priority and channel 8 has the lowest priority
- PDMA operation
  - Peripheral-to-memory, memory-to-peripheral, and memory-to-memory transfer
  - Supports word/half-word/byte transfer data width from/to peripheral
  - Supports address direction: increment, fixed.
- Cyclic Redundancy Check (CRC)
  - 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: $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
  - Supports programmable CRC 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 CPU PIO mode or DMA transfer mode.
  - Supports the follows write data length in CPU PIO mode
    - 8-bit write mode (byte): 1-AHB clock cycle operation.

- 16-bit write mode (half-word): 2-AHB clock cycle operation.
- 32-bit write mode (word): 4-AHB clock cycle operation.

- Supports byte alignment transfer data length and word alignment transfer source address in CRC DMA mode.

6.7.3 Block Diagram

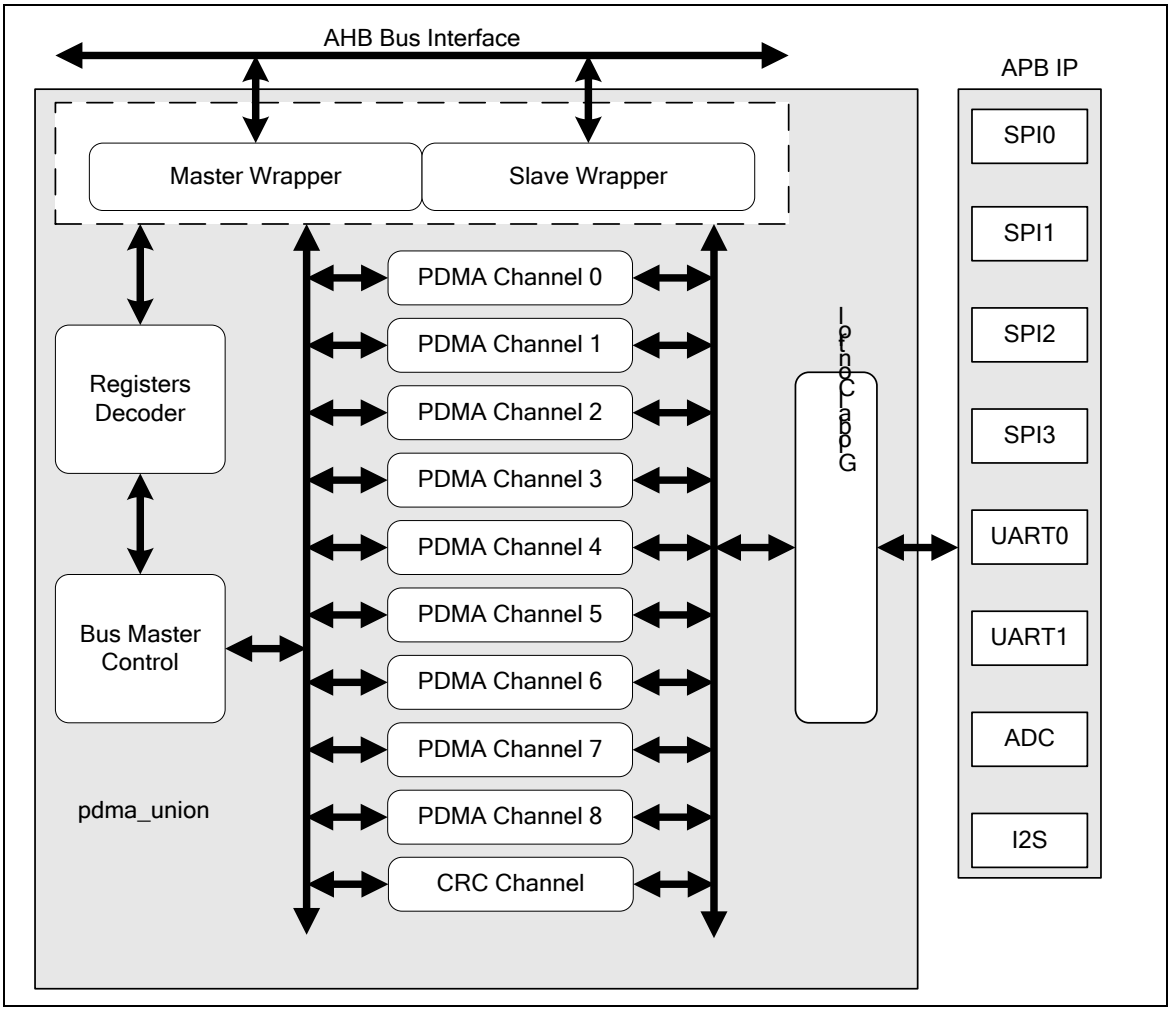


Figure 6.7-1 DMA Controller Block Diagram

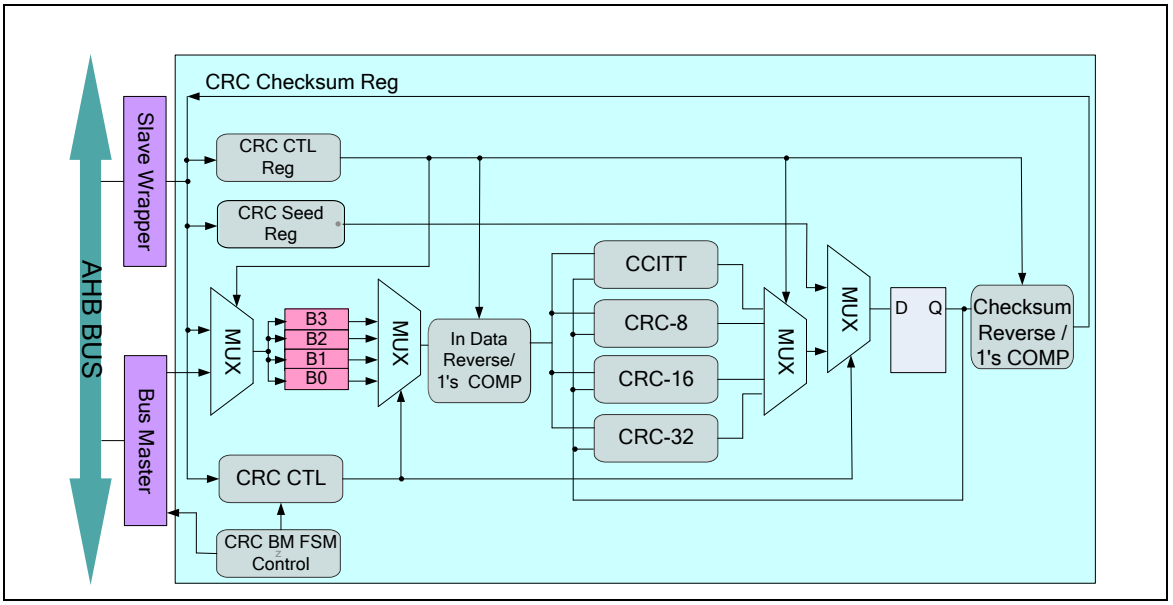


Figure 6.7-2 CRC Generator Block Diagram

6.7.4 Basic Configuration

The PDMA controller peripheral clock can be enabled in PDMA_EN (AHBCLK[1]).

6.7.5 Functional Description

The direct memory access (DMA) controller module transfers data from one address to another address, without CPU intervention. The DMA controller contains nine PDMA (Peripheral-to-Memory or Memory-to-Peripheral or Memory-to-Memory) channels and one CRC generator channel.

The CPU can recognize the completion of a DMA operation by software polling or when it receives an internal DMA interrupt.

6.7.5.1 PDMA

The DMA controller has nine channels PDMA (Peripheral-to-Memory or Memory-to-Peripheral or Memory-to-Memory). As to the source and destination address, the PDMA controller has two modes: increased and fixed.

Every PDMA channel behavior is not pre-defined, users must configure the channel service settings of PDMA_PDSSR0 and PDMA_PDSSR1 registers before starting the related PDMA channel.

Software must enable PDMA channel by setting PDMACEN (PDMA_CSRx[0]) bit and then write a valid source address to the PDMA_SARx register, a destination address to the PDMA_DARx register, and a transfer count to the PDMA_BCRx register. Next, trigger the TRIG_EN (PDMA_CSRx[23]). PDMA will continue the transfer until PDMA_CBCRx counts down to 0. The following sequence is a program sequence example.

- Enable PDMA peripheral clock by setting PDMA_EN (AHBCLK[1]) bit.

- Configure the channel service setting by setting PDMA_PDSSR0/ PDMA_PDSSR1 register.
- Configure PDMA_CSRx register:
  - Enable PDMA channel(PDMACEN (PDMA_CSRx[0]))
  - Set source/destination address direction(SAD_SEL (PDMA_CSRx[5:4]) / DAD_SEL (PDMA_CSRx[7:6]))
  - Configure PDMA mode selection(MODE_SEL (PDMA_CSRx[3:2]))
  - Configure peripheral transfer width selection (APB_TWS (PDMA_CSRx[20:19])).
- Configure source/destination address by setting PDMA_SARx/PDMA_DARx registers.
- Configure PDMA_transfer byte count by setting PDMA_BCRx register.
- Enable PDMA block transfer done interrupt by setting BLKD_IE(PDMA_IERx [1]). (optional)
- Enable PDMA NVIC by setting NVIC_ISER register bit 26 to "1". (optional)
- Enable PDMA read/write transfer by setting TRIG_EN (PDMA_CSRx[23]) bit.
- If PDMA block transfer done interrupt is generated, write "1" to BLKD_IF (PDMA_ISRx[1])by software to clear interrupt flag.
- Enable PDMA read/write transfer by setting the TRIG_EN (PDMA_CSRx[23])bit for the next block transfer.

If an error occurs during the PDMA operation, the channel stops unless software clears the error condition and sets the SW_RST (PDMA_CSRx[1])to reset the PDMA channel and set PDMACEN (PDMA_CSRx[0])and TRIG_EN (PDMA_CSRx[23]) bits field to start again.

In PDMA (Peripheral-to-Memory or Memory-to-Peripheral) mode, DMA can transfer data between the Peripherals (e.g. UART, SPI, ADC) and Memory.

6.7.5.2 CRC

The DMA controller contains a cyclic redundancy check (CRC) generator that can perform CRC calculation with programmable polynomial settings. The operation polynomial includes CRC-CCITT, CRC-8, CRC-16 and CRC-32; Software can choose the CRC operation polynomial mode by setting CRC polynomial mode (CRC_MODE (CRC_CTL[31:30])).

The CRC engine supports CPU PIO mode if CRC channel enable bit CRCCEN (CRC_CLT[0]) is 1, CRC DMA trigger enable bit TRIG_EN (CRC_CTL[23]) is 0 and DMA transfer mode if CRC channel enable bit CRCCEN (CRC_CLT[0]) is 1, CRC DMA trigger enable bit TRIG_EN (CRC_CTL[23]) is 1. The following sequence is a program sequence example.

Procedure when operating in CPU PIO mode:

- Enable CRC engine by setting CRC channel enable bit CRCCEN (CRC_CLT[0]) to 1.
- Set the transfer data format to enable write data order reverse (WDATA_RVS (CRC_CTL[24])), checksum reverse (CHECKSUM_RVS (CRC_CTL[25])), write data 1's complement (WDATA_COM (CRC_CTL[26])), checksum 1's complement (CHECKSUM_COM (CRC_CTL[27])), initial seed value in CRC seed register (CRC_SEED (CRC_SEED[31:0])) and select write data length by setting CPU write data length (CPU_WDLEN (CRC_CTL[29:28])).
- Set the CRC engine reset bit CRC_RST (CRC_CTL[1]) to 1 to load the initial seed value to CRC circuit but others contents of CRT_CTL register will not be cleared. This bit will be cleared automatically.

- Write data to CRC write data register (CRC_WDATA (CRC_WDATA[31:0])) to perform CRC calculation.
- Then, get the CRC checksum results by reading the CRC checksum register (CRC_CHECKSUM (CRC_CHECKSUM[31:0])).

Procedure when operating in CRC DMA mode:

- Enable CRC engine by setting CRC Channel Enable bit CRCCEN (CRC_CLT[0]) to 1.
- Set the transfer data format to enable write data order reverse (WDATA_RVS (CRC_CTL[24])), checksum reverse (CHECKSUM_RVS (CRC_CTL[25])), write data 1's complement (WDATA_COM (CRC_CTL[26])), checksum 1's complement (CHECKSUM_COM (CRC_CTL[27])) and initial seed value in CRC seed register (CRC_SEED (CRC_SEED[31:0])).
- Specify a valid source address (word alignment) and transfer counts by setting CRC DMA transfer source address register (CRC_DMASAR (CRC_DMASAR[31:0])) and CRC DMA transfer byte count register (CRC_DMABCR (CRC_DMABCR[15:0])).
- Set CRC DMA trigger enable bit TRIG_EN (CRC_CTL[23]) to 1 to perform CRC calculation.
- Wait CRC DMA transfer and check if CRC DMA transfer is done by the CRC DMA block transfer done interrupt flag (CRC_BLKD_IF (CRC_DMAISR[1])), and then get the CRC checksum results by reading the CRC checksum register (CRC_CHECKSUM (CRC_CHECKSUM[31:0])).

6.7.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMA_CHx_BA = 0x5000_8000 + 0x100 * x** **x=0,1 .. 8** **CRC_BA = 0x5000_8E00** **PDMA_GCR_BA = 0x5000_8F00**				
PDMA_CSRx **x=0,1 .. 8**	PDMA_CHx_BA+0x00	R/W	PDMA Channel x Control Register	0x0000_0000
PDMA_SARx **x=0,1 .. 8**	PDMA_CHx_BA+0x04	R/W	PDMA Channel x Source Address Register	0x0000_0000
PDMA_DARx **x=0,1 .. 8**	PDMA_CHx_BA+0x08	R/W	PDMA Channel x Destination Address Register	0x0000_0000
PDMA_BCRx **x=0,1 .. 8**	PDMA_CHx_BA+0x0C	R/W	PDMA Channel x Transfer Byte Count Register	0x0000_0000
PDMA_POINTx **x=0,1 .. 8**	PDMA_CHx_BA+0x10	R	PDMA Channel x Internal Buffer Pointer Register	0xXXXX_0000
PDMA_CSARx **x=0,1 .. 8**	PDMA_CHx_BA+0x14	R	PDMA Channel x Current Source Address Register	0x0000_0000
PDMA_CDARx **x=0,1 .. 8**	PDMA_CHx_BA+0x18	R	PDMA Channel x Current Destination Address Register	0x0000_0000
PDMA_CBCRx **x=0,1 .. 8**	PDMA_CHx_BA+0x1C	R	PDMA Channel x Current Transfer Byte Count Register	0x0000_0000
PDMA_IERx **x=0,1 .. 8**	PDMA_CHx_BA+0x20	R/W	PDMA Channel x Interrupt Enable Register	0x0000_0001
PDMA_ISRx **x=0,1 .. 8**	PDMA_CHx_BA+0x24	R/W	PDMA Channel x Interrupt Status Register	0x0000_0000
PDMA_SBUF0_Cx **x=0,1 .. 8**	PDMA_CHx_BA+0x80	R	PDMA Channel x Shared Buffer FIFO 0 Register	0x0000_0000
CRC_CTL	CRC_BA+0x00	R/W	CRC Control Register	0x2000_0000
CRC_DMASAR	CRC_BA+0x04	R/W	CRC DMA Source Address Register	0x0000_0000
CRC_DMABCR	CRC_BA+0x0C	R/W	CRC DMA Transfer Byte Count Register	0x0000_0000
CRC_DMACSAR	CRC_BA+0x14	R	CRC DMA Current Source Address Register	0x0000_0000
CRC_DMACBCR	CRC_BA+0x1C	R	CRC DMA Current Transfer Byte Count Register	0x0000_0000
CRC_DMAIER	CRC_BA+0x20	R/W	CRC DMA Interrupt Enable Register	0x0000_0001
CRC_DMAISR	CRC_BA+0x24	R/W	CRC DMA Interrupt Status Register	0x0000_0000

CRC_WDATA	CRC_BA+0x80	R/W	CRC Write Data Register	0x0000_0000
CRC_SEED	CRC_BA+0x84	R/W	CRC Seed Register	0xFFFF_FFFF
CRC_CHECKSUM	CRC_BA+0x88	R	CRC Checksum Register	0xFFFF_FFFF
PDMA_GCRCSR	PDMA_GCR_BA+0x00	R/W	PDMA Global Control Register	0x0000_0000
PDMA_PDSSR0	PDMA_GCR_BA+0x04	R/W	PDMA Service Selection Control Register 0	0xFFFF_FFFF
PDMA_PDSSR1	PDMA_GCR_BA+0x08	R/W	PDMA Service Selection Control Register 1	0xFFFF_FFFF
PDMA_GCRISR	PDMA_GCR_BA+0x0C	R	PDMA Global Interrupt Status Register	0x0000_0000

6.7.7 Register Description

PDMA Channel x Control Register (PDMA_CSRx)

Register	Offset	R/W	Description	Reset Value
PDMA_CSRx **x=0,1 .. 8**	PDMA_CHx_BA+0x00	R/W	PDMA Channel x Control Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**TRIG_EN**	**Trigger Enable Bit** 0 = No effect. 1 = PDMA data read or write transfer Enabled. **Note:** When PDMA transfer completed, this bit will be cleared automatically. If the bus error occurs, all PDMA transfer will be stopped. Software must reset all PDMA channel, and then trigger again.
[22:21]	**Reserved**	Reserved.
[20:19]	**APB_TWS**	**Peripheral Transfer Width Selection** 00 = One word (32-bit) is transferred for every PDMA operation. 01 = One byte (8-bit) is transferred for every PDMA operation. 10 = One half-word (16-bit) is transferred for every PDMA operation. 11 = Reserved. **Note:** This field is meaningful only when MODE_SEL (PDMA_CSRx[3:2]) is Peripheral to Memory mode (Peripheral-to-Memory) or Memory to Peripheral mode (Memory-to-Peripheral).
[18:8]	**Reserved**	Reserved.
[7:6]	**DAD_SEL**	**Transfer Destination Address Direction Selection** 00 = Transfer destination address is increasing successively. 01 = Reserved. 10 = Transfer destination address is fixed. (This feature can be used when data where transferred from multiple sources to a single destination). 11 = Reserved.

[5:4]	**SAD_SEL**	**Transfer Source Address Direction Selection** 00 = Transfer source address is increasing successively. 01 = Reserved. 10 = Transfer source address is fixed (This feature can be used when data where transferred from a single source to multiple destinations). 11 = Reserved.
[3:2]	**MODE_SEL**	**PDMA Mode Selection** 00 = Memory to Memory mode (Memory-to-Memory). 01 = Peripheral to Memory mode (Peripheral-to-Memory). 10 = Memory to Peripheral mode (Memory-to-Peripheral).
[1]	**SW_RST**	**Software Engine Reset** 0 = No effect. 1 = Reset the internal state machine, pointers and internal buffer. The contents of control register will not be cleared. This bit will be automatically cleared after few clock cycles.
[0]	**PDMACEN**	**PDMA Channel Enable Bit** Setting this bit to 1 enables PDMA operation. If this bit is cleared, PDMA will ignore all PDMA request and force Bus Master into IDLE state. **Note:** SW_RST(PDMA_CSRx[1], x= 0~8) will clear this bit.

PDMA Channel x Source Address Register (PDMA_SARx)

Register	Offset	R/W	Description	Reset Value
PDMA_SARx **x=0,1 .. 8**	PDMA_CHx_BA+0x04	R/W	PDMA Channel x Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**PDMA_SAR**	**PDMA Transfer Source Address Register** This field indicates a 32-bit source address of PDMA. **Note:** The source address must be word alignment.

PDMA Channel x Destination Address Register (PDMA_DARx)

Register	Offset	R/W	Description	Reset Value
PDMA_DARx **x=0,1 .. 8**	PDMA_CHx_BA+0x08	R/W	PDMA Channel x Destination Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**PDMA_DAR**	**PDMA Transfer Destination Address Register** This field indicates a 32-bit destination address of PDMA. **Note:** The destination address must be word alignment

PDMA Channel x Transfer Byte Count Register (PDMA_BCRx)

Register	Offset	R/W	Description	Reset Value
PDMA_BCRx **x=0,1 .. 8**	PDMA_CHx_BA+0x0C	R/W	PDMA Channel x Transfer Byte Count Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PDMA_BCR**	**PDMA Transfer Byte Count Register** This field indicates a 16-bit transfer byte count number of PDMA; it must be word alignment.

PDMA Channel x Internal Buffer Pointer Register (PDMA_POINTx)

Register	Offset	R/W	Description	Reset Value
PDMA_POINTx **x=0,1 .. 8**	PDMA_CHx_BA+0x10	R	PDMA Channel x Internal Buffer Pointer Register	0xXXXX_0000

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

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**PDMA_POINT**	**PDMA Internal Buffer Pointer Register (Read Only)** This field indicates the internal buffer pointer.

PDMA Channel x Current Source Address Register (PDMA_CSARx)

Register	Offset	R/W	Description	Reset Value
PDMA_CSARx **x=0,1 .. 8**	PDMA_CHx_BA+0x14	R	PDMA Channel x Current Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**PDMA_CSAR**	**PDMA Current Source Address Register (Read Only)** This field indicates the source address where the PDMA transfer just occurred.

PDMA Channel x Current Destination Address Register (PDMA_CDARx)

Register	Offset	R/W	Description	Reset Value
PDMA_CDARx **x=0,1 .. 8**	PDMA_CHx_BA+0x18	R	PDMA Channel x Current Destination Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**PDMA_CDAR**	**PDMA Current Destination Address Register (Read Only)** This field indicates the destination address where the PDMA transfer just occurred.

PDMA Channel x Current Byte Count Register (PDMA_CBCRx)

Register	Offset	R/W	Description	Reset Value
PDMA_CBCRx **x=0,1 .. 8**	PDMA_CHx_BA+0x1C	R	PDMA Channel x Current Transfer Byte Count Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PDMA_CBCR**	**PDMA Current Byte Count Register (Read Only)** This field indicates the current remained byte count of PDMA. **Note:** This field value will be cleared to 0, when software set SW_RST (PDMA_CSRx[1]) to “1”.

PDMA Channel x Interrupt Enable Register (PDMA_IERx)

Register	Offset	R/W	Description	Reset Value
PDMA_IERx **x=0,1 .. 8**	PDMA_CHx_BA+0x20	R/W	PDMA Channel x Interrupt Enable 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						**BLKD_IE**	**TABORT_IE**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**BLKD_IE**	**PDMA Block Transfer Done Interrupt Enable Bit** 0 = Interrupt generator Disabled when PDMA transfer is done. 1 = Interrupt generator Enabled when PDMA transfer is done.
[0]	**TABORT_IE**	**PDMA Read/Write Target Abort Interrupt Enable Bit** 0 = Target abort interrupt generation Disabled during PDMA transfer. 1 = Target abort interrupt generation Enabled during PDMA transfer.

PDMA Channel x Interrupt Status Register (PDMA_ISRx)

Register	Offset	R/W	Description	Reset Value
PDMA_ISRx **x=0,1 .. 8**	PDMA_CHx_BA+0x24	R/W	PDMA Channel x 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						**BLKD_IF**	**TABORT_IF**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**BLKD_IF**	**PDMA Block Transfer Done Interrupt Flag** This bit indicates that PDMA has finished all transfers. 0 = Not finished. 1 = Done. Write 1 to clear this bit to 0.
[0]	**TABORT_IF**	**PDMA Read/Write Target Abort Interrupt Flag** Write 1 to clear this bit to 0. 0 = No bus ERROR response received. 1 = Bus ERROR response received. **Note:** This bit filed indicates bus master received ERROR response or not. If bus master received ERROR response, it means that target abort is happened. PDMA controller will stop transfer and respond this event to software then goes to IDLE state. When target abort occurred, software must reset PDMA, and then transfer those data again.

PDMA Shared Buffer FIFO 0 Register (PDMA_SBUF0_Cx)

Register	Offset	R/W	Description	Reset Value
PDMA_SBUF0_Cx x=0,1 .. 8	PDMA_CHx_BA+0x80	R	PDMA Channel x Shared Buffer FIFO 0 Register	0x0000_0000

31	30	29	28	27	26	25	24
PDMA_SBUF0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PDMA_SBUF0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PDMA_SBUF0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PDMA_SBUF0							

Bits	Description	
[31:0]	**PDMA_SBUF0**	**PDMA Shared Buffer FIFO 0 (Read Only)** Each channel has its own 1 word internal buffer.

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
CRC_MODE		CPU_WDLEN		CHECKSUM_COM	WDATA_COM	CHECKSUM_RVS	WDATA_RVS
23	22	21	20	19	18	17	16
TRIG_EN	Reserved						
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CRC_RST	CRCCEN

Bits	Description	
[31:30]	**CRC_MODE**	**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]	**CPU_WDLEN**	**CPU Write Data Length** This field indicates the CPU write data length only when operating in CPU PIO mode. 00 = The write data length is 8-bit mode. 01 = The write data length is 16-bit mode. 10 = The write data length is 32-bit mode. 11 = Reserved. **Note1:** This field is only valid when operating in CPU PIO mode. **Note2:** When the write data length is 8-bit mode, the valid data in CRC_WDATA register is only CRC_WDATA [7:0] bits; if the write data length is 16-bit mode, the valid data in CRC_WDATA register is only CRC_WDATA [15:0].
[27]	**CHECKSUM_COM**	**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]	**WDATA_COM**	**Write Data 1's Complement** This bit is used to enable the 1's complement function for write data value in CRC_WDATA register. 0 = 1's complement for CRC write data in Disabled. 1 = 1's complement for CRC write data in Enabled.

[25]	**CHECKSUM_RVS**	**Checksum Reverse** This bit is used to enable the bit order reverse function for write data value 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]	**WDATA_RVS**	**Write Data Order Reverse** This bit is used to enable the bit order reverse function for write data value in CRC_WDATA register. 0 = Bit order reverse for CRC write data in Disabled. 1 = Bit order reverse 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]	**TRIG_EN**	**Trigger Enable Bit** This bit is used to trigger the CRC DMA transfer. 0 = No effect. 1 = CRC DMA data read or write transfer Enabled. **Note1:** If this bit asserts which indicates the CRC engine operation in CRC DMA mode, do not fill in any data in CRC_WDATA register. **Note2:** When CRC DMA transfer completed, this bit will be cleared automatically. **Note3:** If the bus error occurs when CRC DMA transfer data, all CRC DMA transfer will be stopped. Software must reset all DMA channel before trigger DMA again.
[22:2]	**Reserved**	Reserved.
[1]	**CRC_RST**	**CRC Engine Reset** 0 = No effect. 1 = Reset the internal CRC state machine and internal buffer. The others contents of CRC_CTL register will not be cleared. This bit will be cleared automatically. **Note:** When operated in CPU PIO mode, setting this bit will reload the initial seed value (CRC_SEED register).
[0]	**CRCCEN**	**CRC Channel Enable Bit** 0 = No effect. 1 = CRC operation Enabled. **Note1:** When operating in CRC DMA mode (TRIG_EN (CRC_CTL[23]) = 1), if user clears this bit, the DMA operation will be continuous until all CRC DMA operation is done, and the TRIG_EN (CRC_CTL[23]) bit will keep 1until all CRC DMA operation done. But in this case, the CRC_BLKD_IF (CRC_DMAISR[1])flag will inactive, user can read CRC checksum result only if TRIG_EN (CRC_CTL[23]) clears to 0 **Note2:** When operating in CRC DMA mode (TRIG_EN (CRC_CTL[23]) = 1), if user wants to stop the transfer immediately, user can write 1 to CRC_RST (CRC_CTL [1]) bit to stop the transmission.

CRC DMA Source Address Register (CRC_DMASAR)

Register	Offset	R/W	Description	Reset Value
CRC_DMASAR	CRC_BA+0x04	R/W	CRC DMA Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**CRC_DMASAR**	**CRC DMA Transfer Source Address Register** This field indicates a 32-bit source address of CRC DMA. (CRC_DMASAR + CRC_DMABCR) = (CRC_DMACSAR + CRC_DMACBCR). **Note:** The source address must be word alignment

CRC DMA Transfer Byte Count Register (CRC_DMABCR)

Register	Offset	R/W	Description	Reset Value
CRC_DMABCR	CRC_BA+0x0C	R/W	CRC DMA Transfer Byte Count Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CRC_DMABCR**	**CRC DMA Transfer Byte Count Register** This field indicates a 16-bit total transfer byte count number of CRC DMA (CRC_DMASAR + CRC_DMABCR) = (CRC_DMACSAR + CRC_DMACBCR).

CRC DMA Current Source Address Register (CRC_DMACSAR)

Register	Offset	R/W	Description	Reset Value
CRC_DMACSAR	CRC_BA+0x14	R	CRC DMA Current Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**CRC_DMACSAR**	**CRC DMA Current Source Address Register (Read Only)** This field indicates the current source address where the CRC DMA transfer just occurs. (CRC_DMASAR + CRC_DMABCR) = (CRC_DMACSAR + CRC_DMACBCR).

CRC DMA Current Transfer Byte Count Register (CRC_DMACBCR)

Register	Offset	R/W	Description	Reset Value
CRC_DMACBCR	CRC_BA+0x1C	R	CRC DMA Current Transfer Byte Count Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CRC_DMACBCR**	**CRC DMA Current Remained Byte Count Register (Read Only)** This field indicates the current remained byte count of CRC DMA. (CRC_DMASAR + CRC_DMABCR) = (CRC_DMACSAR + CRC_DMACBCR). **Note:** Setting CRC_RST (CRC_CTL[1]) bit to 1 will clear this register value.

CRC DMA Interrupt Enable Register (CRC_DMAIER)

Register	Offset	R/W	Description	Reset Value
CRC_DMAIER	CRC_BA+0x20	R/W	CRC DMA Interrupt Enable 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						**CRC_BLKD_IE**	**CRC_TABORT_IE**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CRC_BLKD_IE**	**CRC DMA Block Transfer Done Interrupt Enable Bit** Enable this bit will generate the CRC DMA Transfer Done interrupt signal while CRC_BLKD_IF (CRC_DMAISR[1]) bit is set to 1. 0 = Interrupt generator Disabled when CRC DMA transfer done. 1 = Interrupt generator Enabled when CRC DMA transfer done.
[0]	**CRC_TABORT_IE**	**CRC DMA Read/Write Target Abort Interrupt Enable Bit** Enable this bit will generate the CRC DMA Target Abort interrupt signal while CRC_TARBOT_IF (CRC_DMAISR[0]) bit is set to 1. 0 = Target abort interrupt generation Disabled during CRC DMA transfer. 1 = Target abort interrupt generation Enabled during CRC DMA transfer.

CRC DMA Interrupt Status Register (CRC_DMAISR)

Register	Offset	R/W	Description	Reset Value
CRC_DMAISR	CRC_BA+0x24	R/W	CRC DMA 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						CRC_BLKD_IF	CRC_TABORT_IF

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CRC_BLKD_IF**	**CRC DMA Block Transfer Done Interrupt Flag** This bit indicates that CRC DMA transfer has finished or not. 0 = Not finished if TRIG_EN (CRC_CTL[23]) bit has enabled. 1 = CRC transfer done if TRIG_EN (CRC_CTL[23]) bit has enabled. It is cleared by writing 1 to it through software.. (When CRC DMA transfer done, TRIG_EN (CRC_CTL[23]) bit will be cleared automatically)
[0]	**CRC_TABORT_IF**	**CRC DMA Read/Write Target Abort Interrupt Flag** This bit indicates that CRC bus has error or not during CRC DMA transfer. 0 = No bus error response received during CRC DMA transfer. 1 = Bus error response received during CRC DMA transfer. It is cleared by writing 1 to it through software. **Note:** The bit filed indicate bus master received error response or not. If bus master received error response, it means that CRC transfer target abort is happened. DMA will stop transfer and respond this event to software then CRC state machine goes to IDLE state. When target abort occurred, software must reset DMA before transfer those data again.

CRC Write Data Register (CRC_WDATA)

Register	Offset	R/W	Description	Reset Value
CRC_WDATA	CRC_BA+0x80	R/W	CRC Write Data Register	0x0000_0000

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

Bits	Description	
[31:0]	CRC_WDATA	**CRC Write Data Register** When operating in CPU PIO mode, software can write data to this field to perform CRC operation. When operating in DMA mode, this field indicates the DMA read data from memory and cannot be written. **Note:** When the write data length is 8-bit mode, the valid data in CRC_WDATA register is only CRC_WDATA [7:0] bits; if the write data length is 16-bit mode, the valid data in CRC_WDATA register is only CRC_WDATA [15:0].

CRC Seed Register (CRC_SEED)

Register	Offset	R/W	Description	Reset Value
CRC_SEED	CRC_BA+0x84	R/W	CRC Seed Register	0xFFFF_FFFF

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

Bits	Description	
[31:0]	CRC_SEED	**CRC Seed Register** This field indicates the CRC seed value.

CRC Checksum Register (CRC_CHECKSUM)

Register	Offset	R/W	Description	Reset Value
CRC_CHECKSUM	CRC_BA+0x88	R	CRC Checksum Register	0xFFFF_FFFF

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

Bits	Description	
[31:0]	CRC_CHECKSUM	**CRC Checksum Register** This fields indicates the CRC checksum result

PDMA Global Control Register (PDMA_GCRCSR)

Register	Offset	R/W	Description	Reset Value
PDMA_GCRCSR	PDMA_GCR_BA+0x00	R/W	PDMA Global Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							CRC_CLK_EN
23	22	21	20	19	18	17	16
Reserved							CLK8_EN
15	14	13	12	11	10	9	8
CLK7_EN	CLK6_EN	CLK5_EN	CLK4_EN	CLK3_EN	CLK2_EN	CLK1_EN	CLK0_EN
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**CRC_CLK_EN**	**CRC Controller Clock Enable Bit** 0 = Disabled. 1 = Enabled.
[23:17]	**Reserved**	Reserved.
[16]	**CLK8_EN**	**PDMA Controller Channel 8 Clock Enable Bit** 0 = Disabled. 1 = Enabled.
[15]	**CLK7_EN**	**PDMA Controller Channel 7 Clock Enable Bit** 0 = Disabled. 1 = Enabled.
[14]	**CLK6_EN**	**PDMA Controller Channel 6 Clock Enable Bit** 0 = Disabled. 1 = Enabled.
[13]	**CLK5_EN**	**PDMA Controller Channel 5 Clock Enable Bit** 0 = Disabled. 1 = Enabled.
[12]	**CLK4_EN**	**PDMA Controller Channel 4 Clock Enable Bit** 0 = Disabled. 1 = Enabled.
[11]	**CLK3_EN**	**PDMA Controller Channel 3 Clock Enable Bit** 0 = Disabled. 1 = Enabled.

[10]	**CLK2_EN**	**PDMA Controller Channel 2 Clock Enable Bit** 0 = Disabled. 1 = Enabled.
[9]	**CLK1_EN**	**PDMA Controller Channel 1 Clock Enable Bit** 0 = Disabled. 1 = Enabled.
[8]	**CLK0_EN**	**PDMA Controller Channel 0 Clock Enable Bit** 0 = Disabled. 1 = Enabled.
[7:0]	**Reserved**	Reserved.

PDMA Service Selection Control Register 0 (PDMA_PDSSR0)

Register	Offset	R/W	Description	Reset Value
PDMA_PDSSR0	PDMA_GCR_BA+0x04	R/W	PDMA Service Selection Control Register 0	0xFFFF_FFFF

31	30	29	28	27	26	25	24
SPI3_TXSEL				SPI3_RXSEL			
23	22	21	20	19	18	17	16
SPI2_TXSEL				SPI2_RXSEL			
15	14	13	12	11	10	9	8
SPI1_TXSEL				SPI1_RXSEL			
7	6	5	4	3	2	1	0
SPI0_TXSEL				SPI0_RXSEL			

Bits	Description	
[31:28]	SPI3_TXSEL	**PDMA SPI3 TX Selection** This field defines which PDMA channel is connected to the on-chip peripheral SPI3 TX. Software can configure the TX channel setting by this field. The channel configuration is the same as SPI0_RXSEL (PDMA_PDSSR0[3:0]) field. Please refer to the explanation of SPI0_RXSEL (PDMA_PDSSR0[3:0]).
[27:24]	SPI3_RXSEL	**PDMA SPI3 RX Selection** This field defines which PDMA channel is connected to the on-chip peripheral SPI3 RX. Software can configure the RX channel setting by this field. The channel configuration is the same as SPI0_RXSEL (PDMA_PDSSR0[3:0]) field. Please refer to the explanation of SPI0_RXSEL (PDMA_PDSSR0[3:0]).
[23:20]	SPI2_TXSEL	**PDMA SPI2 TX Selection** This field defines which PDMA channel is connected to the on-chip peripheral SPI2 TX. Software can configure the TX channel setting by this field. The channel configuration is the same as SPI0_RXSEL (PDMA_PDSSR0[3:0]) field. Please refer to the explanation of SPI0_RXSEL (PDMA_PDSSR0[3:0]).
[19:16]	SPI2_RXSEL	**PDMA SPI2 RX Selection** This field defines which PDMA channel is connected to the on-chip peripheral SPI2 RX. Software can configure the RX channel setting by this field. The channel configuration is the same as SPI0_RXSEL (PDMA_PDSSR0[3:0]) field. Please refer to the explanation of SPI0_RXSEL (PDMA_PDSSR0[3:0]).
[15:12]	SPI1_TXSEL	**PDMA SPI1 TX Selection** This field defines which PDMA channel is connected to the on-chip peripheral SPI1 TX. Software can configure the TX channel setting by this field. The channel configuration is the same as SPI0_RXSEL (PDMA_PDSSR0[3:0]) field. Please refer to the explanation of SPI0_RXSEL (PDMA_PDSSR0[3:0]).
[11:8]	SPI1_RXSEL	**PDMA SPI1 RX Selection** This field defines which PDMA channel is connected to the on-chip peripheral SPI1 RX. Software can configure the RX channel setting by this field. The channel configuration is the same as SPI0_RXSEL (PDMA_PDSSR0[3:0]) field. Please refer to the explanation of SPI0_RXSEL (PDMA_PDSSR0[3:0]).

[7:4]	**SPI0_TXSEL**	**PDMA SPI0 TX Selection** This field defines which PDMA channel is connected to the on-chip peripheral SPI0 TX. Software can configure the TX channel setting by this field. The channel configuration is the same as SPI0_RXSEL (PDMA_PDSSR0[3:0]) field. Please refer to the explanation of SPI0_RXSEL (PDMA_PDSSR0[3:0]).
[3:0]	**SPI0_RXSEL**	**PDMA SPI0 RX Selection** This field defines which PDMA channel is connected to the on-chip peripheral SPI0 RX. Software can change the channel RX setting by this field. For example, SPI0_RXSEL (PDMA_PDSSR0[3:0]) = 0110, that means SPI0_RX is connected to PDMA_CH6. 0000: CH0 0001: CH1 0010: CH2 0011: CH3 0100: CH4 0101: CH5 0110: CH6 0111: CH7 1000: CH8 Others : Reserved

PDMA Service Selection Control Register 1 (PDMA_PDSSR1)

Register	Offset	R/W	Description	Reset Value
PDMA_PDSSR1	PDMA_GCR_BA+0x08	R/W	PDMA Service Selection Control Register 1	0xFFFF_FFFF

31	30	29	28	27	26	25	24
Reserved				ADC_RXSEL			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
UART1_TXSEL				UART1_RXSEL			
7	6	5	4	3	2	1	0
UART0_TXSEL				UART0_RXSEL			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**ADC_RXSEL**	**PDMA ADC RX Selection** This field defines which PDMA channel is connected to the on-chip peripheral ADC RX. Software can configure the RX channel setting by this field. The channel configuration is the same as UART0_RXSEL (PDMA_PDSSR1[3:0]) field. Please refer to the explanation of UART0_RXSEL (PDMA_PDSSR1[3:0]).
[23:16]	**Reserved**	Reserved.
[15:12]	**UART1_TXSEL**	**PDMA UART1 TX Selection** This field defines which PDMA channel is connected to the on-chip peripheral UART1 TX. Software can configure the TX channel setting by this field. The channel configuration is the same as UART0_RXSEL (PDMA_PDSSR1[3:0]) field. Please refer to the explanation of UART0_RXSEL (PDMA_PDSSR1[3:0]).
[11:8]	**UART1_RXSEL**	**PDMA UART1 RX Selection** This field defines which PDMA channel is connected to the on-chip peripheral UART1 RX. Software can configure the RX channel setting by this field. The channel configuration is the same as UART0_RXSEL (PDMA_PDSSR1[3:0]) field. Please refer to the explanation of UART0_RXSEL (PDMA_PDSSR1[3:0]).
[7:4]	**UART0_TXSEL**	**PDMA UART0 TX Selection** This field defines which PDMA channel is connected to the on-chip peripheral UART0 TX. Software can configure the TX channel setting by this field. The channel configuration is the same as UART0_RXSEL (PDMA_PDSSR1[3:0]) field. Please refer to the explanation of UART0_RXSEL (PDMA_PDSSR1[3:0]).

[3:0]	**UART0_RXSEL**	**PDMA UART0 RX Selection** This field defines which PDMA channel is connected to the on-chip peripheral UART0 RX. Software can change the channel RX setting by this field. For example, UART0_RXSEL (PDMA_PDSSR1[3:0]) = 0110, which means UART0_RX is connected to PDMA_CH6. 0000: CH0 0001: CH1 0010: CH2 0011: CH3 0100: CH4 0101: CH5 0110: CH6 0111: CH7 1000: CH8 Others : Reserved

PDMA Global Interrupt Status Register (PDMA_GCRISR)

Register	Offset	R/W	Description	Reset Value
PDMA_GCRISR	PDMA_GCR_BA+0x0C	R	PDMA Global Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
INTR	Reserved						
23	22	21	20	19	18	17	16
Reserved							INTRCRC
15	14	13	12	11	10	9	8
Reserved							INTR8
7	6	5	4	3	2	1	0
INTR7	INTR6	INTR5	INTR4	INTR3	INTR2	INTR1	INTR0

Bits	Description	
[31]	**INTR**	**Interrupt Status** This bit is the interrupt status of PDMA controller. **Note:** This bit is read only.
[30:17]	**Reserved**	Reserved.
[16]	**INTRCRC**	**Interrupt Status Of CRC Controller** This bit is the interrupt status of CRC controller **Note:** This bit is read only
[15:9]	**Reserved**	Reserved.
[8]	**INTR8**	**Interrupt Status Of Channel 8** This bit is the interrupt status of PDMA channel8. **Note:** This bit is read only.
[7]	**INTR7**	**Interrupt Status Of Channel 7** This bit is the interrupt status of PDMA channel7. **Note:** This bit is read only.
[6]	**INTR6**	**Interrupt Status Of Channel 6** This bit is the interrupt status of PDMA channel6. **Note:** This bit is read only.
[5]	**INTR5**	**Interrupt Status Of Channel 5** This bit is the interrupt status of PDMA channel5. **Note:** This bit is read only.
[4]	**INTR4**	**Interrupt Status Of Channel 4** This bit is the interrupt status of PDMA channel4. **Note:** This bit is read only.

[3]	**INTR3**	**Interrupt Status Of Channel 3** This bit is the interrupt status of PDMA channel3. **Note:** This bit is read only.
[2]	**INTR2**	**Interrupt Status Of Channel 2** This bit is the interrupt status of PDMA channel2. **Note:** This bit is read only.
[1]	**INTR1**	**Interrupt Status Of Channel 1** This bit is the interrupt status of PDMA channel1. **Note:** This bit is read only.
[0]	**INTR0**	**Interrupt Status Of Channel 0** This bit is the interrupt status of PDMA channel0. **Note:** This bit is read only.

6.8 Timer Controller (TIMER)

6.8.1 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.

6.8.2 Features

- Four sets of 32-bit timers with 24-bit up counter and one 8-bit prescale counter
- Independent clock source for each timer
- Provides four timer counting modes: one-shot, periodic, toggle and continuous counting
- Time-out period = (Period of timer clock input) * (8-bit prescale counter + 1) * (24-bit TCMP)
- Maximum counting cycle time = (1 / T MHz) * (2^8) * (2^{24}), T is the period of timer clock
- 24-bit up counter value is readable through TDR (Timer Data Register)
- Supports event counting function to count the event from external counter pin (TM0~TM3)
- Supports external pin capture (TM0_EXT~TM3_EXT) for interval measurement
- Supports external pin capture (TM0_EXT~TM3_EXT) for reset 24-bit up counter
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated

6.8.3 Block Diagram

The Timer Controller block diagram and clock control are shown as follows.

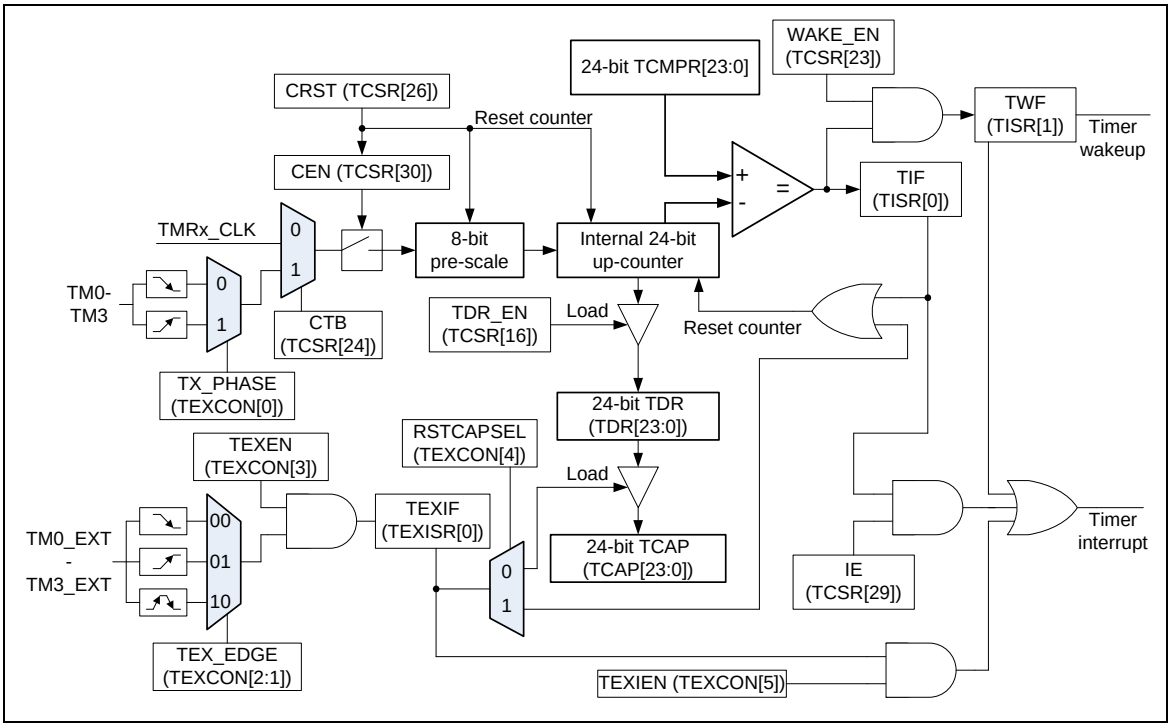


Figure 6.8-1 Timer Controller Block Diagram

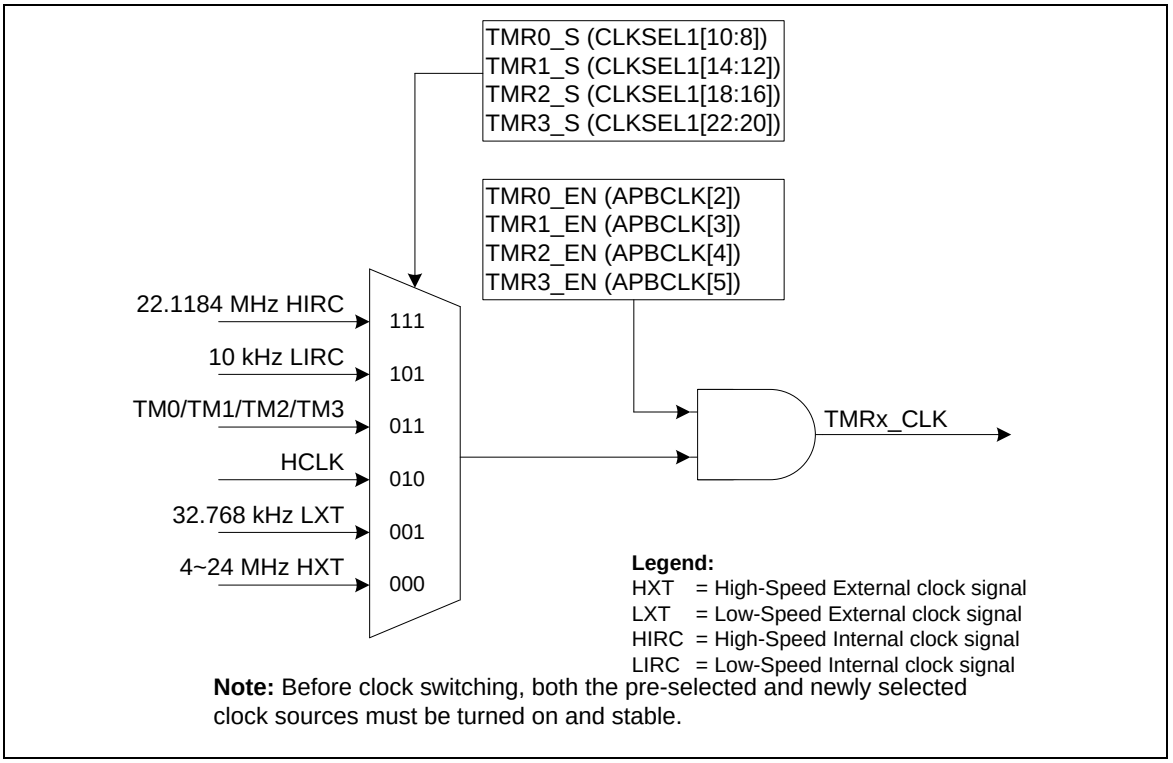


Figure 6.8-2 Clock Source of Timer Controller

6.8.4 Basic Configuration

The peripheral clock source of Timer0 ~ Timer3 can be enabled in APBCLK[5:2] and selected as different frequency in CLKSEL1[10:8] for Timer0, CLKSEL1[14:12] for Timer1, CLKSEL1[18:16] for Timer2 and CLKSEL1[22:20] for Timer3.

6.8.5 Functional Description

6.8.5.1 Timer Interrupt Flag

Timer controller supports two interrupt flags; one is TIF flag and its set while timer counter value (TDR) matches the timer compared value (TCMP), the other is TEXIF flag and its set when the transition on the TMx_EXT pin associated TEX_EDGE setting.

6.8.5.2 One–shot Mode

If timer controller is configured at one-shot mode (TCSR[28:27] is 00) and CEN (TCSR[30]) bit is set, the timer counter starts up counting. Once the TDR value reaches TCMP value, the TIF flag will be set to 1, TDR value and CEN bit is cleared by timer controller then timer counting operation stops. In the meantime, if the IE (TCSR[29]) bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also.

6.8.5.3 Periodic Mode

If timer controller is configured at periodic mode (TCSR[28:27] is 01) and CEN bit is set, the timer counter starts up counting. Once the TDR value reaches TCMP value, the TIF flag will be set to 1, TDR value will be cleared by timer controller and timer counter operates counting again. In the meantime, if the IE bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. In this mode, timer controller operates counting and compares with TCMP value periodically until the CEN bit is cleared by software.

6.8.5.4 Toggle-output Mode

If timer controller is configured at toggle-out mode (TCSR[28:27] is 10) and CEN bit is set, the timer counter starts up counting. The counting operation of toggle-out mode is almost the same as periodic mode, except toggle-out mode has associated TM0~MT3 pin to output signal while specify TIF bit is set. Thus, the toggle-output signal on TM0~TM3 pin is changing back and forth with 50% duty cycle.

6.8.5.5 Continuous Counting Mode

If timer controller is configured at continuous counting mode (TCSR[28:27] is 11) and CEN bit is set, the timer counter starts up counting. Once the TDR value reaches TCMP value, the TIF flag will be set to 1 and TDR value keeps up counting. In the meantime, if the IE bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. User can change different TCMP value immediately without disabling timer counting and restarting timer counting in this mode.

For example, TCMP value is set as 80, first. The TIF flag will set to 1 when TDR value is equal to 80, timer counter is kept counting and TDR value will not goes back to 0, it continues to count 81, 82, 83,··· to 2^{24} -1, 0, 1, 2, 3, ··· to 2^{24} -1 again and again. Next, if software programs TCMP value as 200 and clears TIF flag, the TIF flag will set to 1 again when TDR value reaches to 200. At last, software programs TCMP as 500 and clears TIF flag, the TIF flag will set to 1 again when TDR value reaches to 500.

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

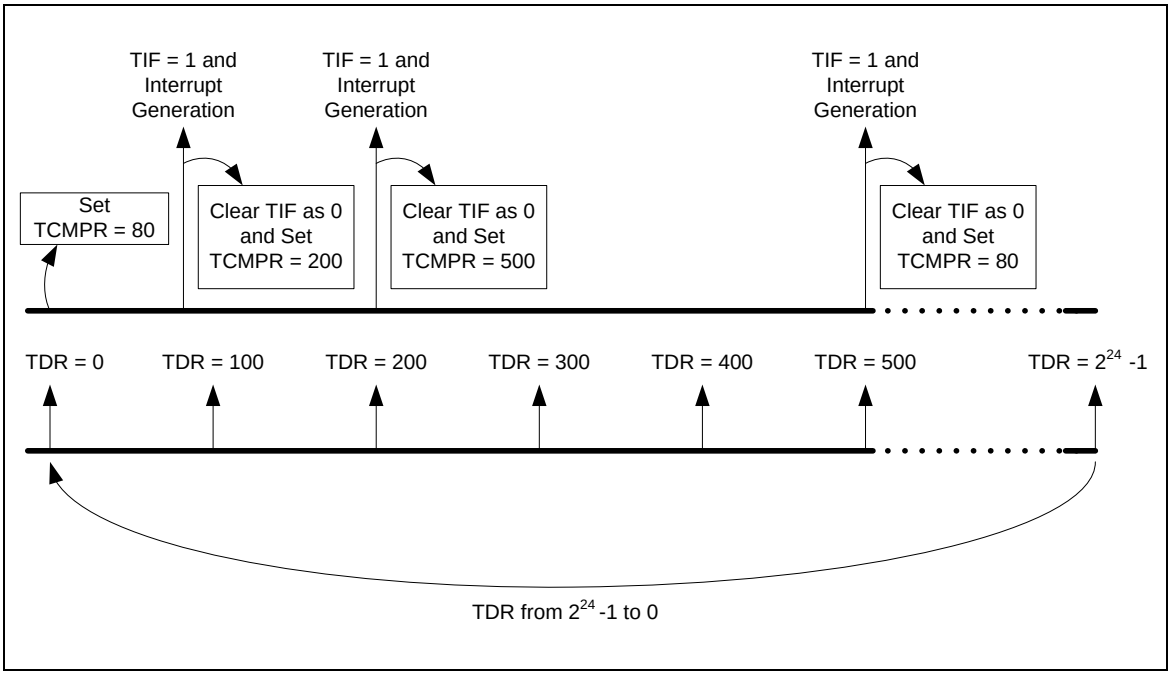


Figure 6.8-3 Continuous Counting Mode

6.8.5.6 Event Counting Mode

Timer controller also provides an application which can count the input event from TMx pin (x= 0~3) and the number of event will reflect to TDR value. It is also called as event counting function. In this function, CTB (TCSR[24]) bit should be set and the timer peripheral clock source should be set as HCLK.

Software can enable or disable TMx pin de-bounce circuit by TCDB (TEXCON[7]) bit. The input event frequency should be less than 1/3 HCLK if TMx pin de-bounce disabled or less than 1/8 HCLK if TMx pin de-bounce enabled to assure the returned TDR value is incorrect, and software can also select edge detection phase of TMx pin by TX_PHASE (TEXCON[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 TDR value by input event from TMx pin.

6.8.5.7 External Capture Mode

The event capture function is used to capture Timer Capture Data Register (TDR) value to TCAP value while edge transition detected on TMx_EXT pin (x= 0~3). In this mode, RSTCAPSEL (TEXCON[4]) bit should be as 0 for select TMx_EXT transition is using as the event capture function and the timer peripheral clock source should be set as HCLK.

Software can enable or disable TxEX pin de-bounce circuit by TEXDB (TEXCON[6]) bit. The transition frequency of TMx_EXT pin should be less than 1/3 HCLK if TMx_EXT pin de-bounce disabled or less than 1/8 HCLK if TMx_EXT pin de-bounce enabled to assure the capture function can be work normally, and software can also select edge transition detection of TMx_EXT pin by TEX_EDGE (TEXCON[2:1]) bits.

In event capture mode, software does not consider what timer counting operation mode is selected, the capture event occurred only if edge transition on TMx_EXT pin is detected.

6.8.5.8 Event Reset Counter Mode

It also provides event reset counter function to reset TDR value while edge transition detected on TMx_EXT pin (x= 0~3). In this mode, most the settings are the same as event capture function except RSTCAPSEL (TEXCON[4]) bit should be as 1 for select TMx_EXT transition is using as the event reset counter.

6.8.6 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 = 0x4001_0000** **TMR23_BA = 0x4011_0000**				
TCSR0	TMR01_BA+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TCMPR0	TMR01_BA+0x04	R/W	Timer0 Compare Register	0x0000_0000
TISR0	TMR01_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TDR0	TMR01_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TCAP0	TMR01_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TEXCON0	TMR01_BA+0x14	R/W	Timer0 External Control Register	0x0000_0000
TEXISR0	TMR01_BA+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TCSR1	TMR01_BA+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TCMPR1	TMR01_BA+0x24	R/W	Timer1 Compare Register	0x0000_0000
TISR1	TMR01_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TDR1	TMR01_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TCAP1	TMR01_BA+0x30	R	Timer1 Capture Data Register	0x0000_0000
TEXCON1	TMR01_BA+0x34	R/W	Timer1 External Control Register	0x0000_0000
TEXISR1	TMR01_BA+0x38	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TCSR2	TMR23_BA+0x00	R/W	Timer2 Control and Status Register	0x0000_0005
TCMPR2	TMR23_BA+0x04	R/W	Timer2 Compare Register	0x0000_0000
TISR2	TMR23_BA+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TDR2	TMR23_BA+0x0C	R	Timer2 Data Register	0x0000_0000
TCAP2	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TEXCON2	TMR23_BA+0x14	R/W	Timer2 External Control Register	0x0000_0000
TEXISR2	TMR23_BA+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TCSR3	TMR23_BA+0x20	R/W	Timer3 Control and Status Register	0x0000_0005
TCMPR3	TMR23_BA+0x24	R/W	Timer3 Compare Register	0x0000_0000
TISR3	TMR23_BA+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000
TDR3	TMR23_BA+0x2C	R	Timer3 Data Register	0x0000_0000

TCAP3	TMR23_BA+0x30	R	Timer3 Capture Data Register	0x0000_0000
TEXCON3	TMR23_BA+0x34	R/W	Timer3 External Control Register	0x0000_0000
TEXISR3	TMR23_BA+0x38	R/W	Timer3 External Interrupt Status Register	0x0000_0000

6.8.7 Register Description

Timer Control Register (TCSR)

Register	Offset	R/W	Description	Reset Value
TCSR0	TMR01_BA+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TCSR1	TMR01_BA+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TCSR2	TMR23_BA+0x00	R/W	Timer2 Control and Status Register	0x0000_0005
TCSR3	TMR23_BA+0x20	R/W	Timer3 Control and Status Register	0x0000_0005

31	30	29	28	27	26	25	24
DBGACK_TMR	CEN	IE	MODE		CRST	CACT	CTB
23	22	21	20	19	18	17	16
WAKE_EN	Reserved						TDR_EN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PRESCALE							

Bits	Description	
[31]	**DBGACK_TMR**	**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.
[30]	**CEN**	**Timer Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. **Note1:** In stop status, and then set CEN to 1 will enable the 24-bit up counter to keep counting from the last stop counting value. **Note2:** This bit is auto-cleared by hardware in one-shot mode (TCSR [28:27] = 00) when the timer interrupt flag TIF (TISR[0]) is generated.
[29]	**IE**	**Interrupt Enable Bit** 0 = Timer Interrupt function Disabled. 1 = Timer Interrupt function Enabled. If this bit is enabled, when the timer interrupt flag TIF (TISR[0]) is set to 1, the timer interrupt signal is generated and inform to CPU.

[28:27]	**MODE**	**Timer Operating Mode** 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]	**CRST**	**Timer Reset** 0 = No effect. 1 = Reset 8-bit prescale counter, 24-bit up counter value and CEN bit if CACT is 1.
[25]	**CACT**	**Timer Active Status (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.
[24]	**CTB**	**Counter Mode Enable Bit** This bit is for external counting pin function enabled. When timer is used as an event counter, this bit should be set to 1 and select HCLK as timer clock source. Please refer to 6.8.5.6 for detail description. 0 = External counter mode Disabled. 1 = External counter mode Enabled.
[23]	**WAKE_EN**	**Wake Up Function Enable Bit** 0 = Wake-up trigger event Disabled. 1 = Wake-up trigger event Enabled.
[22:17]	**Reserved**	Reserved.
[16]	**TDR_EN**	**Data Load Enable Bit** When TDR_EN is set, TDR (Timer Data Register) will be updated continuously with the 24-bit up-timer value as the timer is counting. 0 = Timer Data Register update Disabled. 1 = Timer Data Register update Enabled while Timer counter is active.
[15:8]	**Reserved**	Reserved.
[7:0]	**PRESCALE**	**Prescale Counter** Timer input clock source is divided by (PRESCALE+1) before it is fed to the Timer up counter. If this field is 0 (PRESCALE = 0), then there is no scaling.

Timer Compare Register (TCMPR)

Register	Offset	R/W	Description	Reset Value
TCMPR0	TMR01_BA+0x04	R/W	Timer0 Compare Register	0x0000_0000
TCMPR1	TMR01_BA+0x24	R/W	Timer1 Compare Register	0x0000_0000
TCMPR2	TMR23_BA+0x04	R/W	Timer2 Compare Register	0x0000_0000
TCMPR3	TMR23_BA+0x24	R/W	Timer3 Compare Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**TCMP**	**Timer Compared Value** TCMP is a 24-bit compared value register. When the internal 24-bit up counter value is equal to TCMP value, the TIF flag will set to 1. Time-out period = (Period of Timer clock input) * (8-bit PRESCALE + 1) * (24-bit TCMP). **Note1:** Never write 0x0 or 0x1 in TCMP field, or the core will run into unknown state. **Note2:** 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 TCMP field. But if timer is operating at other modes, the 24-bit up counter will restart counting and using newest TCMP value to be the timer compared value if user writes a new value into TCMP field.

Timer Interrupt Status Register (TISR)

Register	Offset	R/W	Description	Reset Value
TISR0	TMR01_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TISR1	TMR01_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TISR2	TMR23_BA+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TISR3	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						**TWF**	**TIF**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TWF**	**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 TDR value reaches to TCMP value. 0 = No effect. 1 = TDR value matches the TCMP value. **Note:** This bit is cleared by writing 1 to it.

Timer Data Register (TDR)

Register	Offset	R/W	Description	Reset Value
TDR0	TMR01_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TDR1	TMR01_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TDR2	TMR23_BA+0x0C	R	Timer2 Data Register	0x0000_0000
TDR3	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**
TDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TDR							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**TDR**	**Timer Data Register** If TDR_EN (TCSR[16]) is set to 1, TDR register will be updated continuously to monitor 24-bit up counter value.

Timer Capture Data Register (TCAP)

Register	Offset	R/W	Description	Reset Value
TCAP0	TMR01_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TCAP1	TMR01_BA+0x30	R	Timer1 Capture Data Register	0x0000_0000
TCAP2	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TCAP3	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**
TCAP							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TCAP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TCAP							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**TCAP**	**Timer Capture Data Register** When TEXIF (TEXISR[0]) flag and RSTCAPSEL (TEXCON[4]) is set to 1, the current TDR value will be auto-loaded into this TCAP filed immediately.

Timer External Control Register (TEXCON)

Register	Offset	R/W	Description	Reset Value
TEXCON0	TMR01_BA+0x14	R/W	Timer0 External Control Register	0x0000_0000
TEXCON1	TMR01_BA+0x34	R/W	Timer1 External Control Register	0x0000_0000
TEXCON2	TMR23_BA+0x14	R/W	Timer2 External Control Register	0x0000_0000
TEXCON3	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							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TCDB	**TEXDB**	**TEXIEN**	**RSTCAPSEL**	**TEXEN**	**TEX_EDGE**		**TX_PHASE**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**TCDB**	**Timer External Counter Input Pin De-Bounce Enable Bit** 0 = TMx pin de-bounce Disabled. 1 = TMx pin de-bounce Enabled. If this bit is enabled, the edge detection of TMx pin is detected with de-bounce circuit.
[6]	**TEXDB**	**Timer External Capture Input Pin De-Bounce Enable Bit** 0 = TMx_EXT pin de-bounce Disabled. 1 = TMx_EXT pin de-bounce Enabled. If this bit is enabled, the edge detection of TMx_EXT pin is detected with de-bounce circuit.
[5]	**TEXIEN**	**Timer External Capture Interrupt Enable Bit** 0 = TMx_EXT pin detection Interrupt Disabled. 1 = TMx_EXT pin detection Interrupt Enabled. If TEXIEN enabled, Timer will raise an external capture interrupt signal and inform to CPU while TEXIF flag is set to 1.
[4]	**RSTCAPSEL**	**Timer External Reset Counter / Timer External Capture Mode Selection** 0 = Transition on TMx_EXT pin is using to save the TDR value into TCAP.(event capture function) 1 = Transition on TMx_EXT pin is using to reset the 24-bit up counter.(event reset counter function)
[3]	**TEXEN**	**Timer External Pin Function Enable Bit** This bit enables the RSTCAPSEL function on the TMx_EXT pin. 0 = RSTCAPSEL function of TMx_EXT pin will be ignored.

		1 = RSTCAPSEL function of TMx_EXT pin is active.
[2:1]	**TEX_EDGE**	**Timer External Capture Pin Edge Detect Selection** 00 = A 1 to 0 transition on TMx_EXT pin will be detected. 01 = A 0 to 1 transition on TMx_EXT pin will be detected. 10 = Either 1 to 0 or 0 to 1 transition on TMx_EXT pin will be detected. 11 = Reserved.
[0]	**TX_PHASE**	**Timer External Count Pin Phase Detect Selection** This bit indicates the detection phase of TMx_EXT pin. 0 = A falling edge of TMx_EXT pin will be counted. 1 = A rising edge of TMx_EXT pin will be counted.

Timer External Interrupt Status Register (TEXISR)

Register	Offset	R/W	Description	Reset Value
TEXISR0	TMR01_BA+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TEXISR1	TMR01_BA+0x38	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TEXISR2	TMR23_BA+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TEXISR3	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							**TEXIF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**TEXIF**	**Timer External Capture Interrupt Flag** This bit indicates the external capture interrupt flag status. When TEXEN (TEXCON[3]) enabled, TMx_EXT pin selected as external capture function, and a transition on TMx_EXT pin matched the TEX_EDGE (TEXCON[2:1]) setting, this flag will set to 1 by hardware. 0 = TMx_EXT pin interrupt did not occur. 1 = TMx_EXT pin interrupt occurred. **Note:** This bit is cleared by writing 1 to it.

6.9 PWM Generator and Capture Timer (PWM)

6.9.1 Overview

The NuMicro® NUC029LEE/NUC029SEE has 2 sets of PWM group supporting a total of 3 sets of PWM generators that can be configured as 6 independent PWM outputs, PWM0~PWM5, or as 3 complementary PWM pairs, (PWM0, PWM1), (PWM2, PWM3) and (PWM4, PWM5) with 3 programmable Dead-zone generators.

Each PWM generator has one 8-bit prescaler, one clock divider with 5 divided frequencies (1, 1/2, 1/4, 1/8, 1/16), two PWM Timers including two clock selectors, two 16-bit PWM counters for PWM period control, two 16-bit comparators for PWM duty control and one Dead-zone generator. The 3 sets of PWM generators provide eight independent PWM interrupt flags set by hardware when the corresponding PWM period down counter reaches 0. Each PWM interrupt source with its corresponding enable bit can cause CPU to request PWM interrupt. The PWM generators can be configured as one-shot mode to produce only one PWM cycle signal or auto-reload mode to output PWM waveform continuously.

When DZEN01 (PCR[4]) is set, PWM0 and PWM1 perform complementary PWM paired function; the paired PWM period, duty and Dead-time are determined by PWM0 timer and Dead-zone generator 0. Similarly, the complementary PWM pairs of (PWM2, PWM3) and (PWM4, PWM5) are controlled by PWM2 and PWM4 timers and Dead-zone generator 2 and 4, respectively. Refer from Figure 6.9-1 to Figure 6.9-6 for the architecture of PWM Timers.

To prevent PWM driving output pin with unsteady waveform, the 16-bit period down counter and 16-bit comparator are implemented with double buffer. When user writes data to counter/comparator buffer registers the updated value will be load into the 16-bit down counter/ comparator at the time down counter reaching 0. The double buffering feature avoids glitch at PWM outputs.

When the 16-bit period down counter reaches 0, the interrupt request is generated. If PWM-timer is set as auto-reload mode, when the down counter reaches 0, it is reloaded with PWM Counter Register (CNRx) automatically then start decreasing, repeatedly. If the PWM-timer is set as one-shot mode, the down counter will stop and generate one interrupt request when it reaches 0.

The value of PWM counter comparator is used for pulse high width modulation. The counter control logic changes the output to high level when down-counter value matches the value of compare register.

The alternate feature of the PWM-timer is digital input Capture function. If Capture function is enabled the PWM output pin is switched as capture input mode. The Capture0 and PWM0 share one timer which is included in PWM0 and the Capture1 and PWM1 share PWM1 timer, and etc. Therefore user must setup the PWM-timer before enable Capture feature. After capture feature is enabled, the capture always latched PWM-counter to Capture Rising Latch Register (CRLR) when input channel has a rising transition and latched PWM-counter to Capture Falling Latch Register (CFLR) when input channel has a falling transition. Capture channel 0 interrupt is programmable by setting CRL_IE0 (CCR0[1]) (Rising latch Interrupt enable) and CFL_IE0 (CCR0[2]) (Falling latch Interrupt enable) to decide the condition of interrupt occur. Capture channel 1 has the same feature by setting CRL_IE1 (CCR0[17]) and CFL_IE1 (CCR0[18]). And capture channel 2 to channel 3 on each group have the same feature by setting the corresponding control bits in CCR2. For each group, whenever Capture issues Interrupt 0/1/2/3, the PWM counter 0/1/2/3 will be reload at this moment.

The maximum captured frequency that PWM can capture is confined by the capture interrupt latency. When capture interrupt occurred, software will do at least three steps, including: Read PIIR to get interrupt source and Read CRLRx/CFLRx(x=0~3) to get capture value and finally write 1 to clear PIIR to 0. If interrupt latency will take time T0 to finish, the capture signal mustn't transition during this interval (T0). In this case, the maximum capture frequency will be 1/T0. For example:

HCLK = 50 MHz, PWM_CLK = 25 MHz, Interrupt latency is 900 ns

So the maximum capture frequency will be 1/900ns ≈ 1000 kHz

6.9.2 Features

6.9.2.1 PWM Function:

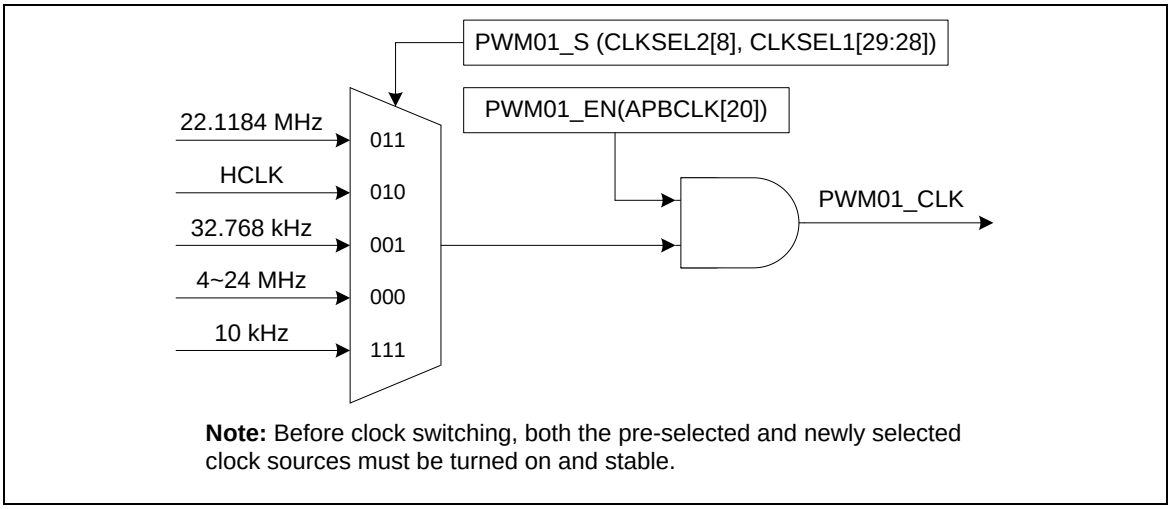
- Up to 2 PWM groups (PWMA/PWMB) to support 6 PWM channels or 3 complementary PWM paired channels
- PWM group A has two PWM generators and PWM group B has one PWM generator with each PWM generator supporting one 8-bit prescaler, two clock dividers, two PWM-timers, one Dead-zone generator and two PWM outputs.
- Up to 16-bit resolution
- PWM Interrupt request synchronized with PWM period
- One-shot or Auto-reload mode
- Edge-aligned type or Center-aligned type option
- PWM trigger ADC start-to-conversion

6.9.2.2 Capture Function:

- Timing control logic shared with PWM Generators
- Supports 6 Capture input channels shared with 6 PWM output channels
- Each channel supports one rising latch register (CRLR), one falling latch register (CFLR) and Capture interrupt flag (CAPIFx)

6.9.3 Block Diagram

Figure 6.9-1 to Figure 6.9-6 illustrates the architecture of PWM in pair (e.g. PWM-Timer 0/1 are in one pair and PWM-Timer 2/3 are in another one).



Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 6.9-1 PWM Generator 0 Clock Source Control

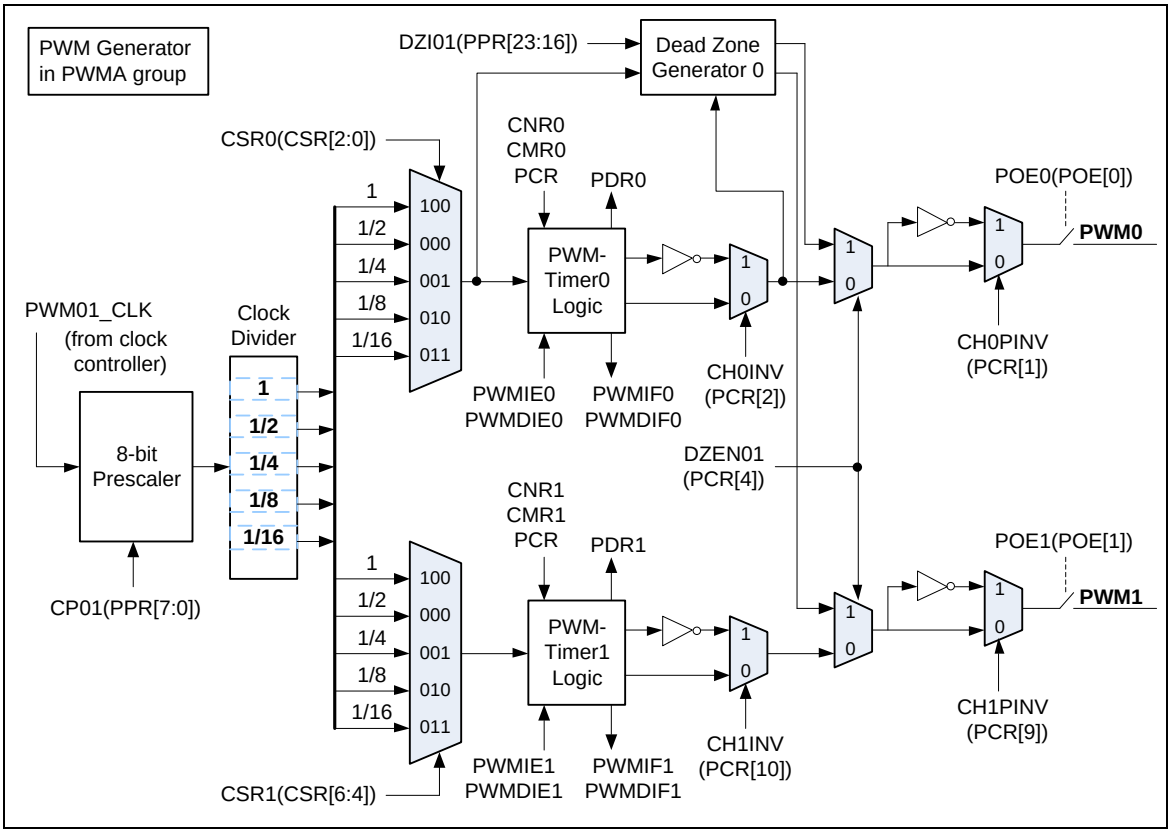


Figure 6.9-2 PWM Generator 0 Architecture Diagram

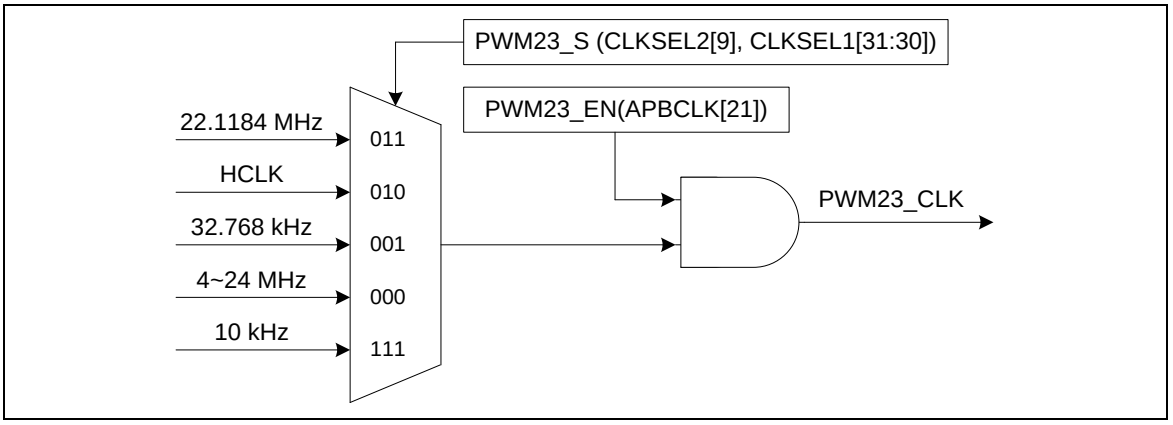


Figure 6.9-3 PWM Generator 2 Clock Source Control

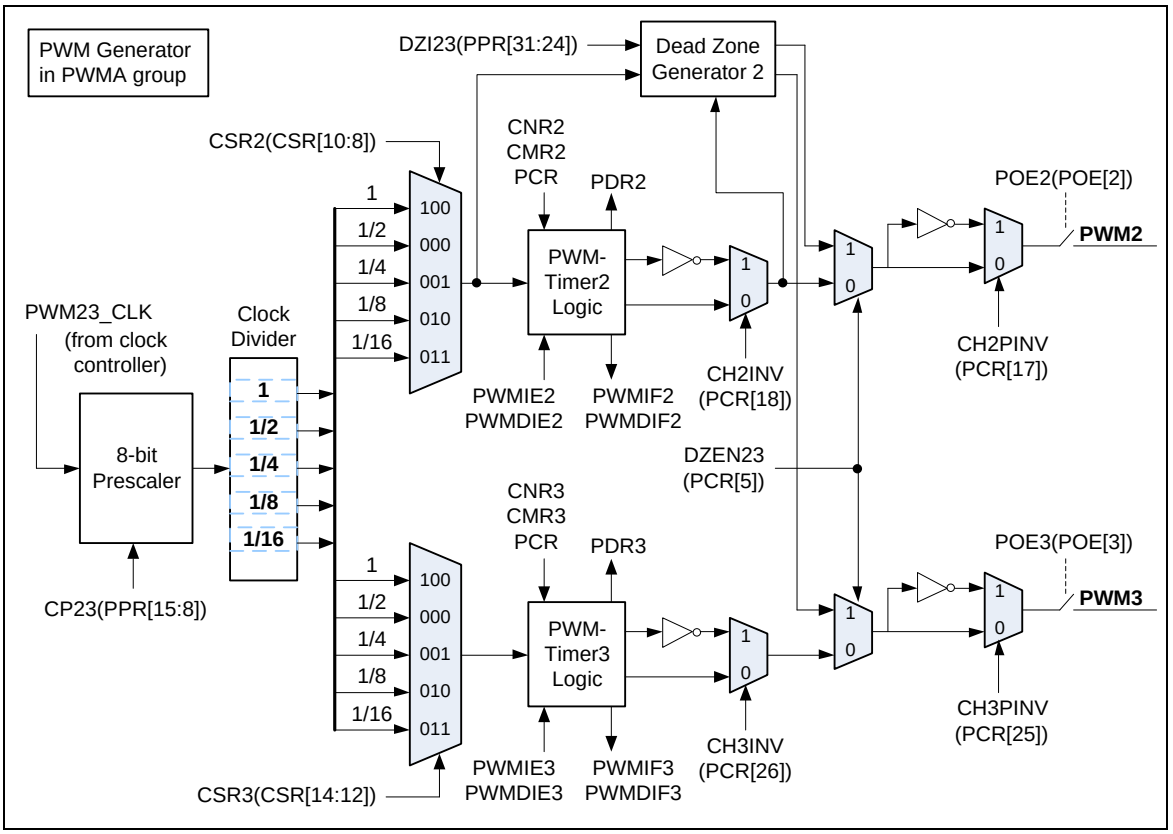


Figure 6.9-4 PWM Generator 2 Architecture Diagram

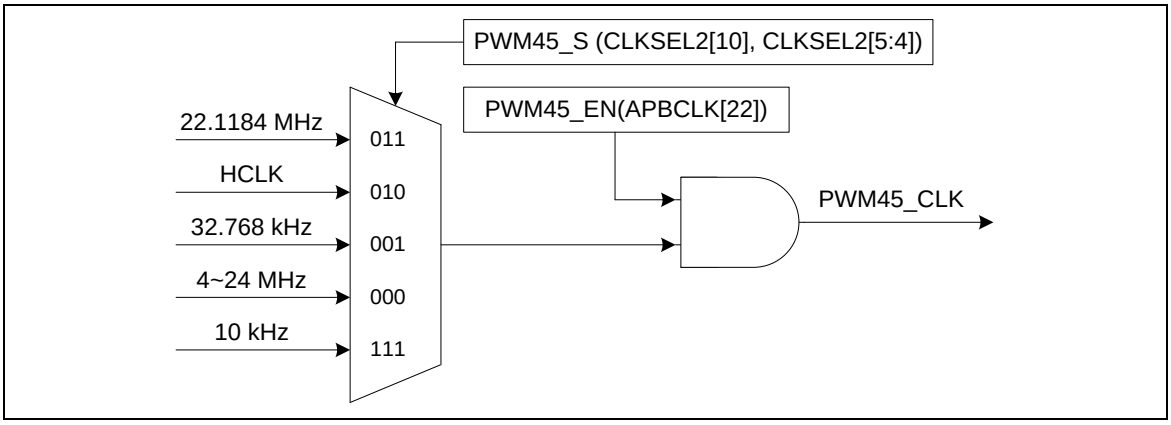


Figure 6.9-5 PWM Generator 4 Clock Source Control

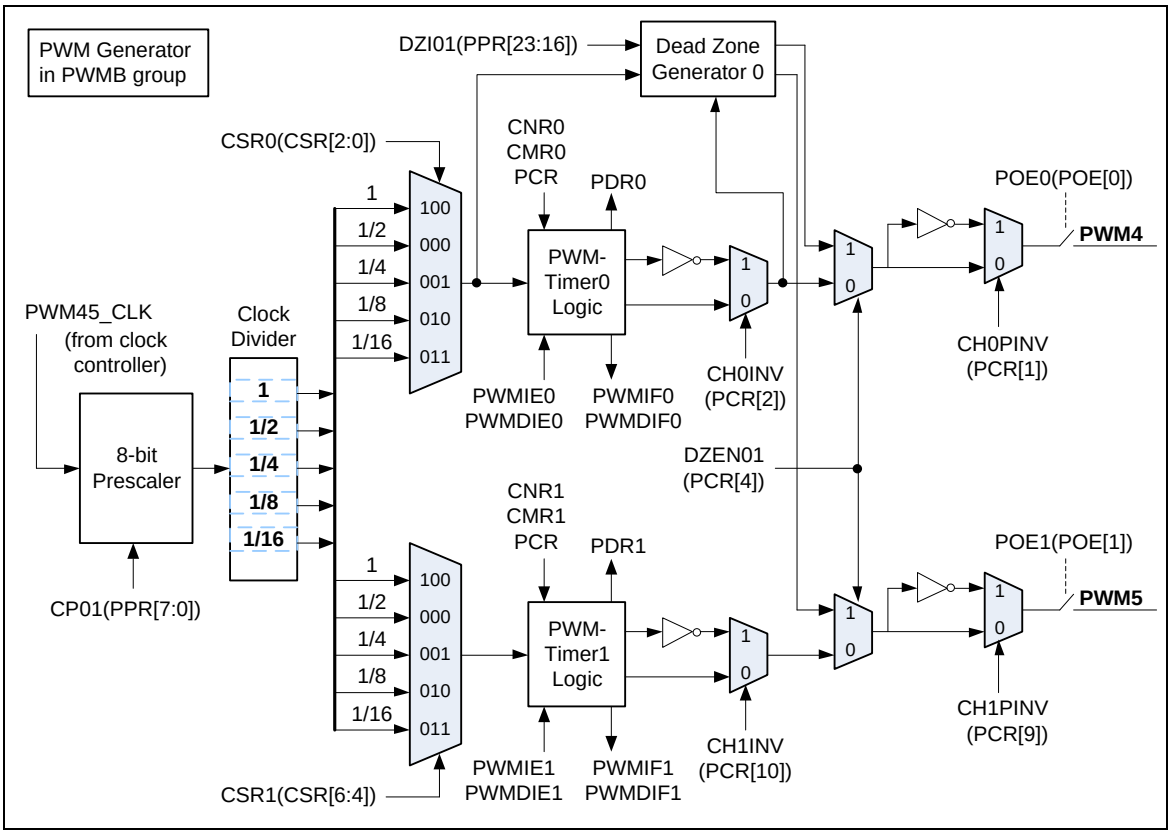


Figure 6.9-6 PWM Generator 4 Architecture Diagram

6.9.4 Basic Configuration

The PWM pin functions are configured in GPA_MFP, GPB_MFP and GPE_MFP registers.

The PWM clock can be enabled in APBCLK[22:20]. The PWM clock source is selected by CLKSEL1[31:28], CLKSEL2[5:4] and CLKSEL2[10:8].

6.9.5 Functional Description

6.9.5.1 PWM-Timer Operation

The PWM controller supports 2 operation types: Edge-aligned and Center-aligned type.

6.9.5.2 Edge-aligned PWM (down-counter)

In Edge-aligned PWM Output mode, the 16 bits PWM counter will starts down-counting from CNRn to match with the value of the duty cycle CMRn (old), when this happen it will toggle the PWMn generator output to low. The counter will continue down-counting to 0, at this moment, it toggles the PWMn generator output to high and CMRn(new) and CNRn(new) are updated with CHnMODE=1 and request the PWM interrupt if PWM interrupt is enabled(PWMIEn (PIER[3:0]) = 1).

The PWM period and duty control are configured by PWM down-counter register (CNR) and PWM comparator register (CMR). The PWM-timer timing operation is shown in Figure 6.9-8. The pulse width modulation follows the formula below and the legend of PWM-Timer Comparator is shown as Figure 6.9-7. Note that the corresponding GPIO pins must be configured as PWM function (enable POE and disable CAPENR) for the corresponding PWM channel.

- PWM frequency = PWMxy_CLK/[(prescale+1)*(clock divider)*(CNR+1)]; where xy, could be 01, 23 or 45, depends on selected PWM channel.
- Duty ratio = (CMR+1)/(CNR+1)
- CMR >= CNR: PWM output is always high
- CMR < CNR: PWM low width= (CNR-CMR) unit[1]; PWM high width = (CMR+1) unit
- CMR = 0: PWM low width = (CNR) unit; PWM high width = 1 unit

Note [1]: unit = one PWM clock cycle.

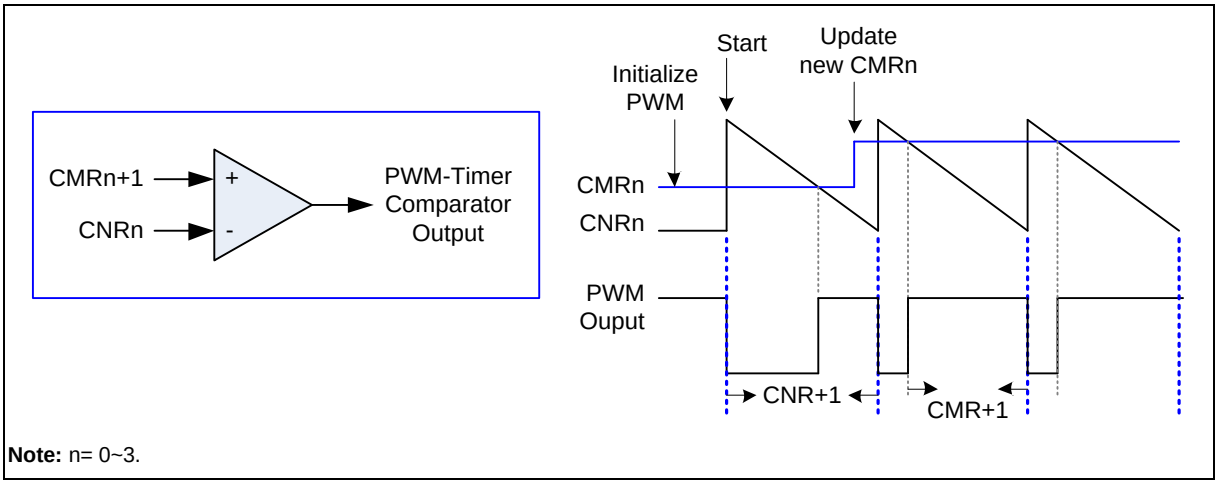


Figure 6.9-7 Legend of Internal Comparator Output of PWM-Timer

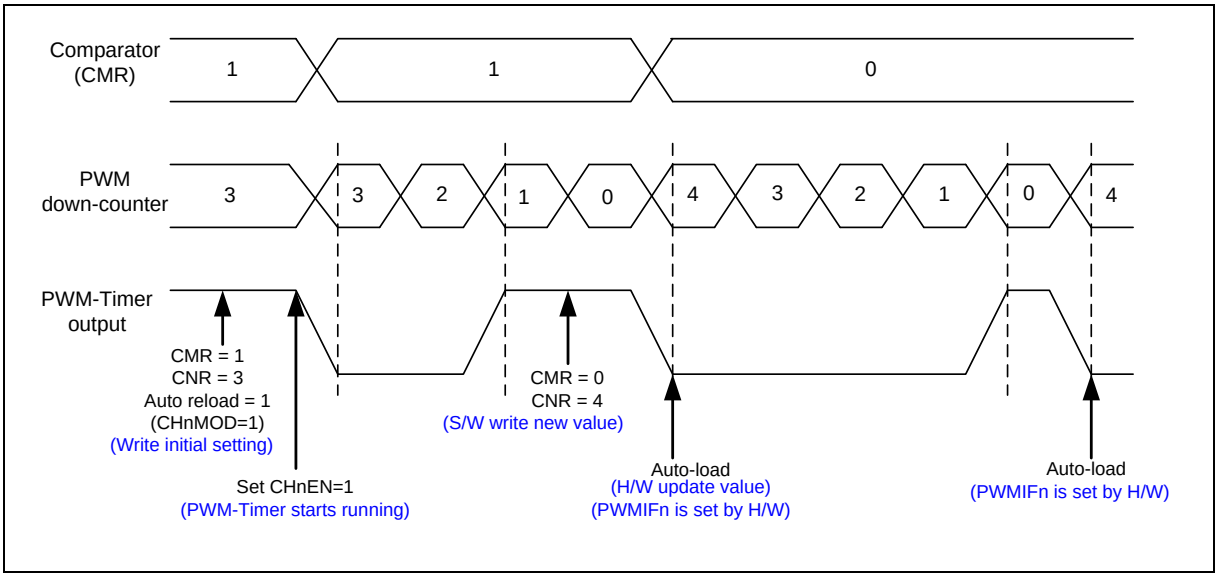


Figure 6.9-8 PWM-Timer Operation Timing

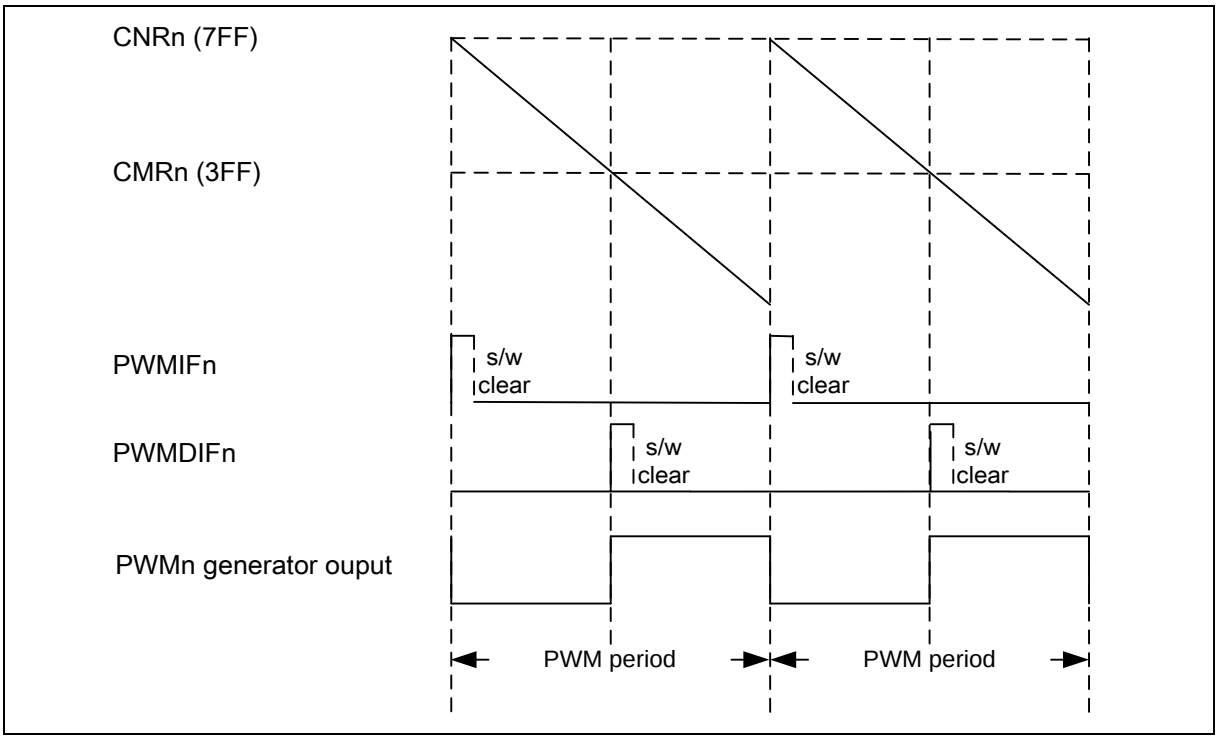


Figure 6.9-9 PWM Edge-aligned Interrupt Generate Timing Waveform

6.9.5.3 Center-aligned PWM (up/down-counter)

The Center-aligned PWM signals are produced by the module when the PWM time base is configured in an Up/Down Counting mode. The PWM counter will start counting-up from 0 to match the value of CMRn (old); this will cause the toggling of the PWMn generator output to low. The counter will continue counting to match with the CNRn (old). Upon reaching this states counter is configured automatically to down counting, when PWM counter matches the CMRn (old) value again the PWMn generator output toggles to high. Once the PWM counter underflows it will update the PWM period register CNRn(new) and duty cycle register CMRn(new) with CHnMODE = 1.

In Center-aligned type, the PWM period interrupt is requested at down-counter underflow if INTxxTYPE (PIER[17:16]) =0, i.e. at start (end) of each PWM cycle or at up-counter matching with CNRn if INTxxTYPE (PIER[17:16]) = 1, i.e. at center point of PWM cycle.

- PWM frequency = PWMxy_CLK/[(prescale+1)*(clock divider)*(CNR+1)]; where xy, could be 01, 23 or 45, depends on selected PWM channel.
- Duty ratio = [(2 x CMR) + 1]/[2 x (CNR+1)]
- CMR > CNR: PWM output is always high
- CMR <= CNR: PWM low width= 2 x (CNR-CMR) + 1 unit[1]; PWM high width = (2 x CMR) + 1 unit
- CMR = 0: PWM low width = 2 x CNR + 1 unit; PWM high width = 1 unit

  Note [1]: unit = one PWM clock cycle.

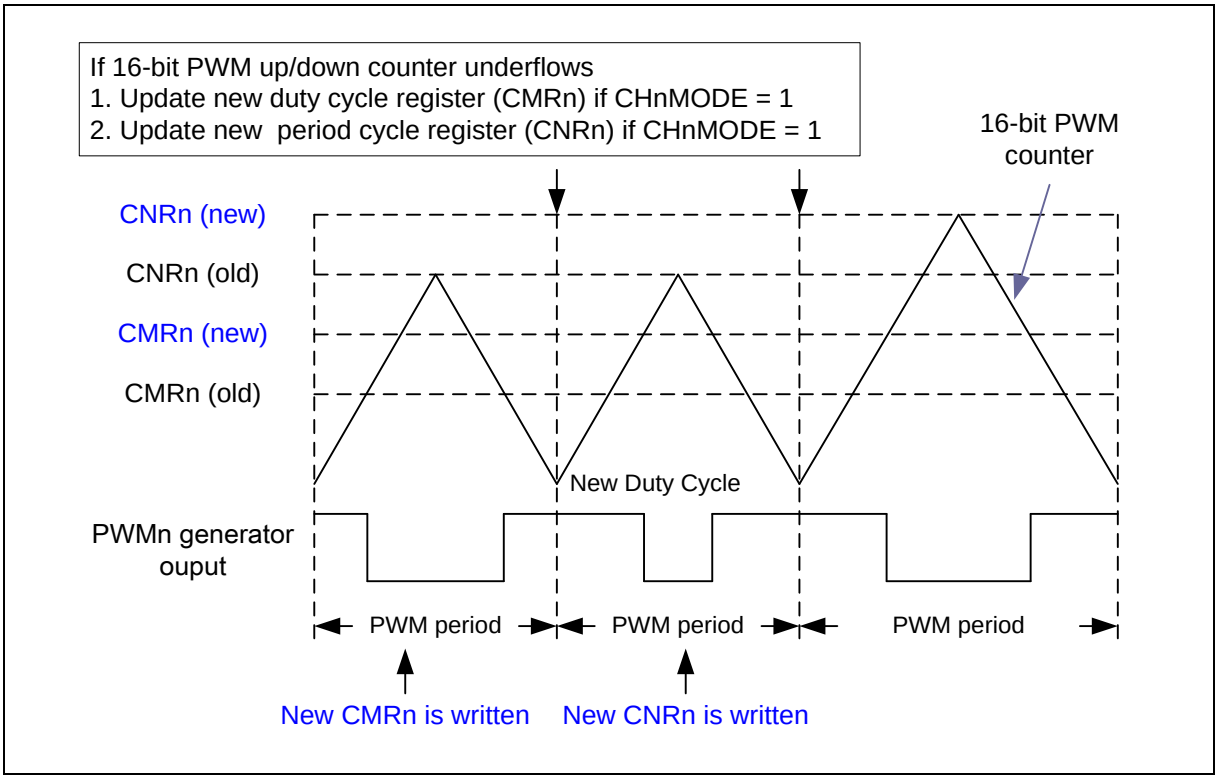


Figure 6.9-10 Center-aligned Type Output Waveform

In Center-aligned type, system can generate period interrupt, at two specified timings. PWM period interrupt is generated at counter equals zero on down-count if INTxxTYPE (PIER[17:16]) = 0 or at counter equals CNRx on up-count if INTxxTYPE (PIER[17:16]) = 1, i.e. at center point of PWM cycle.

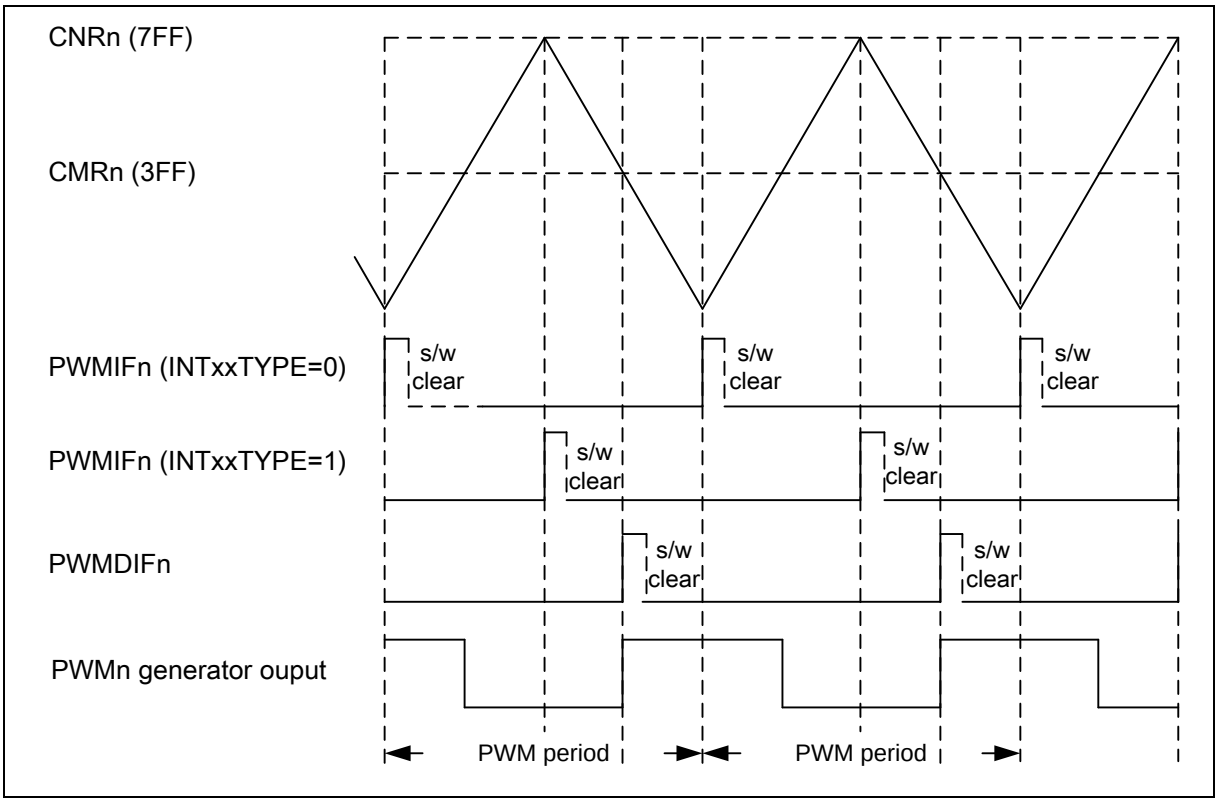


Figure 6.9-11 PWM Center-aligned Interrupt Generate Timing Waveform

6.9.5.4 PWM Double Buffering, Auto-reload and One-shot Operation

PWM Timers have double buffering function and the reload value is updated at the start of next period without affecting current timer operation. The PWM counter value can be written into CNRx and current PWM counter value can be read from PDRx.

PWM0 will operate at One-shot mode if CH0MOD (PCR[3]) bit is set to 0, and operate at Auto-reload mode if CH0MOD (PCR[3]) bit is set to 1. It is recommend that switch PWM0 operating mode before set CH0EN (PCR[0]) bit to 1 to enable PWM0 counter start running because the content of CNR0 and CMR0 will be cleared to 0 to reset the PWM0 period and duty setting when PWM0 operating mode is changed. As PWM0 operate at One-shot mode, CMR0 and CNR0 should be written first and then set CH0EN (PCR[0]) bit to 1 to enable PWM0 counter start running. After PWM0 counter down count from CNR0 value to 0, CNR0 and CMR0 will be cleared to 0 by hardware and PWM counter will be held. Software need to write new CMR0 and CNR0 value to set next one-shot period and duty. When re-start next one-shot operation, the CMR0 should be written first because PWM0 counter will auto re-start counting when CNR0 is written a non-zero value. As PWM0 operates at auto-reload mode, CMR0 and CNR0 should be written first and then set CH0EN (PCR[0]) bit to 1 to enable PWM0 counter start running. The value of CNR0 will reload to PWM0 counter when it down count reaches 0. If CNR0 is set to 0, PWM0 counter will be held. PWM1~PWM5 performs the same function as PWM0.

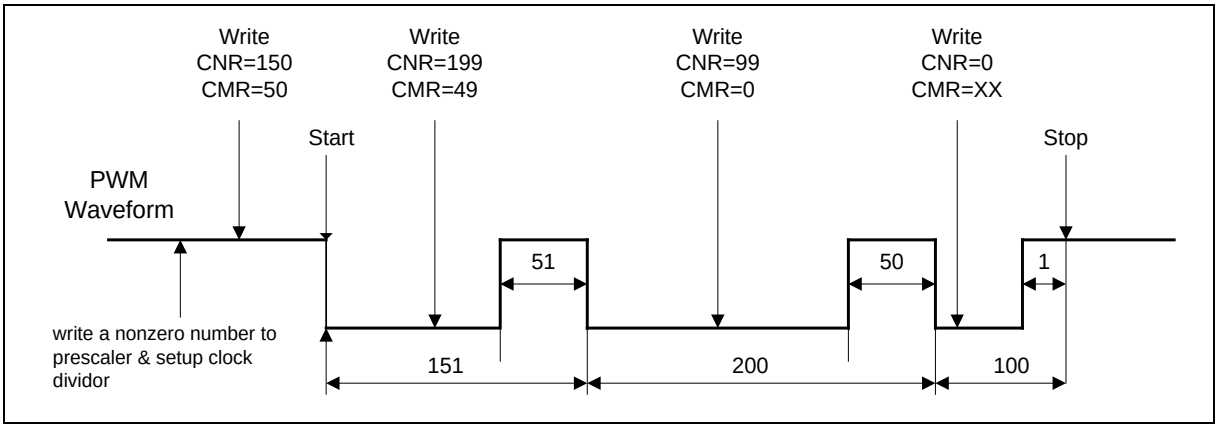


Figure 6.9-12 PWM Double Buffering Illustration

6.9.5.5 Modulate Duty Ratio

The double buffering function allows CMRn written at any point in current cycle. The loaded value will take effect from next cycle.

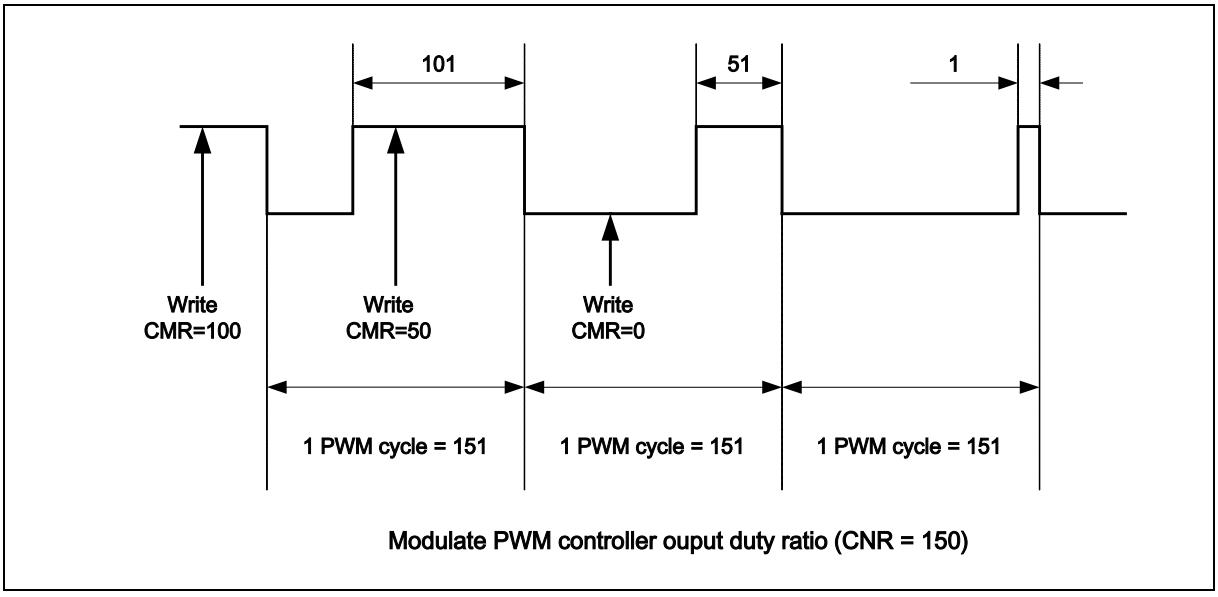


Figure 6.9-13 PWM Controller Output Duty Ratio

6.9.5.6 Dead-Zone Generator

The PWM controller is implemented with Dead-zone generator. They are built for power device protection. This function generates a programmable time gap to delay PWM rising output. User can program DZIxx (PPR[31:16]) to determine the Dead-zone interval.

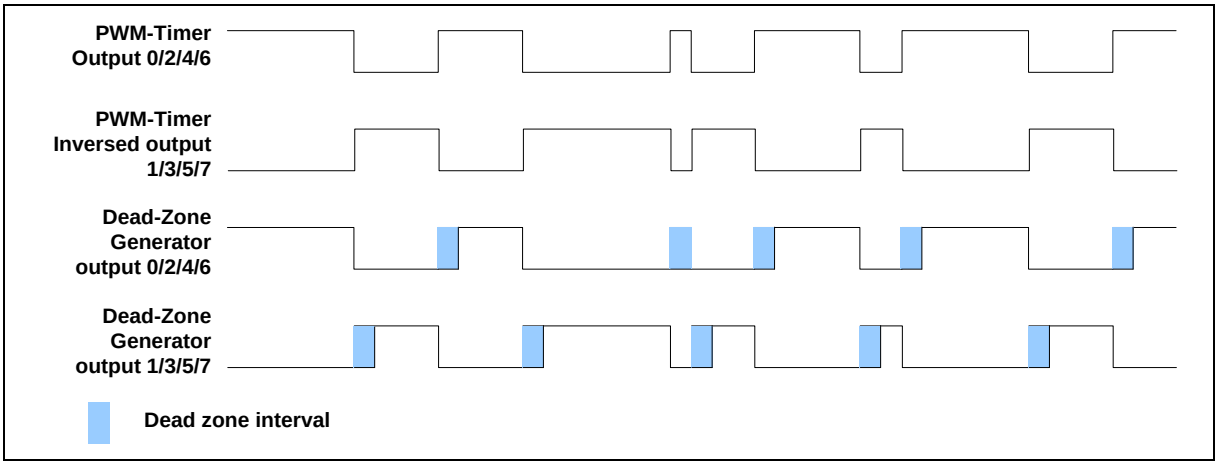


Figure 6.9-14 Paired-PWM Output with Dead-zone Generation Operation

6.9.5.7 PWM Center-aligned Trigger ADC Function

PWM can trigger ADC to start conversion when PWM counter up count to CNR in Center-aligned type by setting PWMnTEN (TCON[3:0]) to “1”.

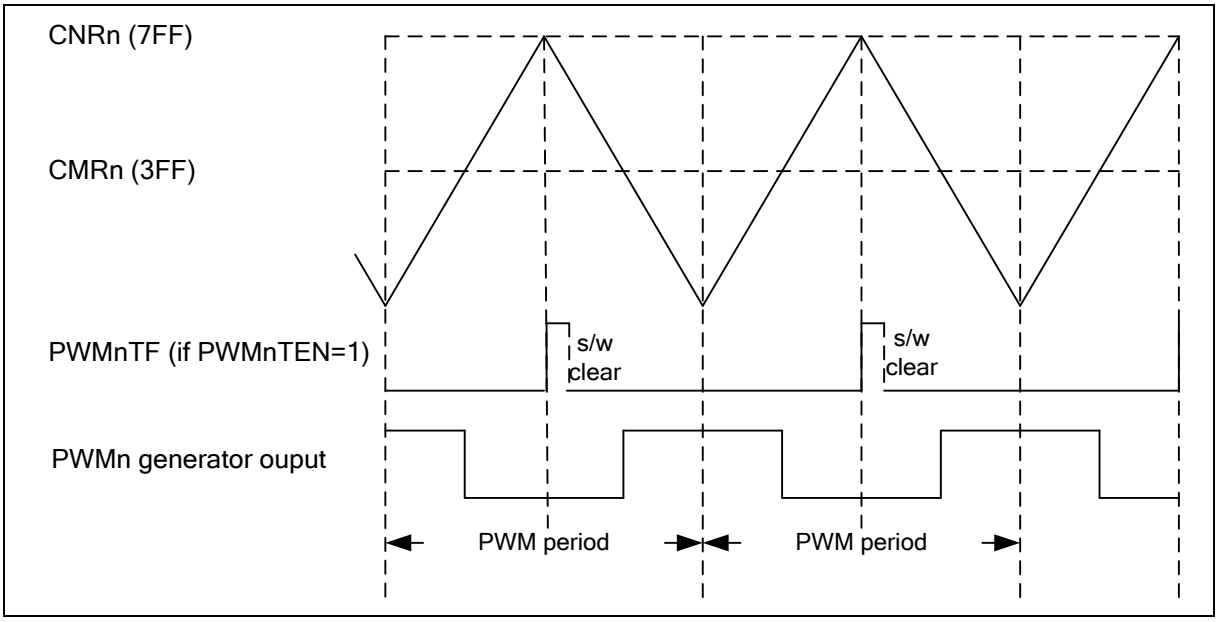


Figure 6.9-15 PWM trigger ADC to conversion in Center-aligned type Timing Waveform

6.9.5.8 Capture Operation

The Capture 0 and PWM 0 share one timer that included in PWM 0; and the Capture 1 and PWM 1 share another timer, and etc. The capture always latches PWM-counter to CRLRn when input channel has a rising transition and latches PWM-counter to CFLRn when input channel has a falling transition. Capture channel 0 interrupt is programmable by setting CRL_IE0 (CCR0[1]) (Rising latch Interrupt enable) and CFL_IE0 (CCR0[2]) (Falling latch Interrupt enable) to decide the condition of interrupt occur. Capture channel 1 has the same feature by setting CRL_IE1 (CCR0[17]) and CFL_IE1 (CCR0[18]), and etc. Whenever the Capture controller issues a capture interrupt, the corresponding PWM counter will be reloaded with CNRn at this moment. Note that the corresponding GPIO pins must be configured as capture function (POE disabled and

CAPENR enabled) for the corresponding capture channel.

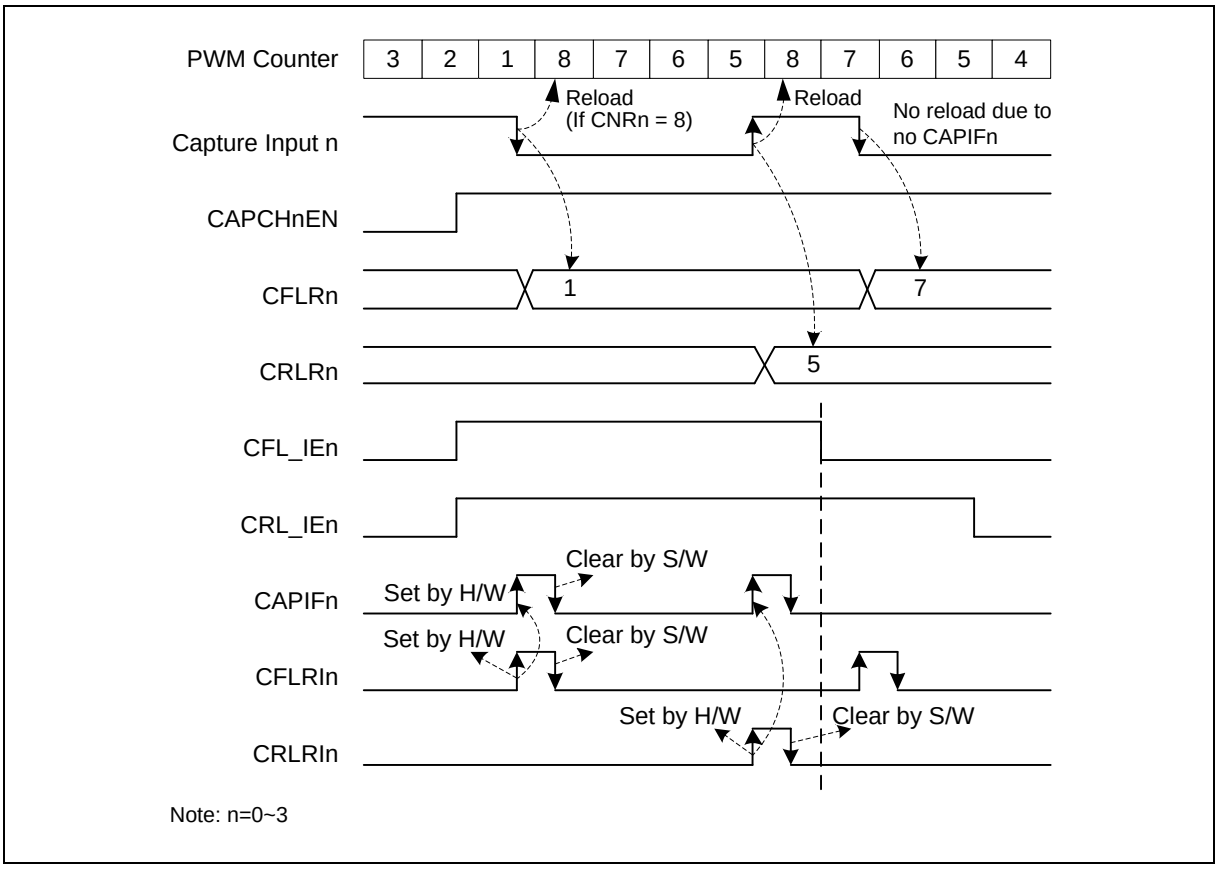


Figure 6.9-16 Capture Operation Timing

In this case, the CNR is 8:

- The PWM counter will be reloaded with CNRn when a capture interrupt flag (CAPIFn) is set.
- The channel low pulse width is (CNR + 1 - CRLR).
- The channel high pulse width is (CNR + 1 - CFLR).

6.9.5.9 PWM-Timer Interrupt Architecture

There are eight PWM interrupts, PWM0_INT~PWM5_INT, which are divided into PWMA_INT and PWMB_INT for Advanced Interrupt Controller (AIC). PWM 0 and Capture 0 share one interrupt, PWM1 and Capture 1 share the same interrupt and so on. Therefore, PWM function and Capture function in the same channel cannot be used at the same time. Figure 6.9-17 and Figure 6.9-18 demonstrates the architecture of PWM-Timer interrupts.

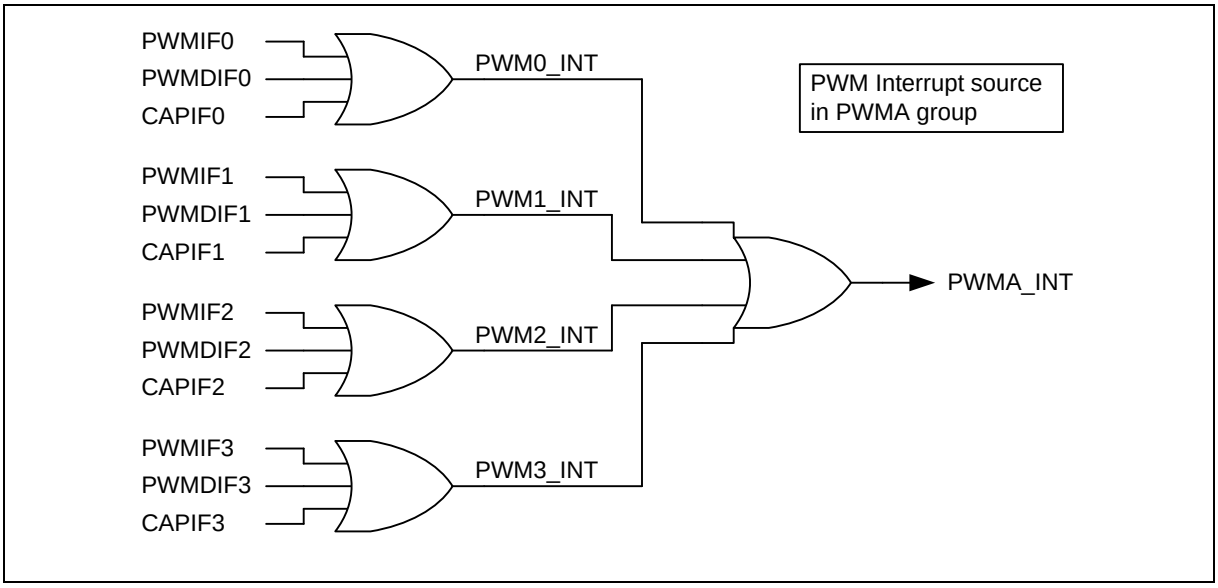


Figure 6.9-17 PWM Group A PWM-Timer Interrupt Architecture Diagram

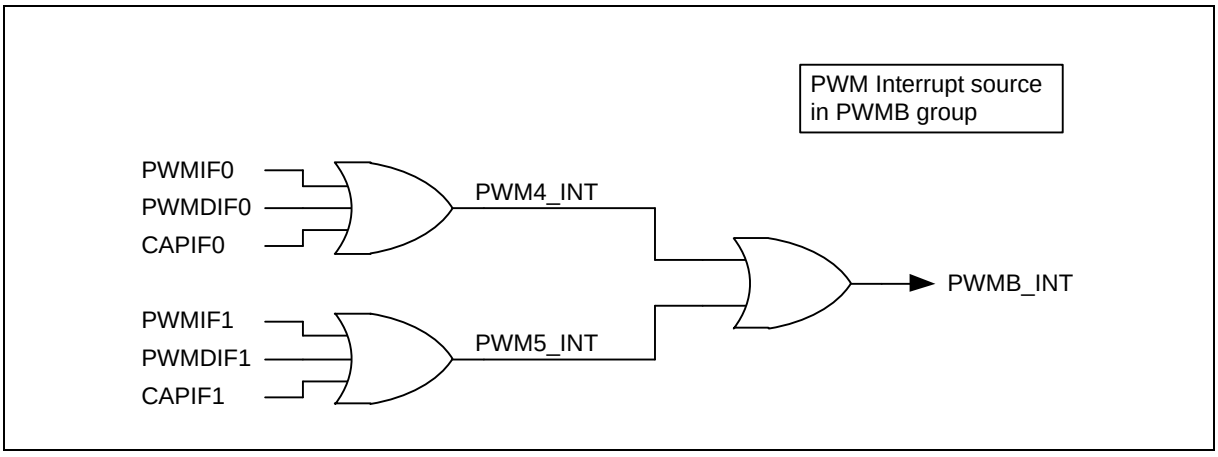


Figure 6.9-18 PWM Group B PWM-Timer Interrupt Architecture Diagram

6.9.5.10 PWM-Timer Start Procedure

The following procedure is recommended for starting a PWM drive.

1. Setup clock source divider select register (CSR)
2. Wait until SYNCBUSYn be set to 0 by hardware (if PWM clock source is not from HCLK)
3. Setup prescaler (PPR)
4. Wait until SYNCBUSYn be set to 0 by hardware (if PWM clock source is not from HCLK)
5. Setup inverter on/off, Dead-zone generator on/off, Auto-reload/One-shot mode and Stop PWM-timer (PCR)
6. Wait until SYNCBUSYn be set to 0 by hardware (if PWM clock source is not from HCLK)
7. Setup comparator register (CMR) for setting PWM duty.
8. Wait until SYNCBUSYn be set to 0 by hardware (if PWM clock source is not from HCLK)
9. Setup PWM down-counter register (CNR) for setting PWM period.
10. Setup interrupt enable register (PIER) (optional)
11. Setup corresponding GPIO pins as PWM function (enable POE and disable CAPENR) for the corresponding PWM channel.
12. Enable PWM timer start running (Set CHnEN = 1 in PCR)

6.9.5.11 Modify PWM counter register (CNR), comparator register (CMR), Clock prescaler(CP01/CP23) and PWM operation mode(CHnMOD in PCR register bit3) Procedure

The following procedure is recommended for modifying CNR/CMR/Clock prescaler/PWM operation mode.

1. Wait until SYNCBUSYn be set to 0 by hardware (if PWM clock source is not from HCLK)
2. Modify CMRn/CNRn/CHnMOD/CP01/CP23

6.9.5.12 PWM-Timer Re-Start Procedure in Single-shot mode

After PWM waveform is generated once in PWM One-shot mode, PWM-Timer will be stopped automatically and both of CNR and CMR will be cleared by hardware. Software must fill CMR and CNR value again to re-start another PWM one-shot waveform. The following procedure is recommended for re-starting PWM one-shot waveform.

1. Wait until SYNCBUSYn be set to 0 by hardware (if PWM clock source is not from HCLK)
2. Setup comparator register (CMR) for setting PWM duty.
3. Wait until SYNCBUSYn be set to 0 by hardware (if PWM clock source is not from HCLK)
4. Setup PWM down-counter register (CNR) for setting PWM period. After setup CNR, PWM wave will be generated.

6.9.5.13 PWM-Timer Stop Procedure

Method 1:

Set 16-bit counter (CNR) as 0, and monitor PDR (current value of 16-bit down-counter). When PDR reaches to 0, disable PWM-Timer (CHnEN in PCR). ***(Recommended)***

Method 2:

Set 16-bit counter (CNR) as 0. When interrupt request happened, disable PWM-Timer (CHnEN in PCR). ***(Recommended)***

Method 3:

Disable PWM-Timer directly ((CHnEN in PCR). ***(Not recommended)***

The reason why method 3 is not recommended is that disable CHnEN will immediately stop PWM output signal and lead to change the duty of the PWM output, this may cause damage to the control circuit of motor

6.9.5.14 Capture Start Procedure

1. Setup clock source divider select register (CSR)
2. Wait until SYNCBUSYn be set to 0 by hardware (if PWM clock source is not from HCLK)
3. Setup prescaler (PPR)
4. Setup channel enabled, rising/falling interrupt enable and input signal inverter on/off (CCR0, CCR2)
5. Wait until SYNCBUSYn be set to 0 by hardware (if PWM clock source is not from HCLK)
6. Setup Auto-reload mode, Edge-aligned type and Stop PWM-timer (PCR)
7. Wait until SYNCBUSYn be set to 0 by hardware (if PWM clock source is not from HCLK)
8. Setup PWM down-counter (CNR)
9. Enable PWM timer start running (Set CHnEN = 1 in PCR)
10. Setup corresponding GPIO pins as capture function (disable POE and enable CAPENR) for the corresponding PWM channel.

6.9.6 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 group A** **PWMA_BA = 0x4004_0000** ● **PWM group B** **PWMB_BA = 0x4014_0000**				
PPR	PWMA_BA+0x00 PWMB_BA+0x00	R/W	PWM Prescaler Register	0x0000_0000
CSR	PWMA_BA+0x04 PWMB_BA+0x04	R/W	PWM Clock Source Divider Select Register	0x0000_0000
PCR	PWMA_BA+0x08 PWMB_BA+0x08	R/W	PWM Control Register	0x0000_0000
CNR0	PWMA_BA+0x0C PWMB_BA+0x0C	R/W	PWM Counter Register 0	0x0000_0000
CMR0	PWMA_BA+0x10 PWMB_BA+0x10	R/W	PWM Comparator Register 0	0x0000_0000
PDR0	PWMA_BA+0x14 PWMB_BA+0x14	R	PWM Data Register 0	0x0000_0000
CNR1	PWMA_BA+0x18 PWMB_BA+0x18	R/W	PWM Counter Register 1	0x0000_0000
CMR1	PWMA_BA+0x1C PWMB_BA+0x1C	R/W	PWM Comparator Register 1	0x0000_0000
PDR1	PWMA_BA+0x20 PWMB_BA+0x20	R	PWM Data Register 1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM Counter Register 2	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM Comparator Register 2	0x0000_0000
PDR2	PWMA_BA+0x2C	R	PWM Data Register 2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM Counter Register 3	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM Comparator Register 3	0x0000_0000
PDR3	PWMA_BA+0x38	R	PWM Data Register 3	0x0000_0000
PBCR	PWMA_BA+0x3C PWMB_BA+0x3C	R/W	PWM Backward Compatible Register	0x0000_0000
PIER	PWMA_BA+0x40 PWMB_BA+0x40	R/W	PWM Interrupt Enable Register	0x0000_0000
PIIR	PWMA_BA+0x44 PWMB_BA+0x44	R/W	PWM Interrupt Indication Register	0x0000_0000

CCR0	PWMA_BA+0x50 PWMB_BA+0x50	R/W	PWM Capture Control Register 0	0x0000_0000
CCR2	PWMA_BA+0x54	R/W	PWM Capture Control Register 2	0x0000_0000
CRLR0	PWMA_BA+0x58 PWMB_BA+0x58	R	PWM Capture Rising Latch Register (Channel 0)	0x0000_0000
CFLR0	PWMA_BA+0x5C PWMB_BA+0x5C	R	PWM Capture Falling Latch Register (Channel 0)	0x0000_0000
CRLR1	PWMA_BA+0x60 PWMB_BA+0x60	R	PWM Capture Rising Latch Register (Channel 1)	0x0000_0000
CFLR1	PWMA_BA+0x64 PWMB_BA+0x64	R	PWM Capture Falling Latch Register (Channel 1)	0x0000_0000
CRLR2	PWMA_BA+0x68	R	PWM Capture Rising Latch Register (Channel 2)	0x0000_0000
CFLR2	PWMA_BA+0x6C	R	PWM Capture Falling Latch Register (Channel 2)	0x0000_0000
CRLR3	PWMA_BA+0x70	R	PWM Capture Rising Latch Register (Channel 3)	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM Capture Falling Latch Register (Channel 3)	0x0000_0000
CAPENR	PWMA_BA+0x78 PWMB_BA+0x78	R/W	PWM Capture Input 0~3 Enable Register	0x0000_0000
POE	PWMA_BA+0x7C PWMB_BA+0x7C	R/W	PWM Output Enable for Channel 0~3	0x0000_0000
TCON	PWMA_BA+0x80 PWMB_BA+0x80	R/W	PWM Trigger Control for Channel 0~3	0x0000_0000
TSTATUS	PWMA_BA+0x84 PWMB_BA+0x84	R/W	PWM Trigger Status Register	0x0000_0000
SYNCBUSY0	PWMA_BA+0x88 PWMB_BA+0x88	R	PWM0 Synchronous Busy Status Register	0x0000_0000
SYNCBUSY1	PWMA_BA+0x8C PWMB_BA+0x8C	R	PWM1 Synchronous Busy Status Register	0x0000_0000
SYNCBUSY2	PWMA_BA+0x90	R	PWM2 Synchronous Busy Status Register	0x0000_0000
SYNCBUSY3	PWMA_BA+0x94	R	PWM3 Synchronous Busy Status Register	0x0000_0000

6.9.7 Register Description

PWM Prescale Register (PPR)

Register	Offset	R/W	Description	Reset Value
PPR	PWMA_BA+0x00 PWMB_BA+0x00	R/W	PWM Prescaler Register	0x0000_0000

31	30	29	28	27	26	25	24
DZI23							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DZI01							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CP23							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CP01							

Bits	Description	
[31:24]	**DZI23**	**Dead-Zone Interval For Pair Of Channel2 And Channel3 (PWM2 And PWM3 Pair For PWM Group A)** These 8-bit determine the Dead-zone length. The unit time of Dead-zone length = [(prescale+1)*(clock source divider)]/ PWM23_CLK.
[23:16]	**DZI01**	**Dead-Zone Interval For Pair Of Channel 0 And Channel 1 (PWM0 And PWM1 Pair For PWM Group A, PWM4 And PWM5 Pair For PWM Group B)** These 8-bit determine the Dead-zone length. The unit time of Dead-zone length = [(prescale+1)*(clock source divider)]/ PWMxy_CLK (where xy could be 01 or 45, depends on selected PWM channel.).
[15:8]	**CP23**	**Clock Prescaler 2 (PWM-Timer2 / 3 For Group A)** Clock input is divided by (CP23 + 1) before it is fed to the corresponding PWM-timer If CP23=0, then the clock prescaler 2 output clock will be stopped. So corresponding PWM-timer will also be stopped.
[7:0]	**CP01**	**Clock Prescaler 0 (PWM-Timer 0 / 1 For Group A And PWM-Timer 4 / 5 For Group B)** Clock input is divided by (CP01 + 1) before it is fed to the corresponding PWM-timer If CP01=0, then the clock prescaler 0 output clock will be stopped. So corresponding PWM-timer will also be stopped.

PWM Clock Source Divider Select Register (CSR)

Register	Offset	R/W	Description	Reset Value
CSR	PWMA_BA+0x04 PWMB_BA+0x04	R/W	PWM Clock Source Divider 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	CSR3			Reserved	CSR2		
7	6	5	4	3	2	1	0
Reserved	CSR1			Reserved	CSR0		

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**CSR3**	**PWM Timer 3 Clock Source Divider Selection (PWM Timer 3 For Group A)** Select clock source divider for PWM timer 3. 000 = 2. 001 = 4. 010 = 8. 011 = 16. 100 = 1.
[11]	**Reserved**	Reserved.
[10:8]	**CSR2**	**PWM Timer 2 Clock Source Divider Selection (PWM Timer 2 For Group A)** Select clock source divider for PWM timer 2. (Table is the same as CSR3)
[7]	**Reserved**	Reserved.
[6:4]	**CSR1**	**PWM Timer 1 Clock Source Divider Selection (PWM Timer 1 For Group A And PWM Timer 5 For Group B)** Select clock source divider for PWM timer 1. (Table is the same as CSR3)
[3]	**Reserved**	Reserved.
[2:0]	**CSR0**	**PWM Timer 0 Clock Source Divider Selection (PWM Timer 0 For Group A And PWM Timer 4 For Group B)** Select clock source divider for PWM timer 0. (Table is the same as CSR3)

PWM Control Register (PCR)

Register	Offset	R/W	Description	Reset Value
PCR	PWMA_BA+0x08 PWMB_BA+0x08	R/W	PWM Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PWM23TYPE	PWM01TYPE	Reserved		CH3MOD	CH3INV	CH3PINV	CH3EN
23	22	21	20	19	18	17	16
Reserved				CH2MOD	CH2INV	CH2PINV	CH2EN
15	14	13	12	11	10	9	8
Reserved				CH1MOD	CH1INV	CH1PINV	CH1EN
7	6	5	4	3	2	1	0
Reserved		DZEN23	DZEN01	CH0MOD	CH0INV	CH0PINV	CH0EN

Bits	Description	
[31]	**PWM23TYPE**	**PWM23 Aligned Type Selection (PWM2 And PWM3 Pair For PWM Group A)** 0 = Edge-aligned type. 1 = Center-aligned type.
[30]	**PWM01TYPE**	**PWM01 Aligned Type Selection (PWM0 And PWM1 Pair For PWM Group A, PWM4 And PWM5 Pair For PWM Group B)** 0 = Edge-aligned type. 1 = Center-aligned type.
[30:28]	**Reserved**	Reserved.
[27]	**CH3MOD**	**PWM-Timer 3 Auto-Reload/One-Shot Mode (PWM Timer 3 For Group A)** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause CNR3 and CMR3 be cleared.
[26]	**CH3INV**	**PWM-Timer 3 Output Inverter Enable (PWM Timer 3 For Group A)** 0 = Inverter Disabled. 1 = Inverter Enabled.
[25]	**CH3PINV**	**PWM-Timer 3 Output Polar Inverse Enable (PWM Timer 3 For Group A)** 0 = PWM3 output polar inverse Disable. 1 = PWM3 output polar inverse Enable.
[24]	**CH3EN**	**PWM-Timer 3 Enable (PWM Timer 3 For Group A)** 0 = Corresponding PWM-Timer Stopped. 1 = Corresponding PWM-Timer Start Running.
[23:20]	**Reserved**	Reserved.
[19]	**CH2MOD**	**PWM-Timer 2 Auto-Reload/One-Shot Mode (PWM Timer 2 For Group A)** 0 = One-shot mode. 1 = Auto-reload mode.

		Note: If there is a transition at this bit, it will cause CNR2 and CMR2 be cleared.
[18]	**CH2INV**	**PWM-Timer 2 Output Inverter Enable (PWM Timer 2 For Group A)** 0 = Inverter Disabled. 1 = Inverter Enabled.
[17]	**CH2PINV**	**PWM-Timer 2 Output Polar Inverse Enable (PWM Timer 2 For Group A)** 0 = PWM2 output polar inverse Disabled. 1 = PWM2 output polar inverse Enabled.
[16]	**CH2EN**	**PWM-Timer 2 Enable (PWM Timer 2 For Group A)** 0 = Corresponding PWM-Timer Stopped. 1 = Corresponding PWM-Timer Start Running.
[15:12]	**Reserved**	Reserved.
[11]	**CH1MOD**	**PWM-Timer 1 Auto-Reload/One-Shot Mode (PWM Timer 1 For Group A And PWM Timer 5 For Group B)** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause CNR1 and CMR1 be cleared.
[10]	**CH1INV**	**PWM-Timer 1 Output Inverter Enable (PWM Timer 1 For Group A And PWM Timer 5 For Group B)** 0 = Inverter Disable. 1 = Inverter Enable.
[9]	**CH1PINV**	**PWM-Timer 1 Output Polar Inverse Enable (PWM Timer 1 For Group A And PWM Timer 5 For Group B)** 0 = PWM1 output polar inverse Disabled. 1 = PWM1 output polar inverse Enabled.
[8]	**CH1EN**	**PWM-Timer 1 Enable (PWM Timer 1 For Group A And PWM Timer 5 For Group B)** 0 = Corresponding PWM-Timer Stopped. 1 = Corresponding PWM-Timer Start Running.
[7:6]	**Reserved**	Reserved.
[5]	**DZEN23**	**Dead-Zone 2 Generator Enable (PWM2 And PWM3 Pair For PWM Group A)** 0 = Disabled. 1 = Enabled. **Note:** When Dead-zone generator is enabled, the pair of PWM2 and PWM3 becomes a complementary pair for PWM group A.
[4]	**DZEN01**	**Dead-Zone 0 Generator Enable (PWM0 And PWM1 Pair For PWM Group A, PWM4 And PWM5 Pair For PWM Group B)** 0 = Disabled. 1 = Enabled. **Note:** When Dead-zone generator is enabled, the pair of PWM0 and PWM1 becomes a complementary pair for PWM group A and the pair of PWM4 and PWM5 becomes a complementary pair for PWM group B.
[3]	**CH0MOD**	**PWM-Timer 0 Auto-Reload/One-Shot Mode (PWM Timer 0 For Group A And PWM Timer 4 For Group B)** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause CNR0 and CMR0 be cleared.
[2]	**CH0INV**	**PWM-Timer 0 Output Inverter Enable (PWM Timer 0 For Group A And PWM Timer 4 For Group B)**

		0 = Inverter Disabled. 1 = Inverter Enabled.
[1]	**CH0PINV**	**PWM-Timer 0 Output Polar Inverse Enable (PWM Timer 0 For Group A And PWM Timer 4 For Group B)** 0 = PWM0 output polar inverse Disabled. 1 = PWM0 output polar inverse Enabled.
[0]	**CH0EN**	**PWM-Timer 0 Enable (PWM Timer 0 For Group A And PWM Timer 4 For Group B)** 0 = The corresponding PWM-Timer stops running. 1 = The corresponding PWM-Timer starts running.

PWM Counter Register 3-0 (CNR3-0)

Register	Offset	R/W	Description	Reset Value
CNR0	PWMA_BA+0x0C PWMB_BA+0x0C	R/W	PWM Counter Register 0	0x0000_0000
CNR1	PWMA_BA+0x18 PWMB_BA+0x18	R/W	PWM Counter Register 1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM Counter Register 2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM Counter Register 3	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNRx**	**PWM Timer Loaded Value** CNR determines the PWM period. PWM frequency = PWMxy_CLK/[(prescale+1)*(clock divider)*(CNR+1)]; where xy, could be 01, 23 or 45, depends on selected PWM channel. For Edge-aligned type: ● Duty ratio = (CMR+1)/(CNR+1). ● CMR >= CNR: PWM output is always high. ● CMR < CNR: PWM low width = (CNR-CMR) unit; PWM high width = (CMR+1) unit. ● CMR = 0: PWM low width = (CNR) unit; PWM high width = 1 unit. For Center-aligned type: ● Duty ratio = [(2 x CMR) + 1]/[2 x (CNR+1)]. ● CMR > CNR: PWM output is always high. ● CMR <= CNR: PWM low width = 2 x (CNR-CMR) + 1 unit; PWM high width = (2 x CMR) + 1 unit. ● CMR = 0: PWM low width = 2 x CNR + 1 unit; PWM high width = 1 unit. (Unit = one PWM clock cycle). **Note:** Any write to CNR will take effect in next PWM cycle. **Note:** When PWM operating at Center-aligned type, CNR value should be set between 0x0000 to 0xFFFE. If CNR equal to 0xFFFF, the PWM will work unpredictable. **Note:** When CNR value is set to 0, PWM output is always high.

PWM Comparator Register 3-0 (CMR3-0)

Register	Offset	R/W	Description	Reset Value
CMR0	PWMA_BA+0x10 PWMB_BA+0x10	R/W	PWM Comparator Register 0	0x0000_0000
CMR1	PWMA_BA+0x1C PWMB_BA+0x1C	R/W	PWM Comparator Register 1	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM Comparator Register 2	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM Comparator Register 3	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMRx**	**PWM Comparator Register** CMR determines the PWM duty. PWM frequency = PWMxy_CLK/[(prescale+1)*(clock divider)*(CNR+1)]; where xy, could be 01, 23 or 45, depends on selected PWM channel. For Edge-aligned type: ● Duty ratio = (CMR+1)/(CNR+1). ● CMR >= CNR: PWM output is always high. ● CMR < CNR: PWM low width = (CNR-CMR) unit; PWM high width = (CMR+1) unit. ● CMR = 0: PWM low width = (CNR) unit; PWM high width = 1 unit. For Center-aligned type: ● Duty ratio = [(2 x CMR) + 1]/[2 x (CNR+1)]. ● CMR > CNR: PWM output is always high. ● CMR <= CNR: PWM low width = 2 x (CNR-CMR) + 1 unit; PWM high width = (2 x CMR) + 1 unit. ● CMR = 0: PWM low width = 2 x CNR + 1 unit; PWM high width = 1 unit. (Unit = one PWM clock cycle). **Note:** Any write to CNR will take effect in next PWM cycle.

PWM Data Register 3-0 (PDR 3-0)

Register	Offset	R/W	Description	Reset Value
PDR0	PWMA_BA+0x14 PWMB_BA+0x14	R	PWM Data Register 0	0x0000_0000
PDR1	PWMA_BA+0x20 PWMB_BA+0x20	R	PWM Data Register 1	0x0000_0000
PDR2	PWMA_BA+0x2C	R	PWM Data Register 2	0x0000_0000
PDR3	PWMA_BA+0x38	R	PWM Data Register 3	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PDRx**	**PWM Data Register** User can monitor PDR to know the current value in 16-bit counter.

PWM Backward Compatible Register (PBCR)

Register	Offset	R/W	Description	Reset Value
PBCR	PWMA_BA+0x3C PWMB_BA+0x3C	R/W	PWM Backward Compatible Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**BCn**	**PWM Backward Compatible Register** 0 = Configure write 0 to clear CFLRI0~3 and CRLRI0~3. 1 = Configure write 1 to clear CFLRI0~3 and CRLRI0~3. Refer to the CCR0/CCR2 register bit 6, 7, 22, 23 description **Note:** It is recommended that this bit be set to 1 to prevent CFLRIx and CRLRIx from being cleared when writing CCR0/CCR2.

PWM Interrupt Enable Register (PIER)

Register	Offset	R/W	Description	Reset Value
PIER	PWMA_BA+0x40 PWMB_BA+0x40	R/W	PWM Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**INT23TYPE**	**INT01TYPE**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**PWMDIE3**	**PWMDIE2**	**PWMDIE1**	**PWMDIE0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PWMIE3**	**PWMIE2**	**PWMIE1**	**PWMIE0**

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**INT23TYPE**	**PWM23 Interrupt Period Type Selection Bit (PWM2 And PWM3 Pair For PWM Group A)** 0 = PWMIFn will be set if PWM counter underflow. 1 = PWMIFn will be set if PWM counter matches CNRn register. **Note:** This bit is effective when PWM in Center-aligned type only.
[16]	**INT01TYPE**	**PWM01 Interrupt Period Type Selection Bit (PWM0 And PWM1 Pair For PWM Group A, PWM4 And PWM5 Pair For PWM Group B)** 0 = PWMIFn will be set if PWM counter underflow. 1 = PWMIFn will be set if PWM counter matches CNRn register. **Note:** This bit is effective when PWM in Center-aligned type only.
[11]	**PWMDIE3**	**PWM Channel 3 Duty Interrupt Enable Bit** 0 = Disabled. 1 = Enabled.
[10]	**PWMDIE2**	**PWM Channel 2 Duty Interrupt Enable Bit** 0 = Disabled. 1 = Enabled.
[9]	**PWMDIE1**	**PWM Channel 1 Duty Interrupt Enable Bit** 0 = Disabled. 1 = Enabled.
[8]	**PWMDIE0**	**PWM Channel 0 Duty Interrupt Enable Bit** 0 = Disabled. 1 = Enabled.
[7:4]	**Reserved**	Reserved.
[3]	**PWMIE3**	**PWM Channel 3 Period Interrupt Enable Bit**

		0 = Disabled. 1 = Enabled.
[2]	**PWMIE2**	**PWM Channel 2 Period Interrupt Enable Bit** 0 = Disabled. 1 = Enabled.
[1]	**PWMIE1**	**PWM Channel 1 Period Interrupt Enable Bit** 0 = Disabled. 1 = Enabled.
[0]	**PWMIE0**	**PWM Channel 0 Period Interrupt Enable Bit** 0 = Disabled. 1 = Enabled.

PWM Interrupt Indication Register (PIIR)

Register	Offset	R/W	Description	Reset Value
PIIR	PWMA_BA+0x44 PWMB_BA+0x44	R/W	PWM Interrupt Indication Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				PWMDIF3	PWMDIF2	PWMDIF1	PWMDIF0
7	6	5	4	3	2	1	0
Reserved				PWMIF3	PWMIF2	PWMIF1	PWMIF0

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**PWMDIF3**	**PWM Channel 3 Duty Interrupt Flag** Flag is set by hardware when channel 3 PWM counter down count and reaches CMR3, software can clear this bit by writing a one to it. **Note:** If CMR equal to CNR, this flag is not working in Edge-aligned type selection
[10]	**PWMDIF2**	**PWM Channel 2 Duty Interrupt Flag** Flag is set by hardware when channel 2 PWM counter down count and reaches CMR2, software can clear this bit by writing a one to it. **Note:** If CMR equal to CNR, this flag is not working in Edge-aligned type selection
[9]	**PWMDIF1**	**PWM Channel 1 Duty Interrupt Flag** Flag is set by hardware when channel 1 PWM counter down count and reaches CMR1, software can clear this bit by writing a one to it. **Note:** If CMR equal to CNR, this flag is not working in Edge-aligned type selection
[8]	**PWMDIF0**	**PWM Channel 0 Duty Interrupt Flag** Flag is set by hardware when channel 0 PWM counter down count and reaches CMR0, software can clear this bit by writing a one to it. **Note:** If CMR equal to CNR, this flag is not working in Edge-aligned type selection
[7:4]	**Reserved**	Reserved.
[3]	**PWMIF3**	**PWM Channel 3 Period Interrupt Status** This bit is set by hardware when PWM3 counter reaches the requirement of interrupt (depend on INT23TYPE bit of PIER register), software can write 1 to clear this bit to 0.
[2]	**PWMIF2**	**PWM Channel 2 Period Interrupt Status** This bit is set by hardware when PWM2 counter reaches the requirement of interrupt (depend on INT23TYPE bit of PIER register), software can write 1 to clear this bit to 0.
[1]	**PWMIF1**	**PWM Channel 1 Period Interrupt Status** This bit is set by hardware when PWM1 counter reaches the requirement of interrupt (depend on INT01TYPE bit of PIER register), software can write 1 to clear this bit to 0.

[0]	**PWMIF0**	**PWM Channel 0 Period Interrupt Status** This bit is set by hardware when PWM0 counter reaches the requirement of interrupt (depend on INT01TYPE bit of PIER register), software can write 1 to clear this bit to 0.

Note: User can clear each interrupt flag by writing 1 to corresponding bit in PIIR.

Capture Control Register (CCR0)

Register	Offset	R/W	Description	Reset Value
CCR0	PWMA_BA+0x50 PWMB_BA+0x50	R/W	PWM Capture Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CFLRI1	CRLRI1	Reserved	CAPIF1	CAPCH1EN	CFL_IE1	CRL_IE1	INV1
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CFLRI0	CRLRI0	Reserved	CAPIF0	CAPCH0EN	CFL_IE0	CRL_IE0	INV0

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**CFLRI1**	**CFLR1 Latched Indicator Bit** When PWM group input channel 1 has a falling transition, CFLR1 was latched with the value of PWM down-counter and this bit is set by hardware. Software can write 0 to clear this bit to 0 if BCn bit is 0, and can write 1 to clear this bit to 0 if BCn bit is 1.
[22]	**CRLRI1**	**CRLR1 Latched Indicator Bit** When PWM group input channel 1 has a rising transition, CRLR1 was latched with the value of PWM down-counter and this bit is set by hardware. Software can write 0 to clear this bit to 0 if BCn bit is 0, and can write 1 to clear this bit to0 if BCn bit is 1.
[5]	**Reserved**	Reserved.
[20]	**CAPIF1**	**Channel 1 Capture Interrupt Indication Flag** If PWM group channel 1 rising latch interrupt is enabled (CRL_IE1 = 1), a rising transition occurs at PWM group channel 1 will result in CAPIF1 to high; Similarly, a falling transition will cause CAPIF1 to be set high if PWM group channel 1 falling latch interrupt is enabled (CFL_IE1 = 1). Write 1 to clear this bit to 0.
[19]	**CAPCH1EN**	**Channel 1 Capture Function Enable Bit** 0 = Capture function on PWM group channel 1 Disabled. 1 = Capture function on PWM group channel 1 Enabled. When Enabled, Capture latched the PWM-counter and saved to CRLR (Rising latch) and CFLR (Falling latch). When Disabled, Capture does not update CRLR and CFLR, and disable PWM group channel 1 Interrupt.
[18]	**CFL_IE1**	**Channel 1 Falling Latch Interrupt Enable Bit** 0 = Falling latch interrupt Disabled. 1 = Falling latch interrupt Enabled. When Enabled, if Capture detects PWM group channel 1 has falling transition, Capture will

		issue an Interrupt.
[17]	**CRL_IE1**	**Channel 1 Rising Latch Interrupt Enable Bit** 0 = Rising latch interrupt Disabled. 1 = Rising latch interrupt Enabled. When Enabled, if Capture detects PWM group channel 1 has rising transition, Capture will issue an Interrupt.
[16]	**INV1**	**Channel 1 Inverter Enable Bit** 0 = Inverter Disabled. 1 = Inverter Enabled. Reverse the input signal from GPIO before fed to Capture timer
[15:8]	**Reserved**	Reserved.
[7]	**CFLRI0**	**CFLR0 Latched Indicator** When PWM group input channel 0 has a falling transition, CFLR0 was latched with the value of PWM down-counter and this bit is set by hardware. Software can write 0 to clear this bit to 0 if the BCn bit is 0, and can write 1 to clear this bit to0 if BCn bit is 1.
[6]	**CRLRI0**	**CRLR0 Latched Indicator** When PWM group input channel 0 has a rising transition, CRLR0 was latched with the value of PWM down-counter and this bit is set by hardware. Software can write 0 to clear this bit to 0 if the BCn bit is 0, and can write 1 to clear this bit to 0 if the BCn bit is 1.
[5]	**Reserved**	Reserved.
[4]	**CAPIF0**	**Channel 0 Capture Interrupt Indication Flag** If PWM group channel 0 rising latch interrupt is enabled (CRL_IE0 = 1), a rising transition occurs at PWM group channel 0 will result in CAPIF0 to high; Similarly, a falling transition will cause CAPIF0 to be set high if PWM group channel 0 falling latch interrupt is enabled (CFL_IE0 = 1). Write 1 to clear this bit to 0.
[3]	**CAPCH0EN**	**Channel 0 Capture Function Enable** 0 = Capture function on PWM group channel 0 Disabled. 1 = Capture function on PWM group channel 0 Enabled. When Enabled, Capture latched the PWM-counter value and saved to CRLR (Rising latch) and CFLR (Falling latch). When Disabled, Capture does not update CRLR and CFLR, and disable PWM group channel 0 Interrupt.
[2]	**CFL_IE0**	**Channel 0 Falling Latch Interrupt Enable Bit** 0 = Falling latch interrupt Disabled. 1 = Falling latch interrupt Enabled. When Enabled, if Capture detects PWM group channel 0 has falling transition, Capture will issue an Interrupt.
[1]	**CRL_IE0**	**Channel 0 Rising Latch Interrupt Enable Bit** 0 = Rising latch interrupt Disabled. 1 = Rising latch interrupt Enabled. When Enabled, if Capture detects PWM group channel 0 has rising transition, Capture will issue an Interrupt.
[0]	**INV0**	**Channel 0 Inverter Enable Bit** 0 = Inverter Disabled. 1 = Inverter Enabled. Reverse the input signal from GPIO before fed to Capture timer

Capture Control Register (CCR2)

Register	Offset	R/W	Description	Reset Value
CCR2	PWMA_BA+0x54	R/W	PWM Capture Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CFLRI3	CRLRI3	Reserved	CAPIF3	CAPCH3EN	CFL_IE3	CRL_IE3	INV3
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CFLRI2	CRLRI2	Reserved	CAPIF2	CAPCH2EN	CFL_IE2	CRL_IE2	INV2

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**CFLRI3**	**CFLR3 Latched Indicator** When PWM group input channel 3 has a falling transition, CFLR3 was latched with the value of PWM down-counter and this bit is set by hardware. Software can write 0 to clear this bit to 0 if the BCn bit is 0, and can write 1 to clear this bit to 0 if the BCn bit is 1.
[22]	**CRLRI3**	**CRLR3 Latched Indicator** When PWM group input channel 3 has a rising transition, CRLR3 was latched with the value of PWM down-counter and this bit is set by hardware. Software can write 0 to clear this bit to 0 if the BCn bit is 0, and can write 1 to clear this bit to 0 if the BCn bit is 1.
[21]	**Reserved**	Reserved.
[20]	**CAPIF3**	**Channel 3 Capture Interrupt Indication Flag** If PWM group channel 3 rising latch interrupt is enabled (CRL_IE3=1), a rising transition occurs at PWM group channel 3 will result in CAPIF3 to high; Similarly, a falling transition will cause CAPIF3 to be set high if PWM group channel 3 falling latch interrupt is enabled (CFL_IE3=1). Write 1 to clear this bit to 0
[19]	**CAPCH3EN**	**Channel 3 Capture Function Enable Bit** 0 = Capture function on PWM group channel 3 Disabled. 1 = Capture function on PWM group channel 3 Enabled. When Enabled, Capture latched the PWM-counter and saved to CRLR (Rising latch) and CFLR (Falling latch). When Disabled, Capture does not update CRLR and CFLR, and disable PWM group channel 3 Interrupt.
[18]	**CFL_IE3**	**Channel 3 Falling Latch Interrupt Enable Bit** 0 = Falling latch interrupt Disabled. 1 = Falling latch interrupt Enabled. When Enabled, if Capture detects PWM group channel 3 has falling transition, Capture will

		issue an Interrupt.
[17]	**CRL_IE3**	**Channel 3 Rising Latch Interrupt Enable Bit** 0 = Rising latch interrupt Disabled. 1 = Rising latch interrupt Enabled. When Enabled, if Capture detects PWM group channel 3 has rising transition, Capture will issue an Interrupt.
[16]	**INV3**	**Channel 3 Inverter Enable Bit** 0 = Inverter Disabled. 1 = Inverter Enabled. Reverse the input signal from GPIO before fed to Capture timer
[15:8]	**Reserved**	Reserved.
[7]	**CFLRI2**	**CFLR2 Latched Indicator** When PWM group input channel 2 has a falling transition, CFLR2 was latched with the value of PWM down-counter and this bit is set by hardware. Software can write 0 to clear this bit to 0 if BCn bit is 0, and can write 1 to clear this bit to 0 if the BCn bit is 1.
[6]	**CRLRI2**	**CRLR2 Latched Indicator** When PWM group input channel 2 has a rising transition, CRLR2 was latched with the value of PWM down-counter and this bit is set by hardware. Software can write 0 to clear this bit to 0 if the BCn bit is 0, and can write 1 to clear this bit to 0 if the BCn bit is 1.
[5]	**Reserved**	Reserved.
[4]	**CAPIF2**	**Channel 2 Capture Interrupt Indication Flag** If PWM group channel 2 rising latch interrupt is enabled (CRL_IE2=1), a rising transition occurs at PWM group channel 2 will result in CAPIF2 to high; Similarly, a falling transition will cause CAPIF2 to be set high if PWM group channel 2 falling latch interrupt is enabled (CFL_IE2=1). Write 1 to clear this bit to 0
[3]	**CAPCH2EN**	**Channel 2 Capture Function Enable Bit** 0 = Capture function on PWM group channel 2 Disabled. 1 = Capture function on PWM group channel 2 Enabled. When Enabled, Capture latched the PWM-counter value and saved to CRLR (Rising latch) and CFLR (Falling latch). When Disabled, Capture does not update CRLR and CFLR, and disable PWM group channel 2 Interrupt.
[2]	**CFL_IE2**	**Channel 2 Falling Latch Interrupt Enable Bit** 0 = Falling latch interrupt Disabled. 1 = Falling latch interrupt Enabled. When Enabled, if Capture detects PWM group channel 2 has falling transition, Capture will issue an Interrupt.
[1]	**CRL_IE2**	**Channel 2 Rising Latch Interrupt Enable Bit** 0 = Rising latch interrupt Disabled. 1 = Rising latch interrupt Enabled. When Enabled, if Capture detects PWM group channel 2 has rising transition, Capture will issue an Interrupt.
[0]	**INV2**	**Channel 2 Inverter Enable Bit** 0 = Inverter Disabled. 1 = Inverter Enabled. Reverse the input signal from GPIO before fed to Capture timer

Capture Rising Latch Register3-0 (CRLR3-0)

Register	Offset	R/W	Description	Reset Value
CRLR0	PWMA_BA+0x58 PWMB_BA+0x58	R	PWM Capture Rising Latch Register (Channel 0)	0x0000_0000
CRLR1	PWMA_BA+0x60 PWMB_BA+0x60	R	PWM Capture Rising Latch Register (Channel 1)	0x0000_0000
CRLR2	PWMA_BA+0x68	R	PWM Capture Rising Latch Register (Channel 2)	0x0000_0000
CRLR3	PWMA_BA+0x70	R	PWM Capture Rising Latch Register (Channel 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**
CRLRx [15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CRLRx							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CRLRx**	**Capture Rising Latch Register** Latch the PWM counter when Channel 0/1/2/3 has rising transition.

Capture Falling Latch Register3-0 (CFLR3-0)

Register	Offset	R/W	Description	Reset Value
CFLR0	PWMA_BA+0x5C PWMB_BA+0x5C	R	PWM Capture Falling Latch Register (Channel 0)	0x0000_0000
CFLR1	PWMA_BA+0x64 PWMB_BA+0x64	R	PWM Capture Falling Latch Register (Channel 1)	0x0000_0000
CFLR2	PWMA_BA+0x6C	R	PWM Capture Falling Latch Register (Channel 2)	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM Capture Falling Latch Register (Channel 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**
CFLRx							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CFLRx							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CFLRx**	**Capture Falling Latch Register** Latch the PWM counter when Channel 0/1/2/3 has Falling transition.

Capture Input Enable Register (CAPENR)

Register	Offset	R/W	Description	Reset Value
CAPENR	PWMA_BA+0x78 PWMB_BA+0x78	R/W	PWM Capture Input 0~3 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				CINEN3	CINEN2	CINEN1	CINEN0

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**CINEN3**	**Channel 3 Capture Input Enable Bit** 0 = PWM Channel 3 capture input path Disabled. The input of PWM channel 3 capture function is always regarded as 0. 1 = PWM Channel 3 capture input path Enabled. The input of PWM channel 3 capture function comes from correlative multifunction pin if GPIO multi-function is set as PWM3.
[2]	**CINEN2**	**Channel 2 Capture Input Enable Bit** 0 = PWM Channel 2 capture input path Disabled. The input of PWM channel 2 capture function is always regarded as 0. 1 = PWM Channel 2 capture input path Enabled. The input of PWM channel 2 capture function comes from correlative multifunction pin if GPIO multi-function is set as PWM2.
[1]	**CINEN1**	**Channel 1 Capture Input Enable Bit** 0 = PWM Channel 1 capture input path Disabled. The input of PWM channel 1 capture function is always regarded as 0. 1 = PWM Channel 1 capture input path Enabled. The input of PWM channel 1 capture function comes from correlative multifunction pin if GPIO multi-function is set as PWM1.
[0]	**CINEN0**	**Channel 0 Capture Input Enable Bit** 0 = PWM Channel 0 capture input path Disabled. The input of PWM channel 0 capture function is always regarded as 0. 1 = PWM Channel 0 capture input path Enabled. The input of PWM channel 0 capture function comes from correlative multifunction pin if GPIO multi-function is set as PWM0.

PWM Output Enable Register (POE)

Register	Offset	R/W	Description	Reset Value
POE	PWMA_BA+0x7C PWMB_BA+0x7C	R/W	PWM Output Enable for Channel 0~3	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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**POE3**	**POE2**	**POE1**	**POE0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**POE3**	**Channel 3 Output Enable Bit** 0 = PWM channel 3 output to pin Disabled. 1 = PWM channel 3 output to pin Enabled. **Note:** The corresponding GPIO pin must also be switched to PWM function
[2]	**POE2**	**Channel 2 Output Enable Bit** 0 = PWM channel 2 output to pin Disabled. 1 = PWM channel 2 output to pin Enabled. **Note:** The corresponding GPIO pin must also be switched to PWM function
[1]	**POE1**	**Channel 1 Output Enable Bit** 0 = PWM channel 1 output to pin Disabled. 1 = PWM channel 1 output to pin Enabled. **Note:** The corresponding GPIO pin must also be switched to PWM function
[0]	**POE0**	**Channel 0 Output Enable Bit** 0 = PWM channel 0 output to pin Disabled. 1 = PWM channel 0 output to pin Enabled. **Note:** The corresponding GPIO pin must also be switched to PWM function

PWM Trigger Control Register (TCON)

Register	Offset	R/W	Description	Reset Value
TCON	PWMA_BA+0x80 PWMB_BA+0x80	R/W	PWM Trigger Control for Channel 0~3	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
Reserved							
7	6	5	4	3	2	1	0
Reserved				PWM3TEN	PWM2TEN	PWM1TEN	PWM0TEN

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PWM3TEN**	**Channel 3 Center-Aligned Trigger Enable Bit** 0 = PWM channel 3 trigger ADC function Disabled. 1 = PWM channel 3 trigger ADC function Enabled. PWM can trigger ADC to start conversion when PWM counter up count to CNR if this bit is set to 1. **Note:** This function is only supported when PWM operating at Center-aligned type.
[2]	**PWM2TEN**	**Channel 2 Center-Aligned Trigger Enable Bit** 0 = PWM channel 2 trigger ADC function Disabled. 1 = PWM channel 2 trigger ADC function Enabled. PWM can trigger ADC to start conversion when PWM counter up count to CNR if this bit is set to 1. **Note:** This function is only supported when PWM operating at Center-aligned type.
[1]	**PWM1TEN**	**Channel 1 Center-Aligned Trigger Enable Bit** 0 = PWM channel 1 trigger ADC function Disabled. 1 = PWM channel 1 trigger ADC function Enabled. PWM can trigger ADC to start conversion when PWM counter up count to CNR if this bit is set to 1. **Note:** This function is only supported when PWM operating at Center-aligned type.
[0]	**PWM0TEN**	**Channel 0 Center-Aligned Trigger Enable Bit** 0 = PWM channel 0 trigger ADC function Disabled. 1 = PWM channel 0 trigger ADC function Enabled. PWM can trigger ADC to start conversion when PWM counter up count to CNR if this bit is set to 1. **Note:** This function is only supported when PWM operating at Center-aligned type.

PWM Trigger Status Register (TSTATUS)

Register	Offset	R/W	Description	Reset Value
TSTATUS	PWMA_BA+0x84 PWMB_BA+0x84	R/W	PWM Trigger 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				PWM3TF	PWM2TF	PWM1TF	PWM0TF

Bits	Description	
[3]	PWM3TF	**Channel 3 Center-Aligned Trigger Flag** For Center-aligned Operating mode, this bit is set to 1 by hardware when PWM counter up count to CNR if PWM3TEN bit is set to 1. After this bit is set to 1, ADC will start conversion if ADC triggered source is selected by PWM. Software can write 1 to clear this bit.
[2]	PWM2TF	**Channel 2 Center-Aligned Trigger Flag** For Center-aligned Operating mode, this bit is set to 1 by hardware when PWM counter up count to CNR if PWM2TEN bit is set to 1. After this bit is set to 1, ADC will start conversion if ADC triggered source is selected by PWM. Software can write 1 to clear this bit.
[1]	PWM1TF	**Channel 1 Center-Aligned Trigger Flag** For Center-aligned Operating mode, this bit is set to 1 by hardware when PWM counter up count to CNR if PWM1TEN bit is set to 1. After this bit is set to 1, ADC will start conversion if ADC triggered source is selected by PWM. Software can write 1 to clear this bit.
[0]	PWM0TF	**Channel 0 Center-Aligned Trigger Flag** For Center-aligned Operating mode, this bit is set to 1 by hardware when PWM counter up counts to CNR if PWM0TEN bit is set to 1. After this bit is set to 1, ADC will start conversion if ADC triggered source is selected by PWM. Software can write 1 to clear this bit.

PWM0 Synchronous Busy Status Register (SYNCBUSY0)

Register	Offset	R/W	Description	Reset Value
SYNCBUSY0	PWMA_BA+0x88 PWMB_BA+0x88	R	PWM0 Synchronous Busy 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							S_BUSY

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**S_BUSY**	**PWM Synchronous Busy** When software writes CNR0/CMR0/PPR or switches PWM0 operation mode (PCR[3]), PWM will have a busy time to update these values completely because PWM clock may be different from system clock domain. Software needs to check this busy status before writing CNR0/CMR0/PPR or switching PWM0 operation mode (PCR[3]) to make sure previous setting has been updated completely. This bit will be set when software writes CNR0/CMR0/PPR or switches PWM0 operation mode (PCR[3]) and will be cleared by hardware automatically when PWM update these value completely.

PWM1 Synchronous Busy Status Register (SYNCBUSY1)

Register	Offset	R/W	Description	Reset Value
SYNCBUSY1	PWMA_BA+0x8C PWMB_BA+0x8C	R	PWM1 Synchronous Busy 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							S_BUSY

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**S_BUSY**	**PWM Synchronous Busy** When Software writes CNR1/CMR1/PPR or switches PWM1 operation mode (PCR[11]), PWM will have a busy time to update these values completely because PWM clock may be different from system clock domain. Software needs to check this busy status before writing CNR1/CMR1/PPR or switching PWM1 operation mode (PCR[11]) to make sure previous setting has been updated completely. This bit will be set when software writes CNR1/CMR1/PPR or switches PWM1 operation mode (PCR[11]) and will be cleared by hardware automatically when PWM update these value completely.

PWM2 Synchronous Busy Status Register (SYNCBUSY2)

Register	Offset	R/W	Description	Reset Value
SYNCBUSY2	PWMA_BA+0x90	R	PWM2 Synchronous Busy 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							**S_BUSY**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**S_BUSY**	**PWM Synchronous Busy** When Software writes CNR2/CMR2/PPR or switch PWM2 operation mode (PCR[19]), PWM will have a busy time to update these values completely because PWM clock may be different from system clock domain. Software needs to check this busy status before writing CNR2/CMR2/PPR or switching PWM2 operation mode (PCR[19]) to make sure previous setting has been updated completely. This bit will be set when software writes CNR2/CMR2/PPR or switch PWM2 operation mode (PCR[19]) and will be cleared by hardware automatically when PWM update these value completely.

PWM3 Synchronous Busy Status Register (SYNCBUSY3)

Register	Offset	R/W	Description	Reset Value
SYNCBUSY3	PWMA_BA+0x94	R	PWM3 Synchronous Busy 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							**S_BUSY**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**S_BUSY**	**PWM Synchronous Busy** When Software writes CNR3/CMR3/PPR or switch PWM3 operation mode (PCR[27]), PWM will have a busy time to update these values completely because PWM clock may be different from system clock domain. Software need to check this busy status before writing CNR3/CMR3/PPR or switching PWM3 operation mode (PCR[27]) to make sure previous setting has been updated completely. This bit will be set when Software writes CNR3/CMR3/PPR or switch PWM3 operation mode (PCR[27]) and will be cleared by hardware automatically when PWM update these value completely.

6.10 Watchdog Timer (WDT)

6.10.1 Overview

The purpose of Watchdog Timer is 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.

6.10.2 Features

- 18-bit free running up counter for Watchdog Timer time-out interval.
- Selectable time-out interval (2^4 ~ 2^{18}) WDT_CLK cycle and the time-out interval period is 104 ms ~ 26.3168 s if WDT_CLK = 10 kHz.
- System kept in reset state for a period of (1 / WDT_CLK) * 63
- Supports Watchdog Timer reset delay period
  - Selectable it includes (1026、130、18 or 3) * WDT_CLK reset delay period.
- Supports to force Watchdog Timer enabled after chip powered on or reset while CWDTEN (CONFIG0[31] Watchdog Enable) bit is set to 0.
- Supports Watchdog Timer time-out wake-up function only if WDT clock source is selected as 10 kHz

6.10.3 Block Diagram

The Watchdog Timer clock control and block diagram are shown as follows.

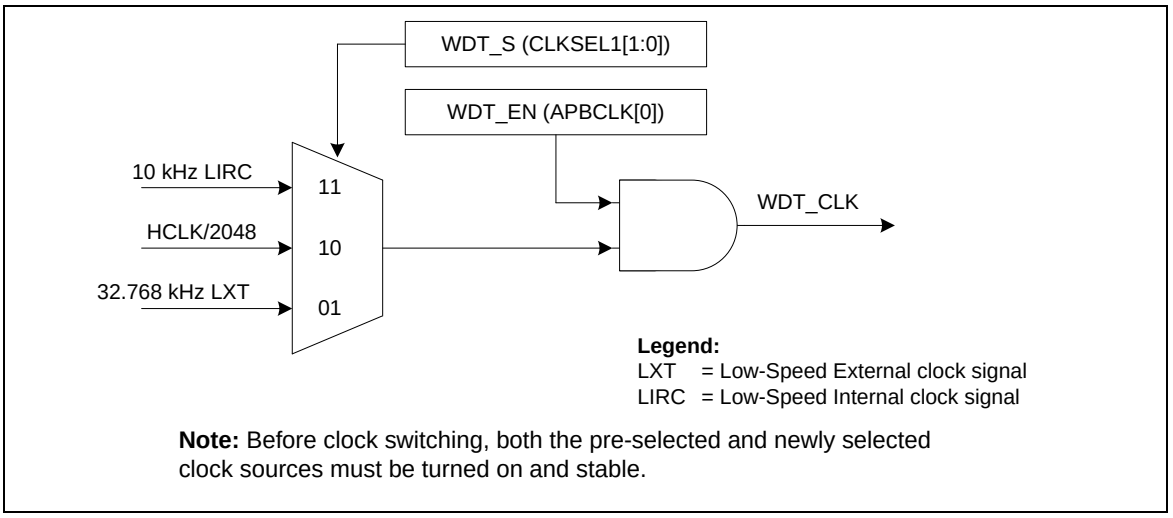


Figure 6.10-1 Watchdog Timer Clock Control

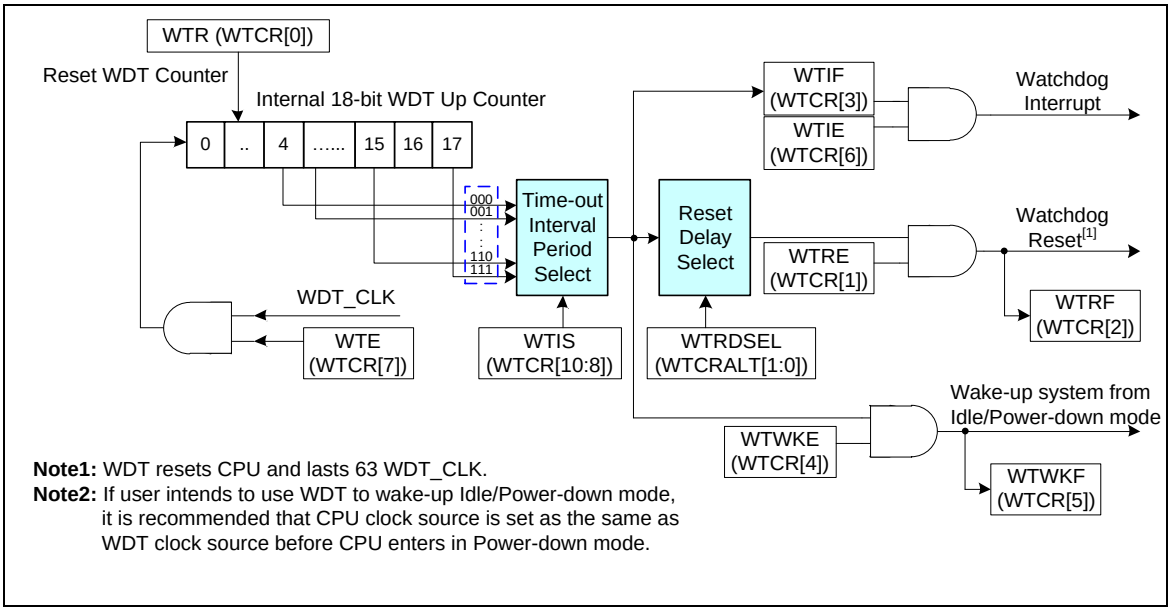


Figure 6.10-2 Watchdog Timer Block Diagram

6.10.4 Basic Configuration

The WDT peripheral clock is enabled in APBCLK[0] and clock source can be selected in CLKSEL1[1:0].
Or user can setting CONFIG0[31] is 0 to force Watchdog Timer enabled and active in 10 kHz after chip powered on or reset.

6.10.5 Functional Description

The Watchdog Timer (WDT) includes an 18-bit free running up counter with programmable time-out intervals. Table 5-29 shows the WDT time-out interval period selection and Figure 6.10-3 Watchdog Timer Time-out Interval and Reset Period Timing shows the WDT time-out interval and reset period timing.

- WDT Time-out Interrupt

Setting WTE bit to 1 will enable the WDT function and the WDT counter to start counting up. There are eight time-out interval period can be selected by setting WTIS. When the WDT up counter reaches the WTIS settings, WDT time-out interrupt will occur then WTIF flag will be set to 1 immediately.

- WDT Reset Delay Period and Reset System

There is a specified T_{RSTD} delay period follows the WTIF flag is setting to 1. User should set WTR bit to reset the 18-bit WDT up counter value to avoid generate WDT time-out reset signal before the T_{RSTD} delay period expires. If the WDT up counter value has not been cleared after the specific T_{RSTD} delay period expires, the WDT control will set WTRF flag to 1 if WTRE bit is enabled, then chip enters to reset state immediately. Refer to Figure 6.10-3 Watchdog Timer Time-out Interval and Reset Period Timing, the T_{RST} reset period will keep last 63 WDT clocks then chip restart executing program from reset vector (0x0000_0000). The WTRF flag will keep 1 after WDT time-out reset the chip, user can check WTRF flag by software to recognize the system has been reset by WDT time-out reset or not.

- WDT Wake-up

If WDT clock source is selected to 10 kHz, system can be waken-up from Power-down mode while WDT time-out interrupt signal is generated and WTWKE bit enabled. In the meanwhile, the WTWKF flag will set to 1 automatically, user can check WTWKF flag by software to recognize the system has been waken-up by WDT time-out interrupt or not.

WTIS	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}

Table 6.10-1 Watchdog Timer Time-out Interval Period Selection

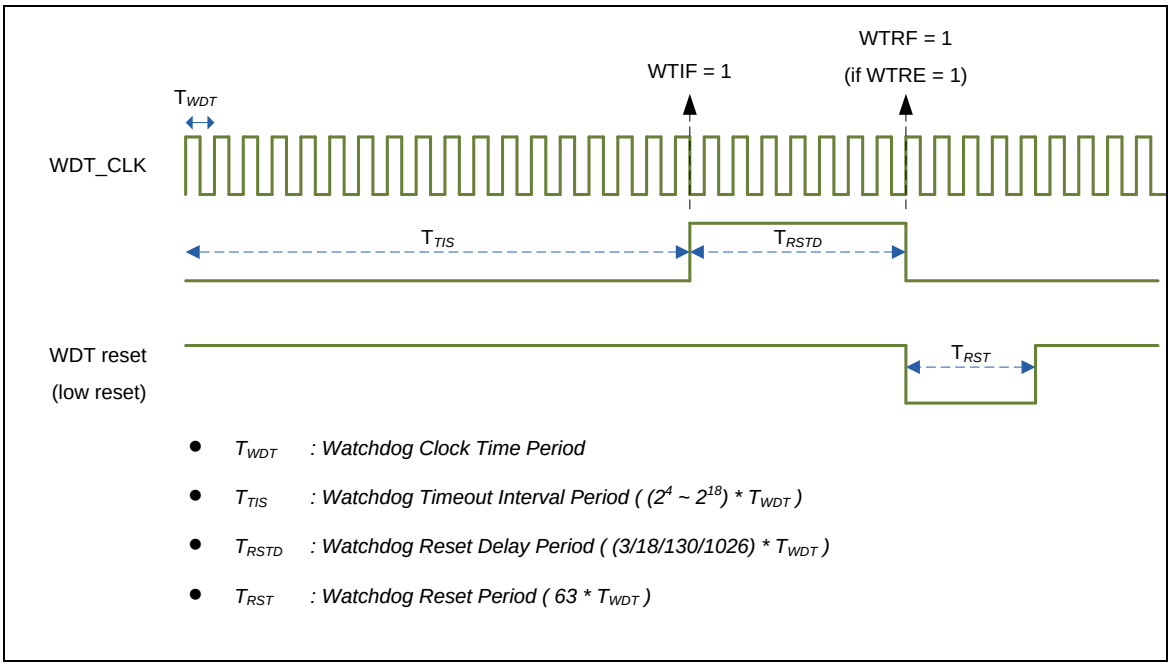


Figure 6.10-3 Watchdog Timer Time-out Interval and Reset Period Timing

6.10.6 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 = 0x4000_4000**				
WTCR	WDT_BA+0x00	R/W	Watchdog Timer Control Register	0x0000_0700
WTCRALT	WDT_BA+0x04	R/W	Watchdog Timer Alternative Control Register	0x0000_0000

6.10.7 Register Description

Watchdog Timer Control Register (WTCR)

Register	Offset	R/W	Description	Reset Value
WTCR	WDT_BA+0x00	R/W	Watchdog Timer Control Register	0x0000_0700

31	30	29	28	27	26	25	24
DBGACK_WDT	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					WTIS		
7	6	5	4	3	2	1	0
WTE	WTIE	WTWKF	WTWKE	WTIF	WTRF	WTRE	WTR

Bits	Description	
[31]	**DBGACK_WDT**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects 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.**
[30:11]	**Reserved**	Reserved.
[10:8]	**WTIS**	**Watchdog Timer Time-Out Interval Selection (Write Protect)** These three bits select the time-out interval period for the WDT. 000 = 2^4 *T_{WDT}. 001 = 2^6 * T_{WDT}. 010 = 2^8 * T_{WDT}. 011 = 2^{10} * T_{WDT}. 100 = 2^{12} * T_{WDT}. 101 = 2^{14} * T_{WDT}. 110 = 2^{16} * T_{WDT}. 111 = 2^{18} * T_{WDT}.
[7]	**WTE**	**Watchdog Timer Enable Bit (Write Protect)** 0 = WDT Disabled. (This action will reset the internal up counter value.) 1 = WDT Enabled. **Note:** If CWDTEN (CONFIG0[31] Watchdog Enable) bit is set to 0, this bit is forced as 1 and user cannot change this bit to 0.
[6]	**WTIE**	**Watchdog Timer 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.
[5]	**WTWKF**	**Watchdog Timer Time-Out Wake-Up Flag** This bit indicates the interrupt wake-up flag status of WDT. 0 = WDT does not cause chip wake-up. 1 = Chip wake-up from Idle or Power-down mode if WDT time-out interrupt signal generated. **Note:** This bit is cleared by writing 1 to it.
[4]	**WTWKE**	**Watchdog Timer Time-Out Wake-Up Function Control (Write Protect)** If this bit is set to 1, while WTIF is generated to 1 and WTIE 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:** Chip can be woken-up by WDT time-out interrupt signal generated only if WDT clock source is selected to 10 kHz oscillator.
[3]	**WTIF**	**Watchdog Timer 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]	**WTRF**	**Watchdog Timer 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]	**WTRE**	**Watchdog Timer 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.
[0]	**WTR**	**Reset Watchdog Timer Up Counter (Write Protect)** 0 = No effect. 1 = Reset the internal 18-bit WDT up counter value. **Note:** This bit will be automatically cleared by hardware.

Watchdog Timer Alternative Control Register (WTCRALT)

Register	Offset	R/W	Description	Reset Value
WTCRALT	WDT_BA+0x04	R/W	Watchdog Timer 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						**WTRDSEL**	

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1:0]	**WTRDSEL**	**Watchdog Timer Reset Delay Selection (Write Protect)** When WDT time-out happened, user has a time named WDT Reset Delay Period to clear WDT counter to prevent WDT time-out reset happened. User can select a suitable value of WDT Reset Delay Period for different WDT time-out period. These bits are protected bit. It means programming this bit needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Reference the register REGWRPROT at address GCR_BA+0x100. 00 = Watchdog Timer Reset Delay Period is 1026 * WDT_CLK. 01 = Watchdog Timer Reset Delay Period is 130 * WDT_CLK. 10 = Watchdog Timer Reset Delay Period is 18 * WDT_CLK. 11 = Watchdog Timer Reset Delay Period is 3 * WDT_CLK. **Note:** This register will be reset to 0 if WDT time-out reset happened.

6.11 Window Watchdog Timer (WWDT)

6.11.1 Overview

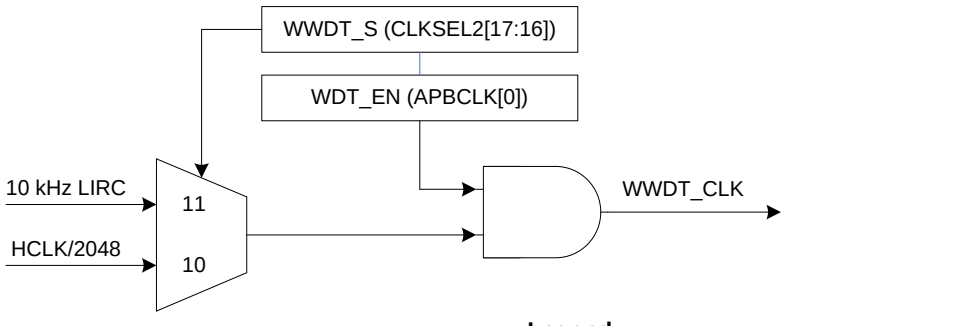
The Window Watchdog Timer is used is to perform a system reset within a specified window period to prevent software run to uncontrollable status by any unpredictable condition.

6.11.2 Features

- 6-bit down counter value (WWDTVAL[5:0]) and 6-bit compare window value (WWDTCR[21:16]) to make the WWDT time-out window period flexible
- Supports 4-bit value to programmable maximum 11-bit prescale counter period of WWDT counter

6.11.3 Block Diagram

The Window Watchdog Timer clock control and block diagram are shown as follows.



Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 6.11-1 Window Watchdog Timer Clock Control

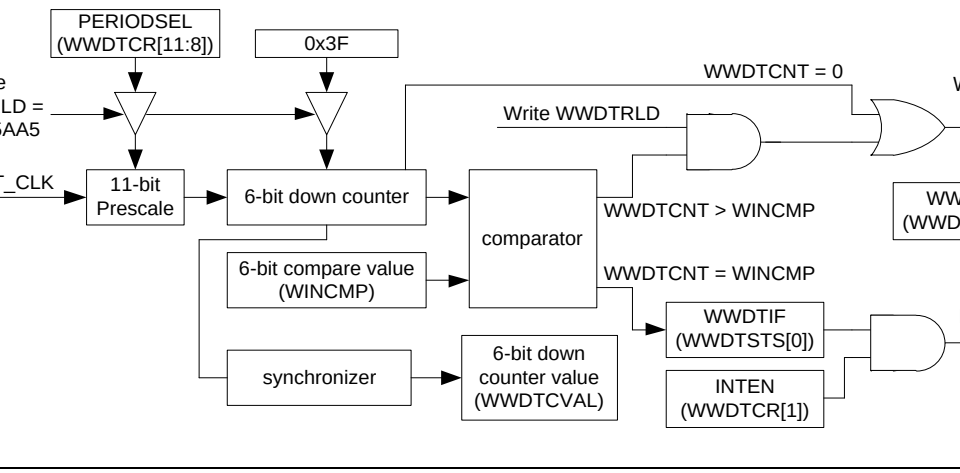


Figure 6.11-2 Window Watchdog Timer Block Diagram

6.11.4 Basic Configuration

The WWDT peripheral clock is enabled in APBCLK[0] and clock source can be selected in CLKSEL2[17:16].

6.11.5 Functional Description

The Window Watchdog Timer (WWDT) includes a 6-bit down counter with programmable prescale value to define different WWDT time-out intervals.
The clock source of 6-bit WWDT is based on system clock divide 2048 (HCLK/2048) or internal 10 kHz oscillator with a programmable 11-bit prescale counter value which controlled by PERIODSEL (WWDTCRL[11:8]) setting. Also, the correlate of PERIODSEL and prescale value are listed in the Table 6.11-1.

PERIODSEL	Prescaler Value	Max. Time-Out Period	Max. Time-Out Interval (WWDT_CLK=10 KHz)
0000	1	1 * 64 * T_{WWDT}	6.4 ms
0001	2	2 * 64 * T_{WWDT}	12.8 ms
0010	4	4 * 64 * T_{WWDT}	25.6 ms
0011	8	8 * 64 * T_{WWDT}	51.2 ms
0100	16	16 * 64 * T_{WWDT}	102.4 ms
0101	32	32 * 64 * T_{WWDT}	204.8 ms
0110	64	64 * 64 * T_{WWDT}	409.6 ms
0111	128	128 * 64 * T_{WWDT}	819.2 ms
1000	192	192 * 64 * T_{WWDT}	1.2288 s
1001	256	256 * 64 * T_{WWDT}	1.6384 s
1010	384	384 * 64 * T_{WWDT}	2.4576 s
1011	512	512 * 64 * T_{WWDT}	3.2768 s
1100	768	768 * 64 * T_{WWDT}	4.9152 s
1101	1024	1024 * 64 * T_{WWDT}	6.5536 s
1110	1536	1536 * 64 * T_{WWDT}	9.8304 s
1111	2048	2048 * 64 * T_{WWDT}	13.1072 s

Table 6.11-1 Window Watchdog Timer Prescale Value Selection

- WWDT Counting

When the WWDTEN bit is set, WWDT down counter will start counting from 0x3F to 0. To prevent program runs to disable WWDT counter counting unexpected, the WWDT control register WWDTCR can only be written once after chip is powered on or reset. User cannot disable WWDT counter counting (WWDTEN), change counter prescale period (PERIODSEL) or change window compare value (WINCMP) while WWDTEN (WWDTCR[0]) bit has been enabled by software unless chip is reset.

- WWDT Compare Match Interrupt

During down counting by the WWDT counter, the WWDTIF is set to 1 while the WWDT counter value (WWDTCVAL) is equal to WINCMP value and WWDTIF can be cleared by software; if WWDTIE is also set to 1 by software, the WWDT compare match interrupt signal is generated also while WWDTIF is set to 1 by hardware.

- WWDT Reset System

When WWDTIF is generated, user must reload WWDT internal counter value to 0x3F by writing 0x00005AA5 to WWDTRLD, and also to prevent WWDT counter value reached to 0 and generate WWDT reset system signal to info system reset.
If current WWDTCVAL value is larger than WINCMP value and user writes 0x00005AA5 to the WWDTRLD register, the WWDT reset system signal will be generated immediately to cause chip reset also.

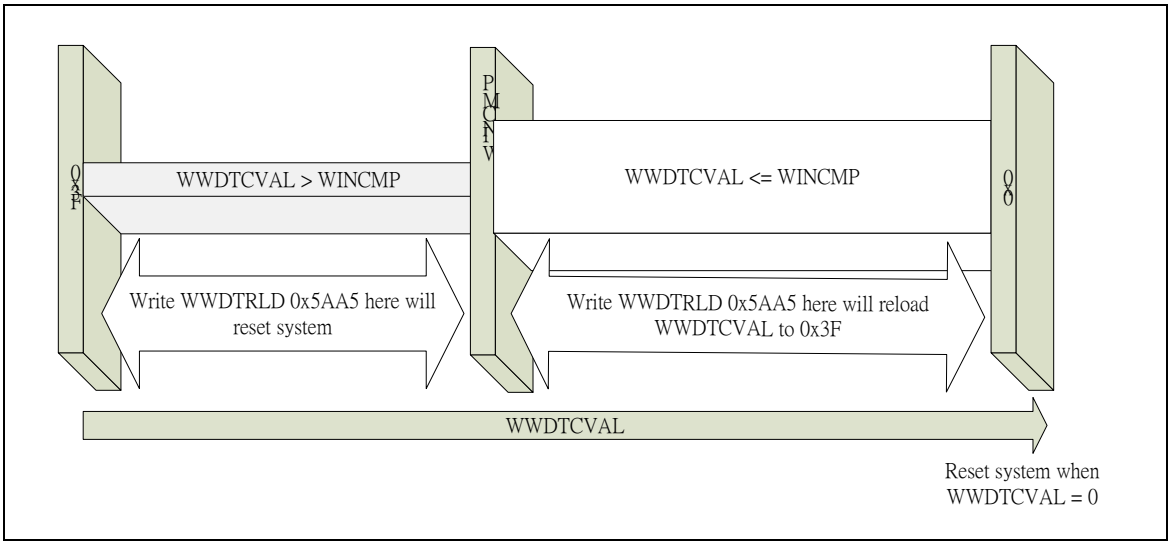


Figure 6.11-3 Window Watchdog Timer Reset and Reload Behavior

- WWDT Window Setting Limitation

  When user writes 0x00005AA5 to WWDTRLD register to reload WWDT counter value to 0x3F, it needs 3 WWDT clocks to sync the reload command to actually perform reload action.
  This means if user set PERIODSEL to 0000, the counter prescale value should be as 1, and the WINCMP value must be larger than 2; otherwise, writing WWDTRLD to reload WWDT counter value to 0x3F is unavailable while WWDTIF is generated and WWDT reset system event always happened.

PERIODSEL	Prescale Value	Valid WINCMP Value
0000	1	0x3 ~ 0x3F
0001	2	0x2 ~ 0x3F
Others	Others	0x0 ~ 0x3F

Table 6.11-2 WINCMP Setting Limitation

6.11.6 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 = 0x4000_4100**				
WWDTRLD	WWDT_BA+0x00	W	Window Watchdog Timer Reload Counter Register	0x0000_0000
WWDTCR	WWDT_BA+0x04	R/W	Window Watchdog Timer Control Register	0x003F_0800
WWDTSR	WWDT_BA+0x08	R/W	Window Watchdog Timer Status Register	0x0000_0000
WWDTCVR	WWDT_BA+0x0C	R	Window Watchdog Timer Counter Value Register	0x0000_003F

6.11.7 Register Description

Window Watchdog Timer Reload Counter Register (WWDTRLD)

Register	Offset	R/W	Description	Reset Value
WWDTRLD	WWDT_BA+0x00	W	Window Watchdog Timer Reload Counter Register	0x0000_0000

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

Bits	Description	
[31:0]	**WWDTRLD**	**WWDT Reload Counter Register** Writing 0x00005AA5 to this register will reload the WWDT counter value to 0x3F. **Note:** User can only write WWDTRLD to reload WWDT counter value when current WWDT counter value between 0 and WINCMP. If user writes WWDTRLD when current WWDT counter value is larger than WINCMP, WWDT reset signal will generate immediately.

Window Watchdog Timer Control Register (WWDTCR)

Register	Offset	R/W	Description	Reset Value
WWDTCR	WWDT_BA+0x04	R/W	Window Watchdog Timer 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
DBGACK_ WWDT	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**WINCMP**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**PERIODSEL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**WWDTIE**	**WWDTEN**

Bits	Description	
[31]	**DBGACK_WWDT**	**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. WWDT down counter will keep going no matter CPU is held by ICE or not.
[30:22]	**Reserved**	Reserved.
[21:16]	**WINCMP**	**WWDT Window Compare Register** Set this register to adjust the valid reload window. **Note:** User can only write WWDTRLD to reload WWDT counter value when current WWDT counter value between 0 and WINCMP. If user writes WWDTRLD when current WWDT counter value larger than WINCMP, WWDT reset signal will generate immediately.
[15:12]	**Reserved**	Reserved.
[11:8]	**PERIODSEL**	**WWDT Counter Prescale Period Selection** 0000 = Pre-scale is 1; Max time-out period is 1 * 64 * TWWDT. 0001 = Pre-scale is 2; Max time-out period is 2 * 64 * TWWDT. 0010 = Pre-scale is 4; Max time-out period is 4 * 64 * TWWDT. 0011 = Pre-scale is 8; Max time-out period is 8 * 64 * TWWDT. 0100 = Pre-scale is 16; Max time-out period is 16 * 64 * TWWDT. 0101 = Pre-scale is 32; Max time-out period is 32 * 64 * TWWDT. 0110 = Pre-scale is 64; Max time-out period is 64 * 64 * TWWDT. 0111 = Pre-scale is 128; Max time-out period is 128 * 64 * TWWDT. 1000 = Pre-scale is 192; Max time-out period is 192 * 64 * TWWDT. 1001 = Pre-scale is 256; Max time-out period is 256 * 64 * TWWDT. 1010 = Pre-scale is 384; Max time-out period is 384 * 64 * TWWDT. 1011 = Pre-scale is 512; Max time-out period is 512 * 64 * TWWDT. 1100 = Pre-scale is 768; Max time-out period is 768 * 64 * TWWDT.

		1101 = Pre-scale is 1024; Max time-out period is 1024 * 64 * TWWDT. 1110 = Pre-scale is 1536; Max time-out period is 1536 * 64 * TWWDT. 1111 = Pre-scale is 2048; Max time-out period is 2048 * 64 * TWWDT.
[7:2]	**Reserved**	Reserved.
[1]	**WWDTIE**	**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** Set this bit to enable WWDT counter counting 0 = WWDT counter is stopped. 1 = WWDT counter is starting counting.

Window Watchdog Timer Status Register (WWDTSR)

Register	Offset	R/W	Description	Reset Value
WWDTSR	WWDT_BA+0x08	R/W	Window Watchdog Timer 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 Time-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 WINCMP value. 0 = No effect. 1 = WWDT counter value matches WINCMP value. **Note:** This bit is cleared by writing 1 to it.

Window Watchdog Timer Counter Value Register (WWDTCVR)

Register	Offset	R/W	Description	Reset Value
WWDTCVR	WWDT_BA+0x0C	R	Window Watchdog Timer 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		**WWDTCVAL**					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**WWDTCVAL**	**WWDT Counter Value** WWDTCVAL will be updated continuously to monitor 6-bit down counter value.

6.12 Real Time Clock (RTC)

6.12.1 Overview

The Real Time Clock (RTC) controller provides the real time and calendar message. The RTC offers programmable time tick and alarm match interrupts. The data format of time and calendar messages are expressed in BCD format. A digital frequency compensation feature is available to compensate external crystal oscillator frequency accuracy.

The RTC controller also offers 80 bytes spare registers to store user's important information.

6.12.2 Features

- Supports real time counter in Time Loading Register (TLR) (hour, minute, second) and calendar counter in Calendar Loading Register (CLR) (year, month, day) for RTC time and calendar check
- Supports alarm time (hour, minute, second) and calendar (year, month, day) settings in Time Alarm Register (TAR) and Calendar Alarm Register (CAR) register
- Selectable 12-hour or 24-hour time scale in Time Scale Selection Register (TSSR) register
- Supports Leap Year indication in Leap Year Indicator Register (LIR) register
- Supports Day of the Week counter in Day of the Week Register (DWR) register
- Frequency of RTC clock source compensate by RTC Frequency Compensation Register (FCR) register
- All time and calendar message expressed in BCD format
- Supports periodic RTC Time Tick interrupt with 8 period interval options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second
- Supports RTC Time Tick and Alarm Match interrupt
- Supports chip wake-up from Idle or Power-down mode while a RTC interrupt signal is generated
- Supports 80 bytes spare registers

6.12.3 Block Diagram

The block diagram of Real Time Clock is depicted as follows:

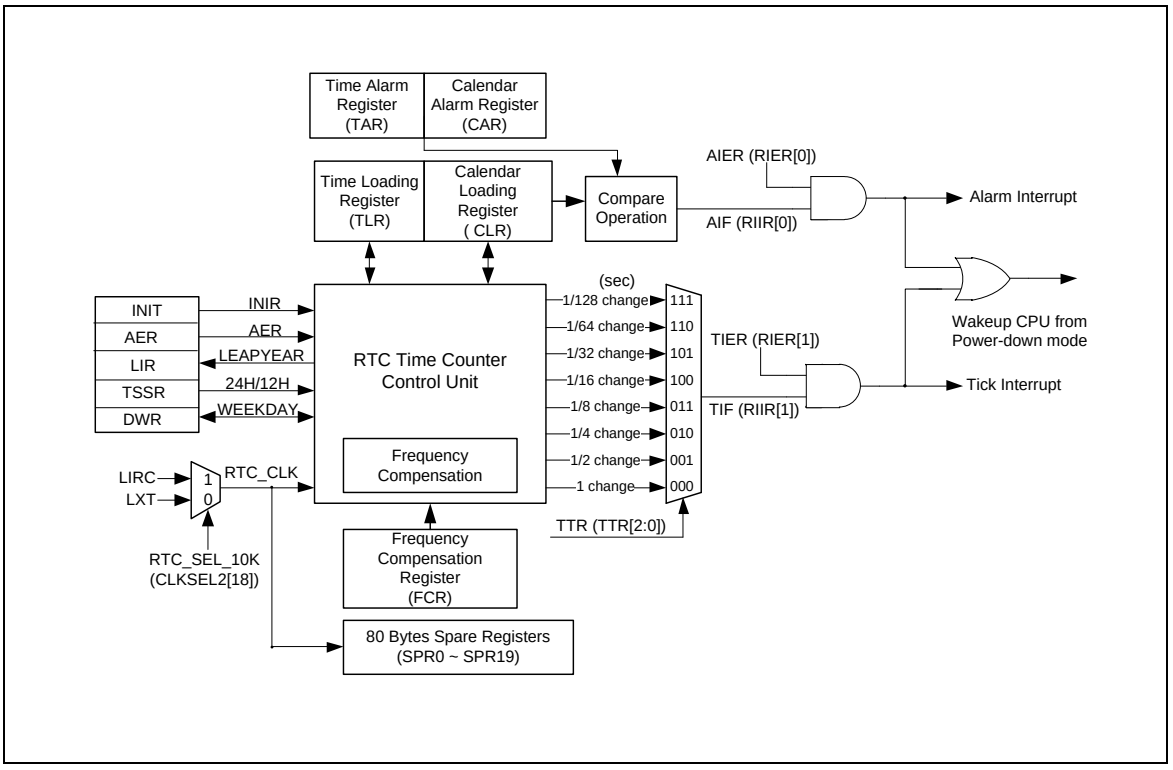


Figure 6.12-1 RTC Block Diagram

6.12.4 Basic Configuration

RTC controller clock enable is in RTC_EN (APBCLK[1]) and low speed 32 kHz oscillator is enabled by XTL32K_EN (PWRCON[1]).

6.12.5 Functional Description

6.12.5.1 RTC Initiation

When a RTC block is powered on, RTC is at reset state. User has to write a number 0xa5eb1357 to INIR (INIR [31:0] RTC Initiation) register to make RTC leaving reset state. Once the INIR is written as 0xa5eb1357, the RTC will be in normal active state permanently. User can read Active bit (INIR[0] RTC Active Status) to check the RTC is at normal active state or reset state.

6.12.5.2 Access to RTC register

Due to clock frequency difference between RTC clock and system clock, when user write new data to any one of the RTC registers, the data will not be updated until 2 RTC clocks later (about 60us).

In addition, user must be aware that RTC controller does not check whether loaded data is out of bounds or not in TLR, CLR, TAR and CAR registers. RTC does not check rationality between DWR and CLR either.

6.12.5.3 RTC Read/Write Enable

AER (AER[15:0] RTC Register Access Enable Password) is served as read/write access of RTC registers to unlock register read/write protection function. If AER[15:0] is written to 0xA965, user can read ENF (AER[16] RTC Register Access Enable Flag) bit status to check the RTC registers are read/write accessible or locked. Once ENF bit enabled, RTC access enable function will keep effect at least 1024 RTC clocks (about 30ms) and ENF bit will be cleared automatically after 1024 RTC clocks.

The RTC control registers access attribute when ENF is 1 and 0 are shown in below table.

Register	ENF=1	ENF=0
INIR	R/W	R/W
AER	R/W	R/W
FCR	R/W	Not available
TLR	R/W	R
CLR	R/W	R
TSSR	R/W	R/W
DWR	R/W	R
TAR	R/W	Not available
CAR	R/W	Not available
LIR	R	R
RIER	R/W	R/W
RIIR	R/W	R/W
TTR	R/W	Not available
SPRCTL	R/W	Not available
SPR0-SPR19	R/W	Not available

6.12.5.4 Frequency Compensation

The RTC source clock may not precise to exactly 32768 Hz and the FCR register (Frequency Compensation Register) allows user to make digital compensation to the RTC source clock only if the frequency of RTC source clock is in the range from 32761 Hz to 32776 Hz.

Integer Part Of Detected Value	INTEGER (FCR[11:8])	Integer Part Of Detected Value	INTEGER (FCR[11:8])
32776	1111	32768	0111
32775	1110	32767	0110
32774	1101	32766	0101
32773	1100	32765	0100
32772	1011	32764	0011
32771	1010	32763	0010
32770	1001	32762	0001
32769	1000	32761	0000

Following are the compensation examples for the real RTC source clock is higher or lower than 32768 Hz.

Example 1: (RTC Source Clock > 32768 Hz)
RTC Source Clock Measured: 32773.65 Hz (> 32768 Hz)
Integer Part: 32773 => 0x8005
INTEGER (FCR [11:8] Integer Part) = 0x05 – 0x01 + 0x08 = 0x0c
Fraction Part: 0.65
FRACTION (FCR [5:0] Fraction Part) = 0.65 X 60 = 39 = 0x27
RTC FCR Register Should Be As 0xC27.
Example 2: (RTC source clock ≦ 32768 Hz)
RTC source clock measured: 32765.27 Hz (≦ 32768 Hz)
Integer part: 32765 => 0x7FFD
INTEGER (FCR [11:8] Integer Part) = 0x0D – 0x01 – 0x08 = 0x04
Fraction part: 0.27
FRACTION (FCR [5:0] Fraction Part) = 0.27 x 60 = 16.2 = 0x10
RTC FCR register should be as 0x410.

6.12.5.5 Time and Calendar counter

TLR and CLR register are used to load the real time and calendar. TAR and CAR register are used for setup alarm time and calendar.

6.12.5.6 12/24 hour Time Scale Selection

The 12/24 hour time scale selection depends on 24H_12H bit (TSSR[0] 24-Hour / 12-Hour Time Scale Selection).

24-Hour Time Scale (24H_12H = 1)	12-Hour Time Scale (24H_12H = 0)	24-Hour Time Scale (24H_12H = 1)	12-Hour Time Scale (PM Time + 20) (24H_12H = 0)
00	12 (AM12)	12	32 (PM12)
01	01 (AM01)	13	21 (PM01)
02	02 (AM02)	14	22 (PM02)
03	03 (AM03)	15	23 (PM03)
04	04 (AM04)	16	24 (PM04)
05	05 (AM05)	17	25 (PM05)
06	06 (AM06)	18	26 (PM06)
07	07 (AM07)	19	27 (PM07)
08	08 (AM08)	20	28 (PM08)
09	09 (AM09)	21	29 (PM09)
10	10 (AM10)	22	30 (PM10)

11	11 (AM11)	23	31 (PM11)

6.12.5.7 Day of the Week counter

The RTC controller provides day of week in DWR (DWR[2:0] Day of the Week Register). The value is defined from 0 to 6 to represent Sunday to Saturday respectively.

6.12.5.8 Periodic Time Tick Interrupt

The Periodic Time Tick interrupt has 8 period interval options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second that are selected by TTR (TTR[2:0] Time Tick Register). When Periodic Time Tick interrupt is enabled by setting TIER (RIER[1] Time Tick Interrupt Enable) to 1, the Periodic Time Tick interrupt is requested periodically in the period selected by TTR[2:0] settings.

6.12.5.9 Alarm Interrupt

When the real time and calendar message in TLR and CLR registers are equal to alarm time and calendar values in RTC TAR and CAR registers, the AIF (RIIR[0] RTC Alarm Interrupt Flag) is set to 1 and the RTC alarm interrupt signal asserted if the AIER (RIER[0] Alarm Interrupt Enable) is enabled.

6.12.5.10 Application Note

1. All data in TAR, CAR, TLR and CLR registers are all expressed in BCD format.
2. User has to make sure that the loaded values are reasonable. For example, load CLR as 201a (year), 13 (month), 00 (day), or CLR does not match with DWR, etc.
3. Registers value after powered on or reset:

Register	Reset State
AER	0
CLR	05/1/1 (year/month/day)
TLR	00:00:00 (hour : minute : second)
CAR	00/00/00 (year/month/day)
TAR	00:00:00 (hour : minute : second)
TSSR	1 (24-hour mode)
DWR	6 (Saturday)
RIER	0
RIIR	0
LIR	0
TTR	0

4. In CLR and CAR, only 2 BCD digits are used to express “year”. The 2 BCD digits of xy means 20xy, rather than 19xy or 21xy.

6.12.5.11 Spare registers

The RTC module is equipped 80 bytes spare registers to store user’s important information. These spare registers are located in RTC clock domain, user needs to enable SPREN (SPRCTL[2] SPR Register Enable) before writing one of 20 spare registers (SPR0 ~ SPR19). User could read SPRRDY (SPRCTL[7] SPR Register Ready) to check if data has been written into registers or not. User could only access the spare registers again once SPRRDY is 1. Any access to spare registers is available if SPRRDY is 0.

If external 32 kHz clock (LXT) is not available in system design, user can choose RTC_SEL_10K (CLKSEL2[18]) to 1 to use on chip 10 kHz clock (LIRC) instead of for spare registers read and write operations.

6.12.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
RTC Base Address: **RTC_BA = 0x4000_8000**				
INIR	RTC_BA+0x00	R/W	RTC Initiation Register	0x0000_0000
AER	RTC_BA+0x04	R/W	RTC Access Enable Register	0x0000_0000
FCR	RTC_BA+0x08	R/W	RTC Frequency Compensation Register	0x0000_0700
TLR	RTC_BA+0x0C	R/W	Time Loading Register	0x0000_0000
CLR	RTC_BA+0x10	R/W	Calendar Loading Register	0x0005_0101
TSSR	RTC_BA+0x14	R/W	Time Scale Selection Register	0x0000_0001
DWR	RTC_BA+0x18	R/W	Day of the Week Register	0x0000_0006
TAR	RTC_BA+0x1C	R/W	Time Alarm Register	0x0000_0000
CAR	RTC_BA+0x20	R/W	Calendar Alarm Register	0x0000_0000
LIR	RTC_BA+0x24	R	Leap Year Indicator Register	0x0000_0000
RIER	RTC_BA+0x28	R/W	RTC Interrupt Enable Register	0x0000_0000
RIIR	RTC_BA+0x2C	R/W	RTC Interrupt Indicator Register	0x0000_0000
TTR	RTC_BA+0x30	R/W	RTC Time Tick Register	0x0000_0000
SPRCTL	RTC_BA+0x3C	R/W	RTC Spare Functional Control Register	0x0000_0080
SPR0	RTC_BA+0x40	R/W	RTC Spare Register 0	0x0000_0000
SPR1	RTC_BA+0x44	R/W	RTC Spare Register 1	0x0000_0000
SPR2	RTC_BA+0x48	R/W	RTC Spare Register 2	0x0000_0000
SPR3	RTC_BA+0x4C	R/W	RTC Spare Register 3	0x0000_0000
SPR4	RTC_BA+0x50	R/W	RTC Spare Register 4	0x0000_0000
SPR5	RTC_BA+0x54	R/W	RTC Spare Register 5	0x0000_0000
SPR6	RTC_BA+0x58	R/W	RTC Spare Register 6	0x0000_0000
SPR7	RTC_BA+0x5C	R/W	RTC Spare Register 7	0x0000_0000
SPR8	RTC_BA+0x60	R/W	RTC Spare Register 8	0x0000_0000
SPR9	RTC_BA+0x64	R/W	RTC Spare Register 9	0x0000_0000

SPR10	RTC_BA+0x68	R/W	RTC Spare Register 10	0x0000_0000
SPR11	RTC_BA+0x6C	R/W	RTC Spare Register 11	0x0000_0000
SPR12	RTC_BA+0x70	R/W	RTC Spare Register 12	0x0000_0000
SPR13	RTC_BA+0x74	R/W	RTC Spare Register 13	0x0000_0000
SPR14	RTC_BA+0x78	R/W	RTC Spare Register 14	0x0000_0000
SPR15	RTC_BA+0x7C	R/W	RTC Spare Register 15	0x0000_0000
SPR16	RTC_BA+0x80	R/W	RTC Spare Register 16	0x0000_0000
SPR17	RTC_BA+0x84	R/W	RTC Spare Register 17	0x0000_0000
SPR18	RTC_BA+0x88	R/W	RTC Spare Register 18	0x0000_0000
SPR19	RTC_BA+0x8C	R/W	RTC Spare Register 19	0x0000_0000

6.12.7 Register Description

RTC Initiation Register (INIR)

Register	Offset	R/W	Description	Reset Value
INIR	RTC_BA+0x00	R/W	RTC Initiation Register	0x0000_0000

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

Bits	Description	
[31:1]	**INIR[31:1]**	**RTC Initiation** When RTC block is powered on, RTC is at reset state. User has to write a number (0xa5eb1357) to INIR to make RTC leaving reset state. Once the INIR is written as 0xa5eb1357, the RTC will be in un-reset state permanently. The INIR is a write-only field and read value will be always 0.
[0]	**INIR[0]/Active**	**RTC Active Status (Read Only)** 0 = RTC is at reset state. 1 = RTC is at normal active state.

RTC Access Enable Register (AER)

Register	Offset	R/W	Description	Reset Value
AER	RTC_BA+0x04	R/W	RTC Access Enable Register	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**ENF**	**RTC Register Access Enable Flag (Read Only)** 0 = RTC register read/write access Disabled. 1 = RTC register read/write access Enabled. **Note:** This bit will be set after AER[15:0] is load a 0xA965, and will be cleared automatically after 1024 RTC clocks.
[15:0]	**AER**	**RTC Register Access Enable Password (Write Only)** Writing 0xA965 to this register will enable RTC access and keep 1024 RTC clocks.

RTC Frequency Compensation Register (FCR)

Register	Offset	R/W	Description	Reset Value
FCR	RTC_BA+0x08	R/W	RTC Frequency Compensation Register	0x0000_0700

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**INTEGER**	**Integer Part** Please refer to 6.12.5.4.
[5:0]	**FRACTION**	**Fraction Part** Formula = (fraction part of detected value) x 60. **Note:** Digit in FCR must be expressed as hexadecimal number..

Note: This register can be read back after the ENF (AER[16] RTC Register Access Enable Flag) is active.

RTC Time Loading Register (TLR)

Register	Offset	R/W	Description	Reset Value
TLR	RTC_BA+0x0C	R/W	Time Loading Register	0x0000_0000

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

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21:20]	**10HR**	10-Hour Time Digit (0~2)
[19:16]	**1HR**	1-Hour Time Digit (0~9)
[15]	**Reserved**	Reserved.
[14:12]	**10MIN**	10-Min Time Digit (0~5)
[11:8]	**1MIN**	1-Min Time Digit (0~9)
[7]	**Reserved**	Reserved.
[6:4]	**10SEC**	10-Sec Time Digit (0~5)
[3:0]	**1SEC**	1-Sec Time Digit (0~9)

Note1: TLR is a BCD digit counter and RTC controller will not check the loaded data is reasonable or not.

Note2: The reasonable value range is listed in the parenthesis.

Note3: When RTC runs as 12-hour time scale mode, the high bit of 10HR field means AM/PM.

RTC Calendar Loading Register (CLR)

Register	Offset	R/W	Description	Reset Value
CLR	RTC_BA+0x10	R/W	Calendar Loading Register	0x0005_0101

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**10YEAR**	10-Year Calendar Digit (0~9)
[19:16]	**1YEAR**	1-Year Calendar Digit (0~9)
[15:13]	**Reserved**	Reserved.
[12]	**10MON**	10-Month Calendar Digit (0~1)
[11:8]	**1MON**	1-Month Calendar Digit (0~9)
[7:6]	**Reserved**	Reserved.
[5:4]	**10DAY**	10-Day Calendar Digit (0~3)
[3:0]	**1DAY**	1-Day Calendar Digit (0~9)

Note1: CLR is a BCD digit counter and RTC will not check the loaded data is reasonable or not.

Note2: The reasonable value range is listed in the parenthesis.

RTC Time Scale Selection Register (TSSR)

Register	Offset	R/W	Description	Reset Value
TSSR	RTC_BA+0x14	R/W	Time Scale Selection 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							**24H_12H**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**24H_12H**	**24-Hour / 12-Hour Time Scale Selection** It indicates that RTC TLR and TAR counter are in 24-hour time scale or 12-hour time scale. Please refer to 6.12.5.6. 0 = 24-hour time scale selected. 1 = 24-hour time scale selected.

RTC Day of the Week Register (DWR)

Register	Offset	R/W	Description	Reset Value
DWR	RTC_BA+0x18	R/W	Day of the Week Register	0x0000_0006

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**DWR**	**Day Of The Week Register** 000 = Sunday. 001 = Monday. 010 = Tuesday. 011 = Wednesday. 100 = Thursday. 101 = Friday. 110 = Saturday. 111 = Reserved.

RTC Time Alarm Register (TAR)

Register	Offset	R/W	Description	Reset Value
TAR	RTC_BA+0x1C	R/W	Time Alarm Register	0x0000_0000

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

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21:20]	**10HR**	10-Hour Time Digit of Alarm Setting (0~2)
[19:16]	**1HR**	1-Hour Time Digit of Alarm Setting (0~9)
[15]	**Reserved**	Reserved.
[14:12]	**10MIN**	10-Min Time Digit of Alarm Setting (0~5)
[11:8]	**1MIN**	1-Min Time Digit of Alarm Setting (0~9)
[7]	**Reserved**	Reserved.
[6:4]	**10SEC**	10-Sec Time Digit of Alarm Setting (0~5)
[3:0]	**1SEC**	1-Sec Time Digit of Alarm Setting (0~9)

Note1: This register can be read back after the ENF (AER[16] RTC Register Access Enable Flag) is active.

Note2: TAR is a BCD digit counter and RTC controller will not check the loaded data is reasonable or not.

Note3: The reasonable value range is listed in the parenthesis.

Note4: When RTC runs as 12-hour time scale mode, the high bit of 10HR field means AM/PM.

RTC Calendar Alarm Register (CAR)

Register	Offset	R/W	Description	Reset Value
CAR	RTC_BA+0x20	R/W	Calendar Alarm Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**10YEAR**	10-Year Calendar Digit of Alarm Setting (0~9)
[19:16]	**1YEAR**	1-Year Calendar Digit of Alarm Setting (0~9)
[15:13]	**Reserved**	Reserved.
[12]	**10MON**	10-Month Calendar Digit of Alarm Setting (0~1)
[11:8]	**1MON**	1-Month Calendar Digit of Alarm Setting (0~9)
[7:6]	**Reserved**	Reserved.
[5:4]	**10DAY**	10-Day Calendar Digit of Alarm Setting (0~3)
[3:0]	**1DAY**	1-Day Calendar Digit of Alarm Setting (0~9)

Note1: This register can be read back after the ENF (AER [16] RTC Register Access Enable Flag) is active.

Note2: CAR is a BCD digit counter and RTC will not check the loaded data is reasonable or not.

Note3: The reasonable value range is listed in the parenthesis.

RTC Leap Year Indication Register (LIR)

Register	Offset	R/W	Description	Reset Value
LIR	RTC_BA+0x24	R	Leap Year Indicator Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**LIR**	**Leap Year Indication Register (Read Only)** 0 = This year is not a leap year. 1 = This year is a leap year.

RTC Interrupt Enable Register (RIER)

Register	Offset	R/W	Description	Reset Value
RIER	RTC_BA+0x28	R/W	RTC Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TIER**	**Time Tick Interrupt Enable Bit** This bit is used to enable/disable RTC Time Tick Interrupt, and generate an interrupt signal if TIF (RIIR[1] RTC Time Tick Interrupt Flag) is set to 1. 0 = RTC Time Tick Interrupt Disabled. 1 = RTC Time Tick Interrupt Enabled. **Note:** This bit will also trigger a wake-up event while system runs in Idle/Power-down mode and RTC Time Tick Interrupt signal generated.
[0]	**AIER**	**Alarm Interrupt Enable Bit** This bit is used to enable/disable RTC Alarm Interrupt, and generate an interrupt signal if AIF (RIIR[0] RTC Alarm Interrupt Flag) is set to 1. 0 = RTC Alarm Interrupt Disabled. 1 = RTC Alarm Interrupt Enabled. **Note:** This bit will also trigger a wake-up event while system runs in Idle/Power-down mode and RTC Alarm Interrupt signal generated.

RTC Interrupt Indication Register (RIIR)

Register	Offset	R/W	Description	Reset Value
RIIR	RTC_BA+0x2C	R/W	RTC Interrupt Indicator Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TIF**	**RTC Time Tick Interrupt Flag** When RTC time tick happened, this bit will be set to 1 and an interrupt will be generated if RTC Tick Interrupt enabled TIER (RIER[1]) is set to 1. Chip will also be wake-up if RTC Tick Interrupt is enabled and this bit is set to 1 when chip is running at Power-down mode. 0 = Tick condition does not occur. 1 = Tick condition occur. **Note:** Write 1 to clear this bit.
[0]	**AIF**	**RTC Alarm Interrupt Flag** When RTC time counters TLR and CLR match the alarm setting time registers TAR and CAR, this bit will be set to 1 and an interrupt will be generated if RTC Alarm Interrupt enabled AIER (RIER[0]) is set to 1. Chip will be wake-up if RTC Alarm Interrupt is enabled when chip is at Power-down mode. 0 = Alarm condition is not matched. 1 = Alarm condition is matched. **Note:** Write 1 to clear this bit.

RTC Time Tick Register (TTR)

Register	Offset	R/W	Description	Reset Value
TTR	RTC_BA+0x30	R/W	RTC Time Tick Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**TTR**	**Time Tick Register** These bits are used to select RTC time tick period for Periodic Time Tick Interrupt request. 000 = Time tick is 1 second. 001 = Time tick is 1/2 second. 010 = Time tick is 1/4 second. 011 = Time tick is 1/8 second. 100 = Time tick is 1/16 second. 101 = Time tick is 1/32 second. 110 = Time tick is 1/64 second. 111 = Time tick is 1/28 second. **Note:** This register can be read back after the RTC register access enable bit ENF (AER[16]) is active.

RTC Spare Functional Control Register (SPRCTL)

Register	Offset	R/W	Description	Reset Value
SPRCTL	RTC_BA+0x3C	R/W	RTC Spare Functional Control Register	0x0000_0080

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**SPRRDY**	**SPR Register Ready** This bit indicates if the registers SPRCTL, SPR0 ~ SPR19 are ready to be accessed. After user writing registers SPRCTL, SPR0 ~ SPR19, read this bit to check if these registers are updated done is necessary. 0 = SPRCTL, SPR0 ~ SPR19 updating is in progress. 1 = SPRCTL, SPR0 ~ SPR19 are updated done and ready to be accessed. **Note:** This bit is read only and any write to it won't take any effect.
[6:3]	**Reserved**	Reserved.
[2]	**SPREN**	**SPR Register Enable Bit** 0 = Spare register is Disabled. 1 = Spare register is Enabled. **Note:** When spare register is disabled, RTC SPR0 ~ SPR19 cannot be accessed.
[1:0]	**Reserved**	Reserved.

RTC Spare Register (SPRx)

Register	Offset	R/W	Description	Reset Value
SPR0	RTC_BA+0x40	R/W	RTC Spare Register 0	0x0000_0000
SPR1	RTC_BA+0x44	R/W	RTC Spare Register 1	0x0000_0000
SPR2	RTC_BA+0x48	R/W	RTC Spare Register 2	0x0000_0000
SPR3	RTC_BA+0x4C	R/W	RTC Spare Register 3	0x0000_0000
SPR4	RTC_BA+0x50	R/W	RTC Spare Register 4	0x0000_0000
SPR5	RTC_BA+0x54	R/W	RTC Spare Register 5	0x0000_0000
SPR6	RTC_BA+0x58	R/W	RTC Spare Register 6	0x0000_0000
SPR7	RTC_BA+0x5C	R/W	RTC Spare Register 7	0x0000_0000
SPR8	RTC_BA+0x60	R/W	RTC Spare Register 8	0x0000_0000
SPR9	RTC_BA+0x64	R/W	RTC Spare Register 9	0x0000_0000
SPR10	RTC_BA+0x68	R/W	RTC Spare Register 10	0x0000_0000
SPR11	RTC_BA+0x6C	R/W	RTC Spare Register 11	0x0000_0000
SPR12	RTC_BA+0x70	R/W	RTC Spare Register 12	0x0000_0000
SPR13	RTC_BA+0x74	R/W	RTC Spare Register 13	0x0000_0000
SPR14	RTC_BA+0x78	R/W	RTC Spare Register 14	0x0000_0000
SPR15	RTC_BA+0x7C	R/W	RTC Spare Register 15	0x0000_0000
SPR16	RTC_BA+0x80	R/W	RTC Spare Register 16	0x0000_0000
SPR17	RTC_BA+0x84	R/W	RTC Spare Register 17	0x0000_0000
SPR18	RTC_BA+0x88	R/W	RTC Spare Register 18	0x0000_0000
SPR19	RTC_BA+0x8C	R/W	RTC Spare Register 19	0x0000_0000

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

Bits	Descriptions	
[31:0]	**SPARE**	**Spare Register** This field is used to store back-up information defined by user.. Before storing back-up information in to SPARE register, user should write 0xA965 to AER[15:0] to make sure register read/write enable bit ENF (AER[16]) is active.

6.13 UART Interface Controller (UART)

6.13.1 Overview

The NuMicro® NUC029LEE/NUC029SEE provides up to three channels of Universal Asynchronous Receiver/Transmitters (UART). UART0 supports High Speed UART and UART1~2 perform Normal Speed UART. Besides, only UART0 and UART1 support the flow control function. The UART Controller performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the CPU. The UART controller also supports IrDA SIR Function, LIN master/slave function and RS-485 function mode. Each UART Controller channel supports seven types of interrupts.

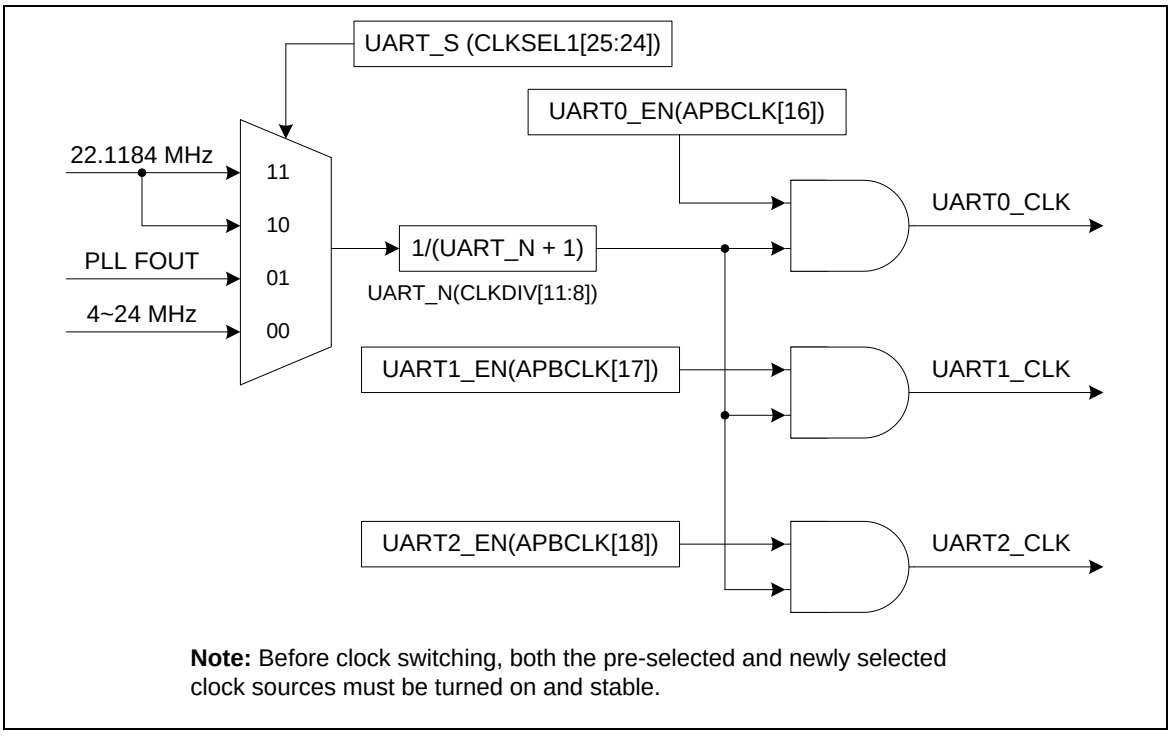
6.13.2 Features

- Full duplex, asynchronous communications
- Separates receive / transmit 64/16/16 bytes (UART0/UART1/UART2) entry FIFO for data payloads
- Supports hardware auto flow control/flow control function (CTS, RTS) and programmable RTS flow control trigger level (UART0 and UART1 support)
- Programmable receiver buffer trigger level
- Supports programmable baud-rate generator for each channel individually
- Supports CTS wake-up function (UART0 and UART1 support)
- Supports 7-bit receiver buffer time-out detection function
- UART0/UART1 can through DMA channels to receive/transmit data
- Programmable transmitting data delay time between the last stop and the next start bit by setting UA_TOR [DLY] register
- Supports break error, frame error, parity error and receive / transmit buffer overflow detect function
- Fully programmable serial-interface characteristics
  - Programmable data bit length, 5-, 6-, 7-, 8-bit character
  - Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
  - Programmable stop bit length, 1, 1.5, or 2 stop bit generation
- IrDA SIR function mode
  - Supports 3-/16-bit duration for normal mode
- LIN function mode
  - Supports LIN master/slave mode
  - Supports programmable break generation function for transmitter
  - Supports break detect function for receiver
- RS-485 function mode.
  - Supports RS-485 9-bit mode

- Supports hardware or software direct enable control provided by RTS pin (UART0 and UART1 support)

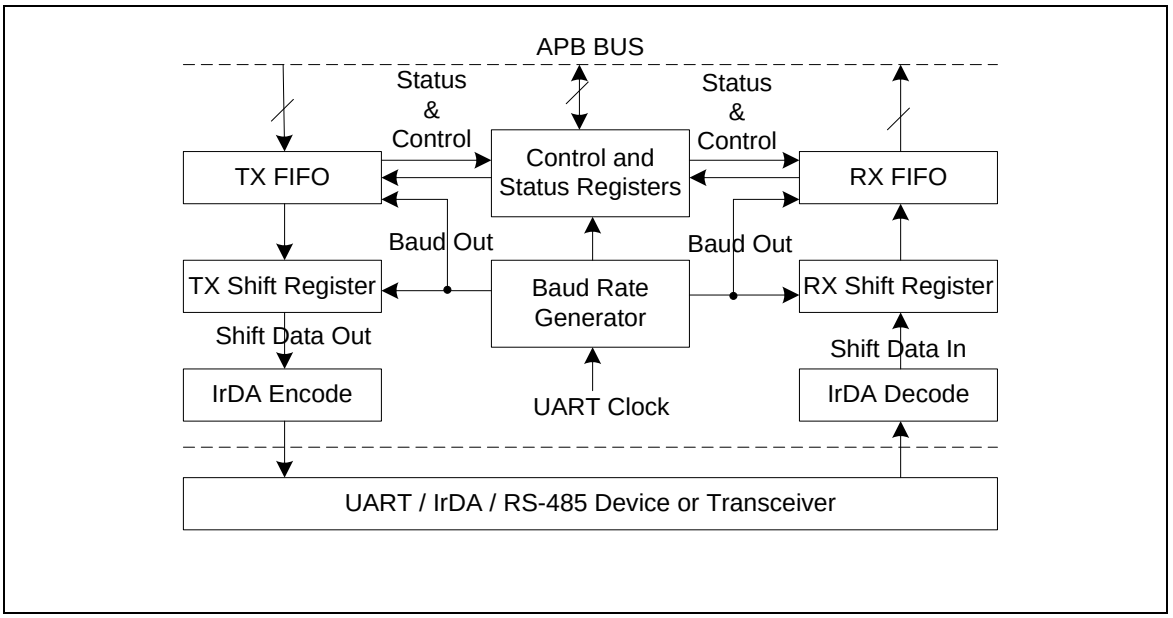
6.13.3 Block Diagram

The UART clock control and block diagram are shown in Figure 5-67 and Figure 5-68 respectively.



Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 6.13-1 UART Clock Control Diagram



Note: UART0 is equipped with 64 bytes FIFO. UART1/UART2 is equipped with 16 bytes FIFO.

Figure 6.13-2 UART Block Diagram

Each block is described in detail as follows:

TX_FIFO
The transmitter is buffered with a 64/16 byte FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO
The receiver is buffered with a 64/16 byte FIFO (plus three error bits per byte) to reduce the number of interrupts presented to the CPU.

TX shift Register
This block is the shifting the transmitting data out of serially control.

RX shift Register
This block is the shifting the receiving data in of serially control.

Modem Control 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. Refer to baud rate equation.

IrDA Encode
This block is IrDA encode control block.

IrDA Decode
This block is IrDA decode control block.

Control and Status Register
This field is register set that including the FIFO control registers (UA_FCR), FIFO status registers (UA_FSR), and line control register (UA_LCR) for transmitter and receiver. The time-out control register (UA_TOR) identifies the condition of time-out interrupt. This register set also includes the interrupt enable register (UA_IER) and interrupt status register (UA_ISR) to enable or disable the responding interrupt and to identify the occurrence of the responding interrupt. There are seven types of interrupts, transmitter FIFO empty interrupt(THRE_INT), receiver threshold level reaching interrupt (RDA_INT), line status interrupt (parity error or framing error or break interrupt) (RLS_INT), time-out interrupt (TOUT_INT), MODEM/Wake-up status interrupt (MODEM_INT), Buffer error interrupt (BUF_ERR_INT) and LIN receiver break field detected interrupt (LIN _INT).

6.13.4 Basic Configuration

The UART Controller function pins are configured in PB_MFP, PD_MFP, ALT_MFP, ALT_MFP1 and ALT_MFP2 registers.

The UART Controller clock are enabled in UART0_EN(APBCLK[16]) for UART0, UART1_EN (APBCLK[17]) for UART1 and UART2_EN(APBCLK[18]) for UART2.

The UART Controller clock source is selected by UART_S(CLKSEL[25:24]).

The UART Controller clock prescaler is determined by UART_N(CLKDIV[11:8]).

UART Interface Controller Pin description is shown as following:

Pin	Type	Description
UART_TXD	Output	UART transmit
UART_RXD	Input	UART receive
UART_nCTS	Input	UART modem clear to send
UART_nRTS	Output	UART modem request to send

Table 6.13-1 UART Interface Controller Pin

6.13.5 Functional Description

The UART Controller supports four function modes including UART, IrDA, LIN and RS-485 mode. User can select a function by setting the UA_FUN_SEL register. The four function modes will be described in following section.

6.13.5.1 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. The baud rate equation is Baud Rate = UART_CLK / M * [BRD + 2], where M and BRD are defined in Baud Rate Divider Register (UA_BAUD). The following tables list the UART baud rate equations in the various conditions and UART baud rate parameter settings. There is no error for the baud rate results calculated through the baud rate parameter and register setting below. In IrDA function mode, the baud rate generator must be set in Mode 0.

Mode	DIV_X_EN	DIV_X_ONE	Divider X	BRD	Baud Rate Equation
0	0	0	Don't care	A	UART_CLK / [16 * (A+2)].
1	1	0	B	A	UART_CLK / [(B+1) * (A+2)] , B must >= 8.
2	1	1	Don't care	A	UART_CLK / (A+2), If UART peripheral clock<= HCLK, A must >=9. If HCLK <UART peripheral clock <= 2*HCLK, A must >=15. If 2*HCLK <UART peripheral clock <= 3*HCLK, A must >=21. If UART peripheral clock > 3*HCLK, it is unsupported. UART peripheral clock = UART clock source/(UART clock divider number+1).

Table 6.13-2 UART Baud Rate Equation

UART Peripheral Clock = 22.1184 MHz			
Baud Rate	**Mode 0**	**Mode 1**	**Mode 2**
921600	Not support	A=0, B=11	A=22
460800	A=1	A=1, B=15 A=2, B=11	A=46

230400	A=4	A=4, B=15 A=6, B=11	A=94
115200	A=10	A=10, B=15 A=14, B=11	A=190
57600	A=22	A=22, B=15 A=30, B=11	A=382
38400	A=34	A=62, B=8 A=46, B=11 A=34, B=15	A=574
19200	A=70	A=126, B=8 A=94, B=11 A=70, B=15	A=1150
9600	A=142	A=254, B=8 A=190, B=11 A=142, B=15	A=2302
4800	A=286	A=510, B=8 A=382, B=11 A=286, B=15	A=4606

Table 6.13-3 UART Controller Baud Rate Parameter Setting Table

UART Peripheral Clock = 22.1184 MHz			
Baud Rate	**Mode 0**	**Mode 1**	**Mode 2**
921600	Not support	0x2B00_0000	0x3000_0016
460800	0x0000_0001	0x2F00_0001 0x2B00_0002	0x3000_002E
230400	0x0000_0004	0x2F00_0004 0x2B00_0006	0x3000_005E
115200	0x0000_000A	0x2F00_000A 0x2B00_000E	0x3000_00BE
57600	0x0000_0016	0x2F00_0016 0x2B00_001E	0x3000_017E
38400	0x0000_0022	0x2800_003E 0x2B00_002E 0x2F00_0022	0x3000_023E
19200	0x0000_0046	0x2800_007E 0x2B00_005E 0x2F00_0046	0x3000_047E
9600	0x0000_008E	0x2800_00FE 0x2B00_00BE 0x2F00_008E	0x3000_08FE
4800	0x0000_011E	0x2800_01FE 0x2B00_017E 0x2F00_011E	0x3000_11FE

Table 6.13-4 UART Controller Baud Rate Register (UA_BAUD) Setting Table

6.13.5.2 UART Controller Transmit Delay Time Value

The UART Controller programs DLY (UA_TOR [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

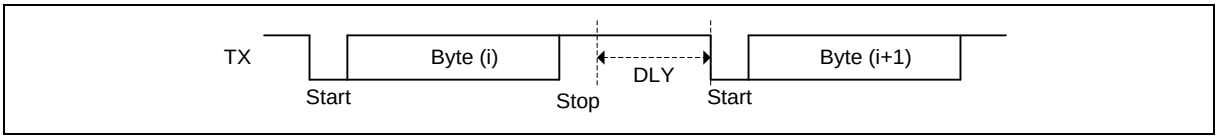


Figure 6.13-3 Transmit Delay Time Operation

6.13.5.3 UART Controller FIFO Control and Status

The UART0 is built-in with a 64-byte transmitter FIFO (TX_FIFO) and a 64-byte receiver FIFO (RX_FIFO) that reduces the number of interrupts presented to the CPU. The UART1~2 are equipped with 16-byte transmitter FIFO (TX_FIFO) and 16-byte receiver FIFO (RX_FIFO).The CPU can read the status of the UART at any time during operation. The reported status information includes the type and condition of the transfer operations being performed by the UART, as well as 3 error conditions (parity error, framing error, break interrupt) probably occur

while receiving data. This FIFO control and status also support all of UART, IrDA, LIN and RS-485 function mode.

6.13.5.4 UART Controller Wake-up Function

When the chip is in Power-down mode, an external CTS change will wake up chip from Power-down mode. This wake-up function is available in every function mode and it is supported for UART0 and UART1. User must enable the MODEN_INT interrupt to use the wake-up function.

6.13.5.5 UART Controller Interrupt and Status

Each UART Controller supports seven types of interrupts including:

- Receiver threshold level reached interrupt (RDA_INT)
- Transmitter FIFO empty interrupt (THRE_INT)
- Line status interrupt (parity error, frame error or break interrupt) (RLS_INT)
- MODEM/Wake-up status interrupt (MODEM_INT)
- Receiver buffer time-out interrupt (TOUT_INT)
- Buffer error interrupt (BUF_ERR_INT)
- LIN bus interrupt (LIN_ INT)

The Table 6.13-5 and Table 6.13-6 describe 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 Cleared By
Receive Data Available Interrupt	RDA_INT	RDA_IEN	**RDA_IF**	Read UA_RBR
Transmit Holding Register Empty Interrupt	THRE_INT	THRE_IEN	**THRE_IF**	Write UA_THR
Receive Line Status Interrupt	RLS_INT	RLS_IEN	**RLS_IF** = BIF	**Writing "1" to** BIF
			RLS_IF = FEF	**Writing "1" to** FEF
			RLS_IF = PEF	**Writing "1" to** PEF
			RLS_IF = RS485_ADD_DETF	**Writing '1' to** RS485_ADD_DETF
Modem Status Interrupt	MODEM_INT	MODEM_IEN	**MODEM_IF** = DCTSF	**Write "1" to** DCTSF
RX Time-out Interrupt	TOUT_INT	RTO_IEN	**TOUT_IF**	Read UA_RBR
Buffer Error Interrupt	BUF_ERR_INT	BUF_ERR_IEN	**BUF_ERR_IF** =TX_OVER_IF	**Write "1" to** TX_OVER_IF
			BUF_ERR_IF = RX_OVER_IF	**Write "1" to** RX_OVER_IF
LIN Bus Interrupt	LIN_ _INT	LIN _IEN	**LIN_IF =** LIN_BKDET_F	**Write "1" to** LIN_IF **and** **Write "1" to** LIN_BKDET_F
			LIN_IF = BIT_ERR_F	**Write "1" to** BIT_ERR_F

			LIN_IF = LIN_IDPERR_F	**Write "1" to o** LIN_IDPERR_F
			LIN_IF = LINS_HERR_F	**Write "1" to** LINS_HERR_F
			LIN_IF = LINS_HDET_F	**Write "1" to** LINS_HDET_F

Table 6.13-5 UART Controller Interrupt Source and Flag List

UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator To Interrupt Controller	Interrupt Flag	Flag Cleared By
Receive Data Available Interrupt	RDA_INT	RDA_IEN	**HW_RDA_IF**	Read UA_RBR
Transmit Holding Register Empty Interrupt	THRE_INT	THRE_IEN	**HW_THRE_IF**	Write UA_THR
Receive Line Status Interrupt	RLS_INT	RLS_IEN	**HW_RLS_IF** = BIF	**Write '1' to** RFR
			HW_RLS_IF = FEF	
			HW_RLS_IF = PEF	
			HW_RLS_IF=RS485_ADD_DET	
Modem Status Interrupt	MODEM_INT	MODEM_IEN	**MODEM_IF** = DCTSF	**Write "1" to** DCTSF
RX Time-out Interrupt	TOUT_INT	RTO_IEN	**HW_TOUT_IF**	Read UA_RBR
Buffer Error Interrupt	BUF_ERR_INT	BUF_ERR_IEN	**HW_BUF_ERR_IF** = TX_OVER_IF	**Write "1" to** RFR
			HW_BUF_ERR_IF = RX_OVER_IF	

Table 6.13-6 Controller Interrupt Source and Flag in DMA Mode List

6.13.5.6 UART Function Mode

The UART Controller provides UART function (user must set UA_FUN_SEL [1:0] to "00" 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 (UA_TOR [15:8]) register. The UART supports hardware auto-flow control and flow control function (CTS, RTS), programmable RTS flow control trigger level and fully programmable serial-interface characteristics.

UART Line Control Function

The UART Controller supports fully programmable serial-interface characteristics by setting the UA_LCR register. Software can use the UA_LCR register to program the word length, stop bit and parity bit. The following tables list the UART word and stop bit length settings and the UART parity bit settings.

NSB (UA_LCR[2])	WLS (UA_LCR[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 6.13-7 UART Line Control of Word and Stop Length Setting

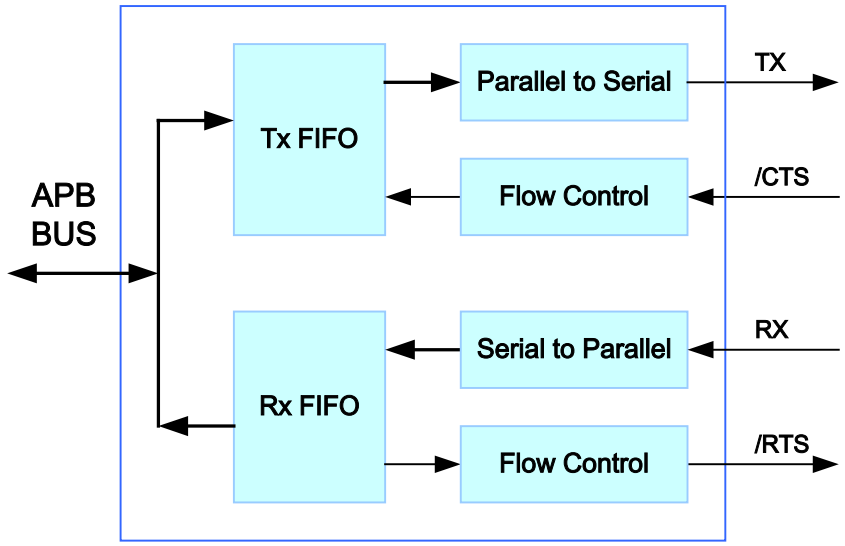
Parity Type	SPE (UA_LCR[5])	EPE (UA_LCR[4])	PBE (UA_LCR[3])	Description
No Parity	x	x	0	No parity bit output.
Odd Parity	0	0	1	Odd Parity is calculated by adding all the “1’s” in a data stream and adding a parity bit to the total bits, to make the total count an odd number.
Even Parity	0	1	1	Even Parity 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	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	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 6.13-8 UART Line Control of Parity Bit Setting

UART Auto-Flow Control Function

The UART supports auto-flow control function that uses two signals, CTS (clear-to-send) and RTS (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 RTS to external device. When the number of bytes in the RX FIFO equals the value of RTS_TRI_LEV (UA_FCR [19:16]), the RTS is de-asserted. The UART sends data out when UART detects CTS is asserted from external device. If the valid asserted CTS is not detected, the UART will not send data out.

The Figure 6.13-4 demonstrates the auto-flow control block.



Note: It is only supported in UART0 and UART1

Figure 6.13-4 Auto Flow Control Block Diagram

The Figure 6.13-5 demonstrates the CTS auto flow control of UART function mode. User must set AUTO_CTS_EN (UA_IER [13]) to enable CTS auto flow control function. The LEV_CTS (UA_MCR [8]) can set CTS pin input active state. The DCTSF (UA_MSR [0]) is set when any state change of CTS pin input has occurred, and then TX data will be automatically transmitted from TX FIFO.

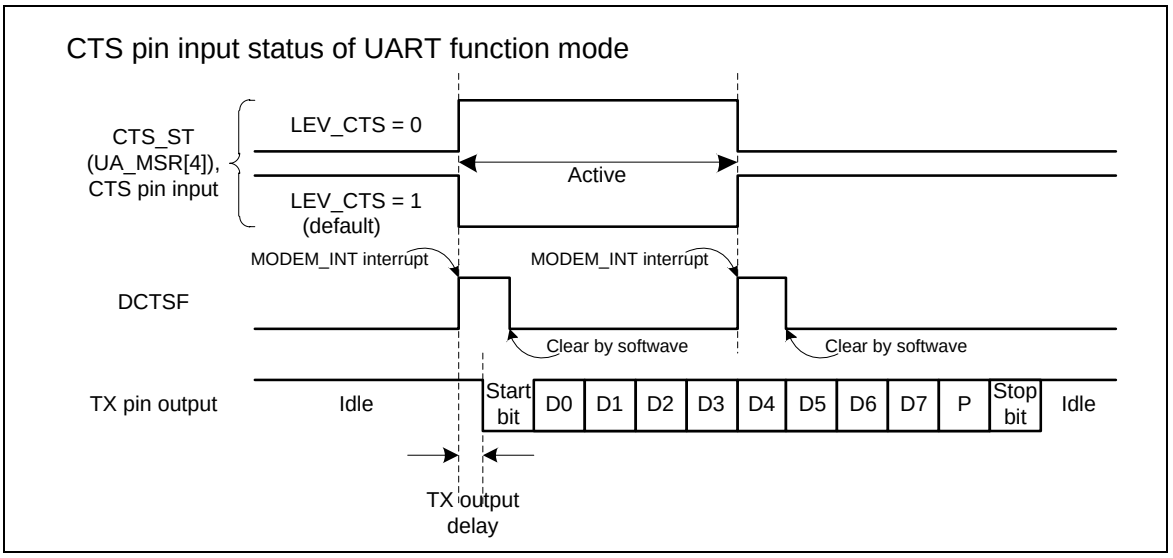


Figure 6.13-5 UART CTS Auto Flow Control Enabled

As shown in theFigure 6.13-6, in UART RTS Auto Flow control mode (AUTO_RTS_EN (UA_IER[12])=1), the RTS internal signal is controlled by UART FIFO controller with RTS_RTI_LEV(UA_FCR[19:16]) trigger level.

Setting LEV_RTS(UA_MCR[9]) can control the RTS pin output is inverse or non-inverse from

RTS signal. User can read the RTS_ST(UA_MCR[13]) bit to get real RTS pin output voltage logic status.

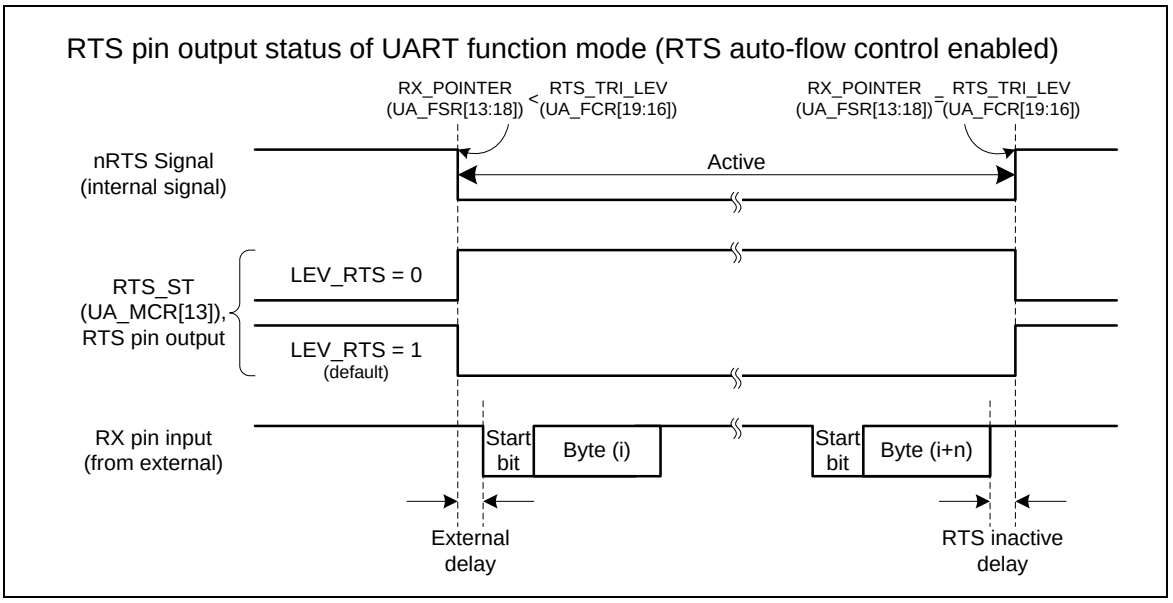


Figure 6.13-6 UART RTS Auto Flow Control Enabled

As shown in the Figure 6.13-7, in software mode (AUTO_RTS_EN(UA_IER[12])=0) the RTS flow is directly controlled by software programming of RTS(UA_MCR[1]) control bit.

Setting LEV_RTS(UA_MCR[9]) can control the RTS pin output is inverse or non-inverse from RTS(UA_MCR[1]) control bit. User can read the RTS_ST(UA_MCR[13]) bit to get real RTS pin output voltage logic status.

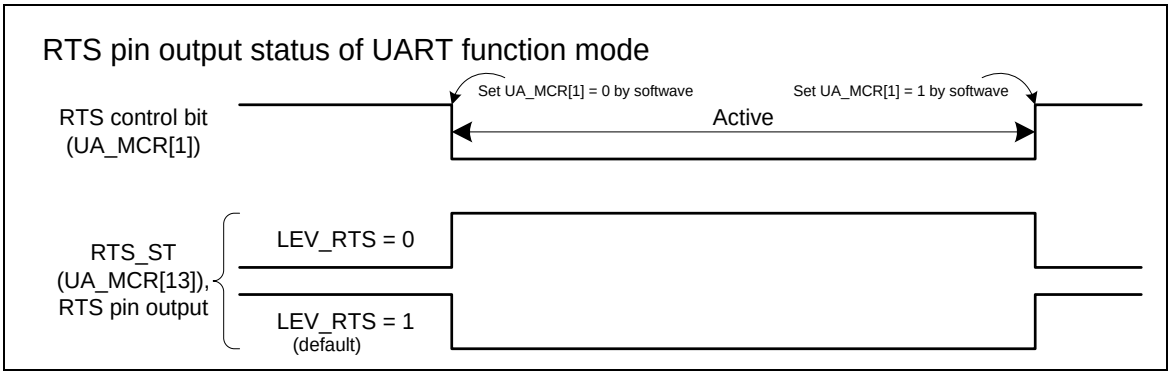


Figure 6.13-7 UART RTS Flow with Software Control

6.13.5.7 IrDA Function Mode

The UART Controller also provides Serial IrDA (SIR, Serial Infrared) function (user must set UA_FUN_SEL [1:0] to '10' 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 DIV_X_EN (UA_BAUD [29]) register must be disabled.

Baud Rate = Clock / (16 * BRD), where BRD is Baud Rate Divider in UA_BAUD register.

The Figure 6.13-8 demonstrates the IrDA control block diagram.

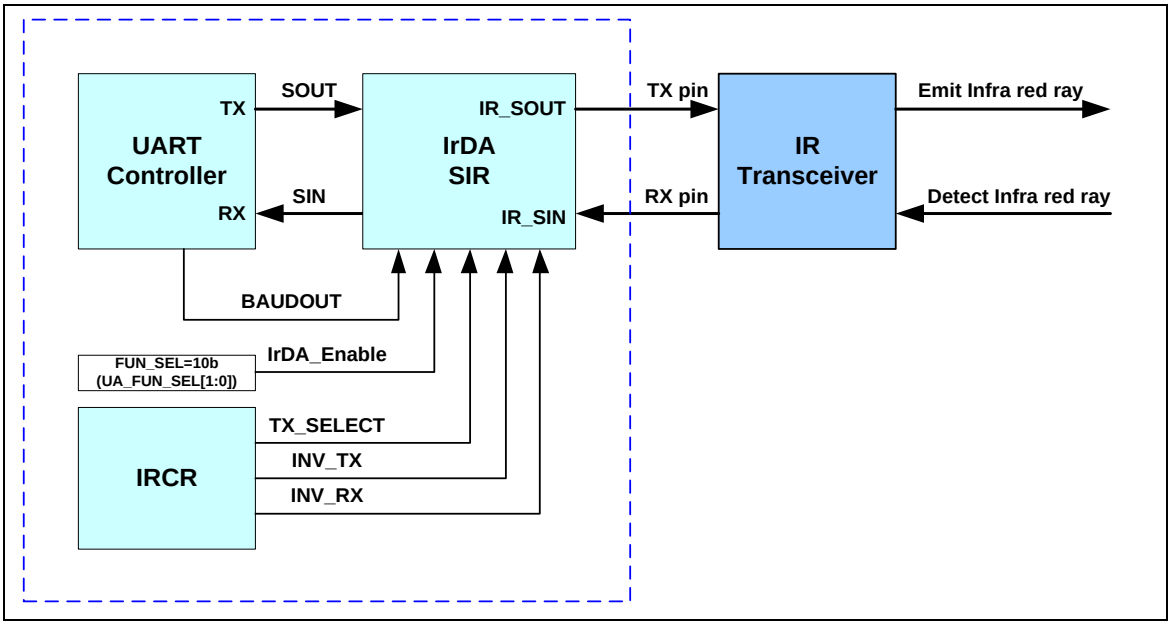


Figure 6.13-8 IrDA Control Block Diagram

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.

In Normal mode, 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. The decoder input is normally high in idle state. (Because of this, IRCR (INV_RX [6]) should be set as 1 by default).

A start bit is detected when the decoder input is LOW.

IrDA SIR Operation

The IrDA SIR encoder/decoder provides functionality which converts between UART data stream and half-duplex serial SIR interface. The Figure 6.13-9 is IrDA encoder/decoder waveform.

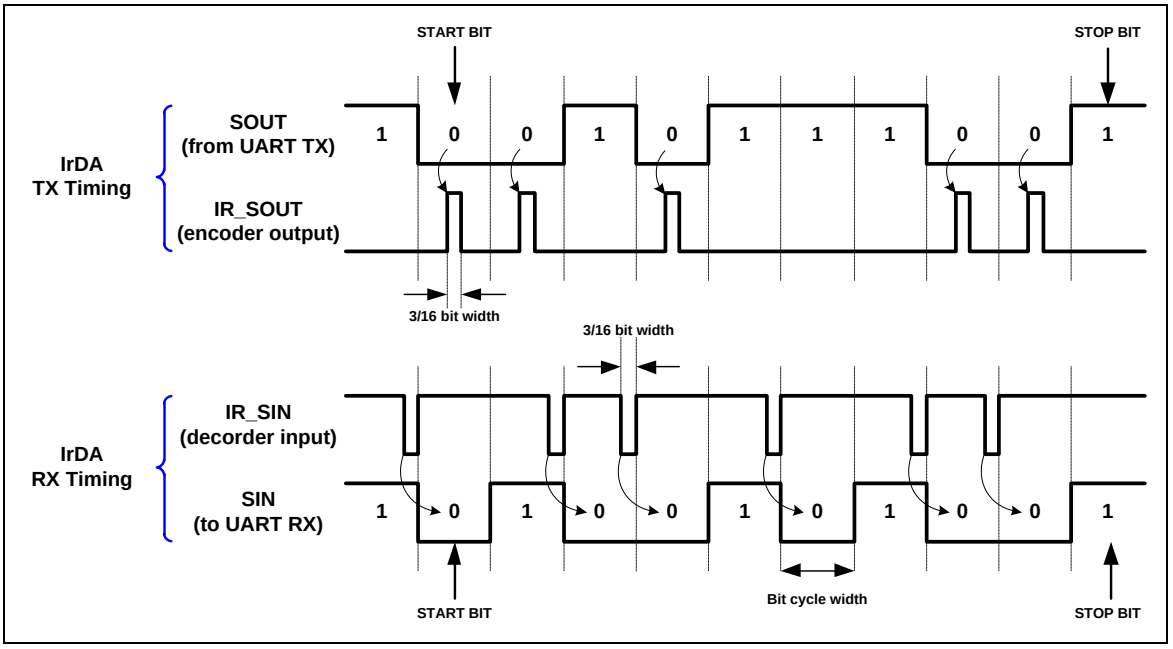


Figure 6.13-9 IrDA TX/RX Timing Diagram

6.13.5.8LIN (Local Interconnection Network) Mode

The UART0~UART2 supports LIN function. Setting FUN_SEL (UA_FUN_SEL[1:0]) to '01' to select LIN mode operation. The UART0~UART2 supports LIN break/delimiter generation and break/delimiter detection in LIN master mode, and supports header detection and automatic resynchronization in LIN Slave mode.

6.13.5.8.1 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), followed by 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. The Figure 6.13-10 is the structure of LIN Frame.

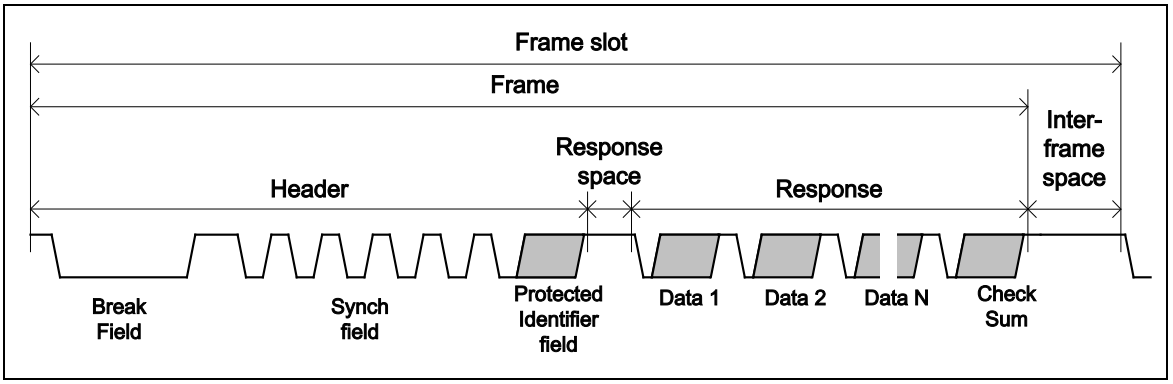


Figure 6.13-10 Structure of LIN Frame

6.13.5.8.2 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 as Figure 6.13-11.

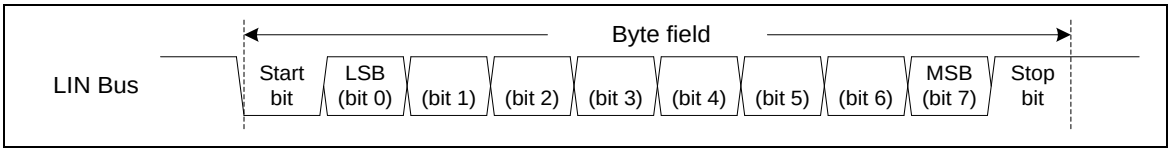


Figure 6.13-11 Structure of LIN Byte

6.13.5.8.3 LIN Master Mode

The UART0~UART2 controller supports LIN Master mode. To enable and initialize the LIN Master mode, the following steps are necessary:

1. Setting the UA_BAUD register to select the desired baud rate.
2. Setting WLS (UA_LCR[1:0]) to “11” to configure the data length with 8 bits, clearing PBE (UA_LCR[3]) bit to disable parity check and clearing NSB (UA_LCR[2]) bit to configure with one stop bit.
3. Setting FUN_SEL (UA_FUN_SEL[1:0]) to “01” 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 UART0/UART1 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 LIN_HEAD_SEL (UA_LIN_CTL[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 UA_THR 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 UA_THR 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 LIN_PID (UA_LIN_CTL [31:24]). When operating in header selected mode in which the selected header is “break field, sync field and frame ID field”, the frame ID parity bit can be calculated by software or hardware depending whether the LIN_IDPEN (UA_LIN_CTL[9]) bit is set or not.

LIN_HEAD_SEL	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 LIN_PID (UA_LIN_CTL[31:24]) first

Table 6.13-9 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 BIT_ERR_EN (UA_LIN_CTL [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 UA_RBR register. The following sequence is a program sequence example.

The procedure without software error monitoring in Master mode:

1. Fill Protected Identifier to LIN_PID (UA_LIN_CTL[31:24]).
2. Select the hardware transmission header field including “break field + sync field + protected identifier field” by setting LIN_HEAD_SEL (UA_LIN_CTL [23:22]) to 10
3. Setting LIN_SHD (UA_LIN_CTL[8]) bit to 1 for requesting header transmission.
4. Wait until LIN_SHD (UA_LIN_CTL[8]) bit cleared by hardware.
5. Wait until TE_FLAG (UA_FSR[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 LIN_BKFL (UA_LIN_CTL [19:16]) and LIN_BS_LEN (UA_LIN_CTL[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 LIN_BS_LEN (UA_LIN_CTL[21:20]) and DLY(UA_TOR[7:0]) 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 LIN_PID (UA_LIN_CTL[31:24]) before trigger header transmission (setting the LIN_SHD (UA_LIN_CTL[8]). The frame ID parity can be generated by software or hardware depending on LIN_IDPEN (UA_LIN_CTL[9]) setting. If the parity generated by software with LIN_IDPEN (UA_LIN_CTL[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 LIN_IDPEN (UA_LIN_CTL[9]) is set to ‘1’ , software fill ID0~ID5 and hardware calculates P0 and P1.

Procedure with software error monitoring in Master mode:

1. Choose the hardware transmission header field only including “break field” by setting LIN_HEAD_SEL (UA_LIN_CTL [23:22])] to ‘00’.
2. Enable break detection function by setting LIN_BKDET_EN (UA_LIN_CTL[10]).
3. Request break + break/sync delimiter transmission by setting the LIN_SHD (UA_LIN_CTL[8])
4. Wait until the LIN_BKDET_F (UA_LIN_SR[8]) flag is set to “1” by hardware..
5. Request sync field transmission by writing 0x55 into UA_THR register.
6. Wait until the RDA_IF (UA_ISR[0]) is set to “1” by hardware and then read back the UA_RBR register.
7. Request header frame ID transmission by writing the protected identifier value to UA_THR register.
8. Wait until the RDA_IF (UA_ISR[0]) is set to “1” by hardware and then read back the UA_RBR register.

LIN break and delimiter detection

When software enables the break detection function by setting LIN_BKDET_EN (UA_LIN_CTL[10]), the break detection circuit is activated. The break detection circuit is totally independent from the UART0/UART1 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 LIN_BKDET_F (UA_LIN_SR[8]) flag is set at the end of break field. If the LIN_IEN (UA_IER[8]) bit is set to 1, an interrupt LIN_INT (UA_ISR[15]) will be generated. The behavior of the break detection and break flag are shown in the Figure 6.13-12.

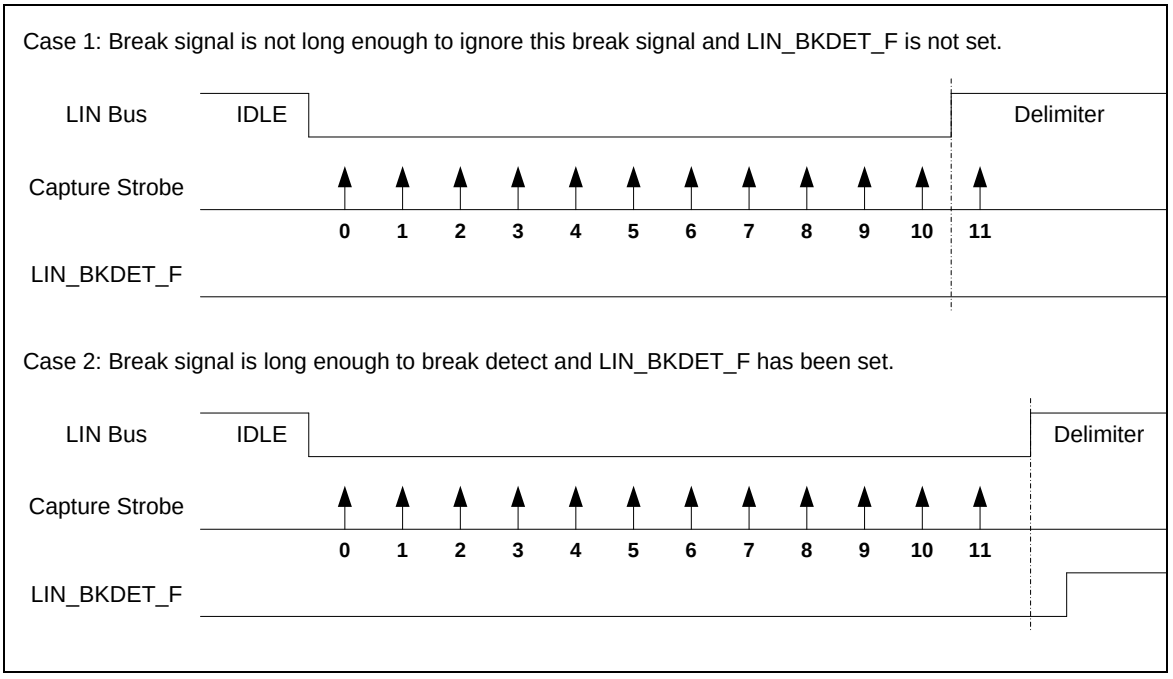


Figure 6.13-12 Break Detection in LIN Mode

LIN break and delimiter detection

The LIN master can transmit response (master is the publisher of the response) and receive response (master is the subscriber of the response). When the master is the publisher of the response, the master sends response by writing the UA_THR register. If the master is the subscriber of the response, the master will receive response from other slave node.

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]) = 1.

If the parity generated by hardware, 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.

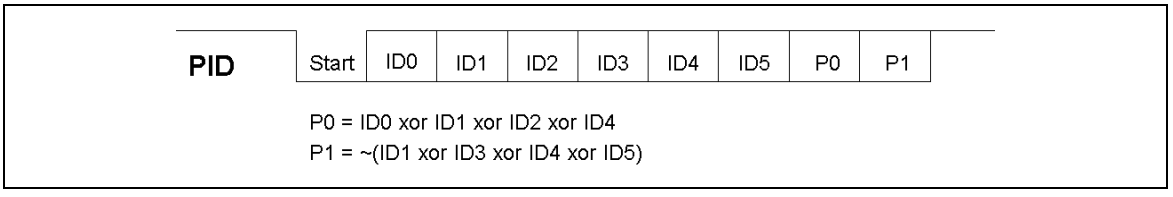


Figure 6.13-13 LIN Frame ID and Parity Format

6.13.5.8.4 LIN Slave Mode

The UART0/UART1 controller supports LIN Slave mode. To enable and initialize the LIN Slave mode, the following steps are necessary:

1. Setting the UA_BAUD register to select the desired baud rate.
2. Configure the data length to 8 bits by setting WLS (UA_LCR[1:0]) to '11' and disable parity check by clearing PBE (UA_LCR[3]) bit and configure with one stop bit by clearing NSB (UA_LCR[2]) bit.
3. Select LIN function mode by setting FUN_SEL (UA_FUN_SEL[1:0]) to "01".
4. Enable LIN slave mode by setting the LINS_EN (UA_LIN_CTL[0]) to 1.

LIN header reception

According to the LIN protocol, a slave node must wait for a valid header which cames 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 LINS_HDET_EN (UA_LIN_CTL[10]) to detect complete frame header (receive "break field", "sync field" and "frame ID field"). When a LIN header is received, the LINS_HDET_F (UA_LIN_SR[0]) flag will be set. If the LIN_IEN (UA_IER[8]) bit is set to 1, an interrupt will be generated. User can enable the frame ID parity check function by setting LIN_IDPEN (UA_LIN_CTL[9]). If only received frame ID parity is not correct (break and sync filed are correct), the LIN_IDPERR_F (UA_LIN_SR[2]) flag is set to '1'. If the LIN_IEN(UA_IER[8]) is set to 1, an interrupt will be generated and LINS_HDET_F (UA_LIN_SR[0]) is set to '1'. User can also put LIN in mute mode by setting LIN_MUTE_EN (UA_LIN_CTL[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 LINS_ARS_EN (UA_LIN_CTL[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 UA_THR 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 LINS_HERR_F (UA_LIN_SR [1]) will be set. The time-out counter is enabled at each break detect edge and stopped in the following conditions.

- A LIN frame ID field has been received.
- The header error flag asserts.
- Writing 1 to the LINS_SYNC_F (UA_LIN_SR[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 LIN_MUTE_EN (UA_LIN_CTL[4]) and exiting from Mute mode condition can be selected by LIN_HEAD_SEL (UA_LIN_CTL[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 LIN_HEAD_SEL (UA_LIN_CTL[23:22]) is set to “break field”, when LIN slave controller detects a valid LIN break + 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 LIN_HEAD_SEL (UA_LIN_CTL[23:22]) is set to “break field and sync field”, when the LIN slave controller detects a valid LIN break + 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 LIN_HEAD_SEL (UA_LIN_CTL[23:22]) is set to “break field, sync field and ID field”, when the LIN slave controller detects a valid LIN break + delimiter and valid sync field without frame error followed by ID data without frame error and received ID data matched LIN_PID (UA_LIN_CTL[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

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 UA_BAUD register.
2. Select LIN function mode by setting UA_FUN_SEL (UA_FUN_SEL[1:0]) to ‘01’.
3. Disable automatic resynchronization function by setting LINS_ARS_EN (UA_LIN_CTL[2]) is set to 0.
4. Enable LIN slave mode by setting the LINS_EN (UA_LIN_CTL[0]) is set to 1.

Slave mode with automatic resynchronization

In Automatic Resynchronization 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 UA_BAUD register.
2. Select LIN function mode by setting UA_FUN_SEL (UA_FUN_SEL[1:0]) to “01”.
3. Enable automatic resynchronization function by setting LINS_ARS_EN (UA_LIN_CTL[2]) to 1.
4. Enable LIN slave mode by setting the LINS_EN (UA_LIN_CTL[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 UART peripheral clock and the result of this measurement is stored in an internal 13-bit register and the UA_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 LIN_HERR_F (UA_LIN_SR [1]) will be set.

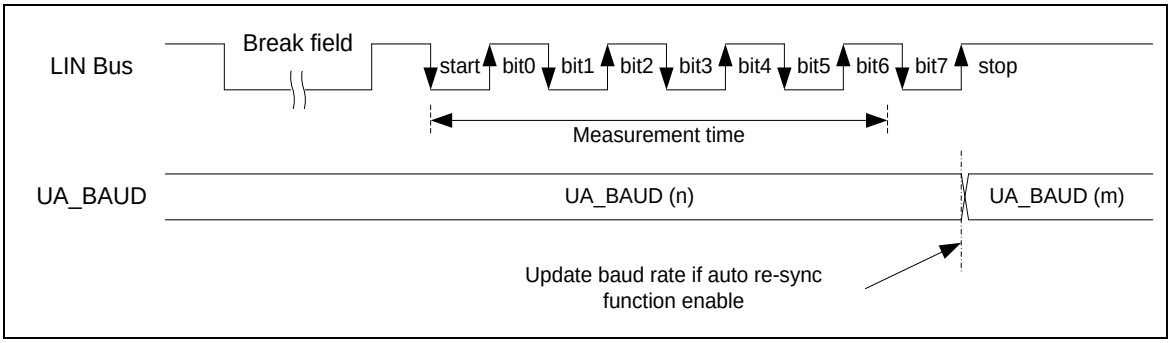


Figure 6.13-14 LIN Sync Field Measurement

When operating in Automatic Resynchronization mode, software must select the desired baud rate by setting the UA_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 UART 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 UA_BAUD register automatically.

In order 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 LINS_DUM_EN (UA_LIN_CTL [3]) to enable auto reload initial baud rate value function. If the LINS_DUM_EN (UA_LIN_CTL [3]) is set, when received the next character, hardware will auto reload the initial value to UA_BAUD, and when the UA_BAUD be updated, the LINS_DUM_EN (UA_LIN_CTL [3]) will be cleared automatically. The behavior of LIN updated method as shown in the following figure.

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

Note2: When a header error is detected, user must write 1 to LINS_SYNC_F (UA_LIN_SR[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 (DIV_X_EN (UA_BAUD [29]) and DIV_X_ONE (UA_BAUD[28]) must be 1).

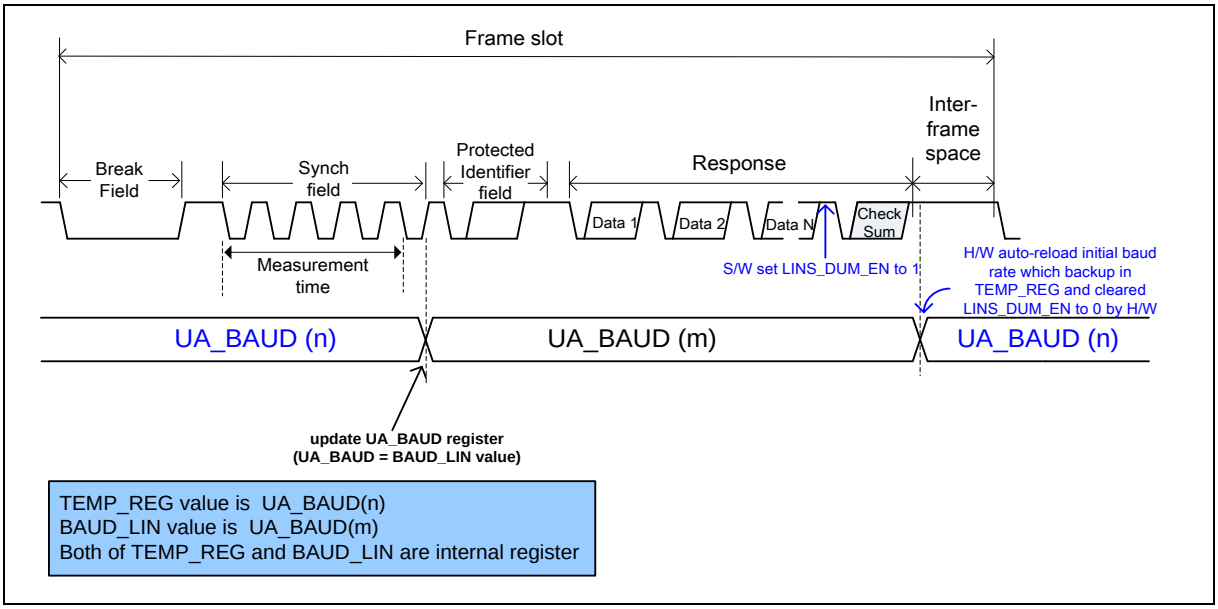


Figure 6.13-15 UA_BAUD Update Sequence in Automatic Resynchronization Mode when LINS_DUM_EN (UA_LIN_CTL[3]) = 1

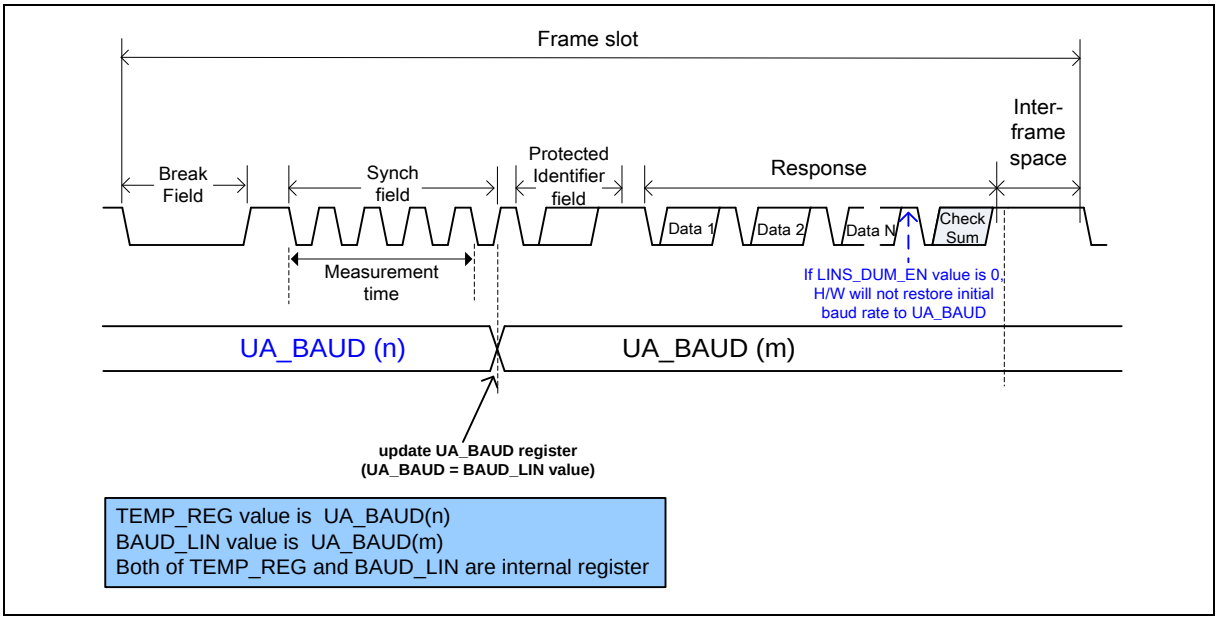


Figure 6.13-16 UA_BAUD Update Sequence in Automatic Resynchronization Mode when LINS_DUM_EN (UA_LIN_CTL[3])= 0

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 LINS_HERR_F (UA_LIN_SR[1]) will be set.
- If the difference is less than 14.06%, the header error flag LINS_HERR_F (UA_LIN_SR[1]) will not be set.
- If the difference is between 14.84% and 14.06%, the header error flag LINS_HERR_F (UA_LIN_SR[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 LINS_HERR_F (UA_LIN_SR[1]) will be set.
- If the difference is less than 15.62%, the header error flag LINS_HERR_F (UA_LIN_SR[1]) will not be set.
- If the difference is between 18.75% and 15.62%, the header error flag LINS_HERR_F (UA_LIN_SR[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 LINS_DUM_EN (UA_LIN_CTL[3]) register (It is recommend setting the LINS_DUM_EN (UA_LIN_CTL[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 LINS_HDET_EN (UA_LIN_CTL[1]), hardware will handle the header detect flow. If the header has an error, the LIN header error flag LIN_HERR_F (UA_LIN_SR[1]) will be set and an interrupt is generated if the LIN_IEN (UA_IER[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 LINS_SYNC_F (UA_LIN_SR[3]) to re-search a new frame header.

The LIN header error flag LIN_HERR_F (UA_LIN_SR[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.

6.13.5.9RS-485 Function Mode

Another alternate function of UART Controller is RS-485 function (user must set UA_FUN_SEL [1:0] to “11” to enable RS-485 function), and direction control provided by RTS pin from an asynchronous serial port. The RS-485 transceiver control is implemented by using the RTS 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 UA_LCR register to control the 9-th bit (When the PBE(UA_LCR[3]), EPE(UA_LCR[4]) and SPE(UA_LCR[5]) 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 UA_ALT_CSR register, and drive the transfer delay time between the last stop bit leaving the TX FIFO and the de-assertion of by setting DLY (UA_TOR [15:8]) register.

6.13.5.9.1 *RS-485 Normal Multidrop Operation Mode (NMM)*

In RS-485 Normal Multidrop Operation Mode (RS485_NMM(UA_ALT_CSR[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 RX_DIS (UA_FCR [8]) then enable RS485_NMM (UA_ALT_CSR [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 RX_DIS (UA_FCR [8]) then enable RS485_NMM (UA_ALT_CSR [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 RX_DIS (UA_FCR [8]) can decide whether accepting the following data bytes are stored in the RX FIFO. If software disables receiver by setting RX_DIS (UA_FCR [8]) register, when a next address byte is detected, the controller will clear the RX_DIS (UA_FCR [8]) bit and the address byte data will be stored in the RX FIFO.

6.13.5.9.2 *RS-485 Auto Address Detection Operation Mode (AAD)*

In RS-485 Auto Address Detection Operation Mode (RS485_AAD(UA_ALT_CSR[9]) = 1), the receiver will ignore any data until an address byte is detected (bit 9 = 1) and the address byte data matches the ADDR_MATCH (UA_ALT_CSR[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 ADDR_MATCH (UA_ALT_CSR[31:24]) value.

6.13.5.9.3 *RS-485 Auto Direction Mode (AUD)*

Another option function of RS-485 controllers is RS-485 auto direction control function (RS485_AUD(UA_ALT_CSR[10) = 1). The RS-485 transceiver control is implemented by using the RTS control signal from an asynchronous serial port. The RTS line is connected to the RS-485 transceiver enable pin such that setting the RTS line to high (logic 1) enables the RS-485 transceiver. Setting the RTS line to low (logic 0) puts the transceiver into the tri-state condition to disabled. User can set LEV_RTS in UA_MCR register to change the RTS driving level.

The following diagram demonstrates the RS-485 RTS driving level in AUD mode. The RTS pin will be automatically driven during TX data transmission.

Setting LEV_RTS(UA_MCR[9]) can control RTS pin output driving level. User can read the

RTS_ST(UA_MCR[13]) bit to get real RTS pin output voltage logic status.

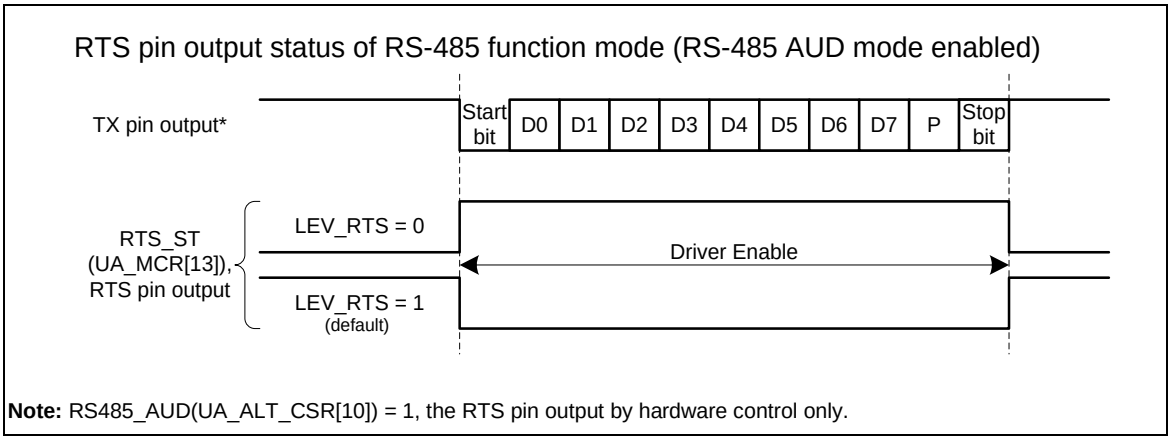


Figure 6.13-17 RS-485 RTS Driving Level in Auto Direction Mode

The following diagram demonstrates the RS-485 RTS driving level in software control (RS485_AUD(UA_ALT_CSR[10])=0). The RTS driving level is controlled by programing the RTS(UA_MCR[1]) control bit.

Setting LEV_RTS(UA_MCR[9]) can control the RTS pin output is inverse or non-inverse from RTS(UA_MCR[1]) control bit. User can read the RTS_ST(UA_MCR[13]) bit to get real RTS pin output voltage logic status.

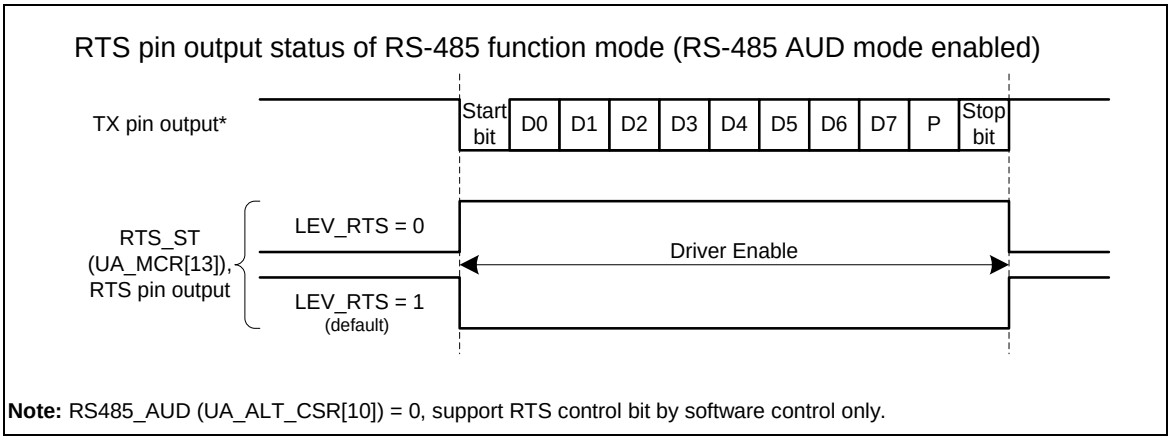
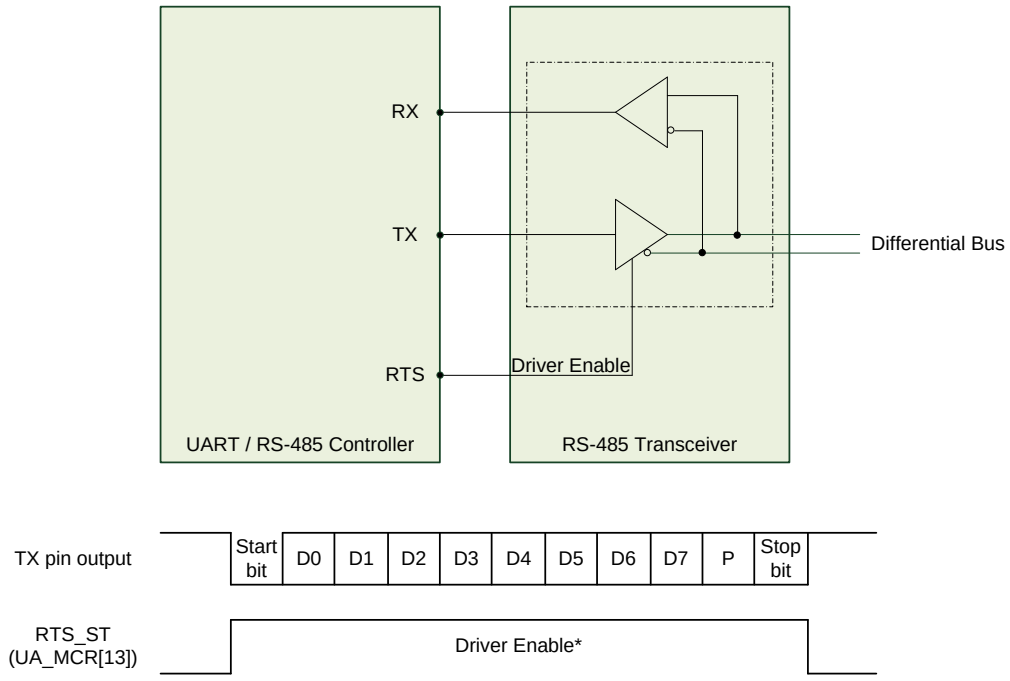


Figure 6.13-18 RS-485 RTS Driving Level with Software Control

Program Sequence Example:

1. Program FUN_SEL in UA_FUN_SEL to select RS-485 function.
2. Program the RX_DIS (UA_FCR[8]) to determine enable or disable the receiver RS-485 receiver
3. Program the RS485_NMM (UA_ALT_CSR[8]) or RS485_AAD (UA_ALT_CSR[9]) mode.
4. If the RS485_AAD (UA_ALT_CSR[9]) mode is selected, the ADDR_MATCH (UA_ALT_CSR[31:24]) is programmed for auto address match value.
5. Determine auto direction control by programming RS485_AUD (UA_ALT_CSR[10]).



Note: RS485_AUD(UA_ALT_CSR[10]) must be set to 1, and LEV_RTS(UA_MCR[9]) must be set to 0.

Figure 6.13-19 Structure of RS-485 Frame

6.13.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UART Base Address: **UART0_BA = 0x4005_0000** **UART1_BA = 0x4015_0000** **UART2_BA = 0x4015_4000**				
UA_RBR **x=0,1,2**	UARTx_BA+0x00	R	UART Receive Buffer Register	Undefined
UA_THR **x=0,1,2**	UARTx_BA+0x00	W	UART Transmit Holding Register	Undefined
UA_IER **x=0,1,2**	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000
UA_FCR **x=0,1,2**	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101
UA_LCR **x=0,1,2**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000
UA_MCR **x=0,1**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200
UA_MSR **x=0,1**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110
UA_FSR **x=0,1,2**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0x1040_4000
UA_ISR **x=0,1,2**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0000_0002
UA_TOR **x=0,1,2**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000
UA_BAUD **x=0,1,2**	UARTx_BA+0x24	R/W	UART Baud Rate Divisor Register	0x0F00_0000
UA_IRCR **x=0,1,2**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040
UA_ALT_CSR **x=0,1,2**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C
UA_FUN_SEL **x=0,1,2**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000
UA_LIN_CTL **x=0,1,2**	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000

UA_LIN_SR **x=0,1,2**	UARTx_BA+0x38	R/W	UART LIN Status Register	0x0000_0000

6.13.7 Register Description

UART Receive Buffer Register (UA_RBR)

Register	Offset	R/W	Description	Reset Value
UA_RBR **x=0,1,2**	UARTx_BA+0x00	R	UART Receive 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RBR							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**RBR**	**Receive Buffer Register (Read Only)** By reading this register, the UART will return the 8-bit data received from RX pin (LSB first).

UART Transmit Holding Register (UA_THR)

Register	Offset	R/W	Description	Reset Value
UA_THR **x=0,1,2**	UARTx_BA+0x00	W	UART Transmit Holding 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
THR							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**THR**	**Transmit Holding Register** 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 TX pin.

UART Interrupt Enable Register (UA_IER)

Register	Offset	R/W	Description	Reset Value
UA_IER x=0,1,2	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							
15	14	13	12	11	10	9	8
DMA_RX_EN	DMA_TX_EN	AUTO_CTS_EN	AUTO_RTS_EN	TIME_OUT_EN	Reserved		LIN_IEN
7	6	5	4	3	2	1	0
Reserved	WAKE_EN	BUF_ERR_IEN	TOUT_IEN	MODEM_IEN	RLS_IEN	THRE_IEN	RDA_IEN

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**DMA_RX_EN**	**RX DMA Enable Bit (Not Available In UART2 Channel)** This bit can enable or disable RX DMA service. 0 = RX DMA Disabled. 1 = RX DMA Enabled.
[14]	**DMA_TX_EN**	**TX DMA Enable Bit (Not Available In UART2 Channel)** This bit can enable or disable TX DMA service. 0 = TX DMA Disabled. 1 = TX DMA Enabled.
[13]	**AUTO_CTS_EN**	**CTS Auto Flow Control Enable Bit (Not Available In UART2 Channel)** 0 = CTS auto flow control Disabled. 1 = CTS auto flow control Enabled. When CTS auto-flow is enabled, the UART will send data to external device when CTS input assert (UART will not send data to device until CTS is asserted).
[12]	**AUTO_RTS_EN**	**RTS Auto Flow Control Enable Bit (Not Available In UART2 Channel)** 0 = RTS auto flow control Disabled. 1 = RTS auto flow control Enabled. When RTS auto-flow is enabled, if the number of bytes in the RX FIFO equals the RTS_TRI_LEV (UA_FCR [19:16]), the UART will de-assert RTS signal.
[11]	**TIME_OUT_EN**	**Time-Out Counter Enable Bit** 0 = Time-out counter Disabled. 1 = Time-out counter Enabled.
[10:9]	**Reserved**	Reserved.
[8]	**LIN_IEN**	**LIN Bus Interrupt Enable Bit**

		0 = Lin bus interrupt Disabled. 1 = Lin bus interrupt Enabled. **Note:** This field is used for LIN function mode.
[7]	**Reserved**	Reserved.
[6]	**WAKE_EN**	**UART Wake-Up Function Enable Bit (Not Available In UART2 Channel)** 0 = UART wake-up function Disabled. 1 = UART wake-up function Enabled, when the chip is in Power-down mode, an external CTS change will wake-up chip from Power-down mode.
[5]	**BUF_ERR_IEN**	**Buffer Error Interrupt Enable Bit** 0 = BUF_ERR_INT Masked off. 1 = BUF_ERR_INT Enabled.
[4]	**TOUT_IEN**	**RX Time-Out Interrupt Enable Bit** 0 = TOUT_INT Masked off. 1 = TOUT_INT Enabled.
[3]	**MODEM_IEN**	**Modem Status Interrupt Enable Bit (Not Available In UART2 Channel)** 0 = MODEM_INT Masked off. 1 = MODEM_INT Enabled..
[2]	**RLS_IEN**	**Receive Line Status Interrupt Enable Bit** 0 = RLS_INT Masked off. 1 = RLS_INT Enabled.
[1]	**THRE_IEN**	**Transmit Holding Register Empty Interrupt Enable Bit** 0 = THRE_INT Masked off. 1 = THRE_INT Enabled.
[0]	**RDA_IEN**	**Receive Data Available Interrupt Enable Bit** 0 = RDA_INT Masked off. 1 = RDA_INT Enabled.

UART FIFO Control Register (UA_FCR)

Register	Offset	R/W	Description	Reset Value
UA_FCR x=0,1,2	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				RTS_TRI_LEV			
15	14	13	12	11	10	9	8
Reserved							RX_DIS
7	6	5	4	3	2	1	0
RFITL				Reserved	TFR	RFR	Reserved

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**RTS_TRI_LEV**	**RTS Trigger Level For Auto-Flow Control Use (Not Available In UART2 Channel)** 0000 = RTS Trigger Level is 1 byte. 0001 = RTS Trigger Level is 4 bytes. 0010 = RTS Trigger Level is 8 bytes. 0011 = RTS Trigger Level is 14 bytes. 0100 = RTS Trigger Level is 30/14 bytes (High Speed/Normal Speed). 0101 = RTS Trigger Level is 46/14 bytes (High Speed/Normal Speed). 0110 = RTS Trigger Level is 62/14 bytes (High Speed/Normal Speed). Other = Reserved. **Note:** This field is used for RTS auto-flow control.
[15:9]	**Reserved**	Reserved.
[8]	**RX_DIS**	**Receiver Disable Bit** The receiver is disabled or not (set 1 to disable receiver) 0 = Receiver Enabled. 1 = Receiver Disabled. **Note:** This field is used for RS-485 Normal Multi-drop mode. It should be programmed before UA_ALT_CSR [RS-485_NMM] is programmed.
[7:4]	**RFITL**	**RX FIFO Interrupt (INT_RDA) Trigger Level** When the number of bytes in the receive FIFO equals the RFITL, the RDA_IF will be set (if UA_IER [RDA_IEN] 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. 0100 = RX FIFO Interrupt Trigger Level is 30/14 bytes (High Speed/Normal Speed). 0101 = RX FIFO Interrupt Trigger Level is 46/14 bytes (High Speed/Normal Speed).

		0110 = RX FIFO Interrupt Trigger Level is 62/14 bytes (High Speed/Normal Speed). Other = Reserved.
[3]	**Reserved**	Reserved.
[2]	**TFR**	**TX Field Software Reset** When TFR is set, all the byte in the transmit FIFO and TX internal state machine are cleared. 0 = No effect. 1 = Reset the TX internal state machine and pointers. **Note:** This bit will automatically clear at least 3 UART peripherial clock cycles.
[1]	**RFR**	**RX Field Software Reset** When RFR is set, all the byte in the receiver FIFO and RX internal state machine are cleared. 0 = No effect. 1 = Reset the RX internal state machine and pointers. **Note:** This bit will automatically clear at least 3 UART peripherial clock cycles.
[0]	**Reserved**	Reserved.

UART Line Control Register (UA_LCR)

Register	Offset	R/W	Description	Reset Value
UA_LCR **x=0,1,2**	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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**BCB**	**SPE**	**EPE**	**PBE**	**NSB**	**WLS**	

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**BCB**	**Break Control Bit** 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 and has no effect on the transmitter logic.
[5]	**SPE**	**Stick Parity Enable Bit** 0 = Stick parity Disabled. 1 = If PBE (UA_LCR[3]) and EBE (UA_LCR[4]) are logic 1, the parity bit is transmitted and checked as logic 0. If PBE (UA_LCR[3]) is 1 and EBE (UA_LCR[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. This bit has effect only when PBE (UA_LCR[3]) is set.
[3]	**PBE**	**Parity Bit Enable Bit** 0 = No parity bit. 1 = 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-bti word length, 2 "STOP bit" is generated in the transmitted data.
[1:0]	**WLS**	**Word Length Selection** 00 = Word length is 5-bit. 01 = Word length is 6-bit. 10 = Word length is 7-bit. 11 = Word length is 8-bit.

UART MODEM Control Register (UA_MCR) (Not Available in UART2 Channel)

Register	Offset	R/W	Description	Reset Value
UA_MCR **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		**RTS_ST**	**Reserved**			**LEV_RTS**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**RTS**	**Reserved**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**RTS_ST**	**RTS Pin State (Read Only) (Not Available In UART2 Channel)** This bit mirror from RTS pin output of voltage logic status. 0 = RTS pin output is low level voltage logic state. 1 = RTS pin output is high level voltage logic state.
[12:10]	**Reserved**	Reserved.
[9]	**LEV_RTS**	**RTS Pin Active Level (Not Available In UART2 Channel)** This bit defines the active level state of RTS pin output. 0 = RTS pin output is high level active. 1 = RTS pin output is low level active. **Note1:** Refer to Figure 6.13-6 and Figure 6.13-7 for UART function mode. **Note2:** Refer to Figure 6.13-17 And Figure 6.13-18 for RS-485 function mode.
[8:2]	**Reserved**	Reserved.
[1]	**RTS**	**RTS (Request-To-Send) Signal Control (Not Available In UART2 Channel)** This bit is direct control internal RTS signal active or not, and then drive the RTS pin output with LEV_RTS bit configuration. 0 = RTS signal is active. 1 = RTS signal is inactive. **Note1:** This RTS signal control bit is not effective when RTS auto-flow control is enabled in UART function mode. **Note2:** This RTS signal control bit is not effective when RS-485 auto direction mode (AUD) is enabled in RS-485 function mode.
[0]	**Reserved**	Reserved.

UART Modem Status Register (UA_MSR) (Not Available in UART2 Channel)

Register	Offset	R/W	Description	Reset Value
UA_MSR **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							**LEV_CTS**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**CTS_ST**	**Reserved**			**DCTSF**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**LEV_CTS**	**CTS Pin Active Level** This bit defines the active level state of CTS pin input. 0 = CTS pin input is high level active. 1 = CTS pin input is low level active. **Note:** Refer to Figure 6.13-5 for more information
[7:5]	**Reserved**	Reserved.
[4]	**CTS_ST**	**CTS Pin Status (Read Only) (Not Available In UART2 Channel)** This bit mirror from CTS pin input of voltage logic status. 0 = CTS pin input is low level voltage logic state. 1 = CTS pin input is high level voltage logic state. **Note:** This bit echoes when UART Controller peripheral clock is enabled, and CTS multi-function port is selected
[3:1]	**Reserved**	Reserved.
[0]	**DCTSF**	**Detect CTS State Change Flag (Read Only) (Not Available In UART2 Channel)** This bit is set whenever CTS input has change state, and it will generate Modem interrupt to CPU when MODEM_IEN (UA_IER [3]) is set to 1. 0 = CTS input has not change state. 1 = CTS input has change state. **Note:** This bit is read only, but can be cleared by writing “1” to it.

UART FIFO Status Register (UA_FSR)

Register	Offset	R/W	Description	Reset Value
UA_FSR x=0,1,2	UARTx_BA+0x18	R/W	UART FIFO Status Register	0x1040_4000

31	30	29	28	27	26	25	24
Reserved			TE_FLAG	Reserved			TX_OVER_IF
23	22	21	20	19	18	17	16
TX_FULL	TX_EMPTY	TX_POINTER					
15	14	13	12	11	10	9	8
RX_FULL	RX_EMPTY	RX_POINTER					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	RS485_ADD_DETF	Reserved		RX_OVER_IF

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**TE_FLAG**	**Transmitter Empty Flag (Read Only)** This bit is set by hardware when TX FIFO (UA_THR) is empty and the STOP bit of the last byte has been transmitted. 0 = TX FIFO is not empty. 1 = TX FIFO is empty. **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]	**TX_OVER_IF**	**TX Overflow Error Interrupt Flag (Read Only)** If TX FIFO (UA_THR) is full, an additional write to UA_THR will cause this bit to logic 1. 0 = TX FIFO is not overflow. 1 = TX FIFO is overflow. **Note:** This bit is read only, but can be cleared by writing “1” to it.
[23]	**TX_FULL**	**Transmitter FIFO Full (Read Only)** This bit indicates TX FIFO full or not. 0 = TX FIFO is not full. 1 = TX FIFO is full. This bit is set when the number of usage in TX FIFO Buffer is equal to 64/16/16(UART0/UART1/UART2), otherwise is cleared by hardware.
[22]	**TX_EMPTY**	**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 THR (TX FIFO not

		empty).
[21:16]	**TX_POINTER**	**TX FIFO Pointer (Read Only)** This field indicates the TX FIFO Buffer Pointer. When CPU writes one byte into UA_THR, then TX_POINTER increases one. When one byte of TX FIFO is transferred to Transmitter Shift Register, then TX_POINTER decreases one. The Maximum value shown in TX_POINTER is 63/15/15 (UART0/UART1/UART2). When the using level of TX FIFO Buffer equal to 64/16/16, the TX_FULL bit is set to 1 and TX_POINTER will show 0. As one byte of TX FIFO is transferred to Transmitter Shift Register, the TX_FULL bit is cleared to 0 and TX_POINTER will show 63/15/15 (UART0/UART1/UART2).
[15]	**RX_FULL**	**Receiver FIFO Full (Read Only)** This bit initiates RX FIFO is 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 64/16/16(UART0/UART1/UART2), otherwise is cleared by hardware.
[14]	**RX_EMPTY**	**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]	**RX_POINTER**	**RX FIFO Pointer (Read Only)** This field indicates the RX FIFO Buffer Pointer. When UART receives one byte from external device, then RX_POINTER increases one. When one byte of RX FIFO is read by CPU, then RX_POINTER decreases one. The Maximum value shown in RX_POINTER is 63/15/15 (UART0/UART1/UART2). When the using level of RX FIFO Buffer equal to 64/16/16, the RX_FULL bit is set to 1 and RX_POINTER will show 0. As one byte of RX FIFO is read by CPU, the RX_FULL bit is cleared to 0 and RX_POINTER will show 63/15/15 (UART0/UART1/UART2).
[7]	**Reserved**	Reserved.
[6]	**BIF**	**Break Interrupt Flag (Read Only)** 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) and is reset whenever the CPU writes 1 to this bit. 0 = No Break interrupt is generated. 1 = Break interrupt is generated. **Note:** This bit is read only, but can be cleared by writing “1” to it.
[5]	**FEF**	**Framing Error Flag (Read Only)** 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), and is reset whenever the CPU writes 1 to this bit. 0 = No framing error is generated. 1 = Framing error is generated. **Note:** This bit is read only, but can be cleared by writing “1” to it.
[4]	**PEF**	**Parity Error Flag (Read Only)** This bit is set to logic 1 whenever the received character does not have a valid “parity bit”, and is reset whenever the CPU writes 1 to this bit. 0 = No parity error is generated. 1 = Parity error is generated. **Note:** This bit is read only, but can be cleared by writing “1” to it.

[3]	**RS485_ADD_DETF**	**RS-485 Address Byte Detection Flag (Read Only)** 0 = Receiver detects a data that is not an address bit (bit 9 ='1'). 1 = Receiver detects a data that is an address bit (bit 9 ='1'). **Note1:** This field is used for RS-485 function mode and RS485_ADD_EN (UA_ALT_CSR[15]) is set to 1 to enable Address detection mode. **Note2:** This bit is read only, but can be cleared by writing '1' to it.
[2:1]	**Reserved**	Reserved.
[0]	**RX_OVER_IF**	**RX Overflow Error IF (Read Only)** This bit is set when RX FIFO overflow. If the number of bytes of received data is greater than RX_FIFO (UA_RBR) size, 64/16/16 bytes of UART0/UART1/UART2, this bit will be set. 0 = RX FIFO is not overflow. 1 = RX FIFO is overflow. **Note:** This bit is read only, but can be cleared by writing "1" to it.

UART Interrupt Status Control Register (UA_ISR)

Register	Offset	R/W	Description	Reset Value
UA_ISR x=0,1,2	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0000_0002

31	30	29	28	27	26	25	24
Reserved		HW_BUF_ERR_INT	HW_TOUT_INT	HW_MODEM_INT	HW_RLS_INT	Reserved	
23	22	21	20	19	18	17	16
Reserved		HW_BUF_ERR_IF	HW_TOUT_IF	HW_MODEM_IF	HW_RLS_IF	Reserved	
15	14	13	12	11	10	9	8
LIN_INT	Reserved	BUF_ERR_INT	TOUT_INT	MODEM_INT	RLS_INT	THRE_INT	RDA_INT
7	6	5	4	3	2	1	0
LIN_IF	Reserved	BUF_ERR_IF	TOUT_IF	MODEM_IF	RLS_IF	THRE_IF	RDA_IF

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**HW_BUF_ERR_INT**	**In DMA Mode, Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUF_ERR_IEN (UA_IER[5]) and HW_BUF_ERR_IF (UA_ISR[5])are both set to 1. 0 = No buffer error interrupt is generated in DMA mode. 1 = Buffer error interrupt is generated in DMA mode.
[28]	**HW_TOUT_INT**	**In DMA Mode, Time-Out Interrupt Indicator (Read Only)** This bit is set if TOUT_IEN (UA_IER[4])and HW_TOUT_IF(UA_ISR[20]) are both set to 1. 0 = No Tout interrupt is generated in DMA mode. 1 = Tout interrupt is generated in DMA mode.
[27]	**HW_MODEM_INT**	**In DMA Mode, MODEM Status Interrupt Indicator (Read Only) (Not Available In UART2 Channel)** This bit is set if MODEM_IEN(UA_IER[3]) and HW_MODEM_IF(UA_ ISR[3]) are both set to 1. 0 = No Modem interrupt is generated in DMA mode. 1 = Modem interrupt is generated in DMA mode.
[26]	**HW_RLS_INT**	**In DMA Mode, Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLS_IEN (UA_IER[2])and HW_RLS_IF(UA_ISR[18]) are both set to 1. 0 = No RLS interrupt is generated in DMA mode. 1 = RLS interrupt is generated in DMA mode.
[25:22]	**Reserved**	Reserved.
[21]	**HW_BUF_ERR_IF**	**In DMA Mode, Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX or RX FIFO overflows (TX_OVER_IF (UA__FSR[24]) or RX_OVER_IF (UA_FSR[0]) is set). When BUF_ERR_IF (UA_ISR[5]) is set, the transfer maybe is not correct. If BUF_ERR_IEN (UA_IER [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 when both TX_OVER_IF (UA_FSR[24]]) and RX_OVER_IF (UA_FSR[0]) are cleared.
[20]	**HW_TOUT_IF**	**In DMA Mode, 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 (UA_TOR[7:0]). If TOUT_IEN (UA_IER [4]) is enabled, the Tout interrupt will be generated. 0 = No Time-out interrupt flag is generated. 1 = Time-out interrupt flag is generated. **Note:** This bit is read only and user can read UA_RBR (RX is in active) to clear it.
[19]	**HW_MODEM_IF**	**In DMA Mode, MODEM Interrupt Flag (Read Only) (Not Available In UART2 Channel)** This bit is set when the CTS pin has state change (DCTSF (US_MSR[0] =1)). If MODEM_IEN (UA_IER [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 the bit DCTSF(US_MSR[0]) is cleared by writing 1 on DCTSF (US_MSR[0]).
[18]	**HW_RLS_IF**	**In DMA 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 (UA_FSR[6]), FEF (UA_FSR[5]) and PEF (UA_FSR[4]) is set). If RLS_IEN (UA_IER [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 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(UA_FSR[6]) , FEF(UA_FSR[5]) and PEF(UA_FSR[4]) are cleared. **Note3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF(UA_FSR[6]) , FEF(UA_FSR[5]) and PEF(UA_FSR[4]) and RS485_ADD_DETF (UA_FSR[3]) are cleared.
[17:16]	**Reserved**	Reserved.
[15]	**LIN_INT**	**LIN Bus Interrupt Indicator (Read Only)** This bit is set if LIN_IEN (UA_IER[8]) and LIN _IF(UA_ISR[7]) are both set to 1. 0 = No LIN Bus interrupt is generated. 1 = The LIN Bus interrupt is generated.
[14]	**Reserved**	Reserved.
[13]	**BUF_ERR_INT**	**Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUF_ERR_IEN(UA_IER[5] and BUF_ERR_IF(UA_ISR[5]) are both set to 1. 0 = No buffer error interrupt is generated. 1 = Buffer error interrupt is generated.
[12]	**TOUT_INT**	**Time-Out Interrupt Indicator (Read Only)** This bit is set if TOUT_IEN(UA_IER[4]) and TOUT_IF(UA_ISR[4]) are both set to 1. 0 = No Tout interrupt is generated. 1 = Tout interrupt is generated.
[11]	**MODEM_INT**	**MODEM Status Interrupt Indicator (Read Only) (Not Available In UART2 Channel)**

		This bit is set if MODEM_IEN(UA_IER[3] and MODEM_IF(UA_ISR[4]) are both set to 1 0 = No Modem interrupt is generated. 1 = Modem interrupt is generated.
[10]	**RLS_INT**	**Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLS_IEN (UA_IER[2]) and RLS_IF(UA_ISR[2]) are both set to 1. 0 = No RLS interrupt is generated. 1 = RLS interrupt is generated.
[9]	**THRE_INT**	**Transmit Holding Register Empty Interrupt Indicator (Read Only)** This bit is set if THRE_IEN (UA_IER[1])and THRE_IF(UA_SR[1]) are both set to 1. 0 = No THRE interrupt is generated. 1 = THRE interrupt is generated.
[8]	**RDA_INT**	**Receive Data Available Interrupt Indicator (Read Only)** This bit is set if RDA_IEN (UA_IER[0]) and RDA_IF (UA_ISR[0]) are both set to 1. 0 = No RDA interrupt is generated. 1 = RDA interrupt is generated.
[7]	**LIN_ IF**	**LIN Bus Flag (Read Only)** This bit is set when LIN slave header detect (LINS_HDET_F (UA_LIN_SR[0] =1)), LIN break detect (LIN_BKDET_F(UA_LIN_SR[9]=1)), bit error detect (BIT_ERR_F(UA_LIN_SR[9]=1), LIN slave ID parity error (LINS_IDPERR_F(UA_LIN_SR[2] = 1) or LIN slave header error detect (LINS_HERR_F (UA_LIN_SR[1])). If LIN_ IEN (UA_IER [8]) is enabled the LIN interrupt will be generated. 0 = None of LINS_HDET_F, LIN_BKDET_F, BIT_ERR_F, LINS_IDPERR_F and LINS_HERR_F is generated. 1 = At least one of LINS_HDET_F, LIN_BKDET_F, BIT_ERR_F, LINS_IDPERR_F and LINS_HERR_F is generated. **Note:** This bit is read only. This bit is cleared when LINS_HDET_F(UA_LIN_SR[0]), LIN_BKDET_F(UA_LIN_SR[9]), BIT_ERR_F(UA_LIN_SR[9]), LINS_IDPENR_F (UA_LIN_SR[2]) and LINS_HERR_F(UA_LIN_SR[1]) all are cleared.
[6]	**Reserved**	Reserved.
[5]	**BUF_ERR_IF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX FIFO or RX FIFO overflows (TX_OVER_IF (UA_FSR[24]) or RX_OVER_IF (UA_FSR[0]) is set). When BUF_ERR_IF (UA_ISR[5])is set, the transfer is not correct. If BUF_ERR_IEN (UA_IER [8]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated. 1 = Buffer error interrupt flag is generated.0 = No buffer error interrupt flag is generated. 1 = Buffer error interrupt flag is generated. **Note:** This bit is read only and reset to 0 when all bits of TX_OVER_IF(UA_FSR[24]) and RX_OVER_IF(UA_FSR[0]) are cleared.
[4]	**TOUT_IF**	**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. If TOUT_IEN (UA_IER [4]) is enabled, the Tout interrupt will be generated. 0 = No Time-out interrupt flag is generated. 1 = Time-out interrupt flag is generated. **Note:** This bit is read only and user can read UA_RBR (RX is in active) to clear it
[3]	**MODEM_IF**	**MODEM Interrupt Flag (Read Only) (Not Available In UART2 Channel)** This bit is set when the CTS pin has state change (DCTSF (UA_MSR[0]) = 1). If MODEM_IEN (UA_IER [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 DCTSF is cleared by a write 1 on DCTSF(UA_MSR[0]).
[2]	**RLS_IF**	**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(UA_FSR[6]), FEF(UA_FSR[5]) and PEF(UA_FSR[4]), is set). If RLS_IEN (UA_IER [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 UA_FSR[RS485_ADD_DETF] is also set. **Note2:** This bit is read only and reset to 0 when all bits of BIF(UA_FSR[6]), FEF(UA_FSR[5]) and PEF(UA_FSR[4]) are cleared. **Note3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF(UA_FSR[6]) , FEF(UA_FSR[5]) and PEF(UA_FSR[4]) and RS485_ADD_DETF (UA_FSR[3]) are cleared.
[1]	**THRE_IF**	**Transmit Holding Register Empty Interrupt Flag (Read Only)** This bit is set when the last data of TX FIFO is transferred to Transmitter Shift Register. If THRE_IEN (UA_IER[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 THR (TX FIFO not empty).
[0]	**RDA_IF**	**Receive Data Available Interrupt Flag (Read Only)** When the number of bytes in the RX FIFO equals the RFITL then the RDA_IF(UA_ISR[0]) will be set. If RDA_IEN (UA_IER [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(UA_FCR[7:4]).

UART Time-out Register (UA_TOR)

Register	Offset	R/W	Description	Reset Value
UA_TOR x=0,1,2	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.
[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. Once the content of time-out counter is equal to that of time-out interrupt comparator (TOIC (UA_TOR[7:0])), a receiver time-out interrupt (INT_TOUT) is generated if TOUT_IEN (UA_IER [4]) enabled. A new incoming data word or RX FIFO empty will clear TOUT_INT(UA_IER[9]). In order to avoid receiver time-out interrupt generation immediately during one character is being received, TOIC (UA_TOR[7:0]) value should be set between 40 and 255. So, for example, if TOIC (UA_TOR[7:0]) 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 (UA_BAUD)

Register	Offset	R/W	Description	Reset Value
UA_BAUD **x=0,1,2**	UARTx_BA+0x24	R/W	UART Baud Rate Divisor Register	0x0F00_0000

31	30	29	28	27	26	25	24
Reserved		DIV_X_EN	DIV_X_ONE	DIVIDER_X			
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]	**DIV_X_EN**	**Divider X Enable Bit** The BRD = Baud Rate Divider, and the baud rate equation is Baud Rate = Clock / [M * (BRD + 2)]; The default value of M is 16. 0 = Divider X Disabled (the equation of M = 16). 1 = Divider X Enabled (the equation of M = X+1, but DIVIDER_X [27:24] must >= 8). Refer to Table 6.13-2 for more information. **Note:** In IrDA mode, this bit must disable.
[28]	**DIV_X_ONE**	**Divider X Equal To 1** 0 = Divider M = X (the equation of M = X+1, but DIVIDER_X[27:24] must >= 8). 1 = Divider M = 1 (the equation of M = 1, but BRD [15:0] must >= 3). Refer to Table 6.13-2 for more information.
[27:24]	**DIVIDER_X**	**Divider X** The baud rate divider M = X+1.
[23:16]	**Reserved**	Reserved.
[15:0]	**BRD**	**Baud Rate Divider** The field indicates the baud rate divider

UART IrDA Control Register (IRCR)

Register	Offset	R/W	Description	Reset Value
UA_IRCR **x=0,1,2**	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	**INV_RX**	**INV_TX**	**Reserved**			**TX_SELECT**	**Reserved**

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**INV_RX**	**IrDA Inverse Receive Input Signal Control** 0 = None inverse receiving input signal. 1 = Inverse receiving input signal.
[5]	**INV_TX**	**IrDA Inverse Transmitting Output Signal Control** 0 = None inverse transmitting signal.. 1 = Inverse transmitting output signal.
[4:2]	**Reserved**	Reserved.
[1]	**TX_SELECT**	**TX_SELECT** 0 = IrDA Transmitter Disabled and Receiver Enabled. 1 = IrDA Transmitter Enabled and Receiver Disabled.
[0]	**Reserved**	Reserved.

Note: In IrDA mode, the UA_BAUD (UA_BAUD [29]) register must be disabled (the baud equation must be Clock / 16 * (BRD)

UART Alternate Control/Status Register (UA_ALT_CSR)

Register	Offset	R/W	Description	Reset Value
UA_ALT_CSR x=0,1,2	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C

31	30	29	28	27	26	25	24
ADDR_MATCH							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RS485_ADD_EN	Reserved				RS485_AUD	RS485_AAD	RS485_NMM
7	6	5	4	3	2	1	0
LIN_TX_EN	LIN_RX_EN	Reserved		LIN_BKFL			

Bits	Description	
[31:24]	**ADDR_MATCH**	**Address Match Value Register** This field contains the RS-485 address match values. **Note:** This field is used for RS-485 auto address detection mode.
[23:16]	**Reserved**	Reserved.
[15]	**RS485_ADD_EN**	**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]	**RS485_AUD**	**RS-485 Auto Direction Mode (AUD)** 0 = RS-485 Auto Direction Operation mode (AUO) Disabled. 1 = RS-485 Auto Direction Operation mode (AUO) Enabled. **Note:** It can be active with RS-485_AAD or RS-485_NMM operation mode.
[9]	**RS485_AAD**	**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]	**RS485_NMM**	**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]	**LIN_TX_EN**	**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]	**LIN_RX_EN**	**LIN RX Enable Bit** 0 = LIN RX mode Disabled. 1 = LIN RX mode Enabled.
[5:4]	**Reserved**	Reserved.
[3:0]	**LIN_BKFL**	**UART LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note1:** This break field length is UA_LIN_BKFL + 1 **Note2:** According to LIN spec, the reset value is 0xC (break field length = 13).

UART Function Select Register (UA_FUN_SEL)

Register	Offset	R/W	Description	Reset Value
UA_FUN_SEL x=0,1,2	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000

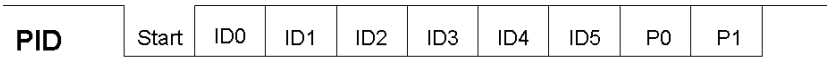
31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**FUN_SEL**	

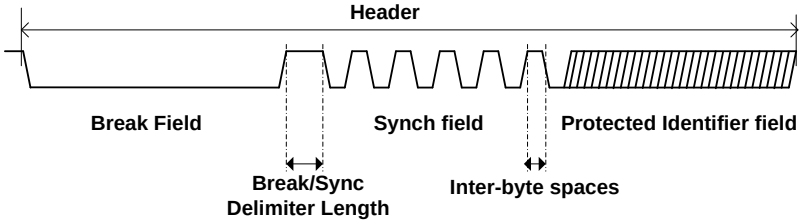
Bits	Description	
[31:2]	**Reserved**	Reserved.
[1:0]	**FUN_SEL**	**Function Select Enable Bit** 00 = UART function Enabled. 01 = LIN function Enabled. 10 = IrDA function Enabled. 11 = RS-485 function Enabled.

UART LIN Control Register (UA_LIN_CTL)

Register	Offset	R/W	Description	Reset Value
UA_LIN_CTL x=0,1,2	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000

31	30	29	28	27	26	25	24
LIN_PID							
23	22	21	20	19	18	17	16
LIN_HEAD_SEL		LIN_BS_LEN		LIN_BKFL			
15	14	13	12	11	10	9	8
Reserved			BIT_ERR_EN	LIN_RX_DIS	LIN_BKDET_EN	LIN_IDPEN	LIN_SHD
7	6	5	4	3	2	1	0
Reserved			LIN_MUTE_EN	LINS_DUM_EN	LINS_ARS_EN	LINS_HDET_EN	LINS_EN

Bits	Description	
[31:24]	LIN_PID	**LIN PID Register** This field contains the LIN frame ID value when in LIN function mode, the frame ID parity can be generated by software or hardware depends on LIN_IDPEN (UA_LIN_CTL[9]) = 1. If the parity generated by hardware, user fill ID0~ID5, (LIN_PID [29:24])hardware will calculate P0 (LIN_PID[30]) and P1 (LIN_PID[31]), otherwise user must filled frame ID and parity in this field.  P0 = ID0 xor ID1 xor ID2 xor ID4 P1 = ~(ID1 xor ID3 xor ID4 xor ID5) **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]	LIN_HEAD_SEL	**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 (LIN_SHD (UA_LIN_CTL [8]) = 1) or used to slave to indicates exit from mute mode condition (LIN_MUTE_EN (UA_LIN_CTL[4] = 1).
[21:20]	LIN_BS_LEN	**LIN Break/Sync Delimiter Length** 00 = The LIN break/sync delimiter length is 1 bit time. 10 = 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]	**LIN_BKFL**	**LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note1:** These registers are shadow registers of LIN_BKFL, User can read/write it by setting LIN_BKFL (UA_ALT_CSR[3:0]) or LIN_BKFL (UA_LIN_CTL[19:16]. **Note2:** This break field length is LIN_BKFL + 1. **Note3:** According to LIN spec, the reset value is 12 (break field length = 13).
[15:13]	**Reserved**	Reserved.
[12]	**BIT_ERR_EN**	**Bit Error Detect Enable Bit** 0 = Bit error detection function Disabled. 1 = Bit error detection Enabled. **Note:** In LIN function mode, when occur bit error, the BIT_ERR_F (UA_LIN_SR[9]) flag will be asserted. If the LIN_IEN (UA_IER[8]) = 1, an interrupt will be generated.
[11]	**LIN_RX_DIS**	**LIN Receiver Disable Bit** If the receiver is enabled (LIN_RX_DIS (UA_LIN_CTL[11]) = 0), all received byte data will be accepted and stored in the RX-FIFO, and if the receiver is disabled (LIN_RX_DIS (UA_LIN_CTL[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 (FUN_SEL (UA_FUN_SEL[1:0]) = 01).
[10]	**LIN_BKDET_EN**	**LIN Break Detection Enable Bit** When detect consecutive dominant greater than 11 bits, and are followed by a delimiter character, the LIN_BKDET_F (UA_LIN_SR[8]) flag is set in UA_LIN_SR register at the end of break field. If the LIN_IEN (UA_IER [8])=1, an interrupt will be generated. 0 = LIN break detection Disabled. 1 = LIN break detection Enabled.
[9]	**LIN_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 (LIN_SHD (UA_LIN_CTL[8])) = 1 and LIN_HEAD_SEL (UA_LIN_CTL[23:22]) = 10) or be used for enable LIN slave received frame ID parity checked. **Note2:** This bit is only use when the operation header transmitter is in LIN_HEAD_SEL (UA_LIN_CTL[23:22]) = 10.
[8]	**LIN_SHD**	**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 LIN_HEAD_SEL (UA_LIN_CTL[23:22]). 0 = Send LIN TX header Disabled. 1 = Send LIN TX header Enabled. **Note1:** These registers are shadow registers of LIN_SHD (UA_ALT_CSR [7]); user can read/write it by setting LIN_SHD (UA_ALT_CSR [7]) or LIN_SHD (UA_LIN_CTL [8]). **Note2:** When transmitter header field (it may be “break” or “break + sync” or “break + sync + frame ID” selected by LIN_HEAD_SEL (UA_LIN_CTL[23:22]) field) transfer operation

		finished, this bit will be cleared automatically.
[7:5]	**Reserved**	Reserved.
[4]	**LIN_MUTE_EN**	**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 (LIN slave mode).
[3]	**LINS_DUM_EN**	**LIN Slave Divider Update Method Enable Bit** 0 = UA_BAUD updated is written by software (if no automatic resynchronization update occurs at the same time). 1 = UA_BAUD is updated at the next received character. User must set the bit before checksum reception. **Note1:** This bit only valid when in LIN slave mode (LINS_EN (UA_LIN_CTL[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 6.13.5.8.4. (Slave mode with automatic resynchronization).
[2]	**LINS_ARS_EN**	**LIN Slave Automatic Resynchronization Mode Enable Bit** 0 = LIN automatic resynchronization Disabled. 1 = LIN automatic resynchronization Enabled. **Note1:** This bit only valid when in LIN slave mode (LINS_EN (UA_LIN_CTL[0]) = 1). **Note2:** When operation in Automatic Resynchronization mode, the baud rate setting must be mode2 (BAUD_M1 (UA_BAUD [29]) and BAUD_M0 (UA_BAUD [28]) must be 1). **Note3:** The control and interactions of this field are explained in 6.13.5.8.4. (Slave mode with automatic resynchronization).
[1]	**LINS_HDET_EN**	**LIN Slave Header Detection Enable Bit** 0 = LIN slave header detection Disabled. 1 = LIN slave header detection Enabled. **Note1:** This bit only valid when in LIN slave mode (LINS_EN (UA_LIN_CTL[0]) = 1). **Note2:** In LIN function mode, when detect header field (break + sync + frame ID), LINS_HDET_F (UA_LIN_SR [0]) flag will be asserted. If the LIN_IEN (UA_IER[8]) = 1, an interrupt will be generated.
[0]	**LINS_EN**	**LIN Slave Mode Enable Bit** 0 = LIN slave mode Disabled. 1 = LIN slave mode Enabled.

UART LIN Status Register (UA_LIN_SR)

Register	Offset	R/W	Description	Reset Value
UA_LIN_SR x=0,1,2	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						BIT_ERR_F	LIN_BKDET_F
7	6	5	4	3	2	1	0
Reserved				LINS_SYNC_F	LINS_IDPERR_F	LINS_HERR_F	LINS_HDET_F

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**BIT_ERR_F**	**Bit Error Detect Status Flag (Read Only)** At TX transfer state, hardware will monitoring the bus state, if the input pin (SIN) state not equals to the output pin (SOUT) state, BIT_ERR_F (UA_LIN_SR[9]) will be set. When occur bit error, if the LIN_IEN (UA_IER[8]) = 1, an interrupt will be generated. **Note1:** This bit is read only, but it can be cleared by writing 1 to it. **Note2:** This bit is only valid when enable bit error detection function (BIT_ERR_EN (UA_LIN_CTL [12]) = 1).
[8]	**LIN_BKDET_F**	**LIN Break Detection Flag (Read Only)** 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 is read only, but it can be cleared by writing 1 to it. **Note2:** This bit is only valid when LIN break detection function is enabled (LIN_BKDET_EN (UA_LIN_CTL[10]) =1).
[7:4]	**Reserved**	Reserved.
[3]	**LINS_SYNC_F**	**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 is only valid when in LIN Slave mode (LINS_EN(UA_LIN_CTL[0]) = 1). **Note2:** This bit is read only, but it can be cleared by writing 1 to it. **Note3:** When writing 1 to it, hardware will reload the initial baud rate and re-search a new frame header.

[2]	**LINS_IDPERR_F**	**LIN Slave ID Parity Error Flag (Read Only)** 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 is read only, but it can be cleared by writing “1” to it. **Note2:** This bit is only valid when in LIN slave mode (LINS_EN (UA_LIN_CTL [0])= 1) and enable LIN frame ID parity check function LIN_IDPEN (UA_LIN_CTL [9]).
[1]	**LINS_HERR_F**	**LIN Slave Header Error Flag (Read Only)** 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 is read only, but it can be cleared by writing 1 to it. **Note2:** This bit is only valid when UART is operated in LIN slave mode (LINS_EN (UA_LIN_CTL [0]) = 1) and enables LIN slave header detection function (LINS_HDET_EN (UA_LIN_CTL [1])).
[0]	**LINS_HDET_F**	**LIN Slave Header Detection Flag (Read Only)** 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 is read only, but it can be cleared by writing 1 to it. **Note2:** This bit is only valid when in LIN slave mode (LINS_EN (UA_LIN_CTL [0]) = 1) and enable LIN slave header detection function (LINS_HDET_EN (UA_LIN_CTL [1])). **Note3:** When enable ID parity check LIN_IDPEN (UA_LIN_CTL [9]), if hardware detect complete header (“break + sync + frame ID”), the LINS_HEDT_F will be set whether the frame ID correct or not.

6.14 I²C Serial Interface Controller (I²C)

6.14.1 Overview

I²C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. The I²C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously.

6.14.2 Features

The I²C bus uses two wires (I2Cn_SDA and I2Cn_SCL) to transfer information between devices connected to the bus. The main features of the I²C bus include:

- Supports up to two I²C serial interface controller
- Master/Slave mode
- Bidirectional data transfer between masters and slaves
- Multi-master bus (no central master)
- Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
- Serial clock synchronization allow devices with different bit rates to communicate via one serial bus
- Built-in a 14-bit time-out counter requesting the I²C interrupt if the I²C bus hangs up and timer-out counter overflows.
- Programmable clocks allow for versatile rate control
- Supports 7-bit addressing mode
- Supports multiple address recognition (four slave address with mask option)
- Supports Power-down wake-up function

6.14.3 Basic Configuration

The basic configurations of I²C0 are as follows:

- I²C0 pins are configured on GPA_MFP [9:8] register
- Enable I²C0 clock by setting I2C0_EN (APBCLK [8])
- Reset I²C0 controller by setting I2C0_RST(IPRSTC2 [8])

The basic configurations of I²C1 are as follows:

- I²C1 pins are configured on GPE_MFP [11:10] register
- Enable I²C1 clock by setting I2C1_EN(APBCLK [9])
- Reset I²C1 controller by setting I2C1_RST (IPRSTC2 [9])

6.14.4 Block Diagram

The basic configurations of I²C are as follows:

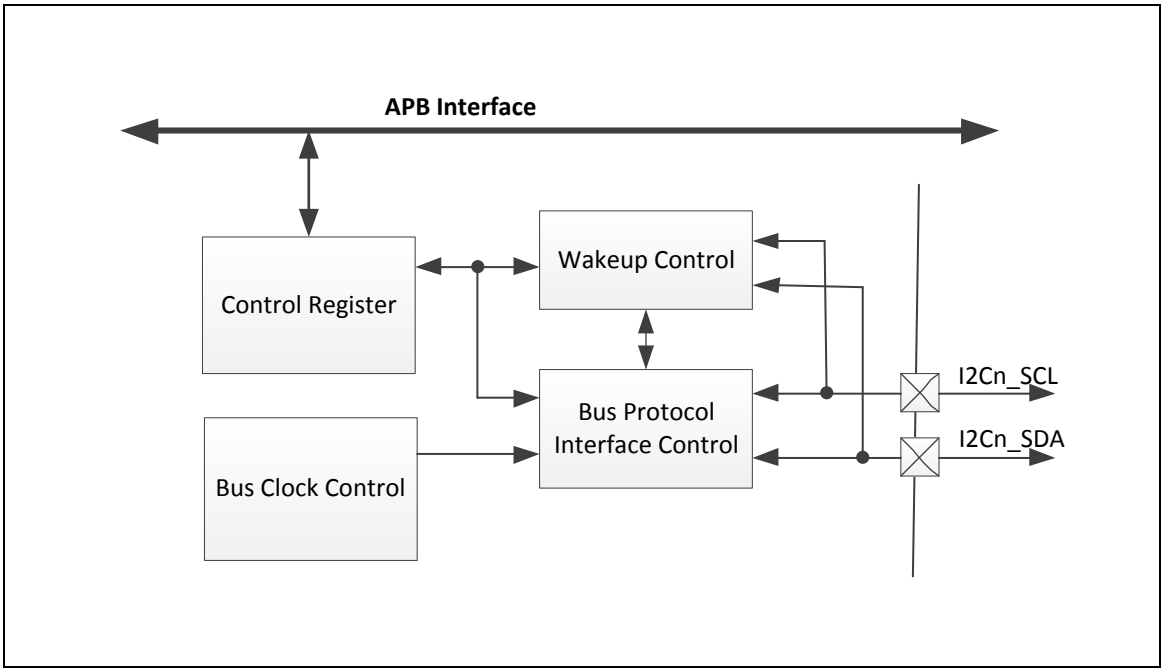


Figure 6.14-1 I²C Controller Block Diagram

6.14.5 Functional Description

On I²C bus, data is transferred between a Master and a Slave. Data bits transfer on the I2Cn_SCL and I2Cn_SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit long. There is one I2Cn_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 I2Cn_SCL; therefore, the I2Cn_SDA line may be changed only during the low period of

I2Cn_SCL and must be held stable during the high period of I2Cn_SCL. A transition on the I2Cn_SDA line while I2Cn_SCL is high is interpreted as a command (START or STOP). Please refer to the following figure for more detailed I²C bus timing.

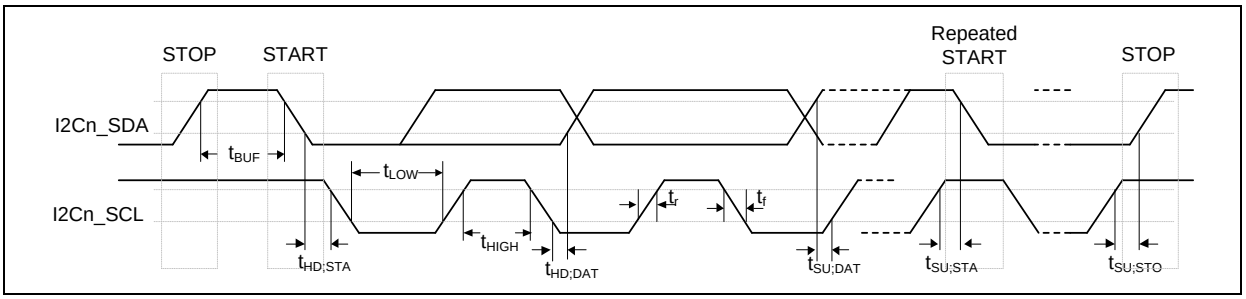


Figure 6.14-2 I²C Bus Timing

The device's on-chip I²C provides the serial interface that meets the I²C bus standard mode specification. The I²C port handles byte transfers autonomously. To enable this port, ENS1 (I2CON[6]) should be set to '1'. The I²C hardware interfaces to the I²C bus via two pins: I2Cn_SDA and I2Cn_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 as the I2Cn_SDA and I2Cn_SCL are open-drain pins.

6.14.5.1 I²C Protocol

The following figure 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

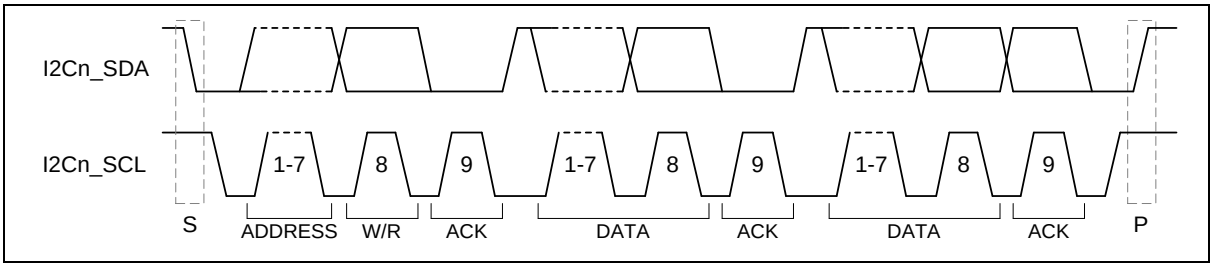


Figure 6.14-3 I²C Protocol

6.14.5.1.1 *START or Repeated START signal*

When the bus is free or idle, meaning no master device is engaging the bus (both I2Cn_SCL and I2Cn_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 I2Cn_SDA line while I2Cn_SCL is HIGH. The START signal denotes the beginning of a new data transmission.

After having sent the address byte (address and read/write bit) the master may send any number of bytes followed by a stop condition. Instead of sending the stop condition it is also allowed to send another start condition again followed by an address (and of course including a read/write bit) and more data. The start condition is called as Repeat START (Sr). This is defined recursively allowing any number of start conditions to be sent. The purpose of this is to allow combined write/read operations to one or more devices without releasing the bus and thus with the guarantee that the operation is not interrupted. The controller uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

6.14.5.1.2 *STOP signal*

The master can terminate the communication by generating a STOP signal. A STOP signal, usually referred to as the “P” bit, is defined as a LOW to HIGH transition on the I2Cn_SDA line while I2Cn_SCL is HIGH.

The following figure shows the waveform of START, Repeat START and STOP.

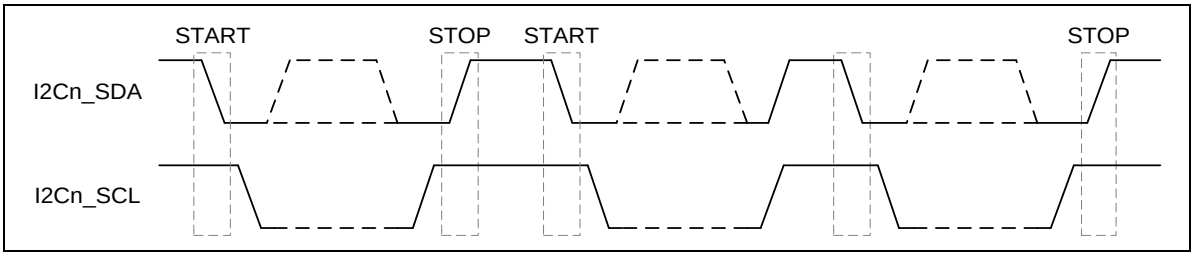


Figure 6.14-4 START and STOP Conditions

6.14.5.1.3 *Slave Address Transfer*

The first byte of data transferred by the master immediately after the START signal is the Slave address (SLA). This is a 7-bit calling address followed by a Read/Write (R/W) bit. The R/W bit signals of the slave indicate the data transfer direction. No two slaves in the system can have the same address. Only the slave with an address that matches the one transmitted by the master will respond by returning an acknowledge bit by pulling the I2Cn_SDA low at the 9th I2Cn_SCL clock cycle.

6.14.5.1.4 *Data Transfer*

When a slave receives a correct address with an R/W bit, the data will follow R/W bit specified to transfer. Each transferred byte is followed by an acknowledge bit on the 9th I2Cn_SCL clock cycle. If the slave signals a Not Acknowledge (NACK), the master can generate a STOP signal to abort the data transfer or generate a Repeated START signal and start a new transfer cycle.

If the master, as a receiving device, does Not Acknowledge (NACK) the slave, the slave releases the I2Cn_SDA line for the master to generate a STOP or Repeated START signal.

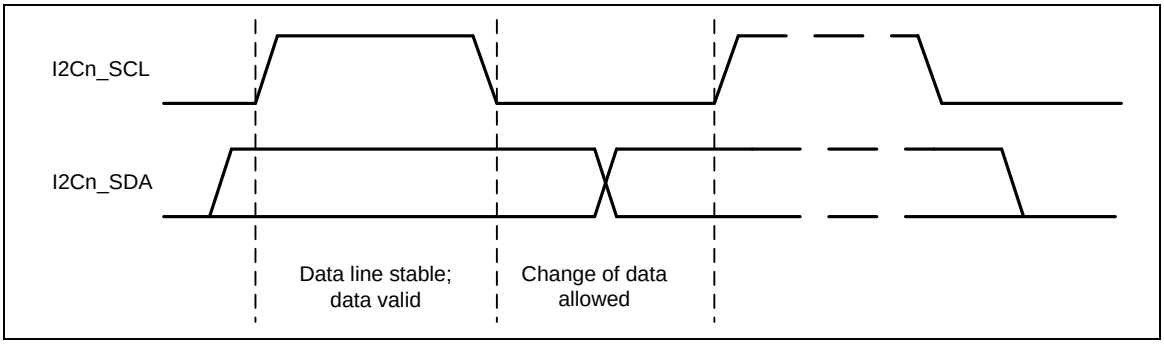


Figure 6.14-5 Bit Transfer on the I^{2}C Bus

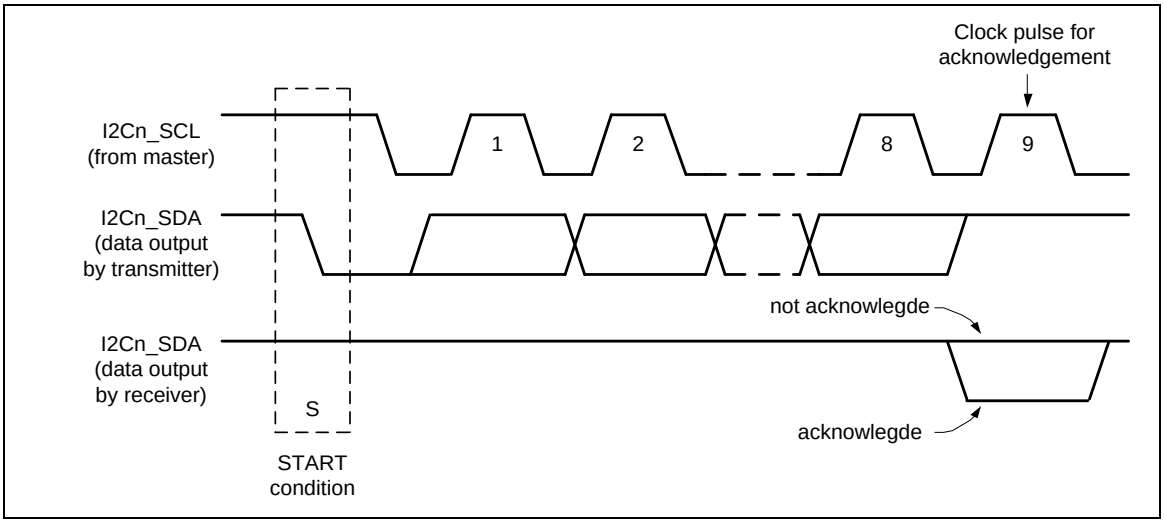


Figure 6.14-6 Acknowledge on the I^{2}C Bus

6.14.5.1.5 *Data transfer on the I^{2}C bus*

The following figure 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.

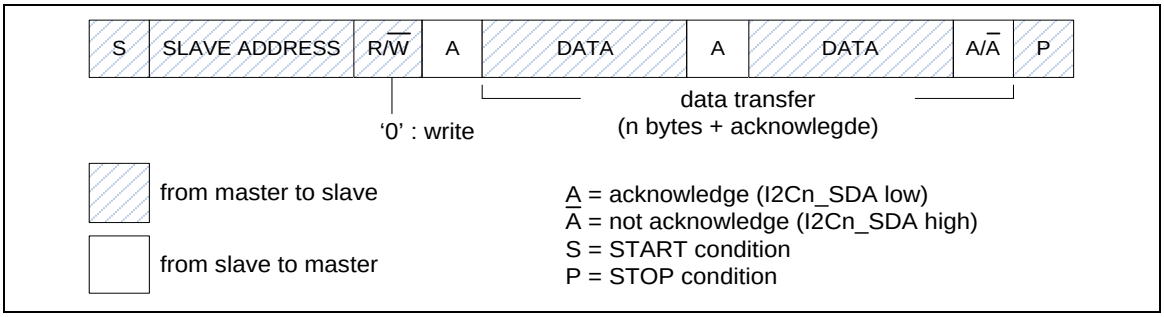


Figure 6.14-7 Master Transmits Data to Slave

The following figure 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.

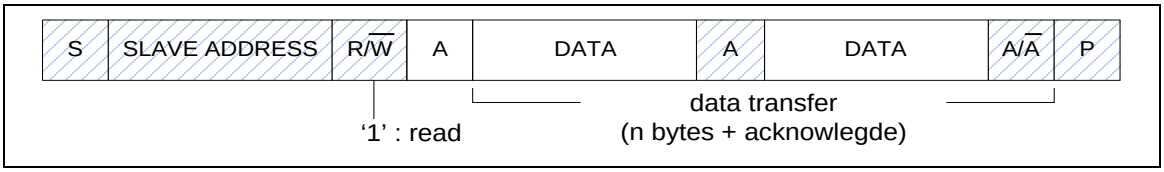


Figure 6.14-8 Master Reads Data from Slave

6.14.5.2 Operation Modes

The on-chip I^2C ports support three operation modes, Master, Slave, and General Call Mode.

In a given application, I^2C port may operate as a master or as a slave. In Slave mode, the I^2C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master(by setting the AA bit), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, hardware waits until the bus is free before entering Master mode so that a possible slave action is not be interrupted. If bus arbitration is lost in Master mode, I^2C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer.

To control the I^2C bus transfer in each mode, user needs to set I2CON, I2CDAT registers according to current status code of I2CSTATUS register. In other words, for each I^2C bus action, user needs to check current status by I2CSTATUS register, and then set I2CON, I2CDAT registers to take bus action. Finally, check the response status by I2CSTATUS.

The bits, STA(I2CON[5]), STO(I2CON[4]) and AA(I2CON[2]) are used to control the next state of the I^2C hardware after SI (I2CON[3]) flag is cleared. Upon completion of the new action, a new status code will be updated in I2CSTATUS register and the SI flag will be set. If the I^2C interrupt control bit EI (I2CON [7]) is set, appropriate action or software branch of the new status code can be performed in the Interrupt service routine.

The following figure shows the current I^2C status code is 0x08, and then set I2CDATA=SLA+W and (STA,STO,SI,AA) = (0,0,1,x) to send the address to I^2C bus. If a slave on the bus matches the address and response ACK, the I2CSTATUS will be updated by status code 0x18.

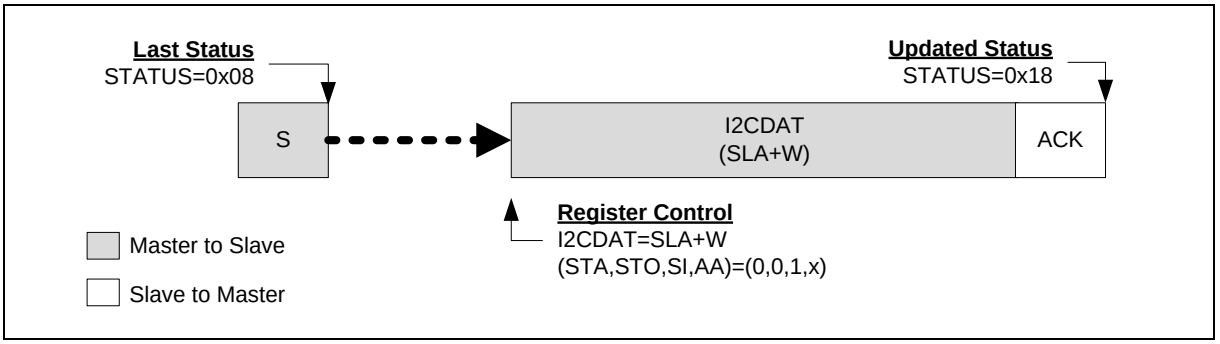


Figure 6.14-9 Control I^2C Bus according to Current I^2C Status

6.14.5.2.1 *Master Mode*

In below figures, all possible protocols for I^{2}C master are shown. User needs to follow proper path of the flow to implement required I^{2}C protocol.

In other words, user can send a START signal to bus and I^{2}C will be in Master Transmitter mode (Figure 6.14-10) or Master receiver mode (Figure 6.14-12) after START signal has been sent successfully and new status code would be 0x08. Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^{2}C protocol.

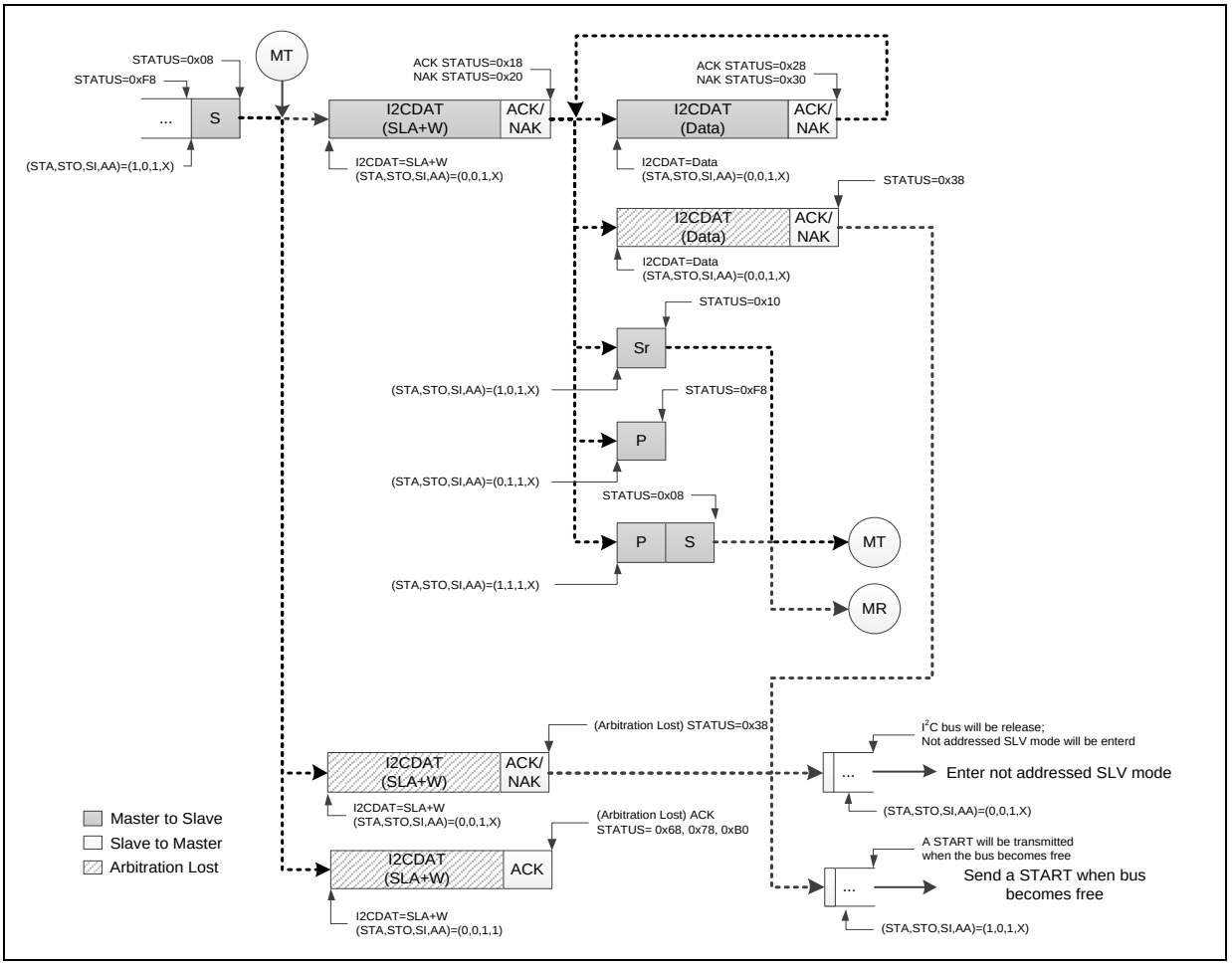


Figure 6.14-10 Master Transmitter Mode Control Flow

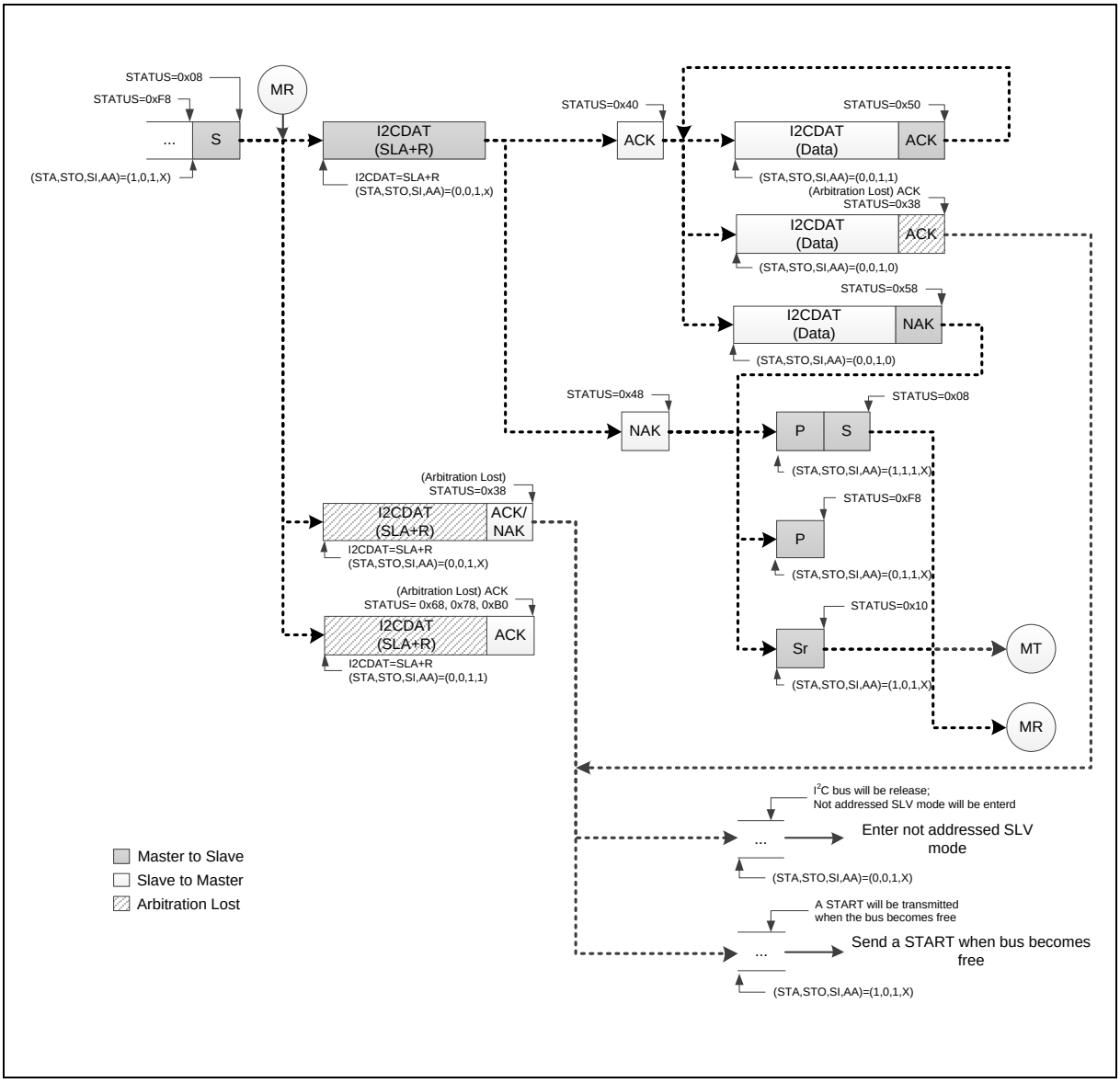


Figure 6.14-11 Master Receiver Mode Control Flow

If the I²C is in Master mode and gets arbitration lost, the status code will be 0x38. In status 0x38, user may set (STA, STO, SI, AA) = (1, 0, 1, X) to send START to re-start Master operation when bus become free. Otherwise, user may set (STA, STO, SI, AA) = (0, 0, 1, X) to release I²C bus and enter not addressed Slave mode.

6.14.5.2.2 *Slave Mode*

When reset default, I²C is not addressed and will not recognize the address on I²C bus. User can set slave address by I2CADDRx and set (STA, STO, SI, AA) = (0, 0, 1, 1) to let I²C recognize the address sent by master. Figure 6.14-12 shows all the possible flow for I²C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.14-12 to implement their own I²C protocol.

If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can

detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) after arbitration lost, the status code is 0x68. If the detected address is SLA+R (Master want to read data from Slave) after arbitration lost, the status code is 0xB0.

Note: During I²C communication, the I2Cn_SCL clock will be released when writing '1' to clear SI flag in Slave mode.

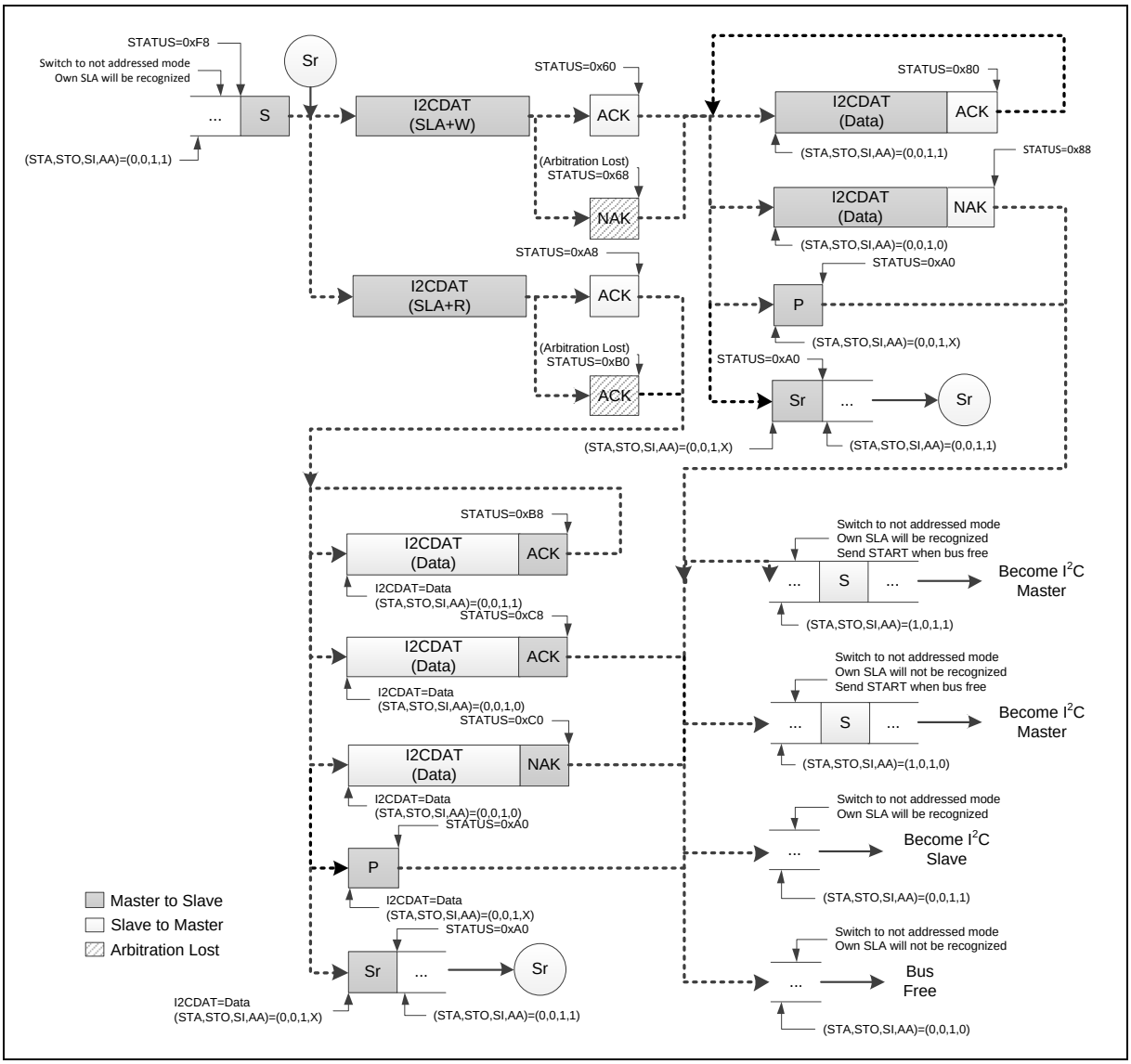


Figure 6.14-12 Save Mode Control Flow

If I²C is still receiving data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x88 as shown in the above figure when getting 0xA0 status.

If I²C is still transmitting data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0xC8 as shown in the above figure when getting 0xA0 status.

Note: After slave gets status of 0x88, 0xC8, 0xC0 and 0xA0, slave can switch to not address

mode and own SLA will not be recognized. If entering this status, slave will not receive any I²C signal or address from master. At this status, I²C should be reset to leave this status.

6.14.5.2.3 *General Call (GC) Mode*

If the GC(I2CADDRn [0]) bit is set, the I²C port hardware will respond to General Call address (0x00). User can clear GC bit to disable general call function. When the GC bit is set and the I²C in Slave mode, it can receive the general call address by 0x00 after master send general call address to I²C bus, then it will follow status of GC mode.

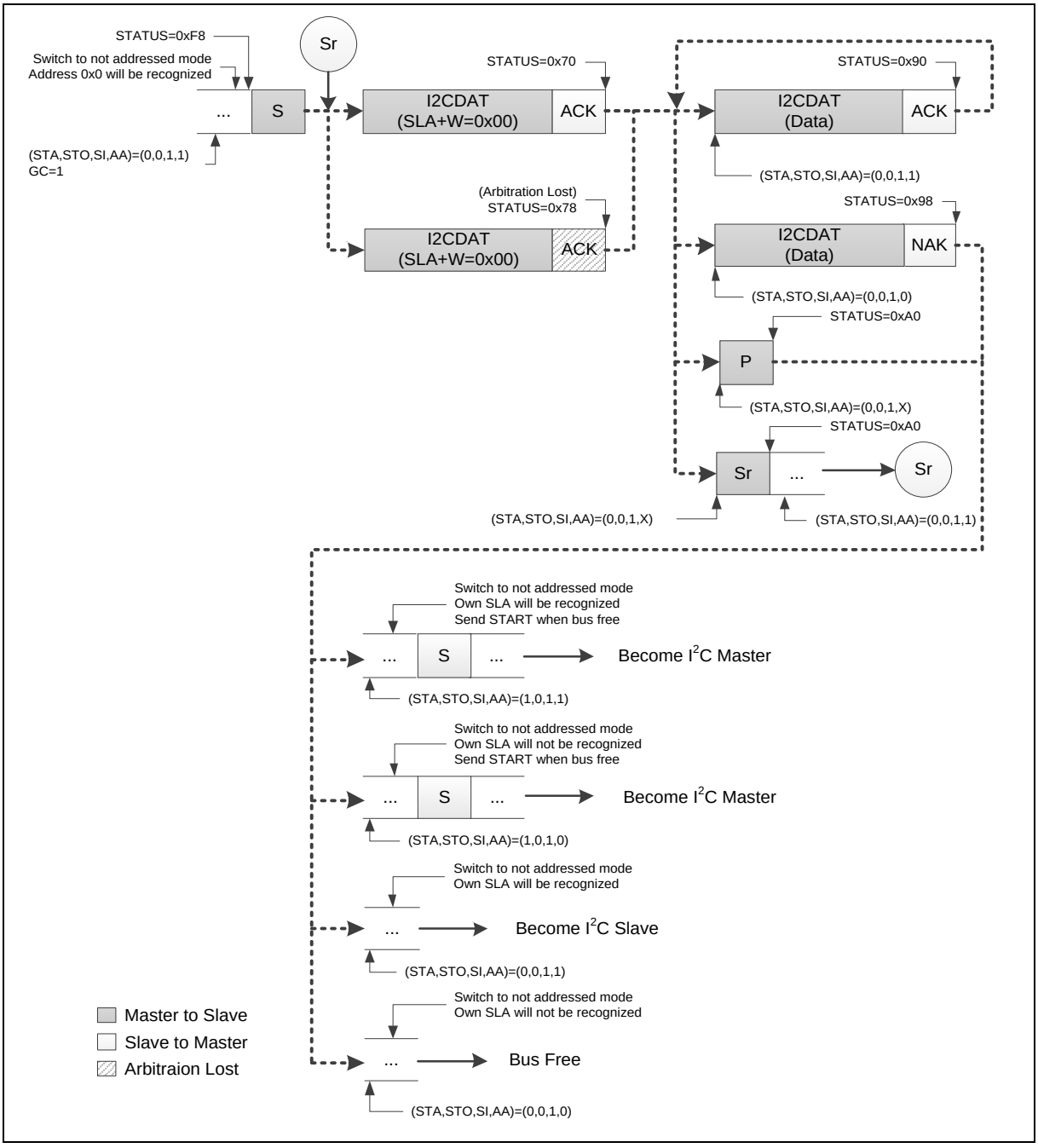


Figure 6.14-13 GC Mode

If I²C is still receiving data in GC mode but got a STOP or Repeat START, the status code will be

0xA0. User could follow the action for status code 0x98 in above figure when getting 0xA0 status.

Note: After slave gets status of 0x98 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I^2C signal or address from master. At this time, I^2C controller should be reset to leave this status.

6.14.5.2.4 *Multi-Master*

In some applications, there are two or more masters on the same I^2C bus to access slaves, and the masters may transmit data simultaneously. The I^2C supports multi-master by including collision detection and arbitration to prevent data corruption.

If for some reason two masters initiate command at the same time, the arbitration procedure determines which master wins and can continue with the command. Arbitration is performed on the I2Cn_SDA signal while the I2Cn_SCL signal is high. Each master checks if the I2Cn_SDA signal on the bus corresponds to the generated I2Cn_SDA signal. If the I2Cn_SDA signal on the bus is low but it should be high, then this master has lost arbitration. The device that has lost arbitration can generate I2Cn_SCL pulses until the byte ends. The arbitration procedure can continue until all the data is transferred. This means that in multi-master system each master must monitor the bus for collisions and act accordingly.

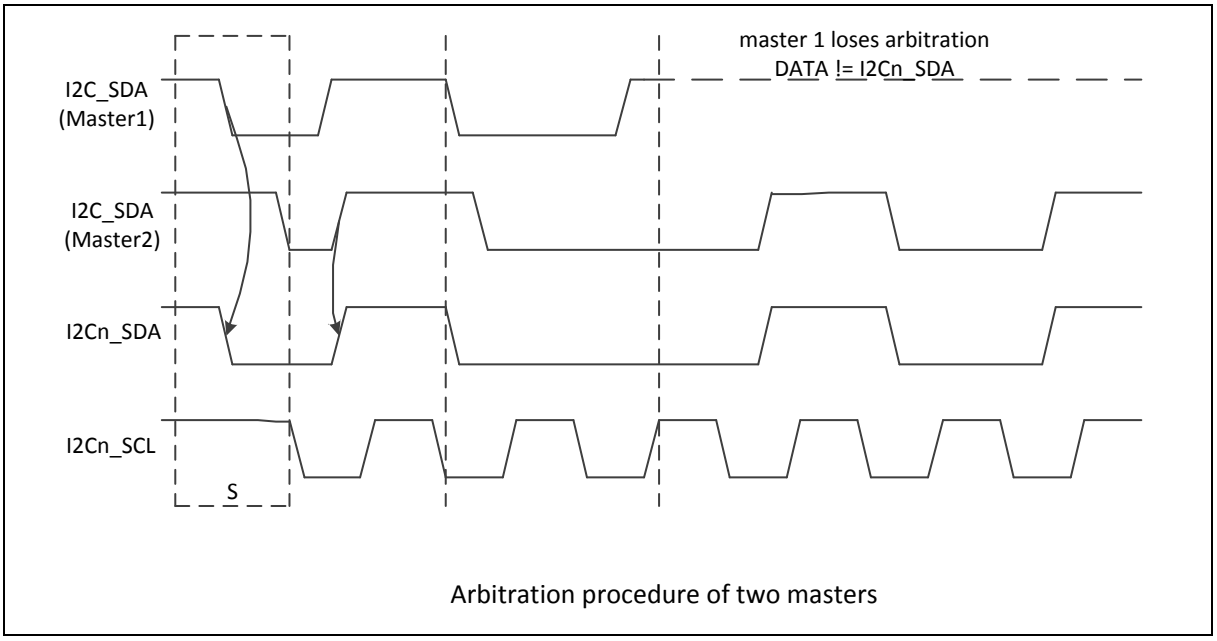


Figure 6.14-14 Arbitration Lost

- When I2CSTATUS = 0x38, an “Arbitration Lost” is received. Arbitration lost event maybe occur during the send START bit, data bits or STOP bit. User could set (STA, STO, SI, AA) = (1, 0, 1, X) to send START again when bus free, or set (STA, STO, SI, AA) = (0, 0, 1, X) back to not addressed Slave mode.
- When I2CSTATUS = 0x00, a “Bus Error” is received. To recover I^2C bus from a bus error, STO(I2CON[4]) should be set and SI(I2CON[3]) should be cleared, and then STO(I2CON[4]) is cleared to release bus.
  - Set (STA, STO, SI, AA) = (0, 1, 1, X) to stop current transfer
  - Set (STA, STO, SI, AA) = (0, 0, 1, X) to release bus

6.14.5.3 I^2C Protocol Registers

To control I²C port through the following fifteen special function registers: I2CON (Control register), I2CSTATUS (Status register), I2CDAT (Data register), I2CADDRn (Address registers, n=0~3), I2CADMn (Address mask registers, n=0~3), I2CLK (Clock rate register), I2CTOC (Time-out counter register), I2CWKCON(Wake up control register), I2CWKSTS(Wake up status register).

6.14.5.3.1 *Address Registers (I2CADDR)*

The I²C port is equipped with four slave address registers, I2CADDRn (n=0~3). The contents of the register are irrelevant when I²C is in Master mode. In Slave mode, the bit field I2CADDRn[7:1] must be loaded with the chip's own slave address. The I²C hardware will react if the contents of I2CADDRn are matched with the received slave address.

The I²C ports support the "General Call" function. If the GC (I2CADDRn [0]) bit is set the I²C port hardware will respond to General Call address (0x00). Clear GC bit to disable general call function.

When the GC bit is set and the I²C is in Slave mode, it can receive the general call address by 0x00 after Master send general call address to I²C bus, then it will follow status of GC mode.

6.14.5.3.2 *Slave Address Mask Registers (I2CADM)*

The I²C bus controller supports multiple address recognition with four address mask registers I2CADMn (n=0~3). When the bit in the address mask register is set to 1, it means the received corresponding address bit is "Don't care". If the bit is set to 0, it means the received corresponding register bit should be exactly the same as address register.

6.14.5.3.3 *Data Register (I2CDAT)*

This register contains a byte of serial data to be transmitted or a byte which just has been received. The CPU can be read from or written to the 8-bit (I2CDAT [7:0]) directly while it is not in the process of shifting a byte. When I²C is in a defined state and the SI (I2CON[3]) is set, data in I2CDAT [7:0] remains stable. While data is being shifted out, data on the bus is simultaneously being shifted in; I2CDAT [7:0] always contains the last data byte presented on the bus.

The acknowledge bit is controlled by the I²C hardware and cannot be accessed by the CPU. Serial data is shifted into I2CDAT [7:0] on the rising edges of serial clock pulses on the I2Cn_SCL line. When a byte has been shifted into I2CDAT [7:0], the serial data is available in I2CDAT [7:0], and the acknowledge bit (ACK or NACK) is returned by the control logic during the ninth clock pulse. In order to monitor bus status while sending data, the bus date will be shifted to I2CDAT[7:0] when sending I2CDAT[7:0] to bus. In the case of sending data, serial data bits are shifted out from I2CDAT [7:0] on the falling edge of I2Cn_SCL clocks, and is shifted to I2CDAT [7:0] on the rising edge of I2Cn_SCL clocks.

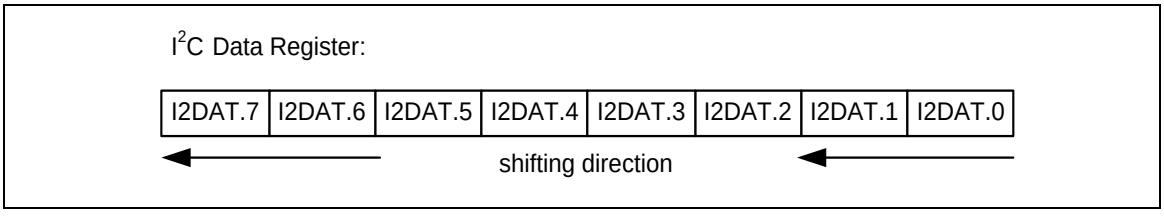


Figure 6.14-15 I²C Data Shifting Direction

6.14.5.3.4 *Control Register (I2CON)*

The CPU can be read from and written to I2CON register directly. When the I²C port is enabled by setting ENS1 (I2CON [6]) to high, the internal states will be controlled by I2CON and I²C logic hardware.

There are two bits are affected by hardware: the SI(I2CON[3]) bit is set when the I²C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO(I2CON[4]) bit is also cleared when ENS1(I2CON[6]) = 0.

Once a new status code is generated and stored in I2CSTATUS, the I²C Interrupt Flag bit SI will be set automatically. If the Enable Interrupt bit EI (I2CON [7]) is set at this time, the I²C interrupt will be generated. These bit fields I2CSTATUS[7:0] stores the internal state code, the content keeps stable until SI(I2CON[3]) is cleared by software.

6.14.5.3.5 *Status Register (I2CSTATUS)*

I2CSTATUS [7:0] is an 8-bit read-only register. The bit fields I2CSTATUS [7:0] contains the status code and there are 26 possible status codes. All states are listed in 0 when I2CSTATUS [7:0] is 0xF8, no serial interrupt is requested. All other I2CSTATUS [7:0] values correspond to the defined I²C states. When each of these states is entered, a status interrupt is requested (SI (I2CON[3]) = 1). A valid status code is present in I2CSTATUS[7:0] one cycle after SI set by hardware and is still present one cycle after SI reset by software.

In addition, the state 0x00 stands for a Bus Error, which occurs when a START or STOP condition is present at an incorrect position in the I²C format frame. A Bus Error may occur during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I²C from bus error, STO (I2CON[4]) should be set and SI(I2CON[3]) should be cleared to enter Not Addressed Slave mode. Then STO(I2CON[4]) is cleared to release bus and to wait for a new communication. The I²C bus cannot recognize stop condition during this action when a bus error occurs.

Master Mode		**Slave Mode**	
STATUS	**Description**	**STATUS**	**Description**
0x08	Start	0xA0	Slave Transmit Repeat Start or Stop
0x10	Master Repeat Start	0xA8	Slave Transmit Address ACK
0x18	Master Transmit Address ACK	0xB0	Slave Transmit Arbitration Lost
0x20	Master Transmit Address NACK	0xB8	Slave Transmit Data ACK
0x28	Master Transmit Data ACK	0xC0	Slave Transmit Data NACK
0x30	Master Transmit Data NACK	0xC8	Slave Transmit Last Data ACK
0x38	Master Arbitration Lost	0x60	Slave Receive Address ACK
0x40	Master Receive Address ACK	0x68	Slave Receive Arbitration Lost
0x48	Master Receive Address NACK	0x80	Slave Receive Data ACK
0x50	Master Receive Data ACK	0x88	Slave Receive Data NACK
0x58	Master Receive Data NACK	0x70	GC mode Address ACK
0x00	Bus error	0x78	GC mode Arbitration Lost
		0x90	GC mode Data ACK
		0x98	GC mode Data NACK
0xF8	Bus Released **Note:** Status “0xF8” exists in both master/slave modes, and it won’t raise interrupt.		

Table 6.14-1 I²C Status Code Description

6.14.5.3.6 *Clock Baud Rate Bits (I2CLK)*

The data baud rate of I²C is determines by I2CLK (I2CLK[7:0]) 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 data baud rate of I²C setting is Data Baud Rate of I²C = (system clock) / (4x (I2CLK [7:0] +1)). If system clock = 16 MHz, the I2CLK [7:0] = 40 (0x28), the data baud rate of I²C = 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec.

6.14.5.3.7 *Time-out Counter Register (I2CTOC)*

There is a 14-bit time-out counter which can be used to deal with the I²C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows (TIF (I2CTOC[0]) = 1) and generates I²C interrupt to CPU or stops counting by clearing ENTI(I2CTOC[2]) to 0. When time-out counter is enabled, writing 1 to the SI (I2CON[3]) flag will reset counter and re-start up counting after SI is cleared. If I²C bus hangs up, it causes the I2CSTATUS and flag SI (I2CON[3]) are not updated for a period, the 14-bit time-out counter may overflow and acknowledge CPU the I²C interrupt. Refer to the following figure for the 14-bit time-out counter. User may write 1 to clear TIF(I2C_TOC[0]) to 0.

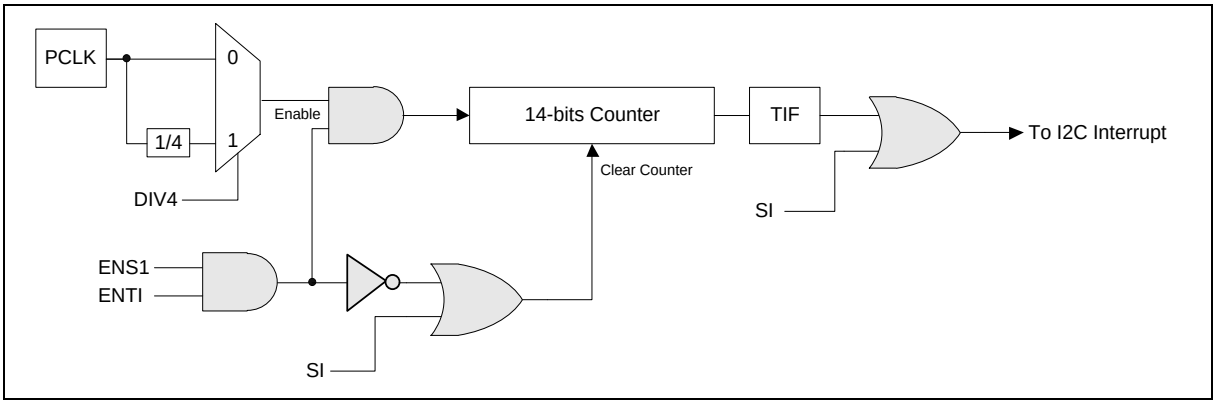


Figure 6.14-16 I²C Time-out Count Block Diagram

6.14.5.3.8 *Wake-up Control Register (I2CWKUPCON)*

When chip enters Power-down mode, other I²C master can wake up our chip by addressing our I²C device, user must configure the related setting before entering Sleep mode. When the chip is woken-up by address match with one of the four address register, the following data will be abandoned at this time.

6.14.5.3.9 *Wake-up Status Register (I2CWKUPSTS)*

When system is woken up by other I²C master device, WKUPIF(I2CWKUPSTS[0]) is set to indicate this event. User needs write “1” to clear this bit.

6.14.6 Example for Random Read on EEPROM

The following steps are used to configure the I²C0 related registers when using I²C to read data from EEPROM.

1. Set the multi-function pin in the “GPA_MFP” registers as I2C0_SCL and I2C0_SDA pins.
2. Enable I²C APB clock by setting I2C0_EN(APBCLK[8]).
3. Set I2C0_RST (IPRSTC2 [8]) = 1 to reset I²C controller then set I²C controller to normal

operation by setting I2C0_RST(IPRSTC2 [8]) = 0;

4. Set ENS1(I2CON[6])=1 to enable I^2C0 controller.
5. Write a divided value by setting I2CLK register for I^2C clock rate.
6. Set SETENA(NVIC_ISER[31:0])=0x00040000 in the "NVIC_ISER" register to set I^2C0 IRQ.
7. Set EI(I2CON[7])=1 to enable I^2C0 Interrupt.
8. Set I^2C0 address registers which are "I2CADDR0~I2CADDR3".

Random read operation is one of the methods of access EEPROM. The method allows the master to access any address of EEPROM space. The following figure shows the EEPROM random read operation.

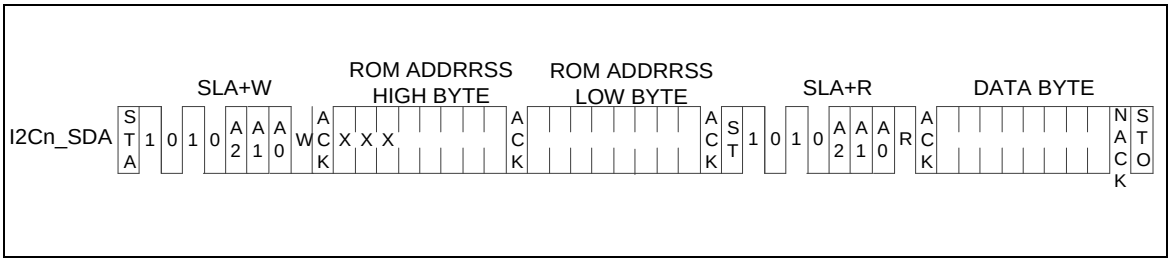


Figure 6.14-17 EEPROM Random Read

The following figure shows how to use I^2C controller to implement the protocol of EEPROM random read.

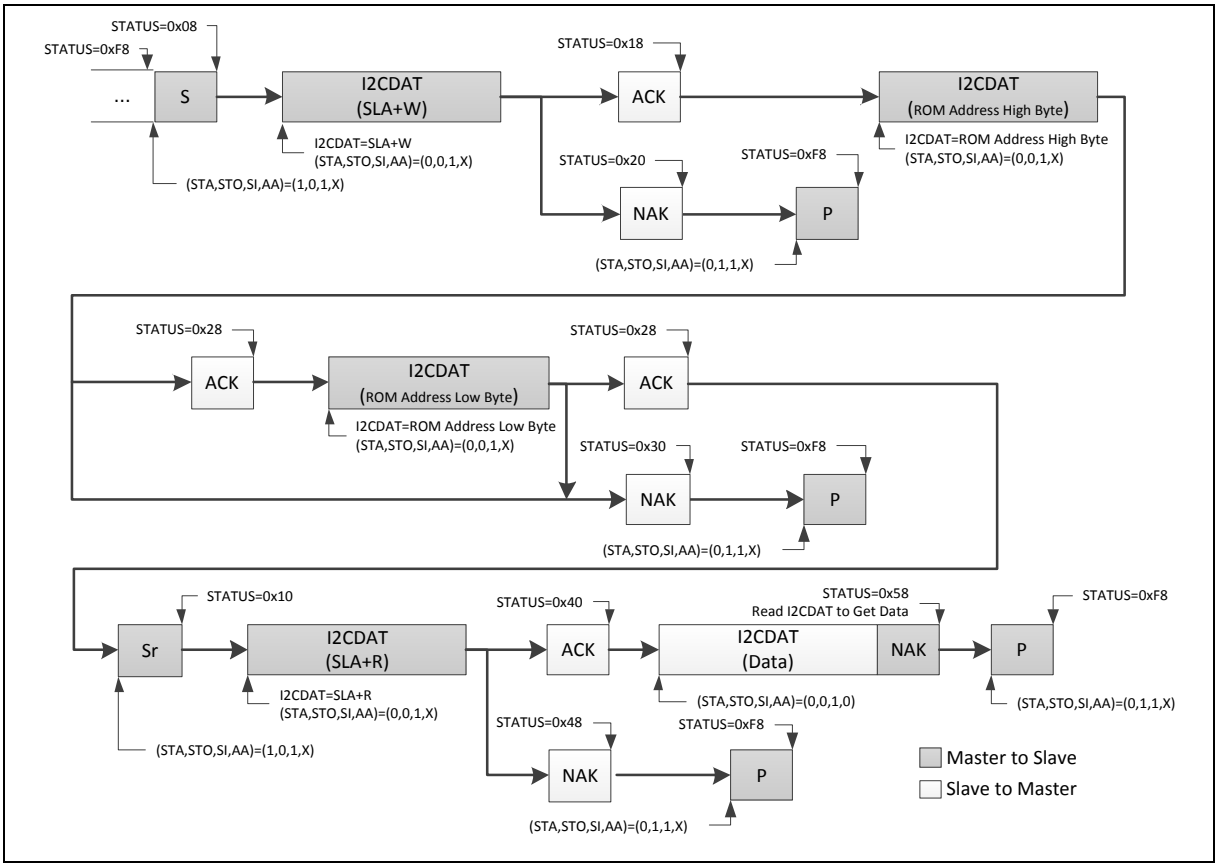


Figure 6.14-18 Protocol of EEPROM Random Read

The I^2C controller sends START to bus to be a master. 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.

6.14.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
I²C Base Address: **I2C0_BA = 0x4002_0000** **I2C1_BA = 0x4001_2000**				
I2CON **n=0,1**	I2Cn_BA+0x00	R/W	I²C Control Register	0x0000_0000
I2CADDR0 **n=0,1**	I2Cn_BA+0x04	R/W	I²C Slave Address Register0	0x0000_0000
I2CDAT **n=0,1**	I2Cn_BA+0x08	R/W	I²C Data Register	0x0000_0000
I2CSTATUS **n=0,1**	I2Cn_BA+0x0C	R	I²C Status Register	0x0000_00F8
I2CLK **n=0,1**	I2Cn_BA+0x10	R/W	I²C Clock Divided Register	0x0000_0000
I2CTOC **n=0,1**	I2Cn_BA+0x14	R/W	I²C Time-out Counter Register	0x0000_0000
I2CADDR1 **n=0,1**	I2Cn_BA+0x18	R/W	I²C Slave Address Register1	0x0000_0000
I2CADDR2 **n=0,1**	I2Cn_BA+0x1C	R/W	I²C Slave Address Register2	0x0000_0000
I2CADDR3 **n=0,1**	I2Cn_BA+0x20	R/W	I²C Slave Address Register3	0x0000_0000
I2CADM0 **n=0,1**	I2Cn_BA+0x24	R/W	I²C Slave Address Mask Register0	0x0000_0000
I2CADM1 **n=0,1**	I2Cn_BA+0x28	R/W	I²C Slave Address Mask Register1	0x0000_0000
I2CADM2 **n=0,1**	I2Cn_BA+0x2C	R/W	I²C Slave Address Mask Register2	0x0000_0000
I2CADM3 **n=0,1**	I2Cn_BA+0x30	R/W	I²C Slave Address Mask Register3	0x0000_0000
I2CWKUPCON **n=0,1**	I2Cn_BA+0x3C	R/W	I²C Wake-up Control Register	0x0000_0000
I2CWKUPSTS **n=0,1**	I2Cn_BA+0x40	R/W	I²C Wake-up Status Register	0x0000_0000

6.14.8 Register Description

I²C Control Register (I2CON)

Register	Offset	R/W	Description	Reset Value
I2CON **n=0,1**	I2Cn_BA+0x00	R/W	I²C 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**
EI	**ENS1**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**EI**	**Interrupt Enable Bit** 0 = I²C interrupt Disabled. 1 = I²C interrupt Enabled.
[6]	**ENS1**	**I²C Controller Enable Bit** 0 = Disabled. 1 = Enabled. Set to enable I²C serial function controller. When ENS1=1 the I²C serial function enables. The multi-function pin function of I2Cn_SDA and I2Cn_SCL must set to I²C function first.
[5]	**STA**	**I²C START Control** Setting STA to logic 1 to enter Master mode, the I²C hardware sends a START or repeat START condition to bus when the bus is free.
[4]	**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. This means it is NO LONGER in the slave receiver mode to receive data from the master transmit device.
[3]	**SI**	**I²C Interrupt Flag** When a new I²C state is present in the I2CSTATUS register, the SI flag is set by hardware, and if bit EI (I2CON [7]) is set, the I²C interrupt is requested. SI must be cleared by software. Clear SI by writing 1 to this bit.
[2]	**AA**	**Assert Acknowledge Control** When AA =1 prior to address or data received, an acknowledged (low level to I2Cn_SDA) will be returned during the acknowledge clock pulse on the I2Cn_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 I2Cn_SDA) will be returned during the acknowledge

		clock pulse on the I2Cn_SCL line.
[1:0]	**Reserved**	Reserved.

I²C Data Register (I2CDAT)

Register	Offset	R/W	Description	Reset Value
I2CDAT n=0,1	I2Cn_BA+0x08	R/W	I²C Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CDAT							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**I2CDAT**	**I²C Data Register** This field is located with the 8-bit transferred data of I²C serial port.

I²C Status Register (I2CSTATUS)

Register	Offset	R/W	Description	Reset Value
I2CSTATUS n=0,1	I2Cn_BA+0x0C	R	I²C Status Register	0x0000_00F8

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CSTATUS							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**I2CSTATUS**	**I²C Status Register** There are 26 possible status codes. When I2CSTATUS contains 0xF8, no serial interrupt is requested. All other I2CSTATUS values correspond to defined I²C states. When each of these states is entered, a status interrupt is requested (SI (I2CON[3])= 1). A valid status code is present in I2CSTATUS one cycle after SI is set by hardware and is still present one cycle after SI has been reset by software. In addition, states 0x00 stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the formation frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit.

I²C Clock Divided Register (I2CLK)

Register	Offset	R/W	Description	Reset Value
I2CLK **n=0,1**	I2Cn_BA+0x10	R/W	I²C Clock Divided Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CLK							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**I2CLK**	**I²C Clock Divided Register** The I²C clock rate bits: Data Baud Rate of I²C = (system clock) / (4x (I2CLK+1)). **Note:** The minimum value of I2CLK is 4.

I²C Time-out Counter Register (I2CTOC)

Register	Offset	R/W	Description	Reset Value
I2CTOC n=0,1	I2Cn_BA+0x14	R/W	I²C Time-out 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					**ENTI**	**DIV4**	**TIF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ENTI**	**Time-Out Counter Enable Bit** 0 = Disabled. 1 = Enabled. When Enabled, the 14-bit time-out counter will start counting when SI(I2CON[3]) is clear. Setting flag SI SI(I2CON[3]) to high will reset counter and re-start up counting after SI SI(I2CON[3]) is cleared.
[1]	**DIV4**	**Time-Out Counter Input Clock Divided By 4** 0 = Disabled. 1 = Enabled. When Enabled, The time-out period is extend 4 times.
[0]	**TIF**	**Time-Out Flag** This bit is set by hardware when I²C time-out happened and it can interrupt CPU if I²C interrupt enable bit EI(I2CON[7]) is set to 1. **Note:** Write 1 to clear this bit.

I²C Slave Address Register (I2CADDRx)

Register	Offset	R/W	Description	Reset Value
I2CADDR0 n=0,1	I2Cn_BA+0x04	R/W	I²C Slave Address Register0	0x0000_0000
I2CADDR1 n=0,1	I2Cn_BA+0x18	R/W	I²C Slave Address Register1	0x0000_0000
I2CADDR2 n=0,1	I2Cn_BA+0x1C	R/W	I²C Slave Address Register2	0x0000_0000
I2CADDR3 n=0,1	I2Cn_BA+0x20	R/W	I²C Slave Address Register3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CADDR							**GC**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**I2CADDR**	**I²C Address Register** The content of this register is irrelevant when I²C is in Master mode. In the slave mode, the seven most significant bits must be loaded with the chip's own address. The I²C hardware will react if either of the address is matched.
[0]	**GC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled.

I²C Slave Address Mask Register (I2CADMx)

Register	Offset	R/W	Description	Reset Value
I2CADM0 n=0,1	I2Cn_BA+0x24	R/W	I²C Slave Address Mask Register0	0x0000_0000
I2CADM1 n=0,1	I2Cn_BA+0x28	R/W	I²C Slave Address Mask Register1	0x0000_0000
I2CADM2 n=0,1	I2Cn_BA+0x2C	R/W	I²C Slave Address Mask Register2	0x0000_0000
I2CADM3 n=0,1	I2Cn_BA+0x30	R/W	I²C Slave Address Mask Register3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CADM							**Reserved**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**I2CADM**	**I²C Address Mask Register** 0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don't care.). I²C bus controllers support multiple address recognition with four address mask register. When the bit in the address mask register is set to one, it means the received corresponding address bit is don't-care. If the bit is set to zero, that means the received corresponding register bit should be exact the same as address register.
[0]	**Reserved**	Reserved.

I²C Wake-up Control Register (I2CWKUPCON)

Register	Offset	R/W	Description	Reset Value
I2CWKUPCON n=0,1	I2Cn_BA+0x3C	R/W	I²C Wake-up Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKUPEN**	**I²C Wake-Up Enable Bit** 0 = I²C wake-up function Disabled. 1= I²C wake-up function Enabled.

I²C Wake-up Status Register (I2CWKUPSTS)

Register	Offset	R/W	Description	Reset Value
I2CWKUPSTS n=0,1	I2Cn_BA+0x40	R/W	I²C Wake-up Status Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKUPIF**	**I²C Wake-Up Flag** 0 = Chip is not woken-up from Power-down mode by I²C. 1 = Chip is woken-up from Power-down mode by I²C. **Note:** Software can write 1 to clear this bit.

6.15 Serial Peripheral Interface (SPI)

6.15.1 Overview

The Serial Peripheral Interface (SPI) is a synchronous serial data communication protocol that operates in full duplex mode. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The NuMicro® NUC029LEE/NUC029SEE contains up to two sets of SPI controllers performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. Each set of SPI controller can be configured as a master or a slave device.

The SPI controller supports the variable bus clock function for special applications. This controller also supports the PDMA function to access the data buffer and also supports Dual I/O Transfer mode.

6.15.2 Features

- Up to two sets of SPI controllers
- Supports Master or Slave mode operation
- Supports Dual I/O Transfer mode
- Configurable bit length of a transaction word from 8 to 32 bits
- Provides separate 8-layer depth transmit and receive FIFO buffers
- Supports MSB first or LSB first transfer sequence
- Two slave select lines in Master mode
- Supports the Byte Reorder function
- Supports Byte or Word Suspend mode
- Variable output bus clock frequency in Master mode
- Supports PDMA transfer
- Supports 3-wire, no slave select signal, bi-direction interface

6.15.3 Block Diagram

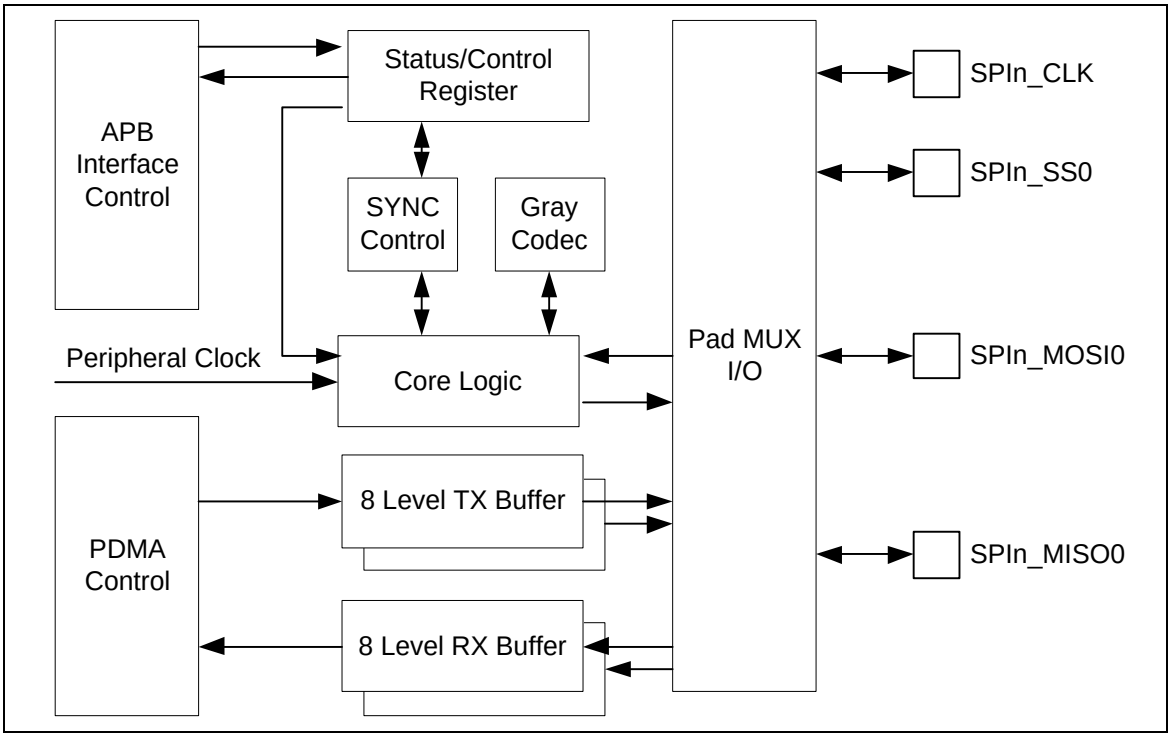


Figure 6.15-1 SPI Block Diagram

6.15.4 Basic Configuration

The basic configurations of SPI0 are as follows:

- SPI0 pin functions are configured in ALT_MFP, GPB_MFP and GPC_MFP registers.
- Select the source of SPI0 peripheral clock on SPI0_S (CLKSEL1[4]).
- Enable SPI0 peripheral clock on SPI0_EN (APBCLK[12]).
- Reset SPI0 controller on SPI0_RST (IPRSC2[12]).

The basic configurations of SPI1 are as follows:

- SPI1 pin functions are configured in ALT_MFP, GPB_MFP and GPC_MFP registers.
- Select the source of SPI1 peripheral clock on SPI1_S (CLKSEL1[5]).
- Enable SPI1 peripheral clock on SPI1_EN (APBCLK[13]).
- Reset SPI1 controller on SPI1_RST (IPRSC2[13]).

6.15.5 Functional Description

6.15.5.1 Terminology

SPI Peripheral Clock and SPI Bus Clock

The SPI controller needs the SPI peripheral clock to drive the SPI logic unit to perform the data

transfer. The SPI bus clock is the clock presented on SPIn_CLK pin.

The SPI peripheral clock rate is determined by the settings of clock source, BCn option and clock divisor. The SPIn_S bit of CLKSEL1 register determines the clock source of the SPI peripheral clock. The clock source can be HCLK or PLL output clock. Set the BCn bit of SPI_CNTRL2 register to 0 for the compatible SPI clock rate calculation of previous products. DIVIDER (SPI_DIVIDER[7:0]) setting determines the divisor of the clock rate calculation.

In Master mode, if the variable clock function is disabled, the output frequency of the SPI bus clock output pin is equal to the SPI peripheral clock rate. In general, the SPI bus clock is denoted as SPI clock. In Slave mode, the SPI bus clock is provided by an off-chip master device. The SPI peripheral clock rate of slave device must be faster than the SPI bus clock rate of the master device connected together. The frequency of SPI peripheral clock cannot be faster than the APB clock rate regardless of Master or Slave mode.

Master/Slave Mode

The SPI controller can be set as Master or Slave mode by setting SLAVE (SPI_CNTRL[18]) to communicate with the off-chip SPI Slave or Master device. The application block diagrams in Master and Slave mode are shown below.

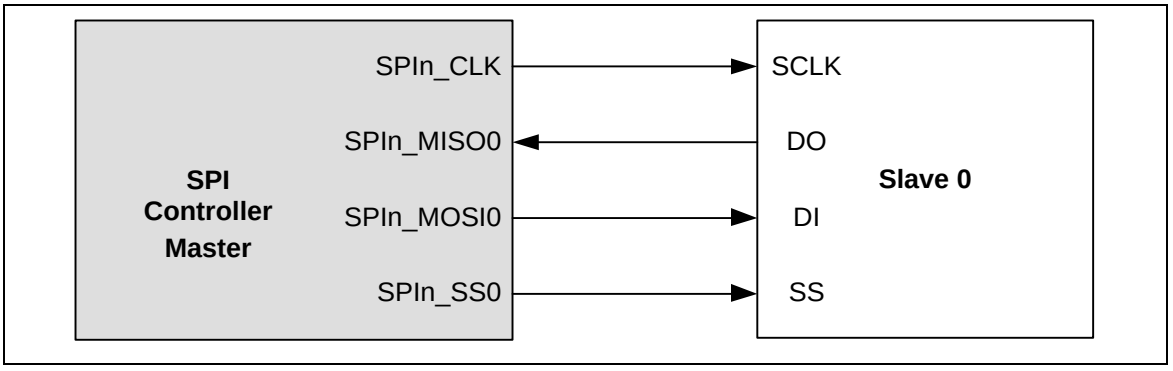


Figure 6.15-2 SPI Master Mode Application Block Diagram

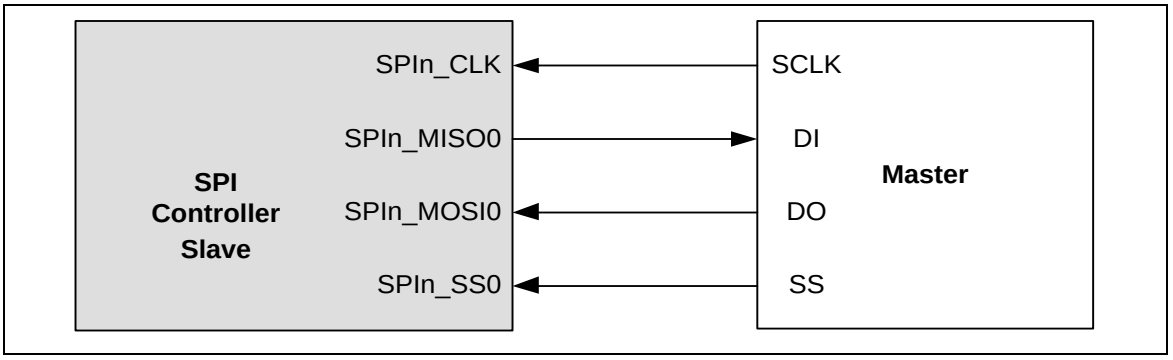


Figure 6.15-3 SPI Slave Mode Application Block Diagram

Clock Polarity

The CLKP (SPI_CTL[11]) defines the bus clock idle state. If CLKP = 1, the SPIn_CLK output is idle at high state, otherwise it is at low state if CLKP = 0.

Transmit/Receive Bit Length

The bit length of a transaction word is defined in TX_BIT_LEN bit field (SPI_CNTRL[7:3]). It can be configured up to 32-bit length in a transaction word for transmitting and receiving.

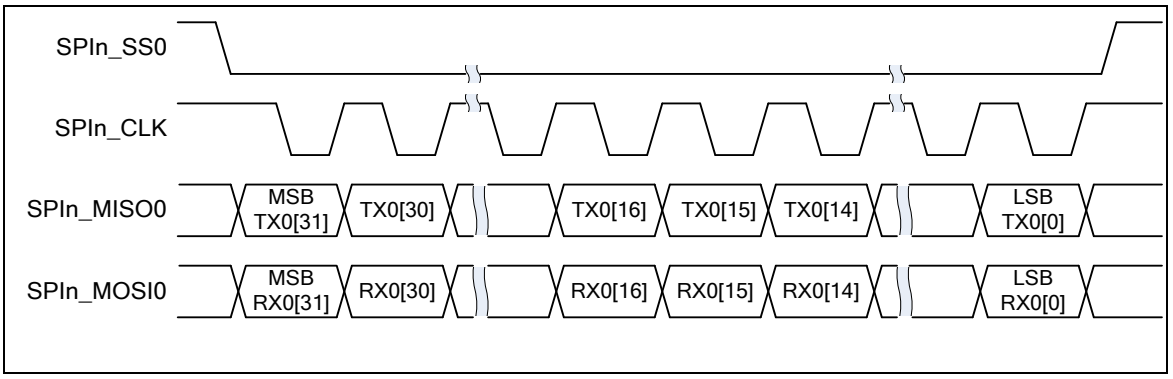


Figure 6.15-4 32-Bit in One Transaction

LSB/MSB First

LSB (SPI_CNTRL[10]) defines the bit transfer sequence in a transaction. If the LSB bit is set to 1, the transfer sequence is LSB first. The bit 0 will be transferred firstly. If the LSB bit is cleared to 0, the transfer sequence is MSB first.

Transmit Edge

TX_NEG (SPI_CNTRL[2]) defines the data transmitted out either on negative edge or on positive edge of SPI bus clock.

Receive Edge

RX_NEG (SPI_CNTRL[1]) defines the data received either on negative edge or on positive edge of SPI clock.

Note: The settings of TX_NEG and RX_NEG are mutual exclusive. In other words, do not transmit and receive data on the same clock edge.

Word Suspend

SP_CYCLE (SPI_CNTRL[15:12]) provide 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 duration between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SP_CYCLE is 0x3 (3.5 SPI bus clock cycles). This SP_CYCLE setting will not take effect to the word suspend interval if FIFO mode is disabled by software.

If both VARCLK_EN (SPI_CNTRL[23]) and FIFO (SPI_CNTRL[21]) bits are set to 1, the minimum word suspend period is (6.5 + SP_CYCLE)*SPI clock period.

Slave Selection

In Master mode, this SPI controller can drive off-chip slave devices through the slave select output pins SPIn_SS0. In Slave mode, the off-chip master device drives the slave select signal from the SPIn_SS0 input pin to this SPI controller. In Master and Slave mode, the active state of slave select signal can be programmed to low or high active in SS_LVL (SPI_SSR[2]), and SS_LTRIG (SPI_SSR[4]) defines the slave select signal SPIn_SS0 is level-triggered or edge-

triggered. The selection of trigger conditions depends on what type of peripheral slave/master device is connected.

In Slave mode, if the SS_LTRIG bit is configured as level trigger, the LTRIG_FLAG (SPI_SSR[5]) is used to indicate if the received bits among one transaction meets the requirement defined in TX_BIT_LEN (SPI_CNTRL[7:3]).

Level-trigger/Edge-trigger

In Slave mode, the slave select signal can be configured as level-trigger or edge-trigger. In edge-trigger, the data transfer starts from an active edge and ends on an inactive edge of the slave select signal. The unit-transfer interrupt flag (SPI_CNTRL[16]) will be set to 1 as an inactive edge is detected. If the master does not send an inactive edge to slave, the transfer procedure will not be completed and the unit transfer interrupt flag of slave will not be set. In level-trigger, the unit-transfer interrupt flag of slave will be set when one of the following two conditions occurs. The first condition is that if the number of transferred bits matches the settings of TX_BIT_LEN, the unit transfer interrupt flag of slave will be set. As to the second condition, if the master set the slave select pin to inactive level during the transfer is in progress, it will force slave device to terminate the current transfer no matter how many bits have been transferred and the unit transfer interrupt flag will be set. User can read the status of LTRIG_FLAG bit to check if the data has been completely transferred.

6.15.5.2 Automatic Slave Selection

In Master mode, if AUTOSS (SPI_SSR[3]) is set to 1, the slave select signals will be generated automatically and output to the SPIn_SS0 pins according to whether SSR[0] (SPI_SSR[0]) are enabled or not. This means that the slave select signals, which are selected in SSR[0], will be asserted by the SPI controller when the SPI data transfer is started by setting the GO_BUSY bit (SPI_CNTRL[0]) and will be de-asserted after the data transfer is finished. If the AUTOSS bit is cleared, the slave select output signals will be asserted/de-asserted by setting/clearing the related bits of SPI_SSR[0]. The active state of the slave select output signals is specified in SS_LVL (SPI_SSR[2]).

In Master mode, if the value of SP_CYCLE[3:0] is less than 3 and the AUTOSS is set as 1, the slave select signal will be kept in active state between two successive transactions.

In Slave mode, to recognize the inactive state of the slave select signal, the inactive period of the slave select signal must be larger than or equal to 6 peripheral clock periods between two successive transactions.

6.15.5.3 Variable Bus Clock Frequency

In Master mode, if VARCLK_EN (SPI_CNTRL[23]) is set to 1, the output of SPI clock can be programmed as variable frequency pattern. The SPI clock period of each cycle depends on the setting of the SPI_VARCLK register. When the variable clock function is enabled, the TX_BIT_LEN setting must be set as 0x10 to configure the data transfer as 16-bit transfer mode. The VARCLK[31] determines the clock period of the first clock cycle. If VARCLK[31] is 0, the first clock cycle depends on the DIVIDER setting; if it is 1, the first clock cycle depends on the DIVIDER2 setting. Two successive bits in VARCLK[30:1] defines one clock cycle. If the two successive bits are 00, the clock cycle depends on the DIVIDER setting; if they are 11, the clock cycle depends on the DIVIDER2 setting. The bit field VARCLK[30:29] defines the second clock cycle of SPI clock of a transaction, and the bit field VARCLK[28:27] defines the third clock cycle, and so on. The VARCLK[0] has no meaning. The following figure shows the timing relationship among the SPI bus clock, the VARCLK setting, the DIVIDER setting and the DIVIDER2 setting.

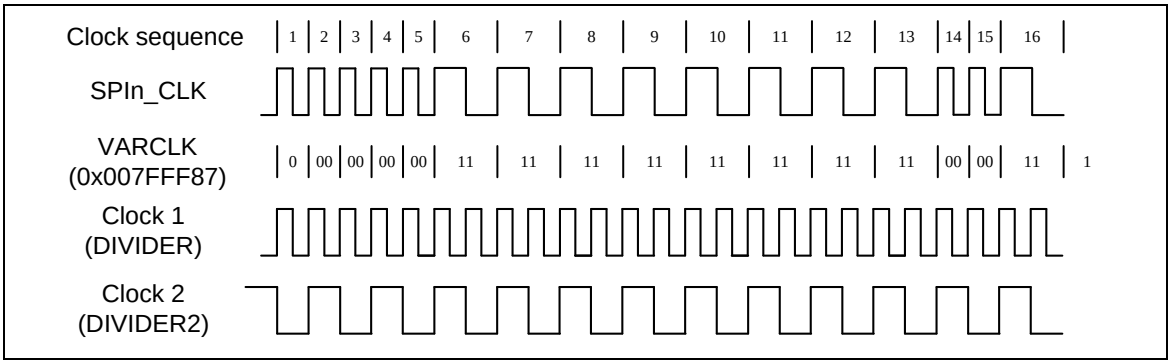


Figure 6.15-5 Variable Bus Clock Frequency

6.15.5.4 Byte Reorder Function

When the transfer is set as MSB first (LSB = 0) and the REORDER bit is set to 1, the data stored in the TX buffer and RX buffer will be rearranged in the order as [Byte0, Byte1, Byte2, Byte3] in 32-bit Transfer mode (TX_BIT_LEN = 0). The sequence of transmitted/received data will be Byte0, Byte1, Byte2, and then Byte3. If the TX_BIT_LEN is set as 24-bit transfer mode, the data in TX buffer and RX buffer will be rearranged as [unknown byte, Byte0, Byte1, Byte2]. The SPI controller will transmit/receive data with the sequence of Byte0, Byte1 and then Byte2. Each byte will be transmitted/received with MSB first. The rule of 16-bit mode is the same as above. Byte Reorder function is only available when TX_BIT_LEN is configured as 16, 24, and 32 bits.

Note: The Byte Reorder function is not supported when the variable bus clock function is enabled.

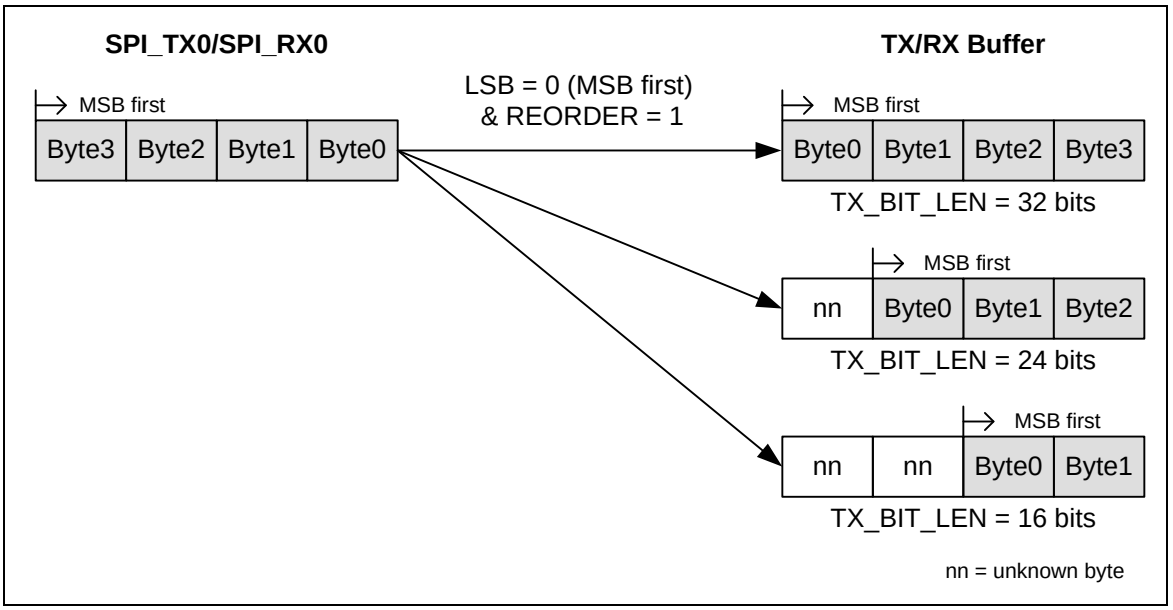


Figure 6.15-6 Byte Reorder Function

6.15.5.5 Byte Suspend Function

In Master mode, if REORDER (SPI_CNTRL[19]) is set to 1, a suspend interval of 0.5 ~ 15.5 SPI clock periods will be inserted by hardware between two successive bytes in a transaction word. Both settings of byte suspend interval and word suspend interval are configured in SP_CYCLE

(SPI_CNTRL[15:12]).

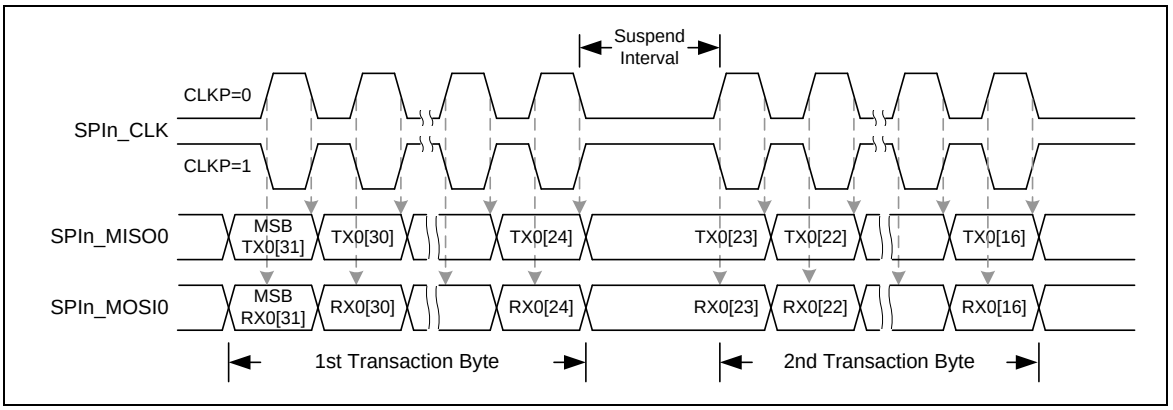


Figure 6.15-7 Timing Waveform for Byte Suspend

6.15.5.6 Slave 3-wire Mode

When NOSLVSEL (SPI_CNTRL2[8]) is set by software to enable the Slave 3-wire mode, the SPI controller can work with no slave select signal in Slave mode. The NOSLVSEL bit only takes effect in Slave mode. Only three pins, SPIn_CLK, SPIn_MISO0, and SPIn_MOSI0, are required to communicate with a SPI master. The SPIn_SS pin can be configured as a GPIO. When the NOSLVSEL bit is set to 1, the SPI slave will be ready to transmit/receive data after the GO_BUSY bit is set to 1. As the number of received bits meets the requirement which defined in TX_BIT_LEN (SPI_CNTRL[7:3]), the unit-transfer interrupt flag, IF (SPI_CNTRL[16]), will be set to 1.

Note: In Slave 3-wire mode, the SS_LTRIG (SPI_SSR[4]) should be set as 1.

6.15.5.7 Dual I/O Mode

The SPI controller also supports Dual I/O transfer when setting the DUAL_IO_EN (SPI_CNTRL2[13]) to 1. Many general SPI flashes support Dual I/O transfer. The DUAL_IO_DIR (SPI_CNTRL2[12]) is used to define the direction of the transfer data. When the DUAL_IO_DIR bit is set to 1, the controller will send the data to external device. When the DUAL_IO_DIR bit is set to 0, the controller will read the data from the external device. This function supports 8, 16, 24, and 32-bit data transfer.

The Dual I/O mode is not supported when the Slave 3-wire mode or the Byte Reorder function is enabled.

If both the DUAL_IO_EN and DUAL_IO_DIR bits are set as 1, the SPIn_MOSI0 is the even bit data output and the SPIn_MISO0 will be set as the odd bit data output. If the DUAL_IO_EN is set as 1 and DUAL_IO_DIR is set as 0, both the SPIn_MISO0 and SPIn_MOSI0 will be set as data input ports.

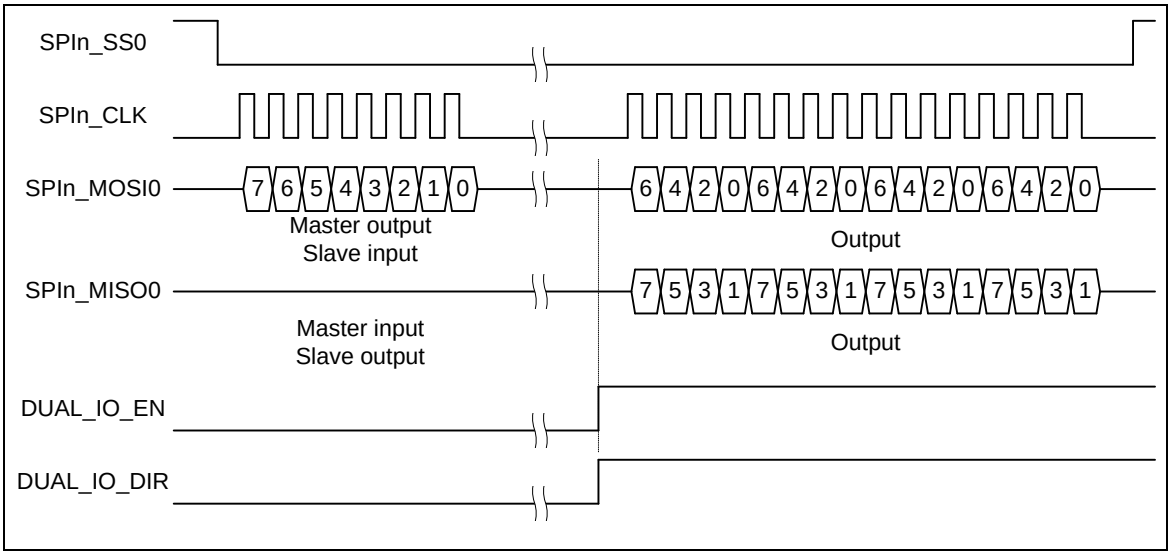


Figure 6.15-8 Bit Sequence of Dual Output Mode

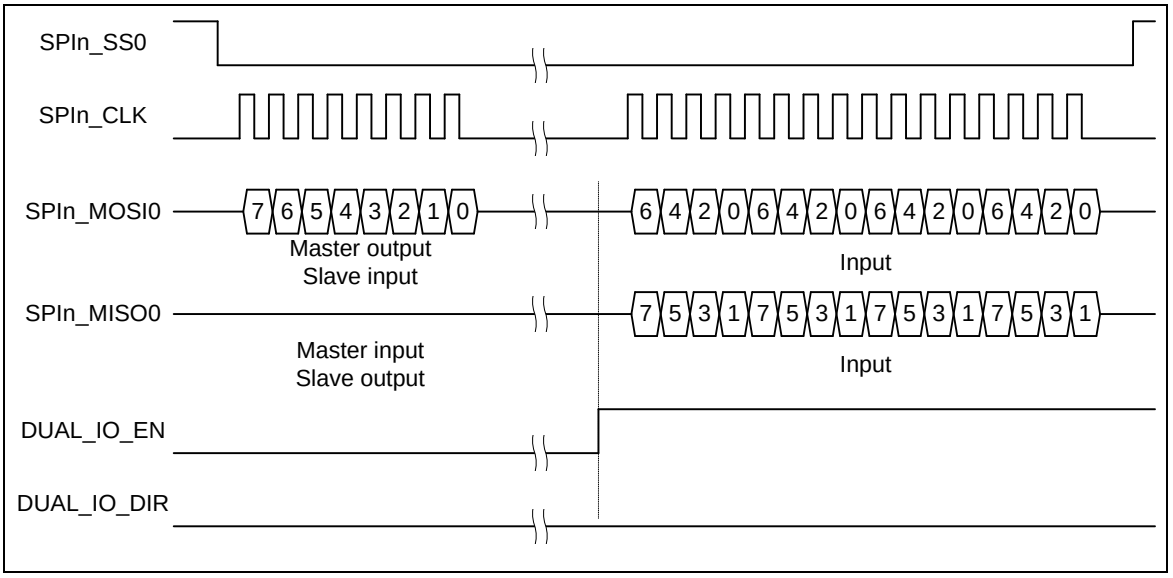


Figure 6.15-9 Bit Sequence of Dual Input Mode

6.15.5.8FIFO Mode

The SPI controller supports FIFO mode when the FIFO bit in SPI_CNTRL[21] is set as 1. The SPI controllers equip with eight 32-bit wide transmit and receive FIFO buffers.

The transmit FIFO buffer is an 8-layer depth, 32-bit wide, first-in, first-out register buffer. Data can be written to the transmit FIFO buffer through software by writing the SPI_TX0 register. The data stored in the transmit FIFO buffer will be read and sent out by the transmission control logic. If the 8-layer transmit FIFO buffer is full, the TX_FULL bit will be set to 1. When the SPI transmission logic unit draws out the last datum of the transmit FIFO buffer, so that the 8-layer transmit FIFO buffer is empty, the TX_EMPTY bit will be set to 1. Notice that the TX_EMPTY flag is set to 1 while the last transaction is still in progress. In Master mode, both the GO_BUSY bit and TX_EMPTY bit should be checked by software to make sure whether the SPI is in idle or not.

The received FIFO buffer is also an 8-layer depth, 32-bit wide, first-in, first-out register buffer. The receive control logic will store the received data to this buffer. The FIFO buffer data can be read from SPI_RX0 register by software. There are FIFO related status bits, like RX_EMPTY and RX_FULL, to indicate the current status of FIFO buffer.

In FIFO mode, the transmitting and receiving threshold can be set through software by setting the TX_THRESHOLD and RX_THRESHOLD settings. When the count of valid data stored in transmit FIFO buffer is less than or equal to TX_THRESHOLD setting, the TX_INTSTS bit will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RX_THRESHOLD setting, the RX_INTSTS bit will be set to 1.

In FIFO mode, 8 data can be written to the SPI transmit FIFO buffer by software in advance. When the SPI controller operates with FIFO mode, the GO_BUSY bit of SPI_CNTRL register will be controlled by hardware, and the content of SPI_CNTRL register should not be modified by software unless the FIFO bit is cleared to disable FIFO mode.

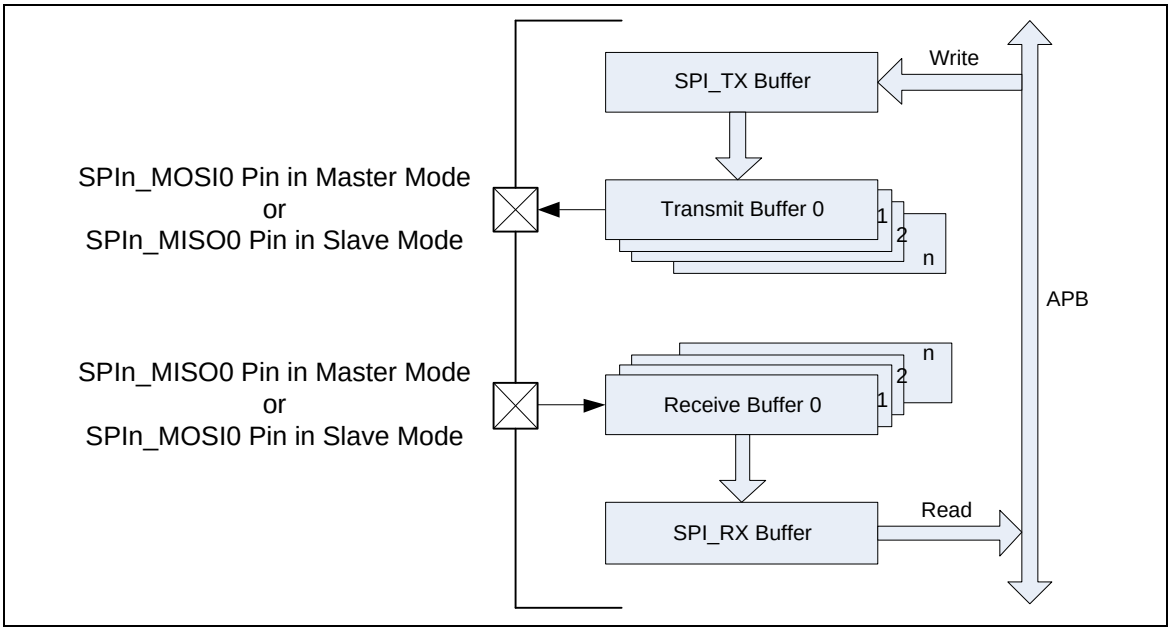


Figure 6.15-10 FIFO Mode Block Diagram

In Master mode, when the FIFO bit is set to 1 and the first datum is written to the SPI_TX0 register, the TX_EMPTY flag will be cleared to 0. The transmission immediately starts as long as the transmit FIFO buffer is not empty. User can write the next data into SPI_TX0 register immediately. The SPI controller will insert a suspend interval between two successive transactions in FIFO mode and the period of suspend interval is decided by the setting of SP_CYCLE (SPI_CNTRL [15:12]). User can write data into SPI_TX0 register as long as the TX_FULL flag is 0.

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the SPI_TX0 register does not be updated after all data transfer are done, the transfer will stop.

In Master mode, during receiving operation, the serial data are received from SPIn_MISO0/1 pin and stored to receive FIFO buffer. The RX_EMPTY flag will be cleared to 0 while the receive FIFO buffer contains unread data. The received data can be read by software from SPI_RX0 register as long as the RX_EMPTY flag is 0. If the receive FIFO buffer contains 8 unread data, the RX_FULL flag will be set to 1. The SPI controller will stop receiving data until the SPI_RX0 register is read by software.

In Slave mode, when the FIFO bit is set as 1, the GO_BUSY bit will be set as 1 by hardware automatically.

In Slave mode, during transmission operation, when data is written to the SPI_TX0 register by software, the data will be loaded into transmit FIFO buffer and the TX_EMPTY flag will be set to 0. The transmission will start when the slave device receives clock signal from master. Data can be written to SPI_TX0 register as long as the TX_FULL flag is 0. After all data have been drawn out by the SPI transmission logic unit and the SPI_TX0 register is not updated by software, the TX_EMPTY flag will be set to 1.

In Slave mode, during receiving operation, the serial data is received from SPIn_MOSI0/1 pin and stored to SPI_RX0 register. The reception mechanism is similar to Master mode reception operation.

6.15.5.9 Interrupt

- SPI unit transfer interrupt

As the SPI controller finishes a unit transfer, the unit transfer interrupt flag IF (SPI_CNTRL[16]) will be set to 1. The unit transfer interrupt event will generate an interrupt to CPU if the unit transfer interrupt enable bit IE (SPI_CNTRL[17]) is set. The unit transfer interrupt flag can be cleared only by writing 1 to it.

- SPI Slave 3-wire mode start interrupt

In 3-wire mode, the slave 3-wire mode start interrupt flag, SLV_START_INTSTS, will be set to 1 when the slave senses the SPI clock signal. The SPI controller will issue an interrupt if the SSTA_INTEN is set to 1. If the count of the received bits is less than the setting of TX_BIT_LEN and there is no more SPI clock input over the expected time period which is defined by the user, the user can set the SLV_ABORT bit to abort the current transfer. The unit transfer interrupt flag, IF, will be set to 1 if the software set the SLV_ABORT bit.

- Receive FIFO time-out interrupt

In FIFO mode, there is a time-out function to inform user. If there is a received data in the FIFO and it is not read by software over 64 SPI peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode, it will send a time-out interrupt to the system if the time-out interrupt enable bit, FIFO_CTL[21], is set to 1.

- Transmit FIFO interrupt

In FIFO mode, if the valid data count of the transmit FIFO buffer is less than or equal to the setting value of TX_THRESHOLD, the transmit FIFO interrupt flag will be set to 1. The SPI controller will generate a transmit FIFO interrupt to the system if the transmit FIFO interrupt enable bit, SPI_FIFO_CTL[3], is set to 1.

- Receive FIFO interrupt

In FIFO mode, if the valid data count of the receive FIFO buffer is larger than the setting value of RX_THRESHOLD, the receive FIFO interrupt flag will be set to 1. The SPI controller will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, SPI_FIFO_CTL[2], is set to 1.

6.15.6 Timing Diagram

The active state of slave select signal can be defined by setting the SS_LVL (SPI_SSR[2]) and SS_LTRIG (SPI_SSR[4]). The SPI clock which is in idle state can be configured as high or low state by setting the CLKP (SPI_CNTRL[11]). It also provides the bit length of a transaction word in TX_BIT_LEN (SPI_CNTRL[7:3]), and transmitting/receiving data from MSB or LSB first in LSB (SPI_CNTRL[10]). User can also select which edge of SPI clock to transmit/receive data in TX_NEG/RX_NEG (SPI_CNTRL[2:1]). Four SPI timing diagrams for master/slave operations and

the related settings are shown below.

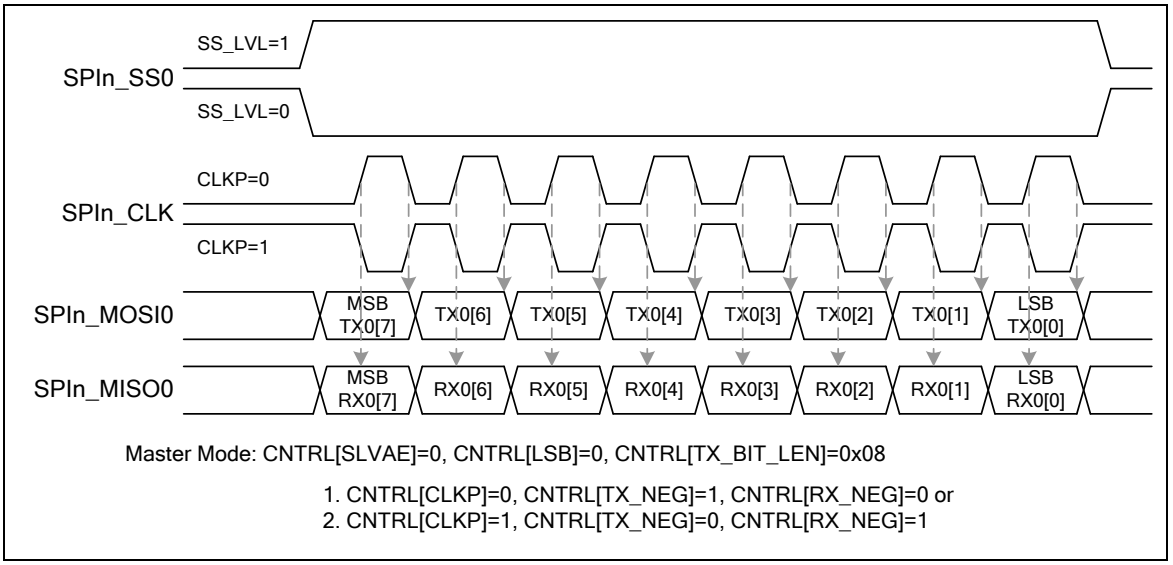


Figure 6.15-11 SPI Timing in Master Mode

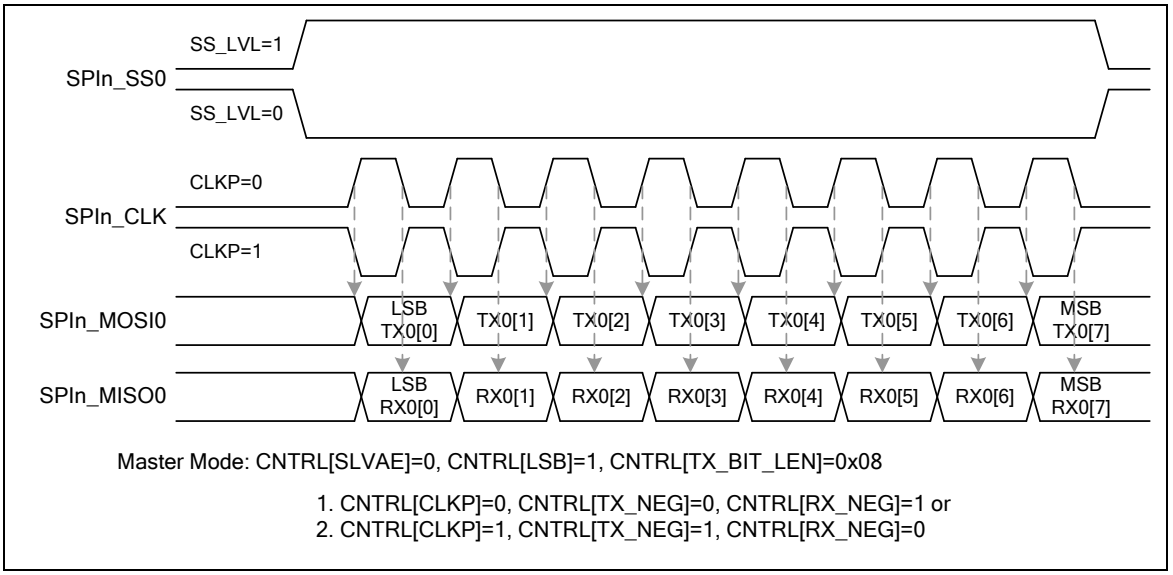


Figure 6.15-12 SPI Timing in Master Mode (Alternate Phase of SPI Bus Clock)

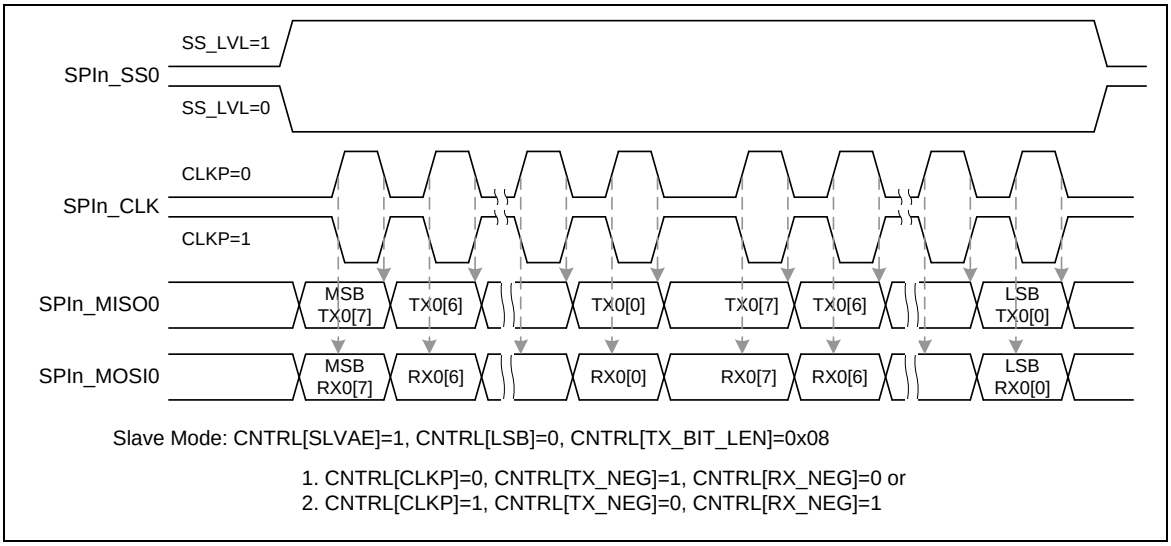


Figure 6.15-13 SPI Timing in Slave Mode

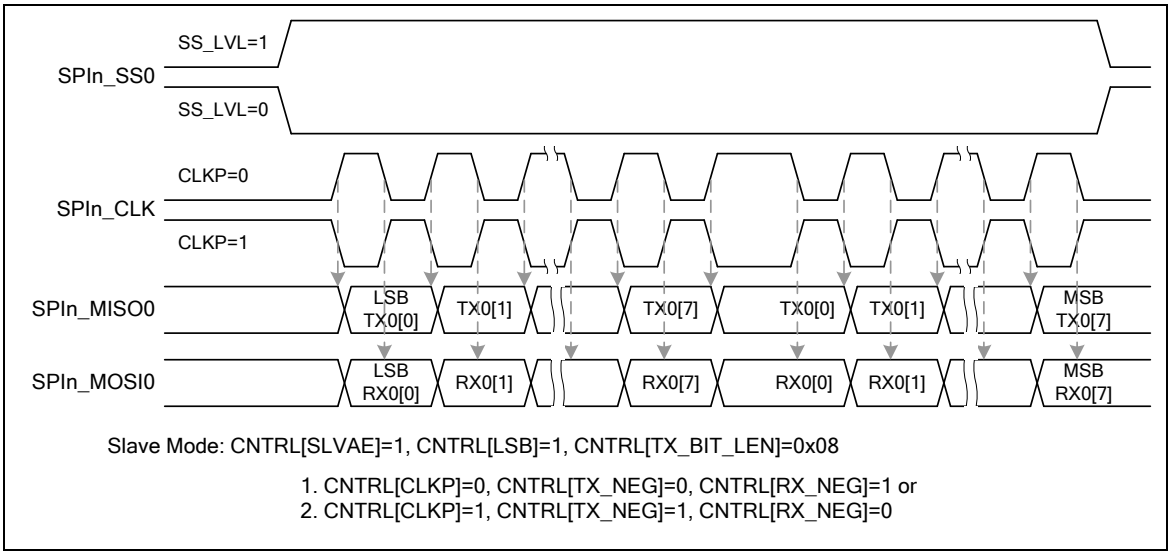


Figure 6.15-14 SPI Timing in Slave Mode (Alternate Phase of SPI Bus Clock)

6.15.7 Programming Examples

Example 1: The SPI controller is set as a master to access an off-chip slave device with the following specifications:

- Data bit is latched on positive edge of SPI clock.
- Data bit is driven on negative edge of SPI clock.
- Data is transferred from MSB first.
- SPI bus clock is idle at low state.
- Only one byte of data to be transmitted/received in a transaction.
- Uses the first SPI slave select pin to connect with an off-chip slave device. The slave select signal is active low.

The operation flow is as follows.

1) Set the DIVIDER (SPI_DIVIDER [7:0]) register to determine the output frequency of SPI clock.
2) Write the SPI_SSR register a proper value for the related settings of Master mode:
    1. Clear the Automatic Slave Selection bit, AUTOSS (SPI_SSR[3]), to 0.
    2. Select low level trigger output of slave select signal in the Slave Select Active Level bit, SS_LVL (SPI_SSR[2]), and Slave Select Level Trigger bit, SS_LTRIG (SPI_SSR[4]).
    3. Select slave select signal to be output active at the I/O pin by setting the Slave Select Register bit SSR[0] (SPI_SSR[0]) to active the off-chip slave device.
3) Write the related settings into the SPI_CNTRL register to control the SPI master actions
    1. Set this SPI controller as master device in SLAVE bit (SPI_CNTRL[18] = 0).
    2. Force the SPI clock idle state at low in CLKP bit (SPI_CNTRL[11] = 0).
    3. Select data transmitted at negative edge of SPI clock in TX_NEG bit (SPI_CNTRL[2] = 1).
    4. Select data latched at positive edge of SPI clock in RX_NEG bit (SPI_CNTRL[1] = 0).
    5. Set the bit length of word transfer as 8-bit in TX_BIT_LEN bit field. (SPI_CNTRL[7:3] = 0x08).
    6. Set MSB transfer first in MSB bit (SPI_CNTRL[10] = 0).
4) If this SPI master attempts to transmit (write) one byte data to the off-chip slave device, write the byte data that will be transmitted into the SPI_TX0 register.
5) If this SPI master just only attempts to receive (read) one byte data from the off-chip slave device and does not care what data will be transmitted, the SPI_TX0 register does not need to be updated by software.
6) Enable the GO_BUSY bit (SPI_CNTRL [0] = 1) to start the data transfer with the SPI interface.
7) Waiting for SPI interrupt (if the Interrupt Enable IE bit is set) or just polling the GO_BUSY bit till it is cleared to 0 by hardware automatically.
8) Read out the received one byte data from SPI_RX0[7:0].
9) Go to 4) to continue another data transfer or set SSR [0] to 0 to inactivate the off-chip slave

device.

Example 2: The SPI controller is set as a slave device and connects with an off-chip master device. The off-chip master device communicates with the on-chip SPI slave controller through the SPI interface with the following specifications:

- Data bit is latched on positive edge of SPI clock.
- Data bit is driven on negative edge of SPI clock.
- Data is transferred from LSB first.
- SPI bus clock is idle at high state.
- Only one byte of data to be transmitted/received in a transaction.
- Slave select signal is high level trigger.

The operation flow is as follows.

1) Write the SPI_SSR register a proper value for the related settings of Slave mode:

   Select high level and level trigger for the input of slave select signal by setting the Slave Select Active Level bit SS_LVL (SPI_SSR[2] = 1) and the Slave Select Level Trigger bit SS_LTRIG (SPI_SSR[4] = 1).

2) Write the related settings into the SPI_CNTRL register to control this SPI slave actions

   1. Set the SPI controller as slave device in SLAVE bit (SPI_CNTRL[18] = 1).
   2. Select the SPI clock idle state at high in CLKP bit (SPI_CNTRL[11] = 1).
   3. Select data transmitted at negative edge of SPI clock in TX_NEG bit (SPI_CNTRL[2] = 1).
   4. Select data latched at positive edge of SPI clock in RX_NEG bit (SPI_CNTRL[1] = 0).
   5. Set the bit length of word transfer as 8-bit in TX_BIT_LEN bit field (SPI_CNTRL[7:3] = 0x08).
   6. Set LSB transfer first in LSB bit (SPI_CNTRL[10] = 1).

3) If this SPI slave attempts to transmit (be read) one byte data to the off-chip master device, write the byte data that will be transmitted into the SPI_TX0 register.

4) If this SPI slave just only attempts to receive (be written) one byte data from the off-chip master device and does not care what data will be transmitted, the SPI_TX0 register does not need to be updated by software.

5) Enable the GO_BUSY bit (SPI_CNTRL[0] = 1) to wait for the slave select trigger input and SPI clock input from the off-chip master device to start the data transfer at the SPI interface.

6) Waiting for SPI interrupt (if the Interrupt Enable IE bit is set), or just polling the GO_BUSY bit till it is cleared to 0 by hardware automatically.

7) Read out the received one byte data from SPI_RX0[7:0].

8) Go to 3) to continue another data transfer or stop data transfer.

6.15.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI Base Address: **SPI0_BA = 0x4003_0000** **SPI1_BA = 0x4003_4000**				
SPI_CNTRL **n=0,1**	SPIn_BA+0x00	R/W	Control and Status Register	0x0500_3004
SPI_DIVIDER **n=0,1**	SPIn_BA+0x04	R/W	Clock Divider Register	0x0000_0000
SPI_SSR **n=0,1**	SPIn_BA+0x08	R/W	Slave Select Register	0x0000_0000
SPI_RX0 **n=0,1**	SPIn_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPI_TX0 **n=0,1**	SPIn_BA+0x20	W	Data Transmit Register 0	0x0000_0000
SPI_VARCLK **n=0,1**	SPIn_BA+0x34	R/W	Variable Clock Pattern Register	0x007F_FF87
SPI_DMA **n=0,1**	SPIn_BA+0x38	R/W	SPI DMA Control Register	0x0000_0000
SPI_CNTRL2 **n=0,1**	SPIn_BA+0x3C	R/W	Control and Status Register 2	0x0000_1000
SPI_FIFO_CTL **n=0,1**	SPIn_BA+0x40	R/W	SPI FIFO Control Register	0x4400_0000
SPI_STATUS **n=0,1**	SPIn_BA+0x44	R/W	SPI Status Register	0x0500_0000

6.15.9 Register Description

SPI Control and Status Register (SPI_CNTRL)

Register	Offset	R/W	Description	Reset Value
SPI_CNTRL	SPIn_BA+0x00	R/W	Control and Status Register	0x0500_3004

31	30	29	28	27	26	25	24
Reserved				TX_FULL	TX_EMPTY	RX_FULL	RX_EMPTY
23	22	21	20	19	18	17	16
VARCLK_EN	Reserved	FIFO	Reserved	REORDER	SLAVE	IE	IF
15	14	13	12	11	10	9	8
SP_CYCLE				CLKP	LSB	Reserved	
7	6	5	4	3	2	1	0
TX_BIT_LEN					TX_NEG	RX_NEG	GO_BUSY

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**TX_FULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** It is a mutual mirror bit of SPI_STATUS[27]. 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.
[26]	**TX_EMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** It is a mutual mirror bit of SPI_STATUS[26]. 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[25]	**RX_FULL**	**Receive FIFO Buffer Full Indicator (Read Only)** It is a mutual mirror bit of SPI_STATUS[25]. 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[24]	**RX_EMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** It is a mutual mirror bit of SPI_CNTRL[24]. 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[23]	**VARCLK_EN**	**Variable Clock Enable Bit (Master Only)** 0 = SPI clock output frequency is fixed and decided only by the value of DIVIDER. 1 = SPI clock output frequency is variable. The output frequency is decided by the value of VARCLK, DIVIDER, and DIVIDER2. **Note:** When this VARCLK_EN bit is set to 1, the setting of TX_BIT_LEN must be programmed as 0x10 (16-bit mode).
[22]	**Reserved**	Reserved.
[21]	**FIFO**	**FIFO Mode EnableBit**

		0 = FIFO mode Disabled. 1 = FIFO mode Enabled. **Note1:** Before enabling FIFO mode, the other related settings should be set in advance. **Note2:** In Master mode, if the FIFO mode is enabled, the GO_BUSY bit will be set to 1 automatically after writing data to the transmit FIFO buffer; the GO_BUSY bit will be cleared to 0 automatically when the SPI controller is in idle. If all data stored at transmit FIFO buffer are sent out, the TX_EMPTY bit will be set to 1 and the GO_BUSY bit will be cleared to 0. **Note3:** After clearing this bit to 0, user must wait for at least 2 peripheral clock periods before setting this bit to 1 again.
[20]	**Reserved**	Reserved.
[19]	**REORDER**	**Byte Reorder Function EnableBit** 0 = Byte Reorder function Disabled. 1 = Byte Reorder function Enabled. A byte suspend interval will be inserted among each byte. The period of the byte suspend interval depends on the setting of SP_CYCLE. **Note1:** Byte Reorder function is only available if TX_BIT_LEN is defined as 16, 24, and 32 bits. **Note2:** In Slave mode with level-trigger configuration, the slave select pin must be kept at active state during the byte suspend interval. **Note3:** The Byte Reorder function is not supported when the variable bus clock function or Dual I/O mode is enabled.
[18]	**SLAVE**	**Slave Mode EnableBit** 0 = Master mode. 1 = Slave mode.
[17]	**IE**	**Unit Transfer Interrupt EnableBit** 0 = SPI unit transfer interrupt Disabled. 1 = SPI unit transfer interrupt Enabled.
[16]	**IF**	**Unit Transfer Interrupt Flag** 0 = No transaction has been finished since this bit was cleared to 0. 1 = SPI controller has finished one unit transfer. **Note:** This bit will be cleared by writing 1 to itself.
[15:12]	**SP_CYCLE**	**Suspend Interval (Master Only)** The four bits provide configurable suspend interval between two successive transmit/receive transaction in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. (SP_CYCLE[3:0] + 0.5) * period of SPI bus clock cycle Example: SP_CYCLE = 0x0 …. 0.5 SPI bus clock cycle. SP_CYCLE = 0x1 …. 1.5 SPI bus clock cycles. …… SP_CYCLE = 0xE …. 14.5 SPI bus clock cycles. SP_CYCLE = 0xF …. 15.5 SPI bus clock cycles. If the variable clock function is enabled and the transmit FIFO buffer is not empty, the minimum period of suspend interval between the successive transactions is (6.5 + SP_CYCLE) * SPI bus clock cycle.
[11]	**CLKP**	**Clock Polarity** 0 = SPI bus clock is idle low. 1 = SPI bus clock is idle high.

[10]	**LSB**	**Send LSB First** 0 = The MSB, which bit of transmit/receive register depends on the setting of TX_BIT_LEN, is transmitted/received first. 1 = The LSB, bit 0 of the SPI TX0 register, is sent first to the SPI data output pin, and the first bit received from the SPI data input pin will be put in the LSB position of the RX register (bit 0 of SPI_RX0).
[9:8]	**Reserved**	Reserved.
[7:3]	**TX_BIT_LEN**	**Transmit Bit Length** This field specifies how many bits can be transmitted / received in one transaction. The minimum bit length is 8 bits and can up to 32 bits. TX_BIT_LEN = 0x08 …. 8 bits. TX_BIT_LEN = 0x09 …. 9 bits. …… TX_BIT_LEN = 0x1F …. 31 bits. TX_BIT_LEN = 0x00 …. 32 bits.
[2]	**TX_NEG**	**Transmit On Negative Edge** 0 = Transmitted data output signal is changed on the rising edge of SPI bus clock. 1 = Transmitted data output signal is changed on the falling edge of SPI bus clock.
[1]	**RX_NEG**	**Receive On Negative Edge** 0 = Received data input signal is latched on the rising edge of SPI bus clock. 1 = Received data input signal is latched on the falling edge of SPI bus clock.
[0]	**GO_BUSY**	**SPI Transfer Control Bit And Busy Status** 0 = Data transfer stopped. 1 = In Master mode, writing 1 to this bit to start the SPI data transfer; in Slave mode, writing 1 to this bit indicates that the slave is ready to communicate with a master. If FIFO mode is disabled, during the data transfer, this bit keeps the value of 1. As the transfer is finished, this bit will be cleared automatically. Software can read this bit to check if the SPI is in busy status. In FIFO mode, this bit will be controlled by hardware. Software should not modify this bit. In Slave mode, this bit always returns 1 when this register is read by software. In Master mode, this bit reflects the busy or idle status of SPI. **Note1:** When FIFO mode is disabled, all configurations should be set before writing 1 to this GO_BUSY bit. **Note2:** When FIFO mode is disabled and the software uses TX or RX PDMA function to transfer data, this bit will be cleared after the PDMA finishes the data transfer.

SPI Divider Register (SPI_DIVIDER)

Register	Offset	R/W	Description	Reset Value
SPI_DIVIDER	SPIn_BA+0x04	R/W	Clock Divider Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
DIVIDER2							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DIVIDER							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**DIVIDER2**	**Clock Divider 2 Register (Master Only)** The value in this field is the 2nd frequency divider for generating the second clock of the variable clock function. The frequency is obtained according to the following equation: $f_{clock2} = \frac{f_{spi\_eclk}}{(DIVIDER2+1)*2}$ If the VARCLK_EN bit is cleared to 0, this setting is unmeaning.
[15:8]	**Reserved**	Reserved.
[7:0]	**DIVIDER**	**Clock Divider 1 Register** The value in this field is the frequency divider for generating the SPI peripheral clock, f_{spi_eclk}, and the SPI bus clock of SPI master. The frequency is obtained according to the following equation. If the bit of BCn, SPI_CNTRL2[31], is set to 0, $f_{spi\_eclk} = \frac{f_{system\_clock}}{(DIVIDER+1)*2}$ else if BCn is set to 1, $f_{spi\_eclk} = \frac{f_{spi\_clock\_src}}{(DIVIDER+1)}$ where $f_{spi\_clock\_src}$ is the SPI peripheral clock source, which is defined in the CLKSEL1 register.

SPI Slave Select Register (SPI_SSR)

Register	Offset	R/W	Description	Reset Value
SPI_SSR	SPIn_BA+0x08	R/W	Slave 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		LTRIG_FLAG	SS_LTRIG	AUTOSS	SS_LVL	Reserved	SSR

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**LTRIG_FLAG**	**Level Trigger Accomplish Flag** In Slave mode, this bit indicates whether the received bit number meets the requirement or not after the current transaction done. 0 = Transferred bit length of one transaction does not meet the specified requirement. 1 = Transferred bit length meets the specified requirement which defined in TX_BIT_LEN. **Note:** This bit is READ only. As the GO_BUSY bit is set to 1 by software, the LTRIG_FLAG will be cleared to 0 after 4 SPI peripheral clock periods plus 1 system clock period. In FIFO mode, this bit has no meaning.
[4]	**SS_LTRIG**	**Slave Select Level Trigger Enable Bit (Slave Only)** 0 = Slave select signal is edge-trigger. This is the default value. The SS_LVL bit decides the signal is active after a falling-edge or rising-edge. 1 = Slave select signal is level-trigger. The SS_LVL bit decides the signal is active low or active high.
[3]	**AUTOSS**	**Automatic Slave Select Function Enable Bit (Master Only)** 0 = If this bit is cleared, slave select signals will be asserted/de-asserted by setting /clearing the corresponding bits of SPI_SSR[1:0]. 1 = If this bit is set, SPIn_SPISS0 signals will be generated automatically. It means that device/slave select signal, which is set in SPI_SSR[1:0], will be asserted by the SPI controller when transmit/receive is started, and will be de-asserted after each transmit/receive is finished.
[2]	**SS_LVL**	**Slave Select Active Level** This bit defines the active status of slave select signal (SPIn_SPISS0). 0 = The slave select signal SPIn_SPISS0 is active on low-level/falling-edge. 1 = The slave select signal SPIn_SPISS0 is active on high-level/rising-edge.
[1]	**Reserved**	Reserved.
[0]	**SSR**	**Slave Select Control Bits (Master Only)** If AUTOSS bit is cleared, writing 1 to any bit of this field sets the SPIn_SPISS0 line to an active state and writing 0 sets the line back to inactive state. If the AUTOSS bit is set, writing 0 to this field will keep the SPIn_SPISS0 line at inactive

		state; writing 1 to this field will select SPIn_SPISS0 line to be automatically driven to active state for the duration of the transmit/receive, and will be driven to inactive state for the rest of the time. The active state of SPIn_SPISS0 is specified in SS_LVL. **Note:** SPIn_SPISS0 is defined as the slave select input in Slave mode.

SPI Data Receive Register (SPI_RX)

Register	Offset	R/W	Description	Reset Value
SPI_RX0	SPIn_BA+0x10	R	Data Receive Register 0	0x0000_0000

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

Bits	Description	
[31:0]	**RX**	**Data Receive Register** The data receive register holds the datum received from SPI data input pin. If FIFO mode is disabled, the last received data can be accessed through software by reading this register. If the FIFO bit is set as 1 and the RX_EMPTY bit, SPI_CNTRL[24] or SPI_STATUS[24], is not set to 1, the receive FIFO buffer can be accessed through software by reading this register. This is a read-only register.

SPI Data Transmit Register (SPI_TX)

Register	Offset	R/W	Description	Reset Value
SPI_TX0	SPIn_BA+0x20	W	Data Transmit Register 0	0x0000_0000

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

Bits	Description	
[31:0]	**TX**	**Data Transmit Register** The data transmit registers hold the data to be transmitted in the next transfer. The number of valid bits depends on the setting of transmit bit length field of the SPI_CNTRL register. For example, if TX_BIT_LEN is set to 0x08, the bits TX[7:0] will be transmitted in next transfer. If TX_BIT_LEN is set to 0x00, the SPI controller will perform a 32-bit transfer. **Note 1:** When the SPI controller is configured as a slave device and FIFO mode is disabled, if the SPI controller attempts to transmit data to a master, the transmit data register should be updated by software before setting the GO_BUSY bit to 1. **Note 2**: In Master mode, SPI controller will start to transfer after 5 peripheral clock cycles since user wrote to this register.

SPI Variable Clock Pattern Register (SPI_VARCLK)

Register	Offset	R/W	Description	Reset Value
SPI_VARCLK	SPIn_BA+0x34	R/W	Variable Clock Pattern Register	0x007F_FF87

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

Bits	Description	
[31:0]	VARCLK	**Variable Clock Pattern** This register defines the clock pattern of the SPI transfer. If the variable clock function is disabled, this setting is unmeaning. Refer to the “Variable Clock Function” paragraph for more detail description.

SPI DMA Control Register (SPI_DMA)

Register	Offset	R/W	Description	Reset Value
SPI_DMA	SPIn_BA+0x38	R/W	SPI DMA 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					PDMA _RST	RX_DMA_GO	TX_DMA_GO

Bits	Description	
[31:2]	**Reserved**	Reserved.
[2]	**PDMA_RST**	**PDMA Reset** 0 = No effect. 1 = Reset the PDMA control logic of the SPI controller. This bit will be cleared to 0 automatically.
[1]	**RX_DMA_GO**	**Receive DMA Start** Setting this bit to 1 will start the receive PDMA process. The SPI controller will issue request to PDMA controller automatically when the SPI receive buffer is not empty. This bit will be cleared to 0 by hardware automatically after PDMA transfer is done. If the software uses the receive PDMA function to access the received data of SPI and does not use the transmit PDMA function, the GO_BUSY bit should be set by software. Enabling FIFO mode is recommended if the software uses more than one PDMA channel to transfer data. In Slave mode and when FIFO mode is disabled, if the software only uses one PDMA channel for SPI receive PDMA function and the other PDMA channels are not in use, the minimal suspend interval between two successive transactions must be larger than (9 SPI slave peripheral clock periods + 4 APB clock periods) for Edge-trigger mode or (9.5 SPI slave peripheral clock periods + 4 APB clock periods) for Level-trigger mode.
[0]	**TX_DMA_GO**	**Transmit DMA Start** Setting this bit to 1 will start the transmit PDMA process. SPI controller will issue request to PDMA controller automatically. Hardware will clear this bit to 0 automatically after PDMA transfer done. If the SPI transmit PDMA function is used to transfer data, the GO_BUSY bit should not be set to 1 by software. The PDMA control logic of SPI controller will set it automatically whenever necessary. In Slave mode and when FIFO mode is disabled, the minimal suspend interval between two successive transactions must be larger than (8 SPI clock periods + 14 APB clock periods) for edge-trigger mode or (9.5 SPI clock periods + 14 APB clock periods) for level-trigger mode. If the 2-bit Transfer mode is enabled, additional 18 APB clock periods for the above conditions is required.

SPI Control and Status Register 2 (SPI_CNTRL2)

Register	Offset	R/W	Description	Reset Value
SPI_CNTRL2	SPIn_BA+0x3C	R/W	Control and Status Register 2	0x0000_1000

31	30	29	28	27	26	25	24
BCn	Reserved						
23	22	21	20	19	18	17	16
Reserved							SS_INT_OPT
15	14	13	12	11	10	9	8
Reserved		DUAL_ IO_EN	DUAL_ IO_DIR	SLV_START _INTSTS	SSTA_ INTEN	SLV_ABORT	NOSLVSEL
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31]	**BCn**	**SPI Peripheral Clock Backward Compatible Option** 0 = Backward compatible clock configuration. 1 = Clock configuration is not backward compatible. Refer to the description of SPI_DIVIDER register for details.
[30:17]	**Reserved**	Reserved.
[16]	**SS_INT_OPT**	**Slave Select Inactive Interrupt Option** This setting is only available if the SPI controller is configured as level trigger slave device. 0 = As the slave select signal goes to inactive level, the IF bit will NOT be set to 1. 1 = As the slave select signal goes to inactive level, the IF bit will be set to 1.
[15:14]	**Reserved**	Reserved.
[13]	**DUAL_IO_EN**	**Dual I/O Mode EnableBit** 0 = Dual I/O mode Disabled. 1 = Dual I/O mode Enabled.
[12]	**DUAL_IO_DIR**	**Dual I/O Mode Direction Control** 0 = Dual Input mode. 1 = Dual Output mode.
[11]	**SLV_START_INTSTS**	**Slave 3-Wire Mode Start Interrupt Status** This bit indicates if a transaction has started in Slave 3-wire mode. It is a mutual mirror bit of SPI_STATUS[11]. 0 = Slave has not detected any SPI clock transition since the SSTA_INTEN bit was set to 1. 1 = A transaction has started in Slave 3-wire mode. It will be cleared automatically when a transaction is done or by writing 1 to this bit.
[10]	**SSTA_INTEN**	**Slave 3-Wire Mode Start Interrupt EnableBit** Used to enable interrupt when the transfer has started in Slave 3-wire mode. If there is no transfer done interrupt over the time period which is defined by user

		after the transfer start, the user can set the SLV_ABORT bit to force the transfer done. 0 = Transaction start interrupt Disabled. 1 = Transaction start interrupt Enabled. It will be cleared to 0 as the current transfer is done or the SLV_START_INTSTS bit is cleared.
[9]	**SLV_ABORT**	**Slave 3-Wire Mode Abort Control** In normal operation, there is an interrupt event when the received data meet the required bits which defined in TX_BIT_LEN. If the received bits are less than the requirement and there is no more SPI clock input over the one transfer time in Slave 3-wire mode, the user can set this bit to force the current transfer done and then the user can get a transfer done interrupt event. **Note:** This bit will be cleared to 0 automatically by hardware after it is set to 1 by software.
[8]	**NOSLVSEL**	**Slave 3-Wire Mode Enable Bit** This is used to ignore the slave select signal in Slave mode. The SPI controller can work with 3-wire interface including SPIn_CLK, SPIn_MISO, and SPIn_MOSI. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface. **Note:** In Slave 3-wire mode, the SS_LTRIG, SPI_SSR[4] will be set as 1 automatically.
[7:0]	**Reserved**	Reserved.

SPI FIFO Control Register (SPI_FIFO_CTL)

Register	Offset	R/W	Description	Reset Value
SPI_FIFO_CTL	SPIn_BA+0x40	R/W	SPI FIFO Control Register	0x4400_0000

31	30	29	28	27	26	25	24
Reserved	TX_THRESHOLD			Reserved	RX_THRESHOLD		
23	22	21	20	19	18	17	16
Reserved		TIMEOUT_INTEN	Reserved				
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RXOV_INTEN	Reserved		TX_INTEN	RX_INTEN	TX_CLR	RX_CLR

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TX_THRESHOLD**	**Transmit FIFO Threshold** If the valid data count of the transmit FIFO buffer is less than or equal to the TX_THRESHOLD setting, the TX_INTSTS bit will be set to 1, else the TX_INTSTS bit will be cleared to 0.
[27]	**Reserved**	Reserved.
[26:24]	**RX_THRESHOLD**	**Receive FIFO Threshold** If the valid data count of the receive FIFO buffer is larger than the RX_THRESHOLD setting, the RX_INTSTS bit will be set to 1, else the RX_INTSTS bit will be cleared to 0.
[23:22]	**Reserved**	Reserved.
[21]	**TIMEOUT_INTEN**	**Receive FIFO Time-Out Interrupt Enable Bit** 0 = Time-out interrupt Disabled. 1 = Time-out interrupt Enabled.
[20:7]	**Reserved**	Reserved.
[6]	**RXOV_INTEN**	**Receive FIFO Overrun Interrupt Enable Bit** 0 = Receive FIFO overrun interrupt Disabled. 1 = Receive FIFO overrun interrupt Enabled.
[5:4]	**Reserved**	Reserved.
[3]	**TX_INTEN**	**Transmit Threshold Interrupt Enable Bit** 0 = TX threshold interrupt Disabled. 1 = TX threshold interrupt Enabled.
[2]	**RX_INTEN**	**Receive Threshold Interrupt Enable Bit** 0 = RX threshold interrupt Disabled. 1 = RX threshold interrupt Enabled.

[1]	**TX_CLR**	**Clear Transmit FIFO Buffer** 0 = No effect. 1 = Clear transmit FIFO buffer. The TX_FULL flag will be cleared to 0 and the TX_EMPTY flag will be set to 1. This bit will be cleared to 0 by hardware after it is set to 1 by software.
[0]	**RX_CLR**	**Clear Receive FIFO Buffer** 0 = No effect. 1 = Clear receive FIFO buffer. The RX_FULL flag will be cleared to 0 and the RX_EMPTY flag will be set to 1. This bit will be cleared to 0 by hardware after it is set to 1 by software.

SPI Status Register (SPI_STATUS)

Register	Offset	R/W	Description	Reset Value
SPI_STATUS	SPIn_BA+0x44	R/W	SPI Status Register	0x0500_0000

31	30	29	28	27	26	25	24
TX_FIFO_COUNT				TX_FULL	TX_EMPTY	RX_FULL	RX_EMPTY
23	22	21	20	19	18	17	16
Reserved			TIMEOUT	Reserved			IF
15	14	13	12	11	10	9	8
RX_FIFO_COUNT				SLV_START _INTSTS	Reserved		
7	6	5	4	3	2	1	0
Reserved			TX_INTSTS	Reserved	RX_ OVERRUN	Reserved	RX_INTSTS

Bits	Description	
[31:28]	TX_FIFO_COUNT	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27]	TX_FULL	**Transmit FIFO Buffer Full Indicator (Read Only)** It is a mutual mirror bit of SPI_CNTRL[27]. 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.
[26]	TX_EMPTY	**Transmit FIFO Buffer Empty Indicator (Read Only)** It is a mutual mirror bit of SPI_CNTRL[26]. 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[25]	RX_FULL	**Receive FIFO Buffer Empty Indicator (Read Only)** It is a mutual mirror bit of SPI_CNTRL[24]. 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[24]	RX_EMPTY	**Receive FIFO Buffer Empty Indicator (Read Only)** It is a mutual mirror bit of SPI_CNTRL[24]. 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[23:21]	Reserved	Reserved.
[20]	TIMEOUT	**Time-Out Interrupt Flag** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 SPI clock period in Master mode or over 576 SPI peripheral clock period in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. **Note:** This bit will be cleared by writing 1 to itself.

[19:17]	**Reserved**	Reserved.
[16]	**IF**	**SPI Unit Transfer Interrupt Flag** It is a mutual mirror bit of SPI_CNTRL[16]. 0 = No transaction has been finished since this bit was cleared to 0. 1 = SPI controller has finished one unit transfer. **Note:** This bit will be cleared by writing 1 to itself.
[15:12]	**RX_FIFO_COUNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[11]	**SLV_START_INTSTS**	**Slave Start Interrupt Status** It is used to dedicate if a transaction has started in Slave 3-wire mode. It is a mutual mirror bit of SPI_CNTRL2[11]. 0 = Slave has not detected any SPI clock transition since the SSTA_INTEN bit was set to 1. 1 = A transaction has started in Slave 3-wire mode. It will be cleared as a transaction is done or by writing 1 to this bit.
[10:5]	**Reserved**	Reserved.
[4]	**TX_INTSTS**	**Transmit FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TX_THRESHOLD. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TX_THRESHOLD. **Note:** If TX_INTEN = 1 and TX_INTSTS = 1, the SPI controller will generate a SPI interrupt request.
[3]	**Reserved**	Reserved.
[2]	**RX_OVERRUN**	**Receive FIFO Overrun Status** When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1. **Note:** This bit will be cleared by writing 1 to itself.
[1]	**Reserved**	Reserved.
[0]	**RX_INTSTS**	**Receive FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the Rx FIFO buffer is smaller than or equal to the setting value of RX_THRESHOLD. 1 = The valid data count within the receive FIFO buffer is larger than the setting value of RX_THRESHOLD. **Note:** If RX_INTEN = 1 and RX_INTSTS = 1, the SPI controller will generate a SPI interrupt request.

6.16 USB Device Controller (USBD)

6.16.1 Overview

There is one set of USB 2.0 full-speed device controller and transceiver in this device. It is compliant with USB 2.0 full-speed device specification and supports control/bulk/interrupt/ isochronous transfer types, and use High Internal RC Oscillator (HIRC48M) obtain to crystal-less option.

In this device controller, there are two main interfaces: the APB bus and USB bus which comes from the USB PHY transceiver. For the APB bus, the CPU can program control registers through it. There are 512 bytes internal SRAM as data buffer in this controller. For IN or OUT transfer, it is necessary to write data to SRAM or read data from SRAM through the APB interface or SIE. User needs to set the effective starting address of SRAM for each endpoint buffer through “buffer segmentation register (USB_BUFSEGx)”.

There are 8 endpoints in this controller. Each of the endpoint can be configured as IN or OUT endpoint. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. The block of “Endpoint Control” is also used to manage the data sequential synchronization, endpoint states, current start address, transaction status, and data buffer status for each endpoint.

There are four different interrupt events in this controller. They are the wake-up function, device plug-in or plug-out event, USB events, and BUS events. Any event will cause an interrupt, and users just need to check the related event flags in interrupt event status register (USB_INTSTS) to acknowledge what kind of interrupt occurring, and then check the related USB Endpoint Status Register (USB_EPSTS) to acknowledge what kind of event occurring in this endpoint.

A software-disconnect function is also supported for this USB controller. It is used to simulate the disconnection of this device from the host. If DRVSE0 (USB_DRVSE0[0]) is set to 1, the USB controller will force the output of USB_D+ and USB_D- to level low. After DRVSE0 bit is cleared to 0, host will enumerate the USB device again.

Please refer to *Universal Serial Bus Specification Revision 1.1* for details.

6.16.2 Features

- Compliant with USB 2.0 Full-Speed specification
- Provides 1 interrupt vector with 4 different interrupt events (WAKEUP, FLDET, USB and BUS)
- Supports Control/Bulk/Interrupt/Isochronous transfer type
- Supports suspend function when no bus activity existing for 3 ms
- Provides 8 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 512 bytes buffer size
- Provides remote wake-up capability
- Supports Crystal-less

6.16.3 Block Diagram

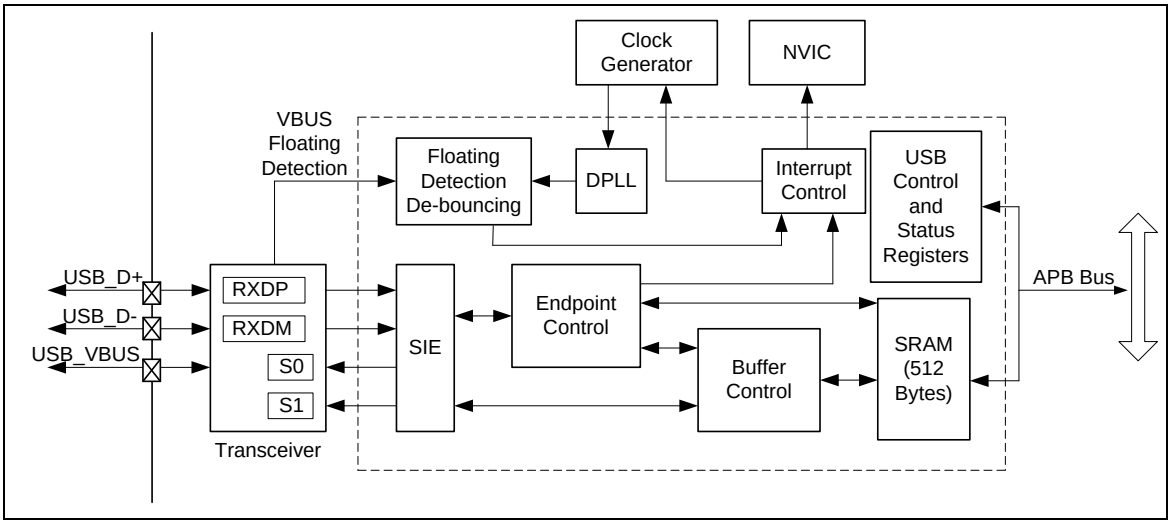


Figure 6.16-1 USB Device Block Diagram

6.16.4 Basic Configuration

USBD clock source is derived from PLL. User has to set the PLL related configurations before USB device controller is enabled. Set the USBD_EN (APBCLK[27]) bit to enable USBD clock and 4-bit pre-scaler USB_N (CLKDIV[7:4]) to generate the proper USBD clock rate.

6.16.5 Functional Description

6.16.5.1 SIE (Serial Interface Engine)

The SIE is the front-end of the device controller and handles most of the USB packet protocol. The SIE typically comprehends signaling up to the transaction level. The functions that it handles could include:

- Packet recognition, transaction sequencing
- SOF, EOP, RESET, RESUME signal detection/generation
- Clock/Data separation
- NRZI Data encoding/decoding and bit-stuffing
- CRC generation and checking (for Token and Data)
- Packet ID (PID) generation and checking/ decoding
- Serial-Parallel/ Parallel-Serial conversion

6.16.5.2 Endpoint Control

There are 8 endpoints in this controller. Each of the endpoint can be configured as Control, Bulk, Interrupt, or Isochronous transfer type. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. It is also used to manage the data sequential synchronization, endpoint state control, current endpoint start address, current transaction status, and data buffer status in each endpoint.

6.16.5.3 Digital Phase Lock Loop (DPLL)

The bit rate of USB data is 12 MHz. The DPLL uses the 48 MHz which comes from the clock controller to lock the input data RXDP and RXDM. The 12 MHz bit rate clock is also converted from DPLL.

6.16.5.4 Floating Detection De-bouncing

A USB device may be plugged-in or plugged-out from the USB host. To monitor the state of a USB device when it is detached from the USB host, the device controller provides hardware de-bouncing for USB floating detection interrupt to avoid bounce problems on USB plug-in or unplug. Floating detection interrupt appears about 10 ms later than USB plug-in or plug-out. User can acknowledge USB plug-in/plug-out by reading USB_FLDET register. The FLDET flag represents the current state of USB_VBUS without de-bouncing. If the FLDET flag is 1, it means the USB cable is plugged-in. If user polls the flag to check USB state, software de-bouncing must be added if needed.

6.16.5.5 Interrupt

This USB provides 1 interrupt vector with 4 interrupt events (WAKE-UP, FLDET, USB and BUS). The WAKE-UP event is used to wake-up the system clock when Power-down mode is enabled. (The power mode function is defined in system Power-down control register, PWRCON). The FLDET event is used for USB plug-in or unplug. The USB event notifies users of some USB requests, such as IN ACK, OUT ACK., and the BUS event notifies users of some bus events, such as suspend and, resume. The related bits must be set in the interrupt enable register (USB_INTEN) of USB Device Controller to enable USB interrupts.

Wake-up interrupt is only present when the chip enters Power- down mode and then wake-up event had happened. After the chip enters Power-down mode, any change on USB_VBUS, USB_D+ and USB_D- can wake up this chip if the USB wake-up function is enabled. If this change is not intentionally, no interrupt but wake-up interrupt will occur. After USB wake-up, this interrupt will occur when no other USB interrupt events are presented for more than 20ms. The following figure is the control flow of wake-up interrupt.

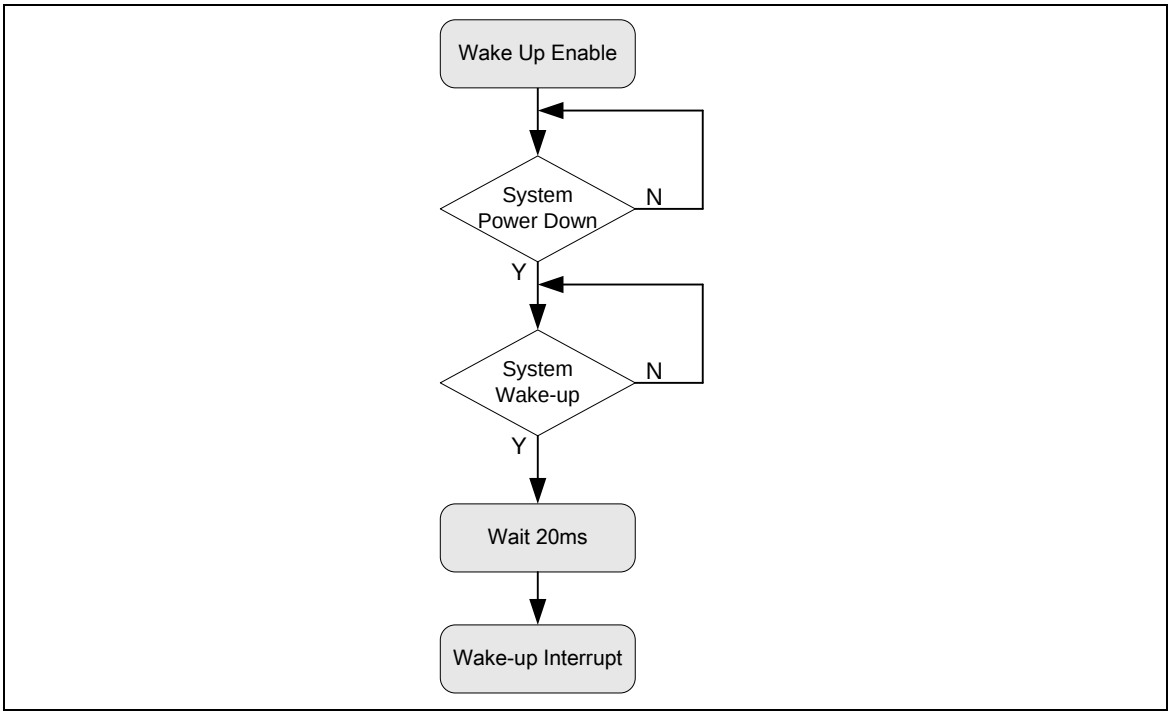


Figure 6.16-2 Wake-up Interrupt Operation Flow

USB interrupt is used to notify users of any USB event on the bus, and user can read EPSTS (USB_EPSTS[31:8]) and EPEVT7~0 (USB_INTSTS[23:16]) to take necessary responses.

Same as USB interrupt, BUS interrupt notifies users of some bus events, like USB reset, suspend, time-out, and resume. A user can read USB_ATTR to acknowledge bus events.

6.16.5.6 Power Saving

The USB turns off PHY transceiver automatically to save power while this chip enters Power-down mode. User can write 0 into USB_ATTR[4] to disable PHY under special circumstances like suspend to save power.

6.16.5.7 Buffer Control

There is 512 bytes SRAM in the controller and the 8 endpoints share this buffer. User shall configure each endpoint's effective starting address in the buffer segmentation register before the USB function active. The "Buffer Control" block is used to control each endpoint's effective starting address and its SRAM size is defined in the USB_MXPLDx register.

Figure 6.16-3 depicts the starting address for each endpoint according the content of USB_BUFSEGx and USB_MXPLDx registers. If the USB_BUFSEG0 is programmed as 0x08h and USB_MXPLD0 is set as 0x40h, the SRAM size of endpoint 0 is start from USBD_BA+0x108h and end in USBD_BA+0x148h. (**Note:** The USB SRAM base is USBD_BA+0x100h).

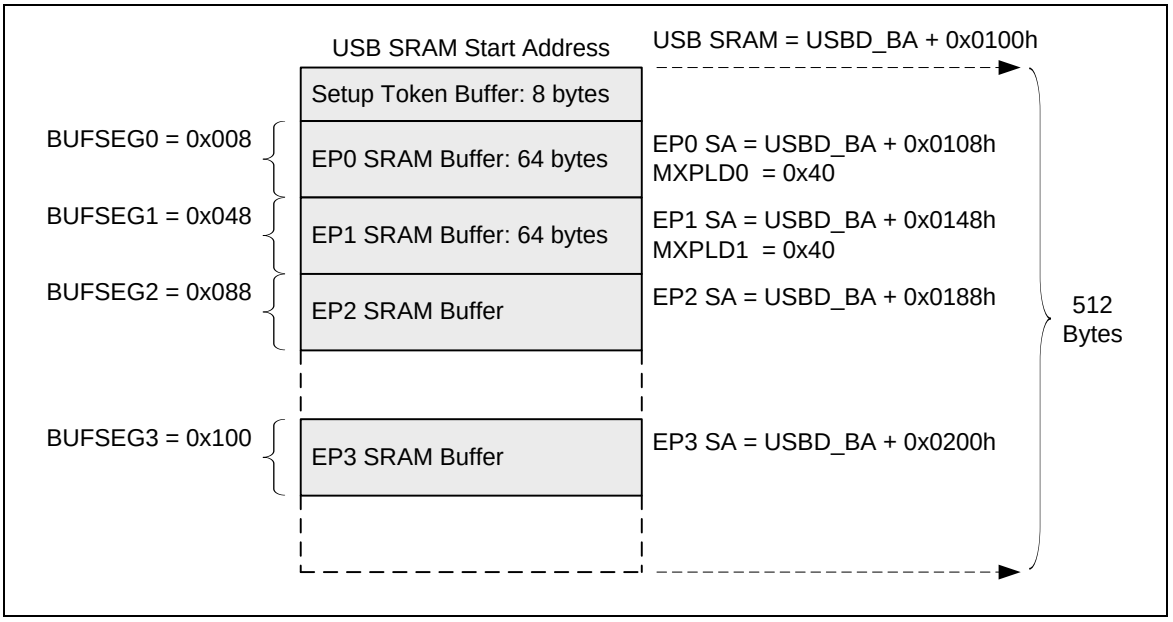


Figure 6.16-3 Endpoint SRAM Structure

6.16.5.8 Handling Transactions with USB Device Peripheral

User can use interrupt or poll USB_INTSTS to monitor the USB transactions. When transactions occur, USB_INTSTS will be set by hardware and send an interrupt request to CPU (if related interrupt enabled), or user can poll USB_INTSTS to get these events without interrupt. The following is the control flow with interrupt enabled.

When USB host has requested data from a device controller, user needs to prepare related data in the specified endpoint buffer in advance. After buffering the required data, user needs to write the actual data length in the specified USB_MXPLDx register. Once this register is written, the internal signal “In_Rdy” will be asserted and the buffering data will be transmitted immediately after receiving associated IN token from Host. Note that after transferring the specified data, the signal “In_Rdy” will be de-asserted automatically by hardware.

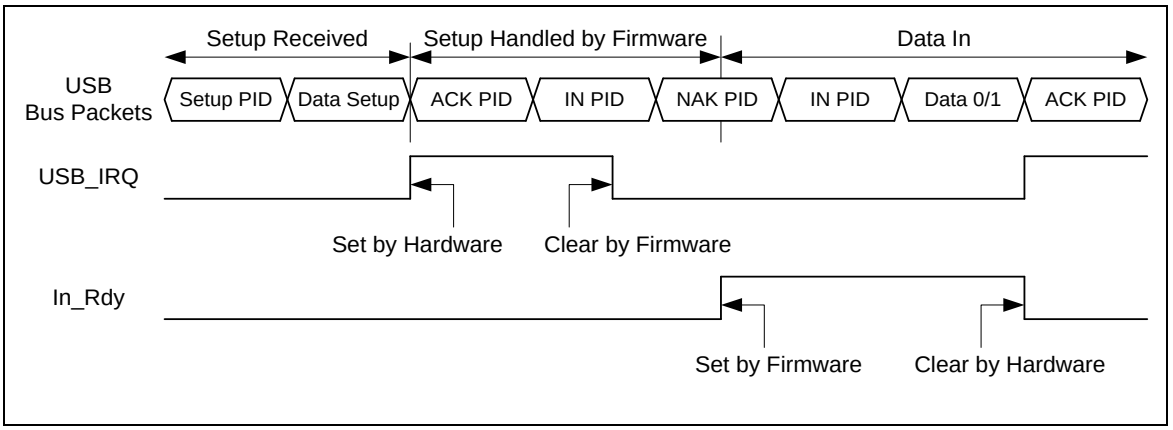


Figure 6.16-4 Setup Transaction Followed by Data IN Transaction

Alternatively, when USB host wants to transmit data to the OUT endpoint in the device controller, hardware will buffer these data to the specified endpoint buffer. After this transaction is completed, hardware will record the data length in specified USB_MXPLDx register and de-assert the internal signal “Out_Rdy”. This will avoid hardware accepting next transaction until user moves out the current data in the related endpoint buffer. Once users have processed this transaction, the specified USB_MXPLDx register needs to be written by firmware to assert the signal “Out_Rdy” again to accept the next transaction.

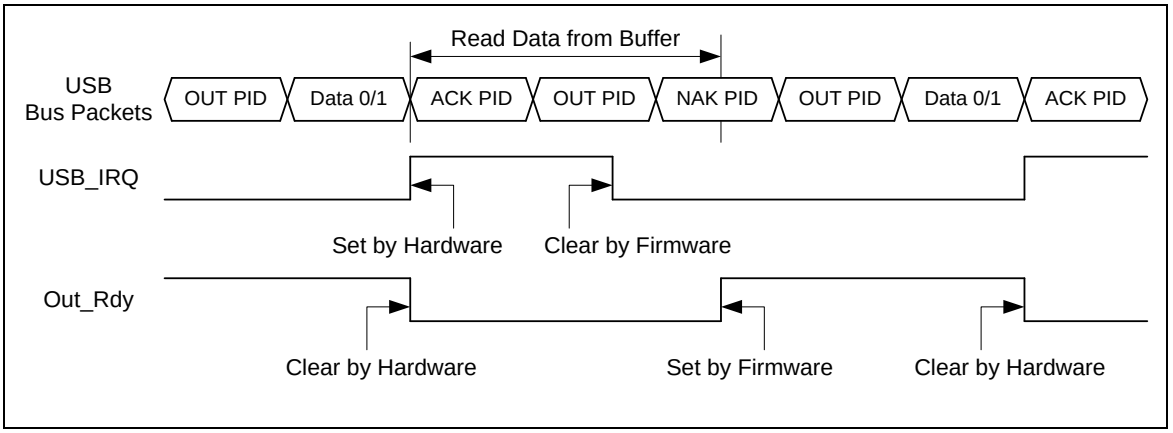


Figure 6.16-5 Data Out Transfer

6.16.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
USB Base Address: **USBD_BA = 0x4006_0000**				
USB_INTEN	USBD_BA+0x000	R/W	USB Interrupt Enable Register	0x0000_0000
USB_INTSTS	USBD_BA+0x004	R/W	USB Interrupt Event Status Register	0x0000_0000
USB_FADDR	USBD_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000
USB_EPSTS	USBD_BA+0x00C	R	USB Endpoint Status Register	0x0000_0000
USB_ATTR	USBD_BA+0x010	R/W	USB Bus Status and Attribution Register	0x0000_0040
USB_FLDET	USBD_BA+0x014	R	USB Floating Detection Register	0x0000_0000
USB_STBUFSEG	USBD_BA+0x018	R/W	Setup Token Buffer Segmentation Register	0x0000_0000
USBD_FN	USBD_BA+0x08C	R	USB Frame Number Register	0x0000_0XXX
USB_DRVSE0	USBD_BA+0x090	R/W	USB Drive SE0 Control Register	0x0000_0001
USB_BUFSEG0	USBD_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USB_MXPLD0	USBD_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USB_CFG0	USBD_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USB_CFGP0	USBD_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG1	USBD_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USB_MXPLD1	USBD_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USB_CFG1	USBD_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USB_CFGP1	USBD_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG2	USBD_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USB_MXPLD2	USBD_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USB_CFG2	USBD_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USB_CFGP2	USBD_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG3	USBD_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USB_MXPLD3	USBD_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USB_CFG3	USBD_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USB_CFGP3	USBD_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG4	USBD_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000

USB_MXPLD4	USBD_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USB_CFG4	USBD_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USB_CFGP4	USBD_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG5	USBD_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USB_MXPLD5	USBD_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USB_CFG5	USBD_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USB_CFGP5	USBD_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG6	USBD_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USB_MXPLD6	USBD_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USB_CFG6	USBD_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USB_CFGP6	USBD_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG7	USBD_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000
USB_MXPLD7	USBD_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000
USB_CFG7	USBD_BA+0x578	R/W	Endpoint 7 Configuration Register	0x0000_0000
USB_CFGP7	USBD_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

Memory Type	**Address**	**Size**	**Description**
USBD_BA = 0x4006_0000			
SRAM	USBD_BA+0x100 ~ USBD_BA+0x2FF	512 Bytes	The SRAM is used for the entire endpoint buffers. Refer to section 5.4.4.7 for the endpoint SRAM structure and its description.

6.16.7 Register Description

USB Interrupt Enable Register (USB_INTEN)

Register	Offset	R/W	Description	Reset Value
USB_INTEN	USBD_BA+0x000	R/W	USB 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
INNAK_EN	Reserved						WAKEUP_EN
7	6	5	4	3	2	1	0
Reserved				WAKEUP_IE	FLDET_IE	USB_IE	BUS_IE

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**INNAK_EN**	**Active NAK Function And Its Status In IN Token** 0 = When device responds NAK after receiving IN token, IN NAK status will not be updated to USBD_EPSTS register, so that the USB interrupt event will not be asserted. 1 = IN NAK status will be updated to USBD_EPSTS register and the USB interrupt event will be asserted, when the device responds NAK after receiving IN token.
[14:9]	**Reserved**	Reserved.
[8]	**WAKEUP_EN**	**Wake-Up Function Enable Bit** 0 = USB wake-up function Disabled. 1 = USB wake-up function Enabled.
[7:4]	**Reserved**	Reserved.
[3]	**WAKEUP_IE**	**USB Wake-Up Interrupt Enable Bit** 0 = Wake-up Interrupt Disabled. 1 = Wake-up Interrupt Enabled.
[2]	**FLDET_IE**	**Floating Detection Interrupt Enable Bit** 0 = Floating detection Interrupt Disabled. 1 = Floating detection Interrupt Enabled.
[1]	**USB_IE**	**USB Event Interrupt Enable Bit** 0 = USB event interrupt Disabled. 1 = USB event interrupt Enabled.
[0]	**BUS_IE**	**Bus Event Interrupt Enable Bit** 0 = BUS event interrupt Disabled. 1 = BUS event interrupt Enabled.

USB Interrupt Event Status Register (USB_INTSTS)

Register	Offset	R/W	Description	Reset Value
USB_INTSTS	USBD_BA+0x004	R/W	USB Interrupt Event Status Register	0x0000_0000

31	30	29	28	27	26	25	24
SETUP	Reserved						
23	22	21	20	19	18	17	16
EPEVT7	EPEVT6	EPEVT5	EPEVT4	EPEVT3	EPEVT2	EPEVT1	EPEVT0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			SOF_STS	WAKEUP_STS	FLDET_STS	USB_STS	BUS_STS

Bits	Description	
[31]	SETUP	**Setup Event Status** 0 = No Setup event. 1 = SETUP event occurred, cleared by write 1 to USB_INTSTS[31].
[30:24]	Reserved	Reserved.
[23]	EPEVT7	**Endpoint 7's USB Event Status** 0 = No event occurred on endpoint 7. 1 = USB event occurred on Endpoint 7, check USB_EPSTS[31:29] to know which kind of USB event was occurred, cleared by write 1 to USB_INTSTS[23] or USB_INTSTS[1].
[22]	EPEVT6	**Endpoint 6's USB Event Status** 0 = No event occurred on endpoint 6. 1 = USB event occurred on Endpoint 6, check USB_EPSTS[28:26] to know which kind of USB event was occurred, cleared by write 1 to USB_INTSTS[22] or USB_INTSTS[1].
[21]	EPEVT5	**Endpoint 5's USB Event Status** 0 = No event occurred on endpoint 5. 1 = USB event occurred on Endpoint 5, check USB_EPSTS[25:23] to know which kind of USB event was occurred, cleared by write 1 to USB_INTSTS[21] or USB_INTSTS[1].
[20]	EPEVT4	**Endpoint 4's USB Event Status** 0 = No event occurred on endpoint 4. 1 = USB event occurred on Endpoint 4, check USB_EPSTS[22:20] to know which kind of USB event was occurred, cleared by write 1 to USB_INTSTS[20] or USB_INTSTS[1].
[19]	EPEVT3	**Endpoint 3's USB Event Status** 0 = No event occurred on endpoint 3. 1 = USB event occurred on Endpoint 3, check USB_EPSTS[19:17] to know which kind of USB event was occurred, cleared by write 1 to USB_INTSTS[19] or USB_INTSTS[1].
[18]	EPEVT2	**Endpoint 2's USB Event Status** 0 = No event occurred on endpoint 2.

		1 = USB event occurred on Endpoint 2, check USB_EPSTS[16:14] to know which kind of USB event was occurred, cleared by write 1 to USB_INTSTS[18] or USB_INTSTS[1].
[17]	**EPEVT1**	**Endpoint 1's USB Event Status** 0 = No event occurred on endpoint 1. 1 = USB event occurred on Endpoint 1, check USB_EPSTS[13:11] to know which kind of USB event was occurred, cleared by write 1 to USB_INTSTS[17] or USB_INTSTS[1].
[16]	**EPEVT0**	**Endpoint 0's USB Event Status** 0 = No event occurred on endpoint 0. 1 = USB event occurred on Endpoint 0, check USB_EPSTS[10:8] to know which kind of USB event was occurred, cleared by write 1 to USB_INTSTS[16] or USB_INTSTS[1].
[15:5]	**Reserved**	Reserved.
[4]	**SOF_STS**	**Start of Frame Interrupt Status** 0 = SOF event does not occur. 1 = SOF event occurred, cleared by write 1 to USBD_INTSTS[4].
[3]	**WAKEUP_STS**	**Wake-Up Interrupt Status** 0 = No Wake-up event occurred. 1 = Wake-up event occurred, cleared by write 1 to USB_INTSTS[3].
[2]	**FLDET_STS**	**Floating Detection Interrupt Status** 0 = There is not attached/detached event in the USB. 1 = There is attached/detached event in the USB bus and it is cleared by write 1 to USB_INTSTS[2].
[1]	**USB_STS**	**USB Event Interrupt Status** The USB event includes the SETUP Token, IN Token, OUT ACK, ISO IN, or ISO OUT events in the bus. 0 = No USB event occurred. 1 = USB event occurred, check EPSTS0~7 to know which kind of USB event occurred. Cleared by write 1 to USB_INTSTS[1] or EPEVT0~7 and SETUP (USB_INTSTS[31]).
[0]	**BUS_STS**	**BUS Interrupt Status** The BUS event means that there is one of the suspense or the resume function in the bus. 0 = No BUS event occurred. 1 = Bus event occurred; check USB_ATTR[3:0] to know which kind of bus event was occurred, cleared by write 1 to USB_INTSTS[0].

USB Device Function Address Register (USB_FADDR)

A 7-bit value is used as the address of a device on the USB BUS.

Register	Offset	R/W	Description	Reset Value
USB_FADDR	USBD_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:0]	**FADDR**	USB Device Function Address

USB Endpoint Status Register (USB_EPSTS)

Register	Offset	R/W	Description	Reset Value
USB_EPSTS	USBD_BA+0x00C	R	USB Endpoint Status Register	0x0000_0000

31	30	29	28	27	26	25	24
EPSTS7			EPSTS6			EPSTS5	
23	22	21	20	19	18	17	16
EPSTS5	EPSTS4			EPSTS3			EPSTS2
15	14	13	12	11	10	9	8
EPSTS2		EPSTS1			EPSTS0		
7	6	5	4	3	2	1	0
OVERRUN	Reserved						

Bits	Description	
[31:29]	EPSTS7	**Endpoint 7 Bus Status** These bits are used to indicate the current status of this endpoint 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 110 = Out Packet Data1 ACK. 011 = Setup ACK. 111 = Isochronous transfer end.
[28:26]	EPSTS6	**Endpoint 6 Bus Status** These bits are used to indicate the current status of this endpoint 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 110 = Out Packet Data1 ACK. 011 = Setup ACK. 111 = Isochronous transfer end.
[25:23]	EPSTS5	**Endpoint 5 Bus Status** These bits are used to indicate the current status of this endpoint 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 110 = Out Packet Data1 ACK. 011 = Setup ACK. 111 = Isochronous transfer end.
[22:20]	EPSTS4	**Endpoint 4 Bus Status** These bits are used to indicate the current status of this endpoint 000 = In ACK.

		001 = In NAK. 010 = Out Packet Data0 ACK. 110 = Out Packet Data1 ACK. 011 = Setup ACK. 111 = Isochronous transfer end.
[19:17]	**EPSTS3**	**Endpoint 3 Bus Status** These bits are used to indicate the current status of this endpoint 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 110 = Out Packet Data1 ACK. 011 = Setup ACK. 111 = Isochronous transfer end.
[16:14]	**EPSTS2**	**Endpoint 2 Bus Status** These bits are used to indicate the current status of this endpoint 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 110 = Out Packet Data1 ACK. 011 = Setup ACK. 111 = Isochronous transfer end.
[13:11]	**EPSTS1**	**Endpoint 1 Bus Status** These bits are used to indicate the current status of this endpoint 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 110 = Out Packet Data1 ACK. 011 = Setup ACK. 111 = Isochronous transfer end.
[10:8]	**EPSTS0**	**Endpoint 0 Bus Status** These bits are used to indicate the current status of this endpoint 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 110 = Out Packet Data1 ACK. 011 = Setup ACK. 111 = Isochronous transfer end.
[7]	**OVERRUN**	**Overrun** It indicates that the received data is over the maximum payload number or not. 0 = No overrun. 1 = Out Data is more than the Max Payload in MXPLD register or the Setup Data is more than 8 Bytes.
[6:0]	**Reserved**	Reserved.

USB Bus Status and Attribution Register (USB_ATTR)

Register	Offset	R/W	Description	Reset Value
USB_ATTR	USBD_BA+0x010	R/W	USB Bus Status and Attribution Register	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BYTEM	PWRDN	DPPU_EN
7	6	5	4	3	2	1	0
USB_EN	Reserved	RWAKEUP	PHY_EN	TIMEOUT	RESUME	SUSPEND	USBRST

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**BYTEM**	**CPU Access USB SRAM Size Mode Selection** 0 = Word mode: The size of the transfer from CPU to USB SRAM can be Word only. 1 = Byte mode: The size of the transfer from CPU to USB SRAM can be Byte only.
[9]	**PWRDN**	**Power-Down PHY Transceiver, Low Active** 0 = Power-down related circuit of PHY transceiver. 1 = Turn-on related circuit of PHY transceiver.
[8]	**DPPU_EN**	**Pull-Up Resistor On USB_D+ Enable Bit** 0 = Pull-up resistor in USB_D+ pin Disabled. 1 = Pull-up resistor in USB_D+ pin Enabled.
[7]	**USB_EN**	**USB Controller Enable Bit** 0 = USB Controller Disabled. 1 = USB Controller Enabled.
[6]	**Reserved**	Reserved.
[5]	**RWAKEUP**	**Remote Wake-Up** 0 = Release the USB bus from K state. 1 = Force USB bus to K (USB_D+ low, USB_D- high) state, used for remote wake-up.
[4]	**PHY_EN**	**PHY Transceiver Function Enable Bit** 0 = PHY transceiver function Disabled. 1 = PHY transceiver function Enabled.
[3]	**TIMEOUT**	**Time-Out Status** 0 = No time-out. 1 = No Bus response more than 18 bits time. **Note:** This bit is read only.
[2]	**RESUME**	**Resume Status**

		0 = No bus resume. 1 = Resume from suspend. **Note:** This bit is read only.
[1]	**SUSPEND**	**Suspend Status** 0 = Bus no suspend. 1 = Bus idle more than 3ms, either cable is plugged off or host is sleeping. **Note:** This bit is read only.
[0]	**USBRST**	**USB Reset Status** 0 = Bus no reset. 1 = Bus reset when SE0 (single-ended 0) is presented more than 2.5us. **Note:** This bit is read only.

Floating detection Register (USB_FLDET)

Register	Offset	R/W	Description	Reset Value
USB_FLDET	USBD_BA+0x014	R	USB Floating Detection Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**FLDET**	**Device Floating Detected** 0 = Controller is not attached into the USB host. 1 =Controller is attached into the BUS.

Buffer Segmentation Register (USB_STBUFSEG)

For Setup token only.

Register	Offset	R/W	Description	Reset Value
USB_STBUFSEG	USBD_BA+0x018	R/W	Setup Token Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:3]	**STBUFSEG**	**Setup Token Buffer Segmentation** It is used to indicate the offset address for the SETUP token with the USB Device SRAM starting address The effective starting address is USB_SRAM address + {STBUFSEG[8:3], 3'b000} Where the USB_SRAM address = USBD_BA+0x100h. **Note:** It is used for SETUP token only.
[2:0]	**Reserved**	Reserved.

USB Frame Number Register (USBD_FN)

Register	Offset	R/W	Description	Reset Value
USBD_FN	USBD_BA+0x08C	R	USB Frame Number Register	0x0000_0XXX

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**FN**	**Frame Number** These bits contain the 11-bits frame number in the last received SOF packet.

Buffer Segmentation Register (USB_BUFSEGx)

Register	Offset	R/W	Description	Reset Value
USB_BUFSEG0	USBD_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG1	USBD_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG2	USBD_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG3	USBD_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG4	USBD_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG5	USBD_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG6	USBD_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG7	USBD_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:3]	**BUFSEG**	**Endpoint Buffer Segmentation** It is used to indicate the offset address for each endpoint with the USB SRAM starting address The effective starting address of the endpoint is USB_SRAM address + { BUFSEG[8:3], 3'b000} Where the USB_SRAM address = USBD_BA+0x100h. Refer to the section 5.4.4.7 for the endpoint SRAM structure and its description.
[2:0]	**Reserved**	Reserved.

Maximal Payload Register (USB_MXPLDx)

Register	Offset	R/W	Description	Reset Value
USB_MXPLD0	USBD_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USB_MXPLD1	USBD_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USB_MXPLD2	USBD_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USB_MXPLD3	USBD_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USB_MXPLD4	USBD_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USB_MXPLD5	USBD_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USB_MXPLD6	USBD_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USB_MXPLD7	USBD_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**MXPLD**	**Maximal Payload** Define the data length which is transmitted to host (IN token) or the actual data length which is received from the host (OUT token). It also used to indicate that the endpoint is ready to be transmitted in IN token or received in OUT token. (1) When the register is written by CPU, For IN token, the value of MXPLD is used to define the data length to be transmitted and indicate the data buffer is ready. For OUT token, it means that the controller is ready to receive data from the host and the value of MXPLD is the maximal data length comes from host. (2) When the register is read by CPU, For IN token, the value of MXPLD is indicated by the data length be transmitted to host For OUT token, the value of MXPLD is indicated the actual data length receiving from host. **Note:** Once MXPLD is written, the data packets will be transmitted/received immediately after IN/OUT token arrived.

Configuration Register (USB_CFGx)

Register	Offset	R/W	Description	Reset Value
USB_CFG0	USBD_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USB_CFG1	USBD_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USB_CFG2	USBD_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USB_CFG3	USBD_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USB_CFG4	USBD_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USB_CFG5	USBD_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USB_CFG6	USBD_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USB_CFG7	USBD_BA+0x578	R/W	Endpoint 7 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						**CSTALL**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DSQ_SYNC	**STATE**		**ISOCH**	**EP_NUM**			

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**CSTALL**	**Clear STALL Response** 0 = Disable the device to clear the STALL handshake in setup stage. 1 = Clear the device to response STALL handshake in setup stage.
[8]	**Reserved**	Reserved.
[7]	**DSQ_SYNC**	**Data Sequence Synchronization** 0 = DATA0 PID. 1 = DATA1 PID. **Note:** It is used to specify the DATA0 or DATA1 PID in the following IN token transaction. Hardware will toggle automatically in IN token base on the bit.
[6:5]	**STATE**	**Endpoint STATE** 00 = Endpoint is Disabled. 01 = Out endpoint. 10 = IN endpoint. 11 = Undefined.
[4]	**ISOCH**	**Isochronous Endpoint**

		This bit is used to set the endpoint as Isochronous endpoint, no handshake. 0 = No Isochronous endpoint. 1 = Isochronous endpoint.
[3:0]	**EP_NUM**	**Endpoint Number** These bits are used to define the endpoint number of the current endpoint.

Extra Configuration Register (USB_CFGPx)

Register	Offset	R/W	Description	Reset Value
USB_CFGP0	USBD_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP1	USBD_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP2	USBD_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP3	USBD_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP4	USBD_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP5	USBD_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP6	USBD_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP7	USBD_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**SSTALL**	**Set STALL** 0 = Disable the device to response STALL. 1 = Set the device to respond STALL automatically.
[0]	**CLRRDY**	**Clear Ready** When the USB_MXPLD register is set by user, it means that the endpoint is ready to transmit or receive data. If the user wants to disable this transaction before the transaction start, users can set this bit to 1 to turn it off and it will be cleared to 0 automatically. For IN token, write '1' to clear the IN token had ready to transmit the data to USB. For OUT token, write '1' to clear the OUT token had ready to receive the data from USB. This bit is write 1 only and is always 0 when it is read back.

USB Drive SE0 Register (USB_DRVSE0)

Register	Offset	R/W	Description	Reset Value
USB_DRVSE0	USBD_BA+0x090	R/W	USB Drive SE0 Control Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**DRVSE0**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**DRVSE0**	**Drive Single Ended Zero In USB Bus** The Single Ended Zero (SE0) is when both lines (USB_D+ and USB_D-) are being pulled low. 0 = None. 1 = Force USB PHY transceiver to drive SE0.

6.17 Analog-to-Digital Converter (ADC)

6.17.1 Overview

The NuMicro® NUC029LEE/NUC029SEE contains one 12-bit successive approximation analog-to-digital converters (SAR A/D converter) with 12 input channels. The A/D converter supports three operation modes: single, single-cycle scan and continuous scan mode. The A/D converter can be started by software, PWM Center-aligned trigger and external STADC pin.

6.17.2 Features

- Analog input voltage range: 0~V_{REF}
- 12-bit resolution and 10-bit accuracy is guaranteed
- Up to 12 single-end analog input channels or 5 differential analog input channels
- Up to 1 MSPS conversion rate (chip working at 5V)
- Three operating modes
  - Single mode: A/D conversion is performed one time on a specified channel
  - Single-cycle scan mode: A/D conversion is performed 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:
  - Writing 1 to ADST bit (ADCR[11])through software
  - PWM Center-aligned trigger
  - External pin STADC
- Conversion results are held in data registers for each channel with valid and overrun indicators
- Supports two set digital comparators. The conversion result can be compared with specify value and user can select whether to generate an interrupt when conversion result matches the compare register setting
- Channel 7 supports 3 input sources: external analog voltage, internal Band-gap voltage, and internal temperature sensor output

6.17.3 Block Diagram

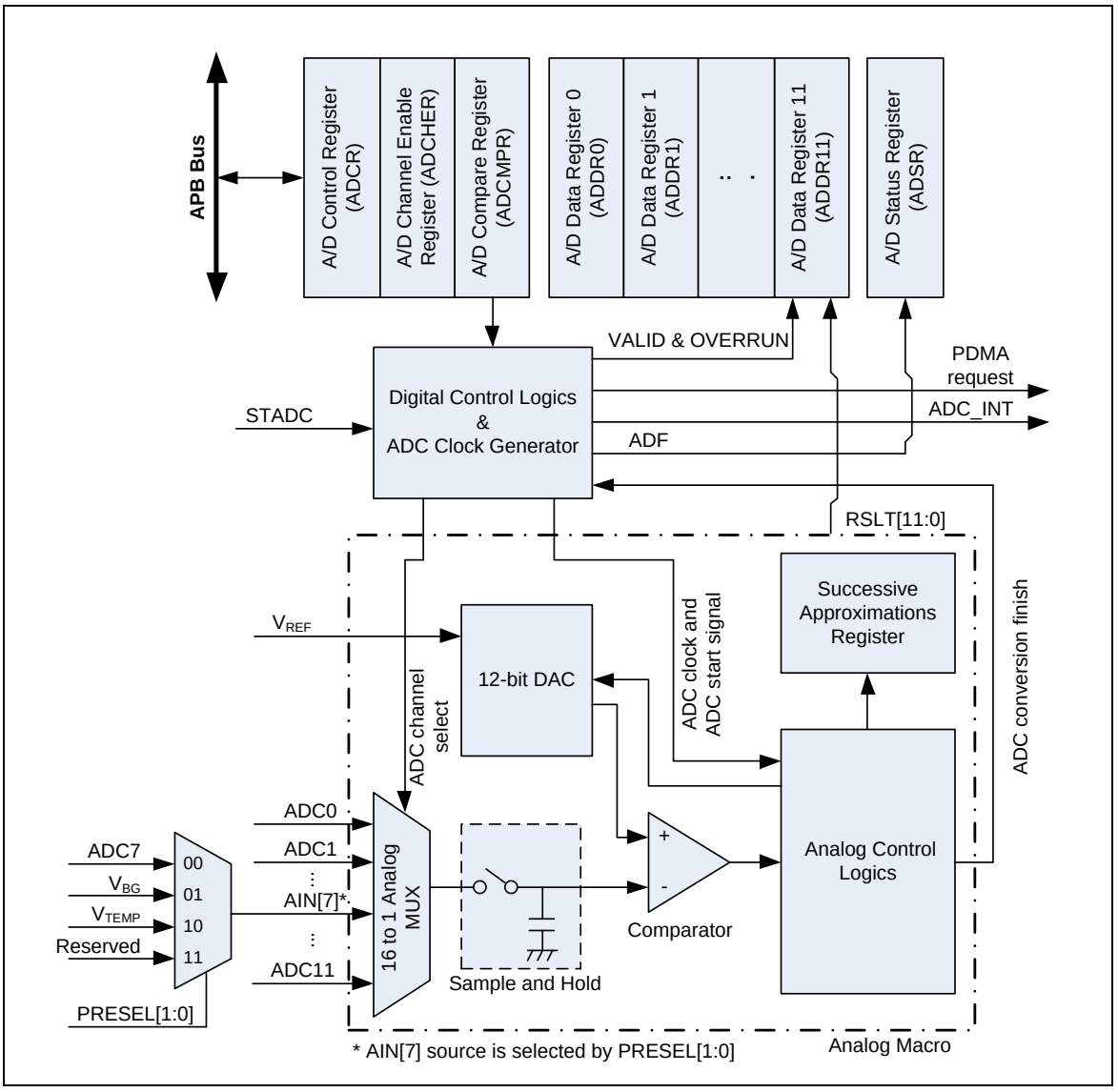


Figure 6.17-1 ADC Controller Block Diagram

6.17.4 Basic Configuration

The ADC Controller clock source is enabled by ADC_EN bit (CLK_APBCLK[28]). After user change the GPA_MFP, GPB_MFP and GPC_MFP register to ADC analog input, user need set OFFD (GPIOA_OFFD [23:16], GPIOB_OFFD [31], GPIOC_OFFD [23:22], GPIOC_OFFD [31:30]) = 1 to disable digital input path.

6.17.5 Functional Description

The A/D converter operates by successive approximation with 12-bit resolution. The ADC has three operation modes: Single mode, Single-cycle Scan mode and Continuous Scan mode. When

changing the operating mode or analog input channel, to prevent incorrect operation, software must clear ADST bit (ADCR[11]) to 0.

6.17.5.1 ADC Clock Generator

The maximum sampling rate is up to 1 MSPS. The ADC engine has four clock sources selected by 2-bit ADC_S (CLKSEL1[3:2]), the ADC clock frequency is divided by an 8-bit prescaler with the formula:

The ADC clock frequency = (ADC clock source frequency) / (ADC_N (CLKDIV[23:16])+1);

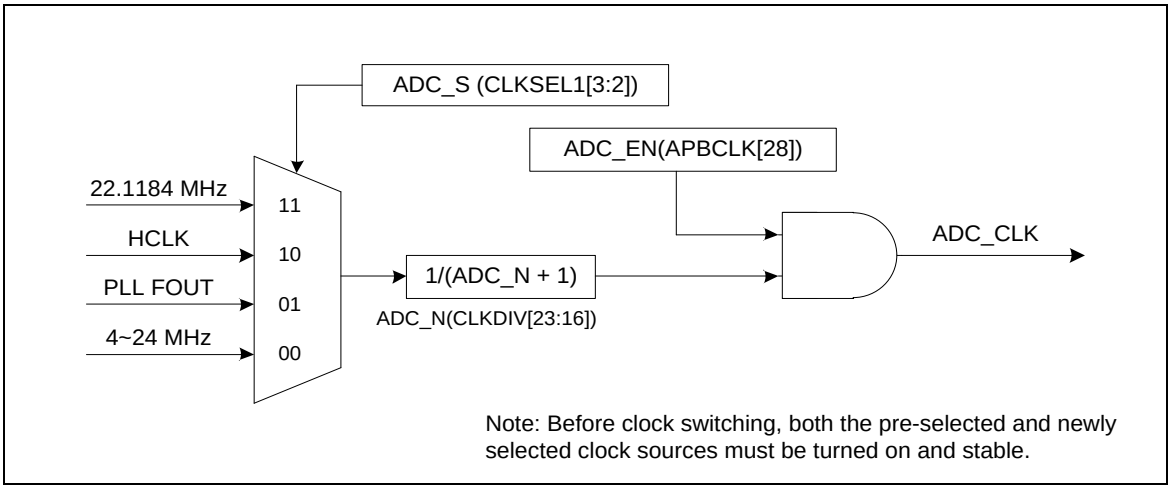


Figure 6.17-2 ADC Clock Control

6.17.5.2 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 (ADCR[11]) is set to 1 by software.
2. When A/D conversion is finished, the result is stored in the A/D data register corresponding to the channel.
3. The ADF bit (ADSR[0]) will be set to 1. If the ADIE bit (ADCR[1]) 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: If software enables more than one channel in single mode, the channel with the smallest number will be selected and the other enabled channels will be ignored.

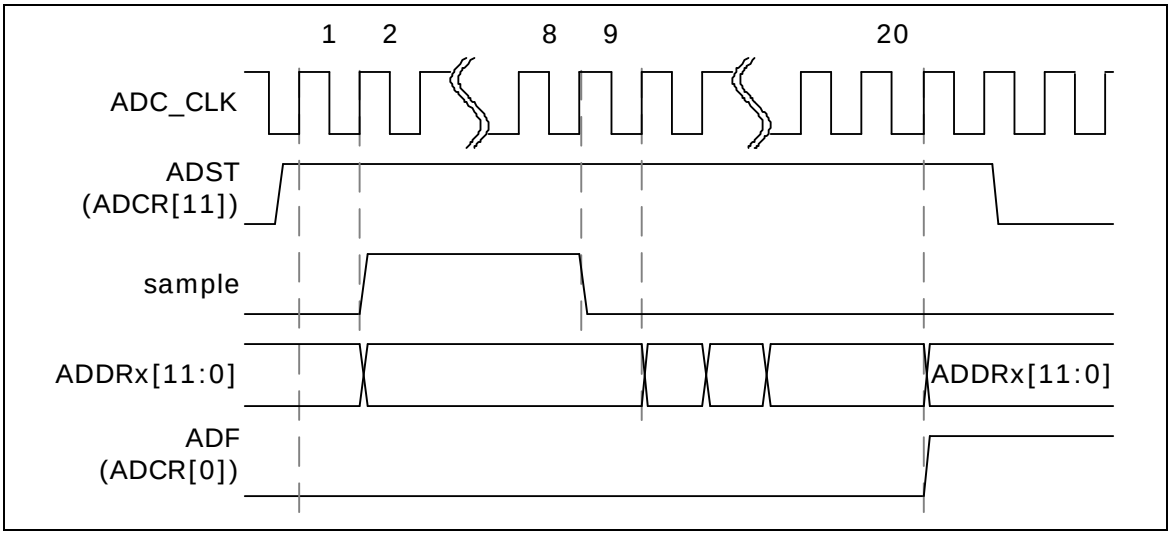


Figure 6.17-3 Single Mode Conversion Timing Diagram

6.17.5.3Single-Cycle Scan Mode

In single-cycle scan mode, A/D conversion will sample and convert the specified channels once in the sequence from the smallest number enabled channel to the largest number enabled channel.

1. When the ADST bit (ADCR[11]) is set to 1 by software or external trigger input, A/D conversion starts on the channel with the smallest number.
2. When A/D conversion for each enabled channel is completed, the result is sequentially transferred to the A/D data register corresponding to each channel.
3. When the conversions of all the enabled channels are completed, the ADF bit (ADSR[0]) is set to 1. If the ADC interrupt function is enabled, the ADC interrupt occurs.
4. After A/D conversion ends, the ADST bit is automatically cleared to 0 and the A/D converter enters idle state. If ADST is cleared to 0 before all enabled ADC channels conversion done, ADC controller will finish current conversion and save the result to the ADDRx of the current conversion channel.

An example timing diagram for single-cycle scan on enabled channels (0, 2, 3 and 8) is shown below:

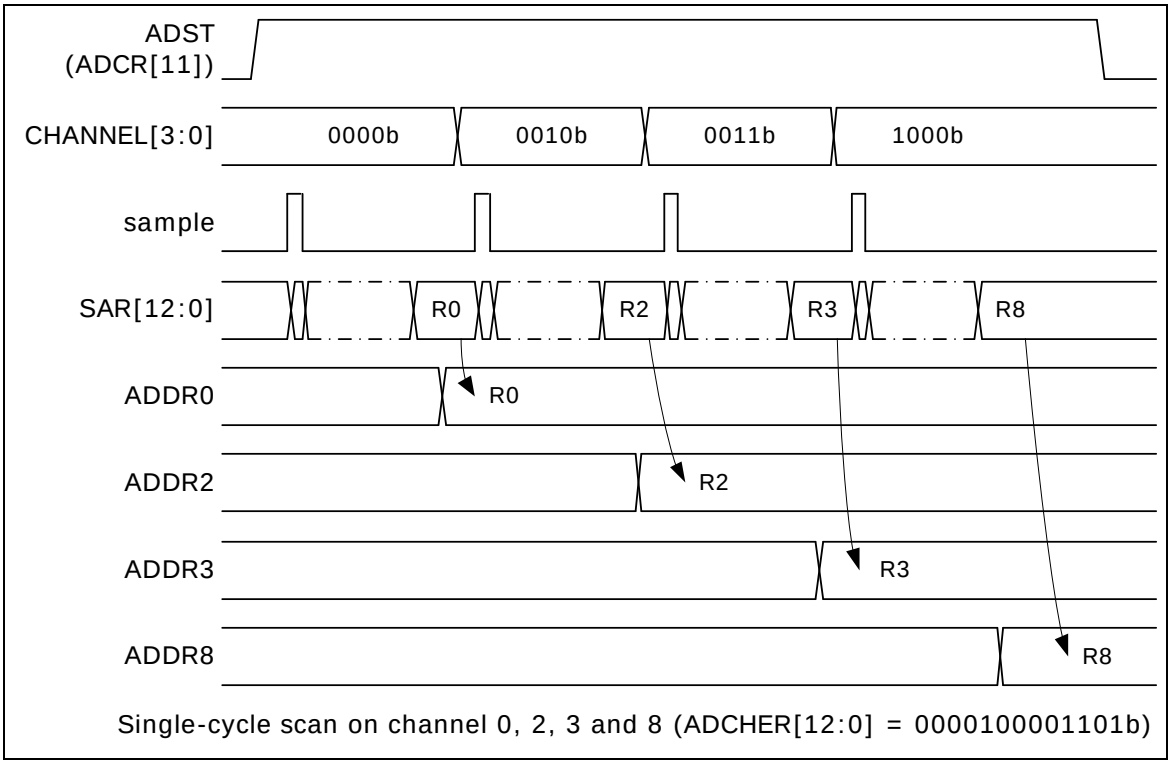


Figure 6.17-4 Single-Cycle Scan on Enabled Channels Timing Diagram

6.17.5.4 Continuous Scan Mode

In continuous scan mode, A/D conversion is performed sequentially on the specified channels that enabled by CHEN bits (ADCHER[12:0]). The operations are as follows:

1. When the ADST bit (ADCR[11]) is set to 1 by software, A/D conversion starts on the 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 A/D data register corresponding to each enabled channel.
3. When A/D converter completes the conversions of all enabled channels sequentially, the ADF bit (ADSR[0]) 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 has not cleared 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 controller will stop conversion.

An example timing diagram for continuous scan on enabled channels (0, 2, 3 and 9) is shown below:

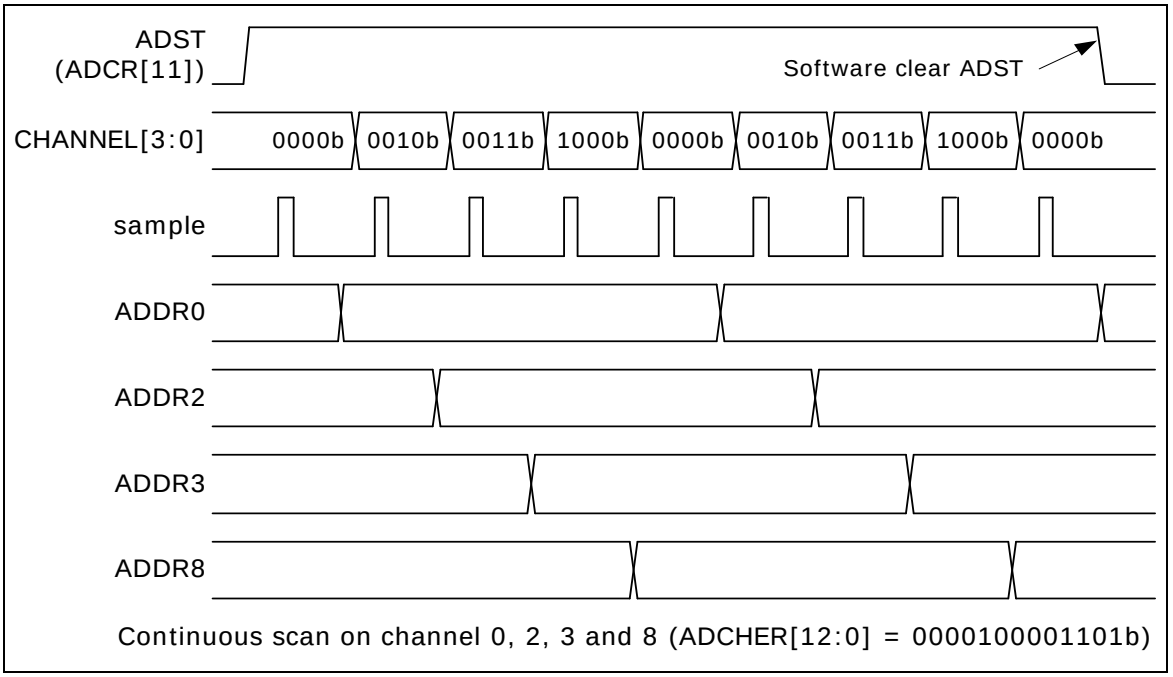


Figure 6.17-5 Continuous Scan on Enabled Channels Timing Diagram

6.17.5.5 Internal Reference Voltage

The band-gap voltage reference (V_{BG}) is an internal fixed reference voltage regardless of power supply variations. The V_{BG} output is internally connected to ADC channel 7 source multiplexer and Analog Comparators's (ACMP) negative input side.

For battery power detection application, user can use the V_{BG} as ADC input channel such that user can convert the A/D conversion result to calculate AV_{DD} with following formula.

$AV_{DD} = ((2 \text{ ^ } N) / R) * V_{BG}$

N: ADC resolution

R: A/D conversion result

V_{BG}: Band-gap voltage

The block diagram is shown as Figure 6.17-6.

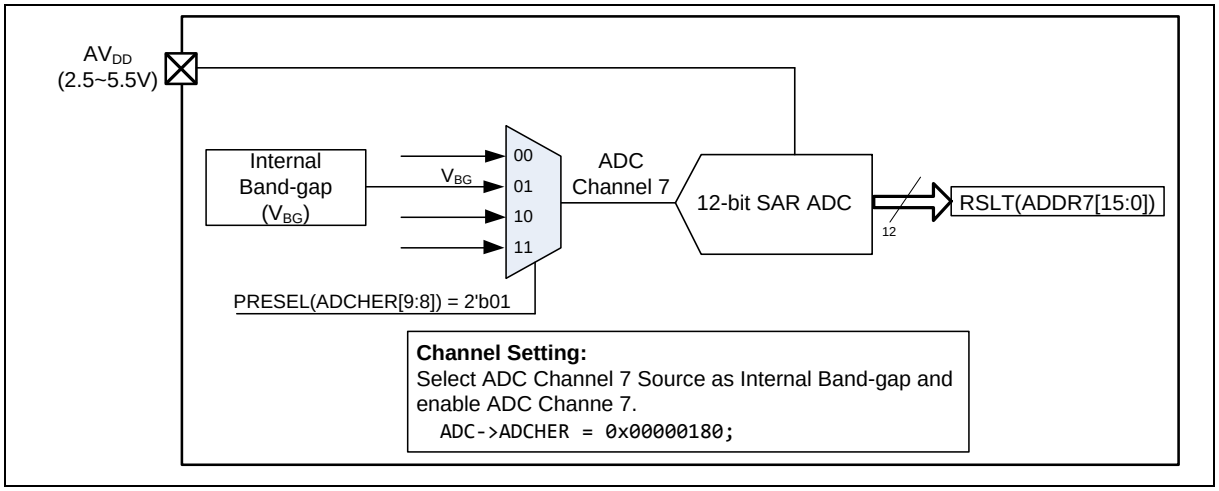


Figure 6.17-6 V_{BG} for Measuring AV_{DD} Application Block Diagram

For example, the V_{BG} typical value is 1.25 V, the ADC is 12-bit resolution, select V_{BG} as ADC channel 7 input source, and enable ADC channel 7. Then trigger ADC to converse.

If the A/D conversion result is 1707:

N = 12

R = 1707

V_{BG} = 1.25 V

AV_{DD} = ((2 ^ 12) / 1707) * 1.25 = (4096 / 1707) * 1.25 = 3 V

If the A/D conversion result is 2048:

AV_{DD} = ((2 ^ 12) / 2048) * 1.25 = (4096 / 2048) * 1.25 = 2.5 V

6.17.5.6External trigger Input Sampling and A/D Conversion Time

In single-cycle scan mode, A/D conversion can be triggered by external pin request. When the TRGEN (ADCR[8]) is set to high to enable ADC external trigger function, setting the TRGS bits (ADCR[5:4]) to 00b is to select external trigger input from the STADC pin. Software can set TRGCOND (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 defined state at least 8 PCLKs. The ADST bit will be set to 1 at the 9th PCLK and start to conversion. Conversion is continuous 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 PLCKs. Pulse that is shorter than this specification will be ignored.

6.17.5.7PWM Center-aligned trigger

In single-cycle scan mode, the PWM can be the trigger source of ADC by setting the TRGEN (ADCR[8]) to 1 and the TRGS (ADCR[5:4]) to 11b.

When PWM enables trigger ADC function, the PWM will generate a trigger signal to ADC when PWM counter is running to PWM center point.

6.17.5.8 Conversion Result Monitor by Compare Function

The ADC controller provide two sets of compare register ADCMPR0 and ADCMPR1, to monitor maximum two specified channels conversion result from A/D conversion controller, refer to Figure 6.17-7. Software can select which channel to be monitored by set CMPCH (ADCMPR0/1[5:3]) and CMPCOND bit (ADCMPR0/1[2]) is used to check conversion result is less than specify value or greater than (equal to) value specified in CMPD (ADCMPR0/1 [27:16]). When the conversion of the channel specified by CMPCH 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. It means the comparing data must be successively matched with the compare condition. Once any comparing data does not match during the comparing, the compare match counter will clear to 0. When counter value reach the setting of (CMPMATCNT (ADCMPR0/1 [11:8])+1) then CMPF0/1 bit (ADSR[1]/[2]) will be set to 1, if CMPIE bit (ADCMPR0/1 [1]) is set then an ADC_INT 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. Detailed logics diagram is shown below:

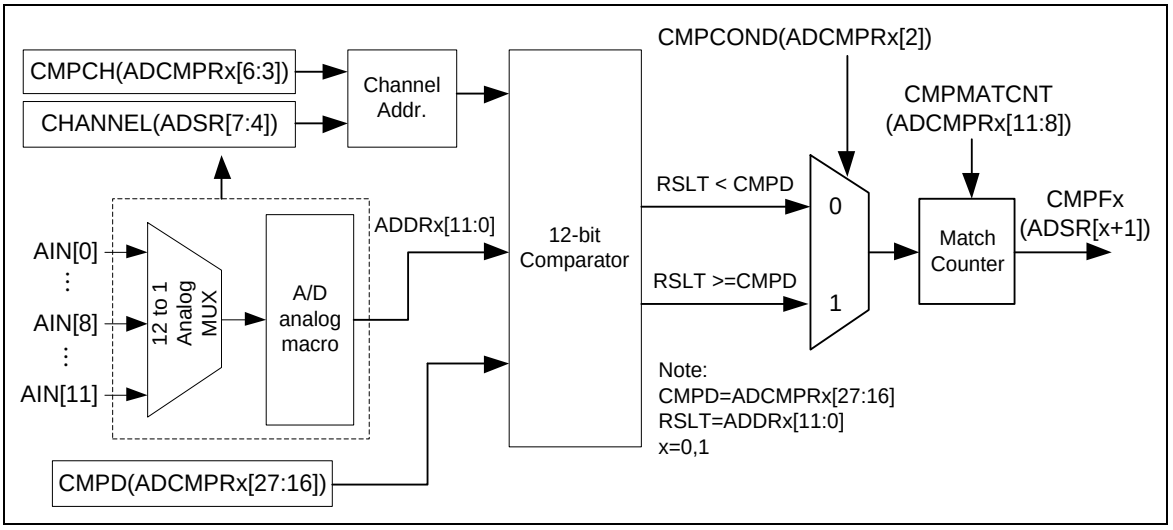


Figure 6.17-7 A/D Conversion Result Monitor Logics Diagram

6.17.5.9 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, will be set to 1. The CMPF0 (ADSR[1]) and CMPF1 (ADSR[2]) are the compare flags of compare function. When the conversion result meets the settings of ADCMPR0/1, the corresponding flag will be set to 1. When one of the flags, ADF (ADSR[0]), CMPF0 and CMPF1, is set to 1 and the corresponding interrupt enable bit, ADIE (ADCR[1]) and CMPIE (ADCMPR0/1[1]), is set to 1, the ADC interrupt will be asserted. Software can clear the flag to revoke the interrupt request.

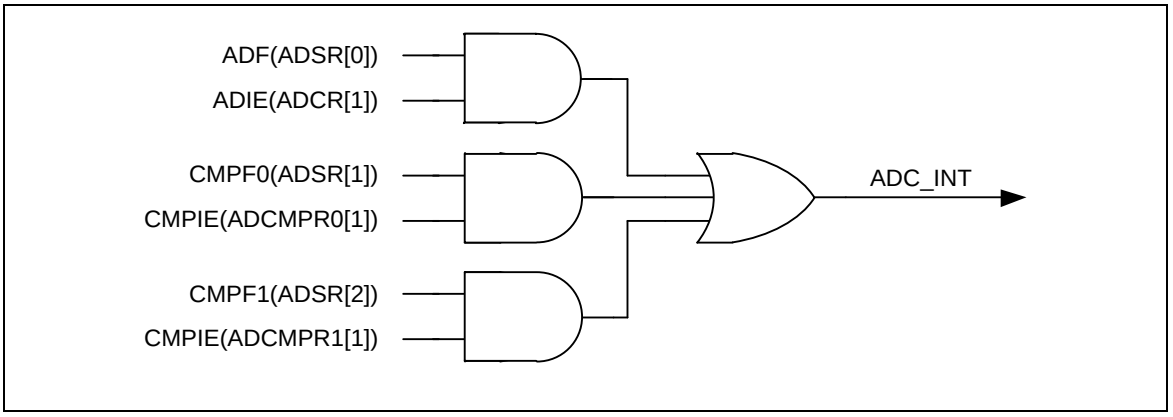


Figure 6.17-8 A/D Controller Interrupt

6.17.5.10 Peripheral DMA Request

When A/D conversion is finished, the conversion result will be loaded into ADDR register and VALID bit will be set to 1. If the PTEN bit (ADCR[9]) 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 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 single scan mode or continuous scan mode. User can monitor current PDMA transfer data through reading 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 bit (ADC_ADDRx[16], x=0~6, 8~12) 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.

6.17.6 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 = 0x400E_0000**				
ADDR0	ADC_BA+0x00	R	ADC Data Register 0	0x0000_0000
ADDR1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000
ADDR2	ADC_BA+0x08	R	ADC Data Register 2	0x0000_0000
ADDR3	ADC_BA+0x0C	R	ADC Data Register 3	0x0000_0000
ADDR4	ADC_BA+0x10	R	ADC Data Register 4	0x0000_0000
ADDR5	ADC_BA+0x14	R	ADC Data Register 5	0x0000_0000
ADDR6	ADC_BA+0x18	R	ADC Data Register 6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	ADC Data Register 7	0x0000_0000
ADCR	ADC_BA+0x20	R/W	ADC Control Register	0x0000_0000
ADCHER	ADC_BA+0x24	R/W	ADC Channel Enable Register	0x0000_0000
ADCMPR0	ADC_BA+0x28	R/W	ADC Compare Register 0	0x0000_0000
ADCMPR1	ADC_BA+0x2C	R/W	ADC Compare Register 1	0x0000_0000
ADSR	ADC_BA+0x30	R/W	ADC Status Register	0x0000_0000
ADPDMA	ADC_BA+0x40	R	ADC PDMA Current Transfer Data Register	0x0000_0000
ADDR8	ADC_BA+0x50	R	ADC Data Register 8	0x0000_0000
ADDR9	ADC_BA+0x54	R	ADC Data Register 9	0x0000_0000
ADDR10	ADC_BA+0x58	R	ADC Data Register 10	0x0000_0000
ADDR11	ADC_BA+0x5C	R	ADC Data Register 11	0x0000_0000

6.17.7 Register Description

ADC Data Registers (ADDR0 ~ ADDR11)

Register	Offset	R/W	Description	Reset Value
ADDR0	ADC_BA+0x00	R	ADC Data Register 0	0x0000_0000
ADDR1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000
ADDR2	ADC_BA+0x08	R	ADC Data Register 2	0x0000_0000
ADDR3	ADC_BA+0x0C	R	ADC Data Register 3	0x0000_0000
ADDR4	ADC_BA+0x10	R	ADC Data Register 4	0x0000_0000
ADDR5	ADC_BA+0x14	R	ADC Data Register 5	0x0000_0000
ADDR6	ADC_BA+0x18	R	ADC Data Register 6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	ADC Data Register 7	0x0000_0000
ADDR8	ADC_BA+0x50	R	ADC Data Register 8	0x0000_0000
ADDR9	ADC_BA+0x54	R	ADC Data Register 9	0x0000_0000
ADDR10	ADC_BA+0x58	R	ADC Data Register 10	0x0000_0000
ADDR11	ADC_BA+0x5C	R	ADC Data Register 11	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** 0 = Data in RSLT bits (ADDRx[15:0], x=0~7) is not valid. 1 = Data in RSLT bits (ADDRx[15:0], x=0~7) is valid. This bit is set to 1 when corresponding channel analog input conversion is completed and cleared by hardware after ADDR register is read. This is a read only bit

[16]	**OVERRUN**	**Overrun Flag** 0 = Data in RSLT (ADDRx[15:0], x=0~11) is recent conversion result. 1 = Data in RSLT (ADDRx[15:0], x=0~11) is overwritten. If converted data in RSLT has not been read before new conversion result is loaded to this register, OVERRUN is set to 1 and previous conversion result is gone. It is cleared by hardware after ADDR register is read. This is a read only bit.
[15:0]	**RSLT**	**A/D Conversion Result** This field contains conversion result of ADC. When DMOF bit (ADCR[31]) set to 0, 12-bit ADC conversion result with unsigned format will be filled in RSLT (ADDRx[11:0], x=0~11) and zero will be filled in RSLT (ADDRx[15:12], x=0~11). When DMOF bit (ADCR[31]) set to 1, 12-bit ADC conversion result with 2'complement format will be filled in RSLT(ADDRx[11:0], x=0~11) and signed bits to will be filled in RSLT (ADDRx[15:12], x=0~11).

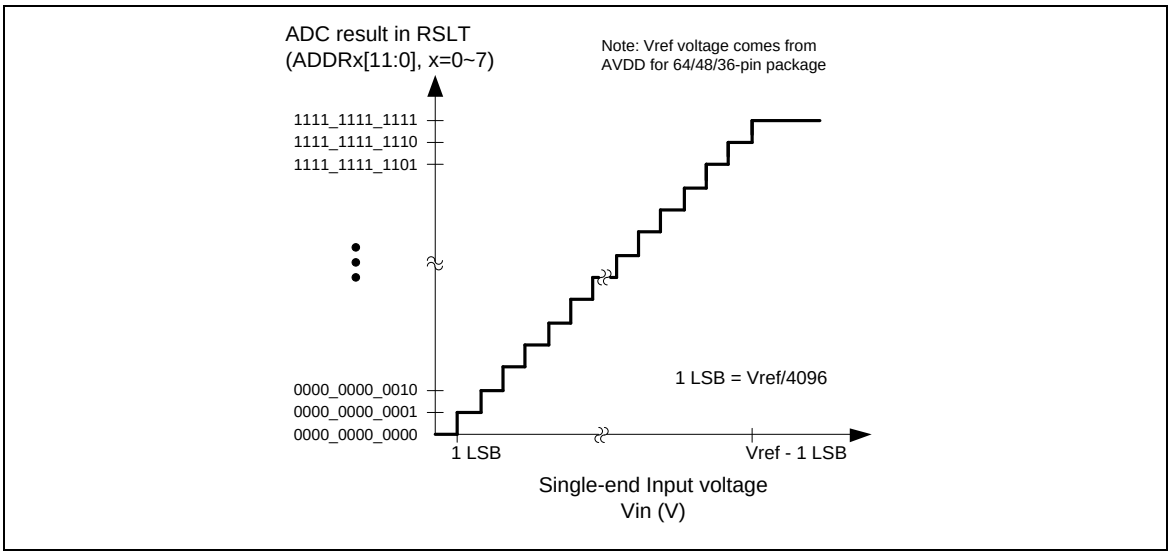


Figure 6.17-9 ADC Single-end Input Conversion Voltage and Conversion Result Mapping

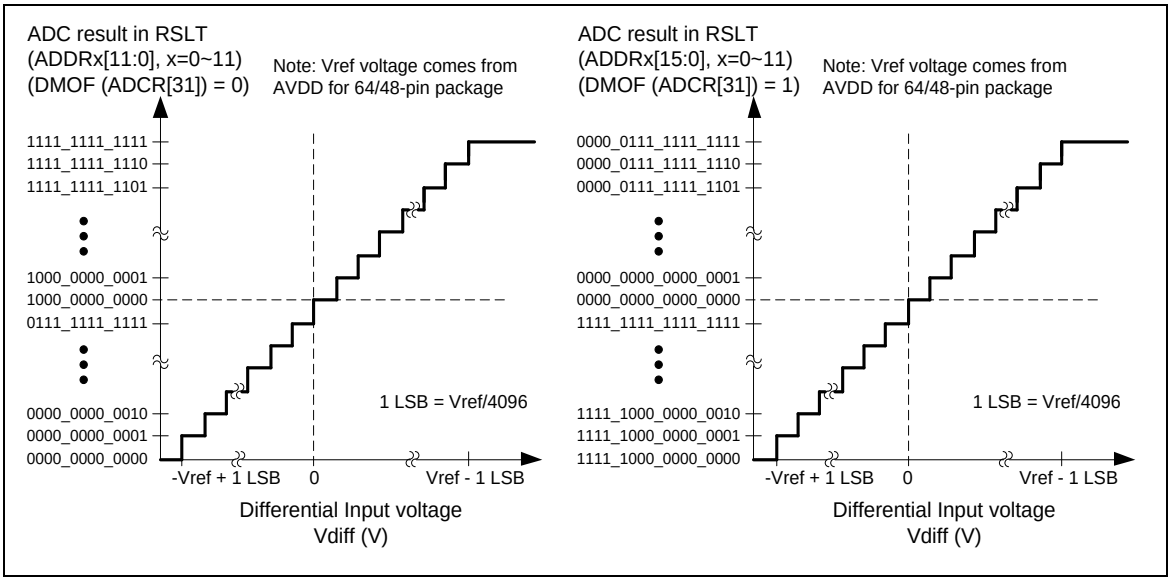


Figure 6.17-10 ADC Differential Input Conversion Voltage and Conversion Result Mapping

ADC Control Register (ADCR)

Register	Offset	R/W	Description	Reset Value
ADCR	ADC_BA+0x20	R/W	ADC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
DMOF	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
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**	**A/D Differential Input Mode Output Format** 0 = A/D Conversion result will be filled in RSLT at ADDRx registers with unsigned format. 1 = A/D Conversion result will be filled in RSLT at ADDRx registers with 2'complement format.
[30:12]	**Reserved**	Reserved.
[11]	**ADST**	**A/D Conversion Start** 0 = Conversion stops and A/D converter enter idle state. 1 = Conversion starts. ADST bit can be set to 1 from three sources: software, PWM Center-aligned trigger and external pin STADC. ADST will be cleared to 0 by hardware automatically at the ends of single mode and single-cycle scan mode. In continuous scan mode, A/D conversion is continuously performed until software writes 0 to this bit or chip reset.
[10]	**DIFFEN**	**Differential Input Mode Control** 0 = Single-end analog input mode. 1 = Differential analog input mode.

Differential input Paired Channel	ADC Analog Input	
	V_{plus}	V_{minus}
0	ADC1	ADC2
1	ADC3	ADC4
2	ADC5	ADC6
3	ADC7	ADC8
4	ADC9	ADC10

Differential input voltage (V_{diff}) = V_{plus} - V_{minus}, where V_{plus} is the analog input; V_{minus} is the inverted analog input.

In differential input mode, only the odd number of the two corresponding channels needs to be enabled in ADCHER. The conversion result will be placed to the corresponding data register of the enabled channel.

[9]	**PTEN**	**PDMA Transfer Enable Bit** 0 = PDMA data transfer Disabled. 1 = PDMA data transfer in ADDR 0~11 Enabled. When A/D conversion is completed, the converted data is loaded into ADDR 0~11, software can enable this bit to generate a PDMA data transfer request. When PTEN=1, software must set ADIE=0 (ADCR[1]) to disable interrupt.
[8]	**TRGEN**	**Hardware Trigger Enable Bit** Enable or disable triggering of A/D conversion by hardware (external STADC pin or PWM Center-aligned trigger). 0 = Disabled. 1 = Enabled. ADC hardware trigger function is only supported in single-cycle scan mode. If hardware trigger mode, the ADST bit (ADCR[11]) can be set to 1 by the selected hardware trigger source.
[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 4 PCLKs at high and low state 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. 11 = A/D conversion is started by PWM Center-aligned trigger. Others = Reserved. Software should disable TRGEN (ADCR[8]) and ADST (ADCR[11]) before change TRGS.
[3:2]	**ADMD**	**A/D Converter Operation Mode** 00 = Single conversion. 01 = Reserved. 10 = Single-cycle scan. 11 = Continuous scan. When changing the operation mode, software should disable ADST bit (ADCR[11]) firstly.
[1]	**ADIE**	**A/D Interrupt Enable Bit** 0 = A/D interrupt function Disabled. 1 = A/D interrupt function Enabled. A/D conversion end interrupt request is generated if ADIE bit (ADCR[1]) is set to 1.
[0]	**ADEN**	**A/D Converter Enable Bit** 0 = Disabled. 1 = Enabled. Before starting A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit for saving power consumption.

ADC Channel Enable Register (ADCHER)

Register	Offset	R/W	Description	Reset Value
ADCHER	ADC_BA+0x24	R/W	ADC Channel 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		**CHEN1**				**PRESEL**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHEN							

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:10]	**CHEN1**	**Analog Input Channel Enable Bit 1** Set CHEN[14:10] to enable the corresponding analog input channel 11 ~ 8. If DIFFEN bit (ADCR[10]) is set to 1, only the even number channels need to be enabled. 0 = ADC input channel Disabled. 1 = ADC input channel Enabled.
[9:8]	**PRESEL**	**Analog Input Channel 7 Selection** 00 = External analog input. 01 = Internal band-gap voltage. 10 = Internal temperature sensor. 11 = Reserved.
[7]	**Reserved**	Reserved.
[7:0]	**CHEN**	**Analog Input Channel Enable Bit** Set CHEN[7:0] to enable the corresponding analog input channel 7 ~ 0. If DIFFEN bit (ADCR[10]) is set to 1, only the even number channels need to be enabled. 0 = ADC input channel Disabled. 1 = ADC input channel Enabled.

ADC Compare Register 0/1 (ADCMPR0/1)

Register	Offset	R/W	Description	Reset Value
ADCMPR0	ADC_BA+0x28	R/W	ADC Compare Register 0	0x0000_0000
ADCMPR1	ADC_BA+0x2C	R/W	ADC Compare Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**CMPD[11:8]**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPD[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CMPMATCNT**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**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. When DMOF bit (ADCR[31]) is set to 0, ADC comparator compares CMPD with conversion result with unsigned format. CMPD should be filled in unsigned format. When DMOF bit (ADCR[31]) is set to 1, ADC comparator compares CMPD with conversion result with 2'complement format. CMPD should be filled in 2'complement format.
[15:12]	**Reserved**	Reserved.
[11:8]	**CMPMATCNT**	**Compare Match Count** When the specified A/D channel analog conversion result matches the compare condition defined by CMPCOND (ADCMPR0/1[2]), the internal match counter will increase 1, The comparing data must successively matched with the compare condition. Once any comparing data does not match during the comparing, the internal counter will clear to 0. When the internal counter reaches the value to (CMPMATCNT (ADCMPR0/1[11:8]) +1), the CMPF0/1 bit (ADSR[1]/[2]) will be set.
[7]	**Reserved**	Reserved.

[6:3]	**CMPCH**	**Compare Channel Selection** 0000 = Channel 0 conversion result is selected to be compared. 0001 = Channel 1 conversion result is selected to be compared. 0010 = Channel 2 conversion result is selected to be compared. 0011 = Channel 3 conversion result is selected to be compared. 0100 = Channel 4 conversion result is selected to be compared. 0101 = Channel 5 conversion result is selected to be compared. 0110 = Channel 6 conversion result is selected to be compared. 0111 = Channel 7 conversion result is selected to be compared. 1000 = Channel 8 conversion result is selected to be compared. 1001 = Channel 9 conversion result is selected to be compared. 1010 = Channel 10 conversion result is selected to be compared. 1011 = Channel 11 conversion result is selected to be compared.
[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 (ADCMPR0/1[27:16]), the internal match counter will increase one. 1 = Set the compare condition as that when a 12-bit A/D conversion result is greater or equal to the 12-bit CMPD (ADCMPR0/1[27:16]), the internal match counter will increase one. **Note:** When the internal counter reaches the value to (CMPMATCNT (ADCMPR0/1[11:8])+1), the CMPF0/1 bit (ADSR[1]/[2]) will be set.
[1]	**CMPIE**	**Compare Interrupt Enable Bit** 0 = Compare function interrupt Disabled. 1 = Compare function interrupt Enabled. If the compare function is enabled and the compare condition matches the setting of CMPCOND (ADCMPR0/1[2]) and CMPMATCNT (ADCMPR0/1[11:8]), CMPF0/1 bit (ADSR[1]/[2]) will be asserted, in the meanwhile, if CMPIE (ADCMPR0/1[1]) is set to 1, a compare interrupt request is generated.
[0]	**CMPEN**	**Compare Enable Bit** 0 = Compare function Disabled. 1 = Compare function Enabled. Set this bit to 1 to enable ADC controller to compare CMPD (ADCMPR0/1[27:16]) with specified channel conversion result when converted data is loaded into ADDR register.

ADC Status Register (ADSR)

Register	Offset	R/W	Description	Reset Value
ADSR	ADC_BA+0x30	R/W	ADC Status Register	0x0000_0000

31	30	29	28	27	26	25	24
OVERRUN1				**VALID1**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
OVERRUN0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VALID0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHANNEL				**BUSY**	**CMPF1**	**CMPF0**	**ADF**

Bits	Description	
[31:28]	**OVERRUN1**	**Overrun Flag** It is a mirror to OVERRUN bit (ADDR8~11[16]). It is read only.
[27:24]	**VALID1**	**Data Valid Flag** It is a mirror of VALID bit (ADDR8~11[17]). It is read only.
[23:16]	**OVERRUN0**	**Overrun Flag** It is a mirror to OVERRUN bit (ADDR0~7[16]). It is read only.
[15:8]	**VALID0**	**Data Valid Flag** It is a mirror of VALID bit (ADDR0~7[17]). It is read only.
[7:4]	**CHANNEL**	**Current Conversion Channel** This field reflects the current conversion channel when BUSY = 1 (ADSR[3]). When BUSY = 0, it shows the number of the next converted channel. It is read only.
[3]	**BUSY**	**BUSY/IDLE** 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion. This bit is mirror of as ADST bit (ADCR[11]). It is read only.
[2]	**CMPF1**	**Compare Flag** When the selected channel A/D conversion result meets setting condition in ADCMPR1 then this bit is set to 1. And it is cleared by writing 1 to self. 0 = Conversion result in ADDR does not meet ADCMPR1 setting. 1 = Conversion result in ADDR meets ADCMPR1 setting.

[1]	**CMPF0**	**Compare Flag** When the selected channel A/D conversion result meets setting condition in ADCMPR0 then this bit is set to 1. And it is cleared by writing 1 to self. 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. ADF is set to 1 at these two conditions: 1. When A/D conversion ends in Single mode. 2. When A/D conversion ends on all specified channels in Scan mode. This flag can be cleared by writing 1 to itself.

ADC PDMA Current Transfer Data Register (ADPDMA)

Register	Offset	R/W	Description	Reset Value
ADPDMA	ADC_BA+0x40	R	ADC PDMA Current Transfer Data Register	0x0000_0000

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

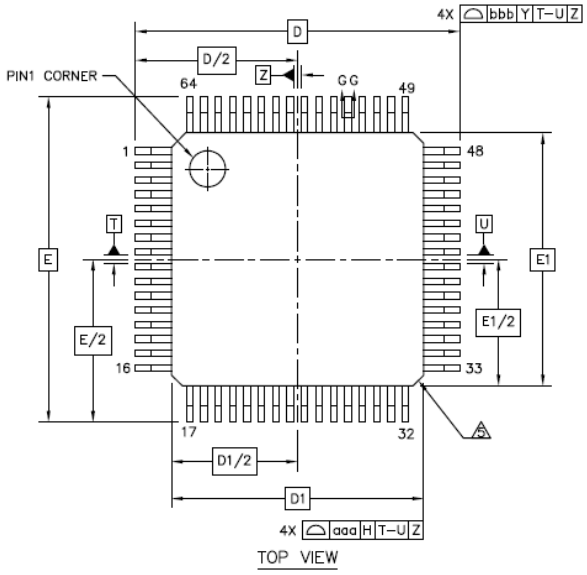
Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:0]	**AD_PDMA**	**ADC PDMA Current Transfer Data Register** When PDMA transferring, read this register can monitor current PDMA transfer data. Current PDMA transfer data is the content of ADDR0 ~ ADDR11. This is a read only register.

7 ELECTRICAL CHARACTERISTICS

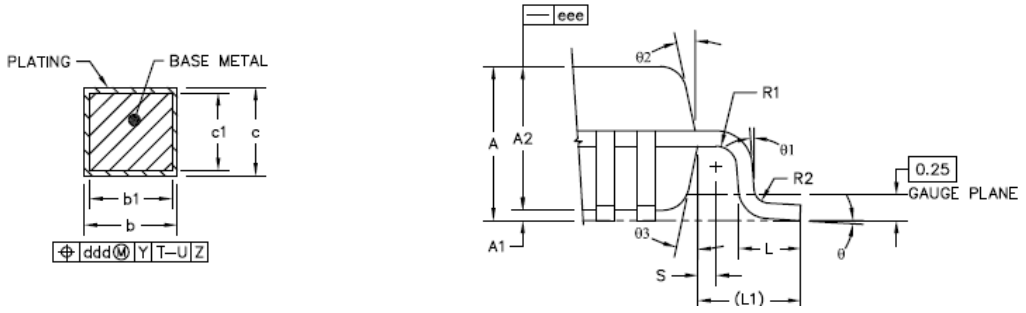
For information on NuMicro® NUC029LEE/NUC029SEE electrical characteristics, please refer to NuMicro® NUC029LEE/NUC029SEE Datasheet.

8 PACKAGE DIMENSIONS

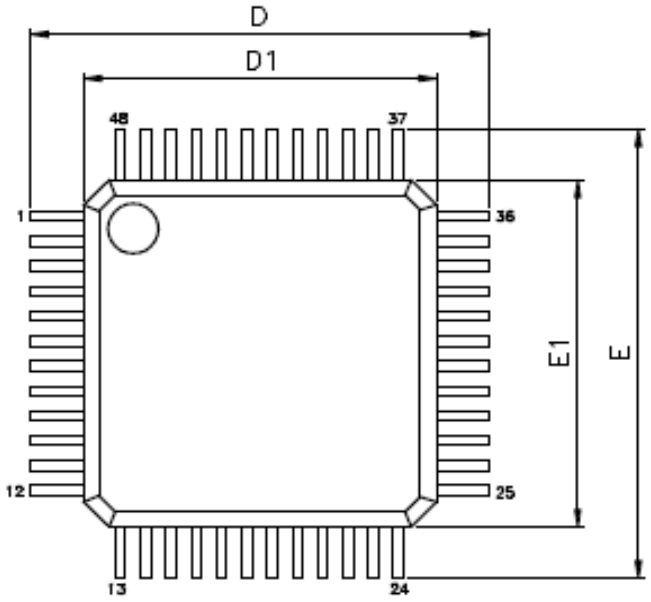
8.1 64-pin LQFP (7x7x1.4 mm footprint 2.0 mm)

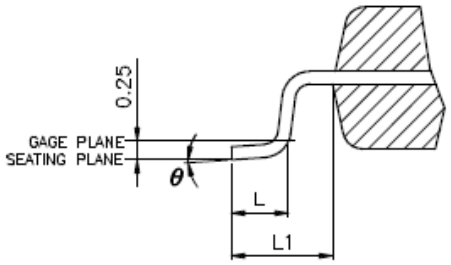


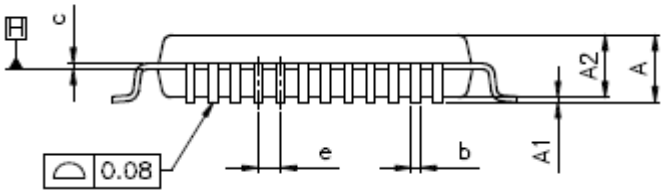
		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	---	---	1.6
STAND OFF		A1	0.05	---	0.15
MOLD THICKNESS		A2	1.35	1.4	1.45
LEAD WIDTH(PLATING)		b	0.13	0.18	0.23
LEAD WIDTH		b1	0.13	0.16	0.19
L/F THICKNESS(PLATING)		c	0.09	---	0.2
L/F THICKNESS		c1	0.09	---	0.16
BODY SIZE	X	D	9 BSC		
	Y	E	9 BSC		
	X	D1	7 BSC		
	Y	E1	7 BSC		
LEAD PITCH		e	0.4 BSC		
		L	0.45	0.6	0.75
FOOTPRINT		L1	1 REF		
		θ	0°	3.5°	7°
		θ1	0°	---	---
		θ2	11°	12°	13°
		θ3	11°	12°	13°
		R1	0.08	---	---
		R2	0.08	---	0.2
		S	0.2	---	---
PACKAGE EDGE TOLERANCE		aaa	0.2		
LEAD EDGE TOLERANCE		bbb	0.2		
COPLANARITY		ccc	0.08		
LEAD OFFSET		ddd	0.07		
MOLD FLATNESS		eee	0.05		



8.2 48-pin LQFP (7x7x1.4 mm footprint 2.0 mm)







VARIATIONS (ALL DIMENSIONS SHOWN IN MM)

SYMBOLS	MIN.	NOM.	MAX.
A	--	--	1.60
A1	0.05	--	0.15
A2	1.35	1.40	1.45
b	0.17	0.22	0.27
c	0.09	--	0.20
D	9.00 BSC		
D1	7.00 BSC		
E	9.00 BSC		
E1	7.00 BSC		
e	0.50 BSC		
L	0.45	0.60	0.75
L1	1.00 REF		
θ	0°	3.5°	7°

9 REVISION HISTORY

Date	Revision	Description
2018.08.20	1.00	Initial version.
2018.12.18	1.01	Added the part number NUC029KGE in section 4.1 and section 4.2.
2020.04.09	1.02	Added external circuit recommendation of ICE_DAT, ICE_CLK, and nReset in section 4.4.1.

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.