ARM® Cortex®-M
32-bit Microcontroller

NuMicro® NUC100 Series
NUC100/120xxxDN
Technical Reference Manual

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

www.nuvoton.com

Table of Contents

List of Figures

List of Tables

1 GENERAL DESCRIPTION

The NuMicro® NUC100 series 32-bit microcontroller (MCU) is embedded with the ARM® Cortex®-M0 core with the cost equivalent to traditional 8-bit MCU. The NUC100 series can be used in consumer electronics, industrial control and applications which requiring rich communication interfaces such as industrial automation, alarm system, energy system and power system.

The NuMicro® NUC100 Advanced Line and NUC120 USB Line are embedded with the Cortex®-M0 core running up to 50 MHz and features 32/64/128 Kbytes Flash, 4/8/16 Kbytes embedded SRAM and 4 Kbytes loader ROM for the ISP. It operates at a wide voltage range of 2.5V ~ 5.5V and temperature range of -40℃ ~ +85℃. The NUC100 series is also provided with plenty of peripheral devices, such as Timers, Watchdog Timer, Window Watchdog Timer, RTC, PDMA with CRC calculation unit, UART, SPI, I^2C, I^2S, PWM Timer, GPIO, PS/2, EBI, Smart Card Host, 12-bit ADC, Analog Comparator, Low Voltage Reset Controller and Brown-out Detector. Additionally, the NUC120 USB Line is equipped with a USB 2.0 Full-speed Device. These peripherals have been incorporated into the NUC100 series to reduce component count, board space and system cost.

The NUC100 series is equipped with ISP (In-System Programming), IAP (In-Application-Programming) and ICP (In-Circuit Programming) functions, which allows the user to update the program under software control through the on-chip connectivity interface, such as SWD, UART and USB.

Product Line	UART	SPI	I^2C	USB	PS/2	I^2S	SC
NUC100xxxDN	3	4	2	-	1	1	3
NUC120xxxDN	3	4	2	1	1	1	3

Table 1-1 NuMicro® NUC100 Series Connectivity Support Table

2 FEATURES

The equipped features are dependent on the product line and their sub products.

2.1 NuMicro® NUC100 Features – Advanced Line

- ARM® Cortex®-M0 core
  - Runs up to 50 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 ranged from 2.5 V to 5.5 V
- Flash Memory
  - 32/64/128 Kbytes Flash for program code
  - 4 KB 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 KB system, fixed 4 KB Data Flash for the 32 KB and 64 KB system
  - Supports 2-wired ICP update through SWD/ICE interface
  - Supports fast parallel programming mode by external programmer
- SRAM Memory
  - 4/8/16 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 ℃ ~ +85 ℃ and V_{DD} = 2.5 V ~ 5.5 V
  - Built-in 10 kHz low speed oscillator for Watchdog Timer and Wake-up operation
  - Supports one PLL, up to 50 MHz, for high performance system operation
  - External 4~24 MHz high speed crystal input for 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
- 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 wake-up function
- PWM/Capture
  - Up to four built-in 16-bit PWM generators providing eight PWM outputs or four 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
  - Up to eight 16-bit digital capture timers (shared with PWM timers) providing eight rising/falling capture inputs
  - Supports Capture interrupt
- UART
  - Up to three UART controllers
  - UART ports with flow control (TXD, RXD, CTS and RTS)
  - 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 PDMA mode
- SPI
  - Up to four sets of SPI controllers
  - SPI clock rate of Master can be up to 36 MHz (chip working at 5V); SPI clock rate of Slave can be 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
- I²C
  - Up to two sets of I²C device
  - 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
- I²S
  - Interface with external audio CODEC
  - Operate as either Master or Slave mode
  - Capable of handling 8-, 16-, 24- and 32-bit word sizes
  - Supports mono and stereo audio data
  - Supports I²S and MSB justified data format
  - Provides two 8 word FIFO data buffers, one for transmitting and the other for receiving
  - Generates interrupt requests when buffer levels cross a programmable boundary
  - Supports two DMA requests, one for transmitting and the other for receiving
- PS/2 Device
  - Host communication inhibit and request to send detection
  - Reception frame error detection
  - Programmable 1 to 16 bytes transmit buffer to reduce CPU intervention
  - Double buffer for data reception
  - Software override bus
- EBI (External bus interface)
  - Accessible space: 64 KB in 8-bit mode or 128 KB in 16-bit mode
  - Supports 8-/16-bit data width
  - Supports byte write in 16-bit data width mode
- ADC
  - 12-bit SAR ADC with 760 kSPS
  - Up to 8-ch single-end input or 4-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 or external input
  - Supports PDMA mode
- Analog Comparator
  - Up to two analog comparators
  - External input or internal Band-gap voltage selectable at negative node
  - Interrupt when compare results change
  - Supports Power-down wake-up
- Smart Card Host (SC)
  - Compliant to ISO-7816-3 T=0, T=1
  - Supports up to three ISO-7816-3 ports

- Separate receive / transmit 4 bytes entry FIFO for data payloads
- Programmable transmission clock frequency
- Programmable receiver buffer trigger level
- Programmable guard time selection (11 ETU ~ 266 ETU)
- One 24-bit and two 8-bit time-out counters for Answer to Request (ATR) and waiting times processing
- Supports auto inverse convention function
- Supports transmitter and receiver error retry and error limit function
- Supports hardware activation sequence process
- Supports hardware warm reset sequence process
- Supports hardware deactivation sequence process
- Supports hardware auto deactivation sequence when detecting the card is removal

- 96-bit unique ID (UID)
- 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℃ ~ 85℃
- Packages:
  - All Green package (RoHS)
  - LQFP 100-pin
  - LQFP 64-pin
  - LQFP 48-pin

2.2 NuMicro® NUC120 Features – USB Line

- ARM® Cortex®-M0 core
  - Runs up to 50 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
  - 32/64/128 Kbytes Flash for program code
  - 4 KB 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 KB system, fixed 4 KB Data Flash for the 32 KB and 64 KB system
  - Supports 2-wired ICP update through SWD/ICE interface
  - Supports fast parallel programming mode by external programmer
- SRAM Memory
  - 4/8/16 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 ℃ ~ +85 ℃ and V_{DD} = 2.5 V ~ 5.5 V
  - Built-in 10 kHz low speed oscillator for Watchdog Timer and Wake-up operation
  - Supports one PLL, up to 50 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
- 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 wake-up function
- PWM/Capture
  - Up to four built-in 16-bit PWM generators providing eight PWM outputs or four 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
  - Up to eight 16-bit digital capture timers (shared with PWM timers) providing eight rising/falling capture inputs
  - Supports Capture interrupt
- UART
  - Up to three UART controllers
  - UART ports with flow control (TXD, RXD, CTS and RTS)
  - 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 PDMA mode
- SPI
  - Up to four 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 device
  - 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
- I^2S
  - Interface with external audio CODEC
  - Operate as either Master or Slave mode
  - Capable of handling 8-, 16-, 24- and 32-bit word sizes
  - Supports mono and stereo audio data
  - Supports I^2S and MSB justified data format
  - Provides two 8 word FIFO data buffers, one for transmitting and the other for receiving
  - Generates interrupt requests when buffer levels cross a programmable boundary
  - Supports two DMA requests, one for transmitting and the other for receiving
- PS/2 Device
  - Host communication inhibit and request to send detection
  - Reception frame error detection
  - Programmable 1 to 16 bytes transmit buffer to reduce CPU intervention
  - Double buffer for data reception
  - Software override bus
- EBI (External bus interface)
  - Accessible space: 64 KB in 8-bit mode or 128 KB in 16-bit mode
  - Supports 8-/16-bit data width
  - Supports byte write in 16-bit data width mode
- 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 6 programmable endpoints
  - Includes 512 Bytes internal SRAM as USB buffer
  - Provides remote wake-up capability
- ADC
  - 12-bit SAR ADC with 760 kSPS
  - Up to 8-ch single-end input or 4-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 or external input
  - Supports PDMA mode

- Analog Comparator
  - Up to two analog comparators
  - External input or internal Band-gap voltage selectable at negative node
  - Interrupt when compare results change
  - Supports Power-down wake-up
- Smart Card Host (SC)
  - Compliant to ISO-7816-3 T=0, T=1
  - Supports up to three ISO-7816-3 ports
  - Separate receive / transmit 4 bytes entry FIFO for data payloads
  - Programmable transmission clock frequency
  - Programmable receiver buffer trigger level
  - Programmable guard time selection (11 ETU ~ 266 ETU)
  - One 24-bit and two 8-bit time-out counters for Answer to Request (ATR) and waiting times processing
  - Supports auto inverse convention function
  - Supports transmitter and receiver error retry and error limit function
  - Supports hardware activation sequence process
  - Supports hardware warm reset sequence process
  - Supports hardware deactivation sequence process
  - Supports hardware auto deactivation sequence when detecting the card removal
- 96-bit unique ID (UID)
- 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℃ ~ 85℃
- Packages:
  - All Green package (RoHS)
  - LQFP 100-pin
  - LQFP 64-pin
  - LQFP48-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 3-1 List of Abbreviations

4 PARTS INFORMATION LIST AND PIN CONFIGURATION

4.1 NuMicro® NUC100/120xxxDN Selection Guide

4.1.1 NuMicro® NUC100 Advanced Line Selection Guide

Part Number	APROM	RAM	Data Flash	ISP Loader ROM	I/O	Timer	Connectivity						I²S	SC	Comp.	PWM	ADC	RTC	EBI	ISP ICP	Package
							UART	SPI	I²C	USB	LIN	CAN									
NUC100LC1DN	32 KB	4 KB	4 KB	4 KB	up to 37	4x32-bit	2	1	2	-	-	-	1	3	1	6	8x12-bit	v	-	v	LQFP48
NUC100LD2DN	64 KB	8 KB	4 KB	4 KB	up to 37	4x32-bit	2	1	2	-	-	-	1	3	1	6	8x12-bit	v	-	v	LQFP48
NUC100LE3DN	128 KB	16 KB	Definable	4 KB	up to 37	4x32-bit	2	1	2	-	-	-	1	3	1	6	8x12-bit	v	-	v	LQFP48
NUC100RC1DN	32 KB	4 KB	4 KB	4 KB	up to 51	4x32-bit	3	2	2	-	-	-	1	3	2	6	8x12-bit	v	v	v	LQFP64
NUC100RD1DN	64 KB	4 KB	4 KB	4 KB	up to 51	4x32-bit	3	2	2	-	-	-	1	3	2	6	8x12-bit	v	v	v	LQFP64
NUC100RD2DN	64 KB	8 KB	4 KB	4 KB	up to 51	4x32-bit	3	2	2	-	-	-	1	3	2	6	8x12-bit	v	v	v	LQFP64
NUC100RE3DN	128 KB	16 KB	Definable	4 KB	up to 51	4x32-bit	3	2	2	-	-	-	1	3	2	6	8x12-bit	v	v	v	LQFP64
NUC100VE3DN	128 KB	16 KB	Definable	4 KB	up to 84	4x32-bit	3	4	2	-	-	-	1	3	2	8	8x12-bit	v	v	v	LQFP100

4.1.2 NuMicro® NUC120 USB Line Selection Guide

Part Number	APROM	RAM	Data Flash	ISP Loader ROM	I/O	Timer	Connectivity						I²S	SC	Comp.	PWM	ADC	RTC	EBI	ISP ICP	Package
							UART	SPI	I²C	USB	LIN	CAN									
NUC120LC1DN	32 KB	4 KB	4 KB	4 KB	up to 33	4x32-bit	2	1	2	1	-	-	1	3	1	4	8x12-bit	v	-	v	LQFP48
NUC120LD2DN	64 KB	8 KB	4 KB	4 KB	up to 33	4x32-bit	2	1	2	1	-	-	1	3	1	4	8x12-bit	v	-	v	LQFP48
NUC120LE3DN	128 KB	16 KB	Definable	4 KB	up to 33	4x32-bit	2	1	2	1	-	-	1	3	1	4	8x12-bit	v	-	v	LQFP48
NUC120RC1DN	32 KB	4 KB	4 KB	4 KB	up to 47	4x32-bit	2	2	2	1	-	-	1	3	2	6	8x12-bit	v	v	v	LQFP64
NUC120RD2DN	64 KB	8 KB	4 KB	4 KB	up to 47	4x32-bit	2	2	2	1	-	-	1	3	2	6	8x12-bit	v	v	v	LQFP64
NUC120RE3DN	128 KB	16 KB	Definable	4 KB	up to 47	4x32-bit	2	2	2	1	-	-	1	3	2	6	8x12-bit	v	v	v	LQFP64
NUC120VE3DN	128 KB	16 KB	Definable	4 KB	up to 80	4x32-bit	3	4	2	1	-	-	1	3	2	8	8x12-bit	v	v	v	LQFP100

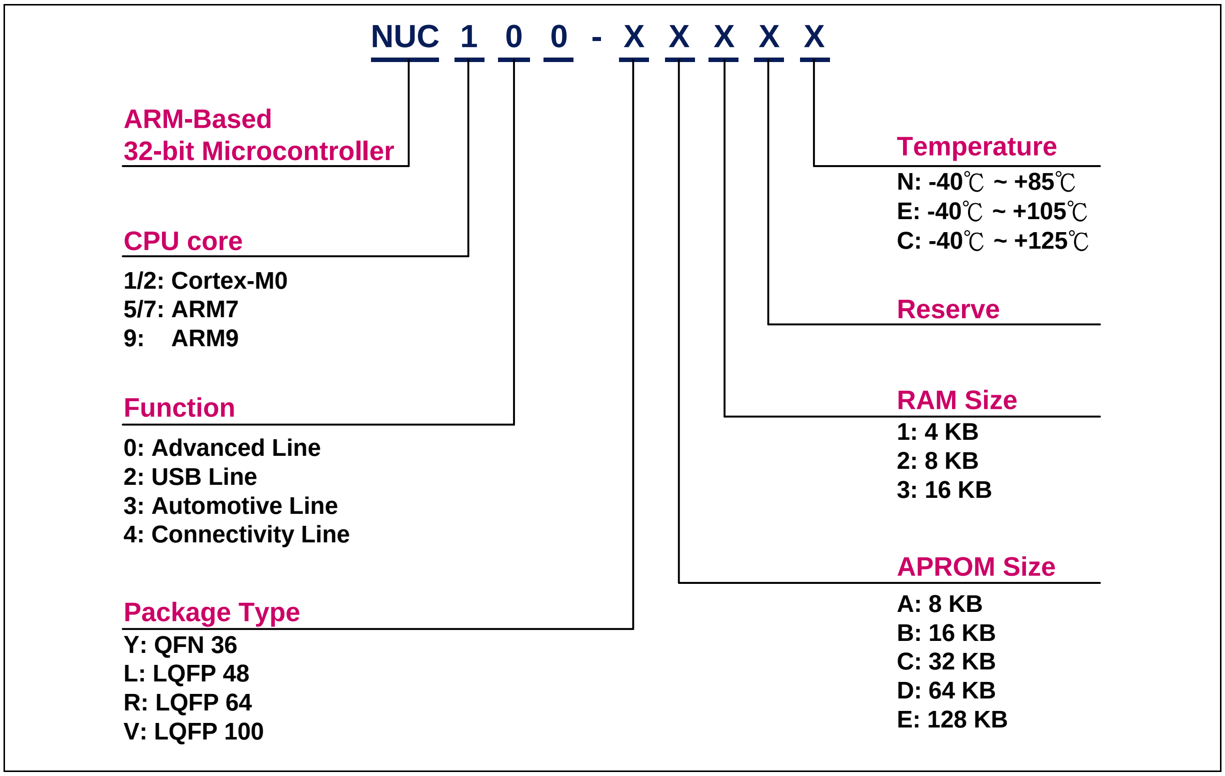


Figure 4-1 NuMicro® NUC100 Series Selection Code

4.2 Pin Configuration

4.2.1 NuMicro® NUC100 Pin Diagram

4.2.1.1 NuMicro® NUC100VxxDN LQFP 100 pin

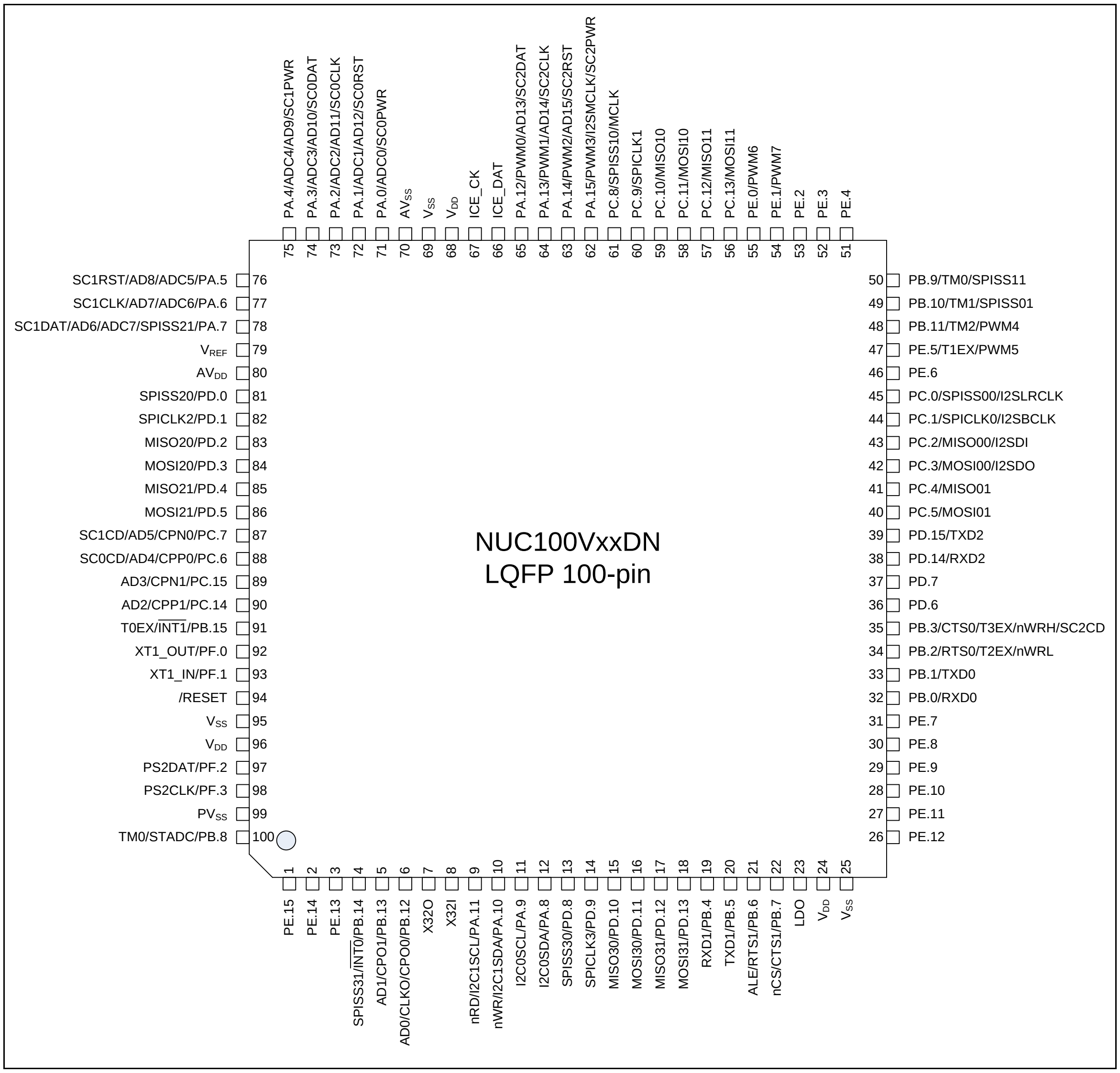


Figure 4-2 NuMicro® NUC100VxxDN LQFP 100-pin Diagram

4.2.1.2 NuMicro® NUC100RxxDN LQFP 64 pin

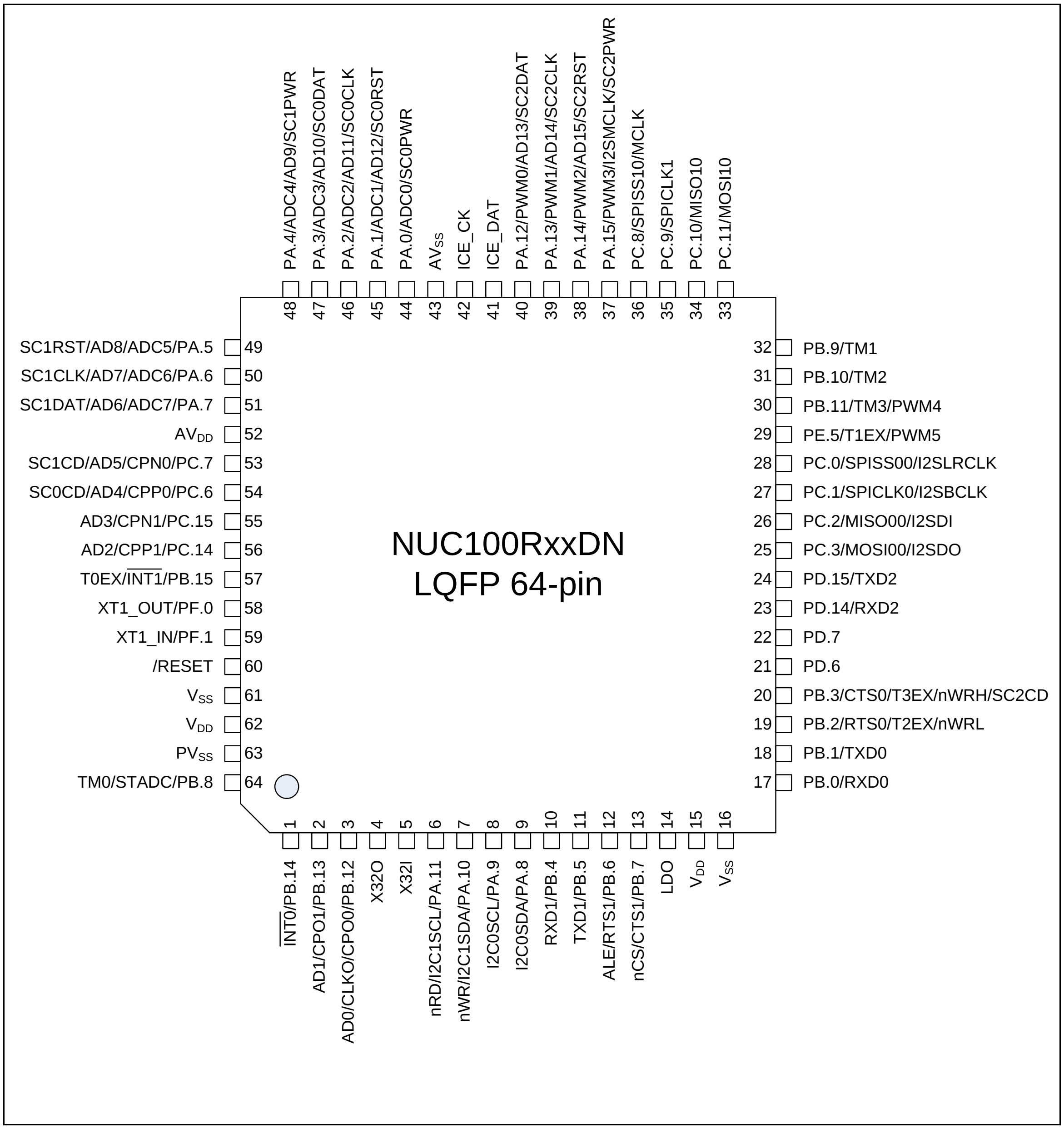


Figure 4-3 NuMicro® NUC100RxxDN LQFP 64-pin Diagram

4.2.1.3 NuMicro® NUC100LxxDN LQFP 48 pin

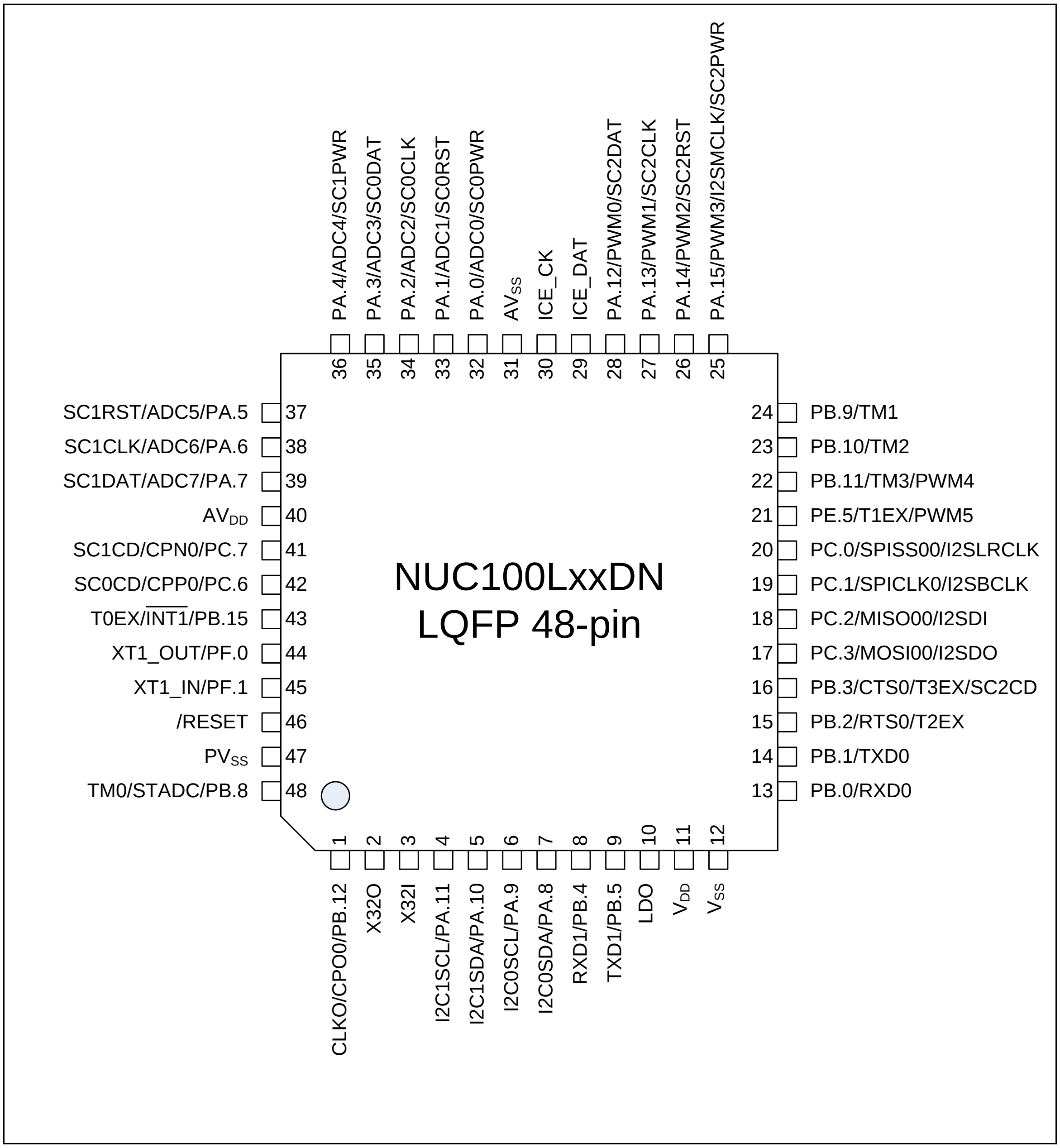


Figure 4-4 NuMicro® NUC100LxxDN LQFP 48-pin Diagram

4.2.2 NuMicro® NUC120 Pin Diagram

4.2.2.1 NuMicro® NUC120VxxDN LQFP 100 pin

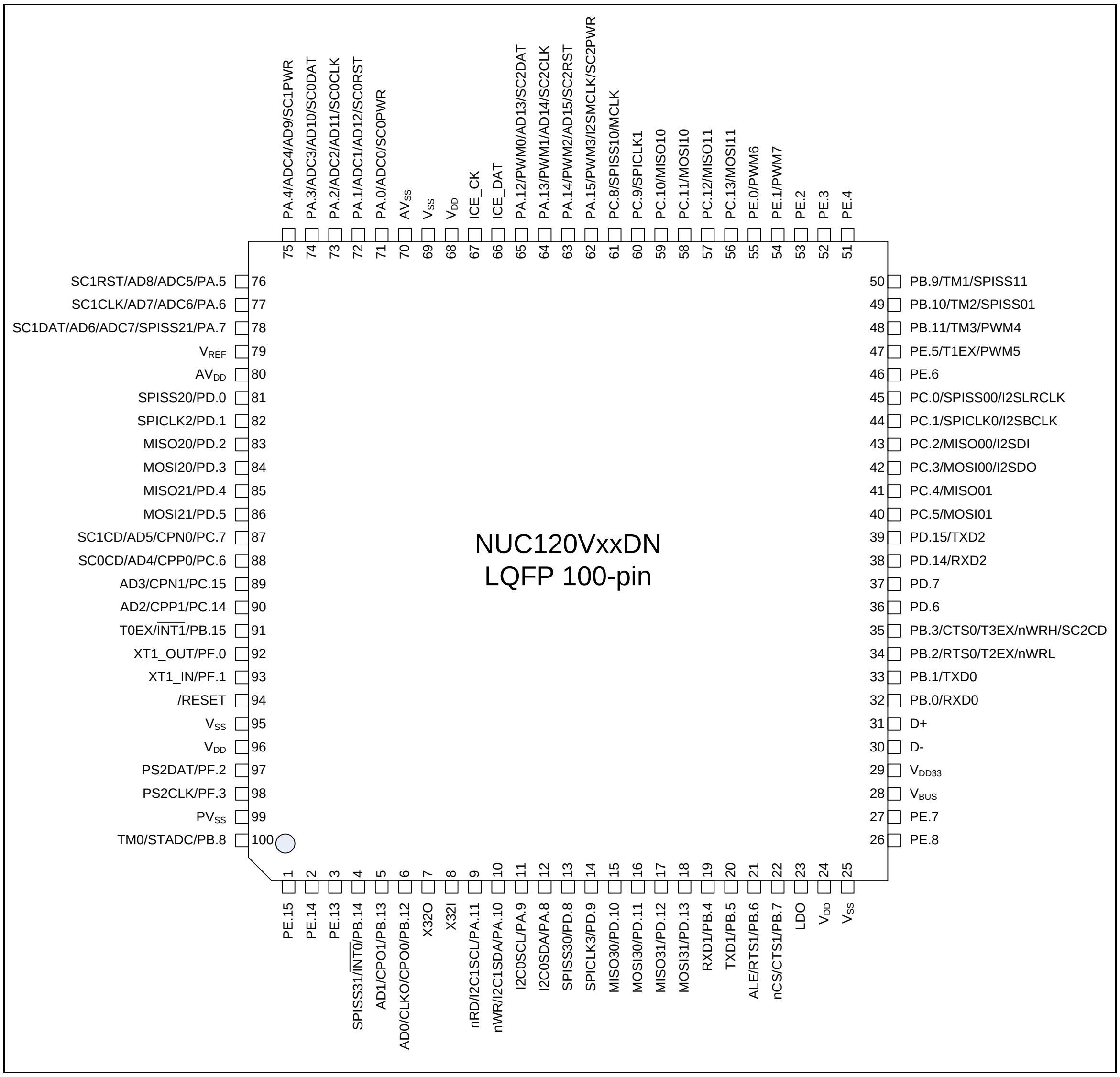


Figure 4-5 NuMicro® NUC120VxxDN LQFP 100-pin Diagram

4.2.2.2 NuMicro® NUC120RxxDN LQFP 64 pin

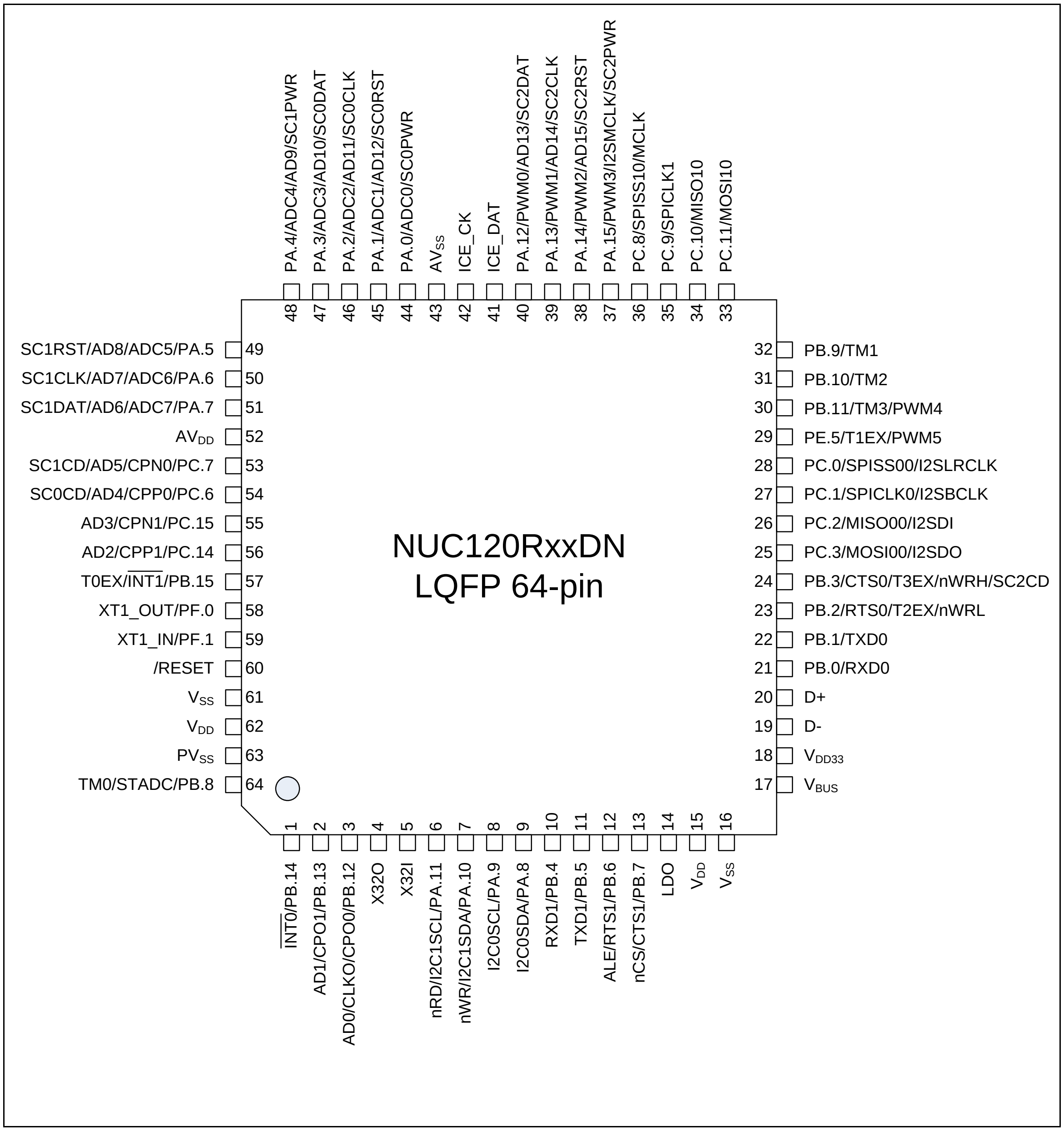


Figure 4-6 NuMicro® NUC120RxxDN LQFP 64-pin Diagram

4.2.2.3 NuMicro® NUC120LxxDN LQFP 48 pin

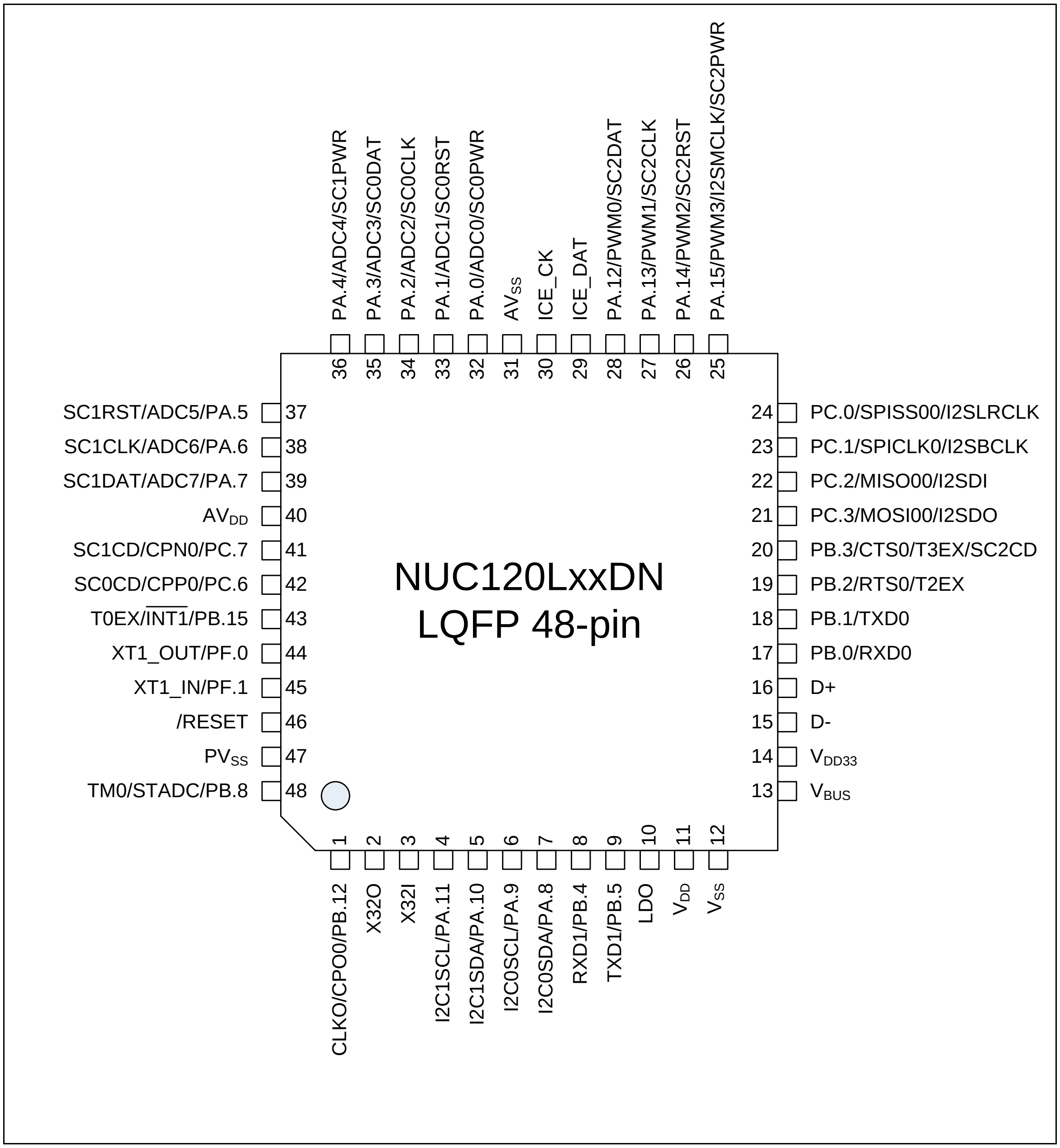


Figure 4-7 NuMicro® NUC120LxxDN LQFP 48-pin Diagram

4.3 Pin Description

4.3.1 NuMicro® NUC100 Pin Description

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
1			PE.15	I/O	General purpose digital I/O pin.
2			PE.14	I/O	General purpose digital I/O pin.
3			PE.13	I/O	General purpose digital I/O pin.
4	1		PB.14	I/O	General purpose digital I/O pin.
			/INT0	I	External interrupt0 input pin.
			SPISS31	I/O	2nd SPI3 slave select pin.
5	2		PB.13	I/O	General purpose digital I/O pin.
			CPO1	O	Comparator1 output pin.
			AD1	I/O	EBI Address/Data bus bit1
6	3	1	PB.12	I/O	General purpose digital I/O pin.
			CPO0	O	Comparator0 output pin
			CLKO	O	Frequency Divider output pin
			AD0	I/O	EBI Address/Data bus bit0
7	4	2	X32O	O	External 32.768 kHz low speed crystal output pin
8	5	3	X32I	I	External 32.768 kHz low speed crystal input pin
9	6	4	PA.11	I/O	General purpose digital I/O pin.
			I2C1SCL	I/O	I^2C1 clock pin.
			nRD	O	EBI read enable output pin
10	7	5	PA.10	I/O	General purpose digital I/O pin.
			I2C1SDA	I/O	I^2C1 data input/output pin.
			nWR	O	EBI write enable output pin
11	8	6	PA.9	I/O	General purpose digital I/O pin.
			I2C0SCL	I/O	I^2C0 clock pin.
12	9	7	PA.8	I/O	General purpose digital I/O pin.
			I2C0SDA	I/O	I^2C0 data input/output pin.
13			PD.8	I/O	General purpose digital I/O pin.
			SPISS30	I/O	1st SPI3 slave select pin.
14			PD.9	I/O	General purpose digital I/O pin.
			SPICLK3	I/O	SPI3 serial clock pin.
15			PD.10	I/O	General purpose digital I/O pin.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
			MISO30	I/O	1st SPI3 MISO (Master In, Slave Out) pin.
16			PD.11	I/O	General purpose digital I/O pin.
			MOSI30	I/O	1st SPI3 MOSI (Master Out, Slave In) pin.
17			PD.12	I/O	General purpose digital I/O pin.
			MISO31	I/O	2nd SPI3 MISO (Master In, Slave Out) pin.
18			PD.13	I/O	General purpose digital I/O pin.
			MOSI31	I/O	2nd SPI3 MOSI (Master Out, Slave In) pin.
19	10	8	PB.4	I/O	General purpose digital I/O pin.
			RXD1	I	Data receiver input pin for UART1.
20	11	9	PB.5	I/O	General purpose digital I/O pin.
			TXD1	O	Data transmitter output pin for UART1.
21	12		PB.6	I/O	General purpose digital I/O pin.
			RTS1	O	Request to Send output pin for UART1.
			ALE	O	EBI address latch enable output pin
22	13		PB.7	I/O	General purpose digital I/O pin.
			CTS1	I	Clear to Send input pin for UART1.
			nCS	O	EBI chip select enable output pin
23	14	10	LDO	P	LDO output pin
24	15	11	V_{DD}	P	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
25	16	12	V_{SS}	P	Ground pin for digital circuit.
26			PE.12	I/O	General purpose digital I/O pin.
27			PE.11	I/O	General purpose digital I/O pin.
28			PE.10	I/O	General purpose digital I/O pin.
29			PE.9	I/O	General purpose digital I/O pin.
30			PE.8	I/O	General purpose digital I/O pin.
31			PE.7	I/O	General purpose digital I/O pin.
32	17	13	PB.0	I/O	General purpose digital I/O pin.
			RXD0	I	Data receiver input pin for UART0.
33	18	14	PB.1	I/O	General purpose digital I/O pin.
			TXD0	O	Data transmitter output pin for UART0.
34	19	15	PB.2	I/O	General purpose digital I/O pin.
			RTS0	O	Request to Send output pin for UART0.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
			T2EX	I	Timer2 external capture input pin.
			nWRL	O	EBI low byte write enable output pin
35	20	16	PB.3	I/O	General purpose digital I/O pin.
			CTS0	I	Clear to Send input pin for UART0.
			T3EX	I	Timer3 external capture input pin.
			SC2CD	I	SmartCard2 card detect pin.
			nWRH	O	EBI high byte write enable output pin
36	21		PD.6	I/O	General purpose digital I/O pin.
37	22		PD.7	I/O	General purpose digital I/O pin.
38	23		PD.14	I/O	General purpose digital I/O pin.
			RXD2	I	Data receiver input pin for UART2.
39	24		PD.15	I/O	General purpose digital I/O pin.
			TXD2	O	Data transmitter output pin for UART2.
40			PC.5	I/O	General purpose digital I/O pin.
			MOSI01	I/O	2nd SPI0 MOSI (Master Out, Slave In) pin.
41			PC.4	I/O	General purpose digital I/O pin.
			MISO01	I/O	2nd SPI0 MISO (Master In, Slave Out) pin.
42	25	17	PC.3	I/O	General purpose digital I/O pin.
			MOSI00	I/O	1st SPI0 MOSI (Master Out, Slave In) pin.
			I2SDO	O	I^2S data output.
43	26	18	PC.2	I/O	General purpose digital I/O pin.
			MISO00	I/O	1st SPI0 MISO (Master In, Slave Out) pin.
			I2SDI	I	I^2S data input.
44	27	19	PC.1	I/O	General purpose digital I/O pin.
			SPICLK0	I/O	SPI0 serial clock pin.
			I2SBCLK	I/O	I^2S bit clock pin.
45	28	20	PC.0	I/O	General purpose digital I/O pin.
			SPISS00	I/O	1st SPI0 slave select pin.
			I2SLRCLK	I/O	I^2S left right channel clock.
46			PE.6	I/O	General purpose digital I/O pin.
47	29	21	PE.5	I/O	General purpose digital I/O pin.
			PWM5	I/O	PWM5 output/Capture input.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
			T1EX	I	Timer1 external capture input pin.
48	30	22	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.
49	31	23	PB.10	I/O	General purpose digital I/O pin.
			TM2	I/O	Timer2 event counter input / toggle output.
			SPISS01	I/O	2nd SPI0 slave select pin.
50	32	24	PB.9	I/O	General purpose digital I/O pin.
			TM1	I/O	Timer1 event counter input / toggle output.
			SPISS11	I/O	2nd SPI1 slave select pin.
51			PE.4	I/O	General purpose digital I/O pin.
52			PE.3	I/O	General purpose digital I/O pin.
53			PE.2	I/O	General purpose digital I/O pin.
54			PE.1	I/O	General purpose digital I/O pin.
			PWM7	I/O	PWM7 output/Capture input.
55			PE.0	I/O	General purpose digital I/O pin.
			PWM6	I/O	PWM6 output/Capture input.
56			PC.13	I/O	General purpose digital I/O pin.
			MOSI11	I/O	2nd SPI1 MOSI (Master Out, Slave In) pin.
57			PC.12	I/O	General purpose digital I/O pin.
			MISO11	I/O	2nd SPI1 MISO (Master In, Slave Out) pin.
58	33		PC.11	I/O	General purpose digital I/O pin.
			MOSI10	I/O	1st SPI1 MOSI (Master Out, Slave In) pin.
59	34		PC.10	I/O	General purpose digital I/O pin.
			MISO10	I/O	1st SPI1 MISO (Master In, Slave Out) pin.
60	35		PC.9	I/O	General purpose digital I/O pin.
			SPICLK1	I/O	SPI1 serial clock pin.
61	36		PC.8	I/O	General purpose digital I/O pin.
			SPISS10	I/O	1st SPI1 slave select pin.
			MCLK	O	EBI external clock output pin
62	37	25	PA.15	I/O	General purpose digital I/O pin.
			PWM3	I/O	PWM output/Capture input.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
			I2SMCLK	O	I^2S master clock output pin.
			SC2PWR	O	SmartCard2 power pin.
63	38	26	PA.14	I/O	General purpose digital I/O pin.
			PWM2	I/O	PWM2 output/Capture input.
			SC2RST	O	SmartCard2 reset pin.
			AD15	I/O	EBI Address/Data bus bit15
64	39	27	PA.13	I/O	General purpose digital I/O pin.
			PWM1	I/O	PWM1 output/Capture input.
			SC2CLK	O	SmartCard2 clock pin.
			AD14	I/O	EBI Address/Data bus bit14
65	40	28	PA.12	I/O	General purpose digital I/O pin.
			PWM0	I/O	PWM0 output/Capture input.
			SC2DAT	O	SmartCard2 data pin.
			AD13	I/O	EBI Address/Data bus bit13
66	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
67	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
68			V_{DD}	P	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
69			V_{SS}	P	Ground pin for digital circuit.
70	43	31	AV_{SS}	AP	Ground pin for analog circuit.
71	44	32	PA.0	I/O	General purpose digital I/O pin.
			ADC0	AI	ADC0 analog input.
			SC0PWR	O	SmartCard0 power pin.
72	45	33	PA.1	I/O	General purpose digital I/O pin.
			ADC1	AI	ADC1 analog input.
			SC0RST	O	SmartCard0 reset pin.
			AD12	I/O	EBI Address/Data bus bit12
73	46	34	PA.2	I/O	General purpose digital I/O pin.
			ADC2	AI	ADC2 analog input.
			SC0CLK	O	SmartCard0 clock pin.
			AD11	I/O	EBI Address/Data bus bit11

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
74	47	35	PA.3	I/O	General purpose digital I/O pin.
			ADC3	AI	ADC3 analog input.
			SC0DAT	O	SmartCard0 data pin.
			AD10	I/O	EBI Address/Data bus bit10
75	48	36	PA.4	I/O	General purpose digital I/O pin.
			ADC4	AI	ADC4 analog input.
			SC1PWR	O	SmartCard1 power pin.
			AD9	I/O	EBI Address/Data bus bit9
76	49	37	PA.5	I/O	General purpose digital I/O pin.
			ADC5	AI	ADC5 analog input.
			SC1RST	O	SmartCard1 reset pin.
			AD8	I/O	EBI Address/Data bus bit8
77	50	38	PA.6	I/O	General purpose digital I/O pin.
			ADC6	AI	ADC6 analog input.
			SC1CLK	I/O	SmartCard1 clock pin.
			AD7	I/O	EBI Address/Data bus bit7
78	51	39	PA.7	I/O	General purpose digital I/O pin.
			ADC7	AI	ADC7 analog input.
			SC1DAT	O	SmartCard1 data pin.
			SPISS21	I/O	2nd SPI2 slave select pin.
			AD6	I/O	EBI Address/Data bus bit6
79			V_{REF}	AP	Voltage reference input for ADC.
80	52	40	AV_{DD}	AP	Power supply for internal analog circuit.
81			PD.0	I/O	General purpose digital I/O pin.
			SPISS20	I/O	1st SPI2 slave select pin.
82			PD.1	I/O	General purpose digital I/O pin.
			SPICLK2	I/O	SPI2 serial clock pin.
83			PD.2	I/O	General purpose digital I/O pin.
			MISO20	I/O	1st SPI2 MISO (Master In, Slave Out) pin.
84			PD.3	I/O	General purpose digital I/O pin.
			MOSI20	I/O	1st SPI2 MOSI (Master Out, Slave In) pin.
85			PD.4	I/O	General purpose digital I/O pin.
			MISO21	I/O	2nd SPI2 MISO (Master In, Slave Out) pin.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
86			PD.5	I/O	General purpose digital I/O pin.
			MOSI21	I/O	2nd SPI2 MOSI (Master Out, Slave In) pin.
87	53	41	PC.7	I/O	General purpose digital I/O pin.
			CPN0	AI	Comparator0 negative input pin.
			SC1CD	I	SmartCard1 card detect pin.
			AD5	I/O	EBI Address/Data bus bit5
88	54	42	PC.6	I/O	General purpose digital I/O pin.
			CPP0	AI	Comparator0 positive input pin.
			SC0CD	I	SmartCard0 card detect pin.
			AD4	I/O	EBI Address/Data bus bit4
89	55		PC.15	I/O	General purpose digital I/O pin.
			CPN1	AI	Comparator1 negative input pin.
			AD3	I/O	EBI Address/Data bus bit3
90	56		PC.14	I/O	General purpose digital I/O pin.
			CPP1	AI	Comparator1 positive input pin.
			AD2	I/O	EBI Address/Data bus bit2
91	57	43	PB.15	I/O	General purpose digital I/O pin.
			/INT1	I	External interrupt1 input pin.
			T0EX	I	Timer0 external capture input pin.
92	58	44	PF.0	I/O	General purpose digital I/O pin.
			XT1_OUT	O	External 4~24 MHz (high speed) crystal output pin.
93	59	45	PF.1	I/O	General purpose digital I/O pin.
			XT1_IN	I	External 4~24 MHz (high speed) crystal input pin.
94	60	46	/RESET	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.
95	61		V_{SS}	P	Ground pin for digital circuit.
96	62		V_{DD}	P	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
97			PF.2	I/O	General purpose digital I/O pin.
			PS2DAT	I/O	PS/2 data pin.
98			PF.3	I/O	General purpose digital I/O pin.
			PS2CLK	I/O	PS/2 clock pin.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
99	63	47	PV_{SS}	**P**	PLL ground.
100	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.

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

4.3.2 NuMicro® NUC120 Pin Description

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
1			PE.15	I/O	General purpose digital I/O pin.
2			PE.14	I/O	General purpose digital I/O pin.
3			PE.13	I/O	General purpose digital I/O pin.
4	1		PB.14	I/O	General purpose digital I/O pin.
			/INT0	I	External interrupt0 input pin.
			SPISS31	I/O	2nd SPI3 slave select pin.
5	2		PB.13	I/O	General purpose digital I/O pin.
			CPO1	O	Comparator1 output pin.
			AD1	I/O	EBI Address/Data bus bit1
6	3	1	PB.12	I/O	General purpose digital I/O pin.
			CPO0	O	Comparator0 output pin
			CLKO	O	Frequency Divider output pin
			AD0	I/O	EBI Address/Data bus bit0
7	4	2	X32O	O	External 32.768 kHz low speed crystal output pin
8	5	3	X32I	I	External 32.768 kHz low speed crystal input pin
9	6	4	PA.11	I/O	General purpose digital I/O pin.
			I2C1SCL	I/O	I^2C1 clock pin.
			nRD	O	EBI read enable output pin
10	7	5	PA.10	I/O	General purpose digital I/O pin.
			I2C1SDA	I/O	I^2C1 data input/output pin.
			nWR	O	EBI write enable output pin
11	8	6	PA.9	I/O	General purpose digital I/O pin.
			I2C0SCL	I/O	I^2C0 clock pin.
12	9	7	PA.8	I/O	General purpose digital I/O pin.
			I2C0SDA	I/O	I^2C0 data input/output pin.
13			PD.8	I/O	General purpose digital I/O pin.
			SPISS30	I/O	1st SPI3 slave select pin.
14			PD.9	I/O	General purpose digital I/O pin.
			SPICLK3	I/O	SPI3 serial clock pin.
15			PD.10	I/O	General purpose digital I/O pin.
			MISO30	I/O	1st SPI3 MISO (Master In, Slave Out) pin.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
16			PD.11	I/O	General purpose digital I/O pin.
			MOSI30	I/O	1st SPI3 MOSI (Master Out, Slave In) pin.
17			PD.12	I/O	General purpose digital I/O pin.
			MISO31	I/O	2nd SPI3 MISO (Master In, Slave Out) pin.
18			PD.13	I/O	General purpose digital I/O pin.
			MOSI31	I/O	2nd SPI3 MOSI (Master Out, Slave In) pin.
19	10	8	PB.4	I/O	General purpose digital I/O pin.
			RXD1	I	Data receiver input pin for UART1.
20	11	9	PB.5	I/O	General purpose digital I/O pin.
			TXD1	O	Data transmitter output pin for UART1.
21	12		PB.6	I/O	General purpose digital I/O pin.
			RTS1	O	Request to Send output pin for UART1.
			ALE	O	EBI address latch enable output pin
22	13		PB.7	I/O	General purpose digital I/O pin.
			CTS1	I	Clear to Send input pin for UART1.
			nCS	O	EBI chip select enable output pin
23	14	10	LDO	P	LDO output pin
24	15	11	V_{DD}	P	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
25	16	12	V_{SS}	P	Ground pin for digital circuit.
26			PE.8	I/O	General purpose digital I/O pin.
27			PE.7	I/O	General purpose digital I/O pin.
28	17	13	VBUS	USB	Power supply from USB host or HUB.
29	18	14	V_{DD33}	USB	Internal power regulator output 3.3V decoupling pin.
30	19	15	D-	USB	USB differential signal D-.
31	20	16	D+	USB	USB differential signal D+.
32	21	17	PB.0	I/O	General purpose digital I/O pin.
			RXD0	I	Data receiver input pin for UART0.
33	22	18	PB.1	I/O	General purpose digital I/O pin.
			TXD0	O	Data transmitter output pin for UART0.
34	23	29	PB.2	I/O	General purpose digital I/O pin.
			RTS0	O	Request to Send output pin for UART0.
			T2EX	I	Timer2 external capture input pin.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
			nWRL	O	EBI low byte write enable output pin
35	24	20	PB.3	I/O	General purpose digital I/O pin.
			CTS0	I	Clear to Send input pin for UART0.
			T3EX	I	Timer3 external capture input pin.
			SC2CD	I	SmartCard2 card detect pin.
			nWRH	O	EBI high byte write enable output pin
36			PD.6	I/O	General purpose digital I/O pin.
37			PD.7	I/O	General purpose digital I/O pin.
38			PD.14	I/O	General purpose digital I/O pin.
			RXD2	I	Data receiver input pin for UART2.
39			PD.15	I/O	General purpose digital I/O pin.
			TXD2	O	Data transmitter output pin for UART2.
40			PC.5	I/O	General purpose digital I/O pin.
			MOSI01	I/O	2nd SPI0 MOSI (Master Out, Slave In) pin.
41			PC.4	I/O	General purpose digital I/O pin.
			MISO01	I/O	2nd SPI0 MISO (Master In, Slave Out) pin.
42	25	21	PC.3	I/O	General purpose digital I/O pin.
			MOSI00	I/O	1st SPI0 MOSI (Master Out, Slave In) pin.
			I2SDO	O	I^2S data output.
43	26	22	PC.2	I/O	General purpose digital I/O pin.
			MISO00	I/O	1st SPI0 MISO (Master In, Slave Out) pin.
			I2SDI	I	I^2S data input.
44	27	23	PC.1	I/O	General purpose digital I/O pin.
			SPICLK0	I/O	SPI0 serial clock pin.
			I2SBCLK	I/O	I^2S bit clock pin.
45	28	24	PC.0	I/O	General purpose digital I/O pin.
			SPISS00	I/O	1st SPI0 slave select pin.
			I2SLRCLK	I/O	I^2S left right channel clock.
46			PE.6	I/O	General purpose digital I/O pin.
47	29		PE.5	I/O	General purpose digital I/O pin.
			PWM5	I/O	PWM5 output/Capture input.
			T1EX	I	Timer1 external capture input pin.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
48	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.
49	31		PB.10	I/O	General purpose digital I/O pin.
			TM2	I/O	Timer2 event counter input / toggle output.
			SPISS01	I/O	2nd SPI0 slave select pin.
50	32		PB.9	I/O	General purpose digital I/O pin.
			TM1	I/O	Timer1 event counter input / toggle output.
			SPISS11	I/O	2nd SPI1 slave select pin.
51			PE.4	I/O	General purpose digital I/O pin.
52			PE.3	I/O	General purpose digital I/O pin.
53			PE.2	I/O	General purpose digital I/O pin.
54			PE.1	I/O	General purpose digital I/O pin.
			PWM7	I/O	PWM7 output/Capture input.
55			PE.0	I/O	General purpose digital I/O pin.
			PWM6	I/O	PWM6 output/Capture input.
56			PC.13	I/O	General purpose digital I/O pin.
			MOSI11	I/O	2nd SPI1 MOSI (Master Out, Slave In) pin.
57			PC.12	I/O	General purpose digital I/O pin.
			MISO11	I/O	2nd SPI1 MISO (Master In, Slave Out) pin.
58	33		PC.11	I/O	General purpose digital I/O pin.
			MOSI10	I/O	1st SPI1 MOSI (Master Out, Slave In) pin.
59	34		PC.10	I/O	General purpose digital I/O pin.
			MISO10	I/O	1st SPI1 MISO (Master In, Slave Out) pin.
60	35		PC.9	I/O	General purpose digital I/O pin.
			SPICLK1	I/O	SPI1 serial clock pin.
61	36		PC.8	I/O	General purpose digital I/O pin.
			SPISS10	I/O	1st SPI1 slave select pin.
			MCLK	O	EBI external clock output pin
62	37	25	PA.15	I/O	General purpose digital I/O pin.
			PWM3	I/O	PWM output/Capture input.
			I2SMCLK	O	I^2S master clock output pin.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
			SC2PWR	**O**	SmartCard2 power pin.
63	38	26	PA.14	**I/O**	General purpose digital I/O pin.
			PWM2	**I/O**	PWM2 output/Capture input.
			SC2RST	**O**	SmartCard2 reset pin.
			AD15	**I/O**	EBI Address/Data bus bit15
64	39	27	PA.13	**I/O**	General purpose digital I/O pin.
			PWM1	**I/O**	PWM1 output/Capture input.
			SC2CLK	**O**	SmartCard2 clock pin.
			AD14	**I/O**	EBI Address/Data bus bit14
65	40	28	PA.12	**I/O**	General purpose digital I/O pin.
			PWM0	**I/O**	PWM0 output/Capture input.
			SC2DAT	**O**	SmartCard2 data pin.
			AD13	**I/O**	EBI Address/Data bus bit13
66	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
67	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
68			V_{DD}	**P**	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
69			V_{SS}	**P**	Ground pin for digital circuit.
70	43	31	AV_{SS}	**AP**	Ground pin for analog circuit.
71	44	32	PA.0	**I/O**	General purpose digital I/O pin.
			ADC0	**AI**	ADC0 analog input.
			SC0PWR	**O**	SmartCard0 power pin.
72	45	33	PA.1	**I/O**	General purpose digital I/O pin.
			ADC1	**AI**	ADC1 analog input.
			SC0RST	**O**	SmartCard0 reset pin.
			AD12	**I/O**	EBI Address/Data bus bit12
73	46	34	PA.2	**I/O**	General purpose digital I/O pin.
			ADC2	**AI**	ADC2 analog input.
			SC0CLK	**O**	SmartCard0 clock pin.
			AD11	**I/O**	EBI Address/Data bus bit11
74	47	35	PA.3	**I/O**	General purpose digital I/O pin.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
			ADC3	AI	ADC3 analog input.
			SC0DAT	O	SmartCard0 data pin.
			AD10	I/O	EBI Address/Data bus bit10
75	48	36	PA.4	I/O	General purpose digital I/O pin.
			ADC4	AI	ADC4 analog input.
			SC1PWR	O	SmartCard1 power pin.
			AD9	I/O	EBI Address/Data bus bit9
76	49	37	PA.5	I/O	General purpose digital I/O pin.
			ADC5	AI	ADC5 analog input.
			SC1RST	O	SmartCard1 reset pin.
			AD8	I/O	EBI Address/Data bus bit8
77	50	38	PA.6	I/O	General purpose digital I/O pin.
			ADC6	AI	ADC6 analog input.
			SC1CLK	I/O	SmartCard1 clock pin.
			AD7	I/O	EBI Address/Data bus bit7
78	51	39	PA.7	I/O	General purpose digital I/O pin.
			ADC7	AI	ADC7 analog input.
			SC1DAT	O	SmartCard1 data pin.
			SPISS21	I/O	2nd SPI2 slave select pin.
			AD6	I/O	EBI Address/Data bus bit6
79			V_{REF}	AP	Voltage reference input for ADC.
80	52	40	AV_{DD}	AP	Power supply for internal analog circuit.
81			PD.0	I/O	General purpose digital I/O pin.
			SPISS20	I/O	1st SPI2 slave select pin.
82			PD.1	I/O	General purpose digital I/O pin.
			SPICLK2	I/O	SPI2 serial clock pin.
83			PD.2	I/O	General purpose digital I/O pin.
			MISO20	I/O	1st SPI2 MISO (Master In, Slave Out) pin.
84			PD.3	I/O	General purpose digital I/O pin.
			MOSI20	I/O	1st SPI2 MOSI (Master Out, Slave In) pin.
85			PD.4	I/O	General purpose digital I/O pin.
			MISO21	I/O	2nd SPI2 MISO (Master In, Slave Out) pin.
86			PD.5	I/O	General purpose digital I/O pin.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
			MOSI21	I/O	2nd SPI2 MOSI (Master Out, Slave In) pin.
87	53	41	PC.7	I/O	General purpose digital I/O pin.
			CPN0	AI	Comparator0 negative input pin.
			SC1CD	I	SmartCard1 card detect pin.
			AD5	I/O	EBI Address/Data bus bit5
88	54	42	PC.6	I/O	General purpose digital I/O pin.
			CPP0	AI	Comparator0 positive input pin.
			SC0CD	I	SmartCard0 card detect pin.
			AD4	I/O	EBI Address/Data bus bit4
89	55		PC.15	I/O	General purpose digital I/O pin.
			CPN1	AI	Comparator1 negative input pin.
			AD3	I/O	EBI Address/Data bus bit3
90	56		PC.14	I/O	General purpose digital I/O pin.
			CPP1	AI	Comparator1 positive input pin.
			AD2	I/O	EBI Address/Data bus bit2
91	57	43	PB.15	I/O	General purpose digital I/O pin.
			/INT1	I	External interrupt1 input pin.
			T0EX	I	Timer0 external capture input pin.
92	58	44	PF.0	I/O	General purpose digital I/O pin.
			XT1_OUT	O	External 4~24 MHz (high speed) crystal output pin.
93	59	45	PF.1	I/O	General purpose digital I/O pin.
			XT1_IN	I	External 4~24 MHz (high speed) crystal input pin.
94	60	46	/RESET	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.
95	61		V_{SS}	P	Ground pin for digital circuit.
96	62		V_{DD}	P	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
97			PF.2	I/O	General purpose digital I/O pin.
			PS2DAT	I/O	PS/2 data pin.
98			PF.3	I/O	General purpose digital I/O pin.
			PS2CLK	I/O	PS/2 clock pin.
99	63	47	PV_{SS}	P	PLL ground.

Pin No.			Pin Name	Pin Type	Description
LQFP 100-pin	LQFP 64-pin	LQFP 48-pin			
100	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.

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

5 BLOCK DIAGRAM

5.1 NuMicro® NUC100 Block Diagram

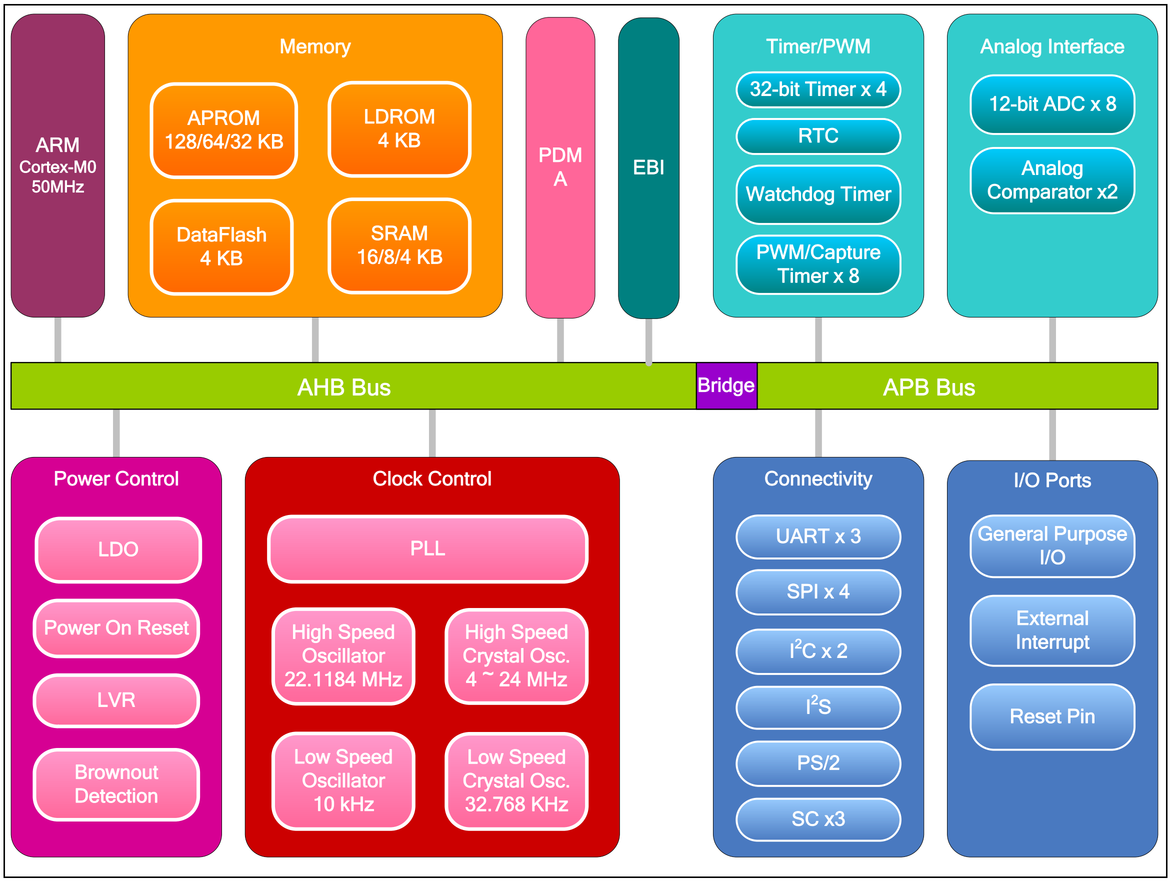


Figure 5-1 NuMicro® NUC100 Block Diagram

5.2 NuMicro® NUC120 Block Diagram

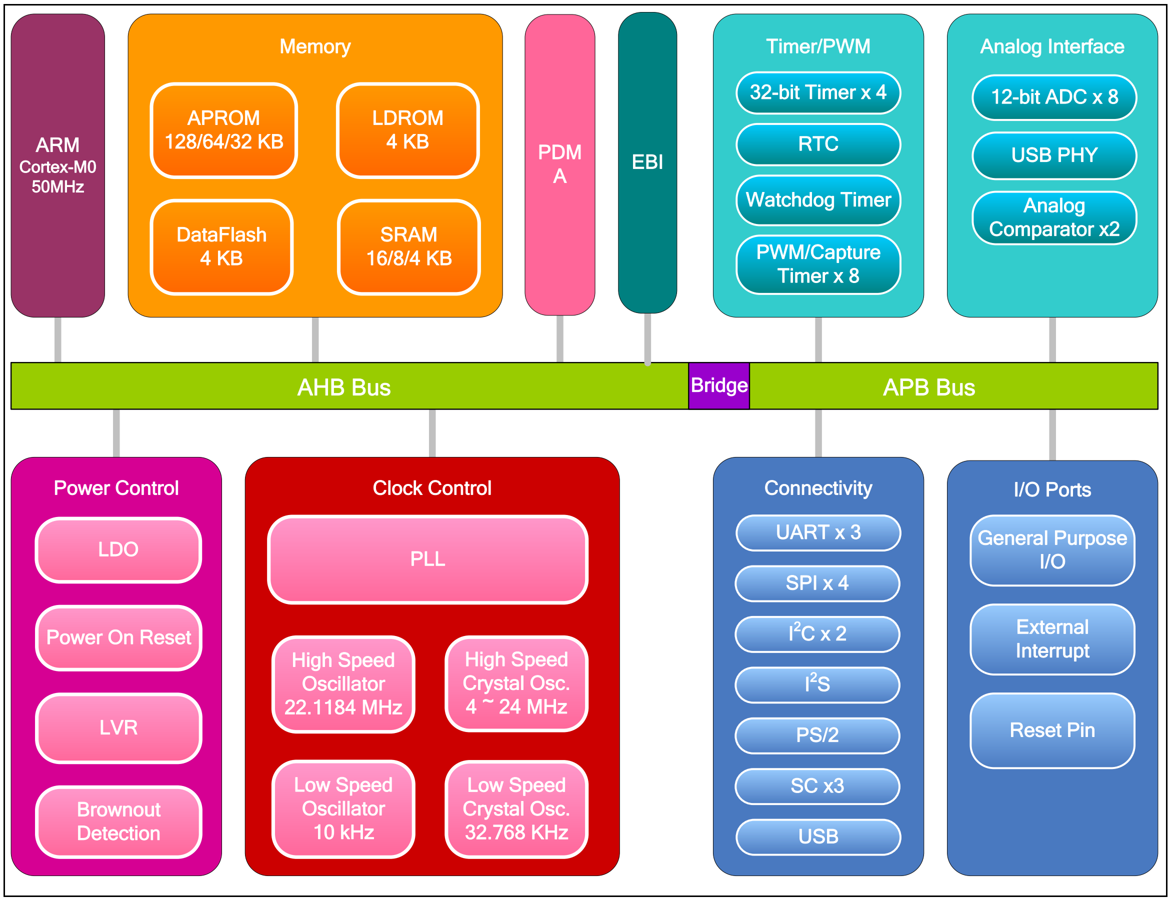


Figure 5-2 NuMicro® NUC120 Block Diagram

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 shows the functional controller of processor.

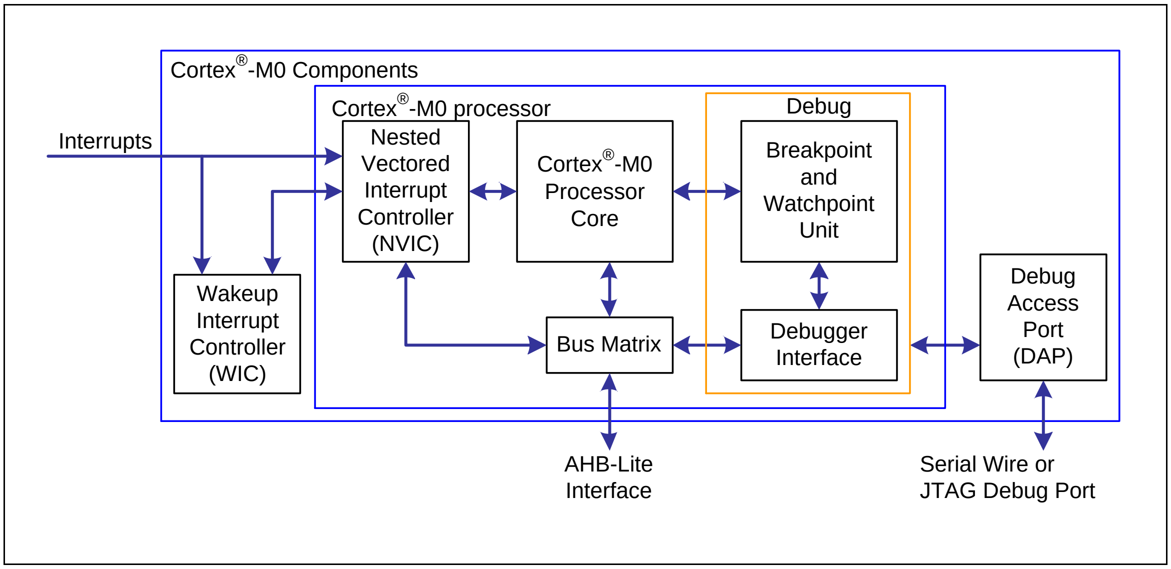


Figure 6-1 Functional Controller Diagram

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

The system manager provides the functions of system control, power modes, wake-up sources, reset sources, system memory map, product ID and multi-function pin control. The following sections describe the functions for

- System Reset
- System Power Architecture
- System Memory Map
- System management registers for Part Number ID, chip reset and on-chip controllers reset, and 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 events listed below. These reset event flags can be read from RSTSRC register to determine the reset source. Hardware reset can reset chip through peripheral reset signals. Software reset can trigger reset through control registers.

- Hardware Reset Sources
  - Power-on Reset (POR)
  - Low level on the nRESET pin
  - Watchdog Time-out Reset and Window Watchdog Reset (WDT/WWDT Reset)
  - Low Voltage Reset (LVR)
  - Brown-out Detector Reset (BOD Reset)
- Software Reset Sources
  - CHIP Reset will reset whole chip by writing 1 to CHIP_RST (IPRSTC1[0])
  - MCU Reset to reboot but keeping the booting setting from APROM or LDROM by writing 1 to SYSRESETREQ (AIRCR[2])
  - CPU Reset for Cortex®-M0 core Only by writing 1 to CPU_RST (IPRSTC1[1])

Power-on Reset or CHIP_RST (IPRSTC1[0]) reset the whole chip including all peripherals, external crystal circuit and BS (ISPCON[1]) bit.

SYSRESETREQ (AIRCR[2]) reset the whole chip including all peripherals, but does not reset external crystal circuit and BS (ISPCON[1]) bit.

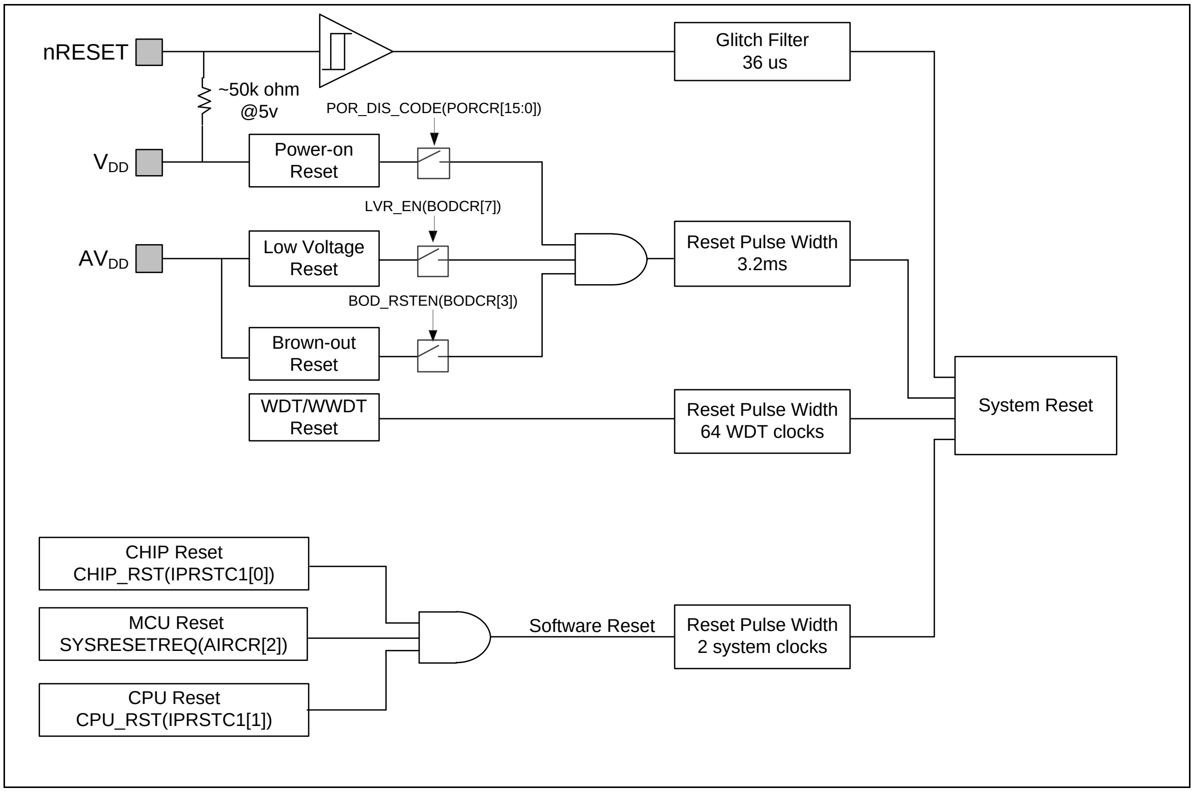


Figure 6-2 System Reset Resources

There are a total of 8 reset sources in the NuMicro® family. In general, CPU reset is used to reset Cortex®-M0 only; the other reset sources will reset Cortex®-M0 and all peripherals. However, there are small differences between each reset source and they are listed in Table 6-1.

Reset Sources / Register	POR	nRESET	WDT	LVR	BOD	CHIP	MCU	CPU
RSTSRC	Bit 0 = 1	Bit 1 = 1	Bit 2 = 1	Bit 3 = 1	Bit 4 = 1	Bit 0 = 1	Bit 5 = 1	Bit 7 = 1
CHIP_RST (IPRSTC1[0])	0x0	-	-	-	-	-	-	-
BOD_EN (BODCR[0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	Reload from CONFIG0	-
BOD_VL (BODCR[2:1])								
BOD_RSTEN (BODCR[3])								
XTL12M_EN (PWRCON[0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-
WDT_EN (APBCLK[0])	0x1	-	0x1	-	-	0x1	-	-
HCLK_S (CLKSEL0[2:0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-

WDT_S (CLKSEL1[1:0])	0x3	0x3	-	-	-	-	-	-
XTL12M_STB (CLKSTATUS[0])	0x0	-	-	-	-	-	-	-
XTL32K_STB (CLKSTATUS[1])	0x0	-	-	-	-	-	-	-
PLL_STB (CLKSTATUS[2])	0x0	-	-	-	-	-	-	-
OSC10K_STB (CLKSTATUS[3])	0x0	-	-	-	-	-	-	-
OSC22M_STB (CLKSTATUS[4])	0x0	-	-	-	-	-	-	-
CLK_SW_FAIL (CLKSTATUS[7])	0x0	0x0	0x0	0x0	0x0	0x0	0x0	-
WTE (WTCR[7])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	-
WTCR	0x0700	0x0700	0x0700	0x0700	0x0700	0x0700	-	-
WTCRALT	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	-	-
WWDTRLD	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	-	-
WWDTCR	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	-	-
WWDTSR	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	-	-
WWDTCVR	0x3F	0x3F	0x3F	0x3F	0x3F	0x3F	-	-
BS (ISPCON[1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	-
DFBADR	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	-	-
CBS (ISPSTA[2:1))	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	-
VECMAP (ISPSTA[20:9])	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	-	-
Other Peripheral Registers	Reset Value							-
FMC Registers	Reset Value							
Note: '-' means that the value of register keeps original setting.								

Table 6-1 Reset Value of Registers

6.2.2.1 nRESET Reset

The nRESET reset means to generate a reset signal by pulling low nRESET pin, which is an asynchronous reset input pin and can be used to reset system at any time. When the nRESET voltage is lower than 0.2 V_{DD} and the state keeps longer than 36 us (glitch filter), chip will be reset. The nRESET reset will control the chip in reset state until the nRESET voltage rises above 0.7 V_{DD} and the state keeps longer than 36 us (glitch filter). The RSTS_RESET (RSTSRC[1]) will be set to 1 if the previous reset source is nRESET reset. Figure 6-3 shows the nRESET reset waveform.

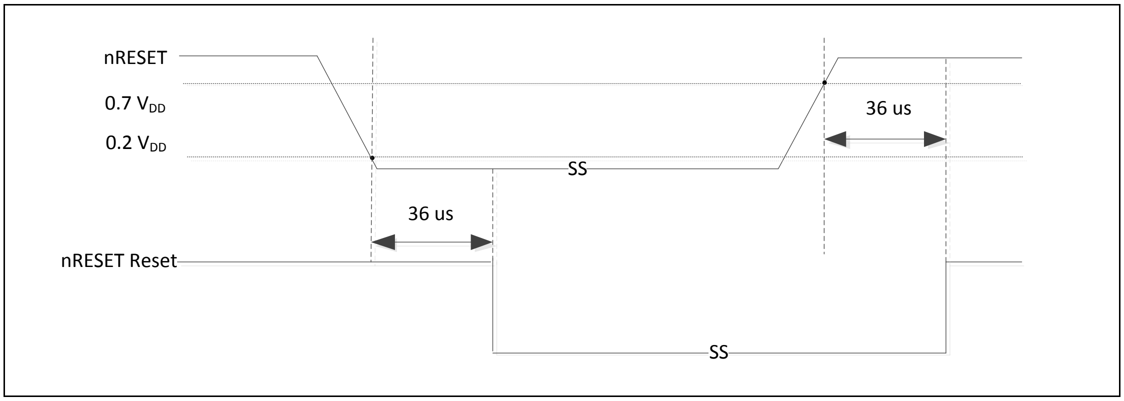


Figure 6-3 nRESET Reset Waveform

6.2.2.2 Power-On Reset (POR)

The Power-on reset (POR) is used to generate a stable system reset signal and forces the system to be reset when power-on to avoid unexpected behavior of MCU. When applying the power to MCU, the POR module will detect the rising voltage and generate reset signal to system until the voltage is ready for MCU operation. At POR reset, the RSTS_POR (RSTSRC[0]) will be set to 1 to indicate there is a POR reset event. The RSTS_POR (RSTSRC[0]) bit can be cleared by writing 1 to it. Figure 6-4 shows the waveform of Power-On reset.

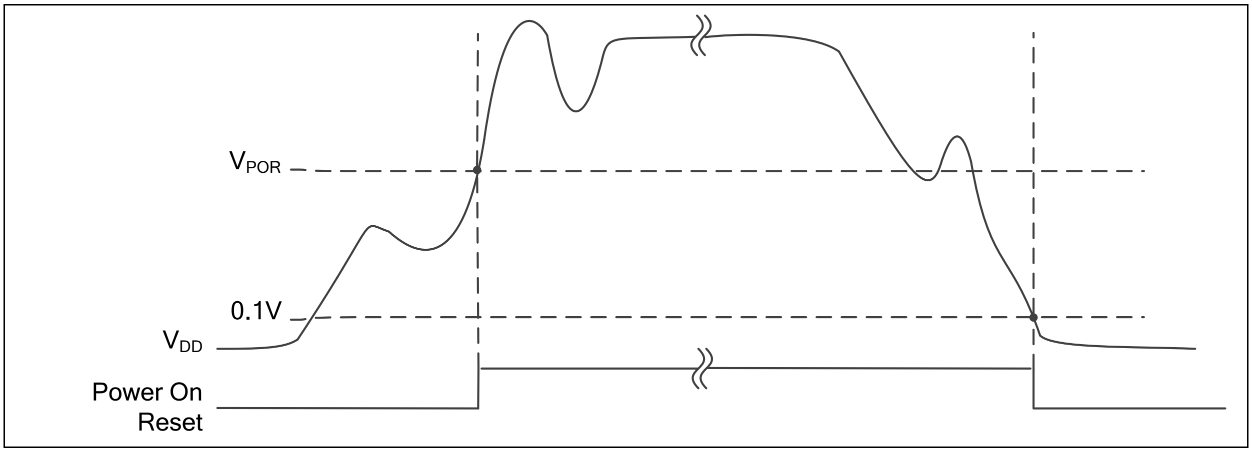


Figure 6-4 Power-on Reset (POR) Waveform

6.2.2.3 Low Voltage Reset (LVR)

If the Low Voltage Reset function is enabled by setting the Low Voltage Reset Enable Bit LVR_EN (BODCR[7]) to 1, after 100us delay, LVR detection circuit will be stable and the LVR function will be active. Then LVR function will detect AV_{DD} during system operation. When the

AV_{DD} voltage is lower than V_{LVR} and the state keeps longer than De-glitch time (16*HCLK cycles), chip will be reset. The LVR reset will control the chip in reset state until the AV_{DD} voltage rises above V_{LVR} and the state keeps longer than De-glitch time. The RSTS_RESET (RSTSRC[1]) will be set to 1 if the previous reset source is nRESET reset. Figure 6-5 shows the Low Voltage Reset waveform.

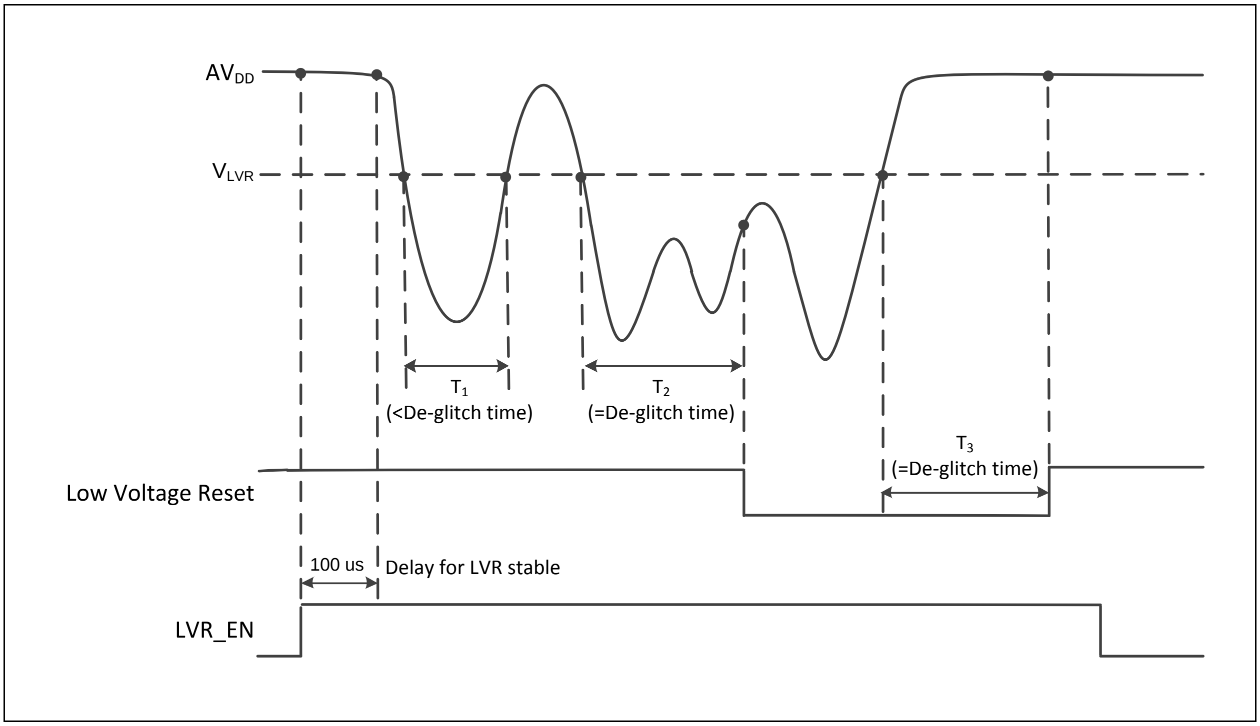


Figure 6-5 Low Voltage Reset (LVR) Waveform

6.2.2.4 Brown-out Detector Reset (BOD Reset)

If the Brown-out Detector (BOD) function is enabled by setting the Brown-out Detector Enable Bit BOD_EN (BODCR[0]), Brown-Out Detector function will detect AV_{DD} during system operation. When the AV_{DD} voltage is lower than V_{BOD} which is decided by BOD_EN (BODCR[0]) and BOD_VL (BODCR[2:1]) and the state keeps longer than De-glitch time (Max(20*HCLK cycles, 1*LIRC cycle)), chip will be reset. The BOD reset will control the chip in reset state until the AV_{DD} voltage rises above V_{BOD} and the state keeps longer than De-glitch time. The default value of BOD_EN, BOD_VL and BOD_RSTEN is set by flash controller user configuration register CBODEN (CONFIG0[23]), CBOV1-0 (CONFIG0[22:21]) and CBORST (CONFIG0[20]) respectively. User can determine the initial BOD setting by setting the CONFIG0 register. Figure 6-6 shows the Brown-Out Detector waveform.

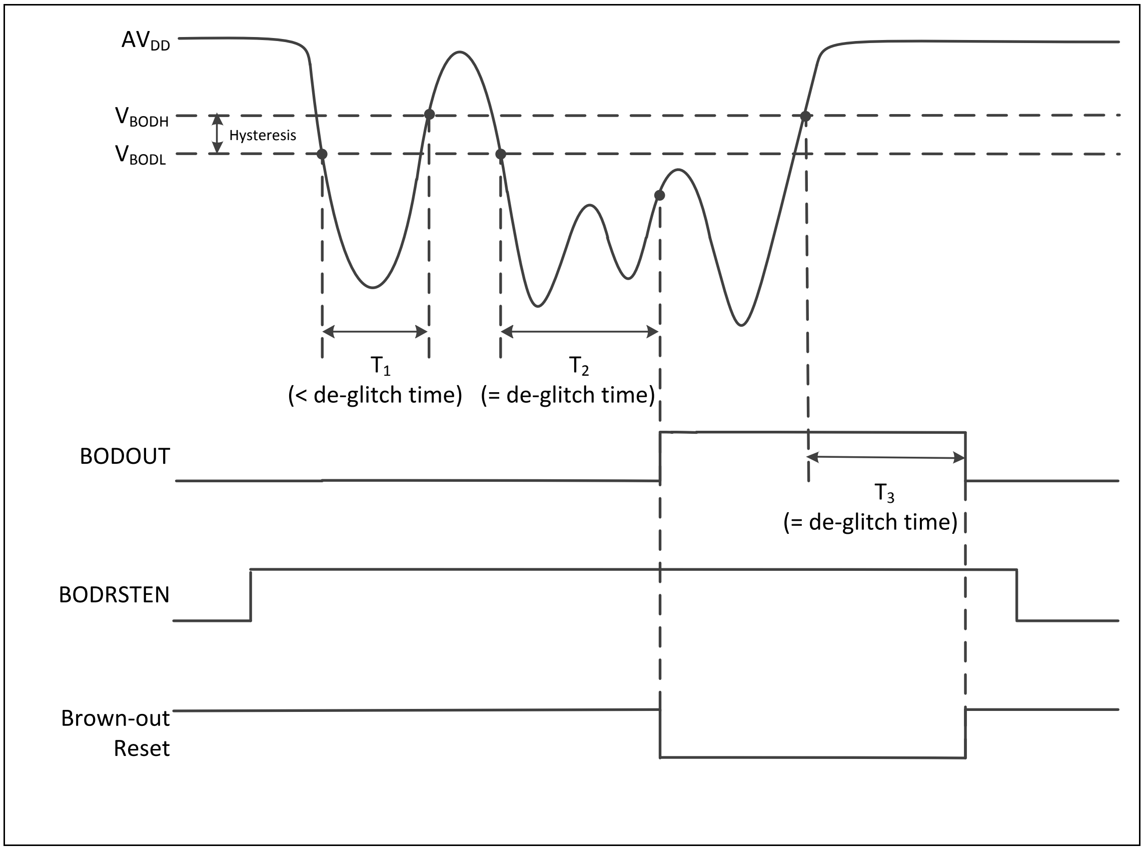


Figure 6-6 Brown-Out Detector (BOD) Waveform

6.2.2.5 Watchdog Timer Reset

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

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

6.2.2.6 CPU Reset, CHIP Reset and MCU Reset

The CPU Reset means only Cortex®-M0 core is reset and all other peripherals remain the same status after CPU reset. User can set the CPU Reset CPU_RST (IPRSTC1[1]) to 1 to assert the CPU Reset signal.

The CHIP Reset is same with Power-On Reset. The CPU and all peripherals are reset and BS (ISPCON[1]) bit is automatically reloaded from CONFIG0 setting. User can set the CHIP Reset CHIP_RST (IPRSTC1[0]) to 1 to assert the CHIP Reset signal.

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

6.2.3 Power Modes and Wake-up Sources

There are several wake-up sources in Idle mode and Power-down mode. Table 6-2 lists the available clocks for each power mode.

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

Table 6-2 Power Mode Difference Table

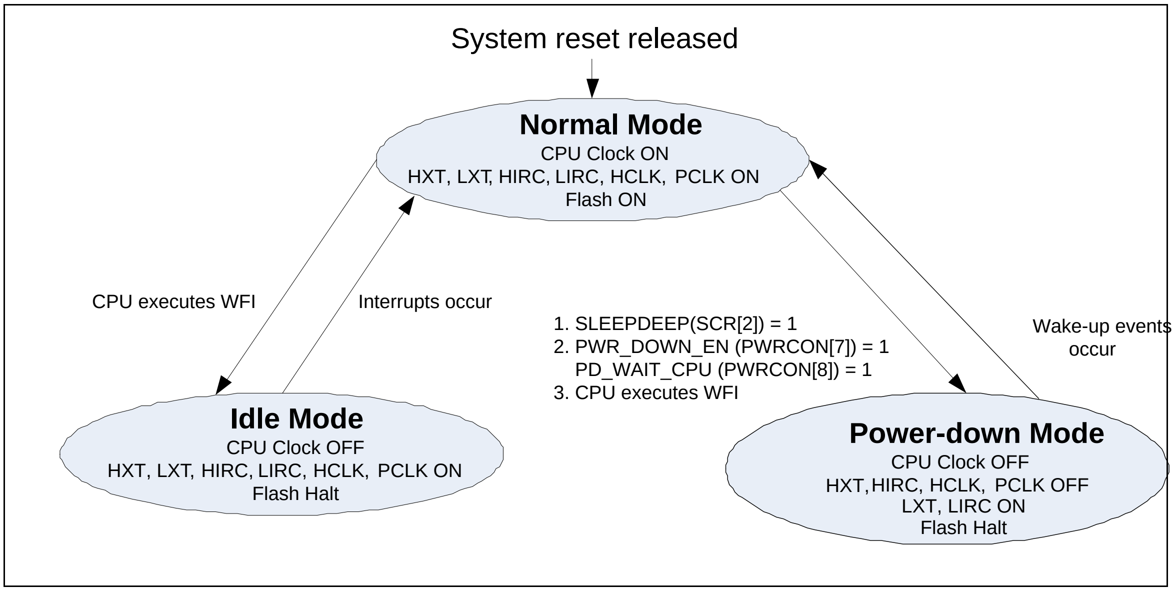


Figure 6-7 Power Mode State Machine

1. LXT (32 kHz XTL) ON or OFF depends on S/W setting in run mode.

2. LIRC (10 kHz OSC) ON or OFF depends on S/W setting in run mode.

3. If TIMER clock source is selected as LXT/LIRC and LXT/LIRC is on.

4. If PWM clock source is selected as LXT and LXT is on.

5. If WDT clock source is selected as LXT/LIRC and LXT/LIRC is on.

6. If RTC clock source LXT is on.

	Normal Mode	Idle Mode	Power-down Mode
HXT (4~20 MHz XTL)	ON	ON	Halt
HIRC (12/16 MHz OSC)	ON	ON	Halt
LXT (32 kHz XTL)	ON	ON	ON/OFF[1]
LIRC (10 kHz OSC)	ON	ON	ON/OFF[2]
PLL	ON	ON	Halt
LDO	ON	ON	ON
CPU	ON	Halt	Halt
HCLK/PCLK	ON	ON	Halt
SRAM retention	ON	ON	ON
FLASH	ON	ON	Halt
EBI	ON	ON	Halt
GPIO	ON	ON	Halt
PDMA	ON	ON	Halt
TIMER	ON	ON	ON/OFF[3]
PWM	ON	ON	ON/OFF[4]
WDT	ON	ON	ON/OFF[5]
WWDT	ON	ON	Halt
RTC	ON	ON	ON/OFF[6]
UART	ON	ON	Halt
SC	ON	ON	Halt
PS/2	ON	ON	Halt
I^2C	ON	ON	Halt
SPI	ON	ON	Halt
I^2S	ON	ON	Halt
USB	ON	ON	Halt
ADC	ON	ON	Halt
ACMP	ON	ON	Halt

Table 6-3 Clocks in Power Modes

Wake-up sources in Power-down mode:

WDT, I^2C, Timer, RTC, UART, BOD, GPIO and USB

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

Wake-Up Source	Wake-Up Condition	System Can Enter Power-Down Mode Again Condition*
BOD	Brown-Out Detector Interrupt	After software writes 1 to clear BOD_INTF (BODCR[4]).
GPIO	GPIO Interrupt	After software write 1 to clear the ISRC[n] bit.
TIMER	Timer Interrupt	After software writes 1 to clear TWF (TISRx[1]) and TIF (TISRx[0]).
WDT	WDT Interrupt	After software writes 1 to clear WTWKF (WTCR[5]) (Write Protect).
RTC	Alarm Interrupt	After software writes 1 to clear AIF (RIIR[0]).
	Time Tick Interrupt	After software writes 1 to clear TIF (RIIR[1]).
UART	nCTS wake-up	After software writes 1 to clear DCTSF (UA_MSR[0]).
I^2C	Addressing I^2C device	After software writes 1 to clear WKUPIF (I2CWKUPSTS[0]).
USB	Remote Wake-up	After software writes 1 to clear BUS_STS (USBD_INTSTS[0]).

*User needs to wait this condition before setting PWR_DOWN_EN (PWRCON[7]) and execute WFI to enter Power-down mode.

Table 6-4*User needs to wait this condition before setting PWR_DOWN_EN (PWRCON[7]) and execute WFI to enter Power-down mode.

Table 6-4 lists the condition about how to enter Power-down mode again for each peripheral.

Wake-Up Source	Wake-Up Condition	System Can Enter Power-Down Mode Again Condition*
BOD	Brown-Out Detector Interrupt	After software writes 1 to clear BOD_INTF (BODCR[4]).
GPIO	GPIO Interrupt	After software write 1 to clear the ISRC[n] bit.
TIMER	Timer Interrupt	After software writes 1 to clear TWF (TISRx[1]) and TIF (TISRx[0]).
WDT	WDT Interrupt	After software writes 1 to clear WTWKF (WTCR[5]) (Write Protect).
RTC	Alarm Interrupt	After software writes 1 to clear AIF (RIIR[0]).
	Time Tick Interrupt	After software writes 1 to clear TIF (RIIR[1]).
UART	nCTS wake-up	After software writes 1 to clear DCTSF (UA_MSR[0]).
I^2C	Addressing I^2C device	After software writes 1 to clear WKUPIF (I2CWKUPSTS[0]).
USB	Remote Wake-up	After software writes 1 to clear BUS_STS (USBD_INTSTS[0]).

*User needs to wait this condition before setting PWR_DOWN_EN (PWRCON[7]) and execute WFI to enter Power-down mode.

Table 6-4 Condition of Entering Power-down Mode Again

6.2.4 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.

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 of the digital power (V_{DD}). Figure 6-8 shows the power distribution of NuMicro® NUC100. Figure 6-9 shows the power distribution of NuMicro® NUC120.

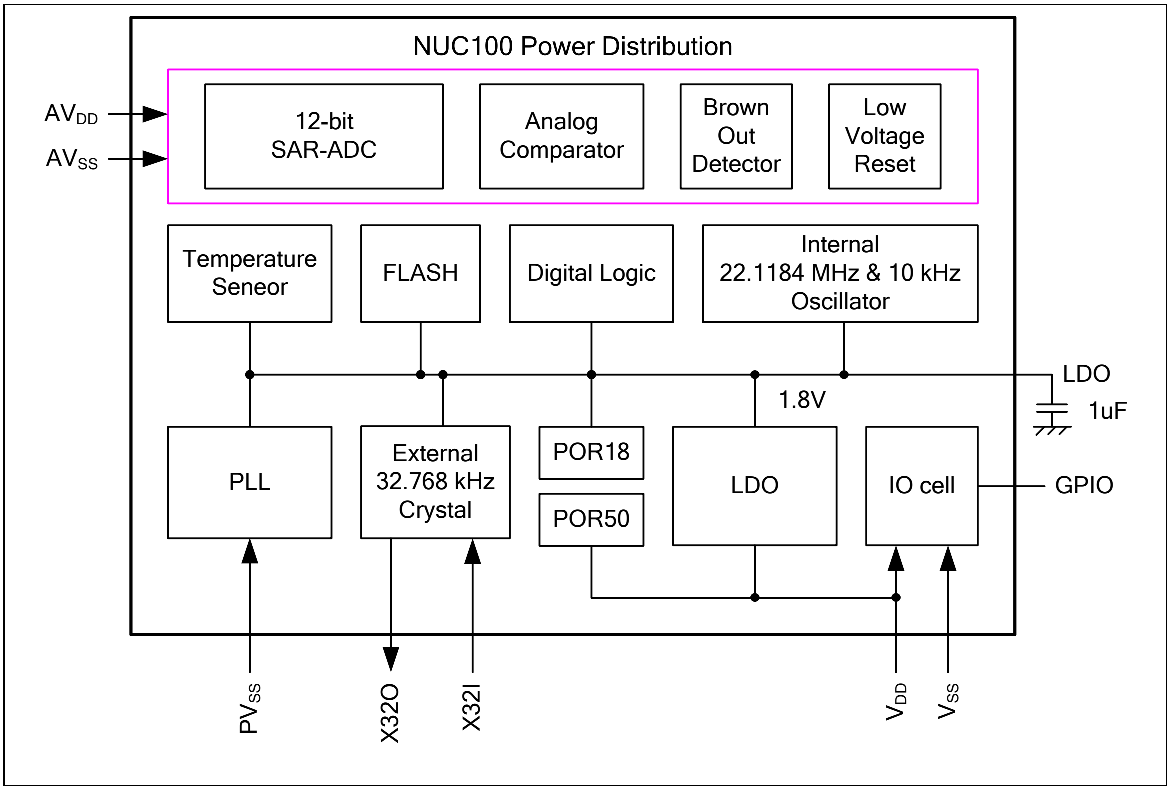


Figure 6-8 NuMicro® NUC100 Power Distribution Diagram

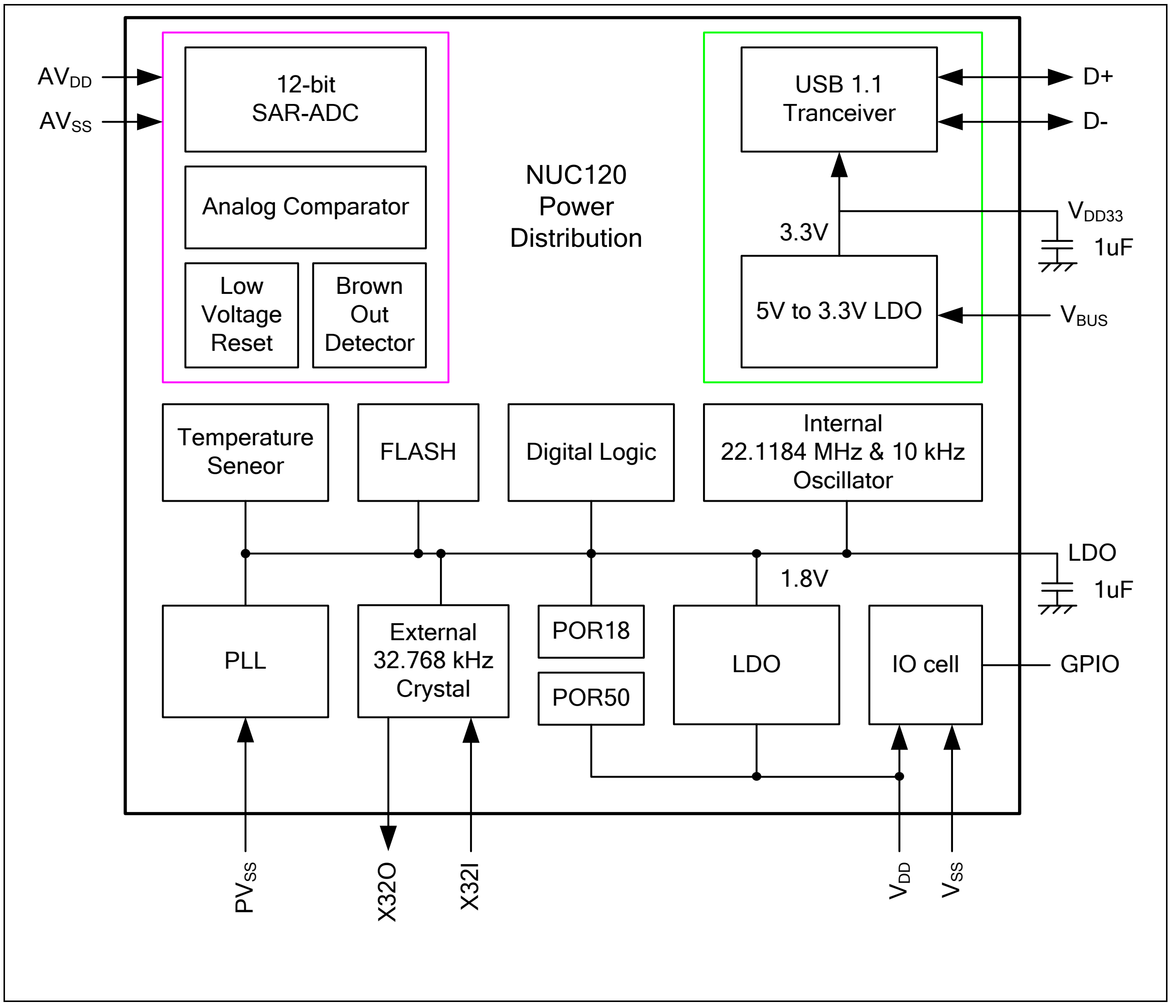


Figure 6-9 NuMicro® NUC120 Power Distribution Diagram

6.2.5 Register Protection

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 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 REGPROTDIS (REGWRPROT [0]), “1” is protection disable, “0” is protection enable. Then user can update the target protected register value and then write any data to REGWRPROT to enable register protection.

The protected registers are listed as following table.

Register	Bit	Description
IPRSTC1	[3] EBI_RST	EBI Controller Reset (Write Protect)
	[2] PDMA_RST	PDMA Controller Reset (Write Protect)
	[1] CPU_RST	CPU Core One-shot Reset (Write Protect)
	[0] CHIP_RST	CHIP One-shot Reset (Write Protect)
BODCR	[7] LVR_EN	Low Voltage Reset Enable Bit (Write Protect)
	[5] BOD_LPM	Brown-out Detector Low Power Mode (Write Protect)
	[3] BOD_RSTEN	Brown-out Reset Enable Bit (Write Protect)
	[2:1] BOD_VL	Brown-out Detector Threshold Voltage Selection (Write Protect)
	[0] BOD_EN	Brown-out Detector Enable Bit (Write Protect)
PORCR	[15:0] POR_DIS_CODE	Power-on-reset Enable Control (Write Protect)
NMI_SEL	[8] NMI_EN	NMI Interrupt Enable Bit (Write Protect)
PWRCON	[8] PD_WAIT_CPU	this Bit Control the Power-down Entry Condition (Write Protect)
	[7] PWR_DOWN_EN	System Power-down Enable Bit (Write Protect)
	[5] PD_WU_INT_EN	Power-down Mode Wake-up Interrupt Enable Bit (Write Protect)
	[4] PD_WU_DLY	Enable the Wake-up Delay Counter (Write Protect)
	[3] OSC10K_EN	Internal 10 KHz Low Speed Oscillator Enable Bit (Write Protect)
	[2] OSC22M_EN	Internal 22.1184 MHz High Speed Oscillator Enable Bit (Write Protect)
	[1] XTL32K_EN	External 32.768 KHz Low Speed Crystal Enable Bit (Write Protect)
	[0] XTL12M_EN	External 4~24 MHz High Speed Crystal 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)
	[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)
	[5] LDUEN	LDROM Update Enable Bit (Write Protect)
	[4] CFGUEN	Enable Config-bits Update by ISP (Write Protect)
	[3] APUEN	APROM Update Enable Bit (Write Protect)
	[1] BS	Boot Select (Write Protect)
	[0] ISPEN	ISP Enable Bit (Write Protect)
ISPTRG	[0] ISPGO	ISP Start Trigger (Write Protect)
FATCON	[4] LFOM	Low Frequency Optimization Mode (Write Protect)
ISPSTA	[6] ISPFF	ISP Fail Flag (Write Protect)
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)
	[10:8] WTIS	Watchdog Timer Interval Selection (Write Protect)
	[7] WTE	Watchdog Timer Enable Bit (Write Protect)
	[6] WTIE	Watchdog Timer Interrupt Enable Bit (Write Protect)
	[4] WTWKE	Watchdog Timer Wake-up Function Enable Bit (Write Protect)
	[1] WTRE	Watchdog Timer Reset Enable Bit (Write Protect)
WTCRALT	[1:0] WTRDSEL	Watchdog Timer Reset Delay Select (Write Protect)

6.2.6 System Memory Map

The NuMicro® NUC100 series 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® NUC100 series 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
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
0x400D_0000 – 0x400D_3FFF	ACMP_BA	Analog Comparator Control Registers
0x400E_0000 – 0x400E_FFFF	ADC_BA	Analog-Digital-Converter (ADC) Control Registers
APB2 Controllers Space (0x4010_0000 ~ 0x401F_FFFF)		
0x4010_0000 – 0x4010_3FFF	PS2_BA	PS/2 Interface Control Registers
0x4011_0000 – 0x4011_3FFF	TMR23_BA	Timer2/Timer3 Control Registers
0x4012_0000 – 0x4012_3FFF	I2C1_BA	I^2C1 Interface Control Registers
0x4013_0000 – 0x4013_3FFF	SPI2_BA	SPI2 with master/slave function Control Registers
0x4013_4000 – 0x4013_7FFF	SPI3_BA	SPI3 with master/slave function Control Registers
0x4014_0000 – 0x4014_3FFF	PWMB_BA	PWM4/5/6/7 Control Registers
0x4015_0000 – 0x4015_3FFF	UART1_BA	UART1 Control Registers
0x4015_4000 – 0x4015_7FFF	UART2_BA	UART2 Control Registers
0x4019_0000 – 0x4019_3FFF	SC0_BA	SC0 Control Registers
0x4019_4000 – 0x4019_7FFF	SC1_BA	SC1 Control Registers
0x4019_8000 – 0x4019_BFFF	SC2_BA	SC2 Control Registers
0x401A_0000 – 0x401A_3FFF	I2S_BA	I^2S Interface 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 6-5 Address Space Assignments for On-Chip Controllers

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	IP Reset Control Register 1	0x0000_0000
IPRSTC2	GCR_BA+0x0C	R/W	IP Reset Control Register 2	0x0000_0000
IPRSTC3	GCR_BA+0x10	R/W	IP Reset Control Register 3	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_00XX
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
GPD_MFP	GCR_BA+0x3C	R/W	GPIOD 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_MFP1	GCR_BA+0x58	R/W	Alternative Multiple Function Pin Control Register 1	0x0000_0000
IRCTRIMCTL	GCR_BA+0x80	R/W	IRC Trim Control Register	0x0000_0000
IRCTRIMIEN	GCR_BA+0x84	R/W	IRC Trim Interrupt Enable Register	0x0000_0000
IRCTRIMINT	GCR_BA+0x88	R/W	IRC 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 cpu core and flash memory controller (fmc). 0 = No reset from CPU. 1 = Cortex®-M0 CPU core and FMC are reset by software setting CPU_RST to 1. **Note:** This bit can be cleared by writing '1' to it.
[6]	Reserved	Reserved.
[5]	RSTS_SYS	**SYS Reset Flag** The RSTS_SYS flag is set by the "reset signal" from the cortex®-m0 core to indicate the previous reset source. 0 = No reset from Cortex®-M0. 1 = The Cortex®-M0 had issued the reset signal to reset the system by writing 1 to bit SYSRESETREQ (AIRCR[2], Application Interrupt and Reset Control Register, address = 0xE000ED0C) in system control registers of Cortex®-M0 core. **Note:** This bit can be cleared by writing '1' to it.
[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:** This bit can be cleared by writing '1' to it.
[3]	RSTS_LVR	**Low Voltage Reset Flag** The RSTS_LVR flag is set by the "reset signal" from fhe low-voltage-reset controller fo indicate the previous reset source. 0 = No reset from LVR.

		1 = The LVR controller had issued the reset signal to reset the system. **Note:** This bit can be cleared by writing '1' to it.
[2]	**RSTS_WDT**	**Watchdog Timer Reset Flag** The RSTS_WDT flag is set by the "reset signal" from the watchdog timer to indicate the previous reset source. 0 = No reset from watchdog timer. 1 = The watchdog timer had issued the reset signal to reset the system. **Note:** This bit can be cleared by writing '1' to it.
[1]	**RSTS_RESET**	**Reset Pin Reset Flag** The RSTS_RESET flag is set by the "reset signal" from the /reset pin to indicate the previous reset source. 0 = No reset from /RESET pin. 1 = The Pin /RESET had issued the reset signal to reset the system. **Note:** This bit can be cleared by writing '1' to it.
[0]	**RSTS_POR**	**Power-on Reset Flag** The RSTS_POR flag is Set by the "Reset Signal" from the Power-on Reset (POR) controller or the bit CHIP_RST (IPRSTC1[0]) to indicate the previous reset source. 0 = No reset from POR or CHIP_RST. 1 = Power-on Reset (POR) or CHIP_RST had issued the reset signal to reset the system. **Note:** This bit can be cleared by writing '1' to it.

IP Reset Control Register 1 (IPRSTC1)

Register	Offset	R/W	Description	Reset Value
IPRSTC1	GCR_BA+0x08	R/W	IP 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 Protect)** 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 0 = EBI controller normal operation. 1 = EBI controller reset.
[2]	**PDMA_RST**	**PDMA Controller Reset (Write Protect)** Setting this bit to 1 will generate a reset signal to the PDMA. User need to set this bit to 0 to release from 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. 0 = PDMA controller normal operation. 1 = PDMA controller reset.
[1]	**CPU_RST**	**CPU Core One-shot Reset (Write Protect)** Setting this bit will only reset the CPU core and Flash Memory Controller(FMC), and this bit will automatically return 0 after the two clock cycles. 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 0 = CPU normal operation. 1 = CPU one-shot reset.
[0]	**CHIP_RST**	**CHIP One-shot Reset (Write Protect)** Setting this bit will reset the whole chip, including CPU core 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 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 0 = CHIP normal operation. 1 = CHIP one-shot reset.

IP Reset Control Register 2 (IPRSTC2)

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

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

31	30	29	28	27	26	25	24
Reserved		I2S_RST	ADC_RST	USBD_RST	Reserved		
23	22	21	20	19	18	17	16
PS2_RST	ACMP_RST	PWM47_RST	PWM03_RST	Reserved	UART2_RST	UART1_RST	UART0_RST
15	14	13	12	11	10	9	8
SPI3_RST	SPI2_RST	SPI1_RST	SPI0_RST	Reserved	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	
[31:30]	**Reserved**	Reserved.
[29]	**I2S_RST**	**I^2S Controller Reset** 0 = I^2S controller normal operation. 1 = I^2S controller reset.
[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.
[26:24]	**Reserved**	Reserved.
[23]	**PS2_RST**	**PS/2 Controller Reset** 0 = PS/2 controller normal operation. 1 = PS/2 controller reset.
[22]	**ACMP_RST**	**Analog Comparator Controller Reset** 0 = Analog Comparator controller normal operation. 1 = Analog Comparator controller reset.
[21]	**PWM47_RST**	**PWM47 Controller Reset** 0 = PWM47 controller normal operation. 1 = PWM47 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]	**SPI3_RST**	**SPI3 Controller Reset** 0 = SPI3 controller normal operation. 1 = SPI3 controller reset.
[14]	**SPI2_RST**	**SPI2 Controller Reset** 0 = SPI2 controller normal operation. 1 = SPI2 controller reset.
[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^2C1 Controller Reset** 0 = I^2C1 controller normal operation. 1 = I^2C1 controller reset.
[8]	**I2C0_RST**	**I^2C0 Controller Reset** 0 = I^2C0 controller normal operation. 1 = I^2C0 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.

IP Reset Control Register 3 (IPRSTC3)

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

Register	Offset	R/W	Description	Reset Value
IPRSTC3	GCR_BA+0x10	R/W	IP Reset Control Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					SC2_RST	SC1_RST	SC0_RST

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**SC2_RST**	**SC2 Controller Reset** 0 = SC2 controller normal operation. 1 = SC2 controller reset.
[1]	**SC1_RST**	**SC1 Controller Reset** 0 = SC1 controller normal operation. 1 = SC1 controller reset.
[0]	**SC0_RST**	**SC0 Controller Reset** 0 = SC0 controller normal operation. 1 = SC0 controller reset.

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). 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
[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. 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. This bit is the protected bit which 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.
[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.

<table>
<tr><td></td><td></td><td>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: This bit can be cleared by writing ‘1’ to it.</td></tr>
<tr><td>[3]</td><td>BOD_RSTEN</td><td>Brown-out Reset Enable Bit (Write Protect)
0 = Brown-out “INTERRUPT” function Enabled.
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).
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).
The default value is set by flash controller user configuration register config0 bit[20].
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.</td></tr>
<tr><td>[2:1]</td><td>BOD_VL</td><td>Brown-out Detector Threshold Voltage Selection (Write Protect)
The default value is set by flash controller user configuration register config0 bit[22:21]
This bit is the protected bit which means programming this needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Reference the register REGWRPROT at address GCR_BA+0x100.
<table>
<tr><th>BOD_VL[1]</th><th>BOD_VL[0]</th><th>Brown-out Voltage</th></tr>
<tr><td>1</td><td>1</td><td>4.4 V</td></tr>
<tr><td>1</td><td>0</td><td>3.7 V</td></tr>
<tr><td>0</td><td>1</td><td>2.7 V</td></tr>
<tr><td>0</td><td>0</td><td>2.2 V</td></tr>
</table></td></tr>
<tr><td>[0]</td><td>BOD_EN</td><td>Brown-out Detector Enable Bit (Write Protect)
The default value is set by flash controller user configuration register config0 bit[23]
0 = Brown-out Detector function Disabled.
1 = Brown-out Detector function Enabled.
This bit is the protected bit which 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.</td></tr>
</table>

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. 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_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**
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 Control (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: /RESET, Watchdog, LVR reset, BOD reset, ICE reset command and the software-chip reset function This bit is the protected bit which 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.

Multiple Function Pin GPIOA 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**	**Input Schmitt Trigger Enable Bit** 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** Bits PA15_SC2PWR (ALT_MFP1[12]), PA15_I2SMCLK (ALT_MFP[9]) and GPA_MFP[15] determine the PA.15 function. **PA15_SC2PWR / PA15_I2SMCLK / GPA_MFP[15] / PA.15 Function**: 0 / 0 / 0 / GPIO 0 / 0 / 1 / PWM3 (PWM) 0 / 1 / 1 / I2SMCLK 1 / 0 / 1 / SC2PWR
[14]	**GPA_MFP14**	**PA.14 Pin Function Selection** Bits EBI_HB_EN[7] (ALT_MFP[23]), EBI_EN (ALT_MFP[11]), PA14_SC2RST (ALT_MFP1[13]) and GPA_MFP[14] determine the PA.14 function. **EBI_HB_EN[7] / EBI_EN / PA14_SC2RST / GPA_MFP[14] / PA.14 Function**: 0 / 0 / 0 / 0 / GPIO 0 / 0 / 0 / 1 / PWM2 (PWM) 0 / 0 / 1 / 1 / SC2RST 1 / 1 / 0 / 1 / AD15

<table>
<tr><td>[13]</td><td>GPA_MFP13</td><td>PA.13 Pin Function Selection
Bits EBI_HB_EN[6] (ALT_MFP[22]), EBI_EN (ALT_MFP[11]), PA13_SC2CLK (ALT_MFP1[10]) and GPA_MFP[13] determine the PA.13 function.
<table>
<tr><th>EBI_HB_EN[6]</th><th>EBI_EN</th><th>PA13_SC2CLK</th><th>GPA_MFP[13]</th><th>PA.13 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>PWM1 (PWM)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC2CLK</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD14</td></tr>
</table></td></tr>
<tr><td>[12]</td><td>GPA_MFP12</td><td>PA.12 Pin Function Selection
Bits EBI_HB_EN[5] (ALT_MFP[21]), EBI_EN (ALT_MFP[11]), PA12_SC2DAT (ALT_MFP1[11]) and GPA_MFP[12] determine the PA.12 function.
<table>
<tr><th>EBI_HB_EN[5]</th><th>EBI_EN</th><th>PA12_SC2DAT</th><th>GPA_MFP[12]</th><th>PA.12 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>PWM0 (PWM)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC2DAT</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD13</td></tr>
</table></td></tr>
<tr><td>[11]</td><td>GPA_MFP11</td><td>PA.11 Pin Function Selection
Bits EBI_EN (ALT_MFP[11]) and GPA_MFP[11] determine the PA.11 function.
<table>
<tr><th>EBI_EN</th><th>GPA_MFP[10]</th><th>PA.10 Function</th></tr>
<tr><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>SCL1 (I^2C)</td></tr>
<tr><td>1</td><td>1</td><td>nRD</td></tr>
</table></td></tr>
<tr><td>[10]</td><td>GPA_MFP10</td><td>PA.10 Pin Function Selection
Bits EBI_EN (ALT_MFP[11]) and GPA_MFP[10] determine the PA.10 function.
<table>
<tr><th>EBI_EN</th><th>GPA_MFP[10]</th><th>PA.10 Function</th></tr>
<tr><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>SDA1 (I^2C1)</td></tr>
<tr><td>1</td><td>1</td><td>nWR</td></tr>
</table></td></tr>
<tr><td>[9]</td><td>GPA_MFP9</td><td>PA.9 Pin Function Selection
0 = GPIOA[9] selected to the pin PA.9.
1 = I^2C0 SCL function selected to the pin PA.9.</td></tr>
<tr><td>[8]</td><td>GPA_MFP8</td><td>PA.8 Pin Function Selection
0 = GPIOA[8] selected to the pin PA.8.
1 = I^2C0 SDA function selected to the pin PA.8.</td></tr>
</table>

<table>
<tr><td>[7]</td><td>GPA_MFP7</td><td>PA.7 Pin Function Selection
Bits EBI_EN (ALT_MFP[11]), PA7_SC1DAT (ALT_MFP1[6]), PA7_S21 (ALT_MFP[2]) and GPA_MFP[7] determine the PA.7 function
<table>
<tr><th>EBI_EN</th><th>PA7_SC1DAT</th><th>PA7_S21</th><th>GPA_MFP[7]</th><th>PA.7 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC7</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SPISS21 (SPI2)</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>SC1DAT</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>AD6</td></tr>
</table></td></tr>
<tr><td>[6]</td><td>GPA_MFP6</td><td>PA.6 Pin Function Selection
Bits EBI_EN (ALT_MFP[11]), PA6_SC1CLK (ALT_MFP1[5]) and GPA_MFP[6] determine the PA.6 function
<table>
<tr><th>EBI_EN</th><th>PA6_SC1CLK</th><th>GPA_MFP[6]</th><th>PA.6 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>ADC6</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>SC1CLK</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>AD7</td></tr>
</table></td></tr>
<tr><td>[5]</td><td>GPA_MFP5</td><td>PA.5 Pin Function Selection
Bits EBI_HB_EN[0] (ALT_MFP[16]), EBI_EN (ALT_MFP[11]), PA5_SC1RST (ALT_MFP1[8]) and GPA_MFP[5] determine the PA.5 function
<table>
<tr><th>EBI_HB_EN[0]</th><th>EBI_EN</th><th>PA5_SC1RST</th><th>GPA_MFP[5]</th><th>PA.5 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC5</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC1RST</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD8</td></tr>
</table></td></tr>
<tr><td>[4]</td><td>GPA_MFP4</td><td>PA.4 Pin Function Selection
Bits EBI_HB_EN[1] (ALT_MFP[17]), EBI_EN (ALT_MFP[11]), PA4_SC1PWR (ALT_MFP1[7]) and GPA_MFP[4] determine the PA.4 function
<table>
<tr><th>EBI_HB_EN[1]</th><th>EBI_EN</th><th>PA4_SC1PWR</th><th>GPA_MFP[4]</th><th>PA.4 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC4</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC1PWR</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD9</td></tr>
</table></td></tr>
</table>

[3]	**GPA_MFP3**	**PA.3 Pin Function Selection** Bits EBI_HB_EN[2] (ALT_MFP[18]), EBI_EN (ALT_MFP[11]), PA3_SC0DAT (ALT_MFP1[1]) and GPA_MFP[3] determine the PA.3 function

EBI_HB_EN[2]	EBI_EN	PA3_SC0DAT	GPA_MFP[3]	PA.3 Function
0	0	0	0	GPIO
0	0	0	1	ADC3
0	0	1	1	SC0DAT
1	1	0	1	AD10

[2]	**GPA_MFP2**	**PA.2 Pin Function Selection** Bits EBI_HB_EN[3] (ALT_MFP[19]), EBI_EN (ALT_MFP[11]), PA2_SC0CLK (ALT_MFP1[0]) and GPA_MFP[2] determine the PA.2 function

EBI_HB_EN[3]	EBI_EN	PA2_SC0CLK	GPA_MFP[2]	PA.2 Function
0	0	0	0	GPIO
0	0	0	1	ADC2
0	0	1	1	SC0CLK
1	1	0	1	AD11

[1]	**GPA_MFP1**	**PA.1 Pin Function Selection** Bit EBI_HB_EN[4] (ALT_MFP[20]), EBI_EN (ALT_MFP[11]), PA1_SC0RST (ALT_MFP1[3]) and GPA_MFP[1] determine the PA.1 function

EBI_HB_EN[4]	EBI_EN	PA1_SC0RST	GPA_MFP[1]	PA.1 Function
0	0	0	0	GPIO
0	0	0	1	ADC1
0	0	1	1	SC0RST
1	1	0	1	AD12

[0]	**GPA_MFP0**	**PA.0 Pin Function Selection** Bit PA0_SC0PWR (ALT_MFP1[2]) and GPA_MFP[0] determine the PA.0 function

PA0_SC0PWR	GPA_MFP[0]	PA.0 Function
0	0	GPIO
0	1	ADC0
1	1	SC0PWR

Multiple Function Pin GPIOB 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	**Input Schmitt Trigger Enable Bit** 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 PB15_T0EX (ALT_MFP[24]) and GPB_MFP[15] determine the PB.15 function. **PB15_T0EX** / **GPB_MFP[15]** / **PB.15 function**: 0 / 0 / GPIO 0 / 1 / /INT1 1 / 1 / T0EX (TMR0)
[14]	GPB_MFP14	**PB.14 Pin Function Selection** Bits PB14_S31 (ALT_MFP[3]) and GPB_MFP[14] determine the PB.14 function. **PB14_S31** / **GPB_MFP[14]** / **PB.14 function**: 0 / 0 / GPIO 0 / 1 / /INT0 1 / 1 / SPISS31 (SPI3)
[13]	GPB_MFP13	**PB.13 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]) and GPB_MFP[13] determine the PB.13 function. **EBI_EN** / **GPB_MFP[13]** / **PB.13 function**: 0 / 0 / GPIO 0 / 1 / CPO1 (CMP) 1 / 1 / AD1
[12]	GPB_MFP12	**PB.12 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]), PB12_CLKO (ALT_MFP[10]) and GPB_MFP[12] determine

		the PB.12 function. EBI_EN / PB12_CLKO / GPB_MFP[12] / PB.12 function: 0 / 0 / 0 / GPIO 0 / 0 / 1 / CPO0 (CMP) 0 / 1 / 1 / CLKO (Clock Driver output) 1 / 0 / 1 / AD0
[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_MFP[11] / PB.11 function: 0 / 0 / GPIO 0 / 1 / TM3 1 / 1 / PWM4 (PWM)
[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_MFP[10] / PB.10 function: 0 / 0 / GPIO 0 / 1 / TM2 1 / 1 / SPISS01 (SPI0)
[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_MFP[9] / PB.9 function: 0 / 0 / GPIO 0 / 1 / TM1 1 / 1 / SPISS11 (SPI1)
[8]	**GPB_MFP8**	**PB.8 Pin Function Selection** 0 = GPIOB[8] selected to the pin PB.8. 1 = TM0 (Timer/Counter external trigger clock input) function selected to the pin PB.8.
[7]	**GPB_MFP7**	**PB.7 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]) and GPB_MFP[7] determine the PB.7 function. EBI_EN / GPB_MFP[7] / PB.7 function: 0 / 0 / GPIO 0 / 1 / CTS1 (UART1) 1 / 1 / nCS
[6]	**GPB_MFP6**	**PB.6 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]) and GPB_MFP[6] determine the PB.6 function. EBI_EN / GPB_MFP[6] / PB.6 function: 0 / 0 / GPIO 0 / 1 / RTS1 (UART1)

		1 | 1 | ALE (EBI)
[5]	**GPB_MFP5**	**PB 5 Pin Function Selection** 0 = GPIOB[5] 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** 0 = GPIOB[4] 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_SC2CD (ALT_MFP1[14]), PB3_T3EX (ALT_MFP[27]) and GPB_MFP[3] determine the PB.3 function

EBI_nWRH_EN	EBI_EN	PB3_SC2CD	PB3_T3EX	GPB_MFP[3]	PB.3 Function
0	0	0	0	0	GPIO
0	0	0	0	1	CTS0
0	0	0	1	1	T3EX
0	0	1	0	1	SC2CD
1	1	0	0	1	nWRH

[2]	**GPB_MFP2**	**PB.2 Pin Function Selection** Bits EBI_nWRH_EN (ALT_MFP[13]), EBI_EN (ALT_MFP[11]), PB2_T2EX (ALT_MFP[26]) and GPB_MFP[2] determine the PB.2 function.

EBI_nWRL_EN	EBI_EN	GPB_MFP[2]	PB2_T2EX	PB.2 function
0	0	0	0	GPIO
0	0	1	0	RTS0 (UART0)
1	1	1	0	nWRL
0	0	1	1	T2EX (TMR2)

[1]	**GPB_MFP1**	**PB.1 Pin Function Selection** 0 = GPIOB[1] 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** 0 = GPIOB[0] is selected to the pin PB.0. 1 = UART0 RXD function is selected to the pin PB.0.

Multiple Function Pin GPIOC 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	**Input Schmitt Trigger Enable Bit** 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_MFP[15] / PC.15 function: 0 / 0 / GPIO 0 / 1 / CPN1 (CMP) 1 / 1 / AD3
[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_MFP[14]** / **PC.14 function**: 0 / 0 / GPIO 0 / 1 / CPP1 (CMP) 1 / 1 / AD2
[13]	GPC_MFP13	**PC.13 Pin Function Selection** 0 = GPIOC[13] is selected to the pin PC.13. 1 = SPI1 MOSI1 (master output, slave input pin-1) function is selected to the pin PC.13.
[12]	GPC_MFP12	**PC.12 Pin Function Selection** 0 = GPIOC[12] is selected to the pin PC.12. 1 = SPI1 MISO1 (master input, slave output pin-1) function is selected to the pin PC.12.
[11]	GPC_MFP11	**PC.11 Pin Function Selection** 0 = GPIOC[11] selected to the pin PC.11. 1 = SPI1 MOSI0 (master output, slave input pin-0) function selected to the pin PC.11.

[10]	**GPC_MFP10**	**PC.10 Pin Function Selection** 0 = GPIOC[10] is selected to the pin PC.10. 1 = SPI1 MISO0 (master input, slave output pin-0) function selected to the pin PC.10.
[9]	**GPC_MFP9**	**PC.9 Pin Function Selection** 0 = GPIOC[9] selected to the pin PC.9. 1 = SPI1 SPICLK function 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]) and GPC_MFP[8] determine the PC.8 function. **EBI_MCLK_EN / EBI_EN / GPC_MFP[8] / PC.8 function** 0 / 0 / 0 / GPIO 0 / 0 / 1 / SPISS10 (SPI1) 1 / 1 / 1 / MCLK
[7]	**GPC_MFP7**	**PC.7 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]), PC7_SC1CD (ALT_MFP1[9]) and GPC_MFP[7] determine the PC.7 function **EBI_EN / PC7_SC1CD / GPC_MFP[7] / PC.7 Function** 0 / 0 / 0 / GPIO 0 / 0 / 1 / CPN0 0 / 1 / 1 / SC1CD 1 / 0 / 1 / AD5
[6]	**GPC_MFP6**	**PC.6 Pin Function Selection** Bits EBI_EN (ALT_MFP[11]), PC6_SC0CD (ALT_MFP1[4]) and GPC_MFP[6] determine the PC.6 function **EBI_EN / PC6_SC0CD / GPC_MFP[6] / PC.6 Function** 0 / 0 / 0 / GPIO 0 / 0 / 1 / CPP0 0 / 1 / 1 / SC0CD 1 / 0 / 1 / AD4
[5]	**GPC_MFP5**	**PC.5 Pin Function Selection** 0 = GPIOC[5] is selected to the pin PC.5. 1 = SPI0 MOSI1 (master output, slave input pin-1) function is selected to the pin PC.5.
[4]	**GPC_MFP4**	**PC.4 Pin Function Selection** 0 = GPIOC[4] is selected to the pin PC.4. 1 = SPI0 MISO1 (master input, slave output pin-1) function is selected to the pin PC.4.

[3]	**GPC_MFP3**	**PC.3 Pin Function Selection** Bits PC3_I2SDO (ALT_MFP[8]) and GPC_MFP[3] determine the PC.3 function. **PC3_I2SDO** / **GPC_MFP[3]** / **PC.3 function** 0 / 0 / GPIO 0 / 1 / MOSI00 (SPI0) 1 / 1 / I2SDO (I^2S)
[2]	**GPC_MFP2**	**PC.2 Pin Function Selection** Bits PC2_I2SDI (ALT_MFP[7]) and GPC_MFP[2] determine the PC.2 function. **PC2_I2SDI** / **GPC_MFP[2]** / **PC.2 function** 0 / 0 / GPIO 0 / 1 / MISO00 (SPI0) 1 / 1 / I2SDI (I^2S)
[1]	**GPC_MFP1**	**PC.1 Pin Function Selection** Bits PC1_I2SBCLK (ALT_MFP[6]) and GPC_MFP[1] determine the PC.1 function. **PC1_I2SBCLK** / **GPC_MFP[1]** / **PC.1 function** 0 / 0 / GPIO 0 / 1 / SPICLK0 (SPI0) 1 / 1 / I2SBCLK (I^2S)
[0]	**GPC_MFP0**	**PC.0 Pin Function Selection** Bits PC0_I2SLRCLK (ALT_MFP[5]) and GPC_MFP[0] determine the PC.0 function. **PC0_I2SLRCLK** / **GPC_MFP[0]** / **PC.0 function** 0 / 0 / GPIO 0 / 1 / SPISS00 (SPI0) 1 / 1 / I2SLRCLK (I^2S)

Multiple Function Pin GPIOD Control Register (GPD_MFP)

Register	Offset	R/W	Description	Reset Value
GPD_MFP	GCR_BA+0x3C	R/W	GPIOD Multiple Function and Input Type Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**GPD_TYPEn**	**Input Schmitt Trigger Enable Bit** 0 = GPIOD[15:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOD[15:0] I/O input Schmitt Trigger function Enabled.
[15]	**GPD_MFP15**	**PD.15 Pin Function Selection** 0 = GPIOD[15] selected to the pin PD.15. 1 = UART2 TXD function is selected to the pin PD.15.
[14]	**GPD_MFP14**	**PD.14 Pin Function Selection** 0 = GPIOD[14] selected to the pin PD.14. 1 = UART2 RXD function is selected to the pin PD.14.
[13]	**GPD_MFP13**	**PD.13 Pin Function Selection** 0 = GPIOD[13] is selected to the pin PD.13. 1 = SPI3 MOSI1 (master output, slave input pin-1) function is selected to the pin PD.13.
[12]	**GPD_MFP12**	**PD.12 Pin Function Selection** 0 = GPIOD[12] is selected to the pin PD.12. 1 = SPI3 MISO1 (master input, slave output pin-1) function is selected to the pin PD.12.
[11]	**GPD_MFP11**	**PD.11 Pin Function Selection** 0 = GPIOD[11] is selected to the pin PD.11. 1 = SPI3 MOSI0 (master output, slave input pin-0) function is selected to the pin PD.11.
[10]	**GPD_MFP10**	**PD.10 Pin Function Selection** 0 = GPIOD[10] is selected to the pin PD.10. 1 = SPI3 MISO0 (master input, slave output pin-0) function is selected to the pin PD.10.
[9]	**GPD_MFP9**	**PD.9 Pin Function Selection** 0 = GPIOD[9] is selected to the pin PD.9. 1 = SPI3 SPICLK function is selected to the pin PD.9.
[8]	**GPD_MFP8**	**PD.8 Pin Function Selection** 0 = GPIOD[8] is selected to the pin PD8.

		1 = SPI3 SS30 function is selected to the pin PD8.
[7]	**GPD_MFP7**	Reserved.
[6]	**GPD_MFP6**	Reserved.
[5]	**GPD_MFP5**	**PD.5 Pin Function Selection** 0 = GPIOD[5] is selected to the pin PD.5. 1 = SPI2 MOSI1 (master output, slave input pin-1) function is selected to the pin PD.5.
[4]	**GPD_MFP4**	**PD.4 Pin Function Selection** 0 = GPIOD[4]is selected to the pin PD.4. 1 = SPI2 MISO1 (master input, slave output pin-1) function is selected to the pin PD.4.
[3]	**GPD_MFP3**	**PD.3 Pin Function Selection** 0 = GPIOD[3] selected to the pin PD.3. 1 = SPI2 MOSI0 (master output, slave input pin-0) function selected to the pin PD.3.
[2]	**GPD_MFP2**	**PD.2 Pin Function Selection** 0 = GPIOD[2] selected to the pin PD.2. 1 = SPI2 MISO0 (master input, slave output pin-0) function selected to the pin PD.2.
[1]	**GPD_MFP1**	**PD.1 Pin Function Selection** 0 = GPIOD[1] selected to the pin PD.1. 1 = SPI2 SPICLK function selected to the pin PD.1.
[0]	**GPD_MFP0**	**PD.0 Pin Function Selection** 0 = GPIOD[0] selected to the pin PD.0. 1 = SPI2 SS20 function selected to the pin PD.0.

Multiple Function Pin GPIOE 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			GPE_MFP1	GPE_MFP0

Bits	Description	
[31:16]	GPE_TYPEn	**Input Schmitt Trigger Enable Bit** 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]) and GPE_MFP5 determine the PE.5 function **PE5_T1EX / GPE_MFP[5] / PE.5 Function** 0 / 0 / GPIO 0 / 1 / PWM5 1 / 1 / TM1_EXT
[4:2]	Reserved	Reserved.
[1]	GPE_MFP1	**PE.1 Pin Function Selection** 0 = GPIOE[1] is selected to the pin PE.1. 1 = PWM7 function is selected to the pin PE.1.
[0]	GPE_MFP0	**PE.0 Pin Function Selection** 0 = GPIOE[0] is selected to the pin PE.0. 1 = PWM6 function is selected to the pin PE.0.

Multiple Function Pin GPIOF 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.

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_MFP3	GPF_MFP2	GPF_MFP1	GPF_MFP0

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**GPF_TYPEn**	**Input Schmitt Trigger Enable Bit** 0 = GPIOF[3:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOF[3:0] I/O input Schmitt Trigger function Enabled.
[15:4]	**Reserved**	Reserved.
[3]	**GPF_MFP3**	**PF.3 Pin Function Selection** 0 = GPIOF[3] is selected to the pin PF.3. 1 = PS2CLK function is selected to the pin PF.3.
[2]	**GPF_MFP2**	**PF.2 Pin Function Selection** 0 = GPIOF[2] is selected to the pin PF.2. 1 = PS2DAT function is selected to the pin PF.2.
[1]	**GPF_MFP1**	**PF.1 Pin Function Selection** 0 = GPIOF[1] 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** 0 = GPIOF[0] 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				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	PB12_CLKO	PA15_I2SMCLK	PC3_I2SDO
7	6	5	4	3	2	1	0
PC2_I2SDI	PC1_I2SBCLK	PC0_I2SLRCLK	PB11_PWM4	PB14_S31	PA7_S21	PB9_S11	PB10_S01

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:28]</td><td>Reserved</td><td>Reserved.</td></tr>
<tr><td>[27]</td><td>PB3_T3EX</td><td>PB.3 Pin Function Selection
Bits EBI_NWRH_EN (ALT_MFP[14]), EBI_EN (ALT_MFP[11]), PB3_SC2CD (ALT_MFP1[14]), PB3_T3EX (ALT_MFP[27]) and GPB_MFP[3] determine the PB.3 function
<table>
<tr><th>EBI_nWRH_EN</th><th>EBI_EN</th><th>PB3_SC2CD</th><th>PB3_T3EX</th><th>GPB_MFP[3]</th><th>PB.3 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>CTS0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>T3EX</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>SC2CD</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>nWRH</td></tr>
</table></td></tr>
<tr><td>[26]</td><td>PB2_T2EX</td><td>PB.2 Pin Function Selection
Bits GPB_MFP2 and EBI_NWRL_EN (ALT_MFP[13]) and EBI_EN (ALT_MFP[11]) and PB2_T2EX (ALT_MFP[26]) determine the PB.2 function
<table>
<tr><th>EBI_nWRL_EN</th><th>EBI_EN</th><th>GPB_MFP[2]</th><th>PB2_T2EX</th><th>PB.2 function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>RTS0 (UART0)</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>nWRL (EBI write low byte enable)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>T2EX (TMR2)</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>[25]</td><td>PE5_T1EX</td><td>PE.5 Pin Function Selection
Bits GPE_MFP5 and PE5_T1EX (ALT_MFP[25]) determine the PE.5 function
<table>
<tr><th>PE5_T1EX</th><th>GPE_MFP[5]</th><th>PE.5 function</th></tr>
<tr><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>PWM5 (PWM)</td></tr>
<tr><td>1</td><td>1</td><td>T1EX (TMR1)</td></tr>
</table></td></tr>
<tr><td>[24]</td><td>PB15_T0EX</td><td>PB.15 Pin Function Selection
Bits PB15_T0EX (ALT_MFP[24]) and GPB_MFP[15] Determine the PB.15 Function
<table>
<tr><th>PB15_T0EX</th><th>GPB_MFP[15]</th><th>PB.15 function</th></tr>
<tr><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>/INT1</td></tr>
<tr><td>1</td><td>1</td><td>T0EX (TMR0)</td></tr>
</table></td></tr>
<tr><td>[23]</td><td>EBI_HB_EN[7]</td><td>PA.14 Pin Function Selection
Bits EBI_HB_EN[7] (ALT_MFP[23]), EBI_EN (ALT_MFP[11]), PA14_SC2RST (ALT_MFP1[13]) and GPA_MFP[14] determine the PA.14 function
<table>
<tr><th>EBI_HB_EN[7]</th><th>EBI_EN</th><th>PA14_SC2RST</th><th>GPA_MFP[14]</th><th>PA.14 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>PWM2 (PWM)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC2RST</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD15</td></tr>
</table></td></tr>
<tr><td>[22]</td><td>EBI_HB_EN[6]</td><td>PA.13 Pin Function Selection
Bits EBI_HB_EN[6] (ALT_MFP[22]), EBI_EN (ALT_MFP[11]), PA13_SC2CLK (ALT_MFP1[10]) and GPA_MFP[13] determine the PA.13 function
<table>
<tr><th>EBI_HB_EN[6]</th><th>EBI_EN</th><th>PA13_SC2CLK</th><th>GPA_MFP[13]</th><th>PA.13 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>PWM1 (PWM)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC2CLK</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD14</td></tr>
</table></td></tr>
<tr><td>[21]</td><td>EBI_HB_EN[5]</td><td>PA.12 Pin Function Selection
Bits EBI_HB_EN[5] (ALT_MFP[21]), EBI_EN (ALT_MFP[11]), PA12_SC2DAT (ALT_MFP1[11]) and GPA_MFP[12] determine the PA.12 function
<table>
<tr><th>EBI_HB_EN[5]</th><th>EBI_EN</th><th>PA12_SC2DAT</th><th>GPA_MFP[12]</th><th>PA.12 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>PWM0 (PWM)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC2DAT</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD13</td></tr>
</table></td></tr>
</table>

[20]	**EBI_HB_EN[4]**	**PA.1 Pin Function Selection** Bit EBI_HB_EN[4] (ALT_MFP[20]), EBI_EN (ALT_MFP[11]), PA1_SC0RST (ALT_MFP1[3]) and GPA_MFP[1] determine the PA.1 function **EBI_HB_EN[4]** | **EBI_EN** | **PA1_SC0RST** | **GPA_MFP[1]** | **PA.1 Function** 0 | 0 | 0 | 0 | GPIO 0 | 0 | 0 | 1 | ADC1 0 | 0 | 1 | 1 | SC0RST 1 | 1 | 0 | 1 | AD12
[19]	**EBI_HB_EN[3]**	**PA.2 Pin Function Selection** Bits EBI_HB_EN[3] (ALT_MFP[19]), EBI_EN (ALT_MFP[11]), PA2_SC0CLK (ALT_MFP1[0]) and GPA_MFP[2] determine the PA.2 function **EBI_HB_EN[3]** | **EBI_EN** | **PA2_SC0CLK** | **GPA_MFP[2]** | **PA.2 Function** 0 | 0 | 0 | 0 | GPIO 0 | 0 | 0 | 1 | ADC2 0 | 0 | 1 | 1 | SC0CLK 1 | 1 | 0 | 1 | AD11
[18]	**EBI_HB_EN[2]**	**PA.3 Pin Function Selection** Bits EBI_HB_EN[2] (ALT_MFP[18]), EBI_EN (ALT_MFP[11]), PA3_SC0DAT (ALT_MFP1[1]) and GPA_MFP[3] determine the PA.3 function **EBI_HB_EN[2]** | **EBI_EN** | **PA3_SC0DAT** | **GPA_MFP[3]** | **PA.3 Function** 0 | 0 | 0 | 0 | GPIO 0 | 0 | 0 | 1 | ADC3 0 | 0 | 1 | 1 | SC0DAT 1 | 1 | 0 | 1 | AD10
[17]	**EBI_HB_EN[1]**	**PA.4 Pin Function Selection** Bits EBI_HB_EN[1] (ALT_MFP[17]), EBI_EN (ALT_MFP[11]), PA4_SC1PWR (ALT_MFP1[7]) and GPA_MFP[4] determine the PA.4 function **EBI_HB_EN[1]** | **EBI_EN** | **PA4_SC1PWR** | **GPA_MFP[4]** | **PA.4 Function** 0 | 0 | 0 | 0 | GPIO 0 | 0 | 0 | 1 | ADC4 0 | 0 | 1 | 1 | SC1PWR 1 | 1 | 0 | 1 | AD9

<table>
<tr><td>[16]</td><td>EBI_HB_EN[0]</td><td>PA.5 Pin Function Selection
Bits EBI_HB_EN[0] (ALT_MFP[16]), EBI_EN (ALT_MFP[11]), PA5_SC1RST (ALT_MFP1[8]) and GPA_MFP[5] determine the PA.5 function
<table>
<tr><th>EBI_HB_EN[0]</th><th>EBI_EN</th><th>PA5_SC1RST</th><th>GPA_MFP[5]</th><th>PA.5 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC5</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC1RST</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD8</td></tr>
</table></td></tr>
<tr><td>[15]</td><td>Reserved</td><td>Reserved.</td></tr>
<tr><td>[14]</td><td>EBI_nWRH_EN</td><td>PB.3 Pin Function Selection
Bits EBI_NWRH_EN (ALT_MFP[14]), EBI_EN (ALT_MFP[11]), PB3_SC2CD (ALT_MFP1[14]), PB3_T3EX (ALT_MFP[27]) and GPB_MFP[3] determine the PB.3 function
<table>
<tr><th>EBI_nWRH_EN</th><th>EBI_EN</th><th>PB3_SC2CD</th><th>PB3_T3EX</th><th>GPB_MFP[3]</th><th>PB.3 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>CTS0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>T3EX</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>SC2CD</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>nWRH</td></tr>
</table></td></tr>
<tr><td>[13]</td><td>EBI_nWRL_EN</td><td>PB.2 Pin Function Selection
Bits GPB_MFP2 and EBI_NWRL_EN (ALT_MFP[13]) and EBI_EN (ALT_MFP[11]) and PB2_T2EX (ALT_MFP[26]) determine the PB.2 function
<table>
<tr><th>EBI_nWRL_EN</th><th>EBI_EN</th><th>GPB_MFP[2]</th><th>PB2_T2EX</th><th>PB.2 function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>RTS0 (UART0)</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>nWRL (EBI write low byte enable)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>T2EX (TMR2)</td></tr>
</table></td></tr>
<tr><td>[12]</td><td>EBI_MCLK_EN</td><td>PC.8 Pin Function Selection
Bits EBI_MCLK_EN, EBI_EN and GPC_MFP[8] determine the PC.8 function
<table>
<tr><th>EBI_MCLK_EN</th><th>EBI_EN</th><th>GPC_MFP[8]</th><th>PC.8 function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>SPISS10 (SPI1)</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>MCLK (EBI Clock output)</td></tr>
</table></td></tr>
<tr><td>[11]</td><td>EBI_EN</td><td>EBI Function Selection
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)
<table>
<tr><th>EBI_EN</th><th>PA6_SC1CLK</th><th>GPA_MFP[6]</th><th>PA.6 Function</th></tr>
</table></td></tr>
</table>

0	0	0	GPIO
0	0	1	ADC6
0	1	1	SC1CLK
1	0	1	AD7

EBI_EN	PA7_SC1DAT	PA7_S21	GPA_MFP[7]	PA.7 Function
0	0	0	0	GPIO
0	0	0	1	ADC7
0	0	1	1	SPISS21 (SPI2)
0	1	0	1	SC1DAT
1	0	0	1	AD6

EBI_EN	PC7_SC1CD	GPC_MFP[7]	PC.7 Function
0	0	0	GPIO
0	0	1	CPN0
0	1	1	SC1CD
1	0	1	AD5

EBI_EN	PC6_SC0CD	GPC_MFP[6]	PC.6 Function
0	0	0	GPIO
0	0	1	CPP0
0	1	1	SC0CD
1	0	1	AD4

EBI_EN	GPC_MFP[15]	PC.15 function
0	0	GPIO
0	1	CPN1 (CMP)
1	1	AD3

EBI_EN	GPC_MFP[14]	PC.14 function
0	0	GPIO
0	1	CPP1 (CMP)
1	1	AD2

EBI_EN	GPB_MFP[13]	PB.13 function
0	0	GPIO

<table>
<tr><td></td><td></td><td>
<table>
<tr><td>0</td><td>1</td><td>CPO1 (CMP)</td></tr>
<tr><td>1</td><td>1</td><td>AD1</td></tr>
</table>
<table>
<tr><th>EBI_EN</th><th>PB12_CLKO</th><th>GPB_MFP[12]</th><th>PB.12 function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>CPO0 (CMP)</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>CLKO (Clock Driver output)</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>AD0</td></tr>
</table>
<table>
<tr><th>EBI_EN</th><th>GPA_MFP[11]</th><th>PA.11 function</th></tr>
<tr><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>SCL1 (I^2C)</td></tr>
<tr><td>1</td><td>1</td><td>nRD</td></tr>
</table>
<table>
<tr><th>EBI_EN</th><th>GPA_MFP[10]</th><th>PA.10 function</th></tr>
<tr><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>SDA1 (I^2C1)</td></tr>
<tr><td>1</td><td>1</td><td>nWR</td></tr>
</table>
<table>
<tr><th>EBI_EN</th><th>GPB_MFP[6]</th><th>PB.6 function</th></tr>
<tr><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>RTS1 (UART1)</td></tr>
<tr><td>1</td><td>1</td><td>ALE</td></tr>
</table>
<table>
<tr><th>EBI_EN</th><th>GPB_MFP[7]</th><th>PB.7 function</th></tr>
<tr><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>CTS1 (UART1)</td></tr>
<tr><td>1</td><td>1</td><td>nCS</td></tr>
</table>
</td></tr>
<tr><td>[10]</td><td>PB12_CLKO</td><td>
PB.12 Pin Function Selection
Bits PB12_CLKO, GPB_MFP[12] and EBI_EN (ALT_MFP[11]) determine the PB.12 function
<table>
<tr><th>EBI_EN</th><th>PB12_CLKO</th><th>GPB_MFP[12]</th><th>PB.12 function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>CPO0 (CMP)</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>CLKO (Clock Driver output)</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>AD0 (EBI AD bus bit 0)</td></tr>
</table>
</td></tr>
</table>

[9]	**PA15_I2SMCLK**	**PA.15 Pin Function Selection** Bits PA15_SC2PWR (ALT_MFP1[12]), PA15_I2SMCLK (ALT_MFP[9]) and GPA_MFP[15] determine the PA.15 function **PA15_SC2PWR / PA15_I2SMCLK / GPA_MFP[15] / PA.15 Function** 0 / 0 / 0 / GPIO 0 / 0 / 1 / PWM3 (PWM) 0 / 1 / 1 / I2SMCLK 1 / 0 / 1 / SC2PWR
[8]	**PC3_I2SDO**	**PC.3 Pin Function Selection** Bits PC3_I2SDO and GPC_MFP[3] determine the PC.3 function **PC3_I2SDO / GPC_MFP[3] / PC.3 function** 0 / 0 / GPIO 0 / 1 / MOSI00 (SPI0) 1 / 1 / I2SDO (I^2S)
[7]	**PC2_I2SDI**	**PC.2 Pin Function Selection** Bits PC2_I2SDI and GPC_MFP[2] determine the PC.2 function **PC2_I2SDI / GPC_MFP[2] / PC.2 function** 0 / 0 / GPIO 0 / 1 / MISO00 (SPI0) 1 / 1 / I2SDI (I^2S)
[6]	**PC1_I2SBCLK**	**PC.1 Pin Function Selection** Bits PC1_I2SBCLK and GPC_MFP[1] determine the PC.1 function **PC1_I2SBCLK / GPC_MFP[1] / PC.1 function** 0 / 0 / GPIO 0 / 1 / SPICLK0 (SPI0) 1 / 1 / I2SBCLK (I^2S)
[5]	**PC0_I2SLRCLK**	**PC.0 Pin Function Selection** Bits PC0_I2SLRCLK and GPC_MFP[0] determine the PC.0 function **PC0_I2SLRCLK / GPC_MFP[0] / PC.0 function** 0 / 0 / GPIO 0 / 1 / SPISS00 (SPI0) 1 / 1 / I2SLRCLK (I^2S)
[4]	**PB11_PWM4**	**PB.11 Pin Function Selection** Bits PB11_PWM4 and GPB_MFP[11] determine the PB.11 function **PB11_PWM4 / GPB_MFP[11] / PB.11 function** 0 / 0 / GPIO 0 / 1 / TM3 1 / 1 / PWM4 (PWM)

<table>
<tr><td>[3]</td><td>PB14_S31</td><td>PB.14 Pin Function Selection
Bits PB14_S31 and GPB_MFP[14] determine the PB.14 function<table><tr><th>PB14_S31</th><th>GPB_MFP[14]</th><th>PB.14 function</th></tr><tr><td>0</td><td>0</td><td>GPIO</td></tr><tr><td>0</td><td>1</td><td>/INT0</td></tr><tr><td>1</td><td>1</td><td>SPISS31 (SPI3)</td></tr></table></td></tr>
<tr><td>[2]</td><td>PA7_S21</td><td>PA.7 Pin Function Selection
Bits EBI_EN (ALT_MFP[11]), PA7_SC1DAT (ALT_MFP1[6]), PA7_S21 (ALT_MFP[2]) and GPA_MFP[7] determine the PA.7 function<table><tr><th>EBI_EN</th><th>PA7_SC1DAT</th><th>PA7_S21</th><th>GPA_MFP[7]</th><th>PA.7 Function</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC7</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SPISS21 (SPI2)</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td><td>SC1DAT</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td><td>AD6</td></tr></table></td></tr>
<tr><td>[1]</td><td>PB9_S11</td><td>PB.9 Pin Function Selection
Bits PB9_S11 and GPB_MFP[9] determine the PB.9 function<table><tr><th>PB9_S11</th><th>GPB_MFP[9]</th><th>PB.9 function</th></tr><tr><td>0</td><td>0</td><td>GPIO</td></tr><tr><td>0</td><td>1</td><td>TM1</td></tr><tr><td>1</td><td>1</td><td>SPISS11 (SPI1)</td></tr></table></td></tr>
<tr><td>[0]</td><td>PB10_S01</td><td>PB.10 Pin Function Selection
Bits PB10_S01 and GPB_MFP[10] determine the PB.10 function<table><tr><th>PB10_S01</th><th>GPB_MFP[10]</th><th>PB.10 function</th></tr><tr><td>0</td><td>0</td><td>GPIO</td></tr><tr><td>0</td><td>1</td><td>TM2</td></tr><tr><td>1</td><td>1</td><td>SPISS01 (SPI0)</td></tr></table></td></tr>
</table>

Alternative Multiple Function Pin Control Register1 (ALT_MFP1)

Register	Offset	R/W	Description	Reset Value
ALT_MFP1	GCR_BA+0x58	R/W	Alternative Multiple Function Pin 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	PB3_ SC2CD	PA14_ SC2RST	PA15_ SC2PWR	PA12_ SC2DAT	PA13_ SC2CLK	PC7_ SC1CD	PA5_ SC1RST
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PA4_ SC1PWR	PA7_ SC1DAT	PA6_ SC1CLK	PC6_ SC0CD	PA1_ SC0RST	PA0_ SC0PWR	PA3_ SC0DAT	PA2_ SC0CLK

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14]	**PB3_SC2CD**	**PB.3 Pin Function Selection** Bits EBI_NWRH_EN (ALT_MFP[14]), EBI_EN (ALT_MFP[11]), PB3_SC2CD (ALT_MFP1[14]), PB3_T3EX (ALT_MFP[27]) and GPB_MFP[3] determine the PB.3 function <table><tr><th>EBI_nWRH_EN</th><th>EBI_EN</th><th>PB3_SC2CD</th><th>PB3_T3EX</th><th>GPB_MFP[3]</th><th>PB.3 Function</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>CTS0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>T3EX</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>SC2CD</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>nWRH</td></tr></table>
[13]	**PA14_SC2RST**	**PA.14 Pin Function Selection** Bits EBI_HB_EN[7] (ALT_MFP[23]), EBI_EN (ALT_MFP[11]), PA14_SC2RST (ALT_MFP1[13]) and GPA_MFP[14] determine the PA.14 function <table><tr><th>EBI_HB_EN[7]</th><th>EBI_EN</th><th>PA14_SC2RST</th><th>GPA_MFP[14]</th><th>PA.14 Function</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>PWM2 (PWM)</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC2RST</td></tr><tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD15</td></tr></table>

<table>
<tr>
<td>[12]</td>
<td>PA15_SC2PWR</td>
<td>PA.15 Pin Function Selection
Bits PA15_SC2PWR (ALT_MFP1[12]), PA15_I2SMCLK (ALT_MFP[9]) and GPA_MFP[15] determine the PA.15 function
<table>
<tr><th>PA15_SC2PWR</th><th>PA15_I2SMCLK</th><th>GPA_MFP[15]</th><th>PA.15 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>PWM3 (PWM)</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>I2SMCLK</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>SC2PWR</td></tr>
</table>
</td>
</tr>
<tr>
<td>[11]</td>
<td>PA12_SC2DAT</td>
<td>PA.12 Pin Function Selection
Bits EBI_HB_EN[5] (ALT_MFP[21]), EBI_EN (ALT_MFP[11]), PA12_SC2DAT (ALT_MFP1[11]) and GPA_MFP[12] determine the PA.12 function
<table>
<tr><th>EBI_HB_EN[5]</th><th>EBI_EN</th><th>PA12_SC2DAT</th><th>GPA_MFP[12]</th><th>PA.12 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>PWM0 (PWM)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC2DAT</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD13</td></tr>
</table>
</td>
</tr>
<tr>
<td>[10]</td>
<td>PA13_SC2CLK</td>
<td>PA.13 Pin Function Selection
Bits EBI_HB_EN[6] (ALT_MFP[22]), EBI_EN (ALT_MFP[11]), PA13_SC2CLK (ALT_MFP1[10]) and GPA_MFP[13] determine the PA.13 function
<table>
<tr><th>EBI_HB_EN[6]</th><th>EBI_EN</th><th>PA13_SC2CLK</th><th>GPA_MFP[13]</th><th>PA.13 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>PWM1 (PWM)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC2CLK</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD14</td></tr>
</table>
</td>
</tr>
<tr>
<td>[9]</td>
<td>PC7_SC1CD</td>
<td>PC.7 Pin Function Selection
Bits EBI_EN (ALT_MFP[11]), PC7_SC1CD (ALT_MFP1[9]) and GPC_MFP[7] determine the PC.7 function
<table>
<tr><th>EBI_EN</th><th>PC7_SC1CD</th><th>GPC_MFP[7]</th><th>PC.7 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>CPN0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>SC1CD</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>AD5</td></tr>
</table>
</td>
</tr>
</table>

<table>
<tr><td>[8]</td><td>PA5_SC1RST</td><td>PA.5 Pin Function Selection
Bits EBI_HB_EN[0] (ALT_MFP[16]), EBI_EN (ALT_MFP[11]), PA5_SC1RST (ALT_MFP1[8]) and GPA_MFP[5] determine the PA.5 function
<table>
<tr><th>EBI_HB_EN[0]</th><th>EBI_EN</th><th>PA5_SC1RST</th><th>GPA_MFP[5]</th><th>PA.5 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC5</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC1RST</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD8</td></tr>
</table></td></tr>
<tr><td>[7]</td><td>PA4_SC1PWR</td><td>PA.4 Pin Function Selection
Bits EBI_HB_EN[1] (ALT_MFP[17]), EBI_EN (ALT_MFP[11]), PA4_SC1PWR (ALT_MFP1[7]) and GPA_MFP[4] determine the PA.4 function
<table>
<tr><th>EBI_HB_EN[1]</th><th>EBI_EN</th><th>PA4_SC1PWR</th><th>GPA_MFP[4]</th><th>PA.4 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC4</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC1PWR</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD9</td></tr>
</table></td></tr>
<tr><td>[6]</td><td>PA7_SC1DAT</td><td>PA.7 Pin Function Selection
Bits EBI_EN (ALT_MFP[11]), PA7_SC1DAT (ALT_MFP1[6]), PA7_S21 (ALT_MFP[2]) and GPA_MFP[7] determine the PA.7 function
<table>
<tr><th>EBI_EN</th><th>PA7_SC1DAT</th><th>PA7_S21</th><th>GPA_MFP[7]</th><th>PA.7 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC7</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SPISS21 (SPI2)</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>SC1DAT</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>AD6</td></tr>
</table></td></tr>
<tr><td>[5]</td><td>PA6_SC1CLK</td><td>PA.6 Pin Function Selection
Bits EBI_EN (ALT_MFP[11]), PA6_SC1CLK (ALT_MFP1[5]) and GPA_MFP[6] determine the PA.6 function
<table>
<tr><th>EBI_EN</th><th>PA6_SC1CLK</th><th>GPA_MFP[6]</th><th>PA.6 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>ADC6</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>SC1CLK</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>AD7</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>[4]</td><td>PC6_SC0CD</td><td>PC.6 Pin Function Selection
Bits EBI_EN (ALT_MFP[11]), PC6_SC0CD (ALT_MFP1[4]) and GPC_MFP[6] determine the PC.6 function
<table>
<tr><th>EBI_EN</th><th>PC6_SC0CD</th><th>GPC_MFP[6]</th><th>PC.6 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>CPP0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>SC0CD</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>AD4</td></tr>
</table></td></tr>
<tr><td>[3]</td><td>PA1_SC0RST</td><td>PA.1 Pin Function Selection
Bit EBI_HB_EN[4] (ALT_MFP[20]), EBI_EN (ALT_MFP[11]), PA1_SC0RST (ALT_MFP1[3]) and GPA_MFP[1] determine the PA.1 function
<table>
<tr><th>EBI_HB_EN[4]</th><th>EBI_EN</th><th>PA1_SC0RST</th><th>GPA_MFP[1]</th><th>PA.1 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC0RST</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD12</td></tr>
</table></td></tr>
<tr><td>[2]</td><td>PA0_SC0PWR</td><td>PA.0 Pin Function Selection
Bit PA0_SC0PWR (ALT_MFP1[2]) and GPA_MFP[0] determine the PA.0 function
<table>
<tr><th>PA0_SC0PWR</th><th>GPA_MFP[0]</th><th>PA.0 Function</th></tr>
<tr><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>ADC0</td></tr>
<tr><td>1</td><td>1</td><td>SC0PWR</td></tr>
</table></td></tr>
<tr><td>[1]</td><td>PA3_SC0DAT</td><td>PA.3 Pin Function Selection
Bits EBI_HB_EN[2] (ALT_MFP[18]), EBI_EN (ALT_MFP[11]), PA3_SC0DAT (ALT_MFP1[1]) and GPA_MFP[3] determine the PA.3 function
<table>
<tr><th>EBI_HB_EN[2]</th><th>EBI_EN</th><th>PA3_SC0DAT</th><th>GPA_MFP[3]</th><th>PA.3 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC3</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC0DAT</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD10</td></tr>
</table></td></tr>
<tr><td>[0]</td><td>PA2_SC0CLK</td><td>PA.2 Pin Function Selection
Bits EBI_HB_EN[3] (ALT_MFP[19]), EBI_EN (ALT_MFP[11]), PA2_SC0CLK (ALT_MFP1[0]) and GPA_MFP[2] determine the PA.2 function
<table>
<tr><th>EBI_HB_EN[3]</th><th>EBI_EN</th><th>PA2_SC0CLK</th><th>GPA_MFP[2]</th><th>PA.2 Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ADC2</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>SC0CLK</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>AD11</td></tr>
</table></td></tr>
</table>

HIRC Trim Control Register (IRCTRIMCTL)

Register	Offset	R/W	Description	Reset Value
IRCTRIMCTL	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** When this bit is set to 1, the trim operation is stopped if clock is inaccuracy. When this bit is set to 0, the trim operation is keep going 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.

HIRC Trim Interrupt Enable Register (IRCTRIMIEN)

Register	Offset	R/W	Description	Reset Value
IRCTRIMIEN	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 is set during auto trim operation. An interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = CLKERR_INT status to trigger an interrupt to CPU Disabled. 1 = CLKERR_INT 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. If this bit is high and TRIM_FAIL_INT 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 status to trigger an interrupt to CPU Disabled. 1 = TRIM_FAIL_INT status to trigger an interrupt to CPU Enabled.
[0]	**Reserved**	Reserved.

HIRC Trim Interrupt Status Register (IRCTRIMINT)

Register	Offset	R/W	Description	Reset Value
IRCTRIMINT	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 will be cleared to 00 by hardware automatically if CLKERR_STOP_EN is set to 1. If this bit is set and CLKERR_IEN 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 will be cleared to 00 by hardware automatically. If this bit is set and TRIM_FAIL_IEN 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.

Register Write Protection Register (REGWRPROT)

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.

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.

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: **SYST_BA = 0xE000_E000**				
SYST_CSR	SYST_BA+0x10	R/W	SysTick Control and Status Register	0x0000_0000
SYST_RVR	SYST_BA+0x14	R/W	SysTick Reload Value Register	0xXXXX_XXXX
SYST_CVR	SYST_BA+0x18	R/W	SysTick Current Value Register	0xXXXX_XXXX

6.2.9.2 System Timer Control Register Description

SysTick Control and Status Register (SYST_CSR)

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

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**COUNTFLAG**	**System Tick Counter Flag** Returns 1 If Timer Counted to 0 Since Last Time this Register Was Read. COUNTFLAG is set by a count transition from 1 to 0. COUNTFLAG is cleared on read or by a write to the Current Value register.
[15:3]	**Reserved**	Reserved.
[2]	**CLKSRC**	**System Tick Clock Source Selection** 0 = Clock source is (optional) external reference clock. 1 = Core clock used for SysTick.
[1]	**TICKINT**	**System Tick Interrupt Enable Bit** 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 register write in software will not cause SysTick to be pended.
[0]	**ENABLE**	**System Tick Counter Enable Bit** 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	SYST_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	SYST_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. Unsupported bits RAZ (see SysTick Reload Value register).

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 core and provides following features:

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

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

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

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

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

6.2.10.1 Exception Model and System Interrupt Map

The following table lists the exception model supported by NuMicro® NUC100 series. Software can set four levels of priority on some of these exceptions as well as on all interrupts. The highest user-configurable priority is denoted as "0" and the lowest priority is denoted as "3". The default priority of all the user-configurable interrupts is "0". Note that priority "0" is treated as the fourth priority on the system, after three system exceptions "Reset", "NMI" and "Hard Fault".

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

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

Table 6-6 Exception Model

Vector Number	Interrupt Number (Bit In Interrupt Registers)	Interrupt Name	Source IP	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[15:0]/PB[13:0]
21	5	**GPCDEF_INT**	GPIO	External interrupt from PC[15:0]/PD[15:0]/PE[15:0]/ PF[3:0]
22	6	**PWMA_INT**	PWM0~3	PWM0, PWM1, PWM2 and PWM3 interrupt
23	7	**PWMB_INT**	PWM4~7	PWM4, PWM5, PWM6 and PWM7 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	**SPI2_INT**	SPI2	SPI2 interrupt
33	17	**SPI3_INT**	SPI3	SPI3 interrupt
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	**SC012_INT**	SC0/1/2	SC0, SC1 and SC2 interrupt
39	23	**USB_INT**	USBD	USB 2.0 FS Device interrupt
40	24	**PS2_INT**	PS/2	PS/2 interrupt
41	25	**ACMP_INT**	ACMP	Analog Comparator-0 or Comaprator-1 interrupt
42	26	**PDMA_INT**	PDMA	PDMA interrupt
43	27	**I2S_INT**	I^2S	I^2S interrupt

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 6-7 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 6-8 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: **NVIC_BA = 0xE000_E000**				
NVIC_ISER	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-Enable Control Register	0x0000_0000
NVIC_ICER	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000
NVIC_ISPR	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000
NVIC_ICPR	NVIC_BA+0x280	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000
NVIC_IPR0	NVIC_BA+0x400	R/W	IRQ0 ~ IRQ3 Priority Control Register	0x0000_0000
NVIC_IPR1	NVIC_BA+0x404	R/W	IRQ4 ~ IRQ7 Priority Control Register	0x0000_0000
NVIC_IPR2	NVIC_BA+0x408	R/W	IRQ8 ~ IRQ11 Priority Control Register	0x0000_0000
NVIC_IPR3	NVIC_BA+0x40C	R/W	IRQ12 ~ IRQ15 Priority Control Register	0x0000_0000
NVIC_IPR4	NVIC_BA+0x410	R/W	IRQ16 ~ IRQ19 Priority Control Register	0x0000_0000
NVIC_IPR5	NVIC_BA+0x414	R/W	IRQ20 ~ IRQ23 Priority Control Register	0x0000_0000
NVIC_IPR6	NVIC_BA+0x418	R/W	IRQ24 ~ IRQ27 Priority Control Register	0x0000_0000
NVIC_IPR7	NVIC_BA+0x41C	R/W	IRQ28 ~ IRQ31 Priority Control Register	0x0000_0000

6.2.10.5 NVIC Control Register Description

IRQ0 ~ IRQ31 Set-Enable Control Register (NVIC_ISER)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER	NVIC_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 within a group of 32. Each Bit Represents an Interrupt Number From IRQ0 ~ IRQ31 (Vector Number From 16 ~ 47) 1 = Associated interrupt Enabled. 0 = No effect. The register reads back with the current enable state.

IRQ0 ~ IRQ31 Clear-Enable Control Register (NVIC_ICER)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER	NVIC_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 within a group of 32. Each Bit Represents an Interrupt Number From IRQ0 ~ IRQ31 (Vector Number From 16 ~ 47) 0 = No effect. 1 = Associated interrupt Enabled. The register reads back with the current enable state.

IRQ0 ~ IRQ31 Set-Pending Control Register (NVIC_ISPR)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR	NVIC_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** 0 = No effect. 1 = Set pending state of the associated interrupt under software control. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). The register reads back with the current pending state.

IRQ0 ~ IRQ31 Clear-Pending Control Register (NVIC_ICPR)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR	NVIC_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** 0 = No effect. 1 = Remove the pending state of associated interrupt under software control. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). The register reads back with the current pending state.

IRQ0 ~ IRQ3 Priority Register (NVIC_IPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR0	NVIC_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	NVIC_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	NVIC_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	NVIC_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	NVIC_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	NVIC_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	NVIC_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	NVIC_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® NUC100 series 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/D/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	IRQ16 (SPI2) Interrupt Source Identity	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (SPI3)) Interrupt Source Identity	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	IRQ22 (SC0/1/2) Interrupt Source Identity	0xXXXX_XXXX

IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USBD) Interrupt Source Identity	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	R	IRQ24 (PS/2) Interrupt Source Identity	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (ACMP) Interrupt Source Identity	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (PDMA) Interrupt Source Identity	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	IRQ27 (I^2S) Interrupt Source Identity	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

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/D/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	IRQ16 (SPI2) Interrupt Source Identity	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (SPI3)) Interrupt Source Identity	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	IRQ22 (SC0/1/2) Interrupt Source Identity	0xXXXX_XXXX
IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USBD) Interrupt Source Identity	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	R	IRQ24 (PS/2) Interrupt Source Identity	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (ACMP) Interrupt Source Identity	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (PDMA) Interrupt Source Identity	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	IRQ27 (I^2S) Interrupt Source Identity	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: GPD_INT

			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: PWM7_INT Bit2: PWM6_INT 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+0x40	16	Bit2: 0 Bit1: 0 Bit0: SPI2_INT
[2:0]	INT_BA+0x44	17	Bit2: 0 Bit1: 0 Bit0: SPI3_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+0x58	22	Bit2: SC2_INT Bit1: SC1_INT Bit0: SC0_INT
[2:0]	INT_BA+0x5C	23	Bit2: 0 Bit1: 0 Bit0: USB_INT
[2:0]	INT_BA+0x60	24	Bit2: 0 Bit1: 0 Bit0: PS2_INT
[2:0]	INT_BA+0x64	25	Bit2: 0 Bit1: 0 Bit0: ACMP_INT
[2:0]	INT_BA+0x68	26	Bit2: 0 Bit1: 0 Bit0: PDMA_INT
[2:0]	INT_BA+0x6C	27	Bit2: 0 Bit1: 0 Bit0: I2S_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	
[31:9]	**Reserved**	Reserved.
[8]	**NMI_EN**	**NMI Interrupt Enable Bit (Write Protect)** 0 = NMI interrupt Disabled. 1 = NMI interrupt Enabled. This bit is the protected bit. It means programming this needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Reference the register REGWRPROT at address GCR_BA+0x100
[7:5]	**Reserved**	Reserved.
[4:0]	**NMI_SEL**	**NMI Interrupt Source Select** 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

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".

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

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** (ARM = 0x41)
[23:20]	**Reserved**	Reserved.
[19:16]	**PART**	Read as 0xC for ARMv6-M parts
[15:4]	**PARTNO**	Read as 0xC20.
[3:0]	**REVISION**	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: 0 = No effect. 1 = Changes NMI exception state to pending. Read: 0 = NMI exception not pending. 1 = NMI exception pending. **Note:** Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it detects a write of 1 to this bit. Entering the handler then clears this bit to 0. This means a read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler.
[30:29]	**Reserved**	Reserved.
[28]	**PENDSVSET**	**PendSV Set-pending Bit** Write: 0 = No effect. 1 = Changes PendSV exception state to pending. Read: 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: 0 = No effect. 1 = Removes the pending state from the PendSV exception. **Note:** 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:

		0 = No effect. 1 = Changes SysTick exception state to pending. Read: 0 = SysTick exception is not pending. 1 = SysTick exception is pending.
[25]	**PENDSTCLR**	**SysTick Exception Clear-pending Bit** Write: 0 = No effect. 1 = Removes the pending state from the SysTick exception. **Note:** 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. **Note:** 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	VECTCLRACTIVE	Reserved

Bits	Description	
[31:16]	**VECTORKEY**	When writing this register, this field should be 0x05FA, otherwise the write action will be unpredictable.
[15:3]	**Reserved**	Reserved.
[2]	**SYSRESETREQ**	**System Reset Request** Writing this bit to 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** Setting this bit to 1 will clear all active state information for fixed and configurable exceptions. The bit is a write only bit and can only be written when the core is halted. **Note:** It is the debugger's responsibility to re-initialize the stack.
[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**	**Controls Whether the Processor Uses Sleep or Deep Sleep As Its Low Power Mode:** 0 = Sleep. 1 = Deep sleep.
[1]	**SLEEPONEXIT**	**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, on return from an ISR to Thread mode. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.
[0]	**Reserved**	Reserved.

System Handler Priority Register 2 (SHPR2)

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

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

Bits	Description	
[31:30]	**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 waits for wake-up interrupt source triggered to leave Power-down mode. In the Power-down mode, the clock controller turns off the external 4~24 MHz high speed crystal and internal 22.1184 MHz high speed oscillator to reduce the overall system power consumption.

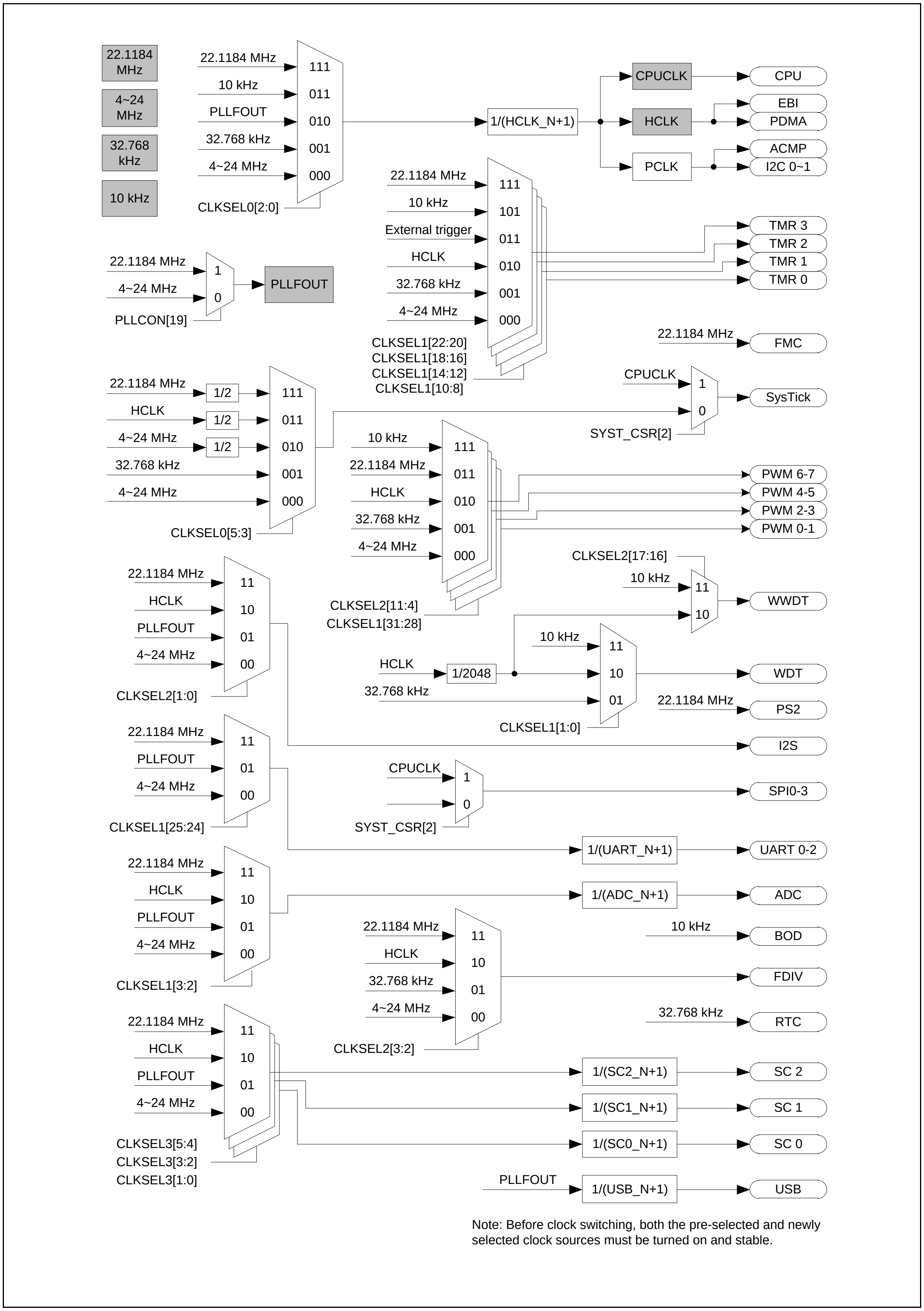


Figure 6-10 Clock Generator Global View Diagram

6.3.2 Clock Generator

The clock generator consists of 5 clock sources as listed below:

- One external 32.768 kHz low speed crystal
- One external 4~24 MHz high speed crystal
- One programmable PLL FOUT (PLL source consists of external 4~24 MHz high speed crystal and internal 22.1184 MHz high speed oscillator)
- One internal 22.1184 MHz high speed oscillator
- One internal 10 kHz low speed oscillator

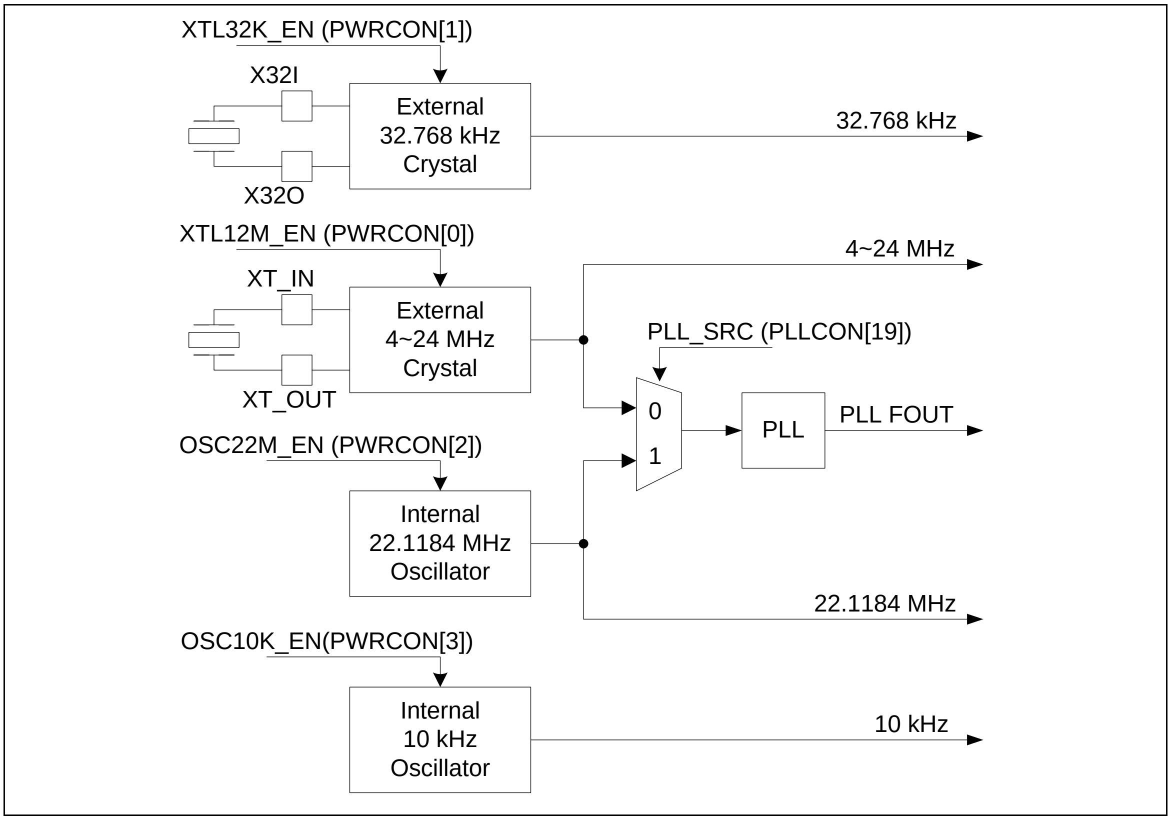


Figure 6-11 Clock Generator Block Diagram

6.3.3 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 6-12.

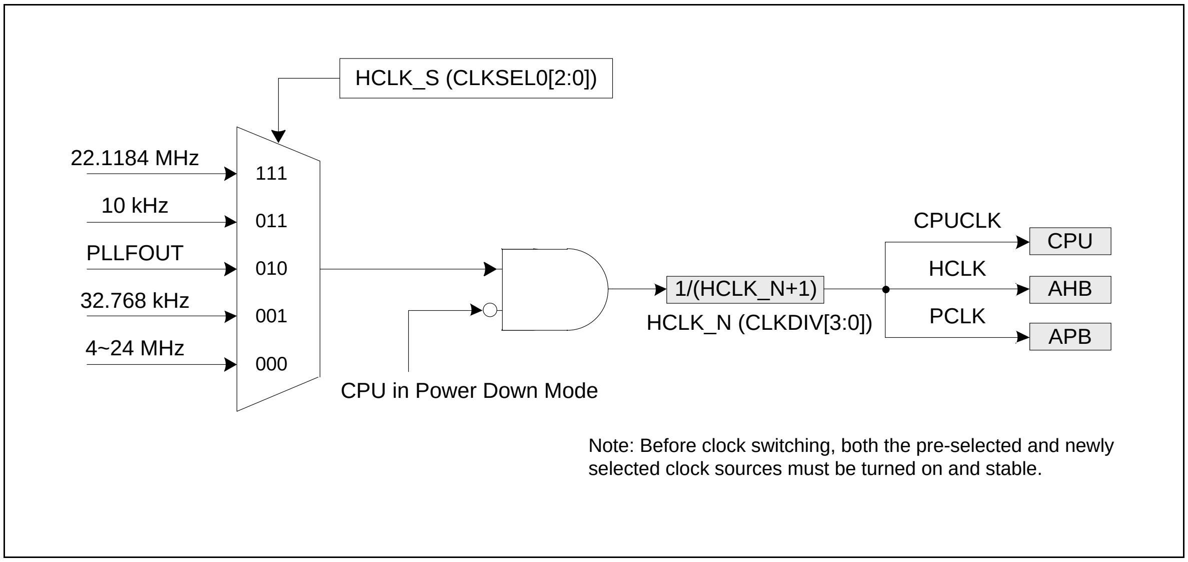


Figure 6-12 System Clock Block Diagram

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 6-13.

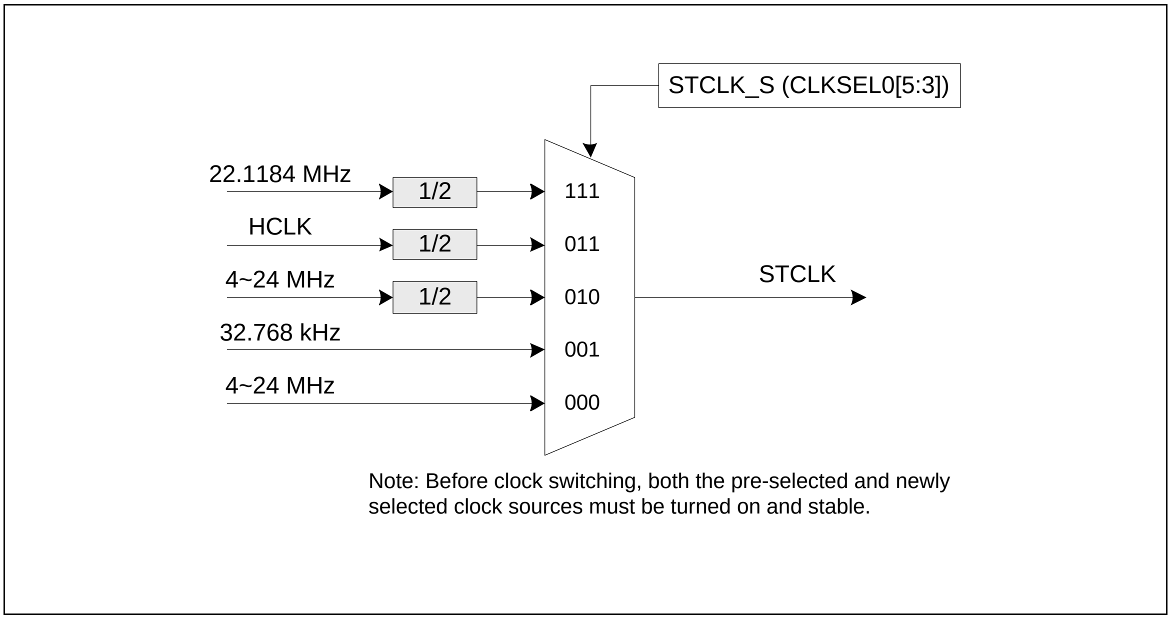


Figure 6-13 SysTick Clock Control Block Diagram

6.3.4 Peripherals Clock

The peripherals clock can be selected as different clock source depends on the clock source select control registers (CLKSEL1, CLKSEL2 and CLKSEL3). Please refer to the register description in 6.3.8

6.3.5 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
  - Internal 10 kHz low speed oscillator clock
  - External 32.768 kHz low speed crystal clock
- Peripherals Clock (when IP adopt external 32.768 kHz low speed crystal oscillator or 10 kHz low speed oscillator as clock source)

6.3.6 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.

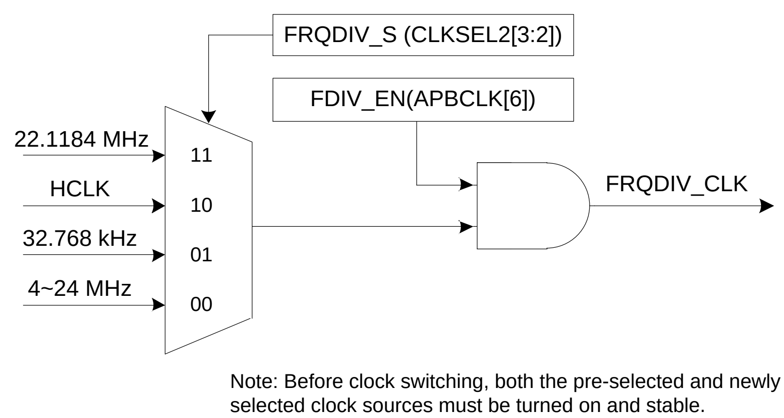


Figure 6-14 Clock Source of Frequency Divider

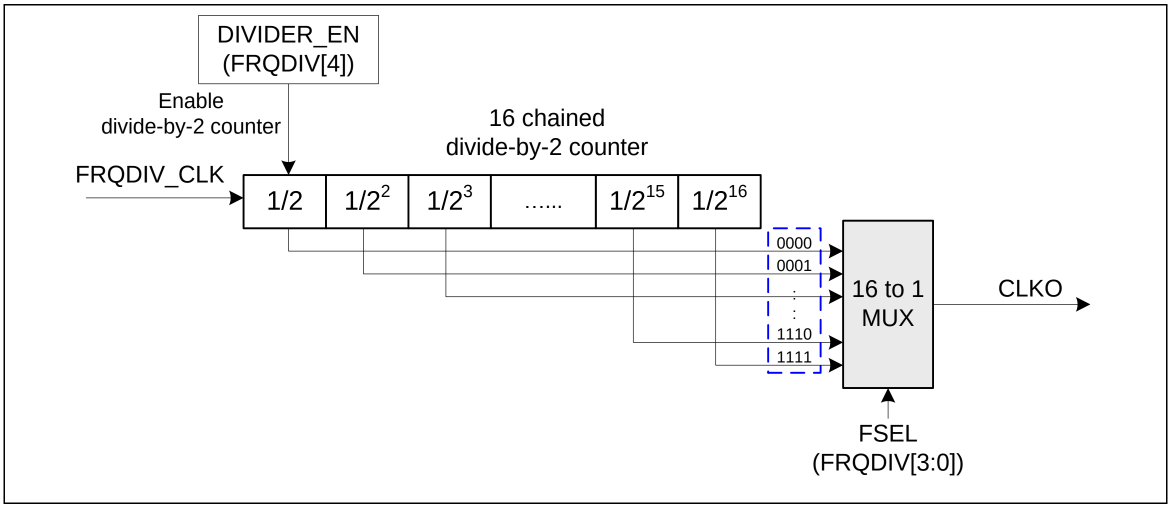


Figure 6-15 Frequency Divider Block Diagram

6.3.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
CLK Base Address: **CLK_BA = 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_0000
APBCLK	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register	0x0000_000X
APBCLK1	CLK_BA+0x30	R/W	APB Devices Clock Enable Control Register 1	0x0000_0000
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
CLKSEL3	CLK_BA+0x34	R/W	Clock Source Select Control Register 3	0x0000_003F
CLKDIV	CLK_BA+0x18	R/W	Clock Divider Number Register	0x0000_0000
CLKDIV1	CLK_BA+0x38	R/W	Clock Divider Number Register 1	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

6.3.8 Register Description

System Power-down Control Register (PWRCON)

Except the BIT[6], all the other bits are protected, program 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							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	
[31:9]	**Reserved**	Reserved.
[8]	**PD_WAIT_CPU**	**this Bit Control the Power-down Entry Condition (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.
[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, then the chip enters Power-down mode immediately after the PWR_DOWN_EN bit set. (b) if the PD_WAIT_CPU is 1, then 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, external 4~24 MHz high speed crystal oscillator and the internal 22.1184 MHz high speed oscillator will be disabled in this mode, but the external 32.768 kHz low speed crystal and internal 10 kHz 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 external 32.768 kHz 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.

[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, ACMP, 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 = Disabled. 1 = Enabled. **Note:** The interrupt will occur when both PD_WU_STS and PD_WU_INT_EN are high.
[4]	**PD_WU_DLY**	**Enable the Wake-up Delay Counter (Write Protect)** When the chip wakes up from Power-down mode, the clock control will delay certain clock cycles to wait system clock stable. The delayed clock cycle is 4096 clock cycles when chip 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.
[3]	**OSC10K_EN**	**Internal 10 KHz Low Speed Oscillator Enable Bit (Write Protect)** 0 = Internal 10 kHz low speed oscillator Disabled. 1 = Internal 10 kHz low speed oscillator Enabled.
[2]	**OSC22M_EN**	**Internal 22.1184 MHz High Speed Oscillator Enable Bit (Write Protect)** 0 = Internal 22.1184 MHz high speed oscillator Disabled. 1 = Internal 22.1184 MHz high speed oscillator Enabled.
[1]	**XTL32K_EN**	**External 32.768 KHz Low Speed Crystal Enable Bit (Write Protect)** 0 = External 32.768 kHz low speed crystal oscillator Disabled. 1 = External 32.768 kHz low speed crystal oscillator Enabled (Normal operation).
[0]	**XTL12M_EN**	**External 4~24 MHz High Speed Crystal 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 external 4~24 MHz high speed crystal, this bit is set to 1 automatically. 0 = External 4~24 MHz high speed crystal oscillator Disabled. 1 = External 4~24 MHz high speed crystal oscillator Enabled.

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 peripheral clock source are selected as 10 kHz or 32.768 kHz.

Table 6-9 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 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 and EBI clock.

Register	Offset	R/W	Description	Reset Value
AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_0000

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

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

APB Devices Clock Enable Control 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
PS2_EN	ACMP_EN	I2S_EN	ADC_EN	USBD_EN	Reserved		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PWM67_EN	PWM45_EN	PWM23_EN	PWM01_EN	Reserved	UART2_EN	UART1_EN	UART0_EN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SPI3_EN	SPI2_EN	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]	**PS2_EN**	**PS/2 Clock Enable Bit** 0 = PS/2 clock Disabled. 1 = PS/2 clock Enabled.
[30]	**ACMP_EN**	**Analog Comparator Clock Enable Bit** 0 = Analog Comparator clock Disabled. 1 = Analog Comparator clock Enabled.
[29]	**I2S_EN**	**I^2S Clock Enable Bit** 0 = I^2S clock Disabled. 1 = I^2S clock Enabled.
[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 Enabled. 1 = USB clock Enabled.
[26:24]	**Reserved**	Reserved.
[23]	**PWM67_EN**	**PWM_67 Clock Enable Bit** 0 = PWM67 clock Disabled. 1 = PWM67 clock Enabled.
[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.

NUC100/120XXXDN TECHNICAL REFERENCE MANUAL

		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]	**SPI3_EN**	**SPI3 Clock Enable Bit** 0 = SPI3 clock Disabled. 1 = SPI3 clock Enabled.
[14]	**SPI2_EN**	**SPI2 Clock Enable Bit** 0 = SPI2 clock Disabled. 1 = SPI2 clock Enabled.
[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 from the external 32.768 kHz low speed crystal. 0 = RTC clock Disabled. 1 = RTC clock Enabled.
[0]	**WDT_EN**	**Watchdog Timer Clock Enable Bit (Write Protect)** 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 = Watchdog Timer clock Disabled. 1 = Watchdog Timer clock Enabled.

APB Devices Clock Enable Control Register 1 (APBCLK1)

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

Register	Offset	R/W	Description	Reset Value
APBCLK1	CLK_BA+0x30	R/W	APB Devices Clock Enable Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**SC2_EN**	**SC1_EN**	**SC0_EN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**SC2_EN**	**SC2 Clock Enable Bit** 0 = SC2 clock Disabled. 1 = SC2 clock Enabled.
[1]	**SC1_EN**	**SC1 Clock Enable Bit** 0 = SC1 clock Disabled. 1 = SC1 clock Enabled.
[0]	**SC0_EN**	**SC0 Clock Enable Bit** 0 = SC0 Clock Disabled. 1 = SC0 Clock Enabled.

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		OSC22M_STB	OSC10K_STB	PLL_STB	XTL32K_STB	XTL12M_STB

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**CLK_SW_FAIL**	**Clock Switching Fail Flag** 0 = Clock switching success. 1 = Clock switching failed. This bit is updated when software switches system clock source. If switch target clock is stable, this bit will be set to 0. If switch target clock is not stable, this bit will be set to 1. **Note:** Write 1 to clear the bit to 0.
[6:5]	**Reserved**	Reserved.
[4]	**OSC22M_STB**	**Internal 22.1184 MHz High Speed Oscillator Clock Source Stable Flag (Read Only)** 0 = Internal 22.1184 MHz high speed oscillator clock is not stable or disabled. 1 = Internal 22.1184 MHz high speed oscillator clock is stable.
[3]	**OSC10K_STB**	**Internal 10 KHz Low Speed Oscillator Clock Source Stable Flag (Read Only)** 0 = Internal 10 kHz low speed oscillator clock is not stable or disabled. 1 = Internal 10 kHz low speed oscillator clock is stable.
[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.
[1]	**XTL32K_STB**	**External 32.768 KHz Low Speed Crystal Clock Source Stable Flag (Read Only)** 0 = External 32.768 kHz low speed crystal clock is not stable or disabled. 1 = External 32.768 kHz low speed crystal clock is stable.
[0]	**XTL12M_STB**	**External 4~24 MHz High Speed Crystal Clock Source Stable Flag (Read Only)** 0 = External 4~24 MHz high speed crystal clock is not stable or disabled.

		1 = External 4~24 MHz high speed crystal clock is stable.

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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**STCLK_S**			**HCLK_S**		

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:3]	**STCLK_S**	**Cortex®-M0 SysTick Clock Source Select (Write Protect)** If SYST_CSR[2] = 1, SysTick clock source is from HCLK. If SYST_CSR[2] = 0, SysTick clock source is defined by STCLK_S(CLKSEL0[5:3]). 000 = Clock source from external 4~24 MHz high speed crystal clock. 001 = Clock source from external 32.768 kHz low speed crystal clock. 010 = Clock source from external 4~24 MHz high speed crystal clock/2. 011 = Clock source from HCLK/2. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock/2. **Note:** 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. **Note:** 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 turn on 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 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 PLL clock. 011 = Clock source from internal 10 kHz low speed oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock.

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**
SPI3_S	**SPI2_S**	**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 peripheral clock source of PWM2 and PWM3 defined by PWM23_S list below: 00 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed 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 internal 10 kHz low speed 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 external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed 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 internal 10 kHz low speed oscillator clock.
[27:26]	**Reserved**	Reserved.
[25:24]	**UART_S**	**UART Clock Source Selection** 00 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from PLL clock. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock.
[23]	**Reserved**	Reserved.
[22:20]	**TMR3_S**	**TIMER3 Clock Source Selection** 000 = Clock source from external 4~24 MHz high speed crystal clock. 001 = Clock source from external 32.768 kHz low speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from external trigger. 101 = Clock source from internal 10 kHz low speed oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed 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 clock. 001 = Clock source from external 32.768 kHz low speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from external trigger. 101 = Clock source from internal 10 kHz low speed oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock. Others = reserved.
[15]	**Reserved**	Reserved.
[14:12]	**TMR1_S**	**TIMER1 Clock Source Selection** 000 = Clock source from external 4~24 MHz high speed crystal clock. 001 = Clock source from external 32.768 kHz low speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from external trigger. 101 = Clock source from internal 10 kHz low speed oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock. Others = reserved.
[11]	**Reserved**	Reserved.
[10:8]	**TMR0_S**	**TIMER0 Clock Source Selection** 000 = Clock source from external 4~24 MHz high speed crystal clock. 001 = Clock source from external 32.768 kHz low speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from external trigger. 101 = Clock source from internal 10 kHz low speed oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock. Others = reserved.
[7]	**SPI3_S**	**SPI3 Clock Source Selection**

		0 = Clock source from PLL clock. 1 = Clock source from HCLK.
[6]	**SPI2_S**	**SPI2 Clock Source Selection** 0 = Clock source from PLL clock. 1 = Clock source from HCLK.
[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 external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from PLL clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock.
[1:0]	**WDT_S**	**Watchdog Timer Clock Source Select (Write Protect)** These bits are protected bit, program this need to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100. 00 = Reserved. 01 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 10 = Clock source from HCLK/2048 clock. 11 = Clock source from internal 10 kHz low speed oscillator clock.

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						WWDT_S	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				PWM67_S_E	PWM45_S_E	PWM23_S_E	PWM01_S_E
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PWM67_S		PWM45_S		FRQDIV_S		I2S_S	

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**WWDT_S**	**Window Watchdog Timer Clock Source Selection** 10 = Clock source from HCLK/2048 clock. 11 = Clock source from internal 10 kHz low speed oscillator clock.
[15:12]	**Reserved**	Reserved.
[11]	**PWM67_S_E**	**PWM6 and PWM7 Clock Source Selection Extend** PWM6 and PWM7 used the same Peripheral clock source; both of them used the same prescaler. The Peripheral clock source of PWM6 and PWM7 is defined by PWM67_S (CLKSEL2[7:6]) and PWM67_S_E (CLKSEL2[11]). If PWM67_S_E = 0, the peripheral clock source of PWM6 and PWM7 defined by PWM67_S list below: 00 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock. If PWM67_S_E = 1, the peripheral clock source of PWM6 and PWM7 defined by PWM67_S list below: 00 = Reserved. 01 = Reserved. 10 = Reserved. 11 = Clock source from internal 10 kHz low speed oscillator clock.
[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 external 4~24 MHz high speed crystal oscillator clock.

		01 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed 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 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 peripheral clock source of PWM2 and PWM3 defined by PWM23_S list below: 00 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed 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 internal 10 kHz low speed 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 external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed 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 internal 10 kHz low speed oscillator clock.
[7:6]	**PWM67_S**	**PWM6 and PWM7 Clock Source Selection** PWM6 and PWM7 used the same Peripheral clock source; both of them used the same prescaler. The Peripheral clock source of PWM6 and PWM7 is defined by PWM67_S (CLKSEL2[7:6]) and PWM67_S_E (CLKSEL2[11]). If PWM67_S_E = 0, the peripheral clock source of PWM6 and PWM7 defined by PWM67_S list below: 00 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock. If PWM67_S_E = 1, the peripheral clock source of PWM6 and PWM7 defined by

		PWM67_S list below:. 00 = Reserved. 01 = Reserved. 10 = Reserved. 11 = Clock source from internal 10 kHz low speed oscillator clock.
[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 external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed 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.
[3:2]	**FRQDIV_S**	**Clock Divider Clock Source Selection** 00 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from external 32.768 kHz low speed crystal oscillator clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock.
[1:0]	**I2S_S**	**I^2S Clock Source Selection** 00 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from PLL clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock.

Clock Source Select Control Register 3 (CLKSEL3)

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

Register	Offset	R/W	Description	Reset Value
CLKSEL3	CLK_BA+0x34	R/W	Clock Source Select Control Register 3	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		SC2_S		SC1_S		SC0_S	

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:4]	**SC2_S**	**SC2 Clock Source Selection** 00 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from PLL clock. 10 = HCLK. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock.
[3:2]	**SC1_S**	**SC1 Clock Source Selection** 00 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from PLL clock. 10 = HCLK. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock.
[1:0]	**SC0_S**	**SC0 Clock Source Selection** 00 = Clock source from external 4~24 MHz high speed crystal oscillator clock. 01 = Clock source from PLL clock. 10 = HCLK. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock.

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	
[31:24]	**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).

Clock Divider Register 1 (CLKDIV1)

Register	Offset	R/W	Description	Reset Value
CLKDIV1	CLK_BA+0x38	R/W	Clock Divider Number Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SC2_N							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SC1_N							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0_N							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**SC2_N**	**SC2 Clock Divide Number From SC2 Clock Source** The SC2 clock frequency = (SC2 clock source frequency) / (SC2_N + 1).
[15:8]	**SC1_N**	**SC1 Clock Divide Number From SC1 Clock Source** The SC1 clock frequency = (SC1 clock source frequency) / (SC1_N + 1).
[7:0]	**SC0_N**	**SC0 Clock Divide Number From SC0 Clock Source** The SC0 clock frequency = (SC0 clock source frequency) / (SC0_N + 1).

PLL Control Register (PLLCON)

The PLL reference clock input is from the external 4~24 MHz high speed crystal clock input or from the internal 22.1184 MHz high speed 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 external 4~24 MHz high speed crystal. 1 = PLL source clock from internal 22.1184 MHz high speed 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. $100MHz < FCO = FIN \times \frac{NF}{NR} < 200MHz$
   $120MHz < FCO$ 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			**DIVIDER_EN**	**FSEL**			

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**DIVIDER_EN**	**Frequency Divider Enable Bit** 0 = Frequency Divider Disabled. 1 = Frequency Divider Enabled.
[3:0]	**FSEL**	**Divider Output Frequency Selection Bits** The formula of output frequency is $F_{out} = F_{in}/2^{(N+1)}$. 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® NUC100 xeries has 128/64/32 Kbytes 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® NUC100 series also provides additional DATA Flash for user to store some application dependent data. For 128 Kbytes APROM device, the Data Flash is shared with original 128K program memory and its start address is configurable in Config1. For 64/32 Kbytes APROM device, the Data Flash is fixed at 4K.

6.4.2 Features

- Runs up to 50 MHz with zero wait state for continuous address read access
- All embedded flash memory supports 512 bytes page erase
- 128/64/32 KB application program memory (APROM)
- 4 KB In-System-Programming (ISP) loader program memory (LDROM)
- 4KB Data Flash for 64/32 KB APROM device
- Configurable Data Flash size for 128KB APROM device
- Configurable or fixed 4 KB Data Flash 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:

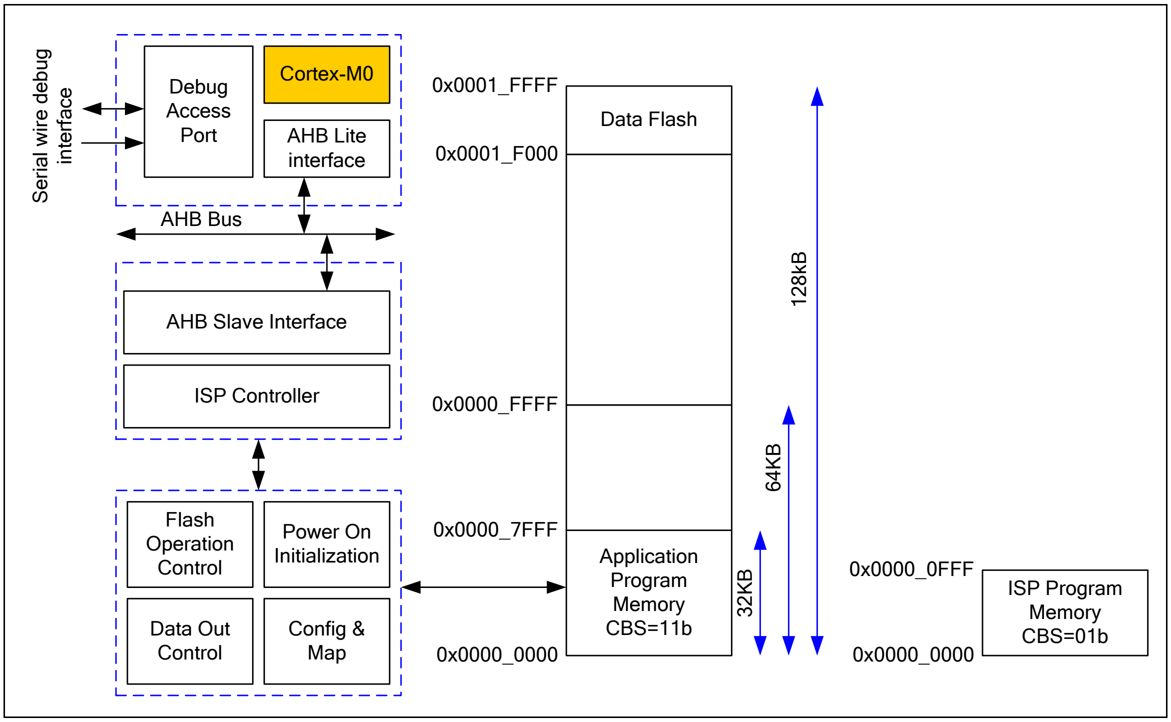


Figure 6-16 Flash Memory Control Block Diagram

6.4.4 Functional Description

6.4.4.1 Flash Memory Organization

The NuMicro® NUC100 series 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 user 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. For 128 KB APROM device, 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). For 64/32 KB APROM devices, Data Flash size is always 4 KB and start address is fixed at 0x0001_F000.

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 ~ 0xFFF. However, if user want to read LDROM by ISP, they still need to read the address of LDROM as 0x0010_0000 ~ 0x0010_0FFF.

Table 6-10 and Figure 6-17 show the address mapping information of APROM, LDROM, Data Flash and user configuration for 32/64 and 128 KB devices.

Block Name	Device Type	Size		Start Address	End Address
APROM	32 KB	32 KB		0x0000_0000	0x0000_7FFF
	64 KB	64 KB		0x0000_0000	0x0000_FFFF
	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	32 KB	4 KB		0x0001_F000	0x0001_FFFF
	64 KB	4 KB		0x0001_F000	
	128 KB	Data Flash Enable	0x20000-DFBADR	DFBADR	
		Data Flash Disable	0 KB	N/A	N/A
LDROM	32/64/128 KB	4 KB		0x0010_0000	0x0010_0FFF
User Configuration	32/64/128 KB	2 words		0x0030_0000	0x0030_0004

Table 6-10 Memory Address Map

The Flash memory organization is shown as Figure 6-17:

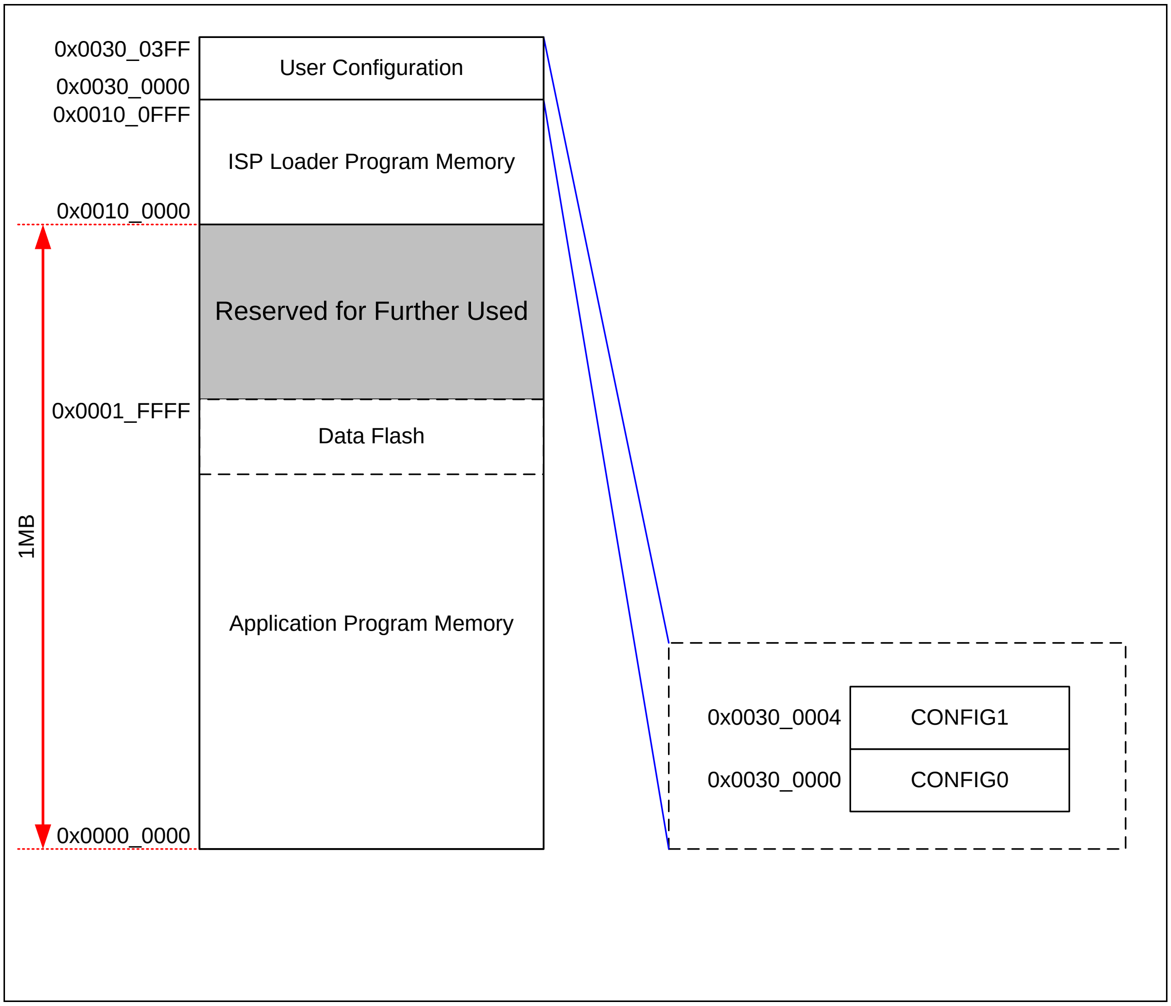


Figure 6-17 Flash Memory Organization

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 Select** 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 (HXT). 111 = Internal RC 22.1184 MHz high speed oscillator clock (HIRC). Others = Reserved. The value of CFOSC will be load to CLKSEL0.HCLK_S[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 = 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.
[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**	**IO 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** When CBS[0] = 0, the LDROM base address is mapping to 0x100000 and APROM base address is mapping to 0x0. User could access both APROM and LDROM without boot switching. In other words, if IAP mode is supported, the code in LDROM and APROM can be called by each other.. 00 = Boot from LDROM with IAP mode. 01 = Boot from LDROM without IAP mode. 10 = Boot from APROM with IAP mode. 11 = Boot from APROM without IAP mode.
[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

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 (Only for 128 KB APROM Device)** For 128 KB APROM device, its 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® NUC100 Series provides In-System-Programming (ISP) feature to enable user to update program memory by a stand-alone ISP firmware. A dedicated 4 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 6-18 shows the memory map when booting from APROM and LDROM.

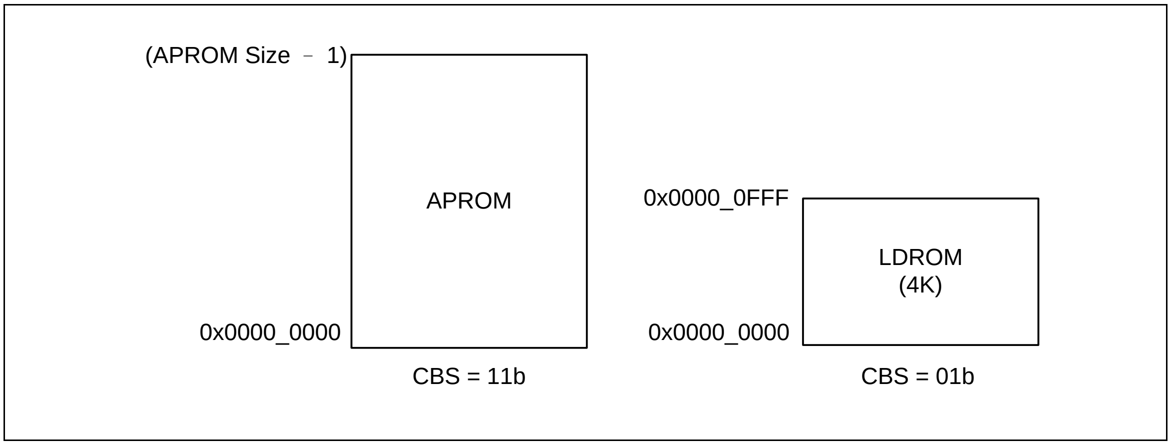


Figure 6-18 Program Executing Range for Booting from APROM and LDROM

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® NUC100 series 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 4 KB LDROM is mapped to 0x0010_0000~ 0x0010_0FFF.

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 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 4 KB LDROM is mapped to 0x0010_0000~0x0010_0FFF.

Please refer to Figure 6-19 for the address map while IAP is activating.

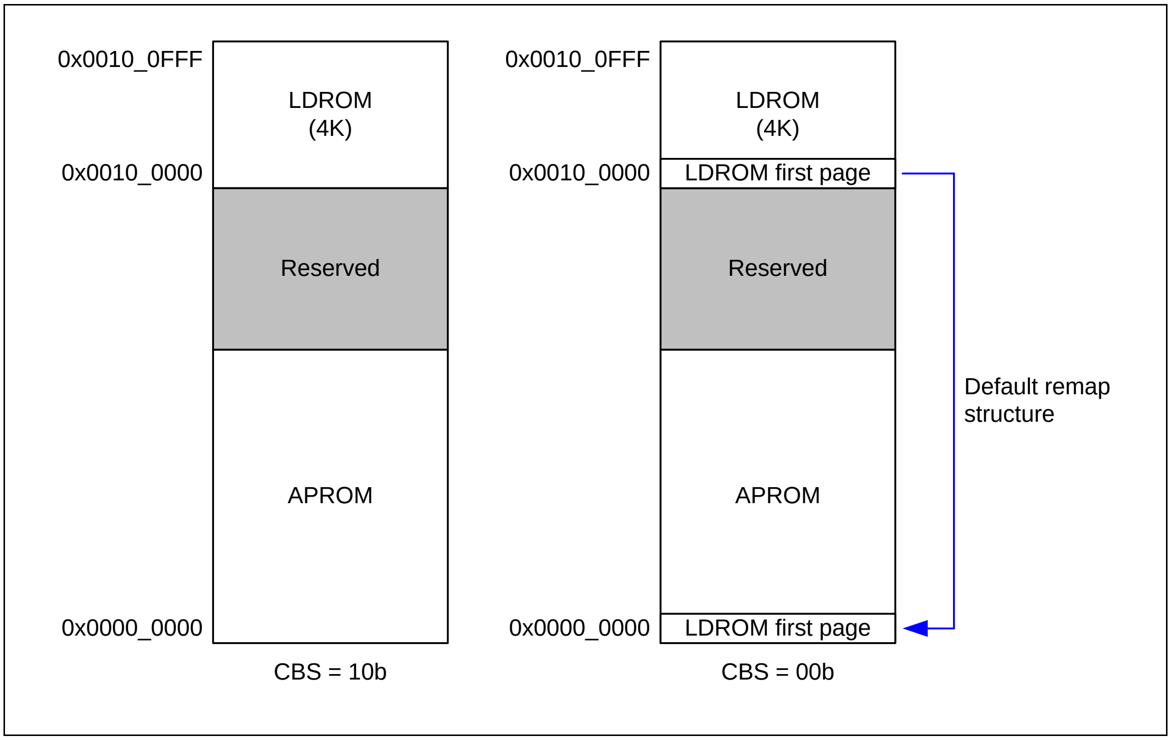


Figure 6-19 Executable Range of Code with IAP Function Enabled

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® NUC100 series 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 receive new 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® NUC100 series 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 following figure.

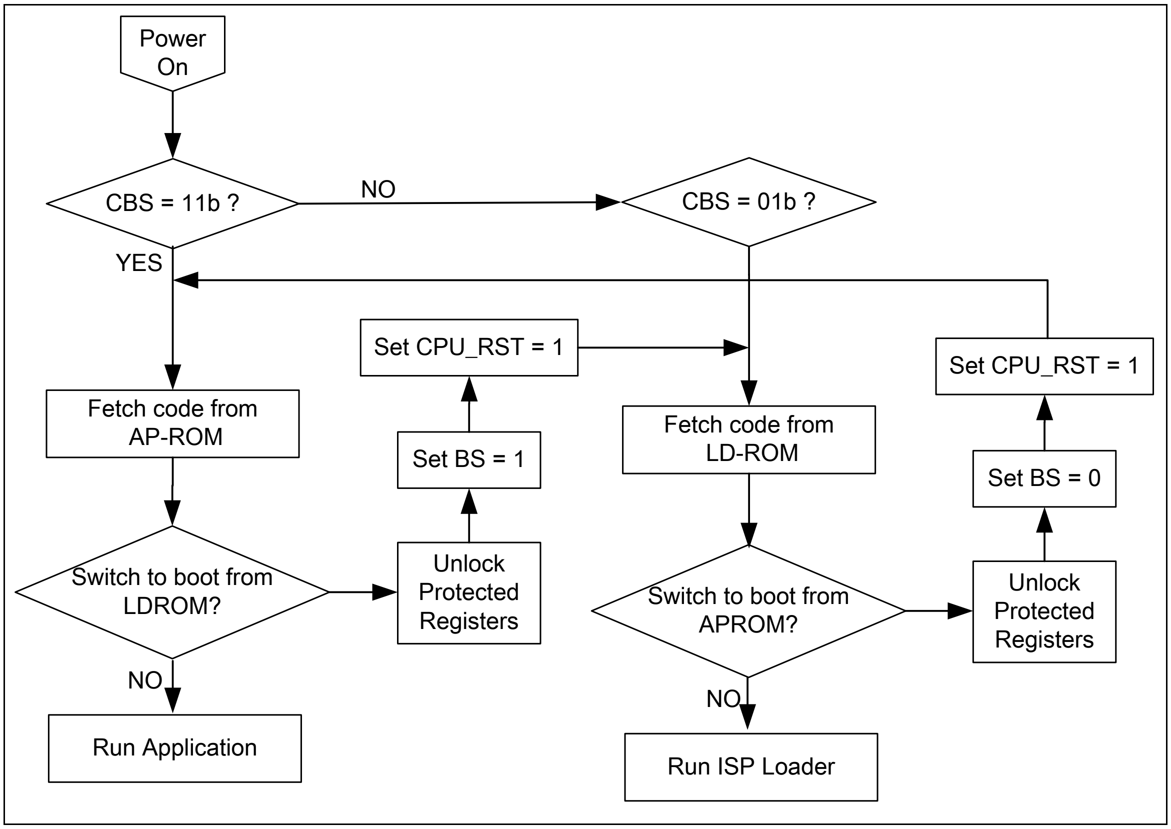


Figure 6-20 Example Flow of Boot Selection by BS Bit

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.

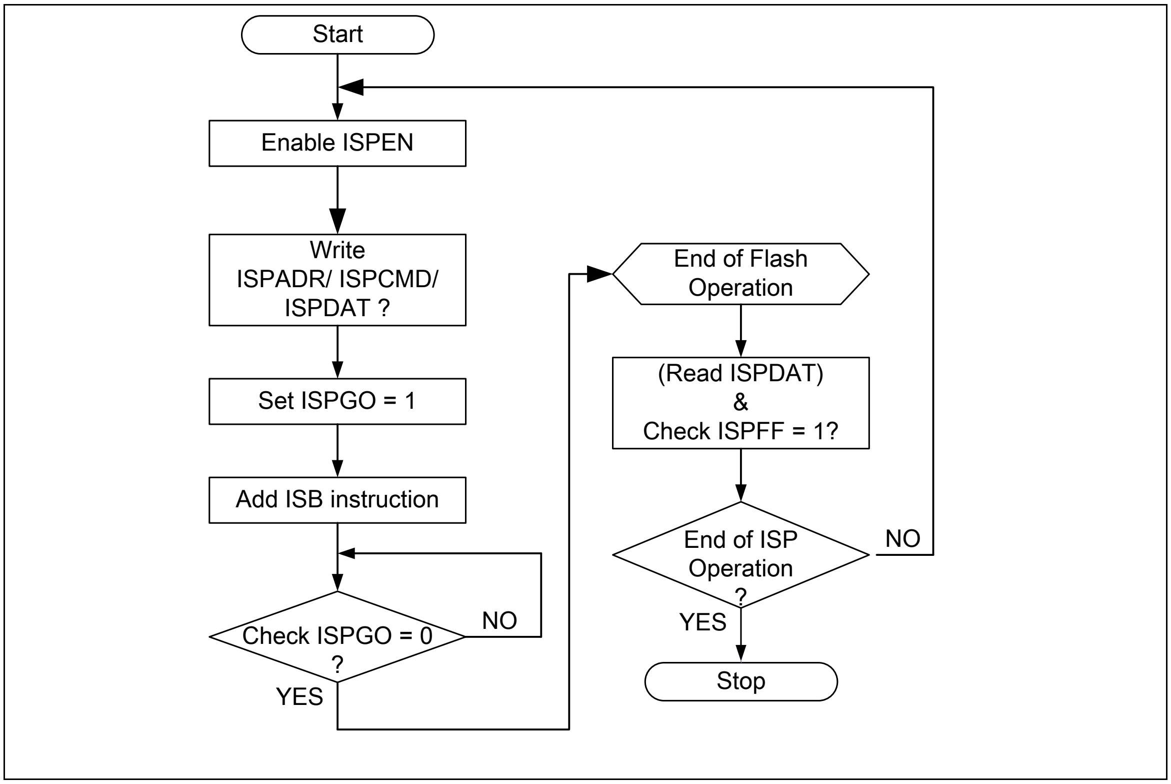


Figure 6-21 ISP Flow Example

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 6-11 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)** LDROM update enable bit. 0 = LDROM cannot be updated. 1 = LDROM can be updated when chip runs in APROM.
[4]	**CFGUEN**	**Enable Config-bits Update by ISP (Write Protect)** 0 = ISP update config-bits Disabled. 1 = ISP update config-bits 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 (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® NUC100 series has a maximum 32Kx32 (128 KB) of 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 (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= FLASH Read. 0x2E= Vector Page Re-Map. 0x21= FLASH Program. 0x22= FLASH Page Erase. 0x04= Read Unique ID. The other commands are invalid.

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 Protect)** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. 0 = ISP operation finished. 1 = ISP progressed. **Note:** 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 (Read Only)** This register indicates Data Flash start address. For 128 KB flash memory device, the Data Flash size is defined by user configuration, register content is loaded from Config1 when chip is powered on but for 64/32 KB device, it is fixed at 0x0001_F000.

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			**LFOM**	**Reserved**			

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**LFOM**	**Low Frequency Optimization Mode (Write Protect)** When chip operation frequency is lower than 25 MHz, chip can work more efficiently by setting this bit to 1 0 = Low frequency optimization mode Disabled. 1 = Low frequency optimization mode Enabled.
[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 Protect)** 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 **Note:** This bit can be cleared by writing '1' to it. **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® NUC100 series LQFP-64 and LQFP-100 package is equipped with an external bus interface (EBI) for accessing an 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

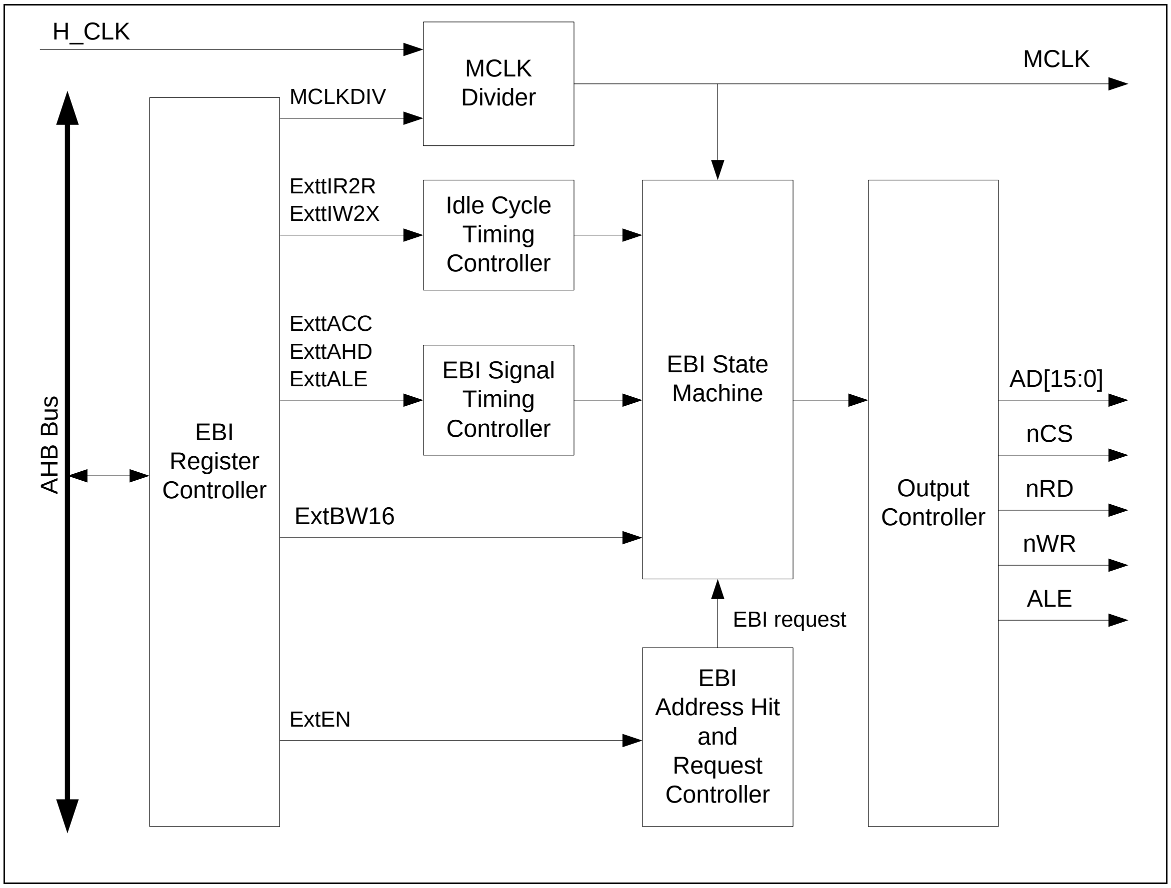


Figure 6-22 EBI Block Diagram

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.

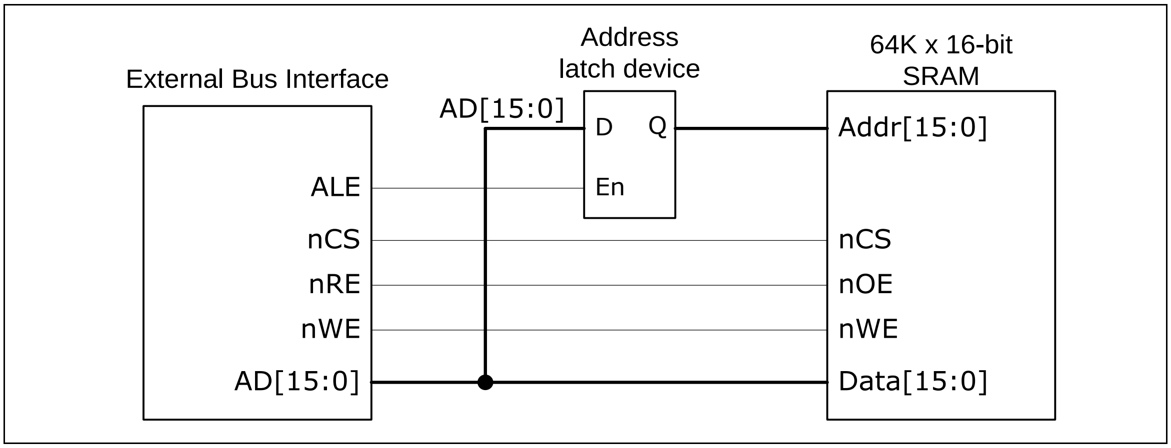


Figure 6-23 Connection of 16-bit EBI Data Width with 16-bit Device

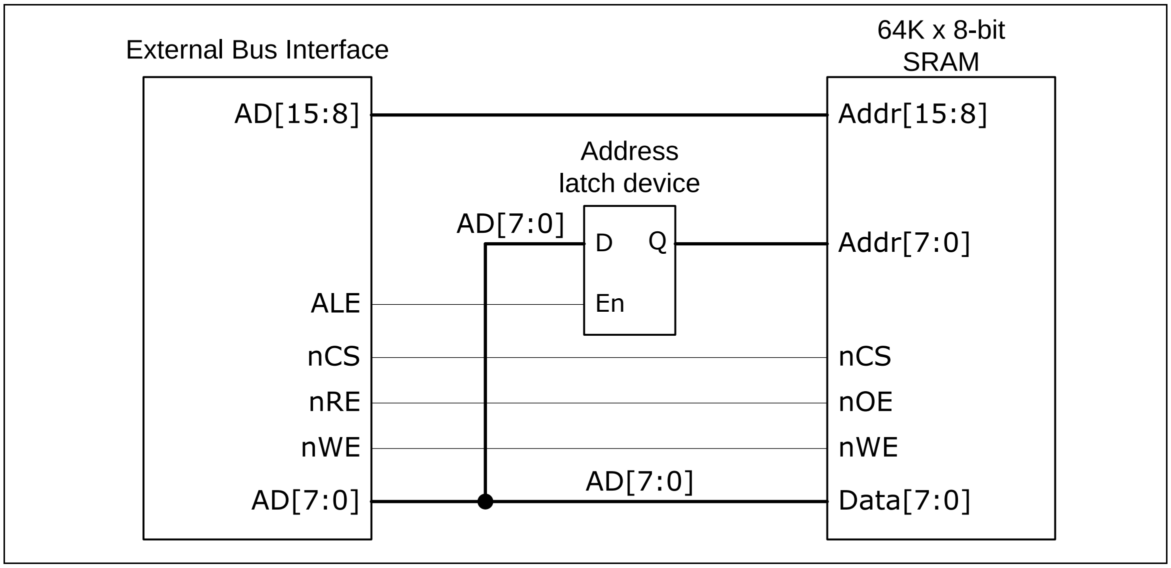


Figure 6-24 Connection of 8-bit EBI Data Width with 8-bit Device

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].

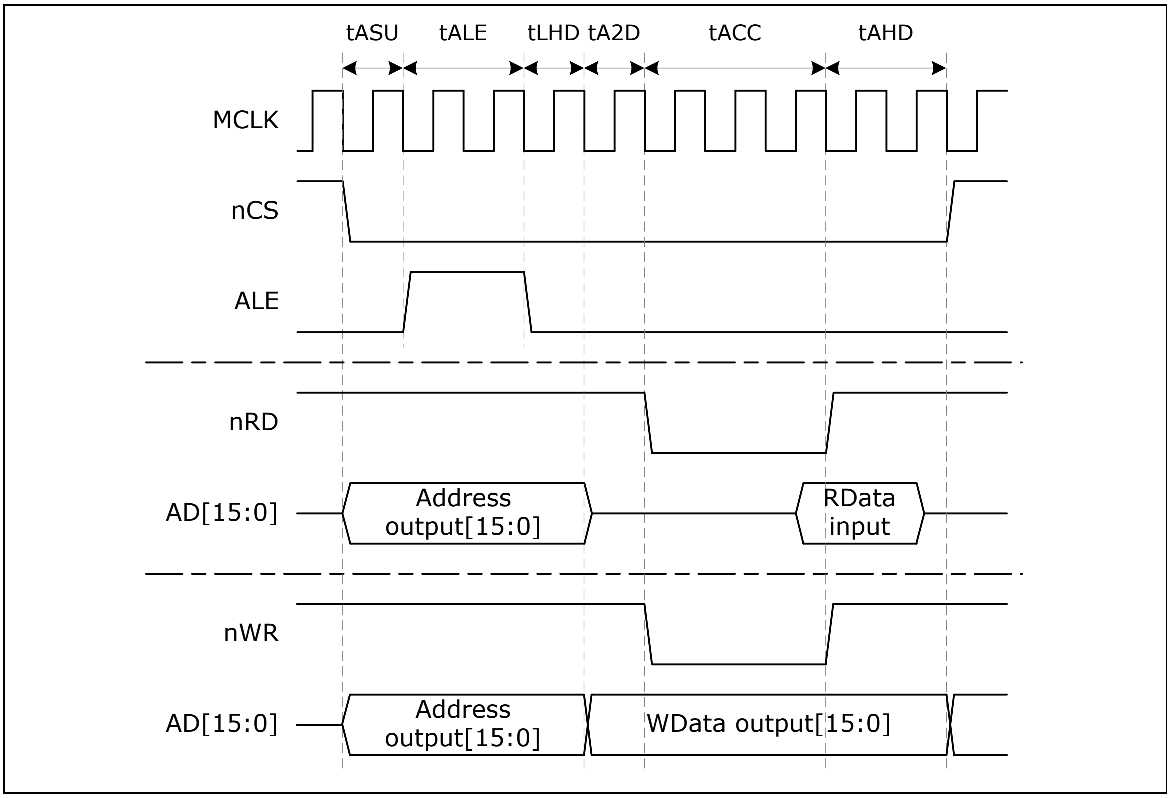


Figure 6-25 Timing Control Waveform for 16-bit Data Width

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.

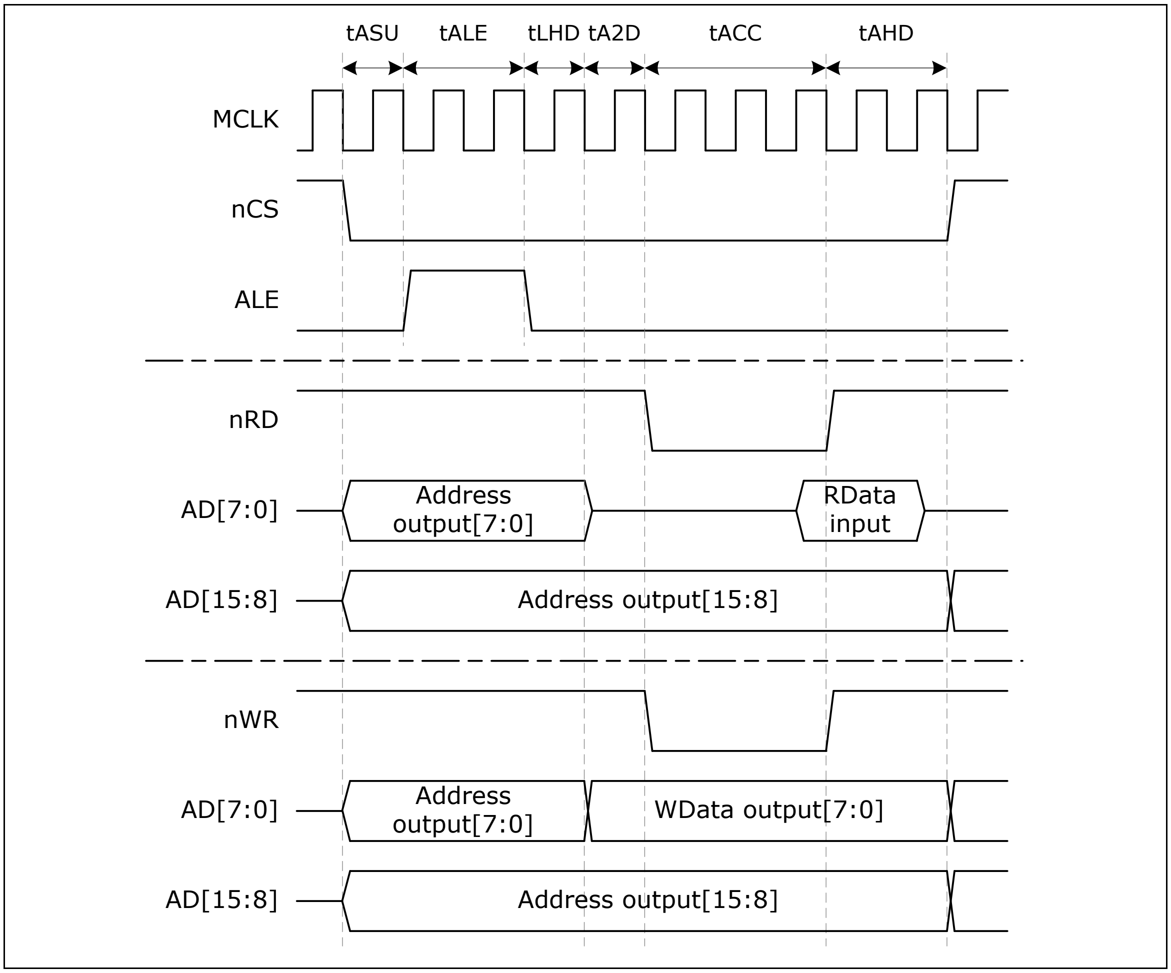


Figure 6-26 Timing Control Waveform for 8-bit Data Width

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.

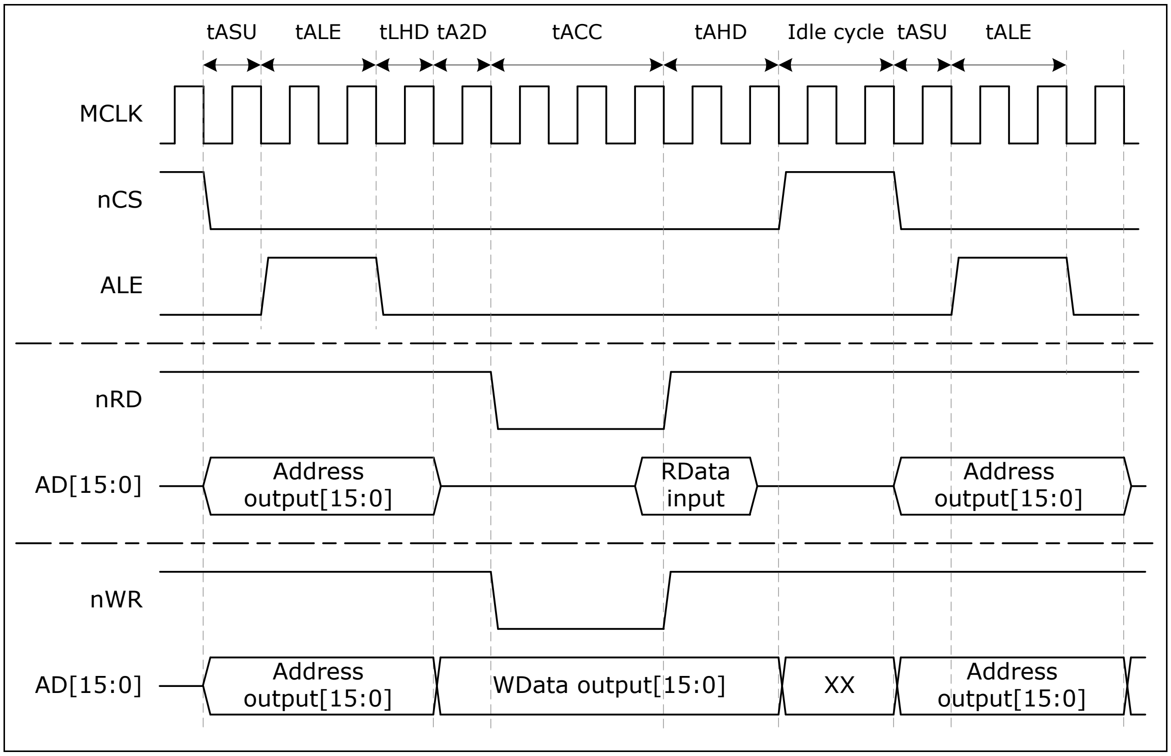


Figure 6-27 Timing Control Waveform for Insert Idle Cycle

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

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**
Reversed					**ExttALE**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reversed					**MCLKDIV**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reversed						**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 shown below. 000 = HCLK/1. 001 = HCLK/2. 010 = HCLK/4. 011 = HCLK/8. 100 = HCLK/16. 101 = HCLK/32. 110 = Default. 111 = 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. 0 = EBI data width is 8-bit. 1 = EBI data width is 16-bit.
[0]	**ExtEN**	**EBI Enable Bit** This bit is the functional enable bit for EBI. 0 = EBI function Disabled.

		1 = EBI function Enabled.

External Bus Interface Timing Control Register (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**
Reversed							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ExtIW2X				**Reversed**	**ExttAHD**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ExttACC					**Reversed**		

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.

6.6 General Purpose I/O (GPIO)

6.6.1 Overview

The NuMicro® NUC100 series has up to 84 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 84 pins are arranged in 6 ports named as GPIOA, GPIOB, GPIOC, GPIOD, GPIOE and GPIOF. The GPIOA/B/C/D/E port has the maximum of 16 pins and GPIOF port has the maximum of 4 pins. Each of the 84 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 Functional Description

6.6.3.1 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.

6.6.3.2 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.

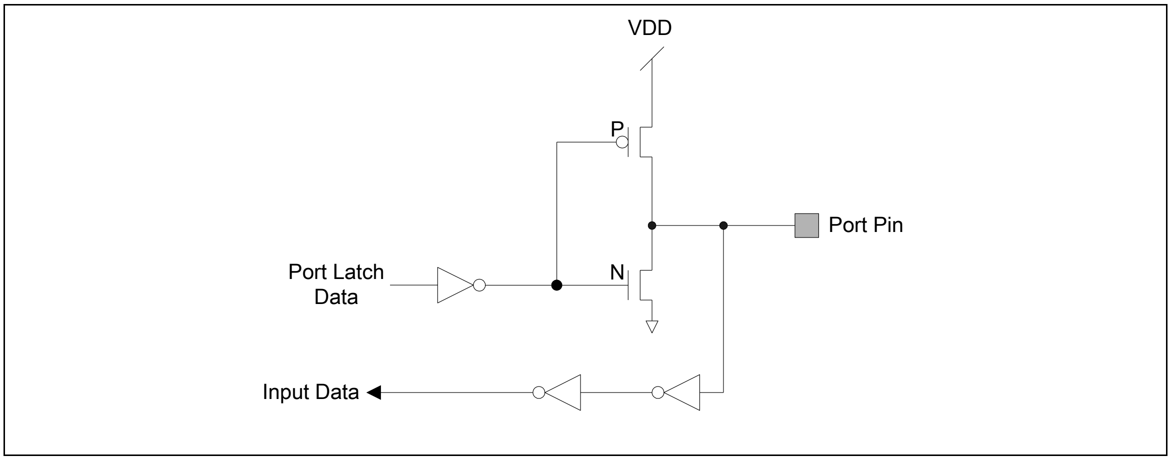


Figure 6-28 Push-Pull Output

6.6.3.3 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.

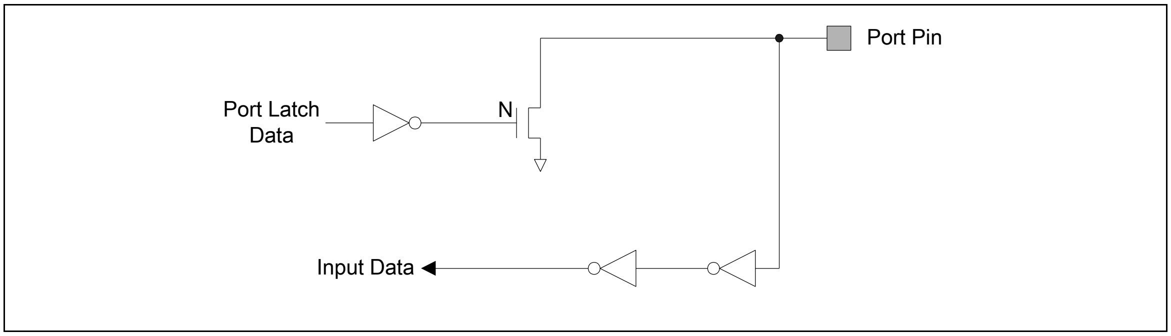


Figure 6-29 Open-Drain Output

6.6.3.4 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.

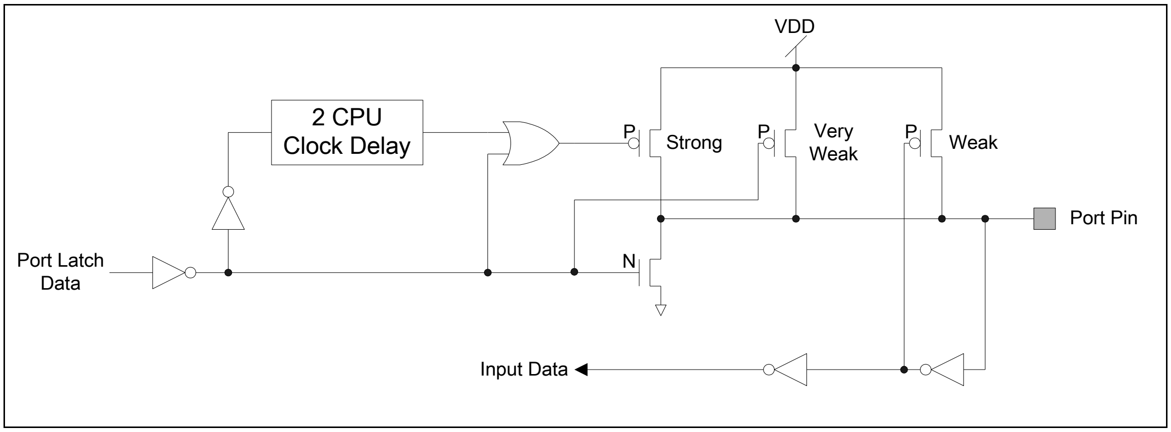


Figure 6-30 Quasi-bidirectional I/O Mode

6.6.3.5 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

1. **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.

6.6.4 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	0xXXXX_XXXX
GPIOA_OFFD	GPIO_BA+0x004	R/W	GPIO Port A Pin Digital Input Path Disable Control	0x0000_0000
GPIOA_DOUT	GPIO_BA+0x008	R/W	GPIO Port A Data Output Value	0x0000_FFFF
GPIOA_DMASK	GPIO_BA+0x00C	R/W	GPIO Port A Data Output Write Mask	0x0000_0000
GPIOA_PIN	GPIO_BA+0x010	R	GPIO Port A Pin Value	0x0000_XXXX
GPIOA_DBEN	GPIO_BA+0x014	R/W	GPIO Port A De-bounce Enable	0x0000_0000
GPIOA_IMD	GPIO_BA+0x018	R/W	GPIO Port A Interrupt Mode Control	0x0000_0000
GPIOA_IEN	GPIO_BA+0x01C	R/W	GPIO Port A Interrupt Enable	0x0000_0000
GPIOA_ISRC	GPIO_BA+0x020	R/W	GPIO Port A Interrupt Source Flag	0x0000_0000
GPIOB_PMD	GPIO_BA+0x040	R/W	GPIO Port B Pin I/O Mode Control	0xXXXX_XXXX
GPIOB_OFFD	GPIO_BA+0x044	R/W	GPIO Port B Pin Digital Input Path Disable Control	0x0000_0000
GPIOB_DOUT	GPIO_BA+0x048	R/W	GPIO Port B Data Output Value	0x0000_FFFF
GPIOB_DMASK	GPIO_BA+0x04C	R/W	GPIO Port B Data Output Write Mask	0x0000_0000
GPIOB_PIN	GPIO_BA+0x050	R	GPIO Port B Pin Value	0x0000_XXXX
GPIOB_DBEN	GPIO_BA+0x054	R/W	GPIO Port B De-bounce Enable	0x0000_0000
GPIOB_IMD	GPIO_BA+0x058	R/W	GPIO Port B Interrupt Mode Control	0x0000_0000
GPIOB_IEN	GPIO_BA+0x05C	R/W	GPIO Port B Interrupt Enable	0x0000_0000
GPIOB_ISRC	GPIO_BA+0x060	R/W	GPIO Port B Interrupt Source Flag	0x0000_0000
GPIOC_PMD	GPIO_BA+0x080	R/W	GPIO Port C Pin I/O Mode Control	0xXXXX_XXXX
GPIOC_OFFD	GPIO_BA+0x084	R/W	GPIO Port C Pin Digital Input Path Disable Control	0x0000_0000
GPIOC_DOUT	GPIO_BA+0x088	R/W	GPIO Port C Data Output Value	0x0000_FFFF
GPIOC_DMASK	GPIO_BA+0x08C	R/W	GPIO Port C Data Output Write Mask	0x0000_0000
GPIOC_PIN	GPIO_BA+0x090	R	GPIO Port C Pin Value	0x0000_XXXX
GPIOC_DBEN	GPIO_BA+0x094	R/W	GPIO Port C De-bounce Enable	0x0000_0000
GPIOC_IMD	GPIO_BA+0x098	R/W	GPIO Port C Interrupt Mode Control	0x0000_0000
GPIOC_IEN	GPIO_BA+0x09C	R/W	GPIO Port C Interrupt Enable	0x0000_0000

Register	Offset	R/W	Description	Reset Value
GPIOC_ISRC	GPIO_BA+0x0A0	R/W	GPIO Port C Interrupt Source Flag	0x0000_0000
GPIOD_PMD	GPIO_BA+0x0C0	R/W	GPIO Port D Pin I/O Mode Control	0xXXXX_XXXX
GPIOD_OFFD	GPIO_BA+0x0C4	R/W	GPIO Port D Pin Digital Input Path Disable Control	0x0000_0000
GPIOD_DOUT	GPIO_BA+0x0C8	R/W	GPIO Port D Data Output Value	0x0000_FFFF
GPIOD_DMASK	GPIO_BA+0x0CC	R/W	GPIO Port D Data Output Write Mask	0x0000_0000
GPIOD_PIN	GPIO_BA+0x0D0	R	GPIO Port D Pin Value	0x0000_XXXX
GPIOD_DBEN	GPIO_BA+0x0D4	R/W	GPIO Port D De-bounce Enable	0x0000_0000
GPIOD_IMD	GPIO_BA+0x0D8	R/W	GPIO Port D Interrupt Mode Control	0x0000_0000
GPIOD_IEN	GPIO_BA+0x0DC	R/W	GPIO Port D Interrupt Enable	0x0000_0000
GPIOD_ISRC	GPIO_BA+0x0E0	R/W	GPIO Port D Interrupt Source Flag	0x0000_0000
GPIOE_PMD	GPIO_BA+0x100	R/W	GPIO Port E Pin I/O Mode Control	0xXXXX_XXXX
GPIOE_OFFD	GPIO_BA+0x104	R/W	GPIO Port E Pin Digital Input Path Disable Control	0x0000_0000
GPIOE_DOUT	GPIO_BA+0x108	R/W	GPIO Port E Data Output Value	0x0000_FFFF
GPIOE_DMASK	GPIO_BA+0x10C	R/W	GPIO Port E Data Output Write Mask	0x0000_0000
GPIOE_PIN	GPIO_BA+0x110	R	GPIO Port E Pin Value	0x0000_XXXX
GPIOE_DBEN	GPIO_BA+0x114	R/W	GPIO Port E De-bounce Enable	0x0000_0000
GPIOE_IMD	GPIO_BA+0x118	R/W	GPIO Port E Interrupt Mode Control	0x0000_0000
GPIOE_IEN	GPIO_BA+0x11C	R/W	GPIO Port E Interrupt Enable	0x0000_0000
GPIOE_ISRC	GPIO_BA+0x120	R/W	GPIO Port E Interrupt Source Flag	0x0000_0000
GPIOF_PMD	GPIO_BA+0x140	R/W	GPIO Port F Pin I/O Mode Control	0x0000_00XX
GPIOF_OFFD	GPIO_BA+0x144	R/W	GPIO Port F Pin Digital Input Path Disable Control	0x0000_0000
GPIOF_DOUT	GPIO_BA+0x148	R/W	GPIO Port F Data Output Value	0x0000_000F
GPIOF_DMASK	GPIO_BA+0x14C	R/W	GPIO Port F Data Output Write Mask	0x0000_0000
GPIOF_PIN	GPIO_BA+0x150	R	GPIO Port F Pin Value	0x0000_000X
GPIOF_DBEN	GPIO_BA+0x154	R/W	GPIO Port F De-bounce Enable	0x0000_0000
GPIOF_IMD	GPIO_BA+0x158	R/W	GPIO Port F Interrupt Mode Control	0x0000_0000
GPIOF_IEN	GPIO_BA+0x15C	R/W	GPIO Port F Interrupt Enable	0x0000_0000
GPIOF_ISRC	GPIO_BA+0x160	R/W	GPIO Port F Interrupt Source Flag	0x0000_0000
DBNCECON	GPIO_BA+0x180	R/W	External Interrupt De-bounce Control	0x0000_0020
PAn_PDIO	GPIO_BA+0x200	R/W	GPIO PA.n Pin Data Input/Output	0x0000_000X

Register	Offset	R/W	Description	Reset Value
n=0,1..15	+ 0x04 * n			
PBn_PDIO **n=0,1..15**	GPIO_BA+0x240 + 0x04 * n	R/W	GPIO PB.n Pin Data Input/Output	0x0000_000X
PCn_PDIO **n=0,1..15**	GPIO_BA+0x280 + 0x04 * n	R/W	GPIO PC.n Pin Data Input/Output	0x0000_000X
PDn_PDIO **n=0,1..15**	GPIO_BA+0x2C0 + 0x04 * n	R/W	GPIO PD.n Pin Data Input/Output	0x0000_000X
PEn_PDIO **n=0,1..15**	GPIO_BA+0x300 + 0x04 * n	R/W	GPIO PE.n Pin Data Input/Output	0x0000_000X
PFn_PDIO **n=0,1..3**	GPIO_BA+0x340 + 0x04 * n	R/W	GPIO PF.n Pin Data Input/Output	0x0000_000X

6.6.5 Register Description

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

Register	Offset	R/W	Description	Reset Value
GPIOA_PMD	GPIO_BA+0x000	R/W	GPIO Port A Pin I/O Mode Control	0xXXXX_XXXX
GPIOB_PMD	GPIO_BA+0x040	R/W	GPIO Port B Pin I/O Mode Control	0xXXXX_XXXX
GPIOC_PMD	GPIO_BA+0x080	R/W	GPIO Port C Pin I/O Mode Control	0xXXXX_XXXX
GPIOD_PMD	GPIO_BA+0x0C0	R/W	GPIO Port D Pin I/O Mode Control	0xXXXX_XXXX
GPIOE_PMD	GPIO_BA+0x100	R/W	GPIO Port E Pin I/O Mode Control	0xXXXX_XXXX
GPIOF_PMD	GPIO_BA+0x140	R/W	GPIO Port F Pin I/O Mode Control	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. **Note:** Max. n = 3 for GPIOF; Max. n = 15 for GPIOA/GPIOB/GPIOC/GPIOD/GPIOE. 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.

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

Register	Offset	R/W	Description	Reset Value
GPIOA_OFFD	GPIO_BA+0x004	R/W	GPIO Port A Pin Digital Input Path Disable Control	0x0000_0000
GPIOB_OFFD	GPIO_BA+0x044	R/W	GPIO Port B Pin Digital Input Path Disable Control	0x0000_0000
GPIOC_OFFD	GPIO_BA+0x084	R/W	GPIO Port C Pin Digital Input Path Disable Control	0x0000_0000
GPIOD_OFFD	GPIO_BA+0x0C4	R/W	GPIO Port D Pin Digital Input Path Disable Control	0x0000_0000
GPIOE_OFFD	GPIO_BA+0x104	R/W	GPIO Port E Pin Digital Input Path Disable Control	0x0000_0000
GPIOF_OFFD	GPIO_BA+0x144	R/W	GPIO Port F Pin Digital Input Path Disable Control	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 Control** 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 creepage 0 = I/O digital input path Enabled. 1 = I/O digital input path Disabled (digital input tied to low). **Note:** Max. n = 3 for GPIOF; Max. n = 15 for GPIOA/GPIOB/GPIOC/GPIOD/GPIOE.
[15:0]	**Reserved**	Reserved.

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

Register	Offset	R/W	Description	Reset Value
GPIOA_DOUT	GPIO_BA+0x008	R/W	GPIO Port A Data Output Value	0x0000_FFFF
GPIOB_DOUT	GPIO_BA+0x048	R/W	GPIO Port B Data Output Value	0x0000_FFFF
GPIOC_DOUT	GPIO_BA+0x088	R/W	GPIO Port C Data Output Value	0x0000_FFFF
GPIOD_DOUT	GPIO_BA+0x0C8	R/W	GPIO Port D Data Output Value	0x0000_FFFF
GPIOE_DOUT	GPIO_BA+0x108	R/W	GPIO Port E Data Output Value	0x0000_FFFF
GPIOF_DOUT	GPIO_BA+0x148	R/W	GPIO Port F Data Output Value	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/D/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/D/E/F] Pin[n] will drive High if the GPIO pin is configured as Push-pull output or Quasi-bidirectional mode. **Note:** Max. n = 3 for GPIOF; Max. n = 15 for GPIOA/GPIOB/GPIOC/GPIOD/GPIOE.

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

Register	Offset	R/W	Description	Reset Value
GPIOA_DMASK	GPIO_BA+0x00C	R/W	GPIO Port A Data Output Write Mask	0x0000_0000
GPIOB_DMASK	GPIO_BA+0x04C	R/W	GPIO Port B Data Output Write Mask	0x0000_0000
GPIOC_DMASK	GPIO_BA+0x08C	R/W	GPIO Port C Data Output Write Mask	0x0000_0000
GPIOD_DMASK	GPIO_BA+0x0CC	R/W	GPIO Port D Data Output Write Mask	0x0000_0000
GPIOE_DMASK	GPIO_BA+0x10C	R/W	GPIO Port E Data Output Write Mask	0x0000_0000
GPIOF_DMASK	GPIO_BA+0x14C	R/W	GPIO Port F Data Output Write Mask	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
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/D/E/F] Data Output Write Mask** These bits are used to protect the corresponding register of GPIOx_DOUT bit[n]. When the DMASK bit[n] 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. **Note:** This function only protects the corresponding GPIOx_DOUT[n] bit, and will not protect the corresponding bit control register (GPIOAx_DOUT, GPIOBx_DOUT, GPIOCx_DOUT, GPIODx_DOUT, GPIOEx_DOUT and GPIOFx_DOUT). **Note:** Max. n = 3 for GPIOF; Max. n = 15 for GPIOA/GPIOB/GPIOC/GPIOD/GPIOE.

GPIO Port [A/B/C/D/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
GPIOD_PIN	GPIO_BA+0x0D0	R	GPIO Port D 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/D/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 **Note:** Max. n = 3 for GPIOF; Max. n = 15 for GPIOA/GPIOB/GPIOC/GPIOD/GPIOE.

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

Register	Offset	R/W	Description	Reset Value
GPIOA_DBEN	GPIO_BA+0x014	R/W	GPIO Port A De-bounce Enable	0x0000_0000
GPIOB_DBEN	GPIO_BA+0x054	R/W	GPIO Port B De-bounce Enable	0x0000_0000
GPIOC_DBEN	GPIO_BA+0x094	R/W	GPIO Port C De-bounce Enable	0x0000_0000
GPIOD_DBEN	GPIO_BA+0x0D4	R/W	GPIO Port D De-bounce Enable	0x0000_0000
GPIOE_DBEN	GPIO_BA+0x114	R/W	GPIO Port E De-bounce Enable	0x0000_0000
GPIOF_DBEN	GPIO_BA+0x154	R/W	GPIO Port F De-bounce Enable	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
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/D/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. **Note:** Max. n = 3 for GPIOF; Max. n = 15 for GPIOA/GPIOB/GPIOC/GPIOD/GPIOE..

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

Register	Offset	R/W	Description	Reset Value
GPIOA_IMD	GPIO_BA+0x018	R/W	GPIO Port A Interrupt Mode Control	0x0000_0000
GPIOB_IMD	GPIO_BA+0x058	R/W	GPIO Port B Interrupt Mode Control	0x0000_0000
GPIOC_IMD	GPIO_BA+0x098	R/W	GPIO Port C Interrupt Mode Control	0x0000_0000
GPIOD_IMD	GPIO_BA+0x0D8	R/W	GPIO Port D Interrupt Mode Control	0x0000_0000
GPIOE_IMD	GPIO_BA+0x118	R/W	GPIO Port E Interrupt Mode Control	0x0000_0000
GPIOF_IMD	GPIO_BA+0x158	R/W	GPIO Port F Interrupt Mode Control	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
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/D/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. **Note:** Max. n = 3 for GPIOF; Max. n = 15 for GPIOA/GPIOB/GPIOC/GPIOD/GPIOE.

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

Register	Offset	R/W	Description	Reset Value
GPIOA_IEN	GPIO_BA+0x01C	R/W	GPIO Port A Interrupt Enable	0x0000_0000
GPIOB_IEN	GPIO_BA+0x05C	R/W	GPIO Port B Interrupt Enable	0x0000_0000
GPIOC_IEN	GPIO_BA+0x09C	R/W	GPIO Port C Interrupt Enable	0x0000_0000
GPIOD_IEN	GPIO_BA+0x0DC	R/W	GPIO Port D Interrupt Enable	0x0000_0000
GPIOE_IEN	GPIO_BA+0x11C	R/W	GPIO Port E Interrupt Enable	0x0000_0000
GPIOF_IEN	GPIO_BA+0x15C	R/W	GPIO Port F Interrupt Enable	0x0000_0000

31	30	29	28	27	26	25	24
IR_EN							
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/D/E/F] Interrupt Enable 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. **Note:** Max. n = 3 for GPIOF; Max. n = 15 for GPIOA/GPIOB/GPIOC/GPIOD/GPIOE.
[n] n = 0,1..15	**IF_EN[n]**	**Port [A/B/C/D/E/F] Interrupt Enable 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. **Note:** Max. n = 3 for GPIOF; Max. n = 15 for GPIOA/GPIOB/GPIOC/GPIOD/GPIOE.

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

Register	Offset	R/W	Description	Reset Value
GPIOA_ISRC	GPIO_BA+0x020	R/W	GPIO Port A Interrupt Source Flag	0x0000_0000
GPIOB_ISRC	GPIO_BA+0x060	R/W	GPIO Port B Interrupt Source Flag	0x0000_0000
GPIOC_ISRC	GPIO_BA+0x0A0	R/W	GPIO Port C Interrupt Source Flag	0x0000_0000
GPIOD_ISRC	GPIO_BA+0x0E0	R/W	GPIO Port D Interrupt Source Flag	0x0000_0000
GPIOE_ISRC	GPIO_BA+0x120	R/W	GPIO Port E Interrupt Source Flag	0x0000_0000
GPIOF_ISRC	GPIO_BA+0x160	R/W	GPIO Port F Interrupt Source Flag	0x0000_0000

31	30	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/D/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. **Note:** Max. n = 3 for GPIOF; Max. n = 15 for GPIOA/GPIOB/GPIOC/GPIOD/GPIOE.

Interrupt De-bounce Cycle Control (DBNCECON)

Register	Offset	R/W	Description	Reset Value
DBNCECON	GPIO_BA+0x180	R/W	External Interrupt De-bounce Control	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 turn off 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** 0000 = Sample interrupt input once per 1 clocks. 0001 = Sample interrupt input once per 2 clocks. 0010 = Sample interrupt input once per 4 clocks. 0011 = Sample interrupt input once per 8 clocks. 0100 = Sample interrupt input once per 16 clocks. 0101 = Sample interrupt input once per 32 clocks. 0110 = Sample interrupt input once per 64 clocks. 0111 = Sample interrupt input once per 128 clocks. 1000 = Sample interrupt input once per 256 clocks. 1001 = Sample interrupt input once per 2*256 clocks. 1010 = Sample interrupt input once per 4*256 clocks. 1011 = Sample interrupt input once per 8*256 clocks. 1100 = Sample interrupt input once per 16*256 clocks. 1101 = Sample interrupt input once per 32*256 clocks. 1110 = Sample interrupt input once per 64*256 clocks. 1111 = Sample interrupt input once per 128*256 clocks.

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

Register	Offset	R/W	Description	Reset Value
PAn_PDIO n=0,1..15	GPIO_BA+0x200 + 0x04 * n	R/W	GPIO PA.n Pin Data Input/Output	0x0000_000X
PBn_PDIO n=0,1..15	GPIO_BA+0x240 + 0x04 * n	R/W	GPIO PB.n Pin Data Input/Output	0x0000_000X
PCn_PDIO n=0,1..15	GPIO_BA+0x280 + 0x04 * n	R/W	GPIO PC.n Pin Data Input/Output	0x0000_000X
PDn_PDIO n=0,1..15	GPIO_BA+0x2C0 + 0x04 * n	R/W	GPIO PD.n Pin Data Input/Output	0x0000_000X
PEn_PDIO n=0,1..15	GPIO_BA+0x300 + 0x04 * n	R/W	GPIO PE.n Pin Data Input/Output	0x0000_000X
PFn_PDIO n=0,1..3	GPIO_BA+0x340 + 0x04 * n	R/W	GPIO PF.n Pin Data Input/Output	0x0000_000X

Note: x = A/B/C/D/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] **Note:** The write operation will not be affected by register GPIOx_DMASK

6.7 PDMA Controller (PDMA)

6.7.1 Overview

The NuMicro® NUC100 series 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 PDMA_CSRx[PDMACEN]. 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

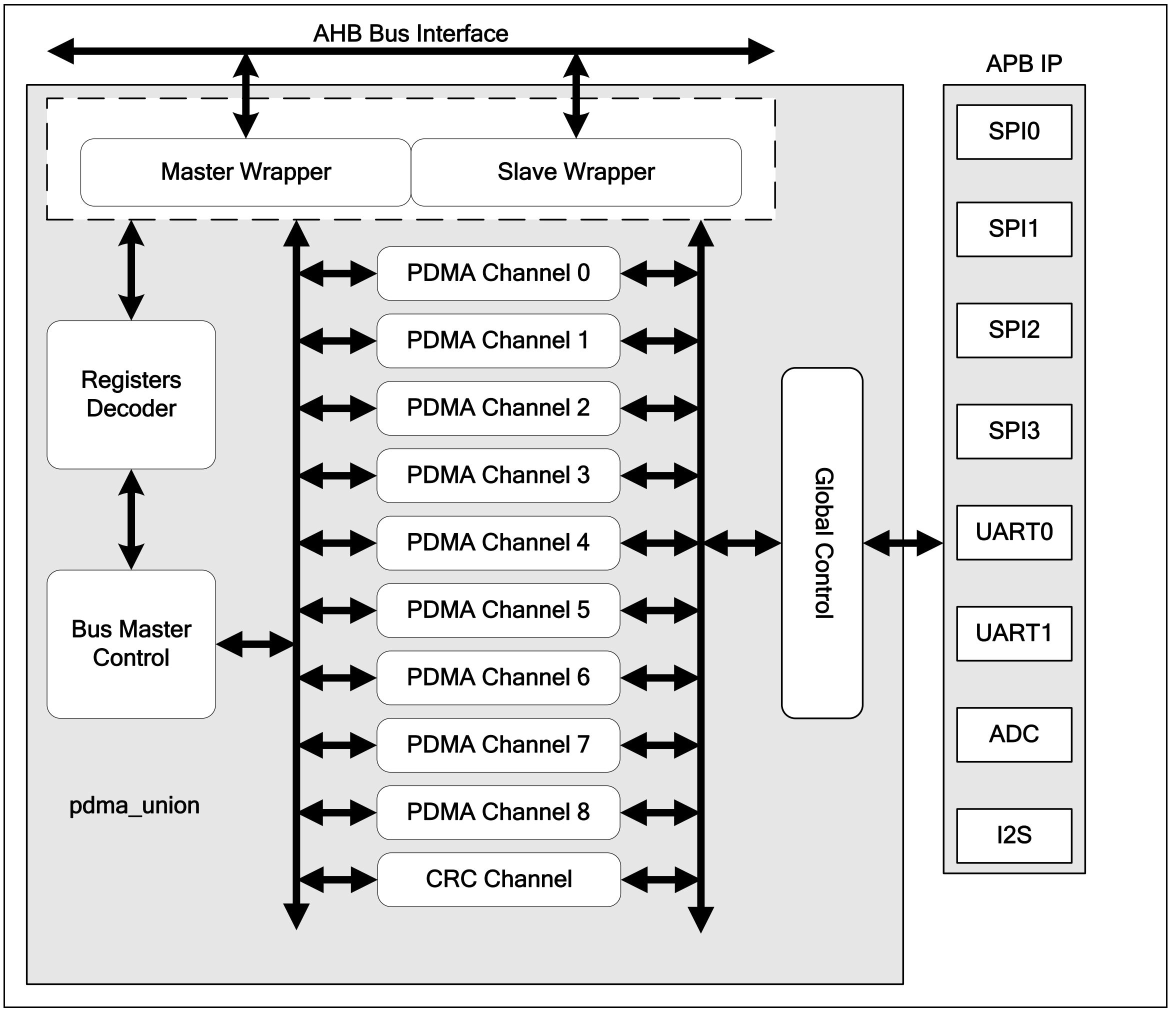


Figure 6-31 DMA Controller Block Diagram

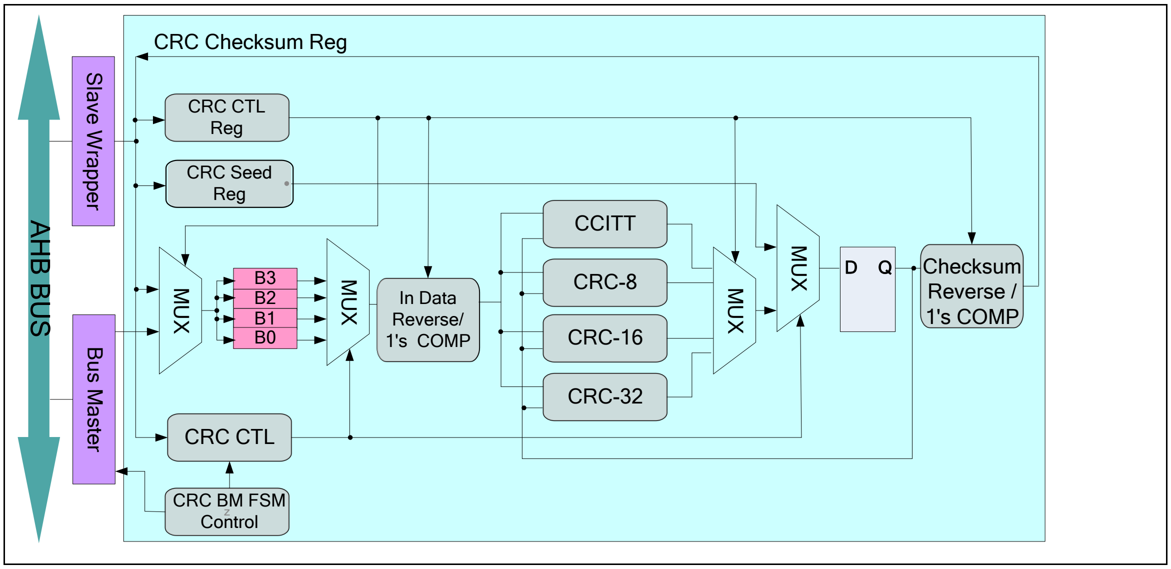


Figure 6-32 CRC Generator Block Diagram

6.7.4 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.

- **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, PDMA_PDSSR1 and PDMA_PDSSR2 registers before starting the related PDMA channel.

Software must enable PDMA channel by setting PDMA_SCRx[PDMACEN] 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 PDMA_CSRx [TRIG_EN]. 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 bit in AHBCLK register.
- Configure the channel service setting by setting PDMA_PDSSR0/ PDMA_PDSSR1/ PDMA_PDSSR2 register.
- Configure PDMA_CSRx register:
  - Enable PDMA channel(PDMACE)
  - Set source/destination address direction(SAD_SEL / DAD_SEL)
  - Configure PDMA mode selection(MODE_SEL)
  - Configure peripherial transfer width selection(APB_TWS).
- 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]. (optional)
- Enable PDMA NVIC by setting NVIC_ISER register bit 26 to “1”. (optional)
- Enable PDMA read/write transfer by setting TRIG_EN bit in PDMA_CSRx register.
- If PDMA block transfer done interrupt is generated, write “1” to PDMA_ISRx[BKLD_IF] by software to clear interrupt flag.
- Enable PDMA read/write transfer by setting the TRIG_EN bit in PDMA_CSRx register 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 PDMA_CSRx [SW_RST] to reset the PDMA channel and set PDMA_CSRx [PDMACEN] and [TRIG_EN] bits field to start again.

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

- **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_MODE[1:0] (CRC_CTL[31:30] CRC Polynomial Mode).

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

Procedure when operating in CPU PIO mode:

- Enable CRC engine by setting CRCCEN bit (CRC_CLT [0] CRC Channel Enable) to 1.
- Set the transfer data format WDATA_RVS (CRC_CTL [24] Write Data Order Reverse), CHECKSUM_RVS (CRC_CTL [25] Checksum Reverse), WDATA_COM (CRC_CTL [26] Write Data 1's Complement), CHECKSUM_COM (CRC_CTL [27] Checksum 1's Complement), initial seed value in CRC_SEED (CRC_SEED [31:0] CRC Seed Register) and select write data length by setting CPU_WDLEN [1:0] (CRC_CTL [29:28] CPU Write Data Length).
- Set the CRC_RST bit (CRC_CTL [1] CRC Engine Reset) 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_WDATA (CRC_WDATA [31:0] CRC Write Data Register) to perform CRC calculation.
- Then, get the CRC checksum results by reading the CRC_CHECKSUM (CRC_CHECKSUM [31:0] CRC Checksum Register).

Procedure when operating in CRC DMA mode:

- Enable CRC engine by setting CRCCEN bit (CRC_CLT [0] CRC Channel Enable) to 1.
- Set the transfer data format WDATA_RVS (CRC_CTL [24] Write Data Order Reverse), CHECKSUM_RVS (CRC_CTL [25] Checksum Reverse), WDATA_COM (CRC_CTL [26] Write Data 1's Complement), CHECKSUM_COM (CRC_CTL [27] Checksum 1's Complement) and initial seed value in CRC_SEED (CRC_SEED [31:0] CRC Seed Register).
- Specify a valid source address (word alignment) and transfer counts by setting CRC_DMASAR (CRC_DMASAR[31:0] CRC DMA Transfer Source Address Register) and CRC_DMABCR (CRC_DMABCR [15:0] CRC DMA Transfer Byte Count Register).
- Set TRIG_EN bit (CRC_CTL [23] CRC DMA Trigger Enable) to 1 to perform CRC calculation.
- Wait CRC DMA transfer and check if CRC DMA transfer is done by the CRC_BLKD_IF bit (CRC DMA Block Transfer Done Interrupt Flag), and then get the CRC checksum results by reading the CRC_CHECKSUM (CRC_CHECKSUM [31:0] CRC Checksum Register).

6.7.5 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	PDMA_CHx_BA+0x00	R/W	PDMA Channel x Control Register	0x0000_0000
PDMA_SARx	PDMA_CHx_BA+0x04	R/W	PDMA Channel x Source Address Register	0x0000_0000
PDMA_DARx	PDMA_CHx_BA+0x08	R/W	PDMA Channel x Destination Address Register	0x0000_0000
PDMA_BCRx	PDMA_CHx_BA+0x0C	R/W	PDMA Channel x Transfer Byte Count Register	0x0000_0000
PDMA_POINTx	PDMA_CHx_BA+0x10	R	PDMA Channel x Internal buffer pointer Register	0xXXXX_0000
PDMA_CSARx	PDMA_CHx_BA+0x14	R	PDMA Channel x Current Source Address Register	0x0000_0000
PDMA_CDARx	PDMA_CHx_BA+0x18	R	PDMA Channel x Current Destination Address Register	0x0000_0000
PDMA_CBCRx	PDMA_CHx_BA+0x1C	R	PDMA Channel x Current Transfer Byte Count Register	0x0000_0000
PDMA_IERx	PDMA_CHx_BA+0x20	R/W	PDMA Channel x Interrupt Enable Register	0x0000_0001
PDMA_ISRx	PDMA_CHx_BA+0x24	R/W	PDMA Channel x Interrupt Status Register	0x0000_0000
PDMA_SBUF0_Cx	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
PDMA_PDSSR2	PDMA_GCR_BA+0x10	R/W	PDMA Service Selection Control Register 2	0x0000_00FF

6.7.6 Register Description

PDMA Channel x Control Register (PDMA_CSRx)

Register	Offset	R/W	Description	Reset Value
PDMA_CSRx	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 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	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	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	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	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	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	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	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 PDMA_CSRx[SW_RST] to "1".

PDMA Channel x Interrupt Enable Control Register (PDMA_IERx)

Register	Offset	R/W	Description	Reset Value
PDMA_IERx	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	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. **Note:** This bit can be cleared by writing '1' to it.
[0]	**TABORT_IF**	**PDMA Read/Write Target Abort Interrupt Flag** 0 = No bus ERROR response received. 1 = Bus ERROR response received. **Note:** This bit can be cleared by writing '1' to it.

Note: PDMA_ISR [TABORT_IF] indicate bus master received ERROR response or not. If bus master received ERROR response, it means that target abort is happened. PDMAC 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 (PDMA_SBUF0_Cx)

Register	Offset	R/W	Description	Reset Value
PDMA_SBUF0_Cx	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_WDTAT 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_WDTAT register. 0 = Bit order reverse for CRC write data in Disabled. 1 = Bit order reverse for CRC write data in Enabled (per byre). **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** Setting this bit to 1 enables CRC operation. When operating in CRC DMA mode (TRIG_EN = 1), if user clears this bit, the DMA operation will be continuous until all CRC DMA operation is done, and the TRIG_EN bit will keep 1until all CRC DMA operation done. But in this case, the CRC_DMAISR [BLKD_IF] flag will inactive, user can read CRC checksum result only if TRIG_EN clears to 0 When operating in CRC DMA mode (TRIG_EN = 1), if user wants to stop the transfer immediately, user can write 1 to CRC_RST bit (CRC_CTL [1] CRC Engine Reset) 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 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 bit (CRCDMAISR [1] CRC DMA Block Transfer Done Interrupt Flag) 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 bit (CRCDMAISR [0] CRC DMA Read/Write Target Abort Interrupt Flag) 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 bit has enabled. 1 = CRC transfer done if TRIG_EN bit has enabled. **Note:** It is cleared by writing 1 to it through software.. (When CRC DMA transfer done, TRIG_EN 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. **Note:** It is cleared by writing 1 to it through software.

Note: The CRC_TABORT_IF bit (CRC_DMAISR [0]) 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 Control** 0 = CRC controller clock Disabled. 1 = CRC controller clock Enabled.
[23:17]	**Reserved**	Reserved.
[16]	**CLK8_EN**	**PDMA Controller Channel 8 Clock Enable Control** 0 = PDMA channel 8 clock Disabled. 1 = PDMA channel 8 clock Enabled.
[15]	**CLK7_EN**	**PDMA Controller Channel 7 Clock Enable Control** 0 = PDMA channel 7 clock Disabled. 1 = PDMA channel 7 clock Enabled.
[14]	**CLK6_EN**	**PDMA Controller Channel 6 Clock Enable Control** 0 = PDMA channel 6 clock Disabled. 1 = PDMA channel 6 clock Enabled.
[13]	**CLK5_EN**	**PDMA Controller Channel 5 Clock Enable Control** 0 = PDMA channel 5 clock Disabled. 1 = PDMA channel 5 clock Enabled.
[12]	**CLK4_EN**	**PDMA Controller Channel 4 Clock Enable Control** 0 = PDMA channel 4 clock Disabled. 1 = PDMA channel 4 clock Enabled.
[11]	**CLK3_EN**	**PDMA Controller Channel 3 Clock Enable Control** 0 = PDMA channel 3 clock Disabled. 1 = PDMA channel 3 clock Enabled.

[10]	**CLK2_EN**	**PDMA Controller Channel 2 Clock Enable Control** 0 = PDMA channel 2 clock Disabled. 1 = PDMA channel 2 clock Enabled.
[9]	**CLK1_EN**	**PDMA Controller Channel 1 Clock Enable Control** 0 = PDMA channel 1 clock Disabled. 1 = PDMA channel 1 clock Enabled.
[8]	**CLK0_EN**	**PDMA Controller Channel 0 Clock Enable Control** 0 = PDMA channel 0 clock Disabled. 1 = PDMA channel 0 clock 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 SPI3_TXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.
[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 SPI3_RXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.
[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 SPI2_TXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.
[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 SPI2_RXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.
[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 SPI1_TXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.
[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 SPI1_RXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.
[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 SPI0_TXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.

[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 SPI0_RXSEL 4'b0000: CH0 4'b0001: CH1 4'b0010: CH2 4'b0011: CH3 4'b0100: CH4 4'b0101: CH5 4'b0110: CH6 4'b0111: CH7 4'b1000: CH8 Others : Reserved **Note:** For example, SPI0_RXSEL = 4'b0110, that means SPI0_RX is connected to PDMA_CH6.

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 ADC_RXSEL. The channel configuration is the same as UART0_RXSEL field. Please refer to the explanation of UART0_RXSEL
[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 UART1_TXSEL. The channel configuration is the same as UART0_RXSEL field. Please refer to the explanation of UART0_RXSEL
[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 UART1_RXSEL. The channel configuration is the same as UART0_RXSEL field. Please refer to the explanation of UART0_RXSEL
[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 UART0_TXSEL. The channel configuration is the same as UART0_RXSEL field. Please refer to the explanation of UART0_RXSEL

[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 UART0_RXSEL 4'b0000: CH0 4'b0001: CH1 4'b0010: CH2 4'b0011: CH3 4'b0100: CH4 4'b0101: CH5 4'b0110: CH6 4'b0111: CH7 4'b1000: CH8 Others : Reserved **Note:** For example, UART0_RXSEL = 4'b0110, which means UART0_RX is connected to PDMA_CH6.

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 (Read Only)** This bit is the interrupt status of PDMA controller.
[30:17]	**Reserved**	Reserved.
[16]	**INTRCRC**	**Interrupt Status of CRC Controller (Read Only)** This bit is the interrupt status of CRC controller
[15:9]	**Reserved**	Reserved.
[8]	**INTR8**	**Interrupt Status of Channel 8 (Read Only)** This bit is the interrupt status of PDMA channel8.
[7]	**INTR7**	**Interrupt Status of Channel 7 (Read Only)** This bit is the interrupt status of PDMA channel7.
[6]	**INTR6**	**Interrupt Status of Channel 6 (Read Only)** This bit is the interrupt status of PDMA channel6.
[5]	**INTR5**	**Interrupt Status of Channel 5 (Read Only)** This bit is the interrupt status of PDMA channel5.
[4]	**INTR4**	**Interrupt Status of Channel 4 (Read Only)** This bit is the interrupt status of PDMA channel4.
[3]	**INTR3**	**Interrupt Status of Channel 3 (Read Only)** This bit is the interrupt status of PDMA channel3.
[2]	**INTR2**	**Interrupt Status of Channel 2 (Read Only)** This bit is the interrupt status of PDMA channel2.
[1]	**INTR1**	**Interrupt Status of Channel 1 (Read Only)** This bit is the interrupt status of PDMA channel1.
[0]	**INTR0**	**Interrupt Status of Channel 0 (Read Only)** This bit is the interrupt status of PDMA channel0.

PDMA Service Selection Control Register 2 (PDMA_PDSSR2)

Register	Offset	R/W	Description	Reset Value
PDMA_PDSSR2	PDMA_GCR_BA+0x10	R/W	PDMA Service Selection Control Register 2	0x0000_00FF

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2S_TXSEL				I2S_RXSEL			

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:4]	**I2S_TXSEL**	**PDMA I^{2}S TX Selection** This field defines which PDMA channel is connected to the on-chip peripheral I^{2}S TX. Software can configure the TX channel setting by I2S_TXSEL. The channel configuration is the same as I2S_RXSEL field. Please refer to the explanation of I2S_RXSEL.
[3:0]	**I2S_RXSEL**	**PDMA I^{2}S RX Selection** This field defines which PDMA channel is connected to the on-chip peripheral I^{2}S RX. Software can change the channel RX setting by I2S_RXSEL 4'b0000: CH0 4'b0001: CH1 4'b0010: CH2 4'b0011: CH3 4'b0100: CH4 4'b0101: CH5 4'b0110: CH6 4'b0111: CH7 4'b1000: CH8 Others : Reserved **Note:** For example: I2S_RXSEL = 4'b0110, that means I2S_RX is connected to PDMA_CH6.

6.8 Timer Controller (TMR)

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, event counting, interval measurement, clock generation, and delay timing. The timer can generate an interrupt signal upon time-out, or provide the current value during operation.

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 one-shot, periodic, toggle and continuous counting operation modes
- 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 pin
- Supports external pin capture function for interval measurement
- Supports external pin capture function for reset timer counter
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated (TIF set to 1)

6.8.3 Block Diagram

Each channel is equipped with an 8-bit prescale counter, a 24-bit up counter, a 24-bit compare register (TCMPR) and an interrupt request signal (TIF). Refer to Figure 6-33 for the timer controller block diagram. There are six options of clock sources for each channel. Figure 6-34 illustrates the clock source control function. Software can program the 8-bit prescale counter to decide the clock period to 24-bit up counter.

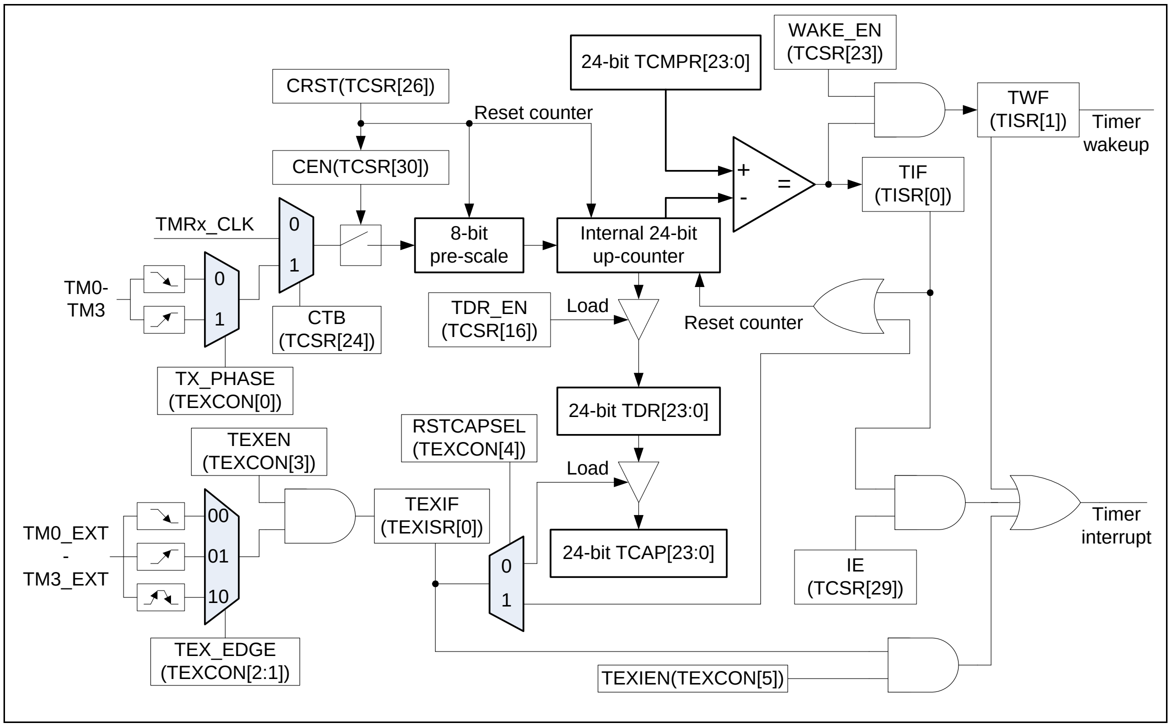


Figure 6-33 Timer Controller Block Diagram

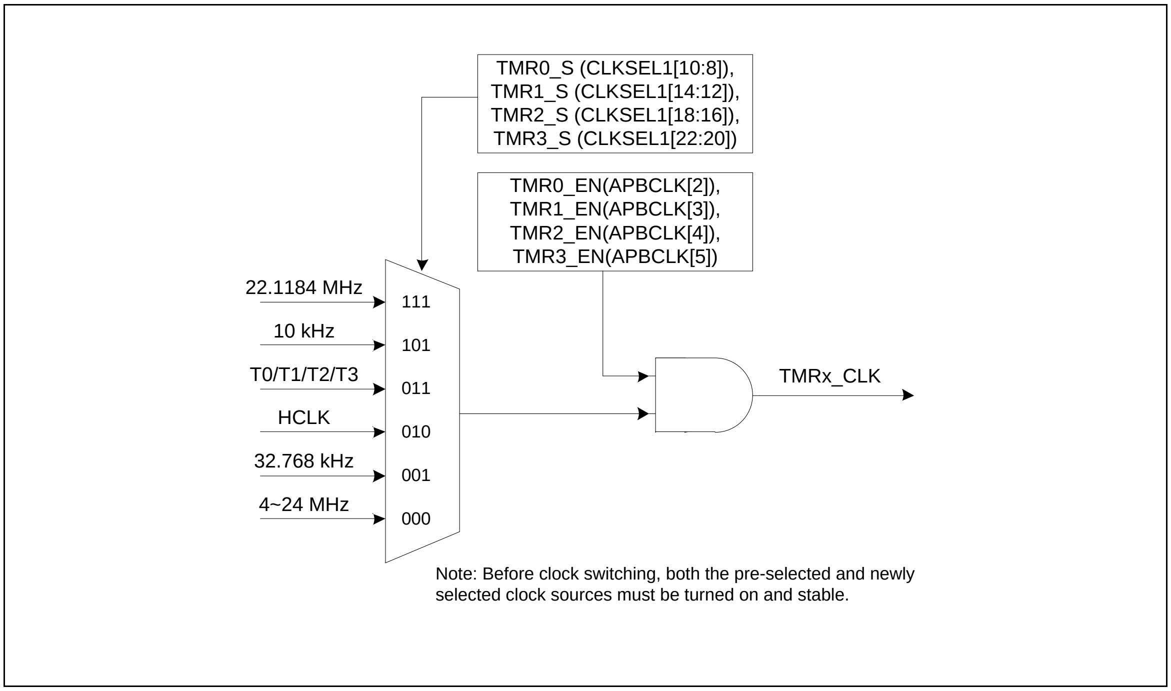


Figure 6-34 Clock Source of Timer Controller

6.8.4 Functional Description

Timer controller provides One-shot, Period, Toggle and Continuous Counting operation modes. The event counting function is also provided to count the events/counts from external pin and external pin capture function for interval measurement or reset timer counter. Each operating function mode is shown as follows:

6.8.4.1 One–shot Mode

If the timer is operated in One-shot mode (TCSR MODE[1:0] is 0x0) and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, the TIF (TISR[0] timer interrupt flag) will be set to 1. If IE (TCSR[29] interrupt enable bit) is set to 1, and TIF (TISR[0] timer interrupt flag) is 1 then the interrupt signal is generated and sent to NVIC to inform CPU for indicating that the timer counting overflow happens. If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

In this operating mode, once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, TIF (TISR[0] timer interrupt flag) will set to 1, timer counting operation stops and the timer counter value (TDR value) goes back to counting initial value then CEN (TCSR[30] timer enable bit) is cleared to 0 by timer controller automatically. That is to say, timer operates timer counting and compares with TCMPR value function only one time after programming the timer compare register (TCMPR) value and CEN (TCSR[30] timer enable bit) is set to 1. So, this operating mode is called One-Shot mode.

6.8.4.2 Periodic Mode

If the timer is operated in Period mode (TCSR MODE[1:0] is 0x1) and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, the TIF (TISR[0] timer interrupt flag) will set to 1. If IE (TCSR[29] interrupt enable bit) is set to 1, and TIF (TISR[0] timer interrupt flag) is 1 then the interrupt signal is generated and sent to NVIC to inform CPU for indicating that the timer counting overflow happens. If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

In this operating mode, once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, TIF (TISR[0] timer interrupt flag) will set to 1, the timer counter value (TDR value) goes back to counting initial value and CEN (TCSR[30] timer enable bit) is kept at 1 (counting enable continuously) and timer counter operates up counting again. If TIF (TISR[0] timer interrupt flag) is cleared by software, once the timer counter value (TDR value) reaches timer compare register (TCMPR) value again, TIF (TISR[0] timer interrupt flag) will set to 1 also. That is to say, timer operates timer counting and compares with TCMPR value function periodically. The timer counting operation does not stop until the CEN (TCSR[30] timer enable bit) is set to 0. The interrupt signal is also generated periodically. So, this operating mode is called Periodic mode.

6.8.4.3 Toggle Mode

If the timer is operated in Toggle mode (TCSR MODE[1:0] is 0x2) and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, the TIF (TISR[0] timer interrupt flag) will set to 1. If IE (TCSR[29] interrupt enable bit) is set to 1, and TIF (TISR[0] timer interrupt flag) is 1 then the interrupt signal is generated and sent to NVIC to inform CPU for indicating that the timer counting overflow happens. If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

In this operating mode, once the timer counter value (TDR value) reaches timer compare register

(TCMPR) value, TIF (TISR[0] timer interrupt flag) will set to 1, toggle out signal (TMx pin) is set to 1, the timer counter value (TDR value) goes back to counting initial value and CEN (TCSR[30] timer enable bit) is kept at 1 (counting enable continuously) and timer counter operates up counting again. If TIF (TISR[0] timer interrupt flag) is cleared by software, once the timer counter value (TDR value) reaches timer compare register (TCMPR) value again, TIF (TISR[0] timer interrupt flag) will set to 1 also and toggle out signal (TMx pin) is set to 0. The timer counting operation does not stop until the CEN (TCSR[30] timer enable bit) is set to 0. Thus, the toggle output signal (TMx pin) is changing back and forth with 50% duty cycle. So, this operating mode is called Toggle mode.

6.8.4.4 Continuous Counting Mode

If the timer is operated in Continuous Counting mode (TCSR MODE[1:0] is 0x3) and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, the TIF (TISR[0] timer interrupt flag) will set to 1. If IE (TCSR[29] interrupt enable bit) is set to 1, and TIF (TISR[0] timer interrupt flag) is 1 then the interrupt signal is generated and sent to NVIC to inform CPU for indicating that the timer counting overflow happens. If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

In this operating mode, once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, TIF (TISR[0] timer interrupt flag) will set to 1 and CEN (TCSR[30] timer enable bit) is kept at 1 (counting enable continuously) and timer counter continuous counting without reload the timer counter value (TDR value) to counting initial value. User can change different timer compare register (TCMPR) value immediately without disabling timer counter and restarting timer counter counting.

For example, the timer compare register (TCMPR) value is set as 80, first. The timer compare register (TCMPR) should be less than 2^{24} and greater than 1. Once the timer counter value (TDR value) reaches 80, TIF (TISR[0] timer interrupt flag) will set to 1 and (TCSR[30] timer enable bit) is kept at 1 (counting enable continuously) and timer counter value (TDR value) will not goes back to 0, it continues counting to 81, 82, 83,⋯ to (2^{24} -1) then 0, 1, 2, 3, ⋯ to 2^{24} -1 again and again. Next, if user programs timer compare register (TCMPR) value as 200 and the TIF (TISR[0] timer interrupt flag) is cleared to 0, then TIF (TISR[0] timer interrupt flag) will set to 1 again when timer counter value (TDR value) reaches 200. At last, user programs timer compare register (TCMPR) value as 500 and clears TIF (TISR[0] timer interrupt flag) to 0, then TIF (TISR[0] timer interrupt flag) will set to 1 again when timer counter value (TDR value) reaches 500. In this mode, the timer counter value (TDR value) is keeping up counting always even if TIF (TISR[0] timer interrupt flag) is 1. So, this operation mode is called as Continuous Counting mode.

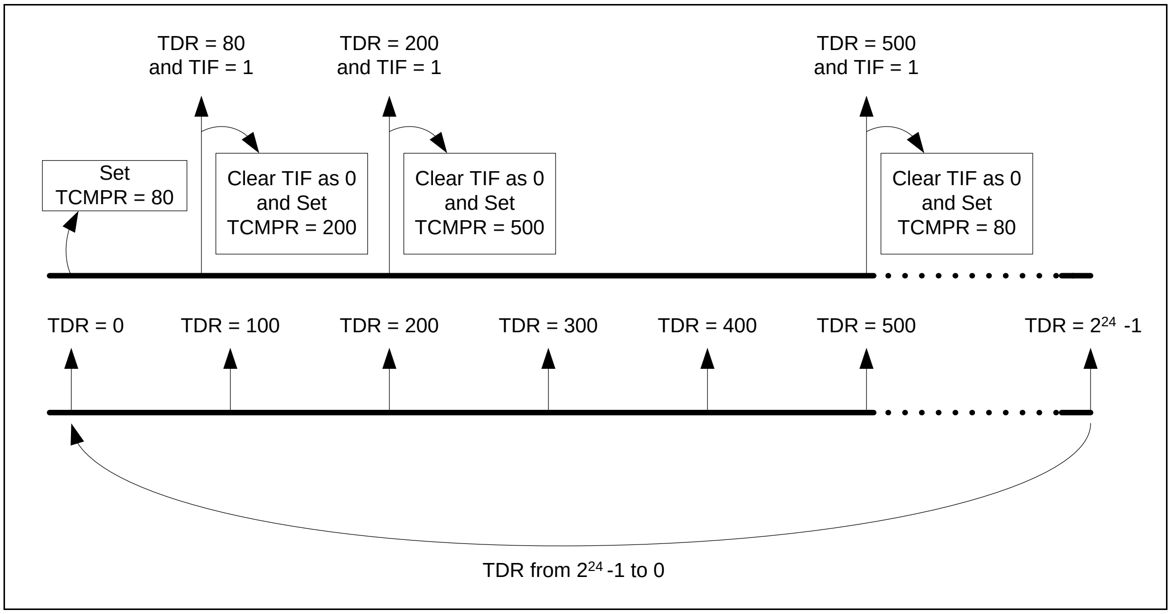


Figure 6-35 Continuous Counting Mode

6.8.4.5 *Event Counting Function*

An application which can count the events/counts from TMx pin (event counting pin) is called as event counting function. In this mode, most of the timer control registers are the same as the timer operating function mode except TCSR MODE[1:0] must be set to 0x2 (toggle out mode will be disabled) and the clock source of timer controller, TMRx_CLK, in Figure 6-34 should be set as HCLK. When status transition on TMx pin, the event counter value (TDR value) will be counted according to TX_PHASE (TEXCON[0] timer external count phase) setting. TCDB (TEXCON[7] timer counter pin de-bounce enable) bit is for enabled or disabled edge detection de-bounce circuit of TMx pin. The max frequency of event counting source on TMx pin should be less than 1/3 HCLK if TCDB (TEXCON[7] timer counter pin de-bounce enable) is 0 or less than 1/8 HCLK if TCDB (TEXCON[7] timer counter pin de-bounce enable) is 1. Otherwise, the event counter value (TDR value) will not be counted normally.

6.8.4.6 *External Pin Capture Function*

In this mode, timer will monitor the transition of TxEX pin (external capture pin) to save the timer counter value (TDR value) to timer capture value (TCAP value) or reset the timer counter value (TDR value) to 0. And the clock source of timer controller, TMRx_CLK, in Figure 6-34 should be set as HCLK.

If RSTCAPSEL (TEXCON[4] timer external reset counter / capture mode select) bit is 0, the transition on TxEX pin is used as timer counter capture function. And when the transition of TxEX pin matches the TEX_EDGE (TEXCON[2:1] timer external pin edge detect) setting, TEXIF (TEXISR[0] timer external interrupt flag) will set to 1 and the timer counter value (TDR value) will be saved into timer capture value (TCAP value).

If RSTCAPSEL (TEXCON[4] timer external reset counter / capture mode select) bit is 1, the transition on TxEX pin is used as timer counter reset function. In this mode, once the transition of TxEX pin matches TEX_EDGE (TEXCON[2:1] timer external pin edge detect) setting, TEXIF (TEXISR[0] timer external interrupt flag) will set to 1 and the timer counter value (TDR value) will be reset to 0.

The max frequency of external capture source on TxEX pin should be less than 1/3 HCLK if

TEXDB (TEXCON[6] timer external capture pin de-bounce enable) is 0 or less than 1/8 HCLK if TEXDB (TEXCON[6] timer external capture pin de-bounce enable) is 1. Otherwise, the transition on TxEX pin will not be detected and TEXIF (TEXISR[0] timer external interrupt flag) will not set to 1 normally.

6.8.5 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

6.8.6 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 (TISR[0] TIF) is generated.
[29]	IE	**Interrupt Enable Bit** 0 = Timer Interrupt Disabled. 1 = Timer Interrupt Enabled. If this bit is enabled, when the timer interrupt flag (TISR[0] TIF) 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. Please refer to 6.8.4.1 for detail description. 01 = The Timer controller is operated in Periodic mode. Please refer to 6.8.4.2 for detail description. 10 = The Timer controller is operated in Toggle-output mode. Please refer to 6.8.4.3 for detail description. 11 = The Timer controller is operated in Continuous Counting mode. Please refer to 6.8.4.4 for detail description.
[26]	**CRST**	**Timer Reset Bit** Setting this bit will reset the 24-bit up counter value (TDR) and also force CEN (TCSR[30] timer enable bit) to 0 if CACT (TCSR[25] timer active status bit) is 1. 0 = No effect. 1 = Reset 8-bit prescale counter, 24-bit up counter value and CEN bit.
[25]	**CACT**	**Timer Active Status Bit (Read Only)** This bit indicates the 24-bit up counter status. 0 = 24-bit up counter is not active. 1 = 24-bit up counter is active.
[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 6.8.4.5 for detail description. 0 = External counter mode Disabled. 1 = External counter mode Enabled.
[23]	**WAKE_EN**	**Wake-up Enable Bit** If this bit is set to 1, while timer interrupt flag (TISR[0] TIF) is generated to 1 and IE (TCSR[29] interrupt enable bit) is enabled, the timer interrupt signal will generate a wake-up trigger event to CPU. 0 = Wake-up trigger event Disabled if timer interrupt signal generated. 1 = Wake-up trigger event Enabled if timer interrupt signal generated.
[22:17]	**Reserved**	Reserved.
[16]	**TDR_EN**	**Data Load Enable Bit** When this bit is set, timer counter value (TDR) will be updated continuously to monitor internal 24-bit up counter value as the counter 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 (TISR[0] timet interrupt 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 software 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 software 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 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. And this bit is set by hardware when the timer counter value) matches the timer compared value (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 will be updated continuously to monitor 24-bit timer 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 TEXEN (TEXCON[3] timer external pin enable) bit is set, RSTCAPSEL (TEXCON[4] timer external reset counter/capture mode select) bit is 0, and a transition on TxEX pin matched the TEX_EDGE (TEXCON[2:1] timer external pin edge detect) setting, TEXIF (TEXISR[0] timer external interrupt flag) will set to 1 and the current timer counter value (TDR value) will be auto-loaded into this TCAP field.

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 Counter Pin De-bounce Enable Bit** 0 = TMx pin de-bounce Disabled. 1 = TMx pin de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of TMx pin is detected with de-bounce circuit.
[6]	**TEXDB**	**Timer External Capture Pin De-bounce Enable Bit** 0 = TxEX pin de-bounce Disabled. 1 = TxEX pin de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of TxEX pin is detected with de-bounce circuit.
[5]	**TEXIEN**	**Timer External Interrupt Enable Bit** 0 = TxEX pin detection Interrupt Disabled. 1 = TxEX pin detection Interrupt Enabled. **Note:** TEXIEN is used to enable timer external interrupt. If TEXIEN enabled, timer will rise an interrupt when TEXIF = 1.
[4]	**RSTCAPSEL**	**Timer External Reset Counter / Capture Mode Selection** 0 = Transition on TxEX pin is using to save the 24-bit timer counter value (TDR value) to timer capture value (TCAP value) if TEXIF (TEXISR[0] timer external interrupt flag) is set to 1. 1 = Transition on TxEX pin is using to reset the 24-bit timer counter.
[3]	**TEXEN**	**Timer External Pin Enable Bit** This bit enables the RSTCAPSEL (TEXCON[4] timer external reset counter/capture mode select) function on the TxEX pin.

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

Timer External Interrupt Status Register (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 Interrupt Flag** This bit indicates the timer external interrupt flag status. When TEXEN (TEXCON[3] timer external pin enable) bit is set, RSTCAPSEL (TEXCON[4] timer external reset counter/capture mode select) bit is 0, and a transition on TxEX pin matched the TEX_EDGE (TEXCON[2:1] timer external pin edge detect) setting, this bit will set to 1 by hardware. **Note:** This bit is cleared by writing 1 to it.

6.9 PWM Generator and Capture Timer (PWM)

6.9.1 Overview

The NuMicro® NUC100 series has 2 sets of PWM groups supporting a total of 4 sets of PWM generators that can be configured as 8 independent PWM outputs, PWM0~PWM7, or as 4 complementary PWM pairs, (PWM0, PWM1), (PWM2, PWM3), (PWM4, PWM5) and (PWM6, PWM7) with 4 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 4 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 PCR.DZEN01 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), (PWM4, PWM5) and (PWM6, PWM7) are controlled by PWM2, PWM4 and PWM6 timers and Dead-zone generator 2, 4 and 6, respectively. Refer to Figure 6-36 and Figure 6-43 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 CCR0.CRL_IE0[1] (Rising latch Interrupt enable) and CCR0.CFL_IE0[2]] (Falling latch Interrupt enable) to decide the condition of interrupt occur. Capture channel 1 has the same feature by setting CCR0.CRL_IE1[17] and CCR0.CFL_IE1[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 8 PWM channels or 4 complementary PWM paired channels
- Each PWM group has two PWM generators with each PWM generator supporting one 8-bit prescaler, two clock divider, 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 PWM
- Edge-aligned type or Center-aligned type option

6.9.2.2 Capture Function:

- Timing control logic shared with PWM Generators
- Supports 8 Capture input channels shared with 8 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 6-36 to Figure 6-43 illustrate 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).

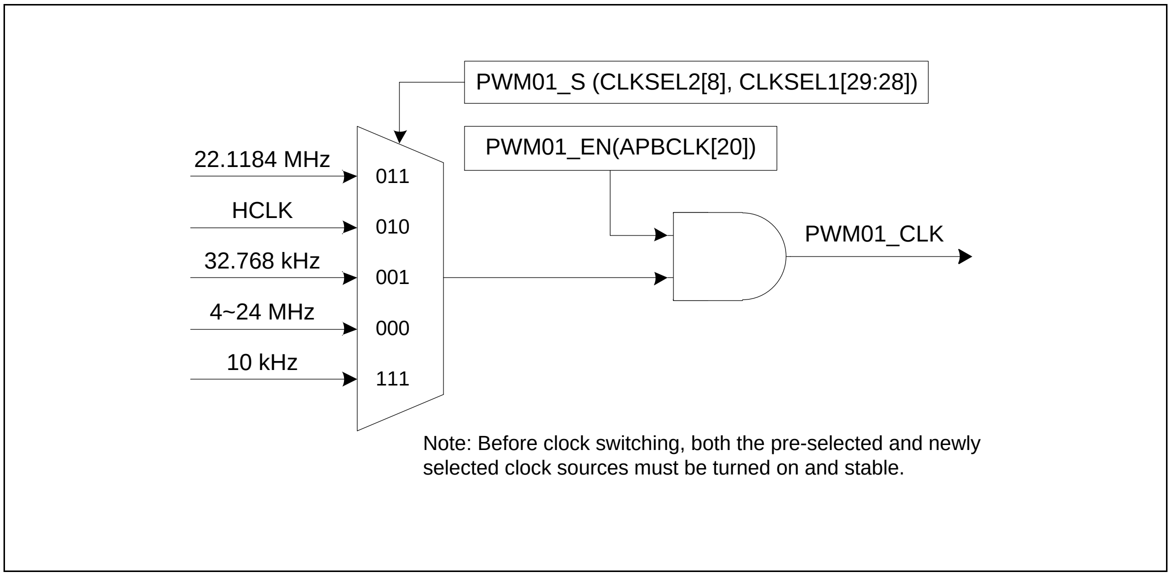


Figure 6-36 PWM Generator 0 Clock Source Control

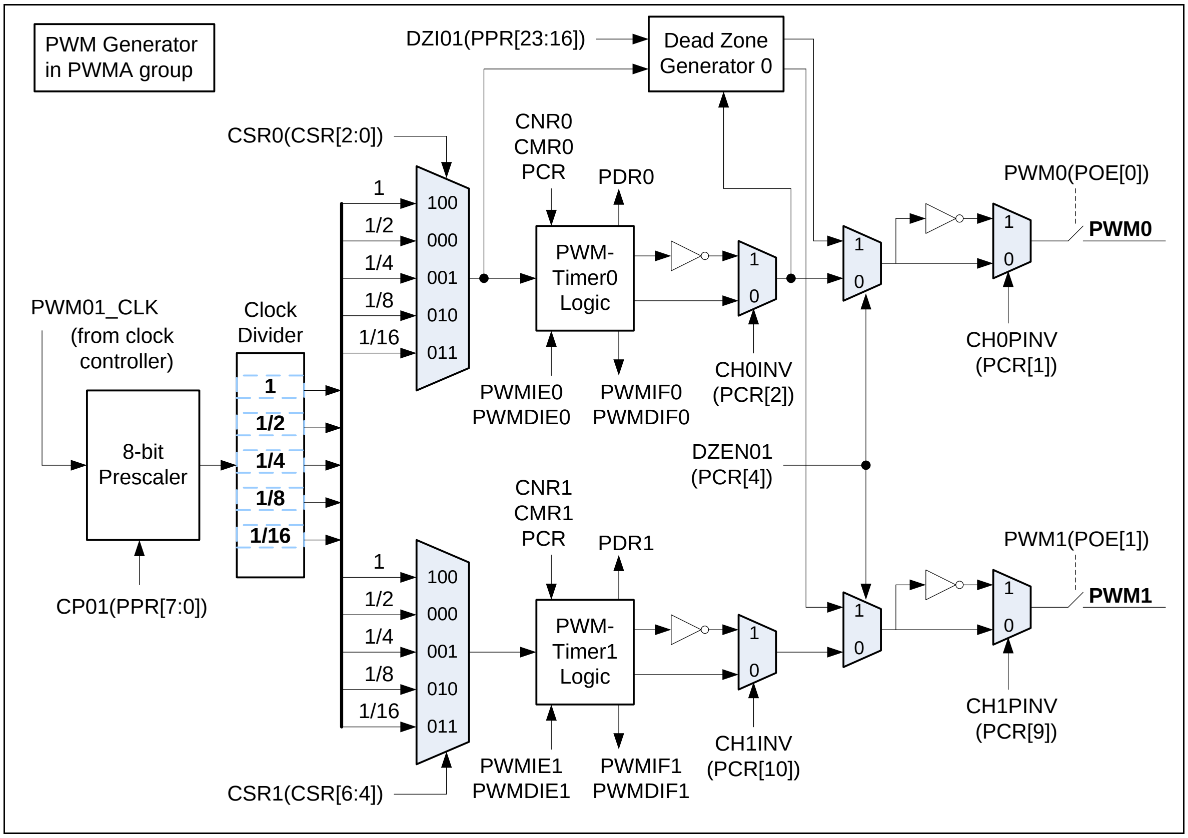


Figure 6-37 PWM Generator 0 Architecture Diagram

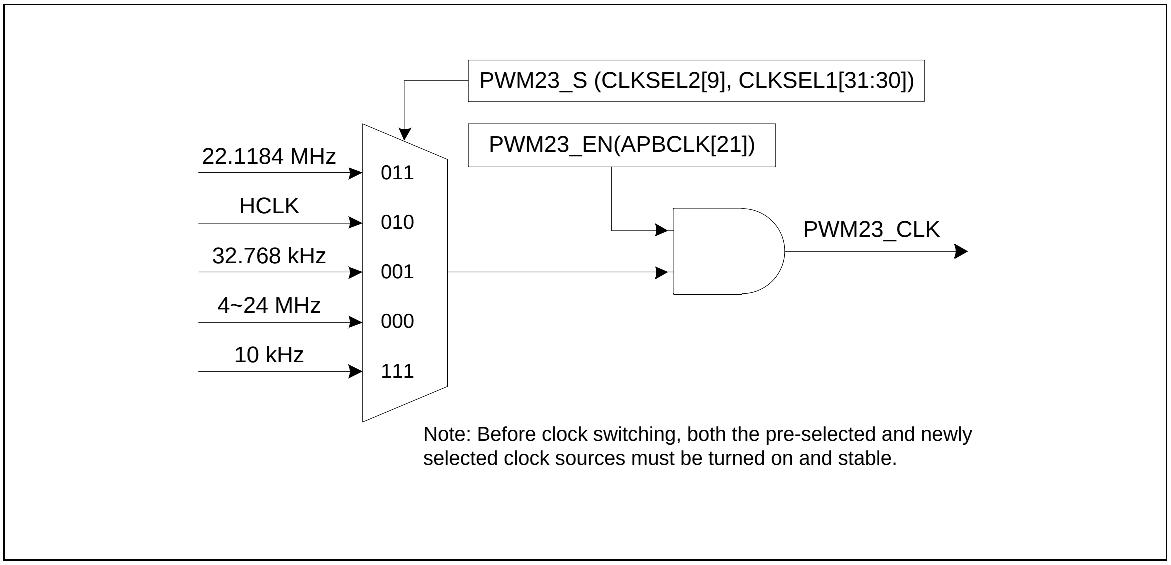


Figure 6-38 PWM Generator 2 Clock Source Control

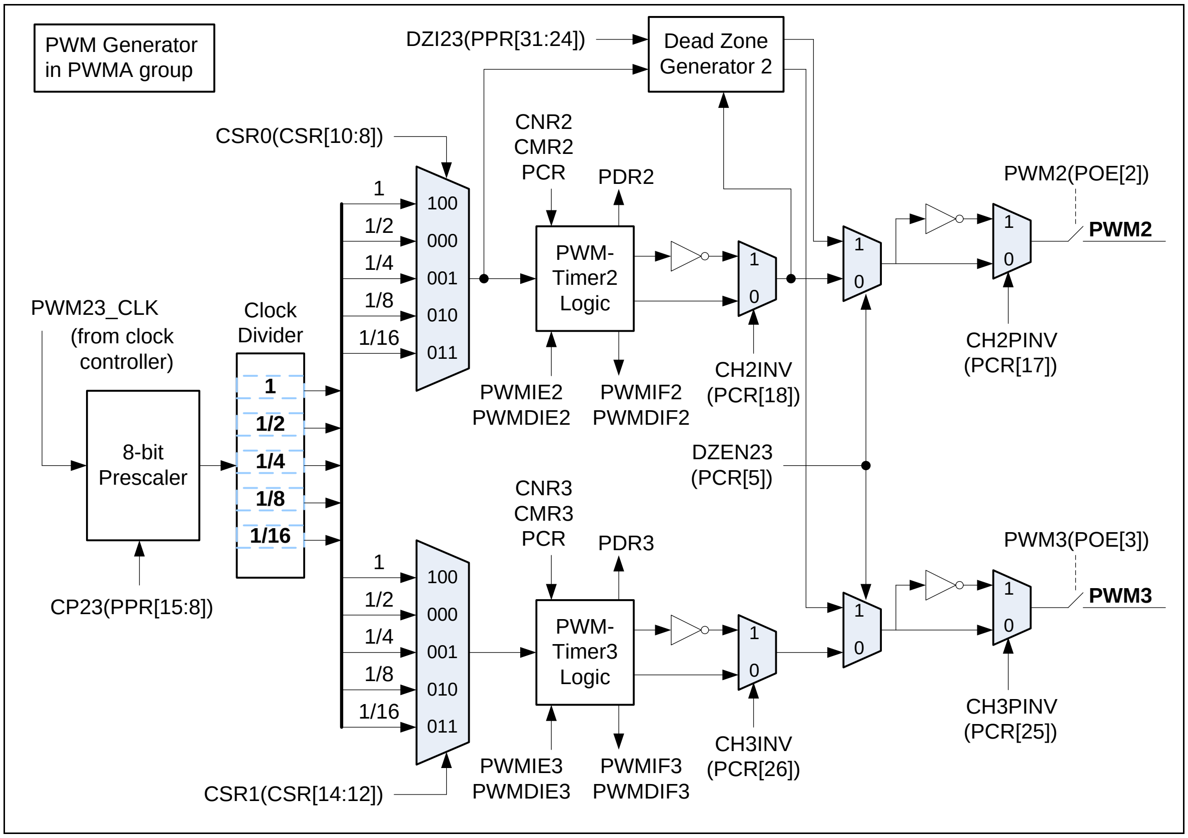


Figure 6-39 PWM Generator 2 Architecture Diagram

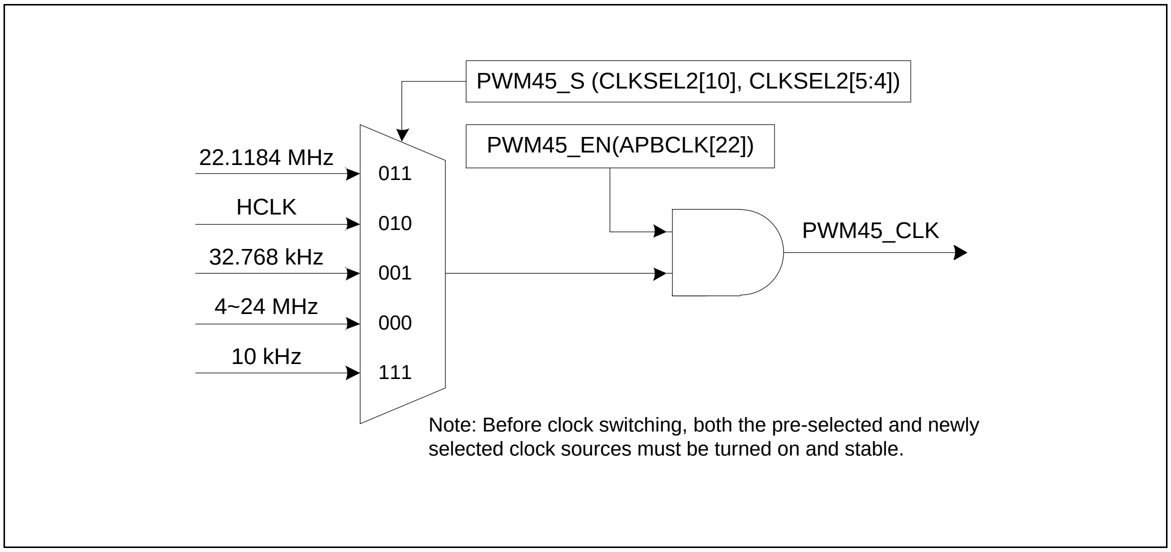


Figure 6-40 PWM Generator 4 Clock Source Control

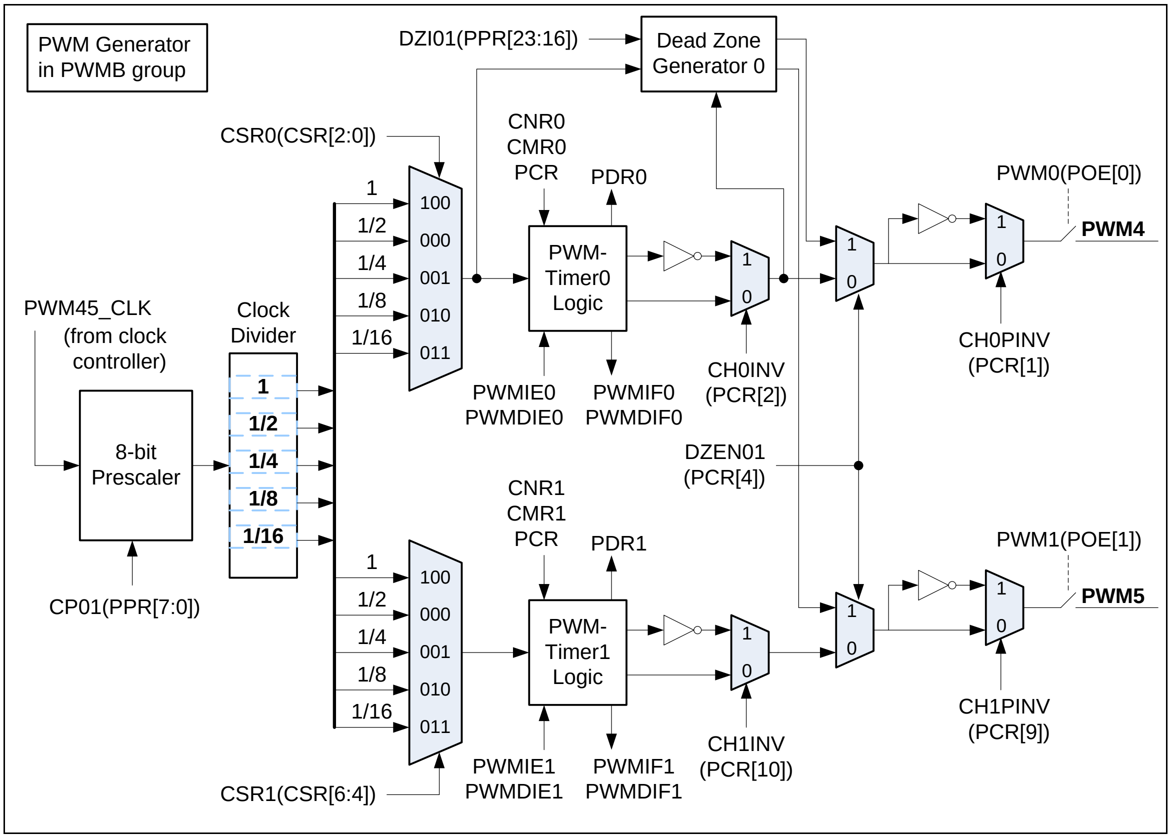


Figure 6-41 PWM Generator 4 Architecture Diagram

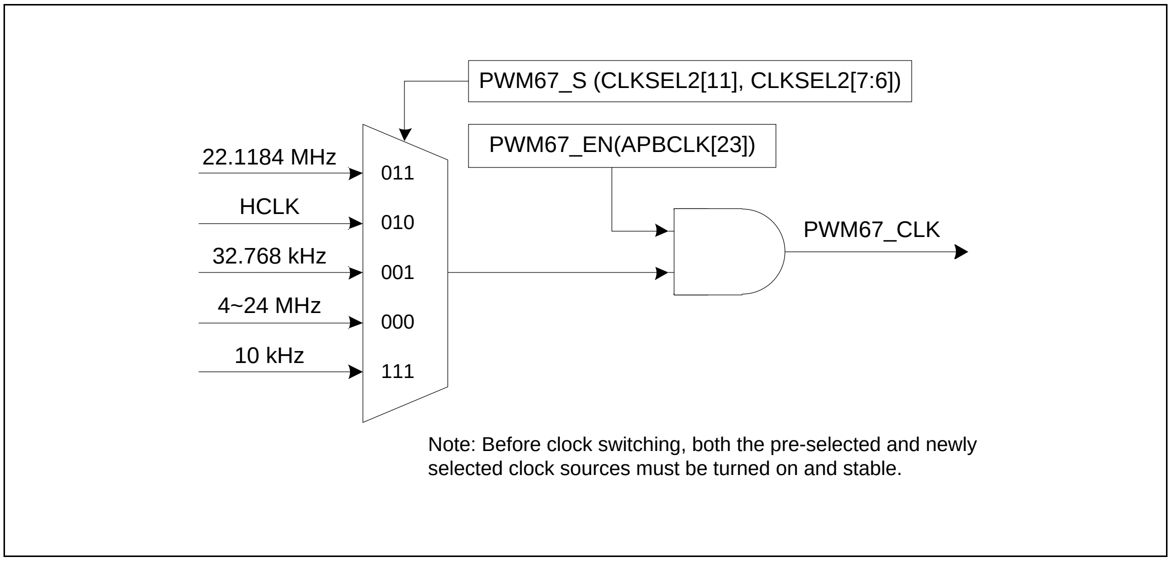


Figure 6-42 PWM Generator 6 Clock Source Control

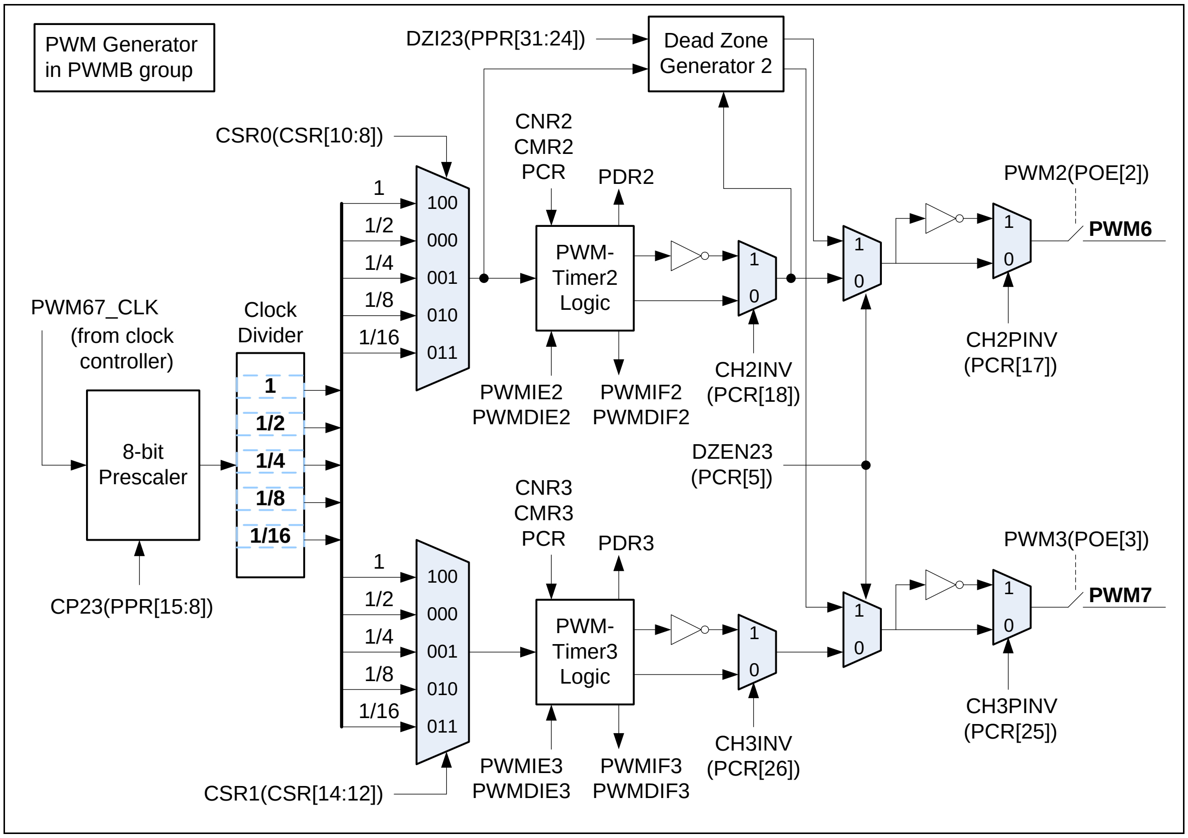


Figure 6-43 PWM Generator 6 Architecture Diagram

6.9.4 Functional Description

6.9.4.1 PWM-Timer Operation

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

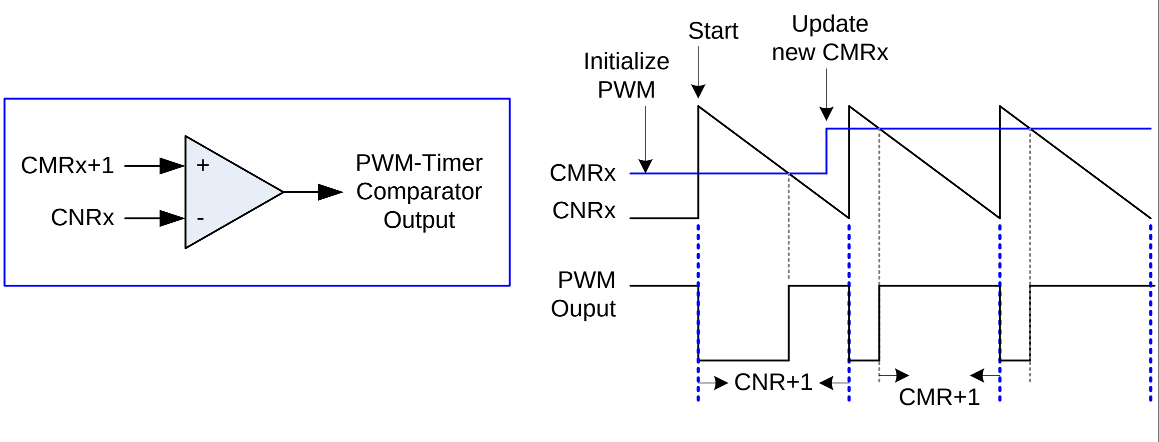
Edge-aligned PWM (down-counter)

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(PIER.n=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 6-45. The pulse width modulation follows the formula below and the legend of PWM-Timer Comparator is shown as Figure 6-44. 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, 45 or 67, 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.



Note: x= 0~3.

Figure 6-44 Legend of Internal Comparator Output of PWM-Timer

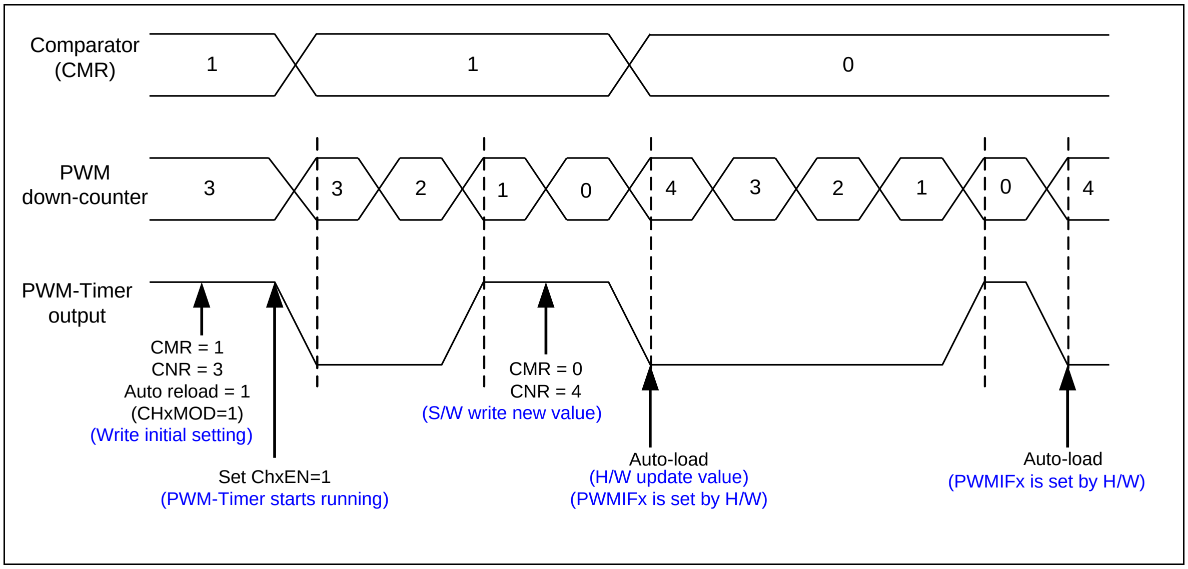


Figure 6-45 PWM-Timer Operation Timing

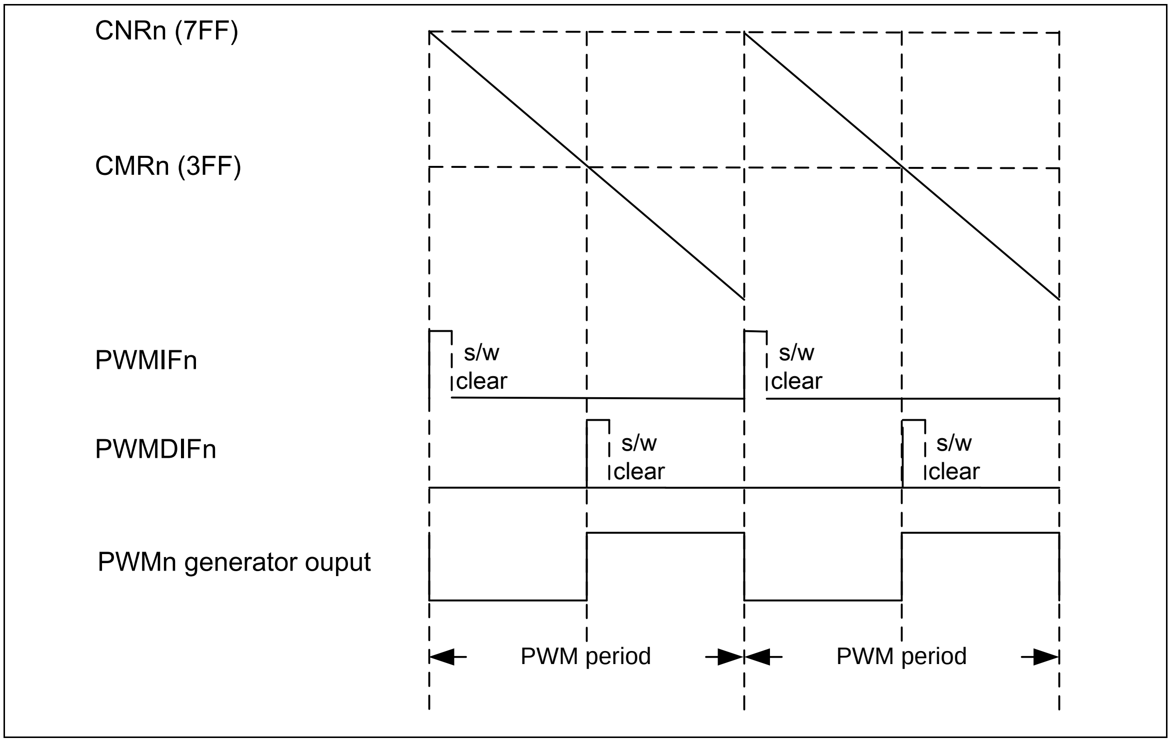


Figure 6-46 PWM Edge-aligned Interrupt Generate Timing Waveform

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, 45 or 67, 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

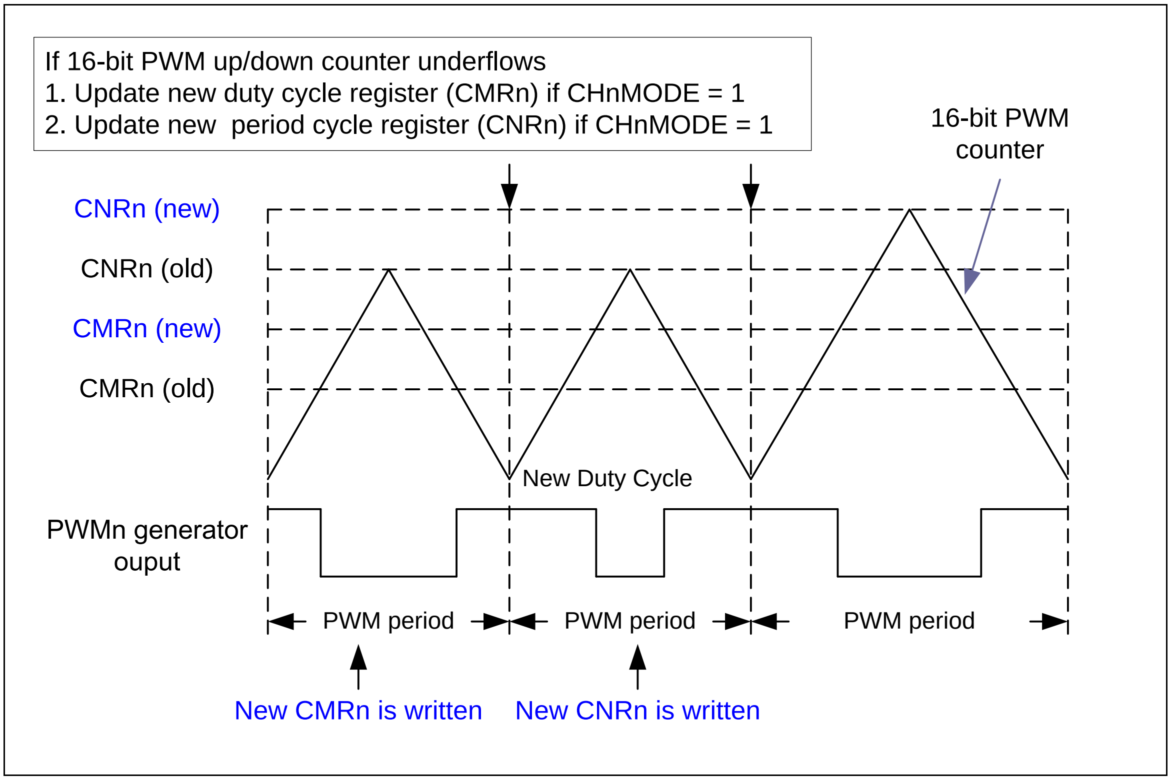


Figure 6-47 Center-aligned Type Output Waveform

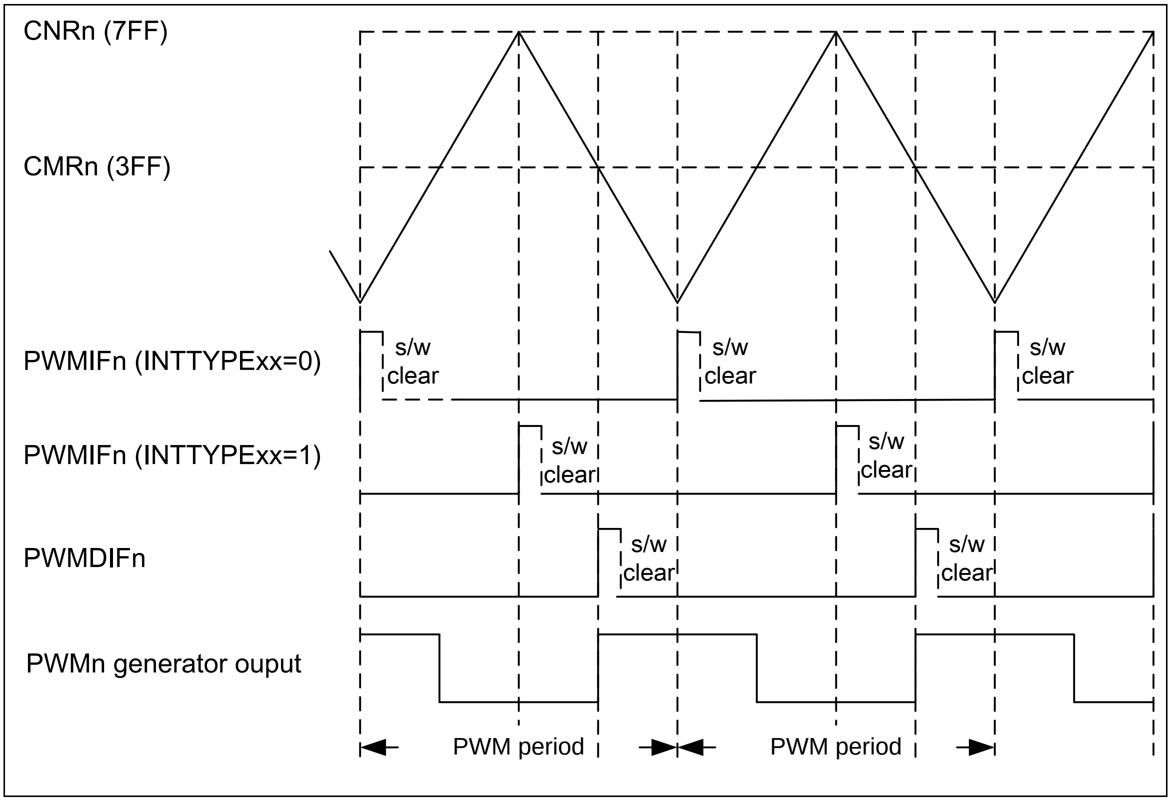


Figure 6-48 PWM Center-aligned Interrupt Generate Timing Waveform

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

PWM Timers have double buffering function 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 bit is set to 0, and operate at Auto-reload mode if CH0MOD bit is set to 1. It is recommend that switch PWM0 operating mode before set CH0EN 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 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 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~PWM7 performs the same function as PWM0.

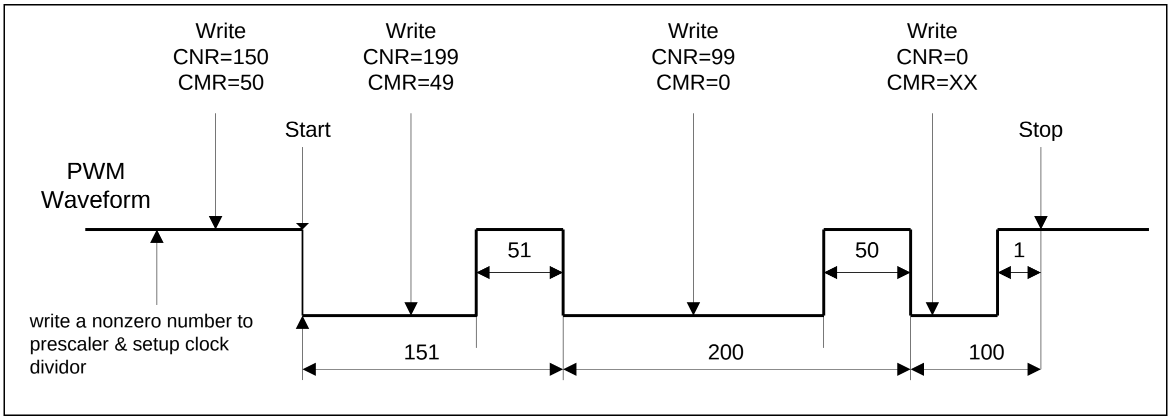


Figure 6-49 PWM Double Buffering Illustration

6.9.4.3 *Modulate Duty Ratio*

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

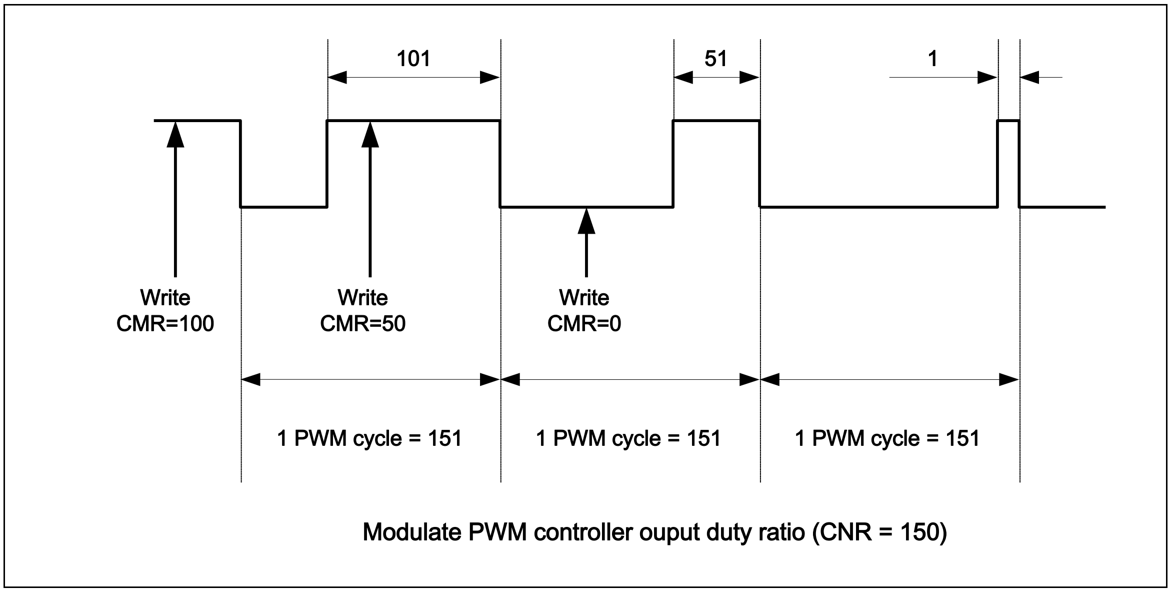


Figure 6-50 PWM Controller Output Duty Ratio

6.9.4.4 *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 PPRx.DZI to determine the Dead-zone interval.

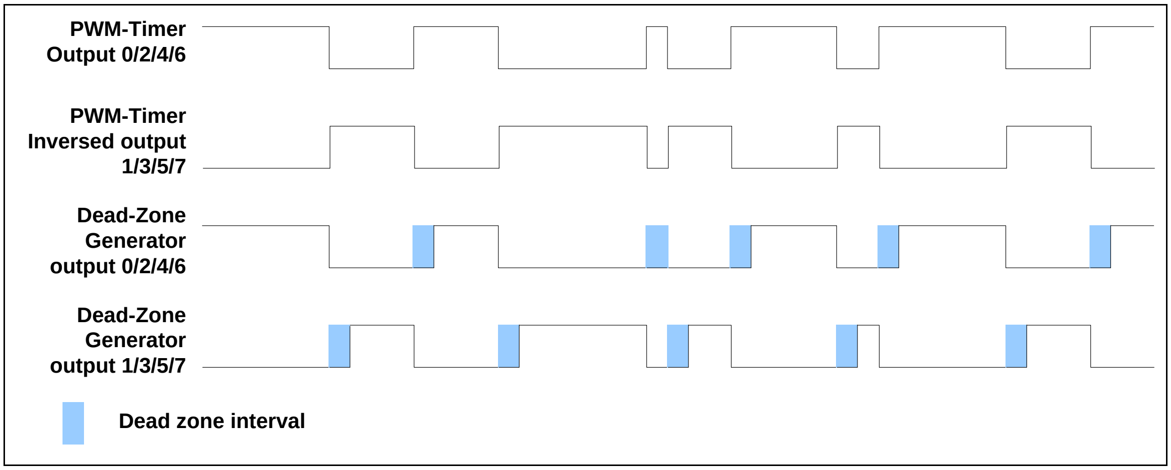


Figure 6-51 Paired-PWM Output with Dead-zone Generation Operation

6.9.4.5 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 to "1".

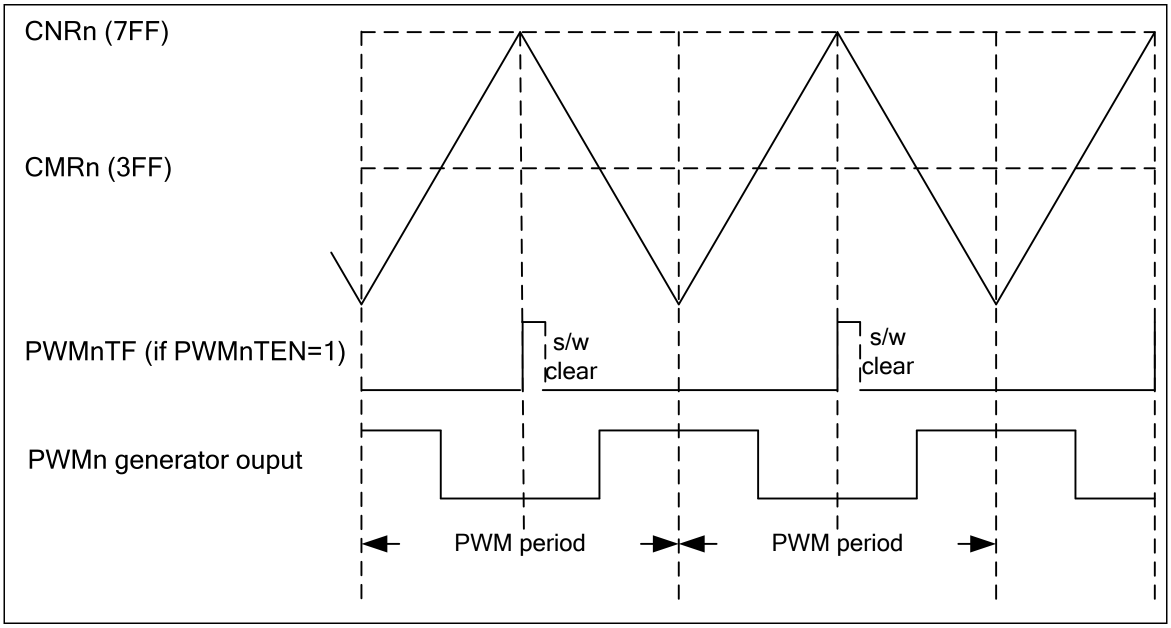


Figure 6-52 PWM trigger ADC to conversion in Center-aligned type Timing Waveform

6.9.4.6 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 CRLRx when input channel has a rising transition and latches PWM-counter to CFLRx when input channel has a falling transition. Capture channel 0 interrupt is programmable by setting CCR0[1] (Rising latch Interrupt enable) and CCR0[2] (Falling latch Interrupt enable) to decide the condition of interrupt occur. Capture channel 1 has the same feature by setting CCR0[17] and CCR0[18], and etc. Whenever the Capture controller issues a capture interrupt, the corresponding PWM counter will be reloaded with CNRx 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.

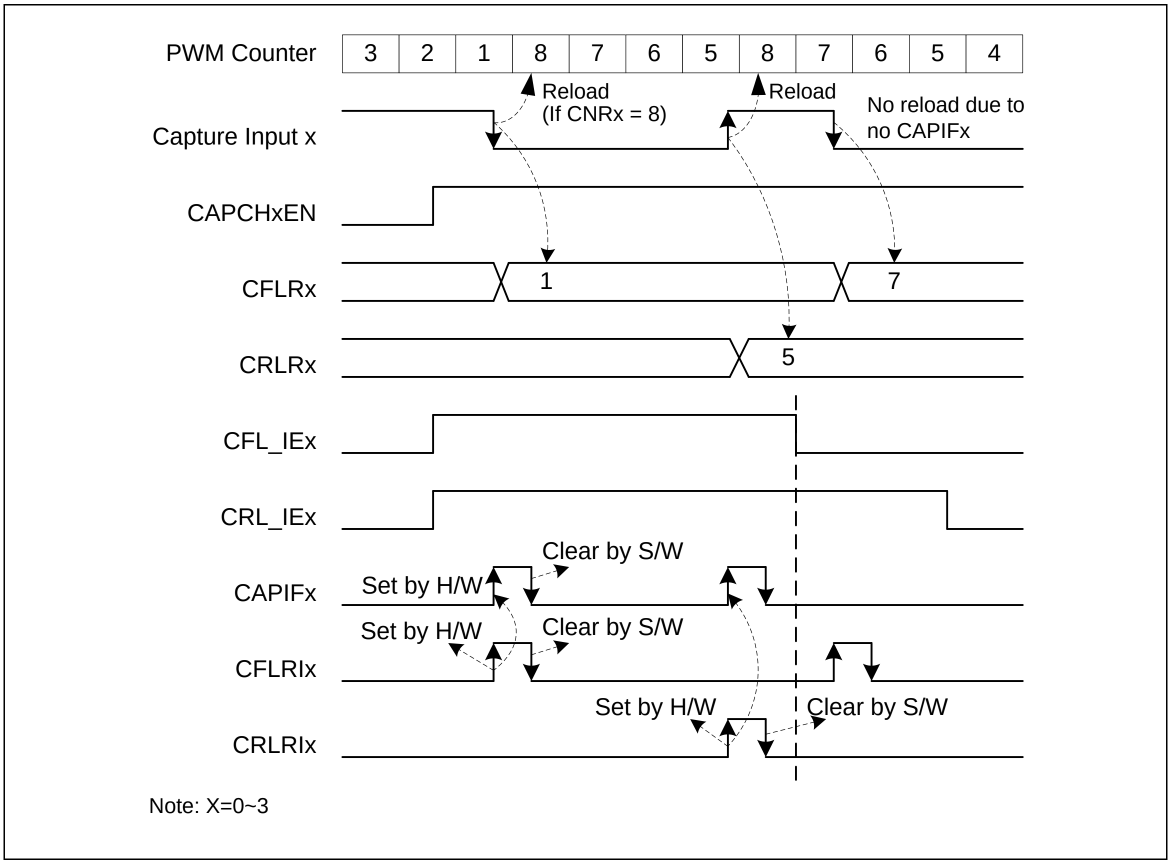


Figure 6-53 Capture Operation Timing

In this case, the CNR is 8:

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

6.9.4.7 PWM-Timer Interrupt Architecture

There are eight PWM interrupts, PWM0_INT~PWM7_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 6-54 and Figure 5-48 demonstrates the architecture of PWM-Timer interrupts.

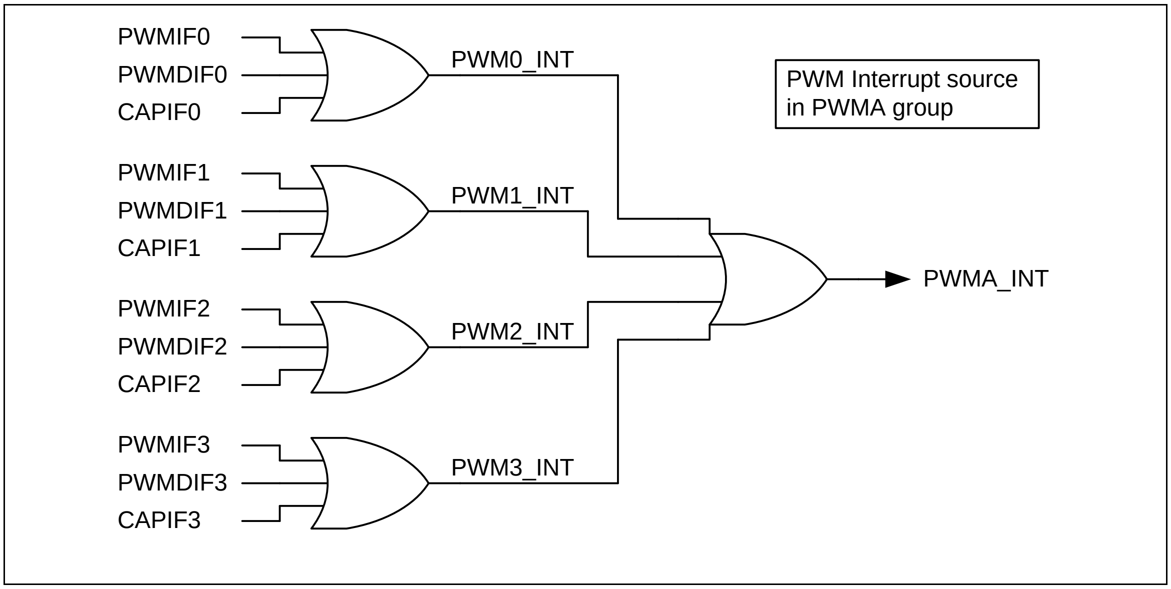


Figure 6-54 PWM Group A PWM-Timer Interrupt Architecture Diagram

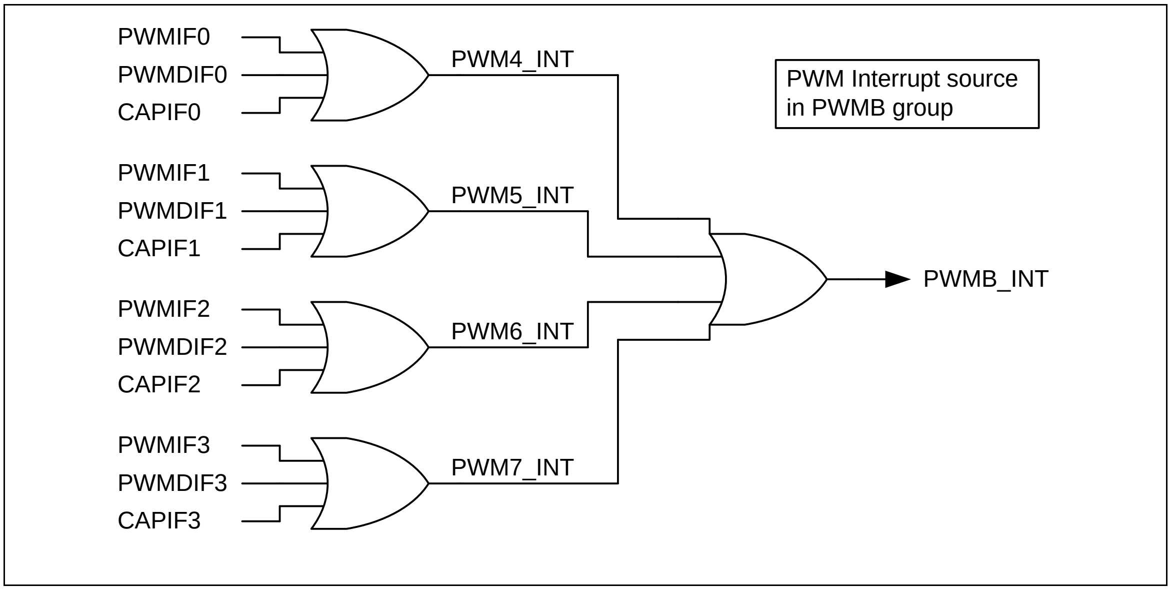


Figure 6-55 PWM Group B PWM-Timer Interrupt Architecture Diagram

6.9.4.8 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 CHxEN = 1 in PCR)

6.9.4.9 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

6.9.4.10 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.

6.9.4.11 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 0, disable PWM-Timer (CHxEN in PCR). ***(Recommended)***

Method 2:

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

Method 3:

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

The reason why method 3 is not recommended is that disable CHxEN 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

6.9.4.12 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 CHxEN = 1 in PCR)
10. Setup corresponding GPIO pins as capture function (disable POE and enable CAPENR) for the corresponding PWM channel.

6.9.5 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 PWMB_BA+0x24	R/W	PWM Counter Register 2	0x0000_0000
CMR2	PWMA_BA+0x28 PWMB_BA+0x28	R/W	PWM Comparator Register 2	0x0000_0000
PDR2	PWMA_BA+0x2C PWMB_BA+0x2C	R	PWM Data Register 2	0x0000_0000
CNR3	PWMA_BA+0x30 PWMB_BA+0x30	R/W	PWM Counter Register 3	0x0000_0000
CMR3	PWMA_BA+0x34 PWMB_BA+0x34	R/W	PWM Comparator Register 3	0x0000_0000
PDR3	PWMA_BA+0x38 PWMB_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 PWMB_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 PWMB_BA+0x68	R	PWM Capture Rising Latch Register (Channel 2)	0x0000_0000
CFLR2	PWMA_BA+0x6C PWMB_BA+0x6C	R	PWM Capture Falling Latch Register (Channel 2)	0x0000_0000
CRLR3	PWMA_BA+0x70 PWMB_BA+0x70	R	PWM Capture Rising Latch Register (Channel 3)	0x0000_0000
CFLR3	PWMA_BA+0x74 PWMB_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 PWMB_BA+0x90	R	PWM2 Synchronous Busy Status Register	0x0000_0000
SYNCBUSY3	PWMA_BA+0x94 PWMB_BA+0x94	R	PWM3 Synchronous Busy Status Register	0x0000_0000

6.9.6 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, PWM6 and PWM7 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 23 or 67, depends on selected PWM channel.).
[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 and PWM-timer 6 / 7 for Group B)** 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 and PWM Timer 7 for Group B)** Select clock source divider for PWM timer 3. 100 = Input Clock Divided by 1. 011 = Input Clock Divided by 16. 010 = Input Clock Divided by 8. 001 = Input Clock Divided by 4. 000 = Input Clock Divided by 2.
[11]	**Reserved**	Reserved.
[10:8]	**CSR2**	**PWM Timer 2 Clock Source Divider Selection (PWM Timer 2 for Group a and PWM Timer 6 for Group B)** 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 Bit (PWM2 and PWM3 Pair for PWM Group A, PWM6 and PWM7 Pair for PWM Group B)** 0 = Edge-aligned type. 1 = Center-aligned type.
[30]	PWM01TYPE	**PWM01 Aligned Type Selection Bit (PWM0 and PWM1 Pair for PWM Group A, PWM4 and PWM5 Pair for PWM Group B)** 0 = Edge-aligned type. 1 = Center-aligned type.
[29:28]	Reserved	Reserved.
[27]	CH3MOD	**PWM-timer 3 Auto-reload/One-shot Mode (PWM Timer 3 for Group a and PWM Timer 7 for Group B)** 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 Bit (PWM Timer 3 for Group a and PWM Timer 7 for Group B)** 0 = Inverter Disabled. 1 = Inverter Enabled.
[25]	CH3PINV	**PWM-timer 3 Output Polar Inverse Enable Bit (PWM Timer 3 for Group a and PWM Timer 7 for Group B)** 0 = PWM3 output polar inverse Disable. 1 = PWM3 output polar inverse Enable.
[24]	CH3EN	**PWM-timer 3 Enable Bit (PWM Timer 3 for Group a and PWM Timer 7 for Group B)** 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 and PWM Timer 6 for Group B)** 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 Bit (PWM Timer 2 for Group a and PWM Timer 6 for Group B)** 0 = Inverter Disabled. 1 = Inverter Enabled.
[17]	**CH2PINV**	**PWM-timer 2 Output Polar Inverse Enable Bit (PWM Timer 2 for Group a and PWM Timer 6 for Group B)** 0 = PWM2 output polar inverse Disabled. 1 = PWM2 output polar inverse Enabled.
[16]	**CH2EN**	**PWM-timer 2 Enable Bit (PWM Timer 2 for Group a and PWM Timer 6 for Group B)** 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 Bit (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 Bit (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 Bit (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 Bit (PWM2 and PWM3 Pair for PWM Group A, PWM6 and PWM7 Pair for PWM Group B)** 0 = Dead-zone 2 generator Disabled. 1 = Dead-zone 2 generator Enabled. **Note:** When Dead-zone generator is enabled, the pair of PWM2 and PWM3 becomes a complementary pair for PWM group A and the pair of PWM6 and PWM7 becomes a complementary pair for PWM group B.
[4]	**DZEN01**	**Dead-zone 0 Generator Enable Bit (PWM0 and PWM1 Pair for PWM Group A, PWM4 and PWM5 Pair for PWM Group B)** 0 = Dead-zone 0 generator Disabled. 1 = Dead-zone 0 generator 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 Bit (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 Bit (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 Bit (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 PWMB_BA+0x24	R/W	PWM Counter Register 2	0x0000_0000
CNR3	PWMA_BA+0x30 PWMB_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, 45 or 67, 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 PWMB_BA+0x28	R/W	PWM Comparator Register 2	0x0000_0000
CMR3	PWMA_BA+0x34 PWMB_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, 45 or 67, 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 PWMB_BA+0x2C	R	PWM Data Register 2	0x0000_0000
PDR3	PWMA_BA+0x38 PWMB_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, PWM6 and PWM7 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.
[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.
[15:12]	**Reserved**	Reserved.
[11]	**PWMDIE3**	**PWM Channel 3 Duty Interrupt Enable Bit** 0 = PWM channel 3 duty interrupt Disabled. 1 = PWM channel 3 duty interrupt Enabled.
[10]	**PWMDIE2**	**PWM Channel 2 Duty Interrupt Enable Bit** 0 = PWM channel 2 duty interrupt Disabled. 1 = PWM channel 2 duty interrupt Enabled.
[9]	**PWMDIE1**	**PWM Channel 1 Duty Interrupt Enable Bit** 0 = PWM channel 1 duty interrupt Disabled. 1 = PWM channel 1 duty interrupt Enabled.
[8]	**PWMDIE0**	**PWM Channel 0 Duty Interrupt Enable Bit** 0 = PWM channel 0 duty interrupt Disabled. 1 = PWM channel 0 duty interrupt Enabled.
[7:4]	**Reserved**	Reserved.

[3]	**PWMIE3**	**PWM Channel 3 Period Interrupt Enable Bit** 0 = PWM channel 3 interrupt Disabled. 1 = PWM channel 3 interrupt Enabled.
[2]	**PWMIE2**	**PWM Channel 2 Period Interrupt Enable Bit** 0 = PWM channel 2 interrupt Disabled. 1 = PWM channel 2 interrupt Enabled.
[1]	**PWMIE1**	**PWM Channel 1 Period Interrupt Enable Bit** 0 = PWM channel 1 interrupt Disabled. 1 = PWM channel 1 interrupt Enabled.
[0]	**PWMIE0**	**PWM Channel 0 Period Interrupt Enable Bit** 0 = PWM channel 0 interrupt Disabled. 1 = PWM channel 0 interrupt 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. **Note:** 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. **Note:** 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.
[21]	**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). **Note:** This bit can be cleared by writing '1' to it.
[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 Bit** 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. **Note:** 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 Bit** 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. **Note:** 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). **Note:** This bit can be cleared by writing ‘1’ to it.
[3]	**CAPCH0EN**	**Channel 0 Capture Function Enable Bit** 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** 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 PWMB_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 Bit** 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. **Note:** 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 Bit** 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. **Note:** 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). **Note:** 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 Bit** 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. **Note:** 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 Bit** 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. **Note:** 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). **Note:** 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 PWMB_BA+0x68	R	PWM Capture Rising Latch Register (Channel 2)	0x0000_0000
CRLR3	PWMA_BA+0x70 PWMB_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							
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 PWMB_BA+0x6C	R	PWM Capture Falling Latch Register (Channel 2)	0x0000_0000
CFLR3	PWMA_BA+0x74 PWMB_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. **Note:** This bit can be cleared by writing '1' to it.
[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. **Note:** This bit can be cleared by writing '1' to it.
[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. **Note:** This bit can be cleared by writing '1' to it.
[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. **Note:** This bit can be cleared by writing '1' to it.

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 PWMB_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 PWMB_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}) and the time-out interval 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 selectable Watchdog Timer reset delay period, it includes (1024+2)、(128+2) 、(16+2) or (1+2) WDT_CLK reset delay period.
- Supports 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 when WDT clock source is selected to 10 kHz low speed oscillator.

6.10.3 Block Diagram

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

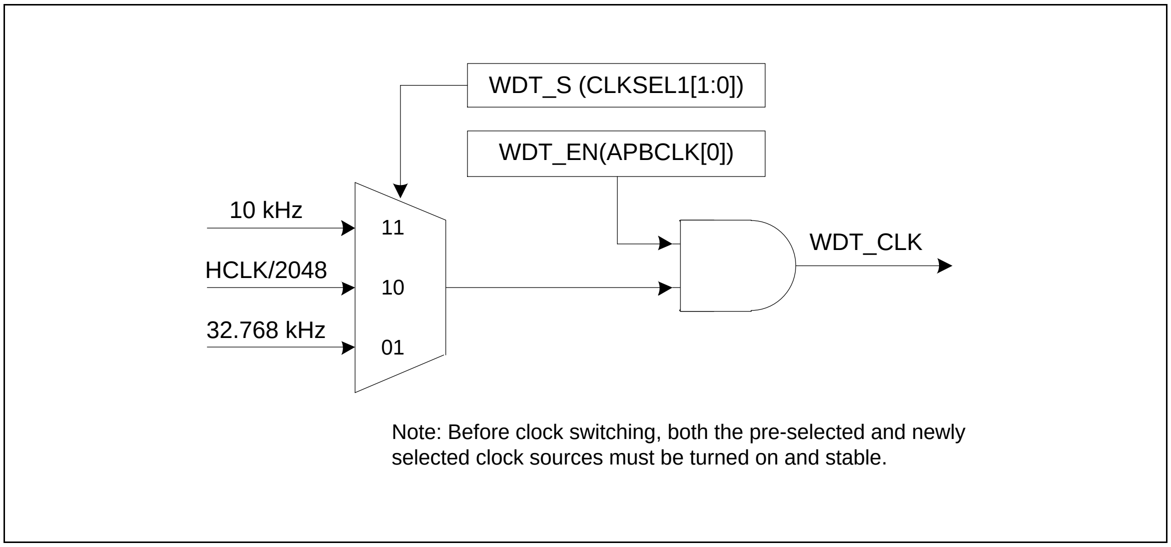


Figure 6-56 Watchdog Timer Clock Control

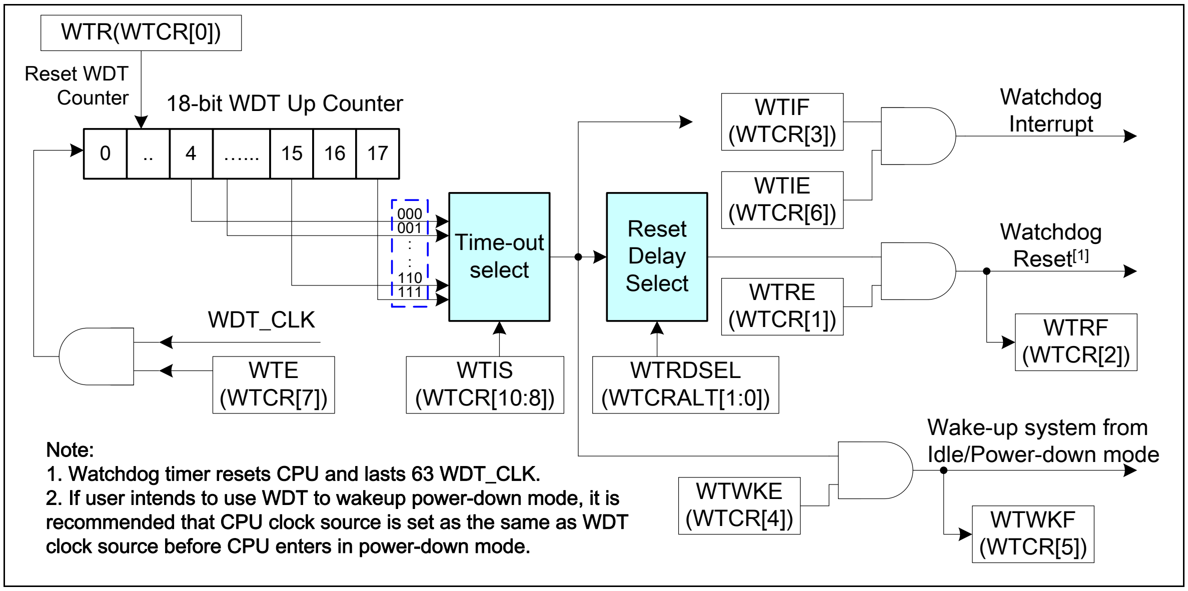


Figure 6-57 Watchdog Timer Block Diagram

6.10.4 Functional Description

The Watchdog Timer (WDT) includes an 18-bit free running up counter with programmable time-out intervals. Table 6-12 shows the WDT time-out interval selection and Figure 6-58 shows the timing of WDT time-out interval and reset period.

Setting WTE (WDTCR[7] WDT enable) bit to 1 will enable the WDT function and the WDT counter to start counting up. When the counter reaches the selected time-out interval (WTIS settings), WTIF (WTCR[3] WDT interrupt flag) will be set to 1 immediately, in the meanwhile, a specified WDT reset delay period (WTCRALT[1:0] WTRDSEL) follows the WTIF is setting to 1. User must set WTR (WDTCR[0] Reset WDT counter) 1 to reset the 18-bit WDT counter value to avoid generate WDT time-out reset signal before the WDT reset delay period expires. And WTR bit is cleared automatically by hardware after WDT counter is reset.

There are eight time-out interval period can be selected by setting WTIS (WTCR[10:8] WDT interval selection). If the WDT counter value has not been cleared after the specific WDT reset delay period expires, the WDT control will set WTRF (WTCR[2] WDT reset flag) to 1 if WTRE (WTCR[1] WDT reset enable) bit is enabled, then chip will reset immediately. This reset period will keep last 63 WDT clocks (T_{RST}) then chip restarts executing program from reset vector (0x0000_0000). And WTRF bit will not be cleared after WDT time-out reset the chip, user can check WTRF bit by software to recognize the system has been reset by WDT time-out reset or not.

The WDT also provides system wake-up function from Idle/Power-Down mode while WTIE (WTCR[6] WDT interrupt enable) bit is enabled and WTIF (WTCR[3] WDT interrupt flag) is set to 1.

WTIS	Time-Out Interval Period T_{TIS}	Reset Delay Period T_{RSTD}
000	2^4 * T_{WDT}	(1/16/128/1024 + 2) * T_{WDT}
001	2^6 * T_{WDT}	(1/16/128/1024 + 2) * T_{WDT}
010	2^8 * T_{WDT}	(1/16/128/1024 + 2) * T_{WDT}
011	2^{10} * T_{WDT}	(1/16/128/1024 + 2) * T_{WDT}
100	2^{12} * T_{WDT}	(1/16/128/1024 + 2) * T_{WDT}
101	2^{14} * T_{WDT}	(1/16/128/1024 + 2) * T_{WDT}
110	2^{16} * T_{WDT}	(1/16/128/1024 + 2) * T_{WDT}
111	2^{18} * T_{WDT}	(1/16/128/1024 + 2) * T_{WDT}

Table 6-12 Watchdog Timer Time-out Interval Selection

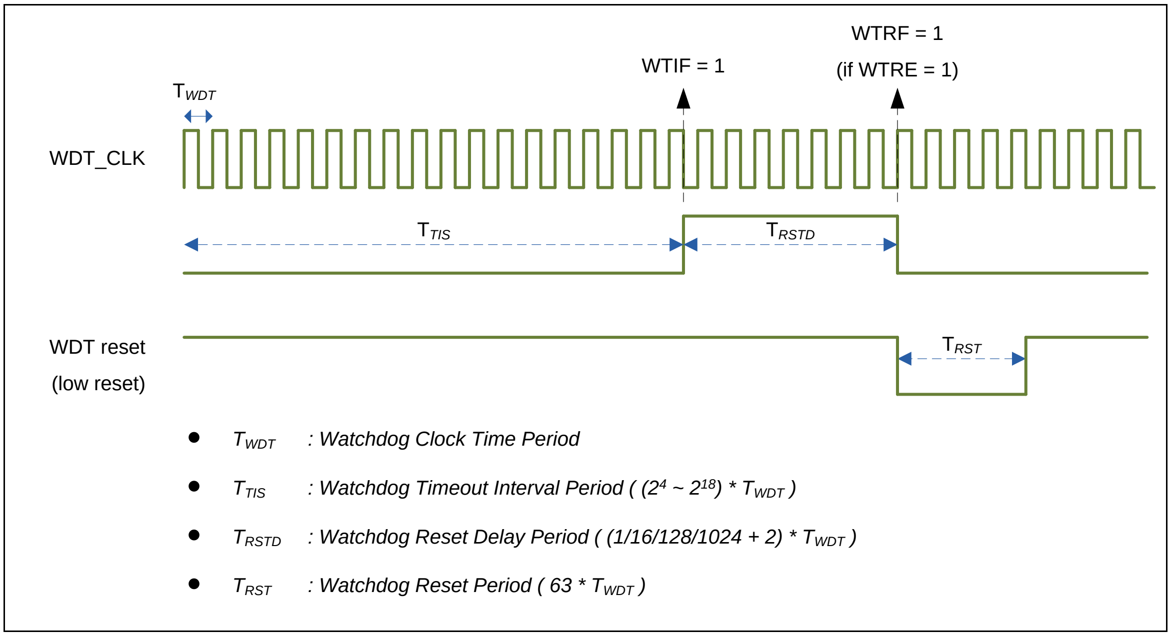


Figure 6-58 Watchdog Timer Time-out Interval and Reset Period Timing

6.10.5 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

6.10.6 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

Note: All bits that can be written in this register are write-protected. To program it needs to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

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 Watchdog Timer counting. Watchdog Timer counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. Watchdog Timer counter will keep going no matter CPU is held by ICE or not.
[30:11]	**Reserved**	Reserved.
[10:8]	**WTIS**	**Watchdog Timer Interval Selection (Write Protect)** These three bits select the time-out interval period for the Watchdog Timer. 000 = 2^{4} * WDT_CLK. 001 = 2^{6} * WDT_CLK. 010 = 2^{8} * WDT_CLK. 011 = 2^{10} * WDT_CLK. 100 = 2^{12} * WDT_CLK. 101 = 2^{14} * WDT_CLK. 110 = 2^{16} * WDT_CLK. 111 = 2^{18} * WDT_CLK.
[7]	**WTE**	**Watchdog Timer Enable Bit (Write Protect)** 0 = Watchdog Timer Disabled (This action will reset the internal counter). 1 = Watchdog Timer Enabled. **Note:** If CWDTEN (Config0[31] watchdog enable) bit is set to 0, this bit is forced as 1 and software cannot change this bit to 0.
[6]	**WTIE**	**Watchdog Timer Interrupt Enable Bit (Write Protect)**

		If this bit is enabled, the WDT time-out interrupt signal is generated and inform to CPU. 0 = Watchdog Timer interrupt Disabled. 1 = Watchdog Timer interrupt Enabled.
[5]	**WTWKF**	**Watchdog Timer Wake-up Flag** This bit indicates the interrupt wake-up flag status of WDT 0 = Watchdog Timer 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 this bit..
[4]	**WTWKE**	**Watchdog Timer Wake-up Function Enable Bit (Write Protect)** If this bit is set to 1, while WDT interrupt flag (WTCR[3] WTIF) is generated to 1 and WTIE (WTCR[6] WDT interrupt enable) is enabled, the WDT time-out interrupt signal will generate a wake-up trigger event to chip. 0 = Wake-up trigger event Disabled if WDT time-out interrupt signal generated. 1 = Wake-up trigger event Enabled if WDT time-out interrupt signal generated.. **Note:** 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 Interrupt Flag** This bit will set to 1 while WDT counter value reaches the selected WDT time-out interval 0 = Watchdog Timer time-out interrupt did not occur. 1 = Watchdog Timer time-out interrupt occurred. **Note:** This bit is cleared by writing 1 to this bit.
[2]	**WTRF**	**Watchdog Timer Reset Flag** This bit indicates the system has been reset by WDT time-out reset or not. 0 = Watchdog Timer time-out reset did not occur. 1 = Watchdog Timer time-out reset occurred. **Note:** This bit is cleared by writing 1 to this bit.
[1]	**WTRE**	**Watchdog Timer Reset Enable Bit (Write Protect)** Setting this bit will enable the Watchdog Timer time-out reset function if the WDT counter value has not been cleared after the specific WDT reset delay period expires. 0 = Watchdog Timer time-out reset function Disabled. 1 = Watchdog Timer time-out reset function Enabled.
[0]	**WTR**	**Reset Watchdog Timer Counter (Write Protect)** 0 = No effect. 1 = Reset the internal 18-bit WDT counter. **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 Select (Write Protect)** When WDT time-out happened, software has a time named WDT reset delay period to clear WDT counter to prevent WDT time-out reset happened. Software 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 (1024+2) * WDT_CLK. 01 = Watchdog Timer reset delay period is (128+2) * WDT_CLK. 10 = Watchdog Timer reset delay period is (16+2) * WDT_CLK. 11 = Watchdog Timer reset delay period is (1+2) * WDT_CLK. This register will be reset to 0 if WDT time-out reset happened

6.11 Window Watchdog Timer (WWDT)

6.11.1 Overview

The purpose of Window Watchdog Timer 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 (WWDTVAL[5:0]) and 6-bit compare value (WWDTCR[21:16] – WINCMP value) to make the window period flexible
- Selectable maximum 11-bit WWDT clock prescale (WWDTCR[11:8] – PERIODSEL value) to make WWDT time-out interval variable

6.11.3 Block Diagram

The Window Watchdog Timer block diagram is shown as follows.

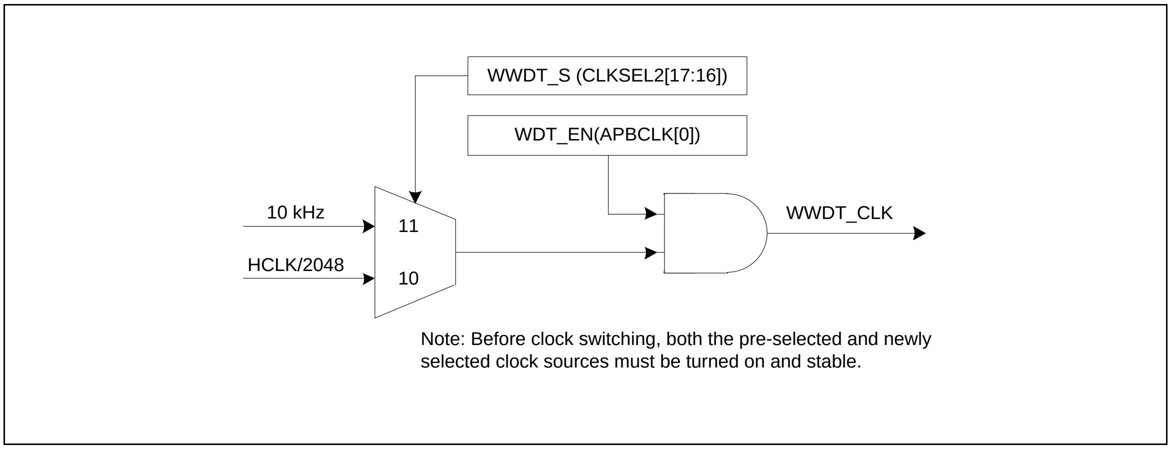


Figure 6-59 Window Watchdog Timer Clock Control

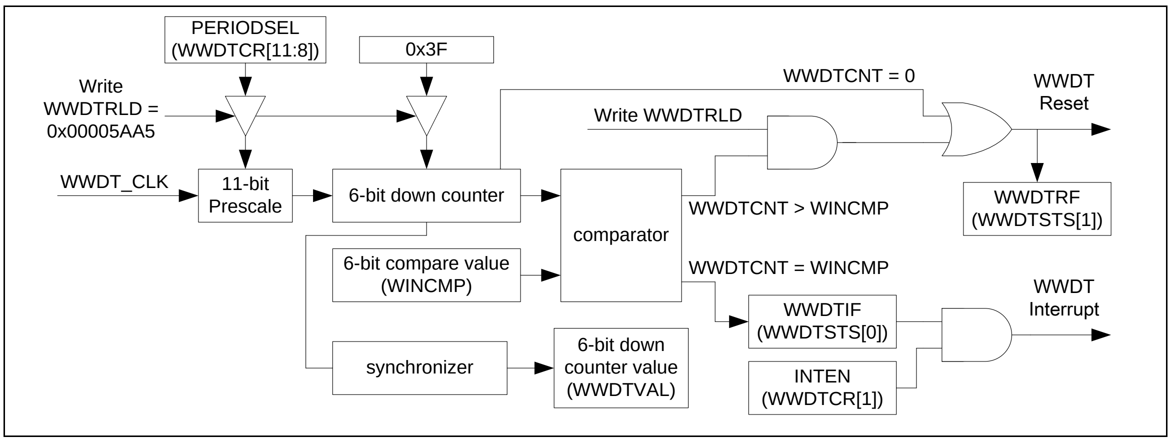


Figure 6-60 Window Watchdog Timer Block Diagram

6.11.4 Functional Description

The Window Watchdog Timer includes a 6-bit down counter with programmable prescale value to define different time-out intervals.

The clock source of 6-bit Window Watchdog Timer is based on system clock divide 2048 (HCLK/2048) or internal 10 kHz oscillator with a programmable maximum of 11-bit prescale value. Also, the programmable 11-bit prescale value is controlled by PERIODSEL (WWDTCR[11:8] WWDT prescale period select) and the correlate of PERIODSEL and prescale value are listed in Table 6-13.

PERIODSEL	Prescaler Value	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 6-13 Window Watchdog Timer Prescale Value Selection

The Window Watchdog Timer can be enabled only once by software setting WWDTEN (WWDTCR[0] WWDT enable) bit to 1 after chip power on or reset and the WWDT down counter will start counting from 0x3F and cannot be stopped by software unless chip has been reset again.

During down counting by the WWDT counter, the WWDTIF (WWDTSR[0] WWDT compare match interrupt flag) is set to 1 if the WWDT counter value is equal to WINCMP (WWDTCR[21:16] WWDT window compare register) value; if WWDTIE (WWDTCR[1] WWDT interrupt enable) is also set to 1 by software, the WWDT time-out interrupt signal is generated also while WWDTIF is set to 1 by hardware.

The WWDT time-out reset signal is generated when the WWDT counter value reaches 0. Before WWDT counter down counting to 0, software can write 0x00005AA5 to WWDTRLD register to reload WWDT internal counter value to 0x3F to prevent WWDT time-out reset happen when current WWDT counter value (WWDTCVR value) is equal to or smaller than WINCMP value. If current WWDT counter value (WWDTCVR value) is larger than WINCMP value and software writes 0x00005AA5 to the WWDTRLD register, WWDT reset signal will be generated to cause chip reset. Figure 6-61 shows the reset and reload behavior of WWDT.

To prevent program from running to disable Window Watchdog Timer counter counting unexpected, the control register WWDTCR can only be written once after chip is powered on or reset. Software cannot disable Window Watchdog Timer counter counting (WWDTEN), change time-out prescale period (PERIODSEL) or change window compare value (WINCMP) while WWDTEN bit has been enabled by software unless chip is reset.

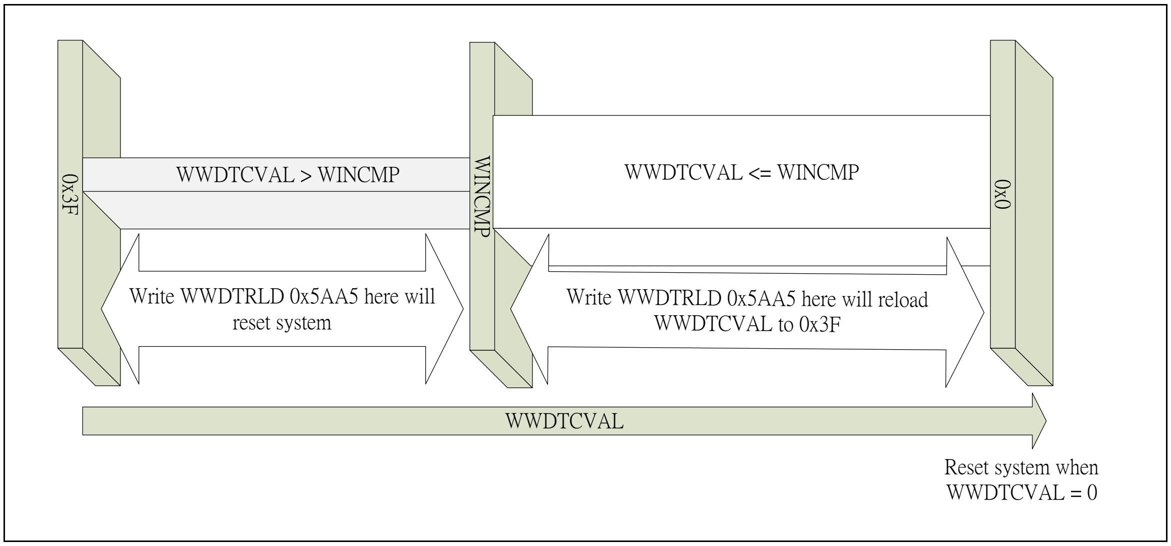


Figure 6-61 Window Watchdog Timer Reset and Reload Behavior

When software writes 0x00005AA5 to WWDTRLD register to reload WWDT counter value to 0x3F, it needs 3 WWDT clocks to sync reload command to actually perform reload action. It means if software set PERIODSEL (WWDTCR[11:8] WWDT prescale period select) to 0, the prescale value should be as 1, and the WINCMP (WWDTCR[21:16] WWDT window compare register) value must be larger than 2; otherwise, writing WWDTRLD by software to reload WWDT counter value to 0x3F is unavailable and WWDT time-out reset always happened. Table 6-14 shows the limitation of WINCMP.

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

Table 6-14 WINCMP Setting Limitation

6.11.5 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

6.11.6 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 Window Watchdog Timer counter value to 0x3F. **Note:** Software can only write WWDTRLD to reload WWDT counter value when current WWDT counter value between 0 and WINCMP. If software 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 = WWDT counter stopped if system is in Debug mode. 1 = WWDT still counted even system is in Debug mode.
[30:22]	**Reserved**	Reserved.
[21:16]	**WINCMP**	**WWDT Window Compare Register** Set this register to adjust the valid reload window. **Note:** Software can only write WWDTRLD to reload WWDT counter value when current WWDT counter value between 0 and WINCMP. If Software writes WWDTRLD when current WWDT counter value larger than WINCMP, WWDT reset signal will generate immediately.
[15:12]	**Reserved**	Reserved.
[11:8]	**PERIODSEL**	**WWDT Prescale Period Select** These 4-bit select the prescale period for the WWDT counter period. 0000 = Pre-scale is 1; Max time-out period is 1 * 64 * WWDT_CLK. 0001 = Pre-scale is 2; Max time-out period is 2 * 64 * WWDT_CLK. 0010 = Pre-scale is 4; Max time-out period is 4 * 64 * WWDT_CLK. 0011 = Pre-scale is 8; Max time-out period is 8 * 64 * WWDT_CLK. 0100 = Pre-scale is 16; Max time-out period is 16 * 64 * WWDT_CLK. 0101 = Pre-scale is 32; Max time-out period is 32 * 64 * WWDT_CLK. 0110 = Pre-scale is 64; Max time-out period is 64 * 64 * WWDT_CLK. 0111 = Pre-scale is 128; Max time-out period is 128 * 64 * WWDT_CLK. 1000 = Pre-scale is 192; Max time-out period is 192 * 64 * WWDT_CLK. 1001 = Pre-scale is 256; Max time-out period is 256 * 64 * WWDT_CLK. 1010 = Pre-scale is 384; Max time-out period is 384 * 64 * WWDT_CLK. 1011 = Pre-scale is 512; Max time-out period is 512 * 64 * WWDT_CLK. 1100 = Pre-scale is 768; Max time-out period is 768 * 64 * WWDT_CLK. 1101 = Pre-scale is 1024; Max time-out period is 1024 * 64 * WWDT_CLK.

		1110 = Pre-scale is 1536; Max time-out period is 1536 * 64 * WWDT_CLK. 1111 = Pre-scale is 2048; Max time-out period is 2048 * 64 * WWDT_CLK.
[7:2]	**Reserved**	Reserved.
[1]	**WWDTIE**	**WWDT Interrupt Enable Bit** Set this bit to enable the Window Watchdog Timer time-out interrupt function. 0 = WWDT time-out interrupt function Disabled if WWDTIF (WWDTSR[0] WWDT compare match interrupt flag) is 1. 1 = WWDT time-out interrupt function Enabled if WWDTIF (WWDTSR[0] WWDT compare match interrupt flag) is 1.
[0]	**WWDTEN**	**WWDT Enable Bit** Set this bit to enable Window Watchdog Timer counter counting. 0 = Window Watchdog Timer counter is stopped. 1 = Window Watchdog Timer 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 Reset Flag** When WWDT counter counts down to 0 or writes WWDTRLD during current WWDT counter value being larger than WINCMP, chip will be reset and this bit is set to 1. This bit will be cleared to 0 by writing 1 to itself.
[0]	**WWDTIF**	**WWDT Compare Match Interrupt Flag** When current WWDT counter value matches WWCMP, this bit is set to 1. This bit will be cleared by writing 1 to itself.

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 (Read Only)** This register reflects the current WWDT counter value.

6.12 Real Time Clock (RTC)

6.12.1 Overview

The Real Time Clock (RTC) controller provides user with the real time and calendar message. The clock source of RTC controller is from an external 32.768 kHz low speed crystal which connected at pins X32I and X32O (refer to pin Description) or from an external 32.768 kHz low speed oscillator output fed at pin X32I. The RTC controller provides the real time message (hour, minute, second) in TLR (RTC Time Loading Register) as well as calendar message (year, month, day) in CLR (RTC Calendar Loading Register). It also offers RTC alarm function that user can preset the alarm time in TAR (RTC Time Alarm Register) and alarm calendar in CAR (RTC Calendar Alarm Register). The data format of RTC time and calendar message are all expressed in BCD format.

The RTC controller supports periodic RTC Time Tick and Alarm Match interrupts. The periodic RTC 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 which are selected by TTR (TTR[2:0] Time Tick Register). When real time and calendar message in TLR and CLR are equal to alarm time and calendar settings in TAR and CAR, the AIF (RIIR [0] RTC Alarm Interrupt Flag) is set to 1 and the RTC alarm interrupt signal is generated if the AIER (RIER [0] Alarm Interrupt Enable) is enabled.

Both RTC Time Tick and Alarm Match interrupt signal can cause chip to wake-up from Idle or Power-down mode if the correlate interrupt enable bit (AIER or TIER) is set to 1 before chip enters Idle or Power-down mode.

6.12.2 Features

- Supports real time counter in TLR (hour, minute, second) and calendar counter in CLR (year, month, day) for RTC time and calendar check
- Supports alarm time (hour, minute, second) and calendar (year, month, day) settings in TAR and CAR
- Selectable 12-hour or 24-hour time scale in TSSR register
- Supports Leap Year indication in LIR register
- Supports Day of the Week counter in DWR register
- Frequency of RTC clock source compensate by 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

6.12.3 Block Diagram

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

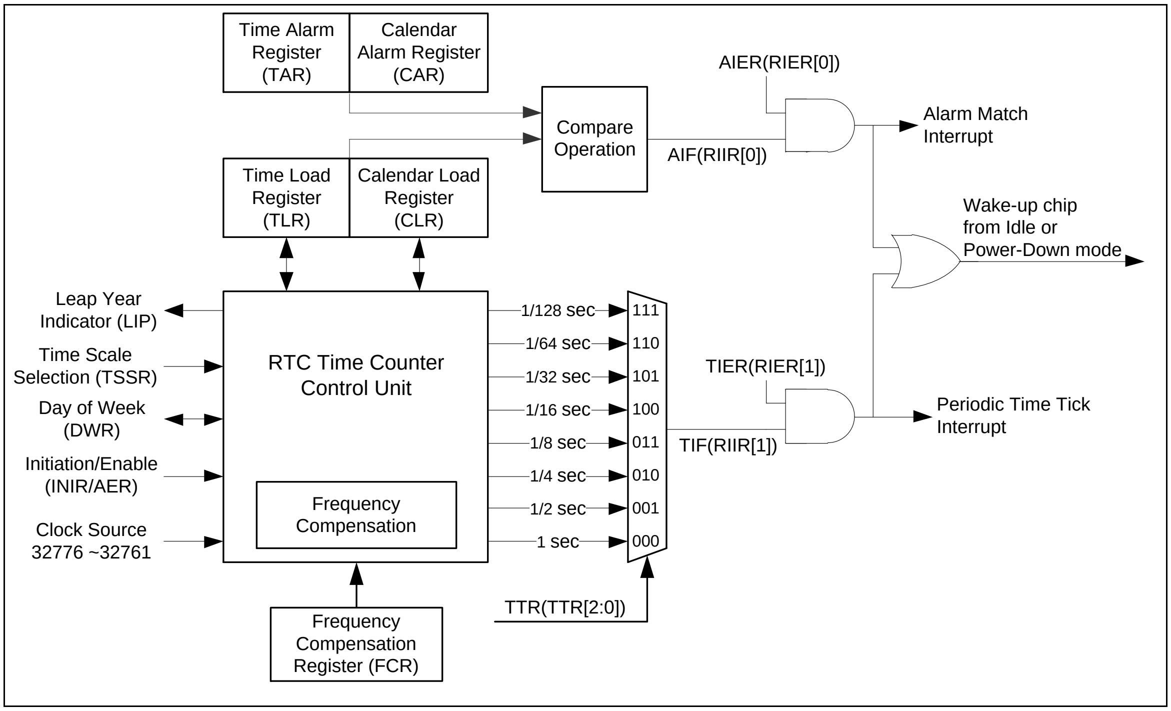


Figure 6-62 RTC Block Diagram

6.12.4 Functional Description

6.12.4.1 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 (INIR[0] RTC Active Status) bit status to check the RTC is at normal active state or reset state.

6.12.4.2 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.

6.12.4.3 RTC Read/Write Enable

AER [AER [15:0] RTC Register Access Enable Password] is served as read/write access of RTC registers to unlock RCT registers read/write protect 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 access 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.

6.12.4.4 Frequency Compensation

The RTC source clock may not precise to exactly 32768 Hz and the FCR register (Frequency Compensation Register) allows software 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	FCR[11:8]	Integer part of detected value	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

Table 6-15 Integer part of frequency compensation

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

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 FCR register should be as 0x410

6.12.4.5 Time and Calendar counter

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

6.12.4.6 12/24 hour Time Scale Selection

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

24-hour Time Scale	12-hour Time Scale	24-hour Time Scale	12-hour Time Scale (PM time + 20)
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)

Table 6-16 12/24 hour Time Scale Selection

6.12.4.7 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.

6.12.4.8 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.

6.12.4.9 Alarm Interrupt

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

6.12.4.10 Application ***Note:***

1. All data in TAR, CAR, TLR and CLR registers are all expressed in BCD format.
2. Programmer 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 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	RTC Time Loading Register	0x0000_0000
CLR	RTC_BA+0x10	R/W	RTC Calendar Loading Register	0x0005_0101
TSSR	RTC_BA+0x14	R/W	RTC Time Scale Selection Register	0x0000_0001
DWR	RTC_BA+0x18	R/W	RTC Day of the Week Register	0x0000_0006
TAR	RTC_BA+0x1C	R/W	RTC Time Alarm Register	0x0000_0000
CAR	RTC_BA+0x20	R/W	RTC Calendar Alarm Register	0x0000_0000
LIR	RTC_BA+0x24	R	RTC Leap Year Indication Register	0x0000_0000
RIER	RTC_BA+0x28	R/W	RTC Interrupt Enable Register	0x0000_0000
RIIR	RTC_BA+0x2C	R/W	RTC Interrupt Indication Register	0x0000_0000
TTR	RTC_BA+0x30	R/W	RTC Time Tick Register	0x0000_0000

6.12.6 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	
[31:0]	**INIR**	**RTC Initiation** Read return current RTC active status 0 = RTC is at reset state. 1 = RTC is at normal active state. **A write of 0xa5eb1357 will make RTC leaving reset state.** When RTC block is powered on, RTC is at reset state. User has to write a number 0x a5eb1357 to INIR register to make RTC leaving reset state. Once the INIR is written as 0xa5eb1357, the RTC will be in normal active state permanently.

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)** This bit indicates the RTC Access Enable status 0 = RTC register read/write access Disabled. 1 = RTC register read/write access Enabled. This bit will be set after AER[15:0] register is loaded with 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 registers read/write 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** Digit in FCR must be expressed as hexadecimal number. Refer to 6.12.4.4 for the examples.
[7:6]	**Reserved**	Reserved.
[5:0]	**FRACTION**	**Fraction Part** Formula = (fraction part of measured value) x 60. **Note:** Digit in FCR must be expressed as hexadecimal number. Refer to 6.12.4.4 for the examples.

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

RTC Time Loading Register (TLR)

Register	Offset	R/W	Description	Reset Value
TLR	RTC_BA+0x0C	R/W	RTC 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)

Note:

1. TLR is a BCD digit counter and RTC controller will not check the loaded data is reasonable or not.
2. The reasonable value range is listed in the parenthesis.
3. 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	RTC Calendar Loading Register	0x0005_0101

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="4">10YEAR</td><td colspan="4">1YEAR</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="3">Reserved</td><td>10MON</td><td colspan="4">1MON</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">10DAY</td><td colspan="4">1DAY</td></tr>
</table>

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)

Note:

1. CLR is a BCD digit counter and RTC will not check the loaded data is reasonable or not.
2. 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	RTC Time Scale Selection Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
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 TLR and TAR counter are in 24-hour time scale or 12-hour time scale 0 = Selected as 12-hour time scale with AM and PM indication (high bit of 10HR field in TLR and TAR). 1 = Selected as 24-hour time scale.

RTC Day of the Week Register (DWR)

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**DWR**	**Day of the Week Register** The field indicates day of the week from 0 to 6 to represent Sunday to Saturday respectively. 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	RTC 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)

Note:

1. This register can be read back only if the ENF (AER [16] RTC Register Access Enable Flag) is enabled
2. TAR is a BCD digit counter and RTC controller will not check the loaded data is reasonable or not.
3. The reasonable value range is listed in the parenthesis.
4. 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	RTC Calendar Alarm Register	0x0000_0000

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="4">10YEAR</td><td colspan="4">1YEAR</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="3">Reserved</td><td>10MON</td><td colspan="4">1MON</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">10DAY</td><td colspan="4">1DAY</td></tr>
</table>

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)

Note:

1. This register can be read back only if the ENF (AER [16] RTC Register Access Enable Flag) is enabled
2. CAR is a BCD digit counter and RTC will not check the loaded data is reasonable or not.
3. 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	RTC Leap Year 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							
7	6	5	4	3	2	1	0
Reserved							LIR

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**LIR**	**Leap Year Indication Register (Read Only)** This bit indicates RTC current year is a leap year or not. 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 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							
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 time-out happened, this bit will be set to 1 and an interrupt signal will be generated if TIER bit is set to 1. **Note:** Software can clear this bit by writing 1 to it.
[0]	**AIF**	**RTC Alarm Interrupt Flag** When RTC real time counters TLR and CLR reach the alarm time setting registers TAR and CAR, this bit will be set to 1 and an interrupt signal will be generated if AIER bit is set to 1. **Note:** Software can clear this bit by writing 1 to it.

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** The 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 only if the ENF (AER [16] RTC Register Access Enable Flag) is enabled

6.13 UART Interface Controller (UART)

The NuMicro® NUC100 series 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.

6.13.1 Overview

The Universal Asynchronous Receiver/Transmitter (UART) 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, LIN master/slave mode and RS-485 mode functions. Each UART channel supports seven types of interrupts including:

- Transmitter FIFO empty interrupt (INT_THRE)
- Receiver threshold level reached interrupt (INT_RDA),
- Line status interrupt (parity error or frame error or break interrupt) (INT_RLS),
- Receiver buffer time-out interrupt (INT_TOUT),
- MODEM/Wake-up status interrupt (INT_MODEM),
- Buffer error interrupt (INT_BUF_ERR)
- LIN interrupt (INT_LIN)

Interrupts of UART0 and UART2 share the interrupt number 12 (vector number is 28); Interrupt number 13 (vector number is 29) only supports UART1 interrupt. Refer to the Nested Vectored Interrupt Controller chapter for System Interrupt Map.

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 the operation. The reported status information includes the type and condition of the transfer operations being performed by the UART, as well as 4 error conditions (parity error, frame error, break interrupt and buffer error) probably occur while receiving data. The UART includes a programmable baud rate generator that is 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). Table 6-17 lists the equations in the various conditions and Table 6-18 lists the UART baud rate setting table.

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), A must >=3

Table 6-17 UART Baud Rate Equation

System Clock = Internal 22.1184 MHz High Speed Oscillator						
Baud Rate	Mode 0		Mode 1		Mode 2	
	Parameter	Register	Parameter	Register	Parameter	Register
921600	x	x	A=0,B=11	0x2B00_0000	A=22	0x3000_0016

460800	A=1	0x0000_0001	A=1,B=15 A=2,B=11	0x2F00_0001 0x2B00_0002	A=46	0x3000_002E
230400	A=4	0x0000_0004	A=4,B=15 A=6,B=11	0x2F00_0004 0x2B00_0006	A=94	0x3000_005E
115200	A=10	0x0000_000A	A=10,B=15 A=14,B=11	0x2F00_000A 0x2B00_000E	A=190	0x3000_00BE
57600	A=22	0x0000_0016	A=22,B=15 A=30,B=11	0x2F00_0016 0x2B00_001E	A=382	0x3000_017E
38400	A=34	0x0000_0022	A=62,B=8 A=46,B=11 A=34,B=15	0x2800_003E 0x2B00_002E 0x2F00_0022	A=574	0x3000_023E
19200	A=70	0x0000_0046	A=126,B=8 A=94,B=11 A=70,B=15	0x2800_007E 0x2B00_005E 0x2F00_0046	A=1150	0x3000_047E
9600	A=142	0x0000_008E	A=254,B=8 A=190,B=11 A=142,B=15	0x2800_00FE 0x2B00_00BE 0x2F00_008E	A=2302	0x3000_08FE
4800	A=286	0x0000_011E	A=510,B=8 A=382,B=11 A=286,B=15	0x2800_01FE 0x2B00_017E 0x2F00_011E	A=4606	0x3000_11FE

Table 6-18 UART Baud Rate Setting Table

The UART0 and UART1 controllers support the auto-flow control function that uses two low-level signals, /CTS (clear-to-send) and /RTS (request-to-send), to control the flow of data transfer between the chip 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 controller detects /CTS is asserted from external device. If a valid asserted /CTS is not detected the UART controller will not send data out.

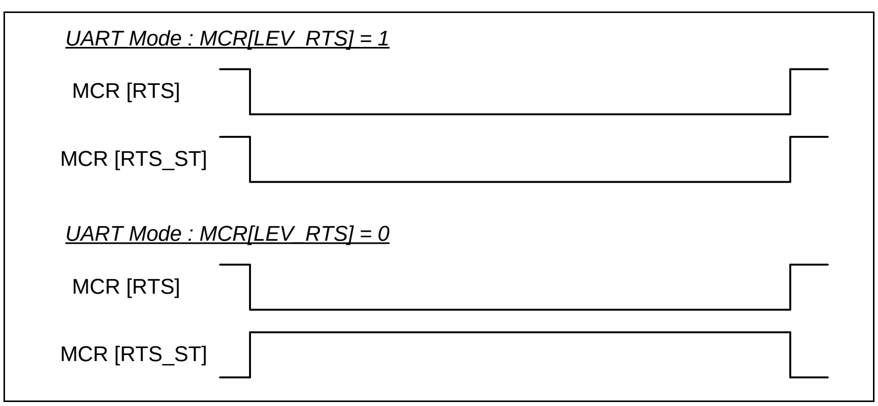


Figure 6-63 UART nRTS Auto-Flow Control Trigger Level

The UART controllers also provides Serial IrDA (SIR, Serial Infrared) function (User must set IrDA_EN (UA_FUN_SEL [1]) to enable IrDA function). The SIR specification defines a short-range infrared asynchronous serial transmission mode with 1 start bit, 8 data bits, and 1 stop bit. The maximum data rate supports up to 115.2 Kbps (half duplex). The IrDA SIR block contains an IrDA

SIR Protocol encoder/decoder. The IrDA SIR Protocol encoder/decoder is half-duplex only. So it cannot transmit and receive data at the same time. The IrDA SIR physical layer specifies a minimum 10ms transfer delay between transmission and reception, and this delay feature must be implemented by software.

The alternate function of UART controllers is LIN (Local Interconnect Network) function. The LIN mode is selected by setting the UA_FUN_SEL[1:0] to '01'. In LIN mode, 1 start bit and 8 data bits format with 1 stop bit are required in accordance with the LIN standard.

For NuMicro® NUC100 series, another alternate function of UART controllers is RS-485 9-bit mode, and direction control provided by /RTS pin or can program GPIO (PB.2 for UART0_nRTS and PB.6 for UART1_nRTS) to implement the function by software. The RS-485 mode is selected by setting the UA_FUN_SEL register to select RS-485 function. The RS-485 transceiver control is implemented using the /RTS control signal from an asynchronous serial port to enable the RS-485 transceiver. In RS-485 mode, many characteristics of the receiving and transmitting are same as UART.

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

6.13.3 Block Diagram

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

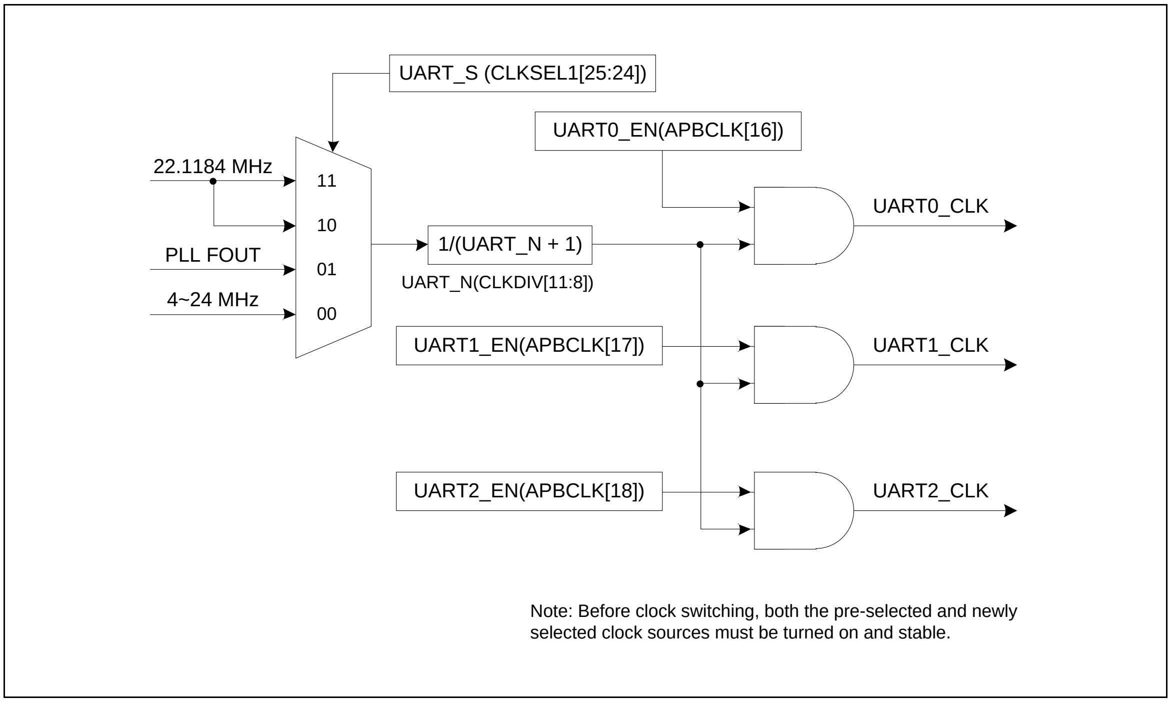


Figure 6-64 UART Clock Control Diagram

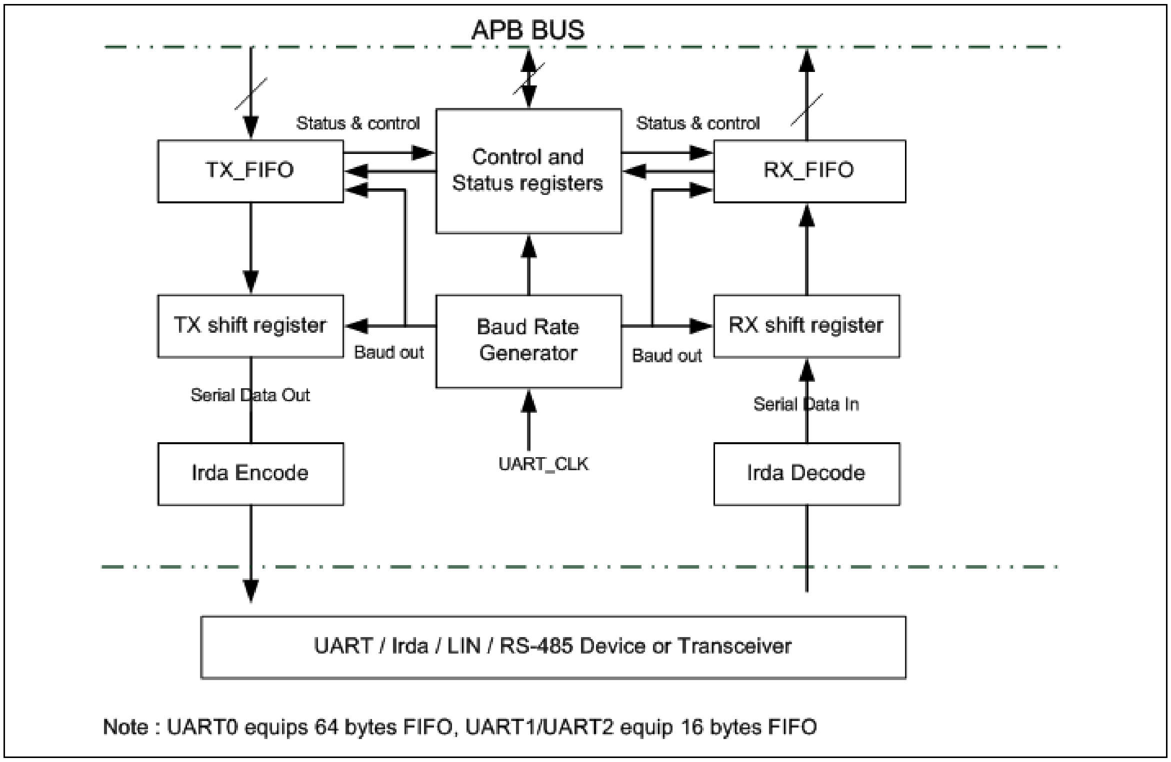


Figure 6-65 UART Block Diagram

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).

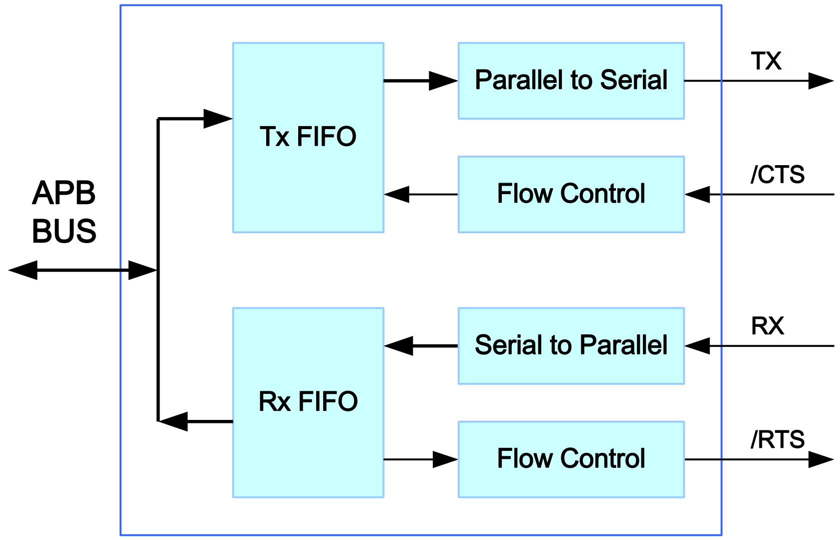
Baud Rate Generator
Divide the external clock by the divisor to get the desired baud rate. Refer to baud rate equation.

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 a set of registers 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 set of registers also include 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 (INT_THRE), receiver threshold level reaching interrupt (INT_RDA), line status interrupt (parity error or framing error or break interrupt) (INT_RLS) , time-out interrupt (INT_TOUT), MODEM/Wake-up status interrupt (INT_MODEM), Buffer error interrupt (INT_BUF_ERR) and LIN bus interrupt.

The following diagram demonstrates the auto-flow control block.



Note: It is only supported in UART0 and UART1

Figure 6-66 Auto Flow Control Block Diagram

6.13.4 IrDA Mode

The UART supports IrDA SIR (Serial Infrared) Transmit Encoder and Receive Decoder, and IrDA mode is selected by setting the in **UA_FUN_SEL** register.

In IrDA mode, the UA_BAUD [DIV_X_EN] register must be disabled.

Baud Rate = Clock / (16 * BRD), where BRD is Baud Rate Divider in UA_BAUD register.

The following diagram demonstrates the IrDA control block.

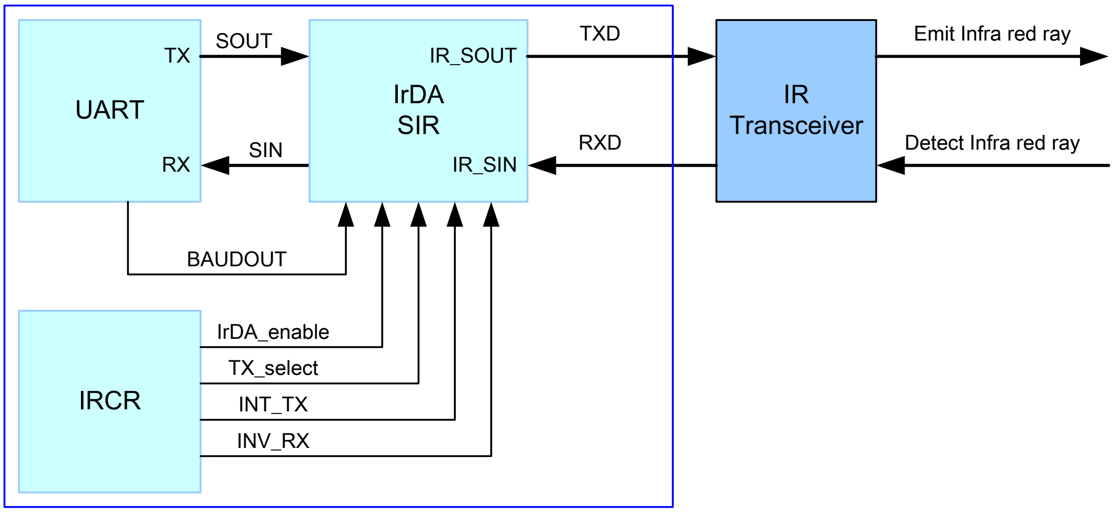


Figure 6-67 IrDA Block Diagram

6.13.4.1 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.

6.13.4.2 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 the idle state. (Because of this, IRCR bit 6 should be set as 1 by default)

A start bit is detected when the decoder input is LOW

6.13.4.3 IrDA SIR Operation

The IrDA SIR Encoder/Decoder provides functionality which converts between UART data stream and half duplex serial SIR interface. The following diagram is IrDA encoder/decoder waveform:

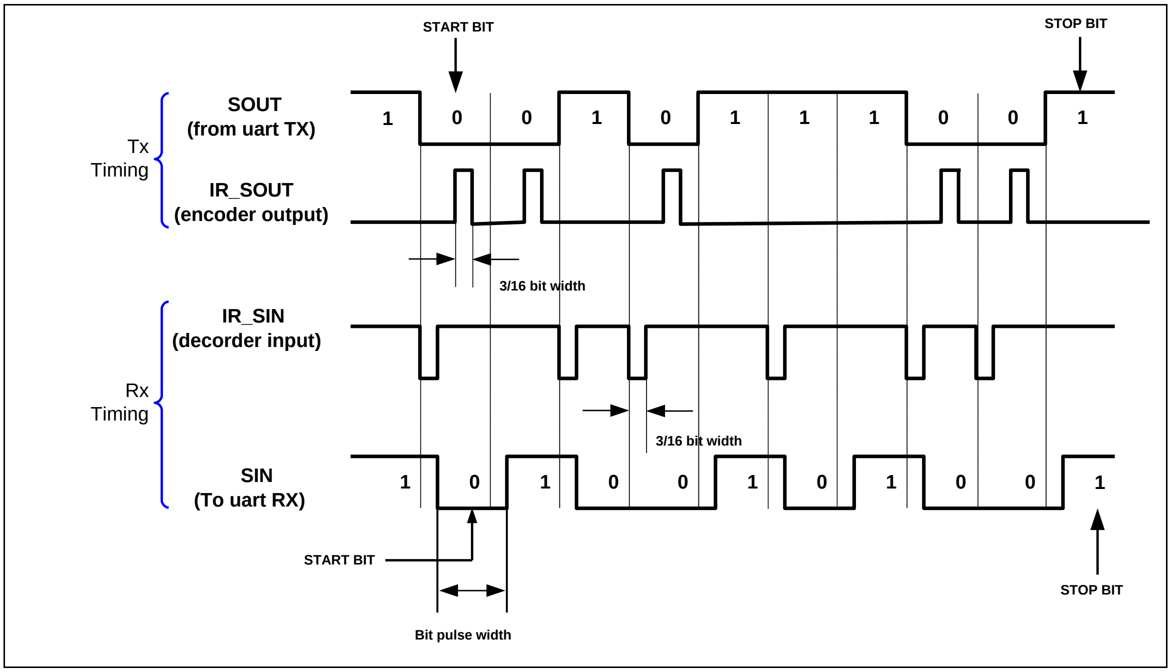


Figure 6-68 IrDA Timing Diagram

6.13.5 LIN (Local Interconnection Network) Mode

The UART supports LIN function, and LIN mode is selected by setting the UA_FUN_SEL[1:0] to '01'. The UART supports LIN break/delimiter generation and break/delimiter detection in LIN Master mode, and supports header detection and automatic resynchronization in LIN Slave mode.

6.13.5.1 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 following diagram is the structure of LIN Frame.

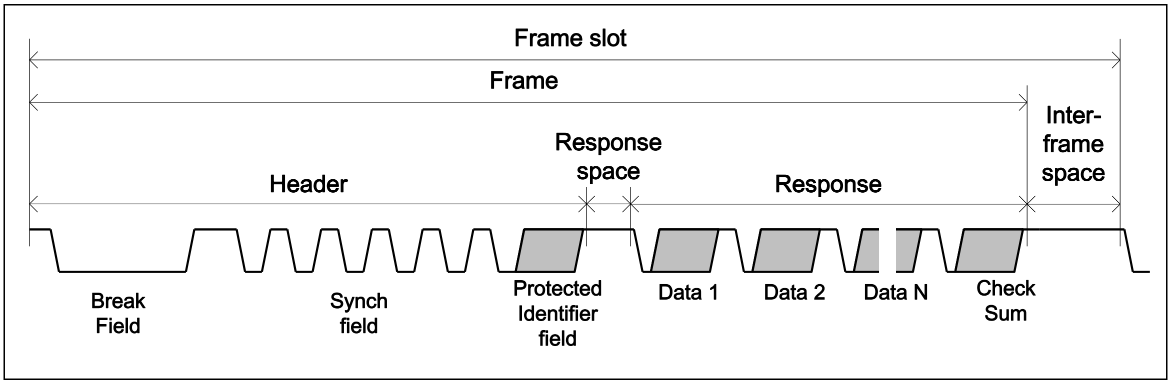


Figure 6-69 Structure of LIN Frame

6.13.5.2 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 (UA_LCR [WLS] = 2'b11) and no parity bit, LSB is first and ended by 1 stop bit (UA_LCR [NSB] = 1) with value 1 (recessive) in accordance with the LIN standard. The structure of Byte is shown as follows.

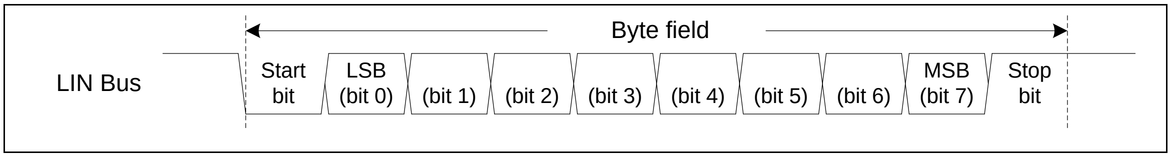


Figure 6-70 Structure of LIN Byte

6.13.5.3 LIN Master Mode

The UART controller supports LIN Master mode by setting the UA_FUN_SEL register. To enable and initialize the LIN Master mode, the following steps are necessary.

1. Select the desired baud-rate by setting the UA_BAUD register.
2. Configure the data length to 8 bits by setting UA_LCR[WLS] = 11 and disable parity check by clearing UA_LCR[PBE] bit and configure the stop bit to 1 by clearing UA_LCR[NSB] bit.
3. Select LIN function mode by setting UA_FUN_SEL register.

A complete header consists of a break field and sync field followed by a frame identifier (frame ID). The UART controller can be selected header sending by three header selected modes. The header selected mode can be "break field" or "break field and sync field" or "break field, sync field and frame ID field" by setting LIN_HEAD_SEL in UA_LIN_CTL register. 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 header selected is "break field and sync field", software must handle the sequence to send a complete header to bus by filled 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 UA_LIN_CTL [LIN_PID] register. 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 UA_LIN_CTL [LIN_IDPEN] 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 UA_LIN_CTL [LIN_PID] first)

Table 6-19 LIN Header Selection in Master mode

When operating in LIN data transmission, LIN bus transfer state can be monitored by hardware or software. User can enable hardware monitoring by setting UA_LIN_CTL [BIT_ERR_EN] to "1", if the input pin (SIN) state is not equal to the output pin (SOUT) 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 [UA_LIN_CTL]LIN_PAD.
2. Select the hardware transmission header field including "break field + sync field + Protected identifier field" by setting UA_LIN_CTL [LIN_HEAD_SEL] = 10
3. Select the hardware transmission header field by setting UA_LIN_CTL [LIN_HEAD_SEL]).
4. Request header transmission by setting the LIN_SHD bit in the UA_LIN_CTL register.

5. Wait until UA_LIN_CTL[LIN_SHD] be cleared by hardware.
6. Wait until UA_FSR[TE_FLAG] set to “1” by hardware

Note1: The default setting of break field + break/sync delimiter is 13 dominant bits(break field) and 1 recessive bit(break/sync delimiter), software can change it by setting UA_LIN_CTL [LIN_BKFL] and UA_LIN_CTL [LIN_BS_LEN], to change the dominant bits.

Note2: The default setting of break/sync delimiter length is 1 bit time and the default setting of inter-byte spaces is also 1 bit time; software can change them by setting UA_LIN_CTL [LIN_BS_LEN] and UA_TOR [DLY].

Note3: If the header includes the “break field, sync field and frame ID field”, software must fill frame ID in UA_LIN_CTL [LIN_PID] register before trigger header transmission (setting the LIN_SHD bit in the UA_LIN_CTL register). The frame ID parity can be generated by software or hardware depends on UA_LIN_CTL [LIN_IDPEN]. If the parity generated by software (UA_LIN_CTL [LIN_IDPEN] = 0), software must fill 8 bit data (include 2 bit parity) in this field, and if the parity generated by hardware (UA_LIN_CTL [LIN_IDPEN] = 1), software fill ID0~ID5, hardware will calculi P0 and P1.

Procedure with software error monitoring in Master mode:

1. Choose the hardware transmission header field only including “break field” by setting UA_LIN_CTL [LIN_HEAD_SEL] = 0x00.
2. Enable break detection function by setting LIN_BKDET_EN bit in UA_LIN_CTL.
3. Request break + break/sync delimiter transmission by setting the LIN_SHD bit in the UA_LIN_CTL register.
4. Wait until the LIN_BKDET_F flag in the UA_LIN_SR register be set to “1” by hardware..
5. Request sync field transmission by writing 0x55 into UA_THR register.
6. Wait until the RDA_IF flag in the UA_ISR register be 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 flag in the UA_ISR register be 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 bit in UA_LIN_CTL register, the break detection circuit is activated. The break detection circuit is totally independent from the UART receiver.

When the break detection function is enabled, the circuit looks at the input SIN pin for a start signal. If circuit detected consecutive dominant is greater than 11 bits dominant followed by a recessive bit (delimiter), the UA_LIN_SR [LIN_BKDET_F] flag is set at the end of break field. If the UA_IER [LIN_IEN] bit =1, an interrupt will be generated. The behavior of the break detection and break flag are shown in the following figure.

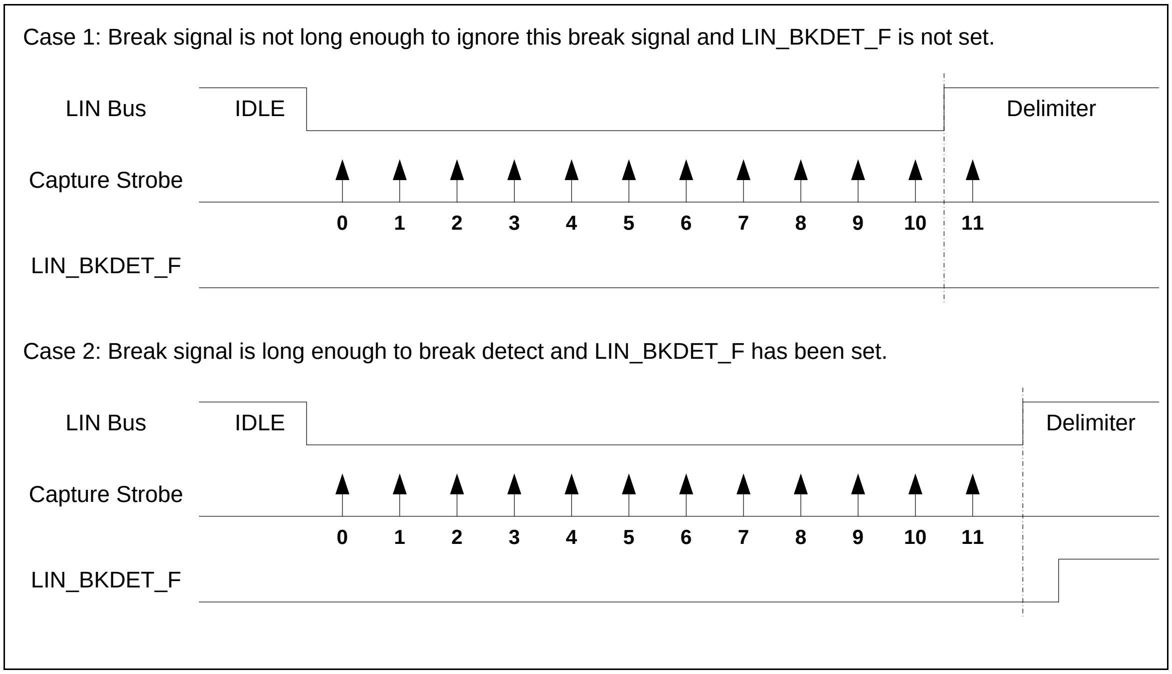


Figure 6-71 Break Detection in LIN Mode

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.

6.13.5.4 LIN Slave Mode

The UART controller supports LIN Slave mode by setting the LINS_EN bit in UA_LIN_CTL register. To enable and initialize the LIN Slave mode, the following steps are necessary:

1. Select the desired baud-rate by setting the UA_BAUD register.
2. Configure the data length to 8 bits by setting UA_LCR[WLS] = 11and disable parity check by clearing UA_LCR[PBE] bit and configure the stop bit to 1 by clearing UA_LCR[NSB] bit.
3. Select LIN function mode by setting UA_FUN_SEL register.
4. Enable LIN Slave mode by setting the LINS_EN bit in UA_LIN_CTL.

LIN header reception

According to the LIN protocol, a slave node must wait for a valid header which came from the master node. Then application 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 bit in UA_LIN_CTL register to detect complete frame header (receive “break field”, “sync field” and “frame ID field”). When a LIN header is received, the LINS_HDET_F flag in UA_LIN_SR register will be set (If the UA_IER [LIN_IEN] bit =1, an interrupt will be generated). User can enable the frame ID parity check function by setting LIN_IDPEN bit in UA_LIN_CTL register. If only received frame ID parity is not correct (break and sync filed are correct), the [UA_LIN_SR]LIN_IDPERR_F flag (If the UA_IER [LIN_IEN] bit =1, an interrupt will be generated) and UA_LIN_SR [LINS_HDET_F] both will be set. User can also put LIN in Mute mode by setting LIN_MUTE_EN bit in UA_LIN_CTL register. 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 bit in UA_LIN_CTL register.

LIN response transmission

A LIN slave node can transmit and receive response. When a slave node is the publisher of the response, the slave node send response by filling data to UA_THR register, and 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 (UA_FSR [LIN_HERR_F]) 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 assert.
- Writing 1 to the LINS_SYNC_F bit in UA_LIN_SR register to re-search a new frame header.

Mute mode and LIN exit from mute mode condition

In Mute mode, LIN slave node will not receive any data until specified condition occurred. It allows detection of headers only and prevents the reception of any other characters. User can enable Mute mode by setting LIN_MUTE_EN bit in UA_LIN_CTL register and exit from Mute mode condition can be selected by LIN_HEAD_SEL in UA_LIN_CTL register.

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 [UA_LIN_CTL] LIN_HEAD_SEL 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 [UA_LIN_CTL]LIN_HEAD_SEL 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 [UA_LIN_CTL]LIN_HEAD_SEL 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 UA_LIN_CTL [LIN_PID] 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 need 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 register.
3. Disable automatic resynchronization function by setting UA_LIN_CTL[LINS_ARS_EN] = 0.
4. Enable LIN Slave mode by setting the LINS_EN bit in UA_LIN_CTL.

Slave mode with automatic resynchronization

User can enable the automatic resynchronization function by setting LINS_ARS_EN bit in UA_LIN_CTL register. 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 register.
3. Enable automatic resynchronization function by setting UA_LIN_CTL[LINS_ARS_EN] = 1.
4. Enable LIN slave mode by setting the LINS_EN bit in UA_LIN_CTL.

When the automatic resynchronization function is enabled, after each LIN break field, the time duration between five falling edges is sampled on peripheral clock and the result of this measurement is stored in an internal 13-bit register and the UA_BAUD register value will be automatically updated at the end of the fifth falling edge. If the measure timer (13bit) overflows before five falling edges, then the header error flag (UA_LIN_SR [LIN_HERR_F]) will be set.

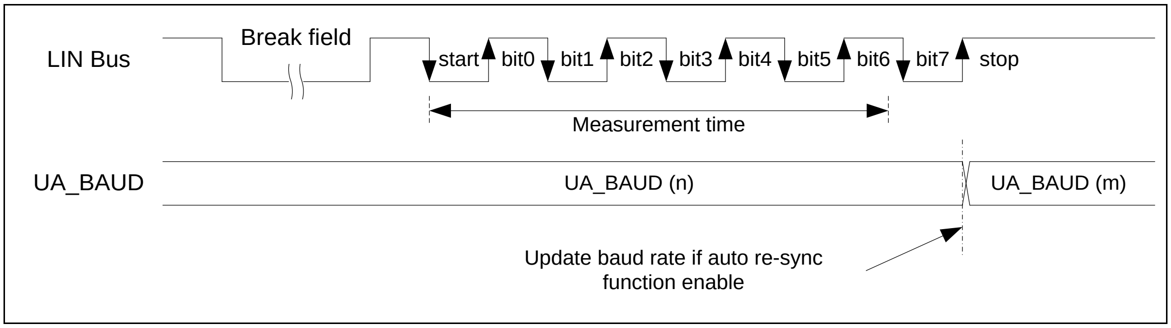


Figure 6-72 LIN Sync Field Measurement

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 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 setting UA_LIN_CTL [LINS_DUM_EN] bit to enable auto reload initial baud rate value function. If the LINS_DUM_EN 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 bit 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 bit in UA_LIN_SR register 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 mode2 (UA_BAUD [DIV_X_EN] and UA_BAUD [DIV_X_ONE] must be 1).

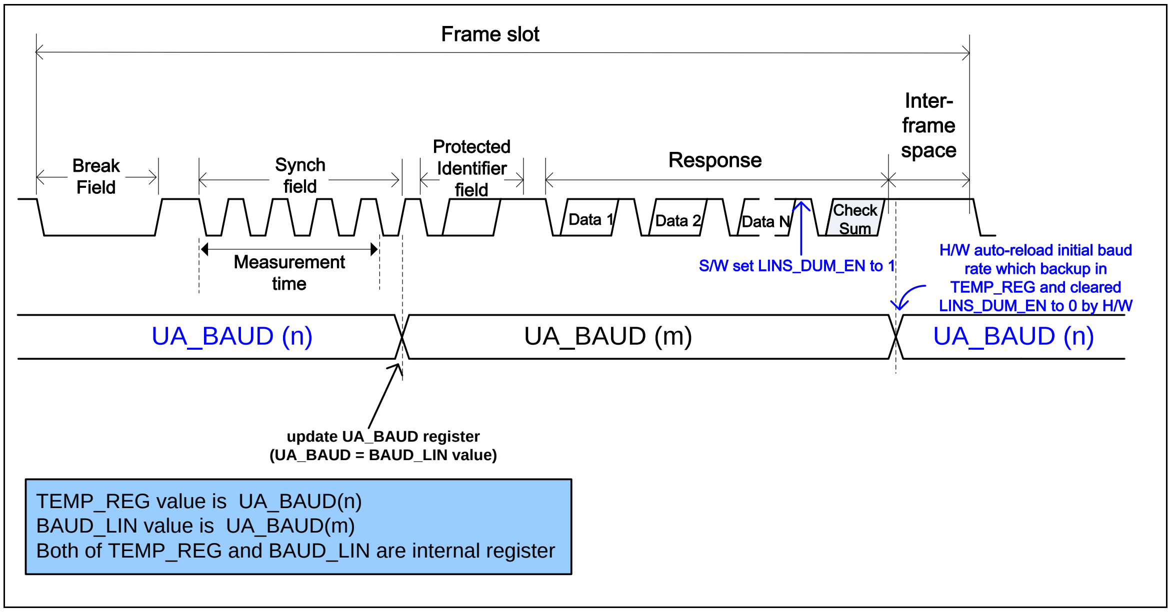


Figure 6-73 UA_BAUD Update Sequence in Automatic Resynchronization Mode when LINS_DUM_EN = 1

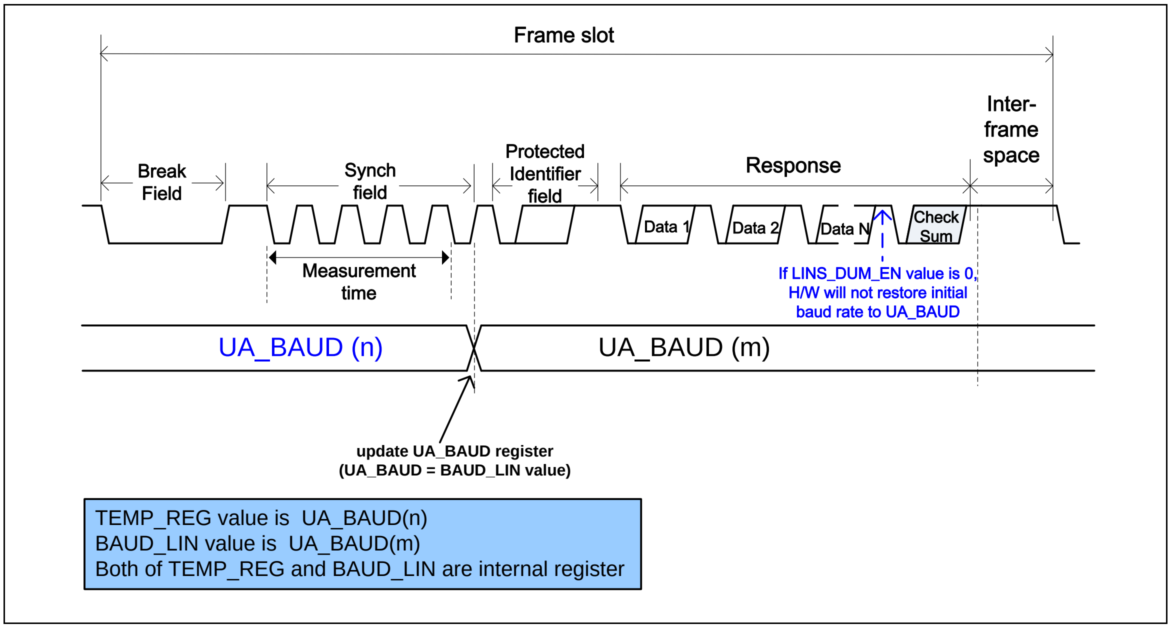


Figure 6-74 UA_BAUD Update Sequence in Automatic Resynchronization Mode when LINS_DUM_EN = 0

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 UA_LIN_SR[LINS_HERR_F]) will be set.
- If the difference is less than 14.06%, the header error flag UA_LIN_SR[LINS_HERR_F] will not be set.
- If the difference is between 14.84% and 14.06%, the header error flag UA_LIN_SR[LINS_HERR_F] may either set or not (depending on the data dephasing).

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 UA_LIN_SR[LINS_HERR_F] will be set.
- If the difference is less than 15.62%, the header error flag UA_LIN_SR[LINS_HERR_F] will not be set.
- If the difference is between 18.75% and 15.62%, the header error flag UA_LIN_SR[LINS_HERR_F] may either set or not (depending on the data dephasing).

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 UA_LIN_CTL [LINS_DUM_EN] register (It is recommended toset the LINS_DUM_EN 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 bit in UA_LIN_CTL register, hardware will handle the header detect flow. If the header has an error, the LIN header error flag (UA_LIN_SR [LIN_HERR_F]) will be set and an interrupt is generated if the UA_IER [LIN_IEN] 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 bit in UA_LIN_SR register to re-search a new frame header.

The LIN header error flag (UA_LIN_SR [LIN_HERR_F]) 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.

6.13.6 RS-485 Function Mode

The UART supports **RS-485 9-bit mode function**. The RS-485 mode is selected by setting the register UA_FUN_SEL[1:0]. The RS-485 transceiver control is implemented using the RTS control signal from an asynchronous serial port. In RS-485 mode, many characteristics of the RX and TX are same as UART.

In RS-485 mode, the controller can configure it 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 , EPE and SPE are set, the 9-th bit is transmitted 0 and when PBE and SPE are set and EPE is cleared, the 9-th bit is transmitted 1).

The controller supports three operation modes – RS-485 Normal Multidrop Operation Mode (NMM), RS-485 Auto Address Detection Operation Mode (AAD) and RS-485 Auto Direction Control Operation Mode (AUD). Software can choose any operation mode by programming UA_ALT_CSR register, and can drive the transfer delay time between the last stop bit leaving the TX-FIFO and the de-assertion of by setting UA_TOR [DLY] register.

Note: When RS485 NMM or AAD mode is selected, the RS485 clock operating frequency should be less than or equal to half of PCLK clock operation frequency. Otherwise, RS485 cannot receive correct data.

RS-485 Normal Multidrop Operation Mode (NMM)

In RS-485 Normal Multidrop Operation mode, first, software must decide which data before the address byte detected will be stored in RX-FIFO. If software wants to ignore any data before address byte detected, the flow is to set UA_FCR[RX_DIS] and then enable UA_ ALT_CSR [RS485_NMM] and the receiver will ignore any data until an address byte is detected (bit9 =1) and the address byte data will be stored in the RX-FIFO. If software wants to receive any data before address byte is detected, the flow disables UA_FCR[RX_DIS] and then enables UA_ ALT_CSR[RS485_NMM] and the receiver will received any data.

If an address byte is detected (bit9 =1), it will generate an interrupt to CPU and UA_FCR[RX_DIS] can decide whether accepting the following data bytes are stored in the RX-FIFO. If software disables the receiver by setting UA_FCR[RX_DIS] register, when a next address byte is detected, the controller will clear the UA_FCR[RX_DIS] bit and the address byte data will be stored in the RX-FIFO.

RS-485 Auto Address Detection Operation Mode (AAD)

In RS-485 Auto Address Detection Operation mode, the receiver will ignore any data until an address byte is detected (bit9 =1) and the address byte data matches the UA_ALT_CSR [ADDR_MATCH] value. The address byte data will be stored in the RX-FIFO. All the received byte data will be accepted and stored in the RX-FIFO until and address byte data not match the UA_ALT_CSR[ADDR_MATCH] value.

RS-485 Auto Direction Mode (AUD)

Another option function of RS-485 controllers is **RS-485 auto direction control function**. The RS-485 transceiver control is implemented 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 disable. User can set LEV_RTS in UA_MCR register to change the RTS driving level.

Program Sequence Example:

1. Program FUN_SEL in UA_FUN_SEL to select RS-485 function.
2. Program the RX_DIS bit in UA_FCR register to determine enable or disable the receiver of RS-485 controller.
3. Program the RS-485_NMM or RS-485_AAD mode.
4. If the RS-485_AAD mode is selected, the ADDR_MATCH is programmed for auto address match value.
5. Determine auto direction control by programming RS-485_AUD.

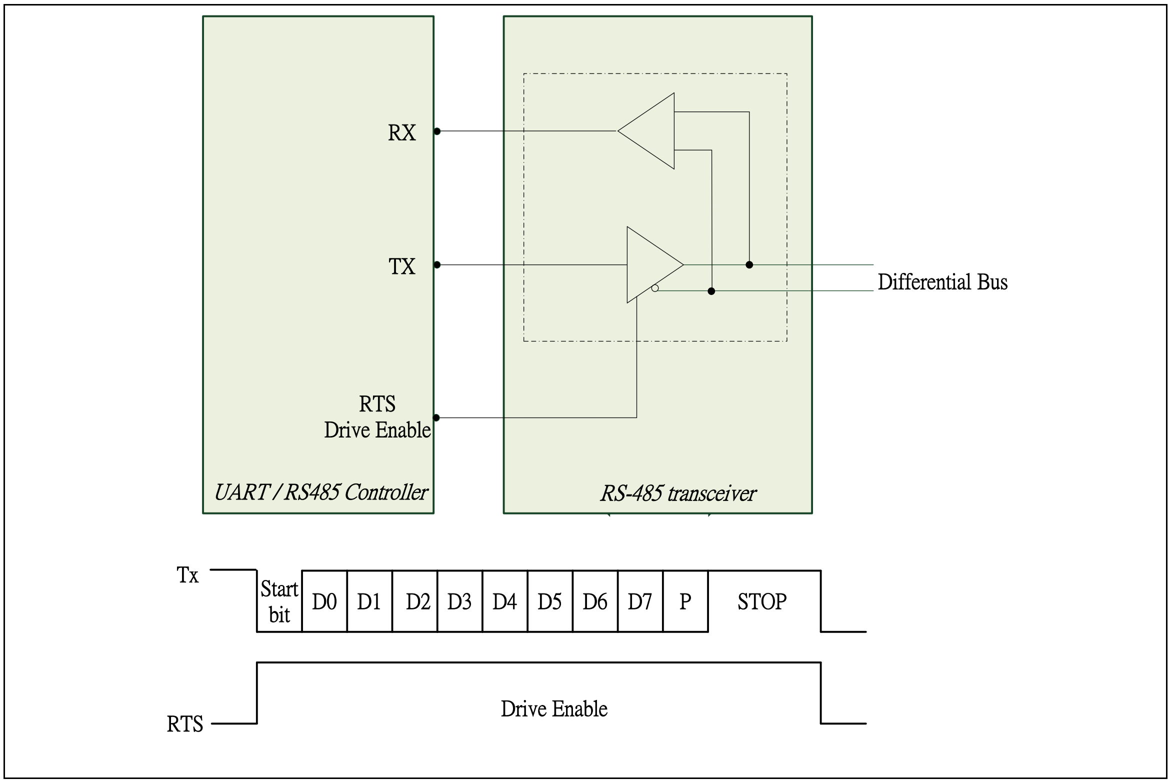


Figure 6-75 Structure of RS-485 Frame

6.13.7 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_0000
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

6.13.8 Register Description

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).

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 to this register, the UART will send out an 8-bit data through the TX pin (LSB first).

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 UA_FCR [RTS_TRI_LEV], 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 = Mask off INT_BUF_ERR Masked off. 1 = INT_BUF_ERR Enabled.
[4]	**TOUT_IEN**	**RX Time-out Interrupt Enable Bit** 0 = INT_TOUT Masked off. 1 = INT_TOUT Enabled.
[3]	**MODEM_IEN**	**Modem Status Interrupt Enable Bit (Not Available in UART2 Channel)** 0 = INT_MODEM Masked off. 1 = INT_MODEM Enabled.
[2]	**RLS_IEN**	**Receive Line Status Interrupt Enable Bit** 0 = INT_RLS Masked off. 1 = INT_RLS Enabled.
[1]	**THRE_IEN**	**Transmit Holding Register Empty Interrupt Enable Bit** 0 = INT_THRE Masked off. 1 = INT_THRE Enabled.
[0]	**RDA_IEN**	**Receive Data Available Interrupt Enable Bit** 0 = INT_RDA Masked off. 1 = INT_RDA Enabled.

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 = 1 byte. 0001 = 4 bytes. 0010 = 8 bytes. 0011 = 14 bytes. 0100 = 30/14 bytes (High Speed/Normal Speed). 0101 = 46/14 bytes (High Speed/Normal Speed). 0110 = 62/14 bytes (High Speed/Normal Speed). Others = 62/14 bytes (High Speed/Normal Speed). **Note:** This field is used for auto RTS flow control.
[15:9]	**Reserved**	Reserved.
[8]	**RX_DIS**	**Receiver Disable Register** 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 = 1 byte. 0001 = 4 bytes. 0010 = 8 bytes. 0011 = 14 bytes. 0100 = 30/14 bytes (High Speed/Normal Speed). 0101 = 46/14 bytes (High Speed/Normal Speed).

		0110 = 62/14 bytes (High Speed/Normal Speed). Others = 62/14 bytes (High Speed/Normal Speed).
[3]	**Reserved**	Reserved.
[2]	**TFR**	**TX Field Software Reset** When TX_RST 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 peripheral clock cycles.
[1]	**RFR**	**RX Field Software Reset** When RX_RST 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 peripheral clock cycles.
[0]	**Reserved**	Reserved.

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** This field sets UART word length. 00 = 5 bits. 01 = 6 bits. 10 = 7 bits. 11 = 8 bits.

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 is the output pin status of RTS.
[12:10]	**Reserved**	Reserved.
[9]	**LEV_RTS**	**RTS Trigger Level (Not Available in UART2 Channel)** This bit can change the RTS trigger level. 0= high level triggered. 1= low level triggered. **Note:** Refer to Figure 6-63 for UART function mode
[8:2]	**Reserved**	Reserved.
[1]	**RTS**	**RTS (Request-to-send) Signal (Not Available in UART2 Channel)** 0 = Drive RTS pin to logic 1 (If the LEV_RTS set to low level triggered). 1 = Drive RTS pin to logic 0 (If the LEV_RTS set to low level triggered). 0 = Drive RTS pin to logic 0 (If the LEV_RTS set to high level triggered). 1 = Drive RTS pin to logic 1 (If the LEV_RTS set to high level triggered).
[0]	**Reserved**	Reserved.

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 Trigger Level (Not Available in UART2 Channel)** This bit can change the CTS trigger level. 0 = High level triggered. 1 = Low level triggered.
[7:5]	**Reserved**	Reserved.
[4]	**CTS_ST**	**CTS Pin Status (Read Only) (Not Available in UART2 Channel)** This bit is the pin status of CTS.
[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 UA_IER [MODEM_IEN] is set to 1. **Note:** Write 1 to clear this bit to 0

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)** Bit is set by hardware when TX FIFO (UA_THR) is empty and the STOP bit of the last byte has been transmitted. 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. **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 is full or not. 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 is empty or not. 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. 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. 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. **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. **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. **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)** This bit sets to 1 while UA_ALT_CSR[RS485_ADD_EN] sets to 1 to enable Address detection mode and receive detect a data with an address bit(bit 9 =’1’). **Note:** This field is used for RS-485 function mode. **Note:** 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. **Note:** This bit is read only, but can be cleared by writing ‘1’ to it.

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 and HW_BUF_ERR_IF 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 and HW_TOUT_IF 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 and HW_MODEM_IF 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 and HW_RLS_IF 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 or RX_OVER_IF is set). When BUF_ERR_IF is set, the transfer maybe is not correct. If UA_IER [BUF_ERR_IEN] is enabled, the buffer error interrupt will be generated. **Note:** This bit is cleared when both TX_OVER_IF and RX_OVER_IF 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. If UA_IER [TOUT_IEN] is enabled, the Tout interrupt will be 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=1). If UA_IER [MODEM_IEN] is enabled, the Modem interrupt will be generated. **Note:** This bit is read only and reset to 0 when bit DCTSF is cleared by a write 1 on DCTSF.
[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, FEF and PEF, is set). If UA_IER [RLS_IEN] is enabled, the RLS interrupt will be generated. **Note:** In RS-485 function mode, this field include “receiver detect any address byte received address byte character (bit9 = ‘1’) bit". **Note:** This bit is read only and reset to 0 when all bits of BIF, FEF and PEF are cleared.
[17:16]	**Reserved**	Reserved.
[15]	**LIN_INT**	**LIN Bus Interrupt Indicator (Read Only)** This bit is set if LIN_IEN and LIN _IF 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 and BUF_ERR_IF 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 and TOUT_IF 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 and MODEM_IF 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 and RLS_IF 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 and THRE_IF 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 and RDA_IF 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=1), LIN break detect (LIN_BKDET_F=1), bit error detect (BIT_ERR_F=1), LIN slave ID parity error (LINS_IDPERR_F) or LIN slave header error detect (LINS_HERR_F) If UA_IER [LIN_IEN] is enabled the LIN interrupt will be generated. **Note:** This bit is cleared when LINS_HDET_F, LIN_BKDET_F, BIT_ERR_F, LINS_IDPENR_F and LINS_HERR_F all are cleared
[6]	**Reserved**	Reserved.
[5]	**BUF_ERR_IF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX or RX FIFO overflows (TX_OVER_IF or RX_OVER_IF is set). When BUF_ERR_IF is set, the transfer is not correct. If UA_IER [BUF_ERR_IEN] is enabled, the buffer error interrupt will be generated.
[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 UA_IER [TOUT_IEN] is enabled, the Tout interrupt will be 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=1). If UA_IER [MODEM_IEN] is enabled, the Modem interrupt will be generated. **Note:** This bit is read only and reset to 0 when bit DCTSF is cleared by a write 1 on DCTSF.
[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, FEF and PEF, is set). If UA_IER [RLS_IEN] is enabled, the RLS interrupt will be generated. **Note:** 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. **Note:** This bit is read only and reset to 0 when all bits of BIF, FEF and PEF 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 UA_IER [THRE_IEN] is enabled, the THRE interrupt will be 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 will be set. If UA_IER [RDA_IEN] is enabled, the RDA interrupt will be generated. **Note:** This bit is read only and it will be cleared when the number of unread bytes of RX FIFO drops below the threshold level (RFITL).

UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator To Interrupt Controller	Interrupt Flag	Flag Cleared By
Buffer Error Interrupt (INT_BUF_ERR)	BUF_ERR_IEN	HW_BUF_ERR_INT	**HW_BUF_ERR_IF** = (TX_OVER_IF or RX_OVER_IF)	Writing ‘1’ to UA_FCR [RFR]
RX Timeout Interrupt (INT_TOUT)	TOUT_IEN	HW_TOUT_INT	**HW_TOUT_IF**	Read UA_RBR
Modem Status Interrupt (INT_MODEM)	MODEM_IEN	HW_MODEM_INT	**HW_MODEM_IF** = (DCTSF)	Write ‘1’ to DCTSF

Receive Line Status Interrupt (INT_RLS)	RLS_IEN	HW_RLS_INT	**HW_RLS_IF** = (BIF or FEF or PEF or RS-485_ADD_DETF)	Writing '1' to UA_FCR [RFR]
Transmit Holding Register Empty Interrupt (INT_THRE)	THRE_IEN	HW_THRE_INT	**HW_THRE_IF**	Write UA_THR
Receive Data Available Interrupt (INT_RDA)	RDA_IEN	HW_RDA_INT	**HW_RDA_IF**	Read UA_RBR

Table 6-20 UART Interrupt Sources and Flags Table In DMA Mode

UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator To Interrupt Controller	Interrupt Flag	Flag Cleared By
LIN interrupt	LIN_IEN	LIN_RX_BREAK_INT	**LIN_RX_BREAK_IF**	Write '1' to LINS_HDET_F/LIN_BKDET_F/BIT_ERR_F/LINS_IDPERR_F/LINS_HERR_F
Buffer Error Interrupt (INT_BUF_ERR)	BUF_ERR_IEN	BUF_ERR_INT	**BUF_ERR_IF** = (TX_OVER_IF or RX_OVER_IF)	Write '1' to TX_OVER_IF/ RX_OVER_IF
RX Timeout Interrupt (INT_TOUT)	TOUT_IEN	TOUT_INT	**TOUT_IF**	Read UA_RBR
Modem Status Interrupt (INT_MODEM)	MODEM_IEN	MODEM_INT	**MODEM_IF** = (DCTSF)	Write '1' to DCTSF
Receive Line Status Interrupt (INT_RLS)	RLS_IEN	RLS_INT	**RLS_IF** = (BIF or FEF or PEF or RS-485_ADD_DETF)	Write '1' to BIF/FEF/PEF/ RS-485_ADD_DETF
Transmit Holding Register Empty Interrupt (INT_THRE)	THRE_IEN	THRE_INT	**THRE_IF**	Write UA_THR
Receive Data Available Interrupt (INT_RDA)	RDA_IEN	RDA_INT	**RDA_IF**	Read UA_RBR

Table 6-21 UART Interrupt Sources and Flags Table In Software Mode

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 (TOUT_CNT) is equal to that of time-out interrupt comparator (TOIC), a receiver time-out interrupt (INT_TOUT) is generated if UA_IER [TOUT_IEN] enabled. A new incoming data word or RX FIFO empty will clear INT_TOUT. In order to avoid receiver time-out interrupt generation immediately during one character is being received, TOIC value should be set between 40 and 255. So, for example, if TOIC is set with 40, the time-out interrupt is generated after four characters are not received when 1 stop bit and no parity check is set for UART transfer.

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 6-17 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 6-17 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

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**	**INV_RX** 0 = No inversion. 1 = Inverse RX input signal.
[5]	**INV_TX**	**INV_TX** 0 = No inversion. 1 = Inverse TX output signal.
[4:2]	**Reserved**	Reserved.
[1]	**TX_SELECT**	**TX_SELECT** 0 = IrDA receiver Enabled. 1 = IrDA transmitter Enabled.
[0]	**Reserved**	Reserved.

Note: In IrDA mode, the UA_BAUD [DIV_X_EN] 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_0000

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 field 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 in LIN function mode, the frame ID parity can be generated by software or hardware depends on UA_LIN_CTL [LIN_IDPEN]. If the parity generated by hardware (UA_LIN_CTL [LIN_IDPEN] = 1), user fill ID0~ID5, (LIN_PID[24:29]hardware will calculate P0(LIN_PID[30]) and P1(LIN_PID[31]), otherwise user must filled frame ID and parity in this field. **Note1:** User can filled 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 in Master mode for LIN to send header field (LIN_SHD = 1) or used to slave to indicate exit from Mute mode condition (LIN_MUTE_EN).
[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 UA_ALT_CSR [LIN_BKFL], User can read/write it by setting UA_ALT_CSR [LIN_BKFL] or UA_LIN_CTL [LIN_BKFL]. **Note2:** This break field length is LIN_BKFL + 1. **Note3:** According to LIN spec, the reset value is 0XC (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 UA_LIN_SR [BIT_ERR_F] flag will be asserted. If the UA_IER[LIN_IEN] = 1, an interrupt will be generated.
[11]	**LIN_RX_DIS**	If the receiver is be enabled (LIN_RX_DIS = 0), all received byte data will be accepted and stored in the RX-FIFO, and if the receiver is disabled (LIN_RX_DIS = 1), all received byte data will be ignore. 0 = Error detection function Disabled. 1 = Bit error detection Enabled. **Note:** This bit **is** only valid when operating in LIN function mode (UA_FUN_SEL[FUN_SEL] = 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 flag is set in UA_LIN_SR register at the end of break field. If the UA_IER [LIN_IEN] =1, an interrupt will be generated. 0 = Disable LIN break detection. 1 = Enable LIN break detection.
[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 = 1 and LIN_HEAD_SEL = 10) or be used for enable LIN slave received frame ID parity checked. **Note2:** This bit is only use when operation header transmitter is in LIN_HEAD_SEL = 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 register. 0 = Send LIN TX header Disable. 1 = Send LIN TX header Enable. **Note1:** These registers are shadow registers of UA_ALT_CSR [LIN_SHD]; user can read/write it by setting UA_ALT_CSR [LIN_SHD] or UA_LIN_CTL [LIN_SHD]. **Note2:** When transmitter header field (it may be “break” or “break + sync” or “break + sync + frame ID” selected by LIN_HEAD_SEL 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 character 6.13.5.4 (LIN Slave mode).
[3]	**LINS_DUM_EN**	**LIN Slave Divider Update Method Enable Bit** 0 = UA_BAUD is updated as soon as UA_BAUD is writing 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 in LIN Slave mode (LINS_EN = 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 character 6.13.5.4 (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 in LIN Slave mode (LINS_EN = 1). **Note2:** When operation in Automatic Resynchronization mode, the baud rate setting must be mode2 (UA_BAUD [DIV_X_EN] and UA_BAUD [DIV_X_ONE] must be 1). **Note3:** The control and interactions of this field are explained in character 6.13.5.4 (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 in LIN Slave mode (LINS_EN = 1). **Note2:** In LIN function mode, when detect header field (break + sync + frame ID), UA_LIN_SR [LINS_HDET_F] flag will be asserted. If the UA_IER[LIN_IEN] = 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 will be set. When occur bit error, if the UA_IER[LIN_IEN] = 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 (UA_LIN_CTL [BIT_ERR_EN] == 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 enable LIN break detection function (UA_ALT_CSR [LIN_BKDET_EN])
[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 a 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 is in LIN Slave mode (LINS_EN = 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 in LIN Slave mode (UA_LIN_CTL [LINS_EN] = 1) and enable LIN frame ID parity check function (UA_LIN_CTL [LIN_IDPEN]).
[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”, “frame error in sync field or Identifier field”, “sync field data is not 0x55 in Non-Automatic Resynchronization mode”, “sync field deviation error in Automatic Resynchronization mode”, “sync field measure time-out in 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 in LIN Slave mode (UA_LIN_CTL [LINS_EN] = 1) and enable LIN slave header detection function (UA_LIN_CTL [LINS_HDET_EN]).
[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 in LIN Slave mode (UA_LIN_CTL [LINS_EN] = 1) and enable LIN slave header detection function (UA_LIN_CTL [LINS_HDET_EN]). **Note3:** When enable ID parity check (UA_LIN_CTL [LIN_IDPEN] = 1), if hardware detect complete header (“break + sync + frame ID”), the LINS_HEDT_F will be set weather the frame ID correct or not.

6.14 Smart Card Host Interface (SC)

6.14.1 Overview

The Smart Card Interface controller (SC controller) is based on ISO/IEC 7816-3 standard and fully compliant with PC/SC Specifications. It also provides status of card insertion/removal.

6.14.2 Features

- ISO7816-3 T=0, T=1 compliant
- EMV2000 compliant
- Supports up to three ISO7816-3 ports
- Separates receive/ transmit 4 byte entry buffer for data payloads
- Programmable transmission clock frequency
- Programmable receiver buffer trigger level
- Programmable guard time selection (11 ETU ~ 266 ETU)
- One 24-bit and two 8-bit time-out counters for Answer to Request (ATR) and waiting times processing
- Supports auto inverse convention function
- Supports transmitter and receiver error retry and error retry number limitation function
- Supports hardware activation sequence process
- Supports hardware warm reset sequence process
- Supports hardware deactivation sequence process
- Supports hardware auto deactivation sequence when detecting the card removal

6.14.3 Block Diagram

The SC clock control and block diagram are shown as follows. The SC controller is completely asynchronous design with two clock domains, PCLK and peripheral clock. Note that PCLK frequency should be higher than the frequency of peripheral clock.

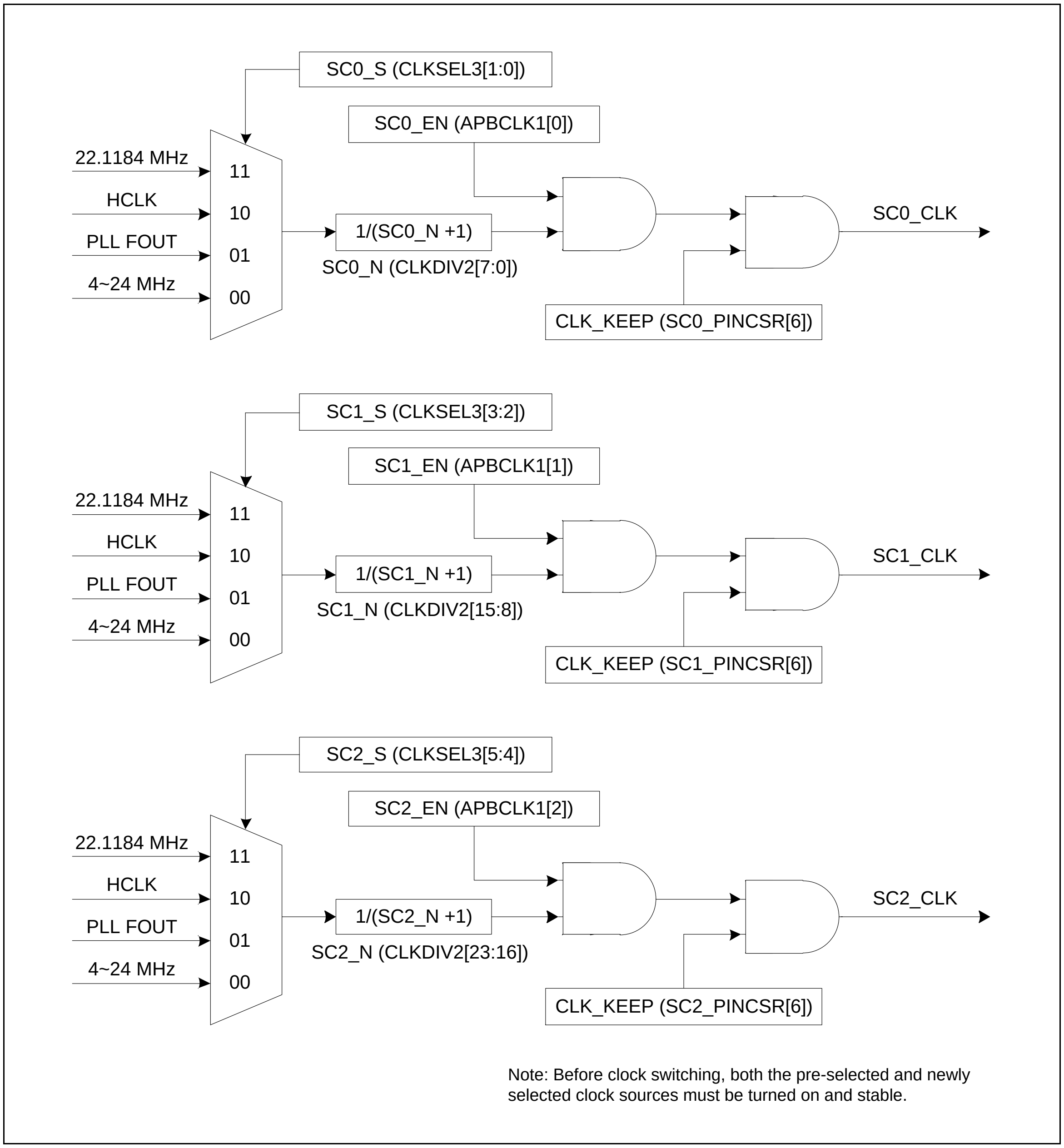


Figure 6-76 SC Clock Control Diagram (8-bit Prescale Counter in Clock Controller)

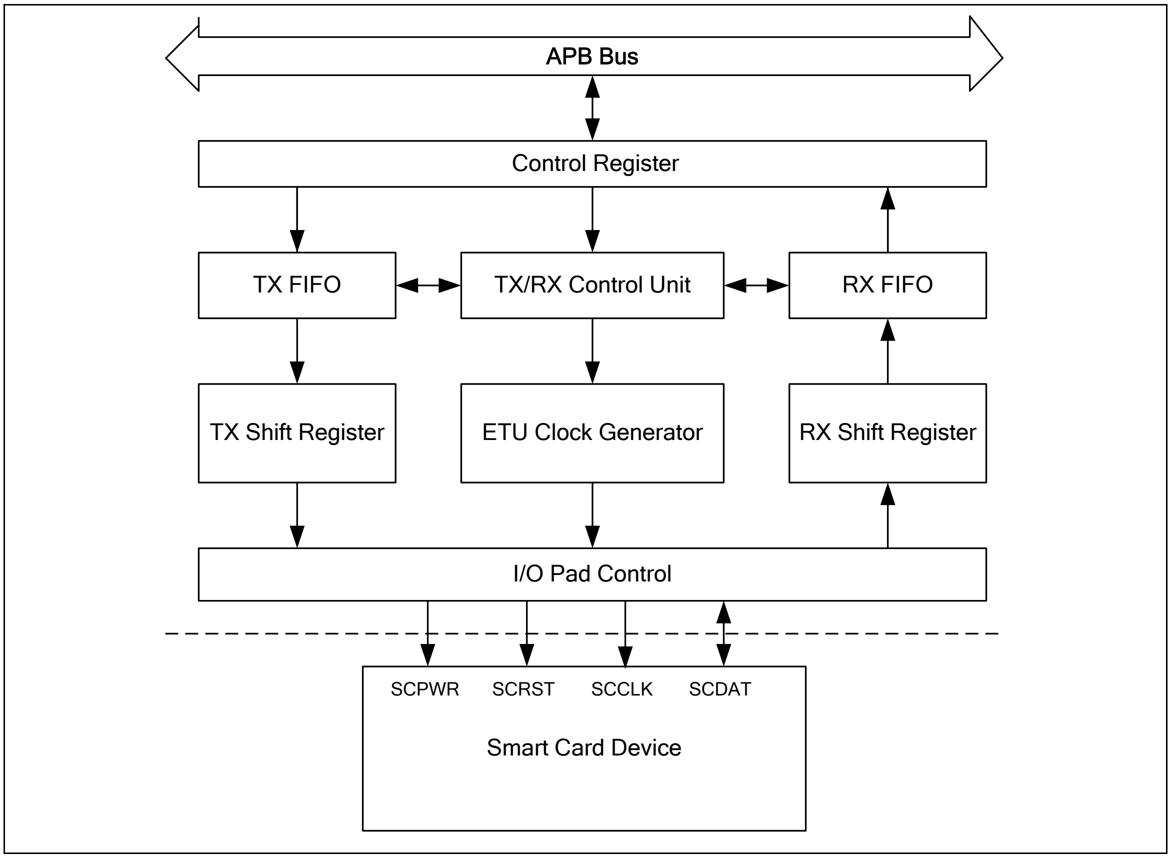


Figure 6-77 SC Controller Block Diagram

6.14.4 Functional Description

Basically, the smart card interface acts as a half-duplex asynchronous communication port and its data format is composed of ten consecutive bits, which is shown follows.

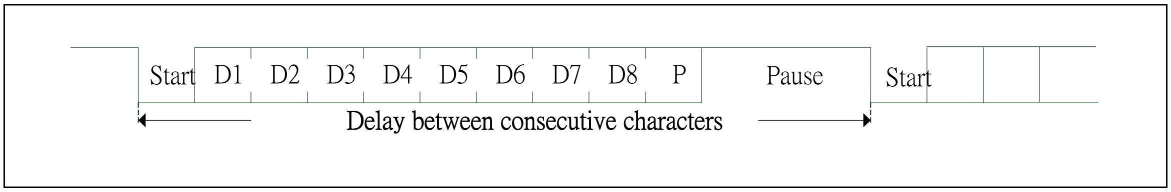


Figure 6-78 SC Data Character

6.14.4.1 Activation, Warm Reset and Deactivation Sequence

The Smart Card Interface controller supports hardware activation, warm reset and deactivation sequence.

Activation

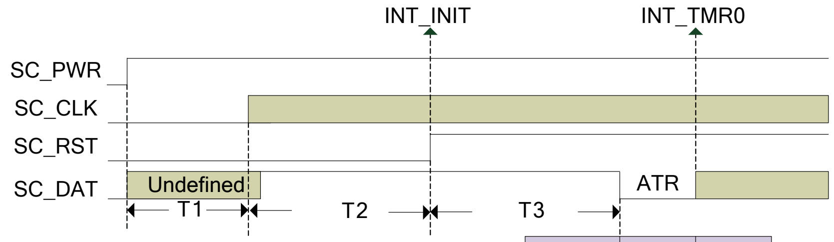
The activation sequence is shown as follows.

- Set SC_RST to low
- Set SC_PWR at high level and SC_DAT at high level (reception mode) at the same time.
- Enable SC_CLK clock
- De-assert SC_RST to high

The activation sequence can be controlled by software or hardware. If software wants to control it, software can control the SC_PINCSR and SC_TMRx register to process the activation sequence or setting SC_ALTCTL[ACT_EN] register, and then the interface will perform hardware activation sequence.

Following is activation control sequence in hardware activation mode:

- Set activation timing by setting SC_ALTCTL[INIT_SEL].
- TMR0 can be selected when SC_CTL[TMR_SEL] is 01, 10 or 11.
- Set operation mode SC_TMR0[MODE] to 0011 and give an Answer to Request value by setting SC_TMR0[CNT] register.
- When hardware de-asserts SC_RST to high, hardware will generator an initial end interrupt to CPU at the same time (if SC_IER[INIT_IE] = 1)
- If the TMR0 decreases the counter to “1” (start from SC_RST) and the card does not respond ATR before that time, hardware will generate interrupt INT_TMR0 to CPU.



Time	Comment
T1	SC_PWR to SC_CLK Start
T2	SC_CLK Start to SC_RST Assert
T3	SC_CLK Start to ART Appear

INIT_SEL	T1	T2
00	85	489
01	133	537
10	165	569
11	165	42060

Unit : SC Clock

Note : These value are measured by chip IO pin, the real value will depended on system design

Figure 6-79 SC Activation Sequence

Warm Reset

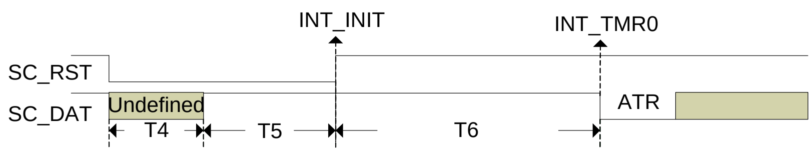
The warm reset sequence is shown as follows.

- Set SC_RST to low and set SC_DAT to high at the same time.
- Set SC_RST to high.

The warm reset sequence can be controlled by software or hardware. If software wants to control it, software can control SC_PINCSR and SC_TMRx register to process the warm reset sequence or set SC_ALTCTL[WARST_EN] register, and then the interface will perform hardware warm reset sequence.

Following is warm reset control sequence in hardware warm reset mode

- Set warm reset timing by setting SC_ALTCTL[INIT_SEL].
- Select TMR0 by setting SC_CTL[TMR_SEL] register (TMR_SEL can be 01, 10, or 11).
- Set operation mode SC_TMR0[MODE] to 0011 and give an Answer to Request value by setting SC_TMR0[CNT] register.
- Set TMR0_SEN and WARST_EN to start counting by SC_ALTCTL register.
- When hardware de-asserts SM_RST to high, hardware will generate an initial end interrupt to CPU at the same time (if SC_IER[INIT_IE] = “1”)
- If the TMR0 decrease the counter to “1” (start from SC_RST) and the card does not response ATR before that time, hardware will generate interrupt INT_TMR0 to CPU.



Time	Comment
T4	SC_RST to SC_DAT Reception Mode
T5	SC_DAT Reception Mode to SC_RST Assert
T6	SC_CLK Start to ATR Appear

INIT_SEL	T4	T5
00	81	483
01	129	531
10	161	563
11	161	42106

Unit:SC Clock

Note: This value is measured by chip IO pin, the real value will depend on system design.

Figure 6-80 SC Warm Reset Sequence

Deactivation

The deactivation sequence is shown as follows

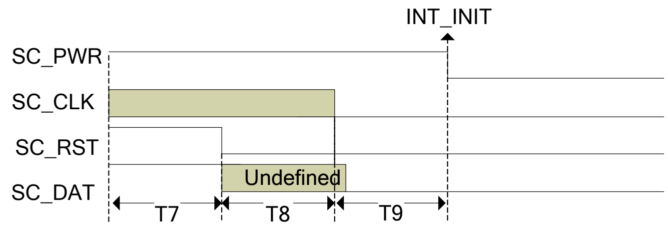
- Set SC_RST to low.
- Stop SC_CLK.
- Set SC_DAT to state low.
- Deactivated SC_PWR.

The deactivation sequence can be controlled by software or hardware. If software wants to control it, software can control SC_PINCSR and SC_TMR0 register to process the deactivation sequence or set SC_ALTCTL[DACT_EN] register, and then the interface will perform hardware deactivation sequence.

The SC controller also supports auto deactivation sequence when the card removal detection is set (SC_PINCSR[ADAC_CDEN]).

Following is deactivation control sequence in hardware deactivation mode:

- Set deactivation timing by setting SC_ALTCTL[INIT_SEL].
- Set DACT_EN to start counting by SC_ALTCTL register.
- When hardware de-asserts SC_PWR to low, the controller will generate an interrupt INT_INIT to CPU at the same time (if SC_IER[INIT_IE] = 1)



Time	Comment
T7	Deactivation Trigger to SC_RST Low
T8	SMC_RST Low to Stop SC_CLK
T9	Stop SC_CLK to Stop SC_PWR

INIT_SEL	T7	T8	T9
00	97	83	87
01	145	131	135
10	177	163	167
11	177	163	167

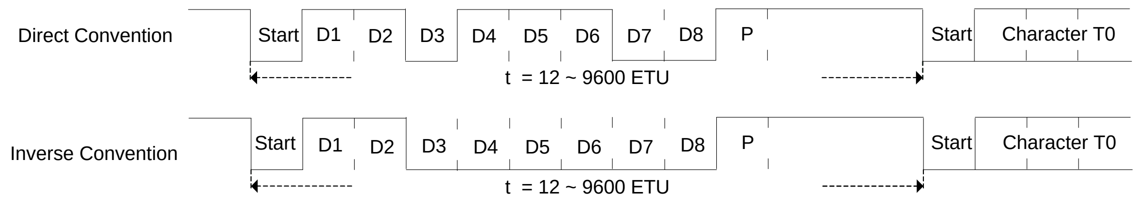
Unit: SC Clock

Figure 6-81 SC Deactivation Sequence

6.14.4.2 *Initial Character TS*

According to ISO7816-3, the initial character TS of answer to request (ATR) has two possible patterns (as shown in the following figure). If the TS pattern is 0_1100_0000_1, it is inverse convention. When decoded by inverse convention, the conveyed byte is equal to ‘3F’. If the TS pattern is 0_1101_1100_1, it is direct convention. When decoded by direct convention, the conveyed byte is equal to ‘3B’. Software can set SC_CTL[AUTO_CON_EN] and then the operating convention will be decided by hardware. Software can also set the SC_CTL[CON_SEL] register (set to 00 or 11) to change the operating convention after SC received TS of answer to request (ATR).

If software enables auto convention function by setting SC_CTL[AUTO_CON_EN] register, the setting step must be done before Answer to Request state and the first data must be 0x3B or 0x3F. After hardware received first data and stored it at buffer, hardware will decide the convention and change the SC_CTL[CON_SEL] register automatically. If the first data is neither 0x3B nor 0x3F, hardware will generate an auto-convention error interrupt INT_ACON_ERR (if SC_IER[ACON_ERR_IE] = 1) to CPU.



Direct Convention	0_1101_1100_1 (0x3B)
Inverse Convention	0_1100_0000_1 (0x3F)

Figure 6-82 Initial Character TS

6.14.4.3 *Error signal and character repetition*

According to ISO7816-3 T=0 mode description, as shown in following, if the receiver receives a wrong parity bit, it will pull the SC_DAT to low one to two bit period to inform the transmitter parity error. Then the transmitter will retransmit the character. The SC interface controller supports hardware error detection function in receiver and supports hardware re-transmit function in transmitter. Software can enable re-transmit function by setting SC_CTL[TX_ERETRY_EN]. Software can also define the retry (re-transmit) number limitation in SC_CTL[TX_ERETRY] register. The re-transmit number is up to TX_ERETRY +1 and if the re-transmit number is equal to TX_ERETRY +1, TX_OVER_REERR flag will be set by hardware and if SC_IER[TERR_IE] = 1, SC controller will generate a transfer error interrupt to CPU. Software can also define the received retry number limitation in SC_CTL[RX_ERETRY] register. The receiver retry number is up to RX_ERETRY +1, if the number of received errors by receiver is equal to RX_ERETRY +1, receiver will receive this error data to buffer and RX_OVER_REERR flag will be set by hardware and if SC_IER[TERR_IE] = 1, SC controller will generate a transfer error interrupt to CPU.

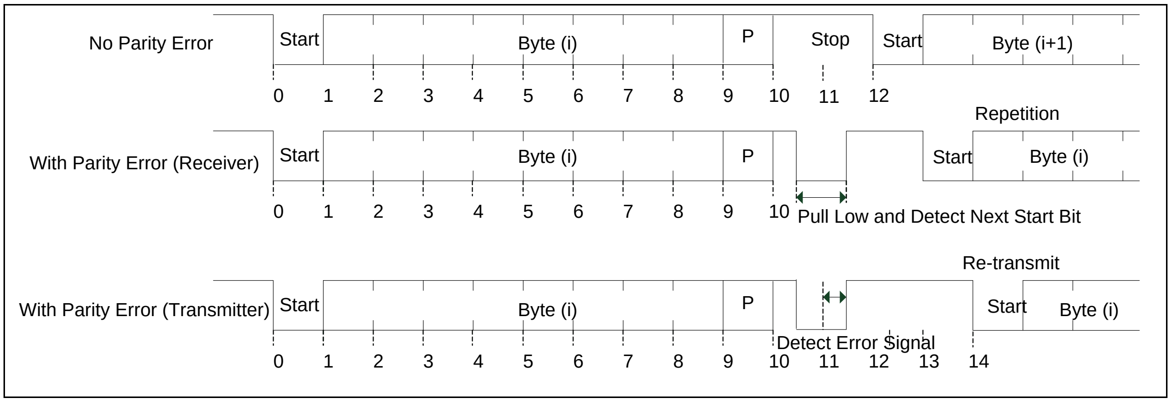


Figure 6-83 SC Error Signal

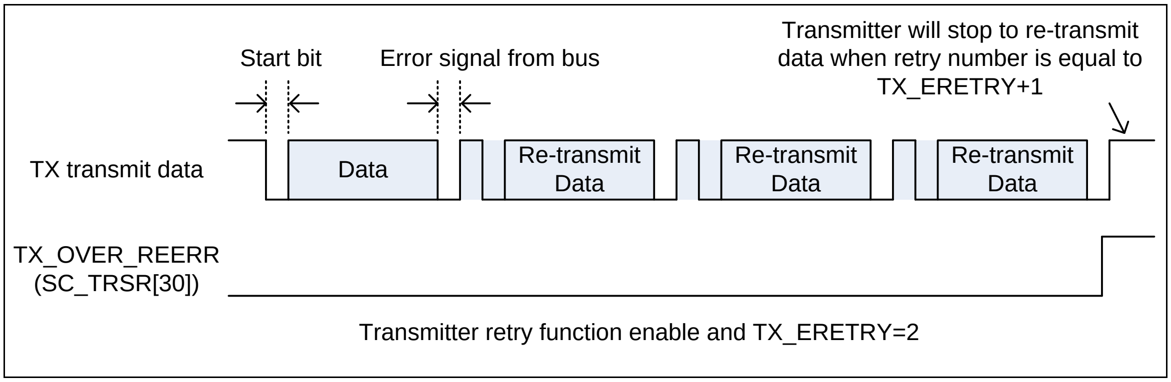


Figure 6-84 SC Transmitter Retry Number and Retry Over Flag

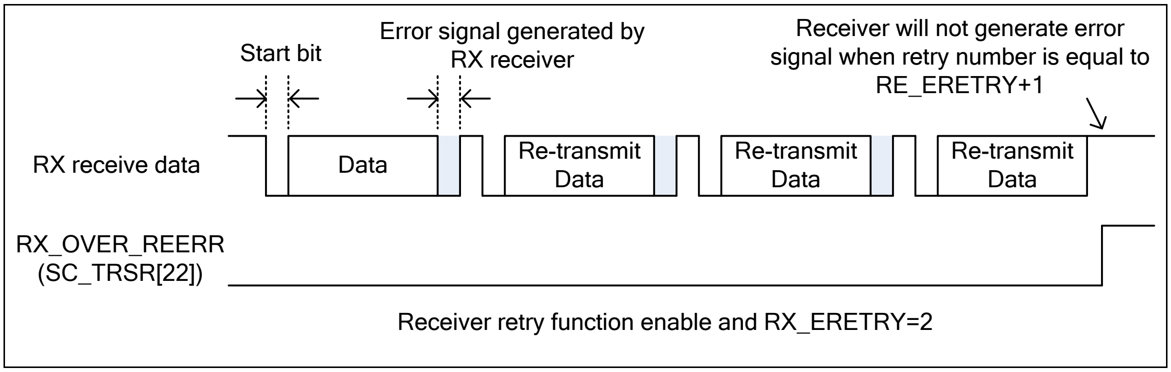


Figure 6-85 SC Receiver Retry Number and Retry Over Flag

6.14.4.4 Internal time-out counter

The smart card interface includes a 24-bit time-out counter and two 8 bit time-out counters. These counters help the controller in processing different real-time interval (ATR, WWT, BWT, etc.). Each counter can be set to start counting once the trigger enable bit has been written or a START bit has been detected.

The following is the programming flow:

- Enable counter by setting SC_CTL[TMR_SEL].
- Select operation mode SC_TMRx[MODE] and give a count value SC_TMRx[CNT] by setting SC_TMRx register.
- Set SC_ALTCTL[TMRx_SEN] to start counting.

OPMODE(SC_ TMRx[27:24]) (X=0 ~2)	Operation Description	
0000	The down counter started when SC_ALTCTL[TMRx_SEN] enabled and ended when counter time-out. The time-out value will be CNT+1	
	Start	Start counting when SC_ALTCTL [TMRx_SEN] enabled
	End	When the down counter equals to 0, hardware will set TMRx_IS and clear SC_ALTCTL [TMRx_SEN] automatically.
0001	The down counter started when the first START bit (reception or transmission) detected and ended when counter time-out. The time-out value will be CNT+1.	
	Start	Start counting when the first START bit (reception or transmission) detected after SC_ALTCTL [TMRx_SEN] set to 1.
	End	When the down counter equals to 0, hardware will set TMR0_IS and clear SC_ALTCTL [TMRx_SEN] automatically.
0010	The down counter started when the first START bit (reception) detected and ended when counter time-out . The time-out value will be CNT+1.	
	Start	Start counting when the first START bit (reception) detected bit after SC_ALTCTL[TMRx_SEN] set to 1.
	End	When the down counter equals to 0, hardware will set TMR0_IS and clear SC_ALTCTL [TMRx_SEN] automatically.
0011	The down counter is only used for hardware activation, warm reset sequence to measure ATR timing. The timing starts when SC_RST de-assertion and ends when ATR response received or time-out. If the counter decreases to 0 before ATR response received, hardware will generate an interrupt to CPU. The time-out value will be CNT+1.	
	Start	Start counting when SC_RST de-assertion after SC_ALTCTL[TMRx_SEN] set to 1. It is used for hardware activation, warm reset mode.
	End	When the down counter equals to 0 before ATR response received, hardware will set TMRx_IS and clear SC_ALTCTL [TMRx_SEN] automatically. When ATR received and down counter does not equal to 0, hardware will clear SC_ALTCTL [TMRx_SEN] automatically.
0100	Same as 0000, but when the down counter equals to 0, hardware will set TMRx_IS and counter will re-load the SC_TMRx [CNT] value and re-count until software clears SC_ALTCTL [TMRx_SEN]. When SC_ALTCTL[TMRx_ATV] =1, software can change SC_TMRx [CNT] value at any time. When the down counter equals to 0, counter will reload the new value of SC_TMRx [CNT] and re-count.	

<table>
<tr><td></td><td colspan="2">The time-out value will be CNT+1.</td></tr>
<tr><td>0101</td><td colspan="2">Same as 0001, but when the down counter equals to 0, hardware will set TMRx_IS and counter will re-load the SC_TMRx [CNT] value. When the next START bit is detected, counter will re-count until software clears SC_ALTCTL[TMRx_SEN].
When SC_ALTCTL [TMRx_ATV] =1 software can change SC_TMRx [CNT] value at any time. When the down counter equal to 0, it will reload the new value of SC_TMRx [CNT] and re-counting.
The time-out value will be CNT+1.</td></tr>
<tr><td>0110</td><td colspan="2">Same as 0010, but when the down counter equals to 0, it will set TMRx_IS and counter will re-load the SC_TMRx [CNT] value. When the next START bit is detected, counter will re-count until software clears SC_ALTCTL [TMRx_SEN].
When SC_ALTCTL[TMRx_ATV] =1, software can change SC_TMRx [CNT] value at any time. When the down counter equals to 0, counter will reload the new value of SC_TMRx [CNT] and re-count.
The time-out value will be CNT+1.</td></tr>
<tr><td rowspan="3">0111</td><td colspan="2">The down counter started when the first START bit (reception or transmission) detected and ended when software clears SC_ALTCTL [TMRx_SEN] bit. If next START bit detected, counter will reload the new value of SC_TMRx [CNT] and re-counting.
If the counter decreases to 0 before the next START bit detected, hardware will generate an interrupt to CPU. The time-out value will be CNT+1.</td></tr>
<tr><td>Start</td><td>Start counting when the first START bit detected after SC_ALTCTL [TMRx_SEN] set to 1.</td></tr>
<tr><td>End</td><td>Stop counting after SC_ALTCTL [TMRx_SEN] set to 0.</td></tr>
<tr><td rowspan="3">1000</td><td colspan="2">The up counter starts when SC_ALTCTL[TMRx_SEN] enabled and ends when SC_ALTCTL [TMRx_SEN] disabled. This count value will be stored in SC_TDRA [23:0]. In this mode, hardware cannot generate any interrupt to CPU. The real count value will be SC_TDRA [23:0] +1.</td></tr>
<tr><td>Start</td><td>Start counting after SC_ALTCTL [TMRx_SEN] set to 1, and the start count value is 0 (hardware will ignore CNT value).</td></tr>
<tr><td>End</td><td>Stop counting after SC_ALTCTL[TMRx_SEN] set to 0 and the value stored to SC_TDRA [23:0] register.</td></tr>
</table>

Table 6-22 Timer2/Timer1/Timer0 Operation Mode

6.14.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
SC Base Address: **SCx_BA = 0x4019_0000 + (0x4000 * x)** **x = 0, 1, 2**				
SC_RBR	SCx_BA+0x00	R	SC Receiving Buffer Register.	Undefined
SC_THR	SCx_BA+0x00	W	SC Transmit Holding Register	Undefined
SC_CTL	SCx_BA+0x04	R/W	SC Control Register	0x0000_0000
SC_ALTCTL	SCx_BA+0x08	R/W	SC Alternate Control State Register	0x0000_0000
SC_EGTR	SCx_BA+0x0C	R/W	SC Extend Guard Time Register	0x0000_0000
SC_RFTMR	SCx_BA+0x10	R/W	SC Receiver buffer Time-out Register	0x0000_0000
SC_ETUCR	SCx_BA+0x14	R/W	SC ETU Control Register	0x0000_0173
SC_IER	SCx_BA+0x18	R/W	SC Interrupt Enable Register	0x0000_0000
SC_ISR	SCx_BA+0x1C	R/W	SC Interrupt Status Register	0x0000_0002
SC_TRSR	SCx_BA+0x20	R/W	SC Transfer Status Register	0x0000_0202
SC_PINCSR	SCx_BA+0x24	R/W	SC Pin Control State Register	0x000x_00x0
SC_TMR0	SCx_BA+0x28	R/W	SC Internal Timer Control Register 0	0x0000_0000
SC_TMR1	SCx_BA+0x2C	R/W	SC Internal Timer Control Register 1	0x0000_0000
SC_TMR2	SCx_BA+0x30	R/W	SC Internal Timer Control Register 2	0x0000_0000
SC_TDRA	SCx_BA+0x38	R	SC Timer Current Data Register A	0x0000_07FF
SC_TDRB	SCx_BA+0x3C	R	SC Timer Current Data Register B	0x0000_7F7F

Note: Post-fix "x" of SCx represents the SC channel.

6.14.6 Register Description

SC Receiving Buffer Register (SC_RBR)

Register	Offset	R/W	Description	Reset Value
SC_RBR	SCx_BA+0x00	R	SC Receiving 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** By reading this register, the SC will return an 8-bit received data.

SC Transmit Holding Register (SC_THR)

Register	Offset	R/W	Description	Reset Value
SC_THR	SCx_BA+0x00	W	SC 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 to this register, the SC will send out an 8-bit data. **Note:** If SC_CTL[SC_CEN] not enabled, this register cannot be programmed.

SC Control Register (SC_CTL)

Register	Offset	R/W	Description	Reset Value
SC_CTL	SCx_BA+0x04	R/W	SC Control Register	0x0000_0000

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="6">Reserved</td><td colspan="2">CD_DEB_SEL</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td>TX_ERETRY_EN</td><td colspan="3">TX_ERETRY</td><td>RX_ERETRY_EN</td><td colspan="3">RX_ERETRY</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td>SLEN</td><td colspan="2">TMR_SEL</td><td colspan="5">BGT</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">RX_FTRI_LEV</td><td colspan="2">CON_SEL</td><td>AUTO_CON_EN</td><td>DIS_TX</td><td>DIS_RX</td><td>SC_CEN</td></tr>
</table>

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:26]</td><td>Reserved</td><td>Reserved.</td></tr>
<tr><td>[25:24]</td><td>CD_DEB_SEL</td><td>Card Detect De-bounce Selection
This field indicates the card detect de-bounce selection.
00 = De-bounce sample card insert once per 384 (128 * 3) peripheral clocks and de-bounce sample card removal once per 128 peripheral clocks.
01 = De-bounce sample card insert once per 192 (64 * 3) peripheral clocks and de-bounce sample card removal once per 64 peripheral clocks.
10 = De-bounce sample card insert once per 96 (32 * 3) peripheral clocks and de-bounce sample card removal once per 32 peripheral clocks.
11 = De-bounce sample card insert once per 48 (16 * 3) peripheral clocks and de-bounce sample card removal once per 16 peripheral clocks.</td></tr>
<tr><td>[23]</td><td>TX_ERETRY_EN</td><td>TX Error Retry Enable Bit
This bit enables transmitter retry function when parity error has occurred.
0 = TX error retry function Disabled.
1 = TX error retry function Enabled.
Note: Software must fill in TX_ERETRY value before enabling this bit.</td></tr>
<tr><td>[22:20]</td><td>TX_ERETRY</td><td>TX Error Retry Count
This field indicates the maximum number of transmitter retries that are allowed when parity error has occurred.
Note1: The real retry number is TX_ERETRY + 1, and 8 is the maximum retry number.
Note2: This field cannot be changed when TX_ERETRY_EN is enabled. The change flow is to disable TX_ETRTRY_EN first and then fill in a new retry value.</td></tr>
<tr><td>[19]</td><td>RX_ERETRY_EN</td><td>RX Error Retry Enable Bit
This bit enables receiver retry function when parity error has occurred.
0 = RX error retry function Disabled.
1 = RX error retry function Enabled.
Note: Software must fill in RX_ERETRY value before enabling this bit.</td></tr>
</table>

[18:16]	**RX_ERETRY**	**RX Error Retry Count** This field indicates the maximum number of receiver retries that are allowed when parity error has occurred. **Note1:** The real maximum retry number is RX_ERETRY + 1, so 8 is the maximum retry number. **Note2:** This field cannot be changed when RX_ERETRY_EN enabled. The change flow is to disable RX_ETRTRY_EN first and then fill new retry value.
[15]	**SLEN**	**Stop Bit Length** This field indicates the length of stop bit. 0 = The stop bit length is 2 ETU. 1 = The stop bit length is 1 ETU. **Note:** The default stop bit length is 2.
[14:13]	**TMR_SEL**	**Timer Selection** 00 = All internal timer functions Disabled. 01 = Internal 24-bit timer Enabled. Software can configure it by setting SC_TMR0 [23:0]. SC_TMR1 and SC_TMR2 will be ignored in this mode. 10 = Internal 24-bit timer and 8-bit internal timer Enabled. Software can configure the 24-bit timer by setting SC_TMR0 [23:0] and configure the 8-bit timer by setting SC_TMR1[7:0]. SC_TMR2 will be ignored in this mode. 11 = Internal 24-bit timer and two 8-bit timers Enabled. Software can configure them by setting SC_TMR0 [23:0], SC_TMR1 [7:0] and SC_TMR2 [7:0].
[12:8]	**BGT**	**Block Guard Time (BGT)** Block guard time means the minimum bit length between the leading edges of two consecutive characters between different transfer directions. This field indicates the counter for the bit length of block guard time. According to ISO7816-3, in T = 0 mode, software must fill 15 (real block guard time = 16.5) to this field; in T = 1 mode, software must fill 21 (real block guard time = 22.5) to it. **Note:** The real block guard time is BGT + 1.
[7:6]	**RX_FTRI_LEV**	**Rx Buffer Trigger Level** When the number of bytes in the receiving buffer equals the RX_FTRI_LEV, the RDA_IF will be set (if IER [RDA_IEN] is enabled, an interrupt will be generated). 00 = INTR_RDA Trigger Level with 1 Byte. 01 = INTR_RDA Trigger Level with 2 Bytes. 10 = INTR_RDA Trigger Level with 3 Bytes. 11 = Reserved.
[5:4]	**CON_SEL**	**Convention Selection** 00 = Direct convention. 01 = Reserved. 10 = Reserved. 11 = Inverse convention. **Note:** If AUTO_CON_EN enabled, this fields must be ignored.
[3]	**AUTO_CON_EN**	**Auto Convention Enable Bit** 0 = Auto-convention Disabled. 1 = Auto-convention Enabled. If auto-convention bit is enabled, when hardware receives TS in answer to reset state and the TS is direct convention, CON_SEL will be set to 00 automatically, otherwise if the TS is inverse convention, and CON_SEL will be set to 11. If software enables auto convention function, the setting step must be done before Answer to Request state and the first data must be 3B or 3F. After hardware received first data and stored it at buffer, hardware will decided the convention and change the SC_CTL[CON_SEL] register automatically. If the first data is not 0x3B or 0x3F, hardware

		will generate an auto-convention error interrupt (if SC_IER [ACON_ERR_IE = 1] to CPU.
[2]	**DIS_TX**	**TX Transition Disable Bit** 0 = Transceiver Enabled. 1 = Transceiver Disabled.
[1]	**DIS_RX**	**RX Transition Disable Bit** 0 = Receiver Enabled. 1 = Receiver Disabled. **Note:** If AUTO_CON_EN enabled, this fields must be ignored.
[0]	**SC_CEN**	**SC Engine Enable Bit** Setting this bit to "1" will enable SC operation. If this bit is cleared, SC will force all transition to IDLE state.

SC Alternate Control Register (SC_ALTCTL)

Register	Offset	R/W	Description	Reset Value
SC_ALTCTL	SCx_BA+0x08	R/W	SC Alternate Control State Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TMR2_ATV	TMR1_ATV	TMR0_ATV	RX_BGT_EN	Reserved		INIT_SEL	
7	6	5	4	3	2	1	0
TMR2_SEN	TMR1_SEN	TMR0_SEN	WARST_EN	ACT_EN	DACT_EN	RX_RST	TX_RST

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**TMR2_ATV**	**Internal Timer2 Active State (Read Only)** This bit indicates the Timer2 counter status 0 = Timer2 is not active. 1 = Timer2 is active.
[14]	**TMR1_ATV**	**Internal Timer1 Active State (Read Only)** This bit indicates the Timer1 counter status 0 = Timer1 is not active. 1 = Timer1 is active.
[13]	**TMR0_ATV**	**Internal Timer0 Active State (Read Only)** This bit indicates the Timer0 counter status 0 = Timer0 is not active. 1 = Timer0 is active.
[12]	**RX_BGT_EN**	**Check Receiver Block Guard Time Function Enable** 0 = Check receiver block guard time function Disabled. 1 = Check receiver block guard time function Enabled.
[11:10]	**Reserved**	Reserved.
[9:8]	**INIT_SEL**	**Initial Timing Selection** This fields indicates the timing of hardware initial state (activation or warm-reset or deactivation). Unit: SC clock Activation: refer to Figure 6-79. Warm-reset: refer to Figure 6-80. Deactivation: refer to Figure 6-81.
[7]	**TMR2_SEN**	**Internal Timer2 Start Enable Bit** This bit enables Timer2 to start counting. Software can fill “0” to stop it and set “1” to

		reload and count. 0 = Stops counting. 1 = Starts counting. **Note1:** This field is used for internal 8 bit timer when SC_CTL [TMR_SEL] = 11. Don't filled TMR2_SEN when SC_CTL[TMR_SEL] = 00 or 01 or 10. **Note2:** If the operation mode is not in auto-reload mode (SC_TMR2[26] = "0"), this bit will be auto-cleared by hardware. **Note3:** This field will be cleared when software set SC_ALTCTL[TX_RST] or SC_ALTCTL [RX_RST] to 1. So don't fill this bit, TX_RST, and RX_RST at the same time. **Note4:** If SC_CTL [SC_CEN] not enabled, this field cannot be programmed.
[6]	**TMR1_SEN**	**Internal Timer1 Start Enable Bit** This bit enables Timer1 to start counting. Software can fill "0" to stop it and set "1" to reload and count. 0 = Stops counting. 1 = Starts counting. **Note1:** This field is used for internal 8 bit timer when SC_CTL[TMR_SEL] = 01 or 10. Don't filled TMR1_SEN when SC_CTL[TMR_SEL] = 00 or 11. **Note2:** If the operation mode is not in auto-reload mode (SC_TMR1[26] = "0"), this bit will be auto-cleared by hardware. **Note3:** This field will be cleared when software set SC_ALTCTL[TX_RST] or SC_ALTCTL[RX_RST] to 1, so don't fill this bit, TX_RST, and RX_RST at the same time. **Note4:** If SC_CTL [SC_CEN] not enabled, this field cannot be programmed.
[5]	**TMR0_SEN**	**Internal Timer0 Start Enable Bit** This bit enables Timer0 to start counting. Software can fill "0" to stop it and set "1" to reload and count. 0 = Stops counting. 1 = Starts counting. **Note1:** This field is used for internal 24 bit timer when SC_CTL[TMR_SEL] = 01. **Note2:** If the operation mode is not in auto-reload mode (SC_TMR0[26] = "0"), this bit will be auto-cleared by hardware. **Note3:** This field will be cleared when software set SC_ALTCTL[TX_RST] or SC_ALTCTL[RX_RST] to 1. So don't fill this bit, TX_RST and RX_RST at the same time. **Note4:** If SC_CTL [SC_CEN] not enabled, this field cannot be programmed.
[4]	**WARST_EN**	**Warm Reset Sequence Generator Enable Bit** This bit enables SC controller to initiate the card by warm reset sequence 0 = No effect. 1 = Warm reset sequence generator Enabled. **Note1:** When the warm reset sequence completed, this bit will be cleared automatically and the SC_ISR[INIT_IS] will be set to "1". **Note2:** This field will be cleared when software set SC_ALTCTL[TX_RST] or SC_ALTCTL[RX_RST] to 1. So don't fill this bit, TX_RST, and RX_RST at the same time. **Note3:** If SC_CTL[SC_CEN] not enabled, this field cannot be programmed.
[3]	**ACT_EN**	**Activation Sequence Generator Enable Bit** This bit enables SC controller to initiate the card by activation sequence 0 = No effect. 1 = Activation sequence generator Enabled. **Note1:** When the activation sequence completed, this bit will be cleared automatically and the SC_ISR [INIT_IS] will be set to "1". **Note2:** This field will be cleared when software set SC_ALTCTL[TX_RST] or SC_ALTCTL[RX_RST] to 1. So don't fill this bit, TX_RST, and RX_RST at the same time. **Note3:** If SC_CTL[SC_CEN] not enabled, this field cannot be programmed.

[2]	**DACT_EN**	**Deactivation Sequence Generator Enable Bit** This bit enables SC controller to initiate the card by deactivation sequence 0 = No effect. 1 = Deactivation sequence generator Enabled. **Note1:** When the deactivation sequence completed, this bit will be cleared automatically and the SC_ISR [INIT_IS] will be set to "1". **Note2:** This field will be cleared when software set SC_ALTCTL[TX_RST] or SC_ALTCTL[RX_RST] to 1. So don't fill this bit, TX_RST, and RX_RST at the same time. **Note3:** If SC_CTL [SC_CEN] not enabled, this field cannot be programmed.
[1]	**RX_RST**	**Rx Software Reset** When RX_RST is set, all the bytes in the receiver buffer and Rx internal state machine will be cleared. 0 = No effect. 1 = Reset the Rx internal state machine and pointers. **Note:** This bit will be auto cleared and it needs at least 3 SC peripheral clock cycles.
[0]	**TX_RST**	**TX Software Reset** When TX_RST is set, all the bytes in the transmit buffer and TX internal state machine will be cleared. 0 = No effect. 1 = Reset the TX internal state machine and pointers. **Note:** This bit will be auto cleared and it needs at least 3 SC peripheral clock cycles.

SC Extend Guard Time Register (SC_EGTR)

Register	Offset	R/W	Description	Reset Value
SC_EGTR	SCx_BA+0x0C	R/W	SC Extend Guard Time Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**EGT**	**Extended Guard Time** This field indicates the extended guard time value. **Note:** The counter is ETU based and the real extended guard time is EGT.

SC Receiver buffer Time-out Register (SC_RFTMR)

Register	Offset	R/W	Description	Reset Value
SC_RFTMR	SCx_BA+0x10	R/W	SC Receiver buffer 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**
Reserved							RFTM
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RFTM							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**RFTM**	**SC Receiver Buffer Time-out Register (ETU Based)** The time-out counter resets and starts counting whenever the RX buffer receives a new data. Once the counter decreases to "1" and no new data is received or CPU does not read data by reading SC_RBR register, a receiver time-out interrupt INT_RTMR will be generated (if SC_IER[RTMR_IE] =1). **Note1:** The counter unit is ETU based and the interval of time-out is RFTM + 0.5 **Note2:** Filling all "0" to this field indicates to disable this function.

SC Clock Divider Control Register (SC_ETUCR)

Register	Offset	R/W	Description	Reset Value
SC_ETUCR	SCx_BA+0x14	R/W	SC ETU Control Register	0x0000_0173

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**COMPEN_EN**	**Compensation Mode Enable Bit** This bit enables clock compensation function. When this bit enabled, hardware will alternate between n-1 clock cycles and n clock cycles, where n is the value to be written into the ETU_RDIV register. 0 = Compensation function Disabled. 1 = Compensation function Enabled.
[14:12]	**Reserved**	Reserved.
[11:0]	**ETU_RDIV**	**ETU Rate Divider** The field indicates the clock rate divider. The real ETU is ETU_RDIV + 1. **Note1:** Software can configure this field, but this field must be greater than 0x04. **Note2:** Software can configure this field, but if the error rate is equal to 2%, this field must be greater than 0x040.

SC Interrupt Control Register (SC_IER)

Register	Offset	R/W	Description	Reset Value
SC_IER	SCx_BA+0x18	R/W	SC 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					ACON_ERR_IE	RTMR_IE	INIT_IE
7	6	5	4	3	2	1	0
CD_IE	BGT_IE	TMR2_IE	TMR1_IE	TMR0_IE	TERR_IE	TBE_IE	RDA_IE

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**ACON_ERR_IE**	**Auto Convention Error Interrupt Enable Bit** This field is used for auto-convention error interrupt enable. 0 = Auto-convention error interrupt Disabled. 1 = Auto-convention error interrupt Enabled.
[9]	**RTMR_IE**	**Receiver Buffer Time-out Interrupt Enable Bit** This field is used for receiver buffer time-out interrupt enable. 0 = Receiver buffer time-out interrupt Disabled. 1 = Receiver buffer time-out interrupt Enabled.
[8]	**INIT_IE**	**Initial End Interrupt Enable Bit** This field is used for activation (SC_ALTCTL [ACT_EN]), deactivation (SC_ALTCTL [DACT_EN]) and warm reset (SC_ALTCTL [WARST_EN]) sequence interrupt enable. 0 = Initial end interrupt Disabled. 1 = Initial end interrupt Enabled.
[7]	**CD_IE**	**Card Detect Interrupt Enable Bit** This field is used for card detect interrupt enable. The card detect status register is SC_PINCSR[CD_INS_F] and SC_PINCSR[CD_REM_F]. 0 = Card detect interrupt Disabled. 1 = Card detect interrupt Enabled.
[6]	**BGT_IE**	**Block Guard Time Interrupt Enable Bit** This field is used for block guard time interrupt enable. 0 = Block guard time Disabled. 1 = Block guard time Enabled.
[5]	**TMR2_IE**	**Timer2 Interrupt Enable Bit** This field is used for TMR2 interrupt enable. 0 = Timer2 interrupt Disabled.

		1 = Timer2 interrupt Enabled.
[4]	**TMR1_IE**	**Timer1 Interrupt Enable Bit** This field is used for TMR1 interrupt enable. 0 = Timer1 interrupt Disabled. 1 = Timer1 interrupt Enabled.
[3]	**TMR0_IE**	**Timer0 Interrupt Enable Bit** This field is used for TMR0 interrupt enable. 0 = Timer0 interrupt Disabled. 1 = Timer0 interrupt Enabled.
[2]	**TERR_IE**	**Transfer Error Interrupt Enable Bit** This field is used for transfer error interrupt enable. The transfer error states is at SC_TRSR register which includes receiver break error (RX_EBR_F), frame error (RX_EFR_F), parity error (RX_EPA_F), receiver buffer overflow error (RX_OVER_F), transmit buffer overflow error (TX_OVER_F), receiver retry over limit error (RX_OVER_REERR) and transmitter retry over limit error (TX_OVER_REERR). 0 = Transfer error interrupt Disabled. 1 = Transfer error interrupt Enabled.
[1]	**TBE_IE**	**Transmit Buffer Empty Interrupt Enable Bit** This field is used for transmit buffer empty interrupt enable. 0 = Transmit buffer empty interrupt Disabled. 1 = Transmit buffer empty interrupt Enabled.
[0]	**RDA_IE**	**Receive Data Reach Interrupt Enable Bit** This field is used for received data reaching trigger level (SC_CTL [RX_FTRI_LEV]) interrupt enable. 0 = Receive data reach trigger level interrupt Disabled. 1 = Receive data reach trigger level interrupt Enabled.

SC Interrupt Status Register (SC_ISR)

Register	Offset	R/W	Description	Reset Value
SC_ISR	SCx_BA+0x1C	R/W	SC Interrupt Status Register	0x0000_0002

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					ACON_ERR_IS	RTMR_IS	INIT_IS
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CD_IS	BGT_IS	TMR2_IS	TMR1_IS	TMR0_IS	TERR_IS	TBE_IS	RDA_IS

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**ACON_ERR_IS**	**Auto Convention Error Interrupt Status Flag (Read Only)** This field indicates auto convention sequence error. If the received TS at ATR state is not 0x3B or 0x3F, this bit will be set. **Note:** This bit is read only, but can be cleared by writing "1" to it.
[9]	**RTMR_IS**	**Receiver Buffer Time-out Interrupt Status Flag (Read Only)** This field is used for receiver buffer time-out interrupt status flag. **Note:** This field is the status flag of receiver buffer time-out state. If software wants to clear this bit, software must read the receiver buffer remaining data by reading SC_RBR register,
[8]	**INIT_IS**	**Initial End Interrupt Status Flag (Read Only)** This field is used for activation (SC_ALTCTL [ACT_EN]), deactivation (SC_ALTCTL [DACT_EN]) and warm reset (SC_ALTCTL [WARST_EN]) sequence interrupt status flag. **Note:** This bit is read only, but it can be cleared by writing "1" to it.
[7]	**CD_IS**	**Card Detect Interrupt Status Flag (Read Only)** This field is card detect interrupt status flag. If SC_PINCSR [CD_INS_F] = 1 or SC_PINCSR [CD_REM_F] = 1, this field will be set by hardware. **Note:** If software wants to clear this field, software must clear SC_PINCSR [CD_INS_F] and SC_PINCSR [CD_REM_F].
[6]	**BGT_IS**	**Block Guard Time Interrupt Status Flag (Read Only)** This field is used for block guard time interrupt status flag. **Note1:** This bit is valid when SC_ALTCTL[RX_BGT_EN] is enabled. **Note2:** This bit is read only, but it can be cleared by writing "1" to it.
[5]	**TMR2_IS**	**Timer2 Interrupt Status Flag (Read Only)** This field is used for TMR2 interrupt status flag. **Note:** This bit is read only, but it can be cleared by writing "1" to it.
[4]	**TMR1_IS**	**Timer1 Interrupt Status Flag (Read Only)**

		This field is used for TMR1 interrupt status flag. **Note:** This bit is read only, but it can be cleared by writing "1" to it.
[3]	**TMR0_IS**	**Timer0 Interrupt Status Flag (Read Only)** This field is used for TMR0 interrupt status flag. **Note:** This bit is read only, but it can be cleared by writing "1" to it.
[2]	**TERR_IS**	**Transfer Error Interrupt Status Flag (Read Only)** This field is used for transfer error interrupt status flag. The transfer error status is at the SC_TRSR register which includes receiver break error (RX_EBR_F), frame error (RX_EFR_F), parity error (RX_EPA_F) and receiver buffer overflow error (RX_OVER_F), transmit buffer overflow error (TX_OVER_F), receiver retry over limit error (RX_OVER_REERR) and transmitter retry over limit error (TX_OVER_REERR). **Note:** This field is the status flag of SC_TRSR[RX_EBR_F], SC_TRSR[RX_EFR_F], SC_TRSR[RX_EPA_F], SC_TRSR[RX_OVER_F], SC_TRSR[TX_OVER_F], SC_TRSR[RX_OVER_REERR] or SC_TRSR[TX_OVER_REERR]. So if software wants to clear this bit, software must write "1" to each field.
[1]	**TBE_IS**	**Transmit Buffer Empty Interrupt Status Flag (Read Only)** This field is used for transmit buffer empty interrupt status flag. **Note:** This field is the status flag of transmit buffer empty state. If software wants to clear this bit, software must write data to SC_THR register and then this bit will be cleared automatically.
[0]	**RDA_IS**	**Receive Data Reach Interrupt Status Flag (Read Only)** This field is used for received data reaching trigger level (SC_CTL[RX_FTRI_LEV]) interrupt status flag. **Note:** This field is the status flag of received data reaching SC_CTL [RX_FTRI_LEV]. If software reads data from SC_RBR and receiver pointer is less than SC_CTL [RX_FTRI_LEV], this bit will be cleared automatically.

SC Transfer Status Register (SC_TRSR)

Register	Offset	R/W	Description	Reset Value
SC_TRSR	SCx_BA+0x20	R/W	SC Transfer Status Register	0x0000_0202

31	30	29	28	27	26	25	24
TX_ATV	**TX_OVER_REERR**	**TX_REERR**	**Reserved**		**TX_POINT_F**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RX_ATV	**RX_OVER_REERR**	**RX_REERR**	**Reserved**		**RX_POINT_F**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**TX_FULL_F**	**TX_EMPTY_F**	**TX_OVER_F**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**RX_EBR_F**	**RX_EFR_F**	**RX_EPA_F**	**Reserved**	**RX_FULL_F**	**RX_EMPTY_F**	**RX_OVER_F**

Bits	Description	
[31]	**TX_ATV**	**Transmit in Active Status Flag (Read Only)** This bit is set by hardware when TX transfer is in active and the STOP bit of the last byte has not been transmitted. This bit is cleared automatically when TX transfer is finished or the last byte transmission has completed.
[30]	**TX_OVER_REERR**	**Transmitter over Retry Error (Read Only)** This bit is set by hardware when transmitter re-transmits over retry number limitation. **Note:** This bit is read only, but it can be cleared by writing "1" to it.
[29]	**TX_REERR**	**Transmitter Retry Error (Read Only)** This bit is set by hardware when transmitter re-transmits. **Note1:** This bit is read only, but it can be cleared by writing "1" to it. **Note2:** This bit is a flag and cannot generate any interrupt to CPU.
[28:26]	**Reserved**	Reserved.
[25:24]	**TX_POINT_F**	**Transmit Buffer Pointer Status Flag (Read Only)** This field indicates the TX buffer pointer status flag. When CPU writes data into SC_THR, TX_POINT_F increases one. When one byte of TX Buffer is transferred to transmitter shift register, TX_POINT_F decreases one.
[23]	**RX_ATV**	**Receiver in Active Status Flag (Read Only)** This bit is set by hardware when RX transfer is in active. This bit is cleared automatically when RX transfer is finished.
[22]	**RX_OVER_REERR**	**Receiver over Retry Error (Read Only)** This bit is set by hardware when RX transfer error retry over retry number limit. **Note1:** This bit is read only, but it can be cleared by writing "1" to it. **Note2:** If CPU enables receiver retries function by setting SC_CTL [RX_ERETRY_EN] register, the RX_EPA_F flag will be ignored (hardware will not set RX_EPA_F).
[21]	**RX_REERR**	**Receiver Retry Error (Read Only)**

		This bit is set by hardware when RX has any error and retries transfer. **Note1:** This bit is read only, but it can be cleared by writing "1" to it. **Note2** This bit is a flag and cannot generate any interrupt to CPU. **Note3:** If CPU enables receiver retry function by setting SC_CTL [RX_ERETRY_EN] register, the RX_EPA_F flag will be ignored (hardware will not set RX_EPA_F).
[20:18]	**Reserved**	Reserved.
[17:16]	**RX_POINT_F**	**Receiver Buffer Pointer Status Flag (Read Only)** This field indicates the RX buffer pointer status flag. When SC receives one byte from external device, RX_POINT_F increases one. When one byte of RX buffer is read by CPU, RX_POINT_F decreases one.
[15:11]	**Reserved**	Reserved.
[10]	**TX_FULL_F**	**Transmit Buffer Full Status Flag (Read Only)** This bit indicates TX buffer full or not. This bit is set when TX pointer is equal to 4, otherwise is cleared by hardware.
[9]	**TX_EMPTY_F**	**Transmit Buffer Empty Status Flag (Read Only)** This bit indicates TX buffer empty or not. When the last byte of TX buffer has been transferred to Transmitter Shift Register, hardware sets this bit high. It will be cleared when writing data into SC_THR (TX buffer not empty).
[8]	**TX_OVER_F**	**TX Overflow Error Interrupt Status Flag (Read Only)** If TX buffer is full, an additional write to SC_THR will cause this bit be set to "1" by hardware. **Note:** This bit is read only, but it can be cleared by writing "1" to it.
[7]	**Reserved**	Reserved.
[6]	**RX_EBR_F**	**Receiver Break Error Status Flag (Read Only)** This bit is set to "1" whenever the received data input (RX) held in the "spacing state" (logic "0") is longer than a full word transmission time (that is, the total time of "start bit" + data bits + parity + stop bits). . **Note1:** This bit is read only, but it can be cleared by writing "1" to it. **Note2:** If CPU sets receiver retries function by setting SC_CTL[RX_ERETRY_EN] register, hardware will not set this flag.
[5]	**RX_EFR_F**	**Receiver Frame Error Status Flag (Read Only)** This bit is set to "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). **Note1:** This bit is read only, but can be cleared by writing "1" to it. **Note2:** If CPU sets receiver retries function by setting SC_CTL[RX_ERETRY_EN] register, hardware will not set this flag.
[4]	**RX_EPA_F**	**Receiver Parity Error Status Flag (Read Only)** This bit is set to "1" whenever the received character does not have a valid "parity bit". **Note1:** This bit is read only, but it can be cleared by writing "1" to it. **Note2:** If CPU sets receiver retries function by setting SC_CTL[RX_ERETRY_EN] register, hardware will not set this flag.
[3]	**Reserved**	Reserved.
[2]	**RX_FULL_F**	**Receiver Buffer Full Status Flag (Read Only)** This bit indicates RX buffer full or not. This bit is set when RX pointer is equal to 4, otherwise it is cleared by hardware.
[1]	**RX_EMPTY_F**	**Receiver Buffer Empty Status Flag(Read Only)** This bit indicates RX buffer empty or not.

		When the last byte of RX buffer has been read by CPU, hardware set this bit to “1”. It will be cleared by hardware when SC receives any new data.
[0]	**RX_OVER_F**	**RX Overflow Error Status Flag (Read Only)** This bit is set when RX buffer overflow. If the number of received bytes is greater than Rx Buffer size (4 bytes), this bit will be set. **Note:** This bit is read only, but it can be cleared by writing “1” to it.

SC PIN Control State Register (SC_PINCSR)

Register	Offset	R/W	Description	Reset Value
SC_PINCSR	SCx_BA+0x24	R/W	SC Pin Control State Register	0x000x_00x0

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							SC_DATA_I_ST
15	14	13	12	11	10	9	8
Reserved				POW_INV	CD_LEV	SC_DATA_O	SC_OEN_ST
7	6	5	4	3	2	1	0
ADAC_CD_EN	CLK_KEEP	Reserved	CD_PIN_ST	CD_INS_F	CD_REM_F	SC_RST	POW_EN

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**SC_DATA_I_ST**	**SC Data Pin Status (Read Only)** This bit is the pin status of SC_DATA 0 = The SC_DATA pin is low. 1 = The SC_DATA pin is high.
[15:12]	**Reserved**	Reserved.
[11]	**POW_INV**	**SC_PWR Pin Inverse** This bit is used for inverse the SC_PWR pin. There are four kinds of combination for SC_PWR pin setting by POW_INV(SC_PINCSR[11]) and POW_EN(SC_PINCSR[0]). POW_INV(SC_PINCSR[11]) is bit 1 and POW_EN(SC_PINCSR[0]) is bit 0 for SC_PWR Pin as high or low voltage selection. 00 = SC_PWR Pin is 0. 01 = SC_PWR Pin is 1. 10 = SC_PWR Pin is 1. 11 = SC_PWR Pin is 0. **Note:** Software must select POW_INV before Smart Card engine is enabled.
[10]	**CD_LEV**	**Card Detect Level** 0 = When hardware detects the card detect pin from high to low, it indicates a card is detected. 1 = When hardware detects the card detect pin from low to high, it indicates a card is detected. **Note:** Software must select card detect level before Smart Card engine is enabled
[9]	**SC_DATA_O**	**SC Data Output Pin** This bit is the pin status of SC_DATA but user can drive SC_DATA pin to high or low by setting this bit. Write this field to drive SC_DATA pin.

		0 = Drive SC_DATA pin to low. 1 = Drive SC_DATA pin to high. Read this field to get SC_DATA pin status. 0 = SC_DATA pin status is low. 1 = SC_DATA pin status is high. **Note:** When SC is at activation, warm reset or deactivation mode, this bit will be changed automatically. So don't fill this field when SC is in these modes.
[8]	**SC_OEN_ST**	**SC Data Output Enable Pin Status (Read Only)** This bit is the output enable status of the SC_DATA pin 0 = The SC_DATA pin state is output. 1 = The SC_DATA pin state is not output.
[7]	**ADAC_CD_EN**	**Auto Deactivation When Card Removal** 0 = Auto deactivation Disabled when hardware detected the card removal. 1 = Auto deactivation Enabled when hardware detected the card removal. **Note:** When the card is removed, hardware will stop any process and then do deactivation sequence (if this bit be setting). If this process completes. Hardware will generate an initial end interrupt to CPU.
[6]	**CLK_KEEP**	**SC Clock Enable Bit** 0 = SC clock generation Disabled. 1 = SC clock always keeps free running. **Note:** When operating at activation, warm reset or deactivation mode, this bit will be changed automatically. So don't fill this field when operating in these modes.
[5]	**Reserved**	Reserved.
[4]	**CD_PIN_ST**	**Card Detect Status of SC_CD Pin Status (Read Only)** This bit is the pin status flag of SC_CD 0 = The SC_CD pin state at low. 1 = The SC_CD pin state at high.
[3]	**CD_INS_F**	**Card Detect Insert Status of SC_CD Pin (Read Only)** This bit is set whenever card has been inserted. 0 = No effect. 1 = Card insert. **Note1:** This bit is read only, but it can be cleared by writing "1" to it. **Note2:** The card detect engine will start after SC_CTL[SC_CEN] set.
[2]	**CD_REM_F**	**Card Detect Removal Status of SC_CD Pin (Read Only)** This bit is set whenever a card has been removed. 0 = No effect. 1 = Card removed. **Note1:** This bit is read only, but it can be cleared by writing "1" to it. **Note2:** Card detect engine will start after SC_CTL[SC_CEN] set.
[1]	**SC_RST**	**SC_RST Pin Signal** This bit is the pin status of SC_RST but user can drive SC_RST pin to high or low by setting this bit. Write this field to drive SC_RST pin. 0 = Drive SC_RST pin to low. 1 = Drive SC_RST pin to high. Read this field to get SC_RST pin status. 0 = SC_RST pin status is low.

		1 = SC_RST pin status is high. **Note:** When operating at activation, warm reset or deactivation mode, this bit will be changed automatically. So don't fill this field when operating in these modes.
[0]	**POW_EN**	**SC_POW_EN Pin Signal** Software can set POW_EN and POW_INV to decide SC_PWR pin is in high or low level. Write this field to drive SC_PWR pin. Refer POW_INV(SC_PINCSR[11]) description for programming SC_PWR pin voltage level. Read this field to get SC_PWR pin status. 0 = SC_PWR pin status is low. 1 = SC_PWR pin status is high. **Note:** When operating at activation, warm reset or deactivation mode, this bit will be changed automatically. So don't fill this field when operating in these modes.

SC Timer Control Register 0 (SC_TMR0)

Register	Offset	R/W	Description	Reset Value
SC_TMR0	SCx_BA+0x28	R/W	SC Internal Timer Control Register 0	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**MODE**	**Timer 0 Operation Mode Selection** This field indicates the internal 24-bit timer operation selection. Refer to Table 6-22 for programming Timer0
[23:0]	**CNT**	**Timer 0 Counter Value Register (ETU Based)** This field indicates the internal timer operation values.

SC Timer Control Register 1 (SC_TMR1)

Register	Offset	R/W	Description	Reset Value
SC_TMR1	SCx_BA+0x2C	R/W	SC Internal Timer Control Register 1	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**MODE**	**Timer 1 Operation Mode Selection** This field indicates the internal 8-bit timer operation selection. Refer to Table 6-22 for programming Timer1
[23:8]	**Reserved**	Reserved.
[7:0]	**CNT**	**Timer 1 Counter Value Register (ETU Based)** This field indicates the internal timer operation values.

SC Timer Control Register 2 (SC_TMR2)

Register	Offset	R/W	Description	Reset Value
SC_TMR2	SCx_BA+0x30	R/W	SC Internal Timer Control Register 2	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**MODE**	**Timer 2 Operation Mode Selection** This field indicates the internal 8-bit timer operation selection. Refer to Table 6-22 for programming Timer2
[23:8]	**Reserved**	Reserved.
[7:0]	**CNT**	**Timer 2 Counter Value Register (ETU Based)** This field indicates the internal timer operation values.

SC Timer Current Data Register A (SC_TDRA)

Register	Offset	R/W	Description	Reset Value
SC_TDRA	SCx_BA+0x38	R	SC Timer Current Data Register A	0x0000_07FF

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TDR0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TDR0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TDR0							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**TDR0**	**Timer0 Current Data Register (Read Only)** This field indicates the current count values of timer0.

SC Timer Current Data Register B (SC_TDRB)

Register	Offset	R/W	Description	Reset Value
SC_TDRB	SCx_BA+0x3C	R	SC Timer Current Data Register B	0x0000_7F7F

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TDR2							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TDR1							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**TDR2**	**Timer2 Current Data Register (Read Only)** This field indicates the current count values of timer2.
[7:0]	**TDR1**	**Timer1 Current Data Register (Read Only)** This field indicates the current count values of timer1.

6.15 PS/2 Device Controller (PS2D)

6.15.1 Overview

The PS/2 device controller provides a basic timing control for PS/2 communication. All communication between the device and the host is managed through the CLK and DATA pins. Unlike PS/2 keyboard or mouse device controller, the receive/transmit code needs to be translated as meaningful code by firmware. The device controller generates the CLK signal after receiving a “Request to Send” state, but host has ultimate control over communication. Data of DATA line sent from the host to the device is read on the rising edge and sent from the device to the host is change after rising edge. A 16 bytes FIFO is used to reduce CPU intervention. Software can select 1 to 16 bytes for a continuous transmission.

6.15.2 Features

- Host communication inhibit and Request to Send state detection
- Reception frame error detection
- Programmable 1 to 16 bytes transmit buffer to reduce CPU intervention
- Double buffer for data reception
- Software override bus

6.15.3 Block Diagram

The PS/2 device controller consists of APB interface and timing control logic for DATA and CLK pins.

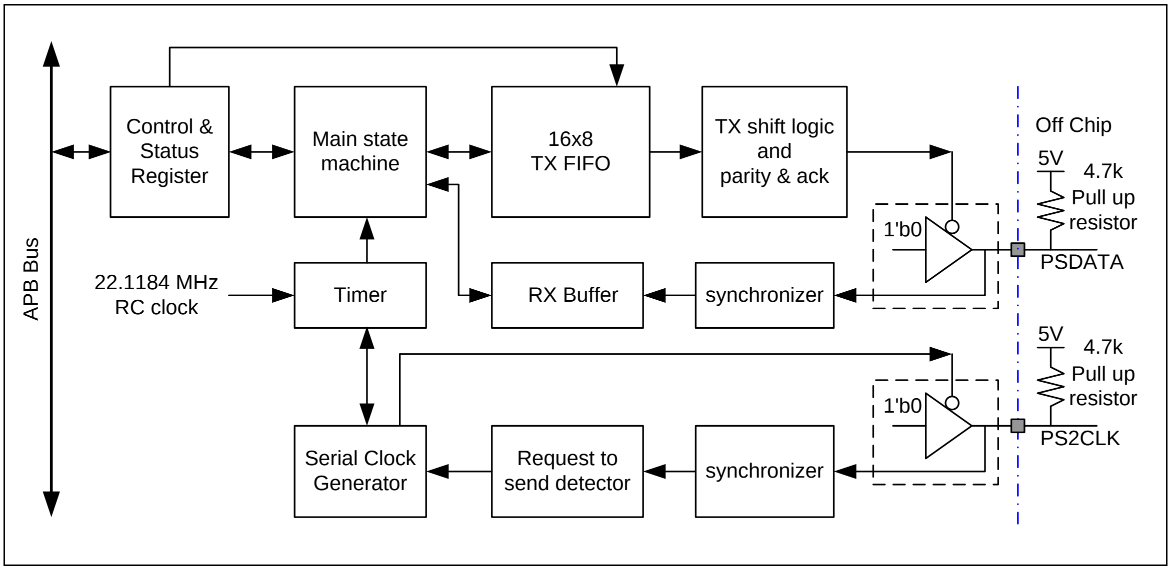


Figure 6-86 PS/2 Device Block Diagram

6.15.4 Functional Description

6.15.4.1 Communication

The PS/2 device implements a bidirectional synchronous serial protocol. The bus is "Idle" when both lines are high (open-collector). This is the only state where the device is allowed start to transmit PS/2 data. The host has ultimate control over the bus and may inhibit communication at any time by pulling the CLK line low.

The CLK signal is generated by PS/2 device. If the host wants to send PS/2 data, it must first inhibit communication from the device by pulling CLK low. The host then pulls DATA low and releases CLK. This is the "Request-to-Send" state and signals the device to start generating CLK pulses.

DATA	CLK	Bus State
High	High	Idle
High	Low	Communication Inhibit
Low	High	Host Request to Send

All data is transmitted one byte at a time and each byte is sent in a frame consisting of 11 or 12 bits. These bits are:

- 1 start bit, which is always 0
- 8 data bits, least significant bit first
- 1 parity bit (odd parity)
- 1 stop bit, which is always 1
- 1 acknowledge bit (host-to-device communication only)

The parity bit is set if there is an even number of 1's in the data bits and cleared to 0 if there is an odd number of 1's in the data bits. This is used for parity error detection that is the number of 1's in the data bits plus the parity bit and always adds up to an odd number then parity bit sets to 1. The device must check this bit and if incorrect it should respond as if it had received an invalid command.

The host may inhibit communication at any time by pulling the CLK line low for at least 100 us. If a transmission is inhibited before the 11th clock pulse, the device must abort the current transmission and prepare to resend the current data when host releases CLK. In order to reserve enough time for software to decode host command, the transmit logic is blocked by RXINT bit, software must clear the RXINT bit to start resend. Software can write CLRFIFO to 1 to reset FIFO pointer if needed.

Device-to-Host

The device uses a serial protocol with 11-bit frames. These bits are:

- 1 start bit, which is always 0
- 8 data bits, least significant bit first
- 1 parity bit (odd parity)
- 1 stop bit, which is always 1

The device writes a bit on the DATA line when CLK is high, and it is read by the host when CLK is low, which is illustrated in Figure 6-87.

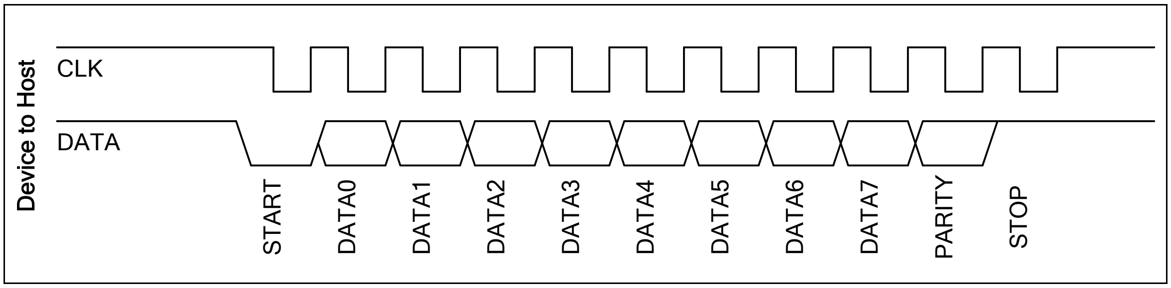


Figure 6-87 Data Format of Device-to-Host

Host-to-Device:

First, the PS/2 device always generates the CLK signal. If the host wants to send PS/2 data, it must first put the CLK and DATA lines in a "Request-to-send" state as follows:

- Inhibit communication by pulling CLK low for at least 100 us
- Apply "Request-to-send" by pulling DATA low, then release CLK

The device should check for this state at intervals not to exceed 10 ms. When the device detects this state, it will begin generating CLK signals and CLK in eight DATA bits, one parity bit and one stop bit. The host changes the DATA line status only when the CLK line status is low, and DATA is read by the device when CLK is high.

After the stop bit is received, the device will acknowledge the received byte by bringing the DATA line low and generating one last CLK pulse. If the host does not release the DATA line after the 11th CLK pulse, the device will continue to generate CLK pulses until the DATA line is released.

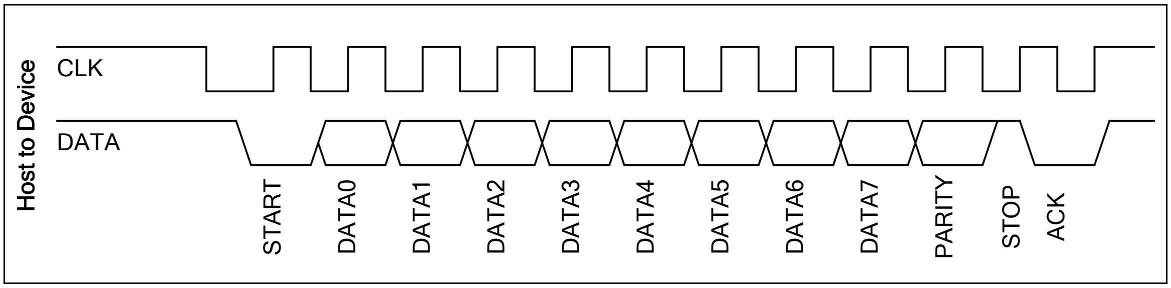


Figure 6-88 Data Format of Host-to-Device

The host and the detailed timing of the DATA and CLK for communication are shown below:

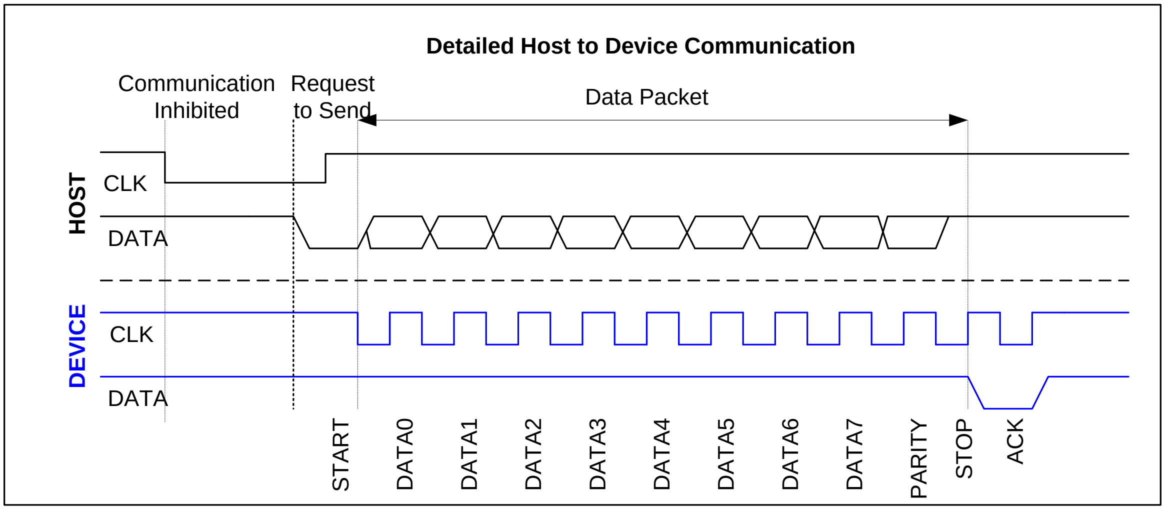


Figure 6-89 PS/2 Bit Data Format

6.15.4.2 PS/2 Bus Timing Specification

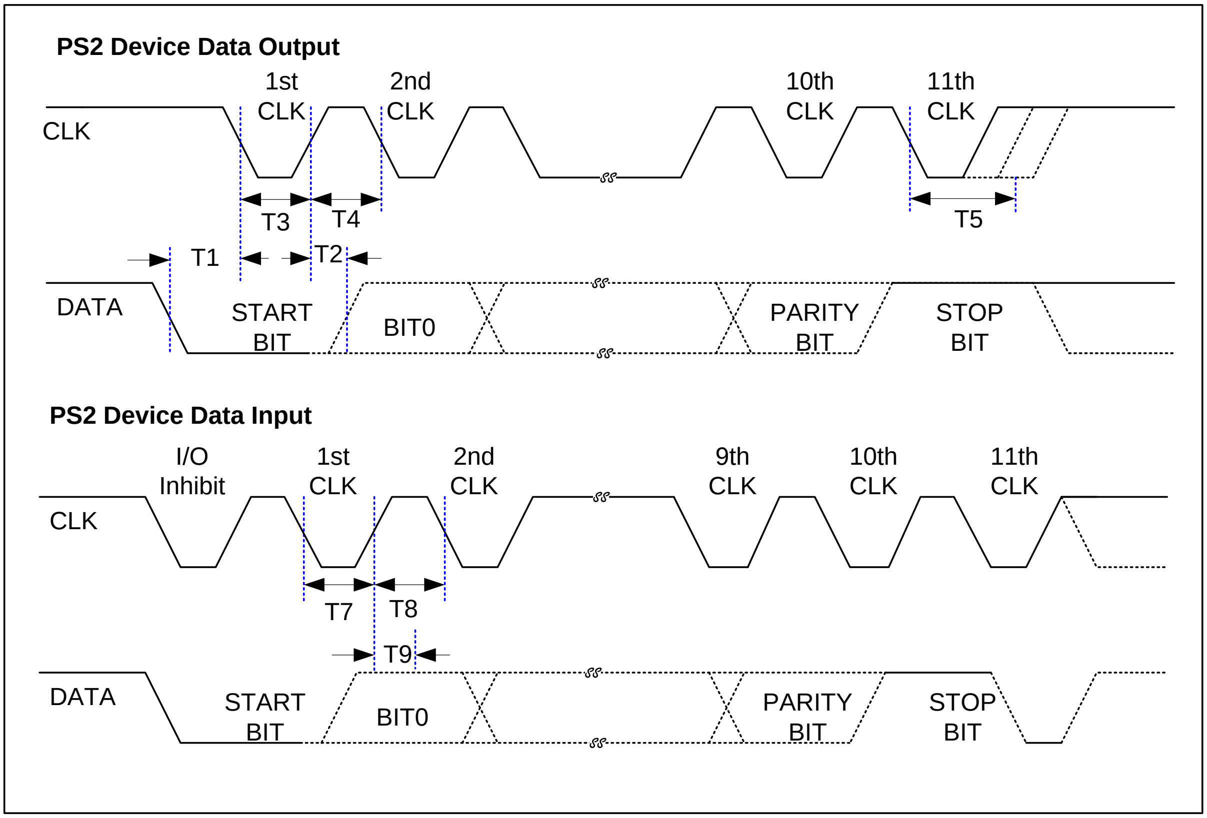


Figure 6-90 PS/2 Bus Timing

Symbol	Timing Parameter	Min.	Max.

T1	DATA transition to the falling edge of CLK	5us	25us
T2	Rising edge of CLK to DATA transition	5us	T4-5us
T3	Duration of CLK inactive	30us	50us
T4	Duration of CLK active	30us	50us
T5	Time to auxiliary device inhibit after 11th clock to ensure auxiliary device does not start another transmission	>0	50us
T7	Duration of CLK inactive	30us	50us
T8	Duration of CLK active	30us	50us
T9	Time from inactive to active CLK transition, use to time auxiliary device sample DATA	5us	25us

6.15.4.3 TX FIFO Operation

Writing data to PS2TXDATA0 register starts device to host communication. Software is required to define the TXFIFO depth before writing transmission data to TX FIFO. The first START bit is sent to PS/2 bus when software writes TX FIFO and after 100us. If there are more than 4 bytes data need to send, software can write residual data to PS2TXDATA1-3 before 4th byte transmit complete. A time delay 100us is added between two consecutive bytes.

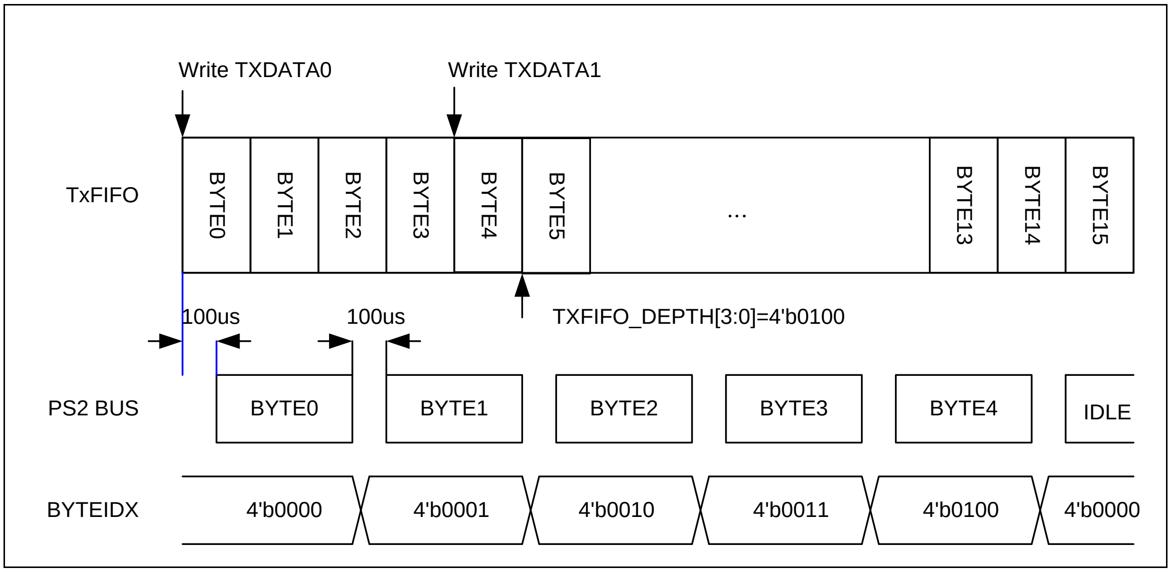


Figure 6-91 PS/2 Data Format

6.15.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
PS/2 Base Address: **PS2_BA = 0x4010_0000**				
PS2CON	PS2_BA+0x00	R/W	PS/2 Control Register	0x0000_0000
PS2TXDATA0	PS2_BA+0x04	R/W	PS/2 Transmit Data Register 0	0x0000_0000
PS2TXDATA1	PS2_BA+0x08	R/W	PS/2 Transmit Data Register 1	0x0000_0000
PS2TXDATA2	PS2_BA+0x0C	R/W	PS/2 Transmit Data Register 2	0x0000_0000
PS2TXDATA3	PS2_BA+0x10	R/W	PS/2 Transmit Data Register 3	0x0000_0000
PS2RXDATA	PS2_BA+0x14	R	PS/2 Receive Data Register	0x0000_0000
PS2STATUS	PS2_BA+0x18	R/W	PS/2 Status Register	0x0000_0083
PS2INTID	PS2_BA+0x1C	R/W	PS/2 Interrupt Identification Register	0x0000_0000

6.15.6 Register Description

PS/2 Control Register (PS2CON)

Register	Offset	R/W	Description	Reset Value
PS2CON	PS2_BA+0x00	R/W	PS/2 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				FPS2DAT	FPS2CLK	OVERRIDE	CLRFIFO
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACK	TXFIFO_DEPTH				RXINTEN	TXINTEN	PS2EN

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**FPS2DAT**	**Force PS2DATA Line** It forces PS2DATA high or low regardless of the internal state of the device controller if OVERRIDE is set to 1. 0 = Force PS2DATA low. 1 = Force PS2DATA high.
[10]	**FPS2CLK**	**Force PS2CLK Line** It forces PS2CLK line high or low regardless of the internal state of the device controller if OVERRIDE is set to 1. 0 = Force PS2CLK line low. 1 = Force PS2CLK line high.
[9]	**OVERRIDE**	**Software Override PS2 CLK/DATA Pin State** 0 = PS2CLK and PS2DATA pins are controlled by internal state machine. 1 = PS2CLK and PS2DATA pins are controlled by software.
[8]	**CLRFIFO**	**Clear TX FIFO** Write 1 to this bit to terminate device to host transmission. The TXEMPTY bit in PS2STATUS bit will be set to 1 and pointer BYTEIDEX is reset to 0 regardless there is residue data in buffer or not. The buffer content is not been cleared. 0 = Not active. 1 = Clear FIFO.
[7]	**ACK**	**Acknowledge Enable Bit** 0 = Always send acknowledge to host at 12th clock for host to device communication. 1 = If parity bit error or stop bit is not received correctly, acknowledge bit will not be sent to host at 12th clock.
[6:3]	**TXFIFO_DEPTH**	**Transmit Data FIFO Depth** There are 16 bytes buffer for data transmit. Software can define the FIFO depth from 1 to

		16 bytes depends on application needs. 0 = 1 byte. 1 = 2 bytes. … 14 = 15 bytes. 15 = 16 bytes.
[2]	**RXINTEN**	**Receive Interrupt Enable Bit** 0 = Data receive complete interrupt Disabled. 1 = Data receive complete interrupt Enabled.
[1]	**TXINTEN**	**Transmit Interrupt Enable Bit** 0 = Data transmit complete interrupt Disabled. 1 = Data transmit complete interrupt Enabled.
[0]	**PS2EN**	**PS/2 Device Enable Bit** 0 = PS/2 device controller Disabled. 1 = PS/2 device controller Enabled.

PS/2 TX DATA Register 0-3 (PS2TXDATA0-3)

Register	Offset	R/W	Description	Reset Value
PS2TXDATA0	PS2_BA+0x04	R/W	PS/2 Transmit Data Register 0	0x0000_0000
PS2TXDATA1	PS2_BA+0x08	R/W	PS/2 Transmit Data Register 1	0x0000_0000
PS2TXDATA2	PS2_BA+0x0C	R/W	PS/2 Transmit Data Register 2	0x0000_0000
PS2TXDATA3	PS2_BA+0x10	R/W	PS/2 Transmit Data Register 3	0x0000_0000

31	30	29	28	27	26	25	24
PS2TXDATAx							
23	22	21	20	19	18	17	16
PS2TXDATAx							
15	14	13	12	11	10	9	8
PS2TXDATAx							
7	6	5	4	3	2	1	0
PS2TXDATAx							

Bits	Description	
[31:0]	**PS2TXDATAx**	**Transmit Data** Writing data to this register starts in device to host communication if bus is in IDLE state. Software must enable PS2EN before writing data to TX buffer.

PS/2 Receiver DATA Register (PS2RXDATA)

Register	Offset	R/W	Description	Reset Value
PS2RXDATA	PS2_BA+0x14	R	PS/2 Receive Data Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**RXDATA**	**Received Data** For host to device communication, after acknowledge bit is sent, the received data is copied from receive shift register to PS2RXDATA register. CPU must read this register before next byte reception complete, otherwise the data will be overwritten and RXOVF bit in PS2STATUS[6] will be set to 1.

PS/2 Status Register (PS2STATUS)

Register	Offset	R/W	Description	Reset Value
PS2STATUS	PS2_BA+0x18	R/W	PS/2 Status Register	0x0000_0083

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**BYTEIDX**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXEMPTY	**RXOVF**	**TXBUSY**	**RXBUSY**	**RXPARITY**	**FRAMERR**	**PS2DATA**	**PS2CLK**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**BYTEIDX**	**Byte Index (Read Only)** It indicates which data byte in transmit data shift register. When all data in FIFO is transmitted and it will be cleared to 0. 0000 = DATA Transmit is TXDATA0[7:0]. 0001 = DATA Transmit is TXDATA0[15:8]. 0010 = DATA Transmit is TXDATA0[23:16]. 0011 = DATA Transmit is TXDATA0[31:24]. 0100 = DATA Transmit is TXDATA1[7:0]. 0101 = DATA Transmit is TXDATA1[15:8]. 0110 = DATA Transmit is TXDATA1[23:16]. 0111 = DATA Transmit is TXDATA1[31:24]. 1000 = DATA Transmit is TXDATA2[7:0]. 1001 = DATA Transmit is TXDATA2[15:8]. 1010 = DATA Transmit is TXDATA2[23:16]. 1011 = DATA Transmit is TXDATA2[31:24]. 1100 = DATA Transmit is TXDATA3[7:0]. 1101 = DATA Transmit is TXDATA3[15:8]. 1110 = DATA Transmit is TXDATA3[23:16]. 1111 = DATA Transmit is TXDATA3[31:24].
[7]	**TXEMPTY**	**TX FIFO Empty (Read Only)** When software writes data to PS2TXDATA0-3, the TXEMPTY bit is cleared to 0 immediately if PS2EN is enabled. When transmitted data byte number is equal to FIFODEPTH then TXEMPTY bit is set to 1. 0 = There is data to be transmitted. 1 = FIFO is empty.
[6]	**RXOVF**	**RX Buffer Overwrite**

		0 = No overwrite. 1 = Data in PS2RXDATA register is overwritten by new received data. **Note:** This bit can be cleared by writing '1' to it.
[5]	**TXBUSY**	**Transmit Busy (Read Only)** This bit indicates that the PS/2 device is currently sending data. 0 = Idle. 1 = Currently sending data.
[4]	**RXBUSY**	**Receive Busy (Read Only)** This bit indicates that the PS/2 device is currently receiving data. 0 = Idle. 1 = Currently receiving data.
[3]	**RXPARITY**	**Received Parity (Read Only)** This bit reflects the parity bit for the last received data byte (odd parity).
[2]	**FRAMERR**	**Frame Error** For host to device communication, this bit sets to 1 if STOP bit (logic 1) is not received. If frame error occurs, the PS2_DATA line may keep at low state after 12th clock. At this moment, software overrides PS2CLK to send clock till PS2DATA release to high state. After that, device sends a "Resend" command to host. 0 = No frame error. 1 = Frame error occur. **Note:** This bit can be cleared by writing '1' to it.
[1]	**PS2DATA**	**DATA Pin State** This bit reflects the status of the PS2DATA line after synchronizing and sampling.
[0]	**PS2CLK**	**CLK Pin State** This bit reflects the status of the PS2CLK line after synchronizing.

PS/2 Interrupt Identification Register (PS2INTID)

Register	Offset	R/W	Description	Reset Value
PS2INTID	PS2_BA+0x1C	R/W	PS/2 Interrupt Identification Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TXINT**	**Transmit Interrupt** This bit is set to 1 after STOP bit is transmitted. Interrupt occur if TXINTEN bit is set to 1. 0 = No interrupt. 1 = Transmit interrupt occurs. **Note:** This bit can be cleared by writing '1' to it.
[0]	**RXINT**	**Receive Interrupt** This bit is set to 1 when acknowledge bit is sent for Host to device communication. Interrupt occurs if RXINTEN bit is set to 1. 0 = No interrupt. 1 = Receive interrupt occurs. **Note:** This bit can be cleared by writing '1' to it.

6.16 I²C Serial Interface Controller (I²C)

6.16.1 Overview

I²C is a two-wire, bidirectional 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.

Data is transferred between a Master and a Slave. Data bits transfer on the SCL and SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit long. There is one SCL clock pulse for each data bit with the MSB being transmitted first, and an acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (START or STOP). Please refer to Figure 6-92 for more detailed I²C BUS Timing.

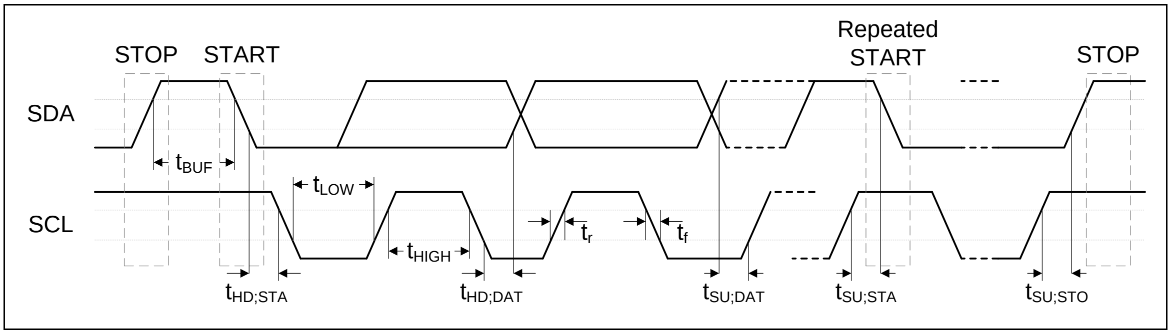


Figure 6-92 I²C Bus Timing

The device's on-chip I²C logic provides a serial interface that meets the I²C bus standard mode specification. The I²C port handles byte transfers autonomously. To enable this port, the bit ENS1 in I2CON should be set to '1'. The I²C hardware interfaces to the I²C bus via two pins: SDA and SCL. Pull-up resistor is needed for I²C operation as the SDA and SCL are open drain pins. When I/O pins are used as I²C ports, user must set the pins function to I²C in advance.

6.16.2 Features

The I²C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main features of the bus include:

- 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
- A built-in 14-bit time-out counter requesting the I²C interrupt if the I²C bus hangs up and timer-out counter overflows
- External pull-up resistors needed for high output
- Programmable clocks allowing for versatile rate control
- Supports 7-bit addressing mode
- Supports multiple address recognition (four slave addresses with mask option)

6.16.3 Functional Description

6.16.3.1 I²C Protocol

Normally, a standard communication consists of four parts:

1) START or Repeated START signal generation
2) Slave address and R/W bit transfer
3) Data transfer
4) STOP signal generation

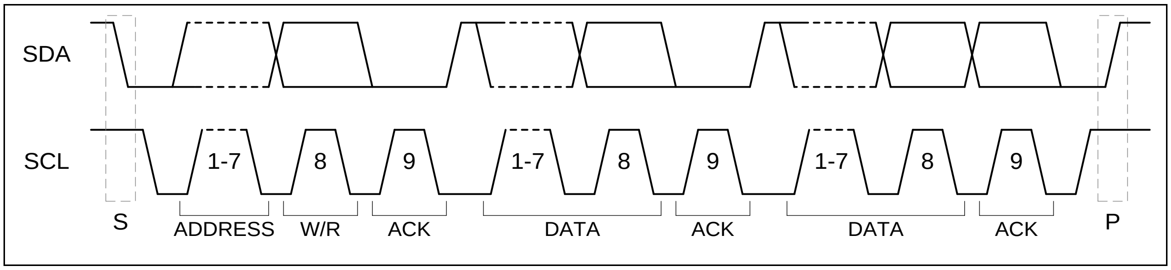


Figure 6-93 I²C Protocol

6.16.3.2 Data transfer on the I²C-bus

Figure 6-94 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.

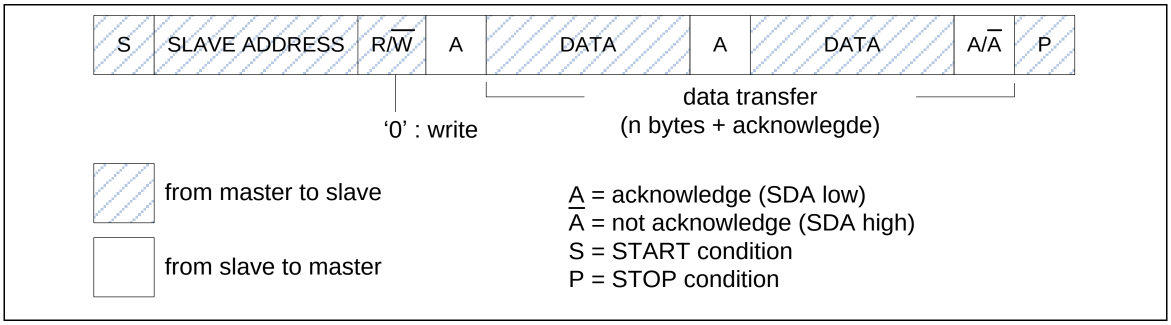


Figure 6-94 Master Transmits Data to Slave

Figure 6-95 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.

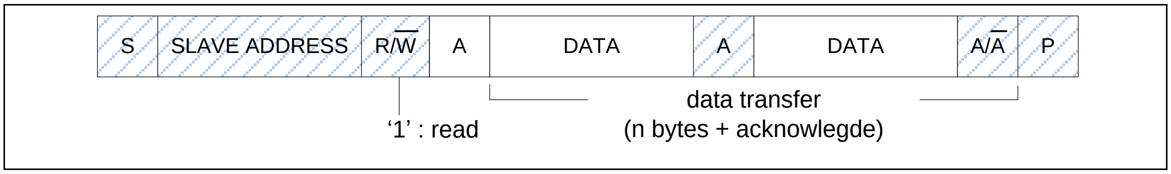


Figure 6-95 Master Reads Data from Slave

6.16.3.3 START or Repeated START signal

When the bus is free/idle, meaning no master device is engaging the bus (both SCL and SDA lines are high), a master can initiate a transfer by sending a START signal. A START signal, usually referred to as the S-bit, is defined as a HIGH to LOW transition on the SDA line while SCL is HIGH. The START signal denotes the beginning of a new data transmission.

A Repeated START (Sr) is not a STOP signal between two START signals. The master uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

STOP signal

The master can terminate the communication by generating a STOP signal. A STOP signal, usually referred to as the P-bit, is defined as a LOW to HIGH transition on the SDA line while SCL is HIGH.

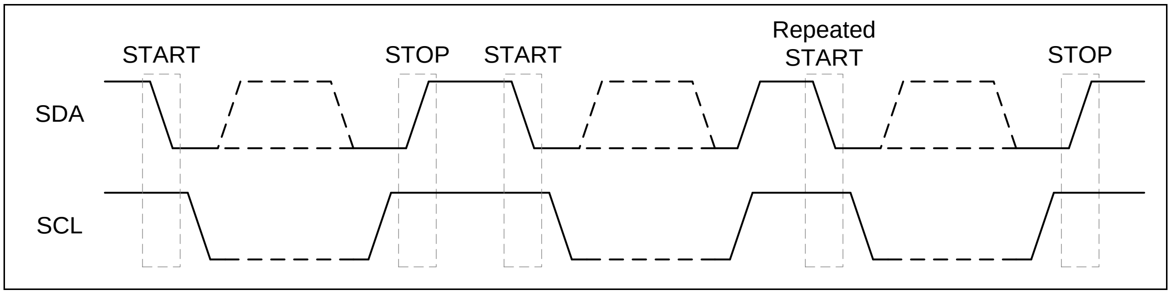


Figure 6-96 START and STOP Conditions

6.16.3.4 Slave Address Transfer

The first byte of data transferred by the master immediately after the START signal is the slave address. This is a 7-bit calling address followed by a RW bit. The RW bit signals the slave 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 SDA low at the 9th SCL clock cycle.

6.16.3.5 Data Transfer

When a slave receives a correct address with a RW bit, the data will follow RW bit specified to transfer. Each transferred byte is followed by an acknowledge bit on the 9th SCL clock cycle. If the slave signals a Not Acknowledge (NACK), the master can generate a STOP signal to abort the data transfer or generate a Repeated START signal and start a new transfer cycle.

If the master, as a receiving device, does Not Acknowledge (NACK) the slave, the slave releases the SDA line for the master to generate a STOP or Repeated START signal.

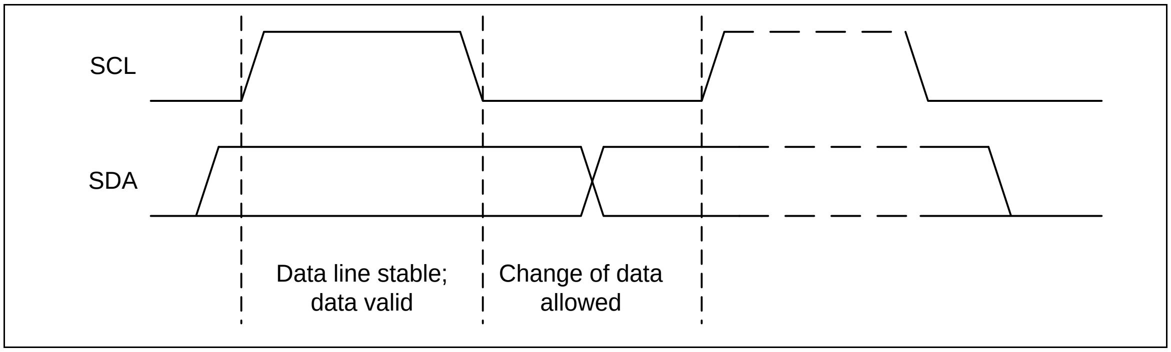


Figure 6-97 Bit Transfer on I²C Bus

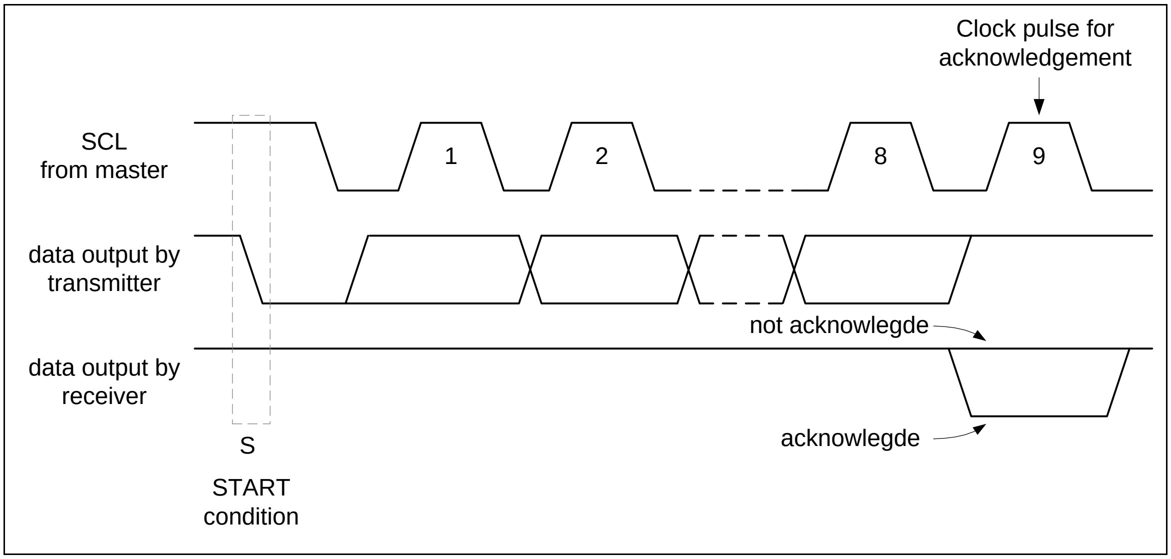


Figure 6-98 Acknowledge on I²C Bus

6.16.4 Protocol Registers

The CPU interfaces are connected to the I²C port through the following thirteen 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) and I2CTOC (Time-out counter register). All bit 31~ bit 8 of these I²C special function registers are reserved. These bits do not have any functions and are all 0 if read them back.

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. Once a new status code is generated and stored in I2CSTATUS, the I²C Interrupt Flag bit SI (I2CON [3]) will be set automatically. If the Enable Interrupt bit EI (I2CON [7]) is set at this time, the I²C interrupt will be generated. The bit field I2CSTATUS[7:3] stores the internal state code, the lowest 3 bits of I2CSTATUS are always zero and the content keeps stable until SI is cleared by software. The base address is 0x4002_0000 and 0x4012_0000.

6.16.4.1 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 bit (I2CADDRn [0]) is set the I²C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.

When the GC bit is set and the I²C is in Slave mode, it can receive the general call address by 00H after Master send general call address to I²C bus, then it will follow status of GC mode.

The I²C bus controllers support multiple address recognition with four address mask registers I2CADMn (n=0~3). 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 0, that means the received corresponding register bit should be exactly the same as address register.

6.16.4.2 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 serial interrupt flag (SI) 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 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 I2CDATA[7:0] when sending I2CDATA[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 SCL clocks, and is shifted to I2CDAT [7:0] on the rising edge of SCL clocks.

I2C Data Register:

I2CDAT.7	I2CDAT.6	I2CDAT.5	I2CDAT.4	I2CDAT.3	I2CDAT.2	I2CDAT.1	I2CDAT.0

← shifting direction ←

Figure 6-99 I^{2}C Data Shifting Direction

6.16.4.3 *Control Register (I2CON)*

The CPU can be read from and written to I2CON [7:0] directly. Two bits are affected by hardware: the SI bit is set when the I^{2}C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when ENS1 = 0.

EI Enable Interrupt.

ENS1 Set to enable I^{2}C serial function controller. When ENS1=1 the I^{2}C serial function is enabled. The function of multi-function pins must be set to I^{2}C.

STA I^{2}C START Control Bit. Set STA to logic 1 to enter Master mode, and the I^{2}C hardware sends a START or repeat START condition to bus when the bus is free.

STO I^{2}C STOP Control Bit. In Master mode, set STO to transmit a STOP condition to bus, then I^{2}C hardware will check the bus condition if a STOP condition is detected this flag will be cleared by hardware automatically. In Slave mode, set STO resets I^{2}C hardware to the defined “not addressed” Slave mode. This means it is NO LONGER in Slave Receiver mode to receive data from the master.

SI I^{2}C Interrupt Flag. When a new I^{2}C state is present in the I2CSTATUS register, the SI flag is set by hardware, and if the bit EI (I2CON [7]) is set, the I^{2}C interrupt is requested. SI must be cleared by software. Clear SI by writing 1 to this bit. All states are listed in section 5.6.6

AA Assert Acknowledge Control Bit. When AA=1 prior to address or data received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.) A slave is acknowledging the address sent from the master, 2.) The receiver devices are acknowledging the data sent by a transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line.

6.16.4.4 *Status Register (I2CSTATUS)*

I2CSTATUS [7:0] is an 8-bit read-only register. The three least significant bits are always 0. The bit field I2CSTATUS [7:3] contains the status code and there are 26 possible status codes. All states are listed in section 5.6.6. When I2CSTATUS [7:0] is F8H, no serial interrupt is requested. All other I2CSTATUS [7:3] values correspond to the defined I^{2}C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2CSTATUS[7:3] one cycle after SI set by hardware and is still present one cycle after SI reset by software.

In addition, the state 00H stands for a Bus Error, which occurs when a START or STOP condition is present at an incorrect position in the I^{2}C format frame. A Bus Error may occur during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I^{2}C from bus error, STO should be set and SI should be cleared to enter Not Addressed Slave mode. Then STO is cleared to release bus and to wait for a new communication. The I^{2}C 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 ACK	0x68	Slave Receive Arbitration Lost
0x48	Master Receive NACK	0x80	Slave Receive Data ACK
0x50	Master Receive ACK	0x88	Slave Receive Data NACK
0x58	Master Receive 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:** The status “0xF8” exists in both Master/Slave modes, and it will not raise any interrupt.		

Table 6-23 I^{2}C Status Code Description Table

6.16.4.5 I^{2}C Clock Baud Rate Bits (I2CLK)

The data baud rate of I^{2}C is determined by the I2CLK [7:0] register when I^{2}C is in Master mode, and it is not necessary in Slave mode. In Slave mode, I^{2}C will automatically synchronize it with any clock frequency from a master I^{2}C device.

The data baud rate of I^{2}C setting is Data Baud Rate of I^{2}C = (system clock) / (4x (I2CLK [7:0] +1)). If system clock = 16 MHz, the I2CLK [7:0] = 40 (28H), the data baud rate of I^{2}C = 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec.

6.16.4.6 The I^{2}C Time-out Counter Register (I2CTOC)

There is a 14-bit time-out counter which can be used to deal with the I^{2}C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows (TIF=1) and generates I^{2}C interrupt to CPU or stops counting by clearing ENTI to 0. When time-out counter is enabled, writing 1 to the SI flag will reset the counter and re-start up counting after SI is cleared. If I^{2}C bus hangs up, it causes the I2CSTATUS and flag SI are not updated for a period, the 14-bit time-out counter may overflow and acknowledge CPU the I^{2}C interrupt. Refer to Figure 6-100 for the 14-bit time-out counter. User may write 1 to clear TIF to 0.

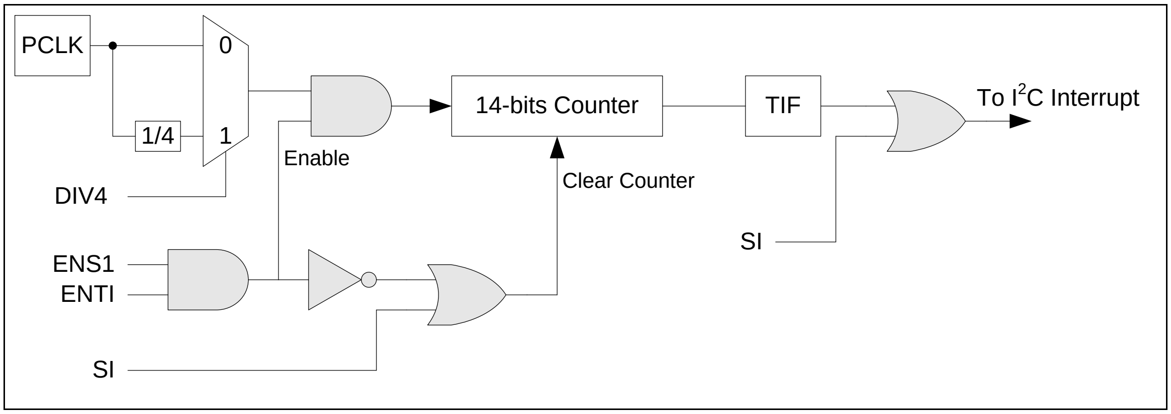


Figure 6-100: I^{2}C Time-out Count Block Diagram

6.16.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
I^2C Base Address: **I2Cx_BA = 0x4002_0000 + (0x10_0000 * x)** **x = 0, 1**				
I2CON	I2Cx_BA+0x00	R/W	I^2C Control Register	0x0000_0000
I2CADDR0	I2Cx_BA+0x04	R/W	I^2C Slave Address Register0	0x0000_0000
I2CDAT	I2Cx_BA+0x08	R/W	I^2C Data Register	0x0000_0000
I2CSTATUS	I2Cx_BA+0x0C	R	I^2C Status Register	0x0000_00F8
I2CLK	I2Cx_BA+0x10	R/W	I^2C Clock Divided Register	0x0000_0000
I2CTOC	I2Cx_BA+0x14	R/W	I^2C Time-out Counter Register	0x0000_0000
I2CADDR1	I2Cx_BA+0x18	R/W	I^2C Slave Address Register1	0x0000_0000
I2CADDR2	I2Cx_BA+0x1C	R/W	I^2C Slave Address Register2	0x0000_0000
I2CADDR3	I2Cx_BA+0x20	R/W	I^2C Slave Address Register3	0x0000_0000
I2CADM0	I2Cx_BA+0x24	R/W	I^2C Slave Address Mask Register0	0x0000_0000
I2CADM1	I2Cx_BA+0x28	R/W	I^2C Slave Address Mask Register1	0x0000_0000
I2CADM2	I2Cx_BA+0x2C	R/W	I^2C Slave Address Mask Register2	0x0000_0000
I2CADM3	I2Cx_BA+0x30	R/W	I^2C Slave Address Mask Register3	0x0000_0000
I2CWKUPCON	I2Cx_BA+0x3C	R/W	I^2C Wake-up Control Register	0x0000_0000
I2CWKUPSTS	I2Cx_BA+0x40	R/W	I^2C Wake-up Status Register	0x0000_0000

6.16.6 Register Description

I²C Control Register (I2CON)

Register	Offset	R/W	Description	Reset Value
I2CON	I2Cx_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 function of multi-function pins must be set to I²C first.
[5]	**STA**	**I²C START Control Bit** Set 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 Bit** In Master mode, set 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 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 is by writing 1 to this bit.
[2]	**AA**	**Assert Acknowledge Control Bit** When AA=1 prior to address or data received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.) A slave is acknowledging the address sent from master, 2.) The receiver devices are acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on

		the SCL line.
[1:0]	**Reserved**	Reserved.

I^2C Data Register (I2CDAT)

Register	Offset	R/W	Description	Reset Value
I2CDAT	I2Cx_BA+0x08	R/W	I^2C 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**	**I^2C Data Register** Bit [7:0] is located with the 8-bit transferred data of I^2C serial port.

I^2C Status Register (I2CSTATUS)

Register	Offset	R/W	Description	Reset Value
I2CSTATUS	I2Cx_BA+0x0C	R	I^2C 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**	**I^2C Status Register** The status register of I^2C: The three least significant bits are always 0. The five most significant bits contain the status code. There are 26 possible status codes. When I2CSTATUS contains F8H, no serial interrupt is requested. All other I2CSTATUS values correspond to defined I^2C states. When each of these states is entered, a status interrupt is requested (SI = 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 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an incorrect position in the I^2C formation frame. A bus error may occur 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	I2Cx_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	I2Cx_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 = Time-out counter Disabled. 1 = Time-out counter Enabled. When Enabled, the 14-bit time-out counter will start counting when SI is cleared. Writing 1 to the SI flag will reset counter and re-start up counting after SI is cleared.
[1]	**DIV4**	**Time-out Counter Input Clock Is Divided by 4** 0 = The time-out counter input clock divided by 4 Disabled. 1 = The time-out counter input clock divided by 4 Enabled. When Enabled, the time-out period is extended 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) is set to 1. **Note:** This bit can be cleared by writing '1' to it.

I^2C Slave Address Register (I2CADDRx)

Register	Offset	R/W	Description	Reset Value
I2CADDR0	I2Cx_BA+0x04	R/W	I^2C Slave Address Register0	0x0000_0000
I2CADDR1	I2Cx_BA+0x18	R/W	I^2C Slave Address Register1	0x0000_0000
I2CADDR2	I2Cx_BA+0x1C	R/W	I^2C Slave Address Register2	0x0000_0000
I2CADDR3	I2Cx_BA+0x20	R/W	I^2C 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**	**I^2C Address Register** The content of this register is irrelevant when I^2C is in Master mode. In Slave mode, the seven most significant bits must be loaded with the chip's own address. The I^2C hardware will react if one of the addresses 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	I2Cx_BA+0x24	R/W	I²C Slave Address Mask Register0	0x0000_0000
I2CADM1	I2Cx_BA+0x28	R/W	I²C Slave Address Mask Register1	0x0000_0000
I2CADM2	I2Cx_BA+0x2C	R/W	I²C Slave Address Mask Register2	0x0000_0000
I2CADM3	I2Cx_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 exactly the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don't care.). I²C bus controller supports multiple address recognition with four address mask registers. 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 0, that means the received corresponding register bit should be exactly the same as address register.
[0]	**Reserved**	Reserved.

I²C Wake-up Control Register (I2CWKUPCON)

Register	Offset	R/W	Description	Reset Value
I2CWKUPCON	I2Cx_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 Function Enable Bit** 0 = I²C wake-up function Disabled. 1 = I²C wake-up function Enabled.

I^2C Wake-up Status Register (I2CWKUPSTS)

Register	Offset	R/W	Description	Reset Value
I2CWKUPSTS	I2Cx_BA+0x40	R/W	I^2C 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**	**I^2C Wake-up Interrupt Flag** When chip is woken up from Power-down mode by I^2C, this bit is set to 1. **Note:** This bit can be cleared by writing '1' to it.

6.16.7 Operation Modes

The on-chip I^2C ports support five operation modes, Master Transmitter, Master Receiver, Slave Transmitter, Slave Receiver, and GC Call.

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 bit 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.

Bits STA, STO and AA in I2CON register will determine the next state of the I^2C hardware after SI flag is cleared. Upon completion of the new action, a new status code will be updated 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.

In the following figure about the five operation modes, detailed data flow is represented. The legend for those data flow figures is shown in Figure 6-101

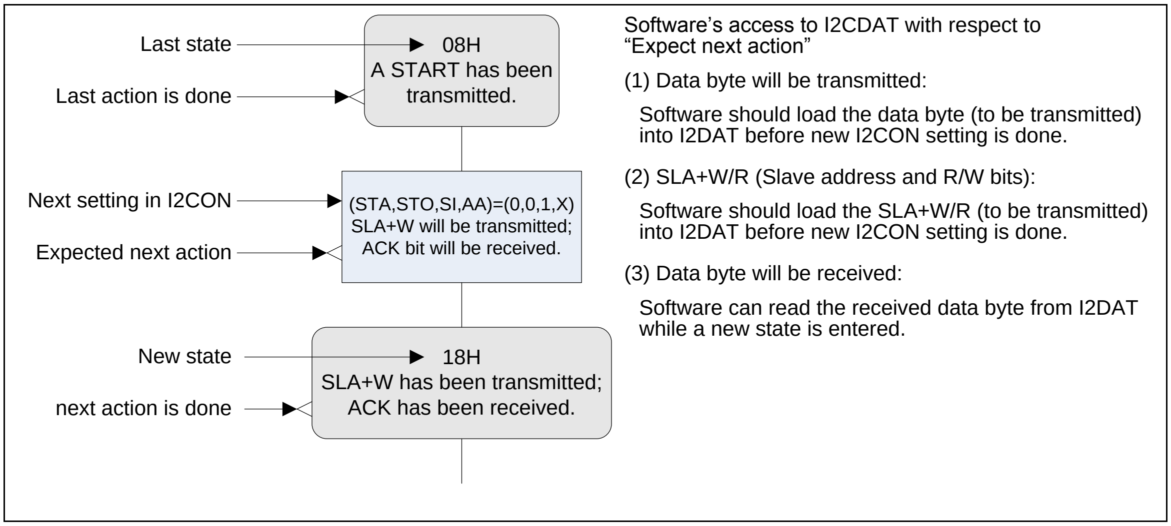


Figure 6-101 Legend for the Following Five Figures

6.16.7.1 Master Transmitter Mode

As shown in Figure 6-102, in Master Transmitter mode, serial data is output through SDA while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7-bit) and the data direction bit. In this case the data direction bit (R/W) will be logic 0, and it is represented by "W" in Figure 6-102. Thus the first byte transmitted is SLA+W. Serial data is transmitted 8-bit at a time. After each byte is transmitted, an acknowledge bit is received. START and STOP conditions are output to indicate the beginning and the end of a serial transfer.

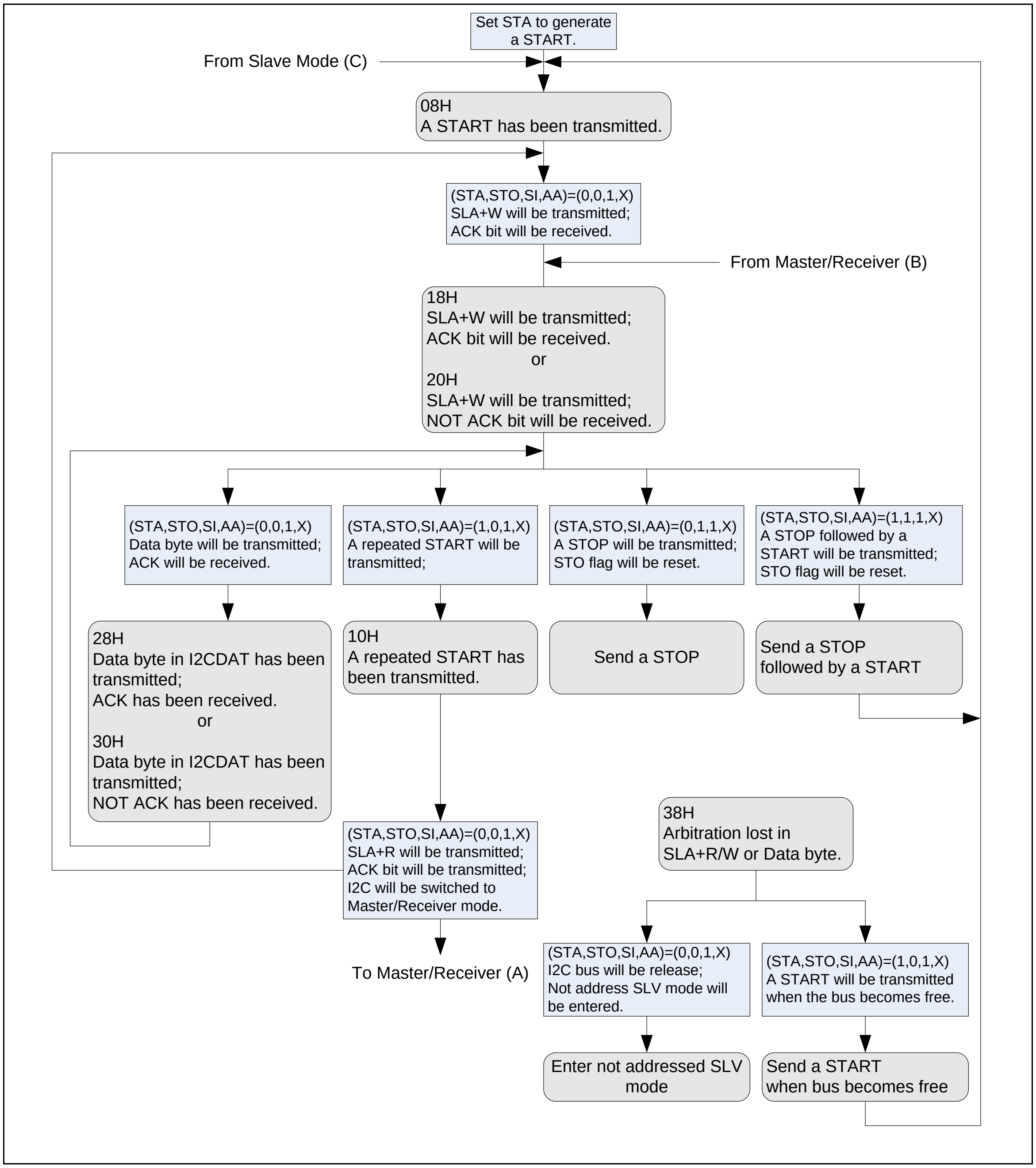


Figure 6-102 Master Transmitter Mode

6.16.7.2 Master Receiver Mode

As shown in Figure 6-103, in this case the data direction bit (R/W) will be logic 1, and it is represented by “R” in Figure 6-103. Thus the first byte transmitted is SLA+R. Serial data is received via SDA while SCL outputs the serial clock. Serial data is received 8-bit at a time. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are output to indicate the beginning and end of a serial transfer.

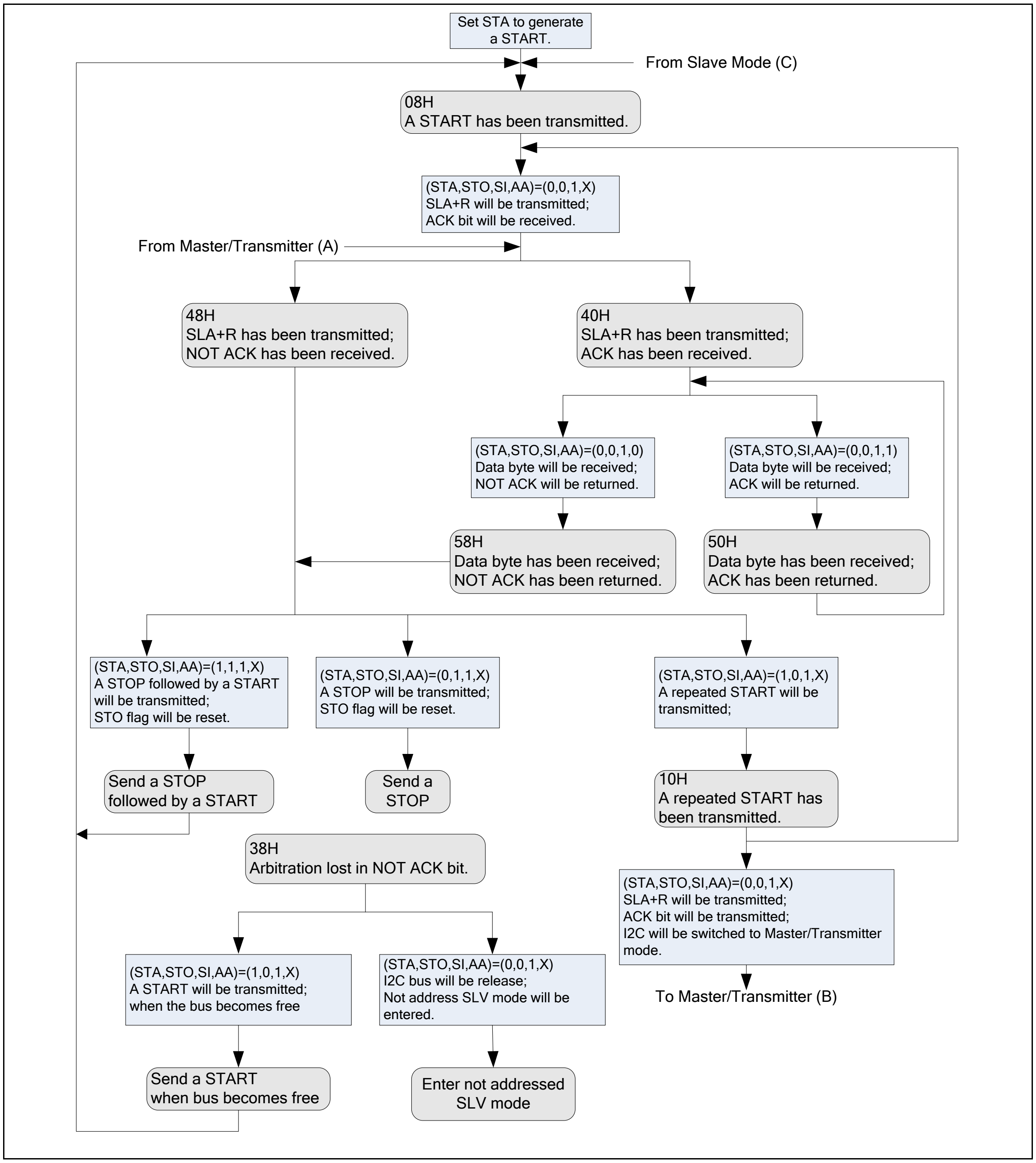


Figure 6-103 Master Receiver Mode

6.16.7.3 Slave Receiver Mode

As shown in Figure 6-104, serial data and the serial clock are received through SDA and SCL. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and direction bit.

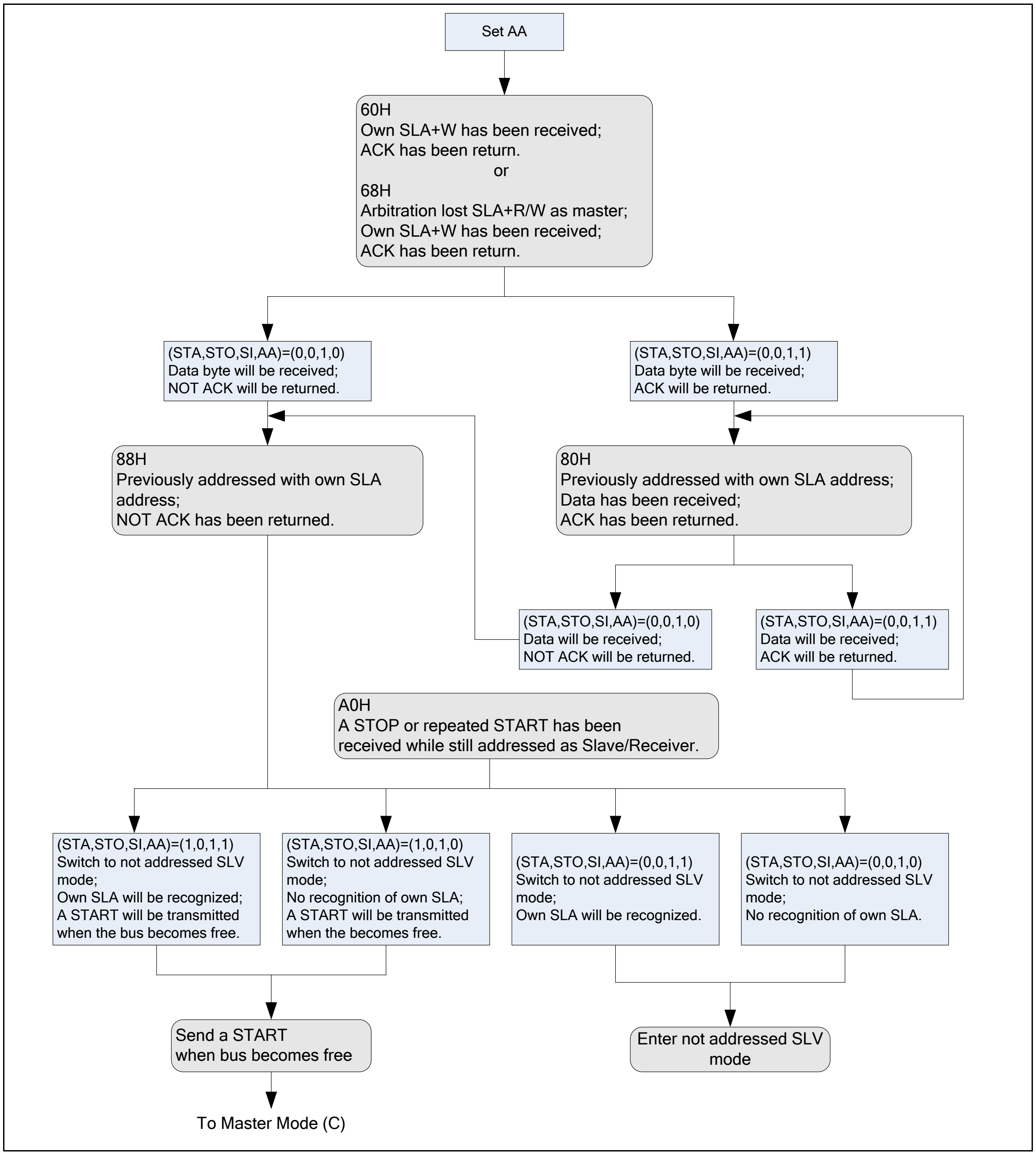


Figure 6-104 Slave Receiver Mode

6.16.7.4 Slave Transmitter Mode

As shown in Figure 6-105, the first byte is received and handled as in the slave receiver mode. However, in this mode, the direction bit will indicate that the transfer direction is reversed. Serial data is transmitted via SDA while the serial clock is input through SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer.

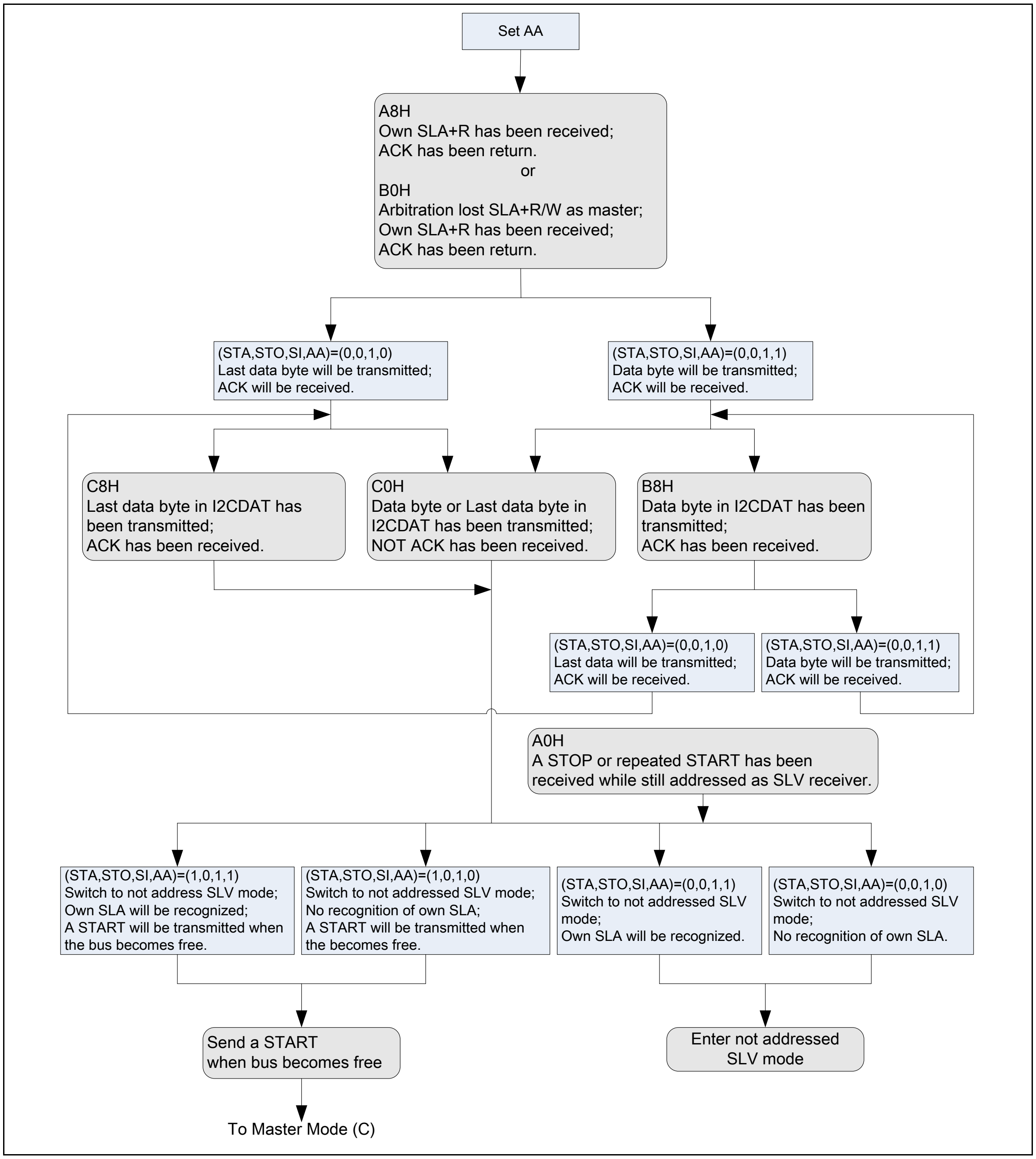


Figure 6-105 Slave Transmitter Mode

6.16.7.5 General Call (GC) Mode

As shown in Figure 6-106, if the GC bit (I2CADDRn [0]) is set, the I^2C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function. When GC bit is set and the I^2C is in Slave mode, it can receive the general call address by 00H after Master send general call address to I^2C bus, then it will follow status of GC mode. Serial data and the serial clock are received through SDA and SCL. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and direction bit.

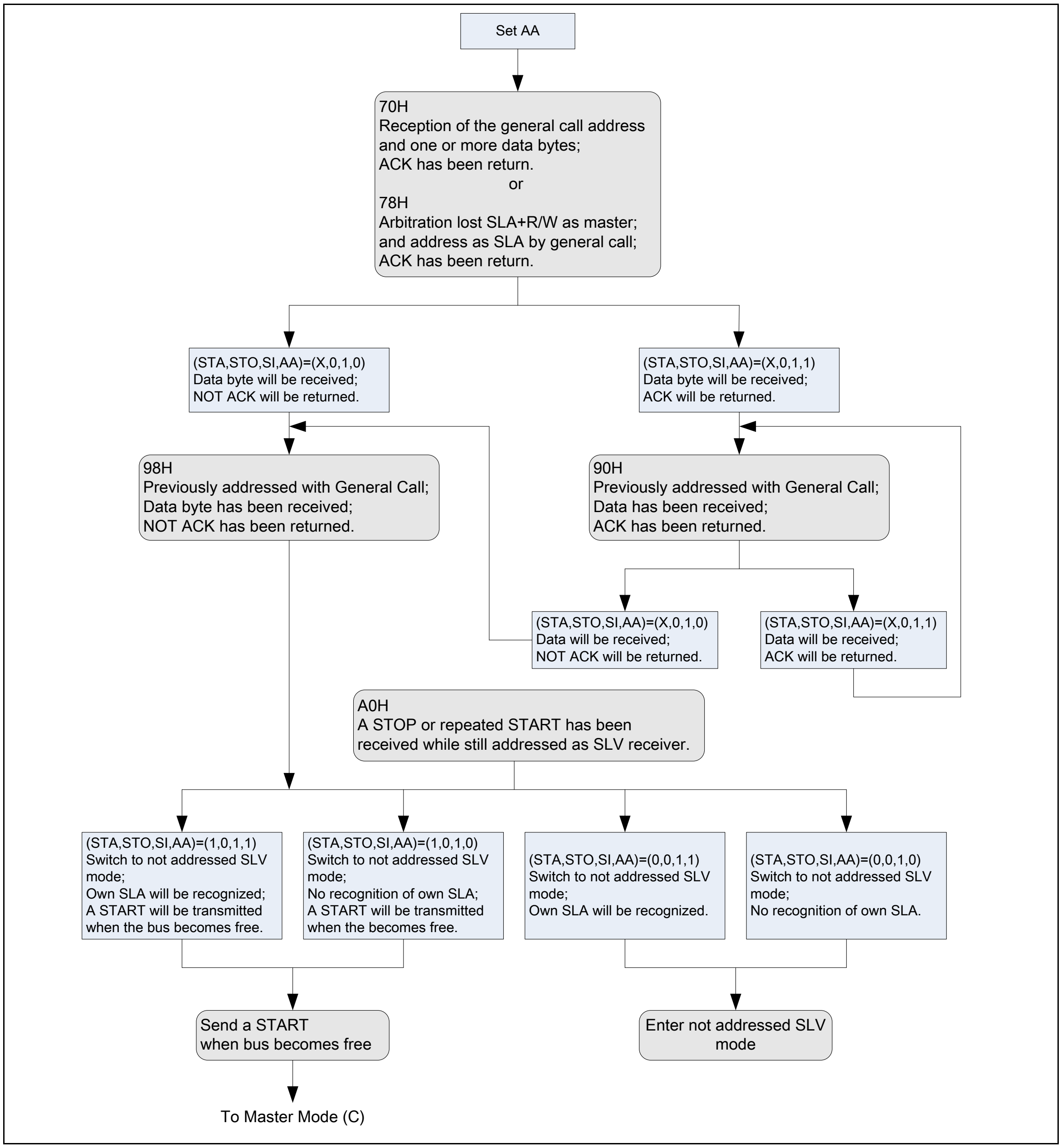


Figure 6-106 GC Mode

6.16.7.6 Example for Random Read on EEPROM

The following steps are used to configure the I^{2}C related registers when using I^{2}C to read data from EEPROM.

1. Set the multi-function pin in the “GPA_MFP” & “ALT_MFP” registers as I^{2}C.
2. Enable I^{2}C APB clock, I2C0_EN=1 and I2C1_EN=1, in the “APBCLK” register.
3. Set I2C0_RST=1 and I2C1_RST=1 to reset I^{2}C controller then set I^{2}C controller to normal operation, I2C0_RST=0 and I2C1_RST=0, in the “IPRSTC2” register
4. Set ENS1=1 to enable I^{2}C controller in the “I2CON” register
5. Give I^{2}C clock a divided register value for I^{2}C clock rate in the “I2CLK”
6. Set SETENA=0x00040000 in the “NVIC_ISER” register to set I^{2}C IRQ
7. Set EI=1 to enable I^{2}C Interrupt in the “I2CON” register
8. Set I^{2}C 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. This operation has two processes in below example. One is “Transmitter Operation”, and another one is “Receive Operation”. Figure 6-107 shows the EEPROM random read operation.

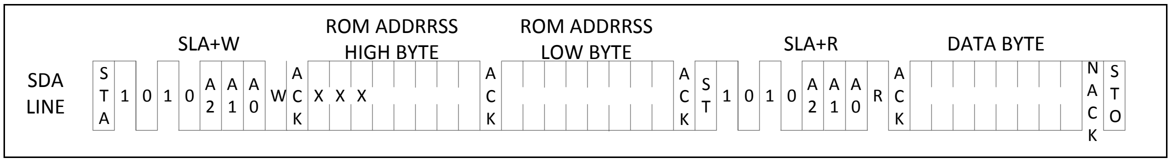


Figure 6-107 Random Read of EEPROM

- **Transmitter Operation**

The I^{2}C controller sends a SLA(Slave address)+W to EERPOM first, then sends a word length (16 bits) data address to set the inside data pointer. In this operation steps can refer to Figure 6-108. The Transmitter Operation steps are as follows:

1. Send a START bit, and wait I2CSTATUS becomes to 0x08

   --- (STA, STO, SI, AA) = (1, 0, 1, X)

2. When I2CSTAUS = 0x08, the START bit has been transmitted. Then write a 7-bit EEPROM slave address with control bit (Write) to I2CDAT register and wait a response of ACK (Acknowledge bit) back from EEPROM.

   --- I2C0DAT=SLA+W, and clear SI (STA, STO, SI, AA) = (0, 0, 1, X);

3. When I2CSTAUS = 0x18, an ACK is received. Then write data pointer high byte to EEPROM and wait a response of ACK. If I2CSTATUS=0x20, an NACK is received it means that the EEPROM didn’t get correct address or EEPROM is not exist. In this situation, send a STOP to terminate this communication.

   --- Get ACK (0x18): I2CDAT=Data pointer (Hi Byte) and clear SI (STA, STO, SI, AA) = (0,0,1,X);

   --- Get NACK (0x20): (STA, STO, SI, AA) = (0, 1, 1, X)

4. When I2CSTATUS=0x28, an ACK is received. Send data pointer low byte to EEPROM and wait next ACK back. If I2CSTATUS=0x30, I^{2}C should be terminate this

communication.

--- Get ACK (0x28): I2CDATA=Data pointer (Low Byte) and clear SI (STA, STO, SI, AA) = (0, 0, 1, X);

--- Get NACK(0x30): (STA, STO, SI, AA) = (0, 1, 1, X)

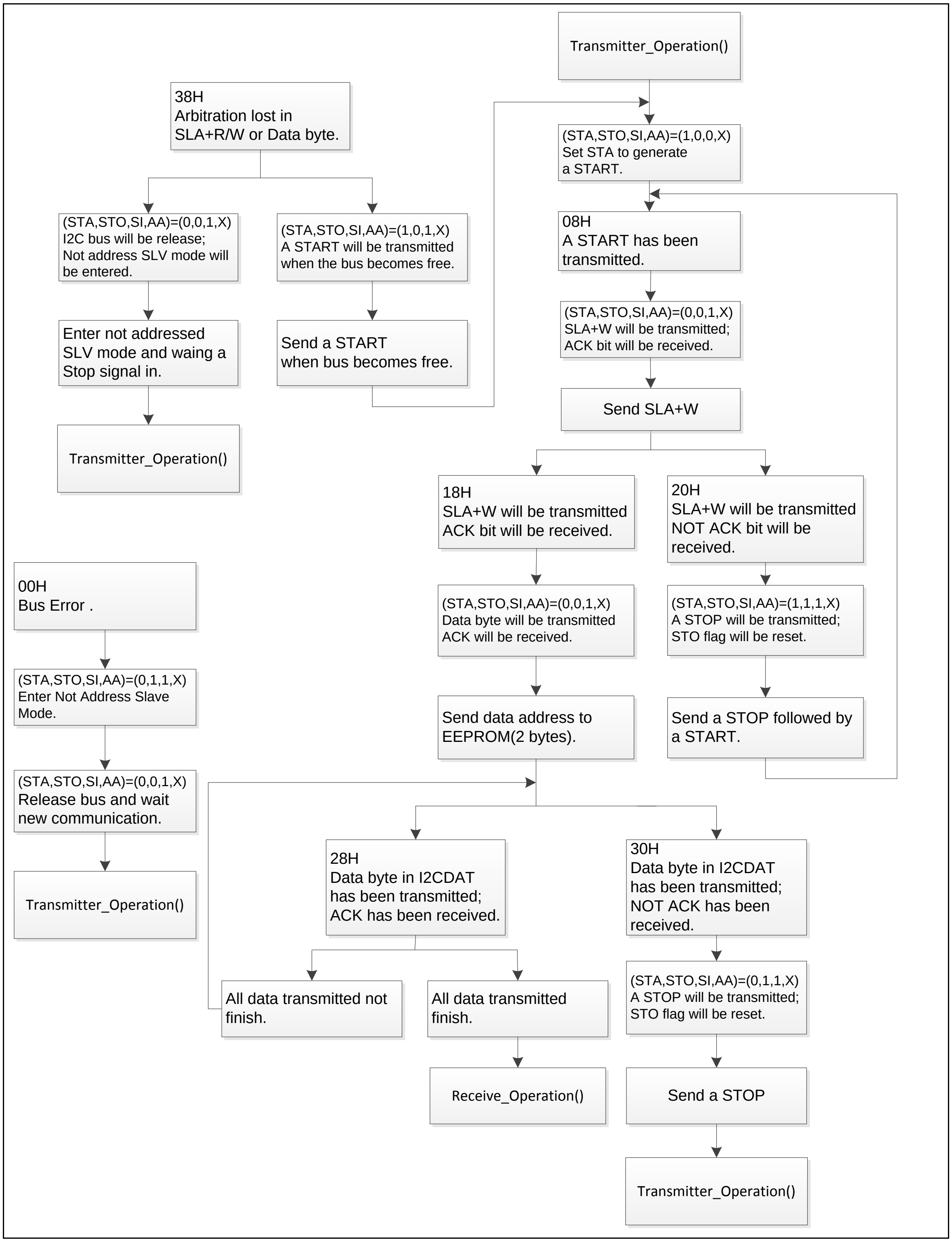


Figure 6-108 Transmitter Operation of Random Read

- **Receive Operation**

Another one process of random read operation is “Receive Operation”. The I^{2}C controller sends a SLA+R to EERPOM and then gets a data byte back where the data pointer point to. Receive operation is continuation of the “Transmitter Operation”. Refer to the following figure for the steps:

The Receive Operation steps are as follows:

1. When I2CSTATUS=0x28, send a RESTART bit. If I2CSTATUS=0x30, I^{2}C should be terminate this communication.

   --- Get ACK (0x28): (STA, STO, SI, AA) = (1, 0, 1, X)

   --- Get NACK (0x30): (STA, STO, SI, AA) = (0, 1, 1, X)

2. When I2CSTATUS=0x10, write an 7-bit EEPROM slave address with a control bit (Read) to I2CDATA register.

   --- I2CDATA=SLA+R, and clear SI (STA, STO, SI, AA) = (0, 0, 1, X)

3. When I2CSTATAUS=0x40, send a NACK. At this status, I^{2}C controller will send 8 clocks to receive EEPROM data back and at ninth clock send NACK out to EEPROM.

   --- (STA, STO, SI, AA) = (0, 0, 1, 0)

4. When I2CSTATUS=0x58, data is ready received, then send a STOP bit to terminate this communication.

   --- (STA, STO, SI, AA) = (0, 1, 1, 0)

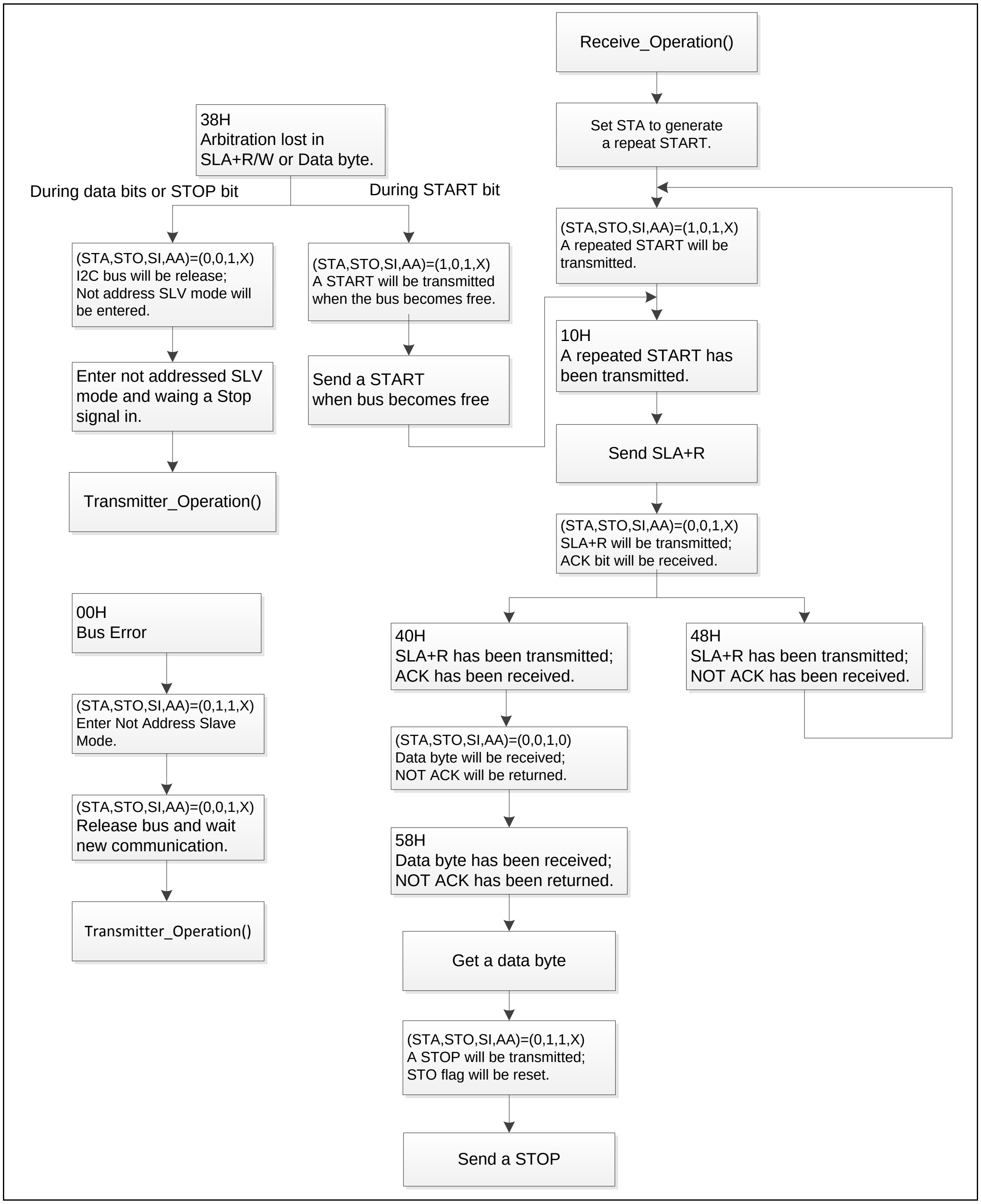


Figure 6-109 Receive Operation of Random Read

- **Other Events (Multi-Master)**

In some applications, there are two or more master microcontrollers on the same I²C bus to access slaves, and the masters maybe transmit data simultaneously. The NuMicro® microcontroller supports true multi-master bus including collision detection and arbitration to prevent data corruption.

1. When I2CSTATUS=0x38, an "Arbitration Lost" is received. Arbitration lost event maybe

occur during the send START bit, data bits or STOP bit. In the process the START bit receive this status, I^2C can send start to try again. Exception during START bit, I^2C sends a STOP to terminate this communication.

--- During START bit, (STA, STO, SI, AA) = (1, 0, 1, X)

--- During data bits or STOP bit, (STA, STO, SI, AA) = (0, 0, 1, X)

2. When I2CSTATUS=0x00, a “Bus Error” is received. To recover I^2C bus from a bus error, STO should be set and SI should be cleared, and then STO is cleared to release bus.

--- Send (STA, STO, SI, AA) = (0, 1, 1, X) first then send (STA, STO, SI, AA) = (0, 0, 1, X)

6.17 Serial Peripheral Interface (SPI)

6.17.1 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® NUC100 series contains up to four sets of SPI controllers performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. Each set of SPI controller can be configured as a master or a slave device.

The SPI controller supports the variable serial clock function for special applications and 2-bit Transfer mode to connect 2 off-chip slave devices at the same time. This controller also supports the PDMA function to access the data buffer and also supports Dual I/O Transfer mode.

6.17.2 Features

- Up to four sets of SPI controllers
- Supports Master or Slave mode operation
- Supports 2-bit Transfer mode
- Supports Dual I/O Transfer mode
- Configurable bit length of a transfer word from 8 to 32-bit
- 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 serial clock frequency in Master mode
- Supports PDMA transfer
- Supports 3-wire, no slave select signal, bi-direction interface

6.17.3 Block Diagram

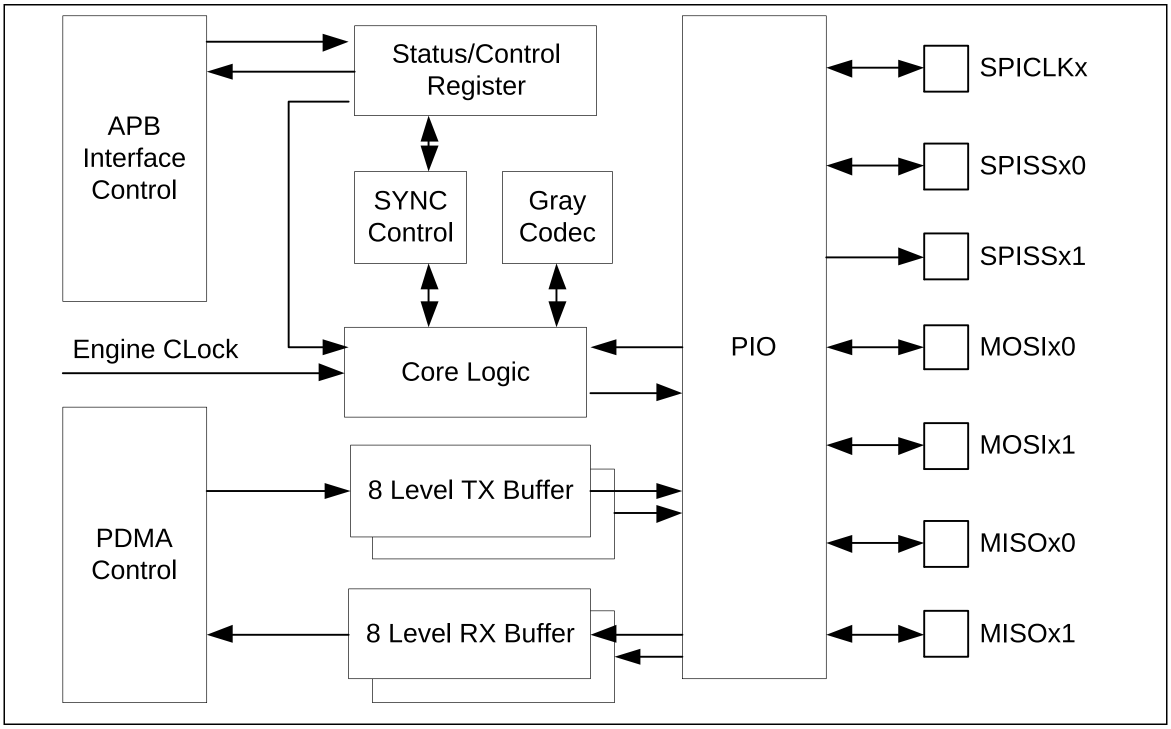


Figure 6-110 SPI Block Diagram

6.17.4 Functional Description

SPI Peripheral clock and SPI Serial Clock

The SPI controller needs the SPI peripheral clock to drive the SPI logic unit to perform the data transfer. The SPI peripheral clock rate is determined by the settings of clock source, BCn option and clock divisor. The SPIx_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. The DIVIDER setting of SPI_DIVIDER register determines the divisor of the clock rate calculation.

In Master mode, if the variable clock function is disabled, the output frequency of the SPI serial clock output pin is equal to the SPI peripheral clock rate. In general, the SPI serial clock is denoted as SPI clock. In Slave mode, the SPI serial clock is provided by an off-chip master device. The SPI peripheral clock rate of slave device must be faster than the SPI serial 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 the SLAVE bit (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.

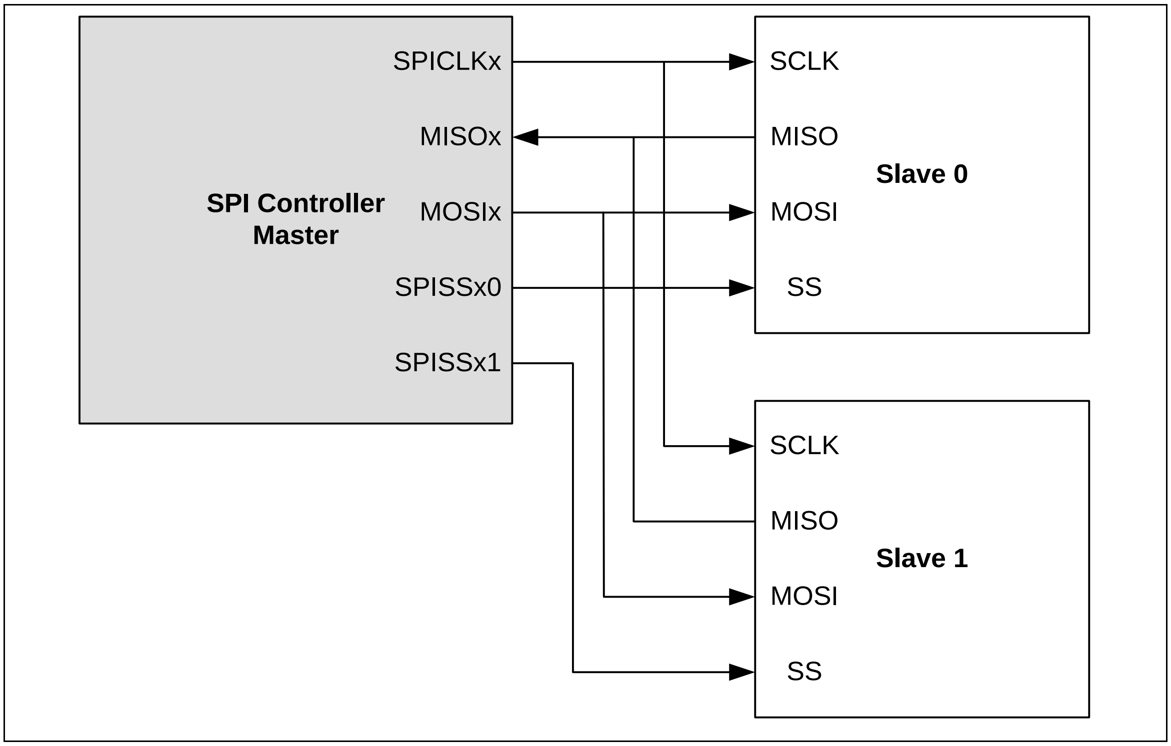


Figure 6-111 SPI Master Mode Application Block Diagram

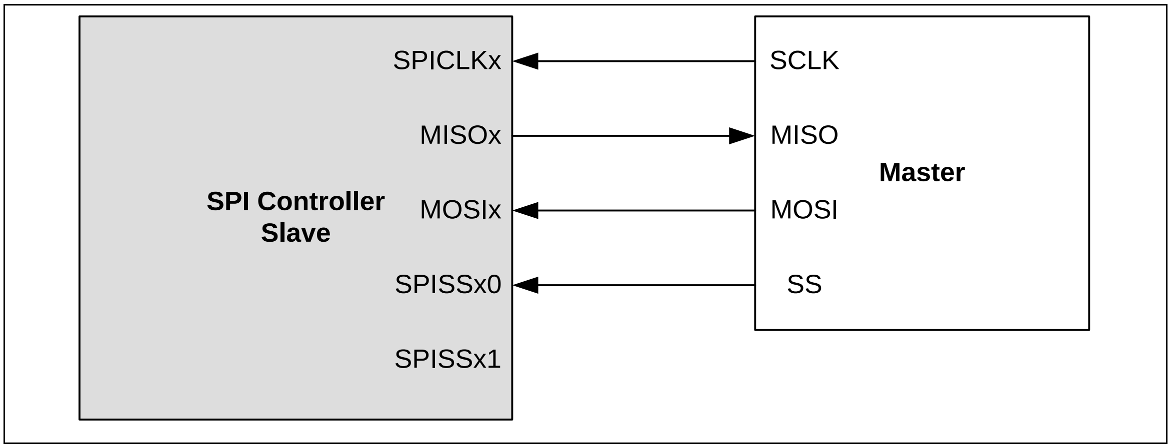


Figure 6-112 SPI Slave Mode Application Block Diagram

Slave Selection

In Master mode, this SPI controller can drive up to two off-chip slave devices through the slave select output pins SPISSx0 and SPISSx1. In Slave mode, the off-chip master device drives the slave select signal from the SPISSx0 input port to this SPI controller. In Master/Slave mode, the active state of slave select signal can be programmed to low or high active in SS_LVL bit (SPI_SSR[2]), and the SS_LTRIG bit (SPI_SSR[4]) defines the slave select signal SPISSx0/1 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 bit (SPI_SSR[5]) is used to indicate if the received bits among one transaction meets the requirement defined in TX_BIT_LEN.

Level-trigger/Edge-trigger

In Slave mode, the slave select signal can be configured as level-trigger or edge-trigger. For edge-trigger, the data transfer starts from an active edge and ends on an inactive edge. 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 following two conditions will terminate the transfer procedure and the unit transfer interrupt flag of slave will be set. 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 see if the data has been completely transferred.

Automatic Slave Selection

In Master mode, if the bit AUTOSS (SPI_SSR[3]) is set, the slave select signals will be generated automatically and output to the SPISSx0 and SPISSx1 pins according to whether SSR[0] (SPI_SSR[0]) and SSR[1] (SPI_SSR[1]) are enabled or not. This means that the slave select signals, which are selected in SSR[1:0], 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 manual setting/clearing the related bits of SPI_SSR[1:0]. The active state of the slave select output signals is specified in SS_LVL bit (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.

Variable Serial Clock Frequency

In Master mode, if the VARCLK_EN bit (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. 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 clock, the VARCLK, the DIVIDER and the DIVIDER2 registers.

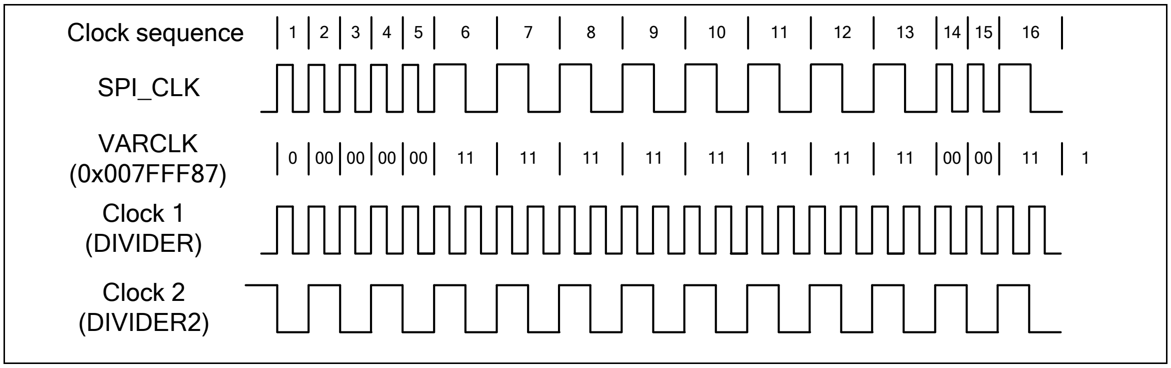


Figure 6-113 Variable Serial Clock Frequency

Clock Polarity

The CLKP bit (SPI_CNTRL[11]) defines the SPI clock idle state. If CLKP = 1, the output SPI clock is idle at high state. If CLKP = 0, it is idle at low state.

Transmit/Receive Bit Length

The bit length of a transaction word is defined in TX_BIT_LEN bit field (SPI_CNTRL[7:3]) and can be configured up to 32-bit length in a transaction word for transmitting and receiving.

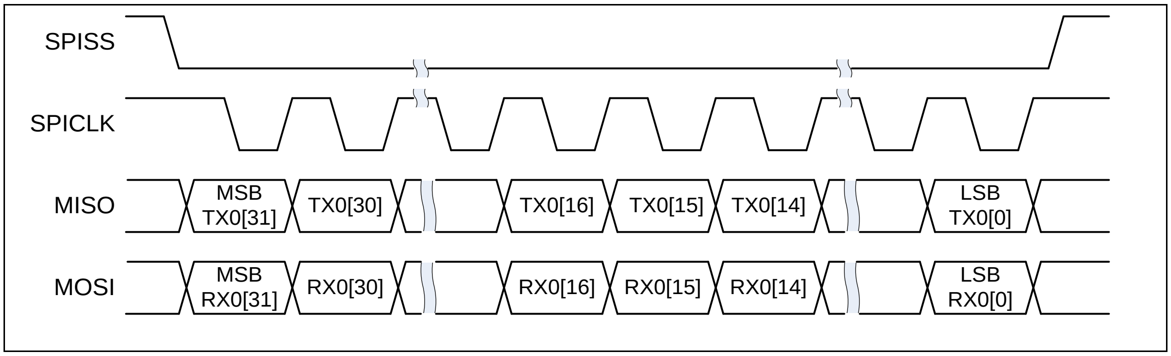


Figure 6-114 32-Bit in One Transaction

LSB/MSB First

The LSB bit (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

The TX_NEG bit (SPI_CNTRL[2]) defines the data transmitted out either on negative edge or on positive edge of SPI clock.

Receive Edge

The Rx_NEG bit (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 at the same clock edge.

Word Suspend

The four bit fields of 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 interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SP_CYCLE is 0x3 (3.5 SPI clock cycles). This SP_CYCLE setting will not take effect to the word suspend interval if FIFO mode is disabled by software.

If both the VARCLK_EN, SPI_CNTRL[23], and the FIFO bit, SPI_CNTRL[21], are set as 1, the minimum word suspend period is (6.5 + SP_CYCLE)*SPI clock period.

Byte Reorder

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 serial clock function is enabled.

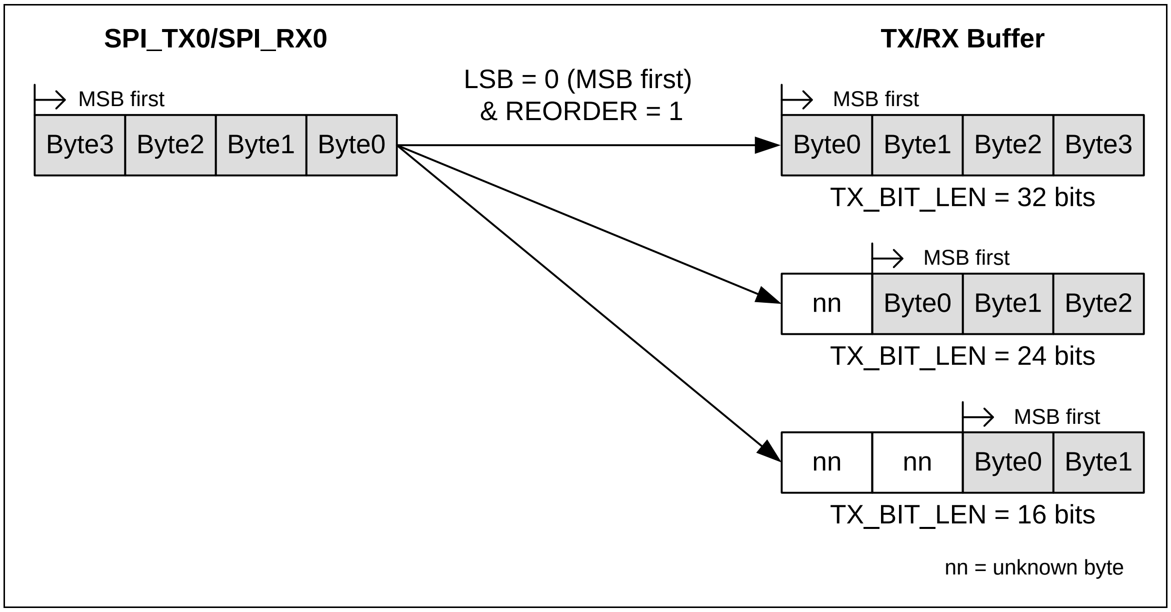


Figure 6-115 Byte Reorder Function

Byte Suspend

In Master mode, if 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.

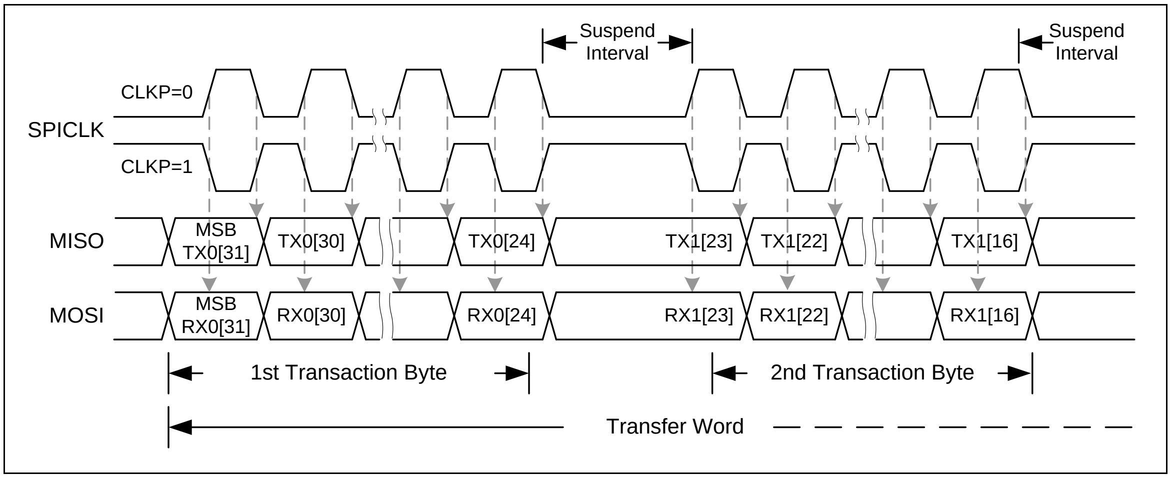


Figure 6-116 Timing Waveform for Byte Suspend

3-Wire Mode

When the NOSLVSEL bit 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, SPICLK, SPI_MISO, and SPI_MOSI, are required to communicate with a SPI master. The SPISS 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. In Slave 3-wire mode, the SS_LTRIG, SPI_SSR[4], should be set as 1.

2-Bit Mode

The SPI controller also supports 2-bit Transfer mode when setting the TWOB bit (SPI_CNTRL[22]) to 1. In 2-bit mode, the SPI controller performs full duplex data transfer. In other words, the 2-bit serial data can be transmitted and received simultaneously.

For example, in Master mode, the data stored in the SPI_TX0 and SPI_TX1 register will be transmitted through the MOSIx0 and MOSIx1 pin respectively. In the meanwhile, the SPI_RX0 and SPI_RX1 will store the data received from MISOx0 pin and MISOx1 pin respectively.

In Slave mode, the data stored in the SPI_TX0 and SPI_TX1 register will be transmitted through the MISOx0 and MISOx1 pin respectively. In the meanwhile, the SPI_RX0 and SPI_RX1 will store the data received from the MOSIx0 and MOSIx1 pin respectively.

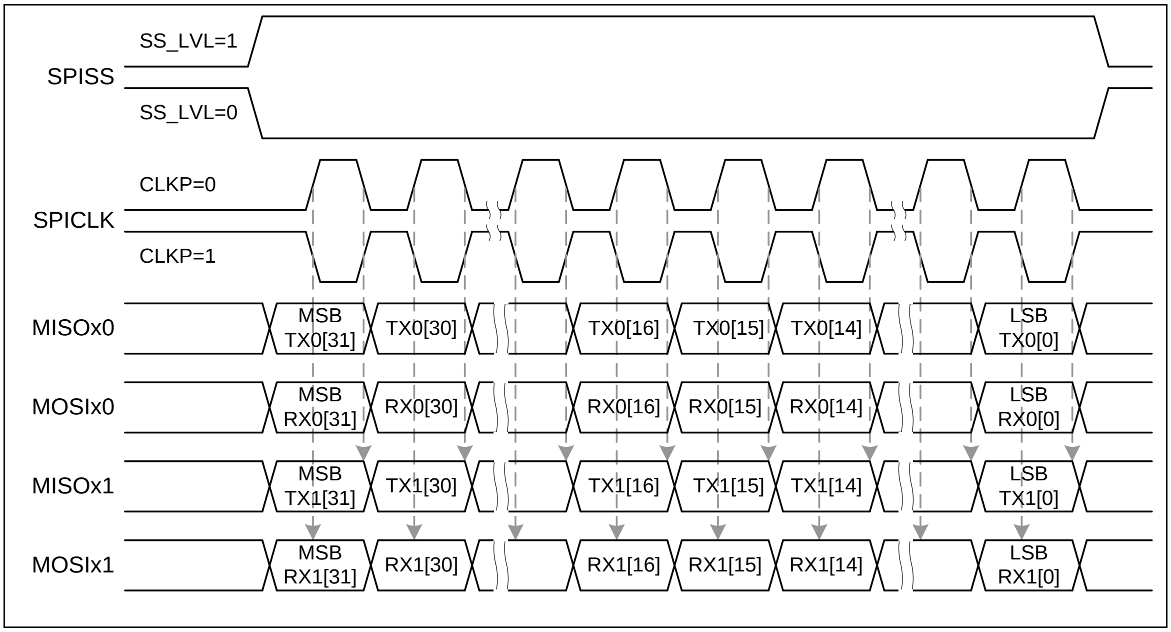


Figure 6-117 2-Bit Mode (Slave Mode)

Dual I/O Mode

The SPI controller also supports dual I/O transfer when setting the DUAL_IO_EN bit (SPI_CNTRL2[13]) to 1. Many general SPI flashes support dual I/O transfer. The DUAL_IO_DIR bit (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-bits of bit length.

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 MOSI0 is the even bit data output and the 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 MISO0 and MOSI0 will be set as data input ports.

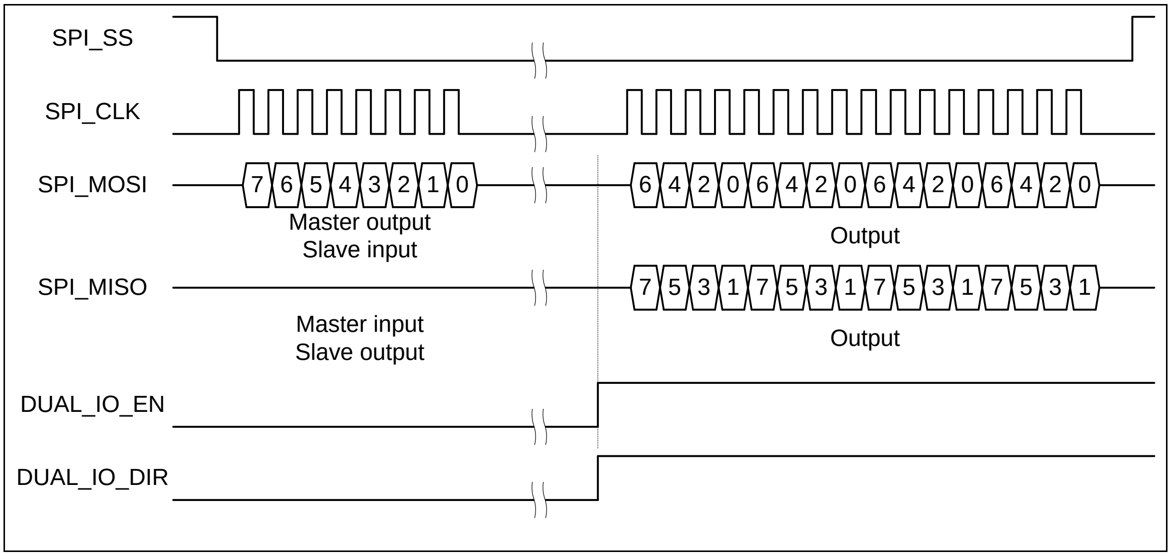


Figure 6-118 Bit Sequence of Dual Output Mode

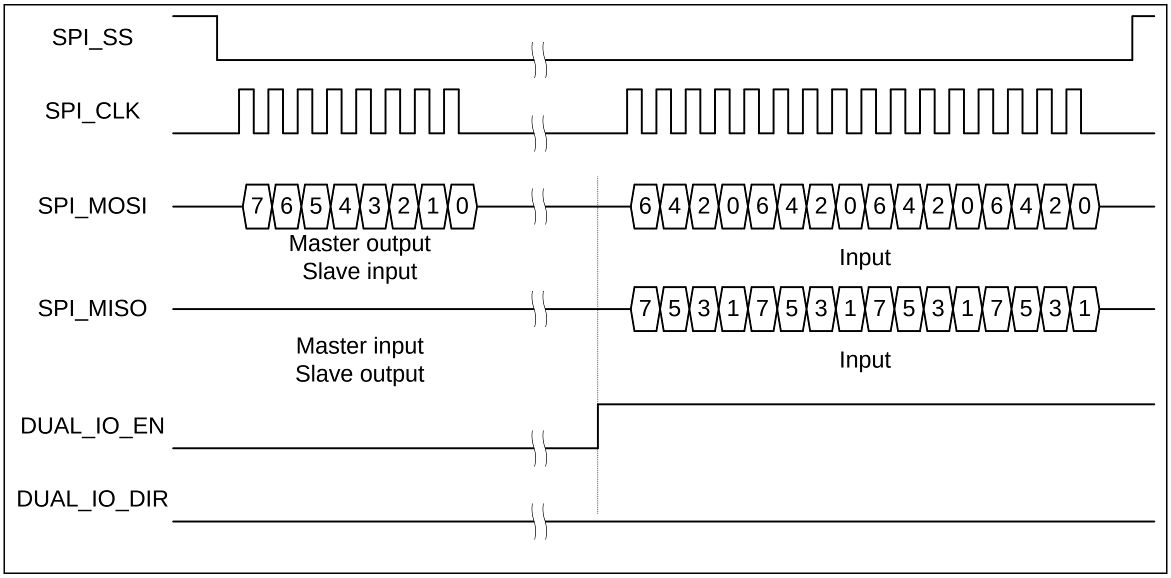


Figure 6-119 Bit Sequence of Dual Input Mode

FIFO Mode

The SPI controller supports FIFO mode when the FIFO bit in SPI_CNTRL[21] is set as 1. The SPI controllers are equipped 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.

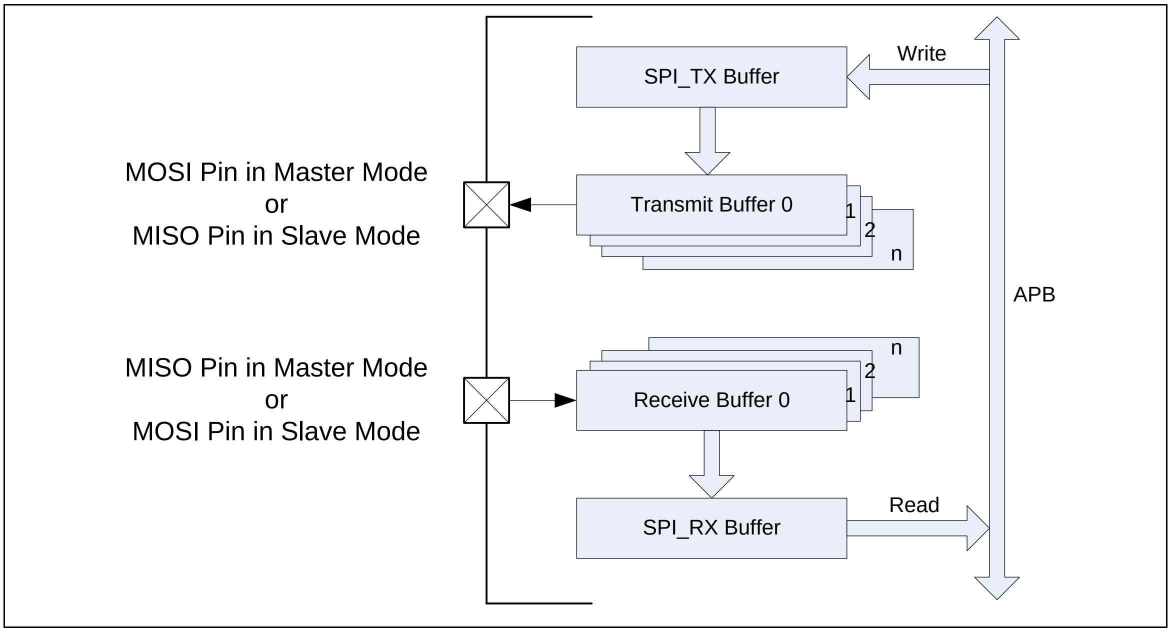


Figure 6-120 FIFO Mode Block Diagram

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 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 MOSI0/1 pin and stored to SPI_RX0 register. The reception mechanism is similar to Master mode reception operation.

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.

6.17.5 Timing Diagram

The active state of slave select signal can be defined by setting the SS_LVL bit (SPI_SSR[2]) and SS_LTRIG bit (SPI_SSR[4]). The SPI clock which is in idle state can be configured as high or low state by setting the CLKP bit (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 bit (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.

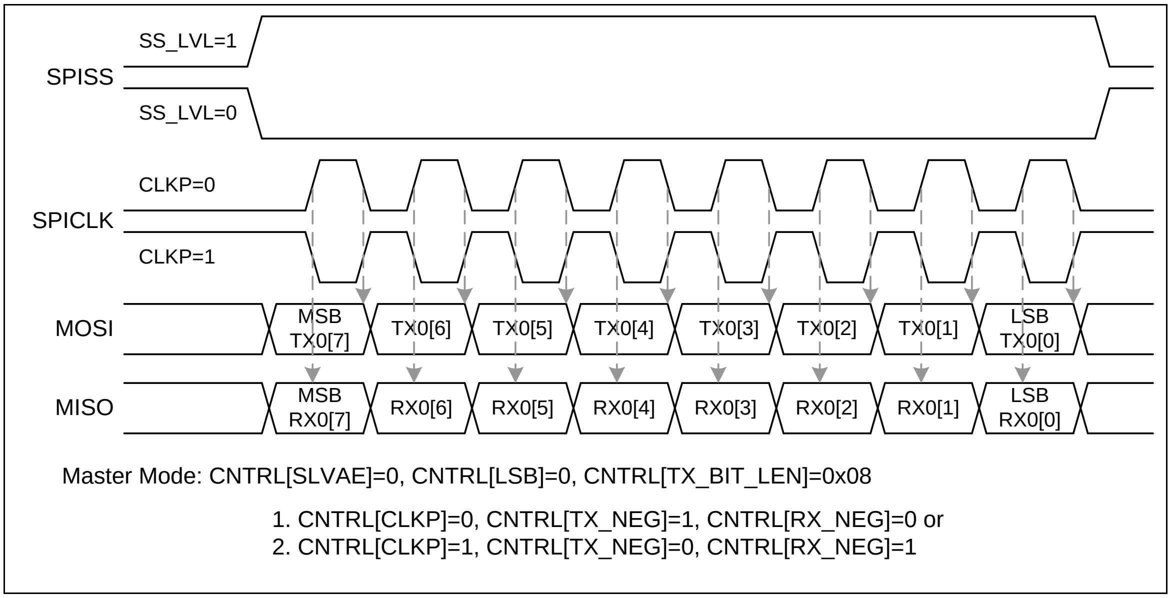


Figure 6-121 SPI Timing in Master Mode

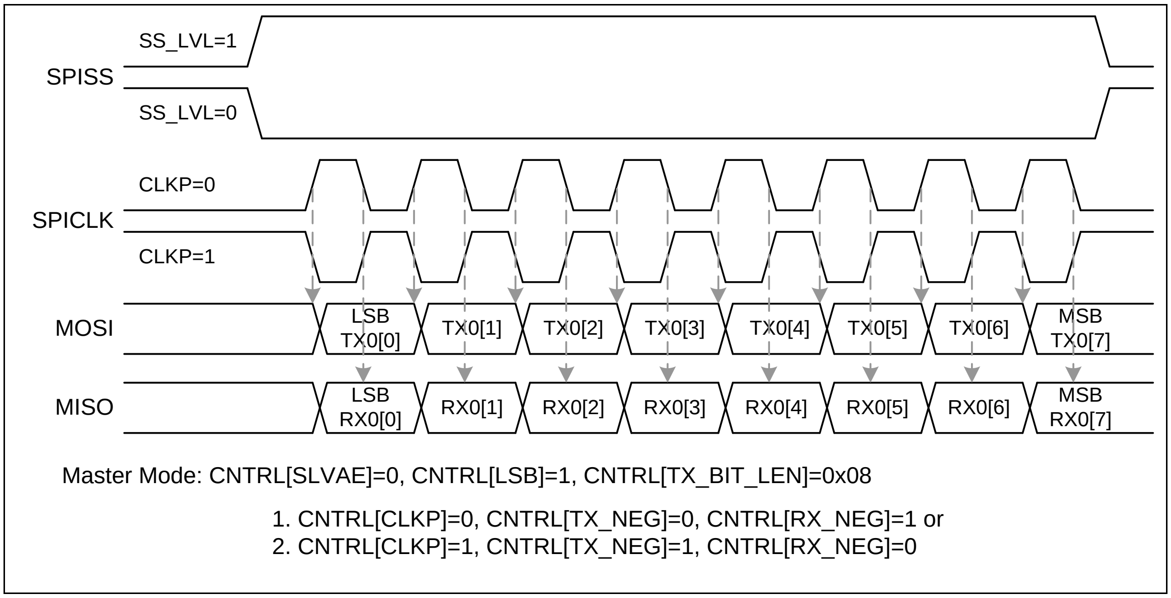


Figure 6-122 SPI Timing in Master Mode (Alternate Phase of SPICLK)

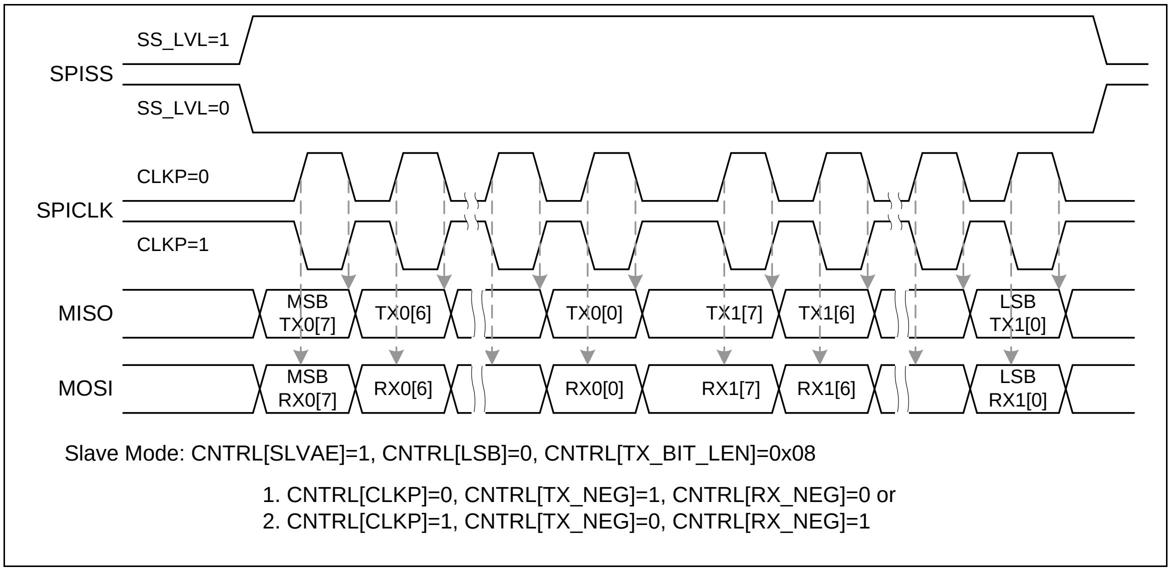


Figure 6-123 SPI Timing in Slave Mode

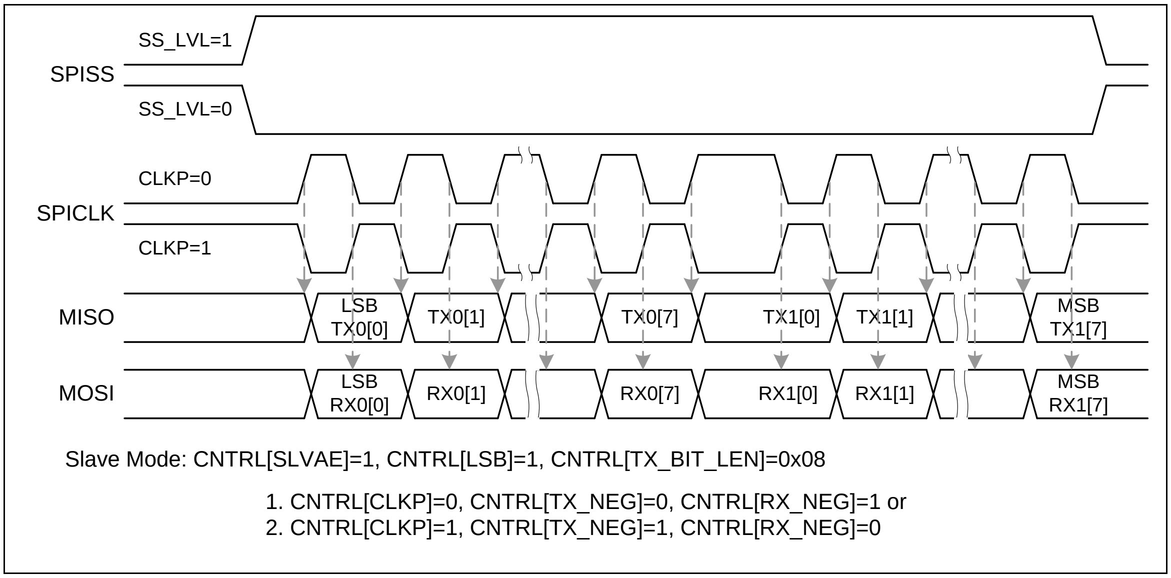


Figure 6-124 SPI Timing in Slave Mode (Alternate Phase of SPICLK)

6.17.6 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.
- SPICLK 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.

2) Set the DIVIDER (SPI_DIVIDER [7:0]) register to determine the output frequency of SPI clock.

3) Write the SPI_SSR register a proper value for the related settings of Master mode:
   1. Disable the Automatic Slave Select bit AUTOSS(SPI_SSR[3] = 0).
   2. Select low level trigger output of slave select signal in the Slave Select Active Level bit SS_LVL (SPI_SSR[2] = 0) and Slave Select Level Trigger bit SS_LTRIG (SPI_SSR[4] = 1).
   3. Select slave select signal to be output active at the I/O pin by setting the Slave Select Register bits SSR[0] (SPI_SSR[0]) to active the off-chip slave device.

4) 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).

5) If this SPI master attempts to transmit (write) one byte data to the off-chip slave device, write the byte data that will be transmitted into the SPI_TX0 register.

6) 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.

7) Enable the GO_BUSY bit (SPI_CNTRL [0] = 1) to start the data transfer with the SPI interface.

8) 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.

9) Read out the received one byte data from SPI_RX0[7:0].

10) 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.
- SPICLK 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.

6.17.7 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** **SPI2_BA = 0x4013_0000** **SPI3_BA = 0x4013_4000**				
SPI_CNTRL **x=0,1,2,3**	SPIx_BA+0x00	R/W	Control and Status Register	0x0500_3004
SPI_DIVIDER **x=0,1,2,3**	SPIx_BA+0x04	R/W	Clock Divider Register	0x0000_0000
SPI_SSR **x=0,1,2,3**	SPIx_BA+0x08	R/W	Slave Select Register	0x0000_0000
SPI_RX0 **x=0,1,2,3**	SPIx_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPI_RX1 **x=0,1,2,3**	SPIx_BA+0x14	R	Data Receive Register 1	0x0000_0000
SPI_TX0 **x=0,1,2,3**	SPIx_BA+0x20	W	Data Transmit Register 0	0x0000_0000
SPI_TX1 **x=0,1,2,3**	SPIx_BA+0x24	W	Data Transmit Register 1	0x0000_0000
SPI_VARCLK **x=0,1,2,3**	SPIx_BA+0x34	R/W	Variable Clock Pattern Register	0x007F_FF87
SPI_DMA **x=0,1,2,3**	SPIx_BA+0x38	R/W	SPI DMA Control Register	0x0000_0000
SPI_CNTRL2 **x=0,1,2,3**	SPIx_BA+0x3C	R/W	Control and Status Register 2	0x0000_1000
SPI_FIFO_CTL **x=0,1,2,3**	SPIx_BA+0x40	R/W	SPI FIFO Control Register	0x4400_0000
SPI_STATUS **x=0,1,2,3**	SPIx_BA+0x44	R/W	SPI Status Register	0x0500_0000

6.17.8 Register Description

SPI Control and Status Register (SPI_CNTRL)

Register	Offset	R/W	Description	Reset Value
SPI_CNTRL	SPIx_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	TWOB	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_STAUTS[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 FIOF 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]	TWOB	**2-bit Mode Enable Bit** 0 = 2-bit mode Disabled.

		1 = 2-bit mode Enabled. **Note:** When 2-bit mode is enabled, the serial transmitted 2-bit data are from SPI_TX1/0, and the received 2-bit data input are put in SPI_RX1/0.
[21]	**FIFO**	**FIFO Mode Enable Bit** 0 = FIFO mode Disabled. 1 = FIFO mode Enabled. **Note:** 1. Before enabling FIFO mode, the other related settings should be set in advance. 2. 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.
[20]	**Reserved**	Reserved.
[19]	**REORDER**	**Byte Reorder Function Enable Bit** 0 = Byte reorder function Disabled. 1 = Byte reorder function Enabled. A byte suspend interval will be inserted among each byte. The period of the byte suspend interval depends on the setting of SP_CYCLE. **Note:** 1. Byte reorder function is only available if TX_BIT_LEN is defined as 16, 24, and 32 bits. 2. In Slave mode with level-trigger configuration, the slave select pin must be kept at active state during the byte suspend interval. 3. The byte reorder function is not supported when the variable serial clock function or Dual I/O mode is enabled.
[18]	**SLAVE**	**Slave Mode Enable Bit** 0 = Master mode. 1 = Slave mode.
[17]	**IE**	**Unit Transfer Interrupt Enable Bit** 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 SPICLK clock cycle Example: SP_CYCLE = 0x0 … 0.5 SPICLK clock cycle. SP_CYCLE = 0x1 … 1.5 SPICLK clock cycle. …… SP_CYCLE = 0xE … 14.5 SPICLK clock cycle. SP_CYCLE = 0xF … 15.5 SPICLK clock cycle. 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) * SPICLK clock cycle.

[11]	**CLKP**	**Clock Polarity** 0 = SPICLK is idle low. 1 = SPICLK 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 TX0/1 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_RX0/1).
[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 SPICLK. 1 = Transmitted data output signal is changed on the falling edge of SPICLK.
[1]	**RX_NEG**	**Receive on Negative Edge** 0 = Received data input signal is latched on the rising edge of SPICLK. 1 = Received data input signal is latched on the falling edge of SPICLK.
[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. **Note:** 1. When FIFO mode is disabled, all configurations should be set before writing 1 to this GO_BUSY bit. 2. 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	SPIx_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 serial 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	SPIx_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**	**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, SPISSx0/1 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 (SPISSx0/1). 0 = The slave select signal SPISSx0/1 is active on low-level/falling-edge. 1 = The slave select signal SPISSx0/1 is active on high-level/rising-edge.
[1:0]	**SSR**	**Slave Select Control Bits (Master Only)** If AUTOSS bit is cleared, writing 1 to any bit of this field sets the proper SPISSx0/1 line to an active state and writing 0 sets the line back to inactive state. If the AUTOSS bit is set, writing 0 to any bit location of this field will keep the corresponding SPISSx0/1 line at inactive state; writing 1 to any bit location of this field will select appropriate SPISSx0/1 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 SPISSx0/1 is specified in SS_LVL. **Note:** SPISSx0 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	SPIx_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPI_RX1	SPIx_BA+0x14	R	Data Receive Register 1	0x0000_0000

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

Bits	Description	
[31:0]	**RX**	**Data Receive Register (Read Only)** 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.

SPI Data Transmit Register (SPI_TX)

Register	Offset	R/W	Description	Reset Value
SPI_TX0	SPIx_BA+0x20	W	Data Transmit Register 0	0x0000_0000
SPI_TX1	SPIx_BA+0x24	W	Data Transmit Register 1	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:** 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

SPI Variable Clock Pattern Register (SPI_VARCLK)

Register	Offset	R/W	Description	Reset Value
SPI_VARCLK	SPIx_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	SPIx_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	
[31:3]	**Reserved**	Reserved.
[2]	**PDMA_RST**	**PDMA Reset** 1 = Reset the PDMA control logic of the SPI controller. This bit will be cleared to 0 automatically. 0 = No effect.
[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	SPIx_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 Enable Bit** 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 Enable Bit** 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 SPICLK, SPI_MISO, and SPI_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	SPIx_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	SPIx_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.

6.18 I²S Controller (I²S)

6.18.1 Overview

The I²S controller consists of I²S protocol to interface with external audio CODEC. Two 8-word deep FIFO for read path and write path respectively and is capable of handling 8-, 16-, 24- and 32-bit word sizes. PDMA controller handles the data movement between FIFO and memory.

6.18.2 Features

- Operated as either Master or Slave
- Capable of handling 8-, 16-, 24- and 32-bit word sizes
- Supports Mono and stereo audio data
- Supports I²S and MSB justified data format
- Provides two 8-word FIFO data buffers, one for transmitting and the other for receiving
- Generates interrupt requests when buffer levels cross a programmable boundary
- Two PDMA requests, one for transmitting and the other for receiving

6.18.3 Block Diagram

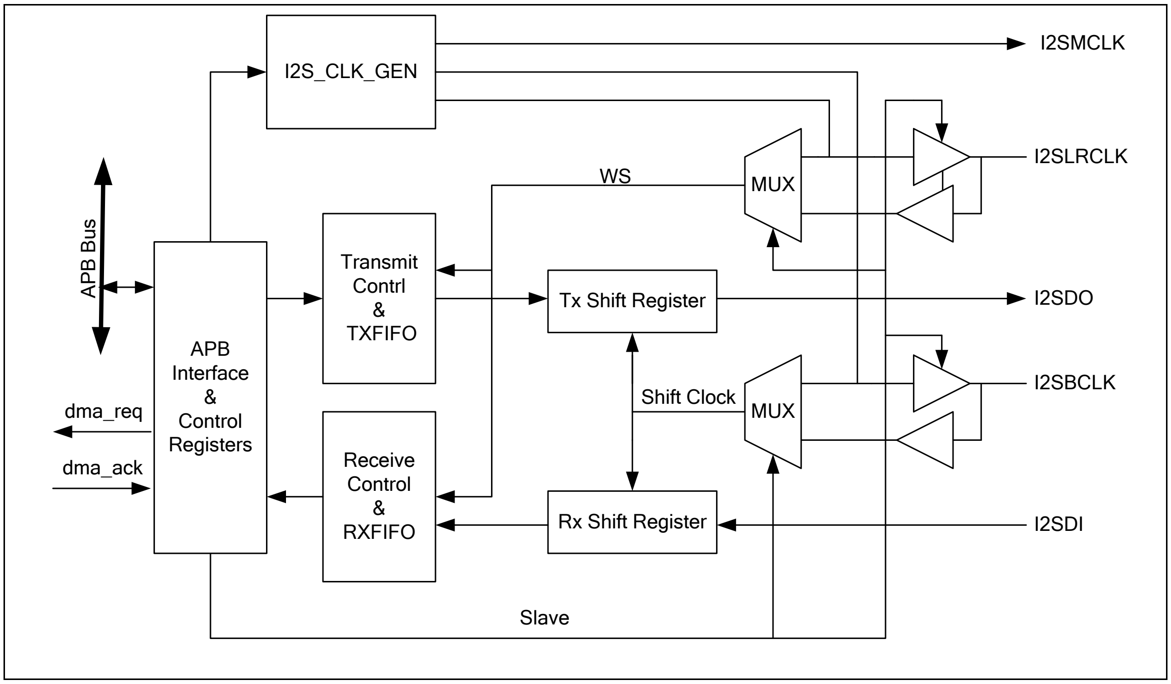


Figure 6-125 I^2S Controller Block Diagram

6.18.4 Functional Description

6.18.4.1 I²S Clock

The I²S controller has four clock sources selected by I2S_S(CLKSEL2[1:0]). The I²S clock rate must be slower than or equal to system clock rate.

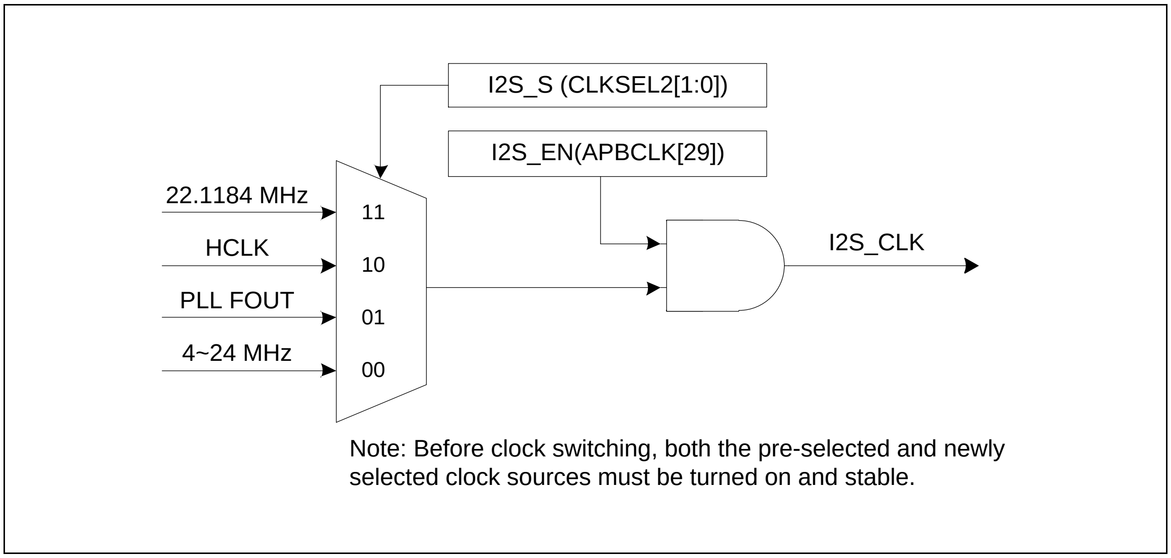


Figure 6-126 I²S Clock Control Diagram

6.18.4.2 I²S Operation

The I²S controller supports MSB justified and I²S data format. The I2SLRCLK signal indicates which audio channel is in transferring. The bit count of an audio channel is determined by WORDWIDTH setting. The transfer sequence is always first from the most significance bit, MSB. Data are read on rising clock edge and are driven on falling clock edge.

In I²S data format, the MSB is sent and latched on the second clock of an audio channel.

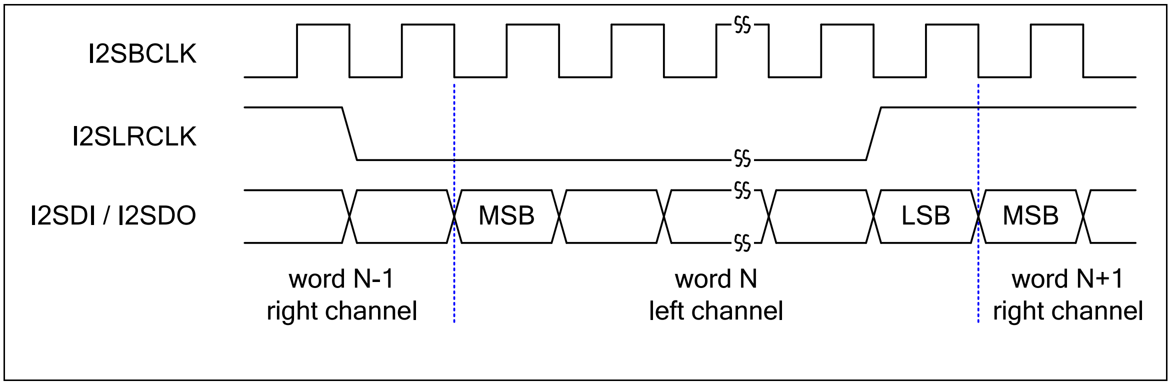


Figure 6-127 I²S Data Format Timing Diagram

In MSB justified data format, the MSB is sent and latched on the first clock of an audio channel.

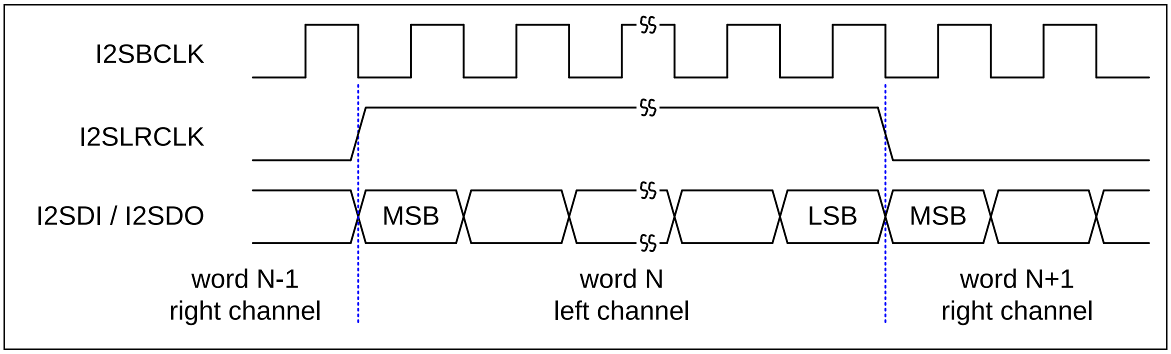


Figure 6-128 MSB Justified Data Format Timing Diagram

6.18.4.3 I²S Interrupt sources

The I²S controller supports left channel zero-cross interrupt, right channel zero-cross interrupt, transmit FIFO threshold level interrupt, transmit FIFO overflow interrupt and transmit FIFO underflow interrupt in transmit operation. In receive operation, it supports receive FIFO threshold level interrupt, receive FIFO overflow interrupt and receive FIFO underflow interrupt. When I²S interrupt occurs, user can check I2STXINT and I2SRXINT flags to recognize the interrupt sources.

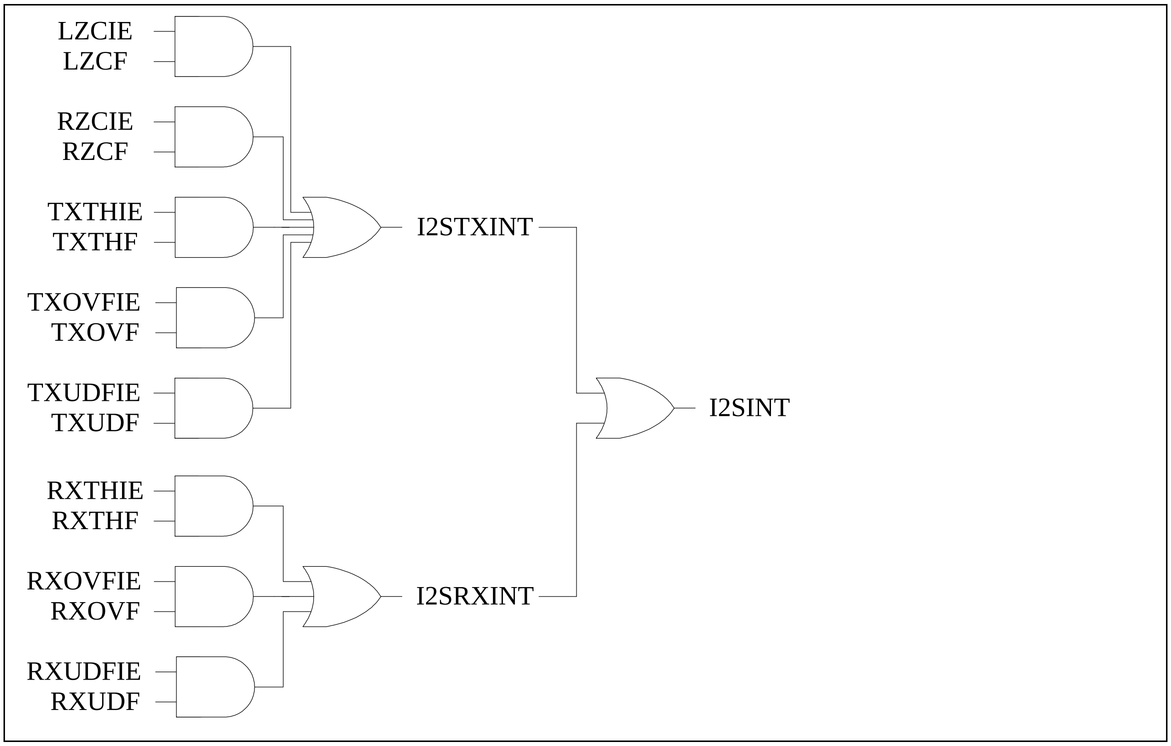


Figure 6-129 I²S Interrupts

6.18.4.4 *FIFO operation*

The word width of an audio channel can be 8, 16, 24 or 32 bits. The memory arrangements for various settings are shown below.

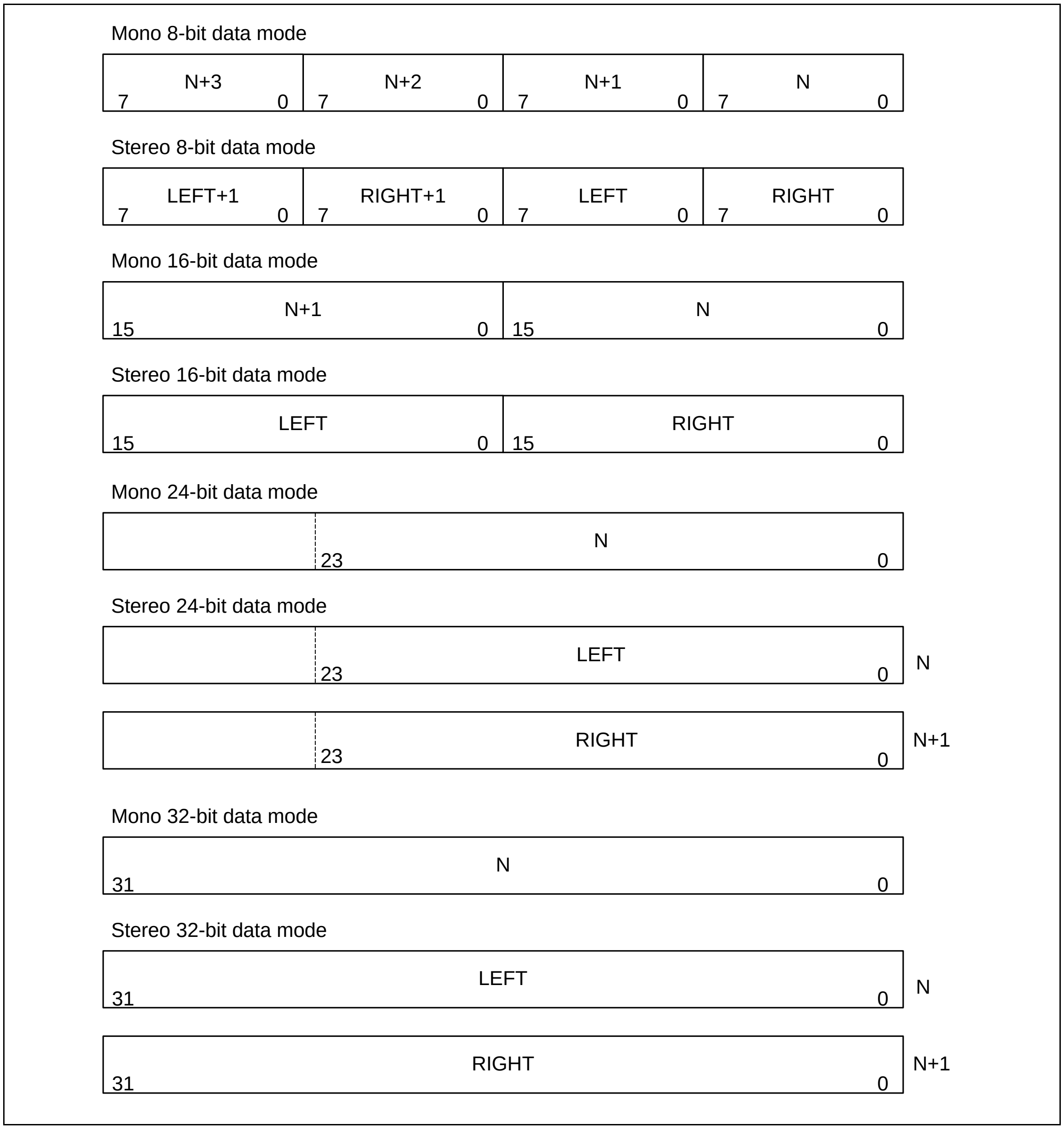


Figure 6-130 FIFO Contents for Various I²S Modes

6.18.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
I^2S Base Address: **I2S_BA = 0x401A_0000**				
I2SCON	I2S_BA+0x00	R/W	I^2S Control Register	0x0000_0000
I2SCLKDIV	I2S_BA+0x04	R/W	I^2S Clock Divider Control Register	0x0000_0000
I2SIE	I2S_BA+0x08	R/W	I^2S Interrupt Enable Register	0x0000_0000
I2SSTATUS	I2S_BA+0x0C	R/W	I^2S Status Register	0x0014_1000
I2STXFIFO	I2S_BA+0x10	W	I^2S Transmit FIFO Register	0x0000_0000
I2SRXFIFO	I2S_BA+0x14	R	I^2S Receive FIFO Register	0x0000_0000

6.18.6 Register Description

I²S Control Register (I2SCON)

Register	Offset	R/W	Description	Reset Value
I2SCON	I2S_BA+0x00	R/W	I²S Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXLCH	Reserved	RXDMA	TXDMA	CLR_RXFIFO	CLR_TXFIFO	LCHZCEN	RCHZCEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MCLKEN	RXTH			TXTH			SLAVE
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FORMAT	MONO	WORDWIDTH		MUTE	RXEN	TXEN	I2SEN

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**RXLCH**	**Receive Left Channel Enable Bit** When monaural format is selected (MONO = 1), I²S controller will receive right channel data if RXLCH is set to 0, and receive left channel data if RXLCH is set to 1. 0 = Receive right channel data in Mono mode. 1 = Receive left channel data in Mono mode.
[22]	**Reserved**	Reserved.
[21]	**RXDMA**	**Enable Receive DMA** When RX DMA is enabled, I²S requests DMA to transfer data from receive FIFO to SRAM if FIFO is not empty. 0 = RX DMA Disabled. 1 = RX DMA Enabled.
[20]	**TXDMA**	**Enable Transmit DMA** When TX DMA is enabled, I²S request DMA to transfer data from SRAM to transmit FIFO if FIFO is not full. 0 = TX DMA Disabled. 1 = TX DMA Enabled.
[19]	**CLR_RXFIFO**	**Clear Receive FIFO** Write 1 to clear receive FIFO, internal pointer is reset to FIFO start point, and RXFIFO_LEVEL[3:0] returns 0 and receive FIFO becomes empty. This bit is cleared by hardware automatically. Returns 0 on read.
[18]	**CLR_TXFIFO**	**Clear Transmit FIFO** Write 1 to clear transmit FIFO, internal pointer is reset to FIFO start point, and TXFIFO_LEVEL[3:0] returns to 0 and transmit FIFO becomes empty but data in transmit FIFO is not changed. This bit is cleared by hardware automatically. Returns 0 on read.

[17]	**LCHZCEN**	**Left Channel Zero Cross Detect Enable Bit** If this bit is set to 1, when left channel data sign bit changes or next shift data bits are all 0 then LZCF flag in I2SSTATUS register is set to 1. This function is only available in transmit operation. 0 = Left channel zero cross detect Disabled. 1 = Left channel zero cross detect Enabled.
[16]	**RCHZCEN**	**Right Channel Zero Cross Detect Enable Bit** If this bit is set to 1, when right channel data sign bit change or next shift data bits are all 0 then RZCF flag in I2SSTATUS register is set to 1. This function is only available in transmit operation. 0 = Right channel zero cross detect Disabled. 1 = Right channel zero cross detect Enabled.
[15]	**MCLKEN**	**Master Clock Enable Bit** If MCLKEN is set to 1, I^2S controller will generate master clock on I2SMCLK pin for external audio devices. 0 = Master clock Disabled. 1 = Master clock Enabled.
[14:12]	**RXTH**	**Receive FIFO Threshold Level** When received data word(s) in buffer is equal to or higher than threshold level then RXTHF flag is set. 000 = 1 word data in receive FIFO. 001 = 2 word data in receive FIFO. 010 = 3 word data in receive FIFO. 011 = 4 word data in receive FIFO. 100 = 5 word data in receive FIFO. 101 = 6 word data in receive FIFO. 110 = 7 word data in receive FIFO. 111 = 8 word data in receive FIFO.
[11:9]	**TXTH**	**Transmit FIFO Threshold Level** If remaining data word (32 bits) in transmit FIFO is the same or less than threshold level then TXTHF flag is set. 000 = 0 word data in transmit FIFO. 001 = 1 word data in transmit FIFO. 010 = 2 words data in transmit FIFO. 011 = 3 words data in transmit FIFO. 100 = 4 words data in transmit FIFO. 101 = 5 words data in transmit FIFO. 110 = 6 words data in transmit FIFO. 111 = 7 words data in transmit FIFO.
[8]	**SLAVE**	**Slave Mode** I^2S can operate as master or slave. For Master mode, I2S_BCLK and I2S_LRCLK pins are output mode and send bit clock from NuMicro® NUC100 series to Audio CODEC chip. In Slave mode, I2S_BCLK and I2S_LRCLK pins are input mode and I2S_BCLK and I2S_LRCLK signals are received from outer Audio CODEC chip. 0 = Master mode. 1 = Slave mode.
[7]	**FORMAT**	**Data Format** 0 = I^2S data format. 1 = MSB justified data format.

[6]	**MONO**	**Monaural Data** 0 = Data is stereo format. 1 = Data is monaural format.
[5:4]	**WORDWIDTH**	**Word Width** 00 = data is 8-bit. 01 = data is 16-bit. 10 = data is 24-bit. 11 = data is 32-bit.
[3]	**MUTE**	**Transmit Mute Enable Bit** 0 = Transmit data is shifted from buffer. 1= Transmit channel zero.
[2]	**RXEN**	**Receive Enable Bit** 0 = Data receiving Disabled. 1 = Data receiving Enabled.
[1]	**TXEN**	**Transmit Enable Bit** 0 = Data transmit Disabled. 1 = Data transmit Enabled.
[0]	**I2SEN**	**I^2S Controller Enable Bit** 0 = I^2S Controller Disabled. 1 = I^2S Controller Enabled.

I^2S Clock Divider (I2SCLKDIV)

Register	Offset	R/W	Description	Reset Value
I2SCLKDIV	I2S_BA+0x04	R/W	I^2S Clock 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**
BCLK_DIV							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**MCLK_DIV**		

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**BCLK_DIV**	**Bit Clock Divider** The I^2S controller will generate bit clock in Master mode. The bit clock rate, F_BCLK, is determined by the following expression. F_BCLK = F_I2SCLK /(2x(BCLK_DIV + 1)) , where F_I2SCLK is the frequency of I^2S clock.
[7:3]	**Reserved**	Reserved.
[2:0]	**MCLK_DIV**	**Master Clock Divider** If MCLKEN is set to 1, I^2S controller will generate master clock for external audio devices. The master clock rate, F_MCLK, is determined by the following expressions. If MCLK_DIV >= 1, F_MCLK = F_I2SCLK/(2x(MCLK_DIV)). If MCLK_DIV = 0, F_MCLK = F_I2SCLK. F_I2SCLK is the frequency of I^2S clock. In general, the master clock rate is 256 times sampling clock rate.

I^2S Interrupt Enable Register (I2SIE)

Register	Offset	R/W	Description	Reset Value
I2SIE	I2S_BA+0x08	R/W	I^2S 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			LZCIE	RZCIE	TXTHIE	TXOVFIE	TXUDFIE
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					RXTHIE	RXOVFIE	RXUDFIE

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**LZCIE**	**Left Channel Zero-cross Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and left channel zero-cross. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[11]	**RZCIE**	**Right Channel Zero-cross Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and right channel zero-cross. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[10]	**TXTHIE**	**Transmit FIFO Threshold Level Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and data words in transmit FIFO is less than TXTH[2:0]. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[9]	**TXOVFIE**	**Transmit FIFO Overflow Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and the transmit FIFO overflow flag is set to 1 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[8]	**TXUDFIE**	**Transmit FIFO Underflow Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and the transmit FIFO underflow flag is set to 1. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[7:3]	**Reserved**	Reserved.

[2]	**RXTHIE**	**Receive FIFO Threshold Level Interrupt Enable Bit** When data word in receive FIFO is equal to or higher then RXTH[2:0] and the RXTHF bit is set to 1. If RXTHIE bit is enabled, interrupt occurs. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[1]	**RXOVFIE**	**Receive FIFO Overflow Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[0]	**RXUDFIE**	**Receive FIFO Underflow Interrupt Enable Bit** If software read receive FIFO when it is empty then RXUDF flag in I2SSTATUS register is set to 1. 0 = Interrupt Disabled. 1 = Interrupt Enabled.

I²S Status Register (I2SSTATUS)

Register	Offset	R/W	Description	Reset Value
I2SSTATUS	I2S_BA+0x0C	R/W	I²S Status Register	0x0014_1000

31	30	29	28	27	26	25	24
TX_LEVEL				RX_LEVEL			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
LZCF	RZCF	TXBUSY	TXEMPTY	TXFULL	TXTHF	TXOVF	TXUDF
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			RXEMPTY	RXFULL	RXTHF	RXOVF	RXUDF
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				RIGHT	I2STXINT	I2SRXINT	I2SINT

Bits	Description	
[31:28]	**TX_LEVEL**	**Transmit FIFO Level** These bits indicate word number in transmit FIFO 0000 = No data. 0001 = 1 word in transmit FIFO. …. 1000 = 8 words in transmit FIFO.
[27:24]	**RX_LEVEL**	**Receive FIFO Level** These bits indicate word number in receive FIFO 0000 = No data. 0001 = 1 word in receive FIFO. …. 1000 = 8 words in receive FIFO.
[23]	**LZCF**	**Left Channel Zero-cross Flag** It indicates left channel next sample data sign bit is changed or all data bits are 0. 0 = No zero-cross. 1 = Left channel zero-cross is detected. **Note:** This bit can be cleared by writing '1' to it.
[22]	**RZCF**	**Right Channel Zero-cross Flag** It indicates right channel next sample data sign bit is changed or all data bits are 0. 0 = No zero-cross. 1 = Right channel zero-cross is detected. **Note:** This bit can be cleared by writing '1' to it.
[21]	**TXBUSY**	**Transmit Busy (Read Only)** This bit is cleared to 0 when all data in transmit FIFO and shift buffer is shifted out. And set to 1 when 1st data is load to shift buffer. 0 = Transmit shift buffer is empty. 1 = Transmit shift buffer is busy.

[20]	**TXEMPTY**	**Transmit FIFO Empty (Read Only)** This bit reflects data word number in transmit FIFO is 0 0 = Not empty. 1 = Empty.
[19]	**TXFULL**	**Transmit FIFO Full (Read Only)** This bit reflects data word number in transmit FIFO is 8 0 = Not full. 1 = Full.
[18]	**TXTHF**	**Transmit FIFO Threshold Flag (Read Only)** When data word(s) in transmit FIFO is equal to or lower than threshold value set in TXTH[2:0] the TXTHF bit becomes to 1. It keeps at 1 till TXFIFO_LEVEL[3:0] is higher than TXTH[1:0] after software writes TXFIFO register. 0 = Data word(s) in FIFO is higher than threshold level. 1 = Data word(s) in FIFO is equal to or lower than threshold level.
[17]	**TXOVF**	**Transmit FIFO Overflow Flag** This bit will be set to 1 if writes data to transmit FIFO when transmit FIFO is full. 0 = No overflow. 1 = Overflow. **Note:** This bit can be cleared by writing ‘1’ to it.
[16]	**TXUDF**	**Transmit FIFO Underflow Flag** When transmit FIFO is empty and shift logic hardware read data from transmit FIFO causes this set to 1. 0 = No underflow. 1 = Underflow. **Note:** This bit can be cleared by writing ‘1’ to it.
[15:13]	**Reserved**	Reserved.
[12]	**RXEMPTY**	**Receive FIFO Empty (Read Only)** This bit reflects data words number in receive FIFO is 0 0 = Not empty. 1 = Empty.
[11]	**RXFULL**	**Receive FIFO Full (Read Only)** This bit reflects data words number in receive FIFO is 8 0 = Not full. 1 = Full.
[10]	**RXTHF**	**Receive FIFO Threshold Flag (Read Only)** When data word(s) in receive FIFO is equal to or higher than threshold value set in RXTH[2:0] the RXTHF bit becomes to 1. It keeps at 1 till RXFIFO_LEVEL[3:0] is less than RXTH[1:0] after software read RXFIFO register. 0 = Data word(s) in FIFO is lower than threshold level. 1 = Data word(s) in FIFO is equal to or higher than threshold level.
[9]	**RXOVF**	**Receive FIFO Overflow Flag** When receive FIFO is full and receive hardware attempt to write data into receive FIFO then this bit is set to 1, data in 1st buffer is overwrote. 0 = No overflow. 1 = Overflow. **Note:** This bit can be cleared by writing ‘1’ to it.

[8]	**RXUDF**	**Receive FIFO Underflow Flag** Read receive FIFO when it is empty, this bit set to 1 indicate underflow occurs. 0 = No underflow. 1 = Underflow. **Note:** This bit can be cleared by writing '1' to it.
[7:4]	**Reserved**	Reserved.
[3]	**RIGHT**	**Right Channel (Read Only)** This bit indicates current transmit data is belong to right channel 0 = Left channel. 1 = Right channel.
[2]	**I2STXINT**	**I^{2}S Transmit Interrupt (Read Only)** 0 = No transmit interrupt. 1 = Transmit interrupt.
[1]	**I2SRXINT**	**I^{2}S Receive Interrupt (Read Only)** 0 = No receive interrupt. 1 = Receive interrupt.
[0]	**I2SINT**	**I^{2}S Interrupt Flag (Read Only)** 0 = No I^{2}S interrupt. 1 = I^{2}S interrupt. It is wire-OR of I2STXINT and I2SRXINT bits.

I²S Transmit FIFO (I2STXFIFO)

Register	Offset	R/W	Description	Reset Value
I2STXFIFO	I2S_BA+0x10	W	I²S Transmit FIFO Register	0x0000_0000

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

Bits	Description	
[31:0]	**TXFIFO**	**Transmit FIFO Register** I²S contains 8 words (8x32 bits) data buffer for data transmit. Write data to this register to prepare data for transmit. The remaining word number is indicated by TX_LEVEL[3:0] in I2SSTATUS register

I²S Receive FIFO (I2SRXFIFO)

Register	Offset	R/W	Description	Reset Value
I2SRXFIFO	I2S_BA+0x14	R	I²S Receive FIFO Register	0x0000_0000

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

Bits	Description	
[31:0]	**RXFIFO**	**Receive FIFO Register** I²S contains 8 words (8x32 bits) data buffer for data receive. Read this register to get data in FIFO. The remaining data word number is indicated by RX_LEVEL[3:0] in I2SSTATUS register.

6.19 USB Device Controller (USB)

6.19.1 Overview

There is one set of USB 2.0 full-speed device controller and transceiver in this device. It is compliant with USB 2.0 full-speed device specification and supports control/bulk/interrupt/ isochronous transfer types.

In this device controller, there are two main interfaces: the APB bus and USB bus which comes from the USB PHY transceiver. For the APB bus, the CPU can program control registers through it. There are 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 6 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, like IN ACK, OUT ACK etc, and BUS events, like suspend and resume, etc. 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-disable function is also supported for this USB controller. It is used to simulate the disconnection of this device from the host. If user enables DRVSE0 bit (USB_DRVSE0), the USB controller will force the output of USB_DP and USB_DM to level low and its function is disabled. After disable the DRVSE0 bit, host will enumerate the USB device again.

Please refer to *Universal Serial Bus Specification Revision 1.1*

6.19.2 Features

This Universal Serial Bus (USB) performs a serial interface with a single connector type for attaching all USB peripherals to the host system. Following is the feature list of this USB.

- 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 6 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 512 bytes buffer size
- Provides remote wake-up capability

6.19.3 Block Diagram

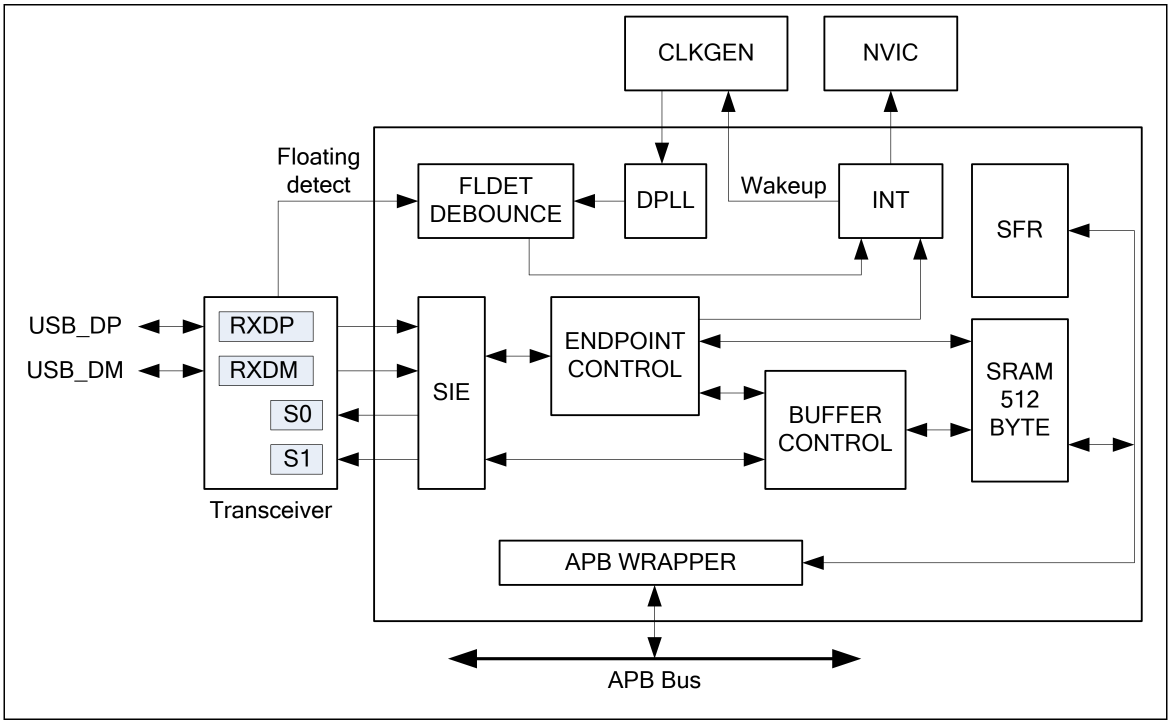


Figure 6-131 USB Block Diagram

6.19.4 Functional Description

6.19.4.1 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
- SOP, EOP, RESET, RESUME signal detection/generation
- Clock/Data separation
- NRZI Data encoding/decoding and bit-stuffing
- CRC generation and checking (for Token and Data)
- Packet ID (PID) generation and checking/ decoding
- Serial-Parallel/ Parallel-Serial conversion

6.19.4.2 Endpoint Control

There are 6 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.

6.19.4.3 Digital Phase Lock Loop

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.

6.19.4.4 Floating De-bounce

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-bounce for USB floating detect interrupt to avoid bounce problems on USB plug-in or unplug. Floating detect interrupt appears about 10 ms later than USB plug-in or plug-out. User can acknowledge USB plug-in/plug-out by reading the register “USB_FLDET”. The flag in “FLDET” represents the current state on the bus without de-bounce. If the FLDET is 1, it means the controller has the USB plugged-in. If user polls the flag to check USB state, software de-bounce must be added if needed.

6.19.4.5 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.

The 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_DP and USB_DM can wake-up this chip (provided that 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 present for more than 20ms. The following figure is the control flow of wake-up interrupt.

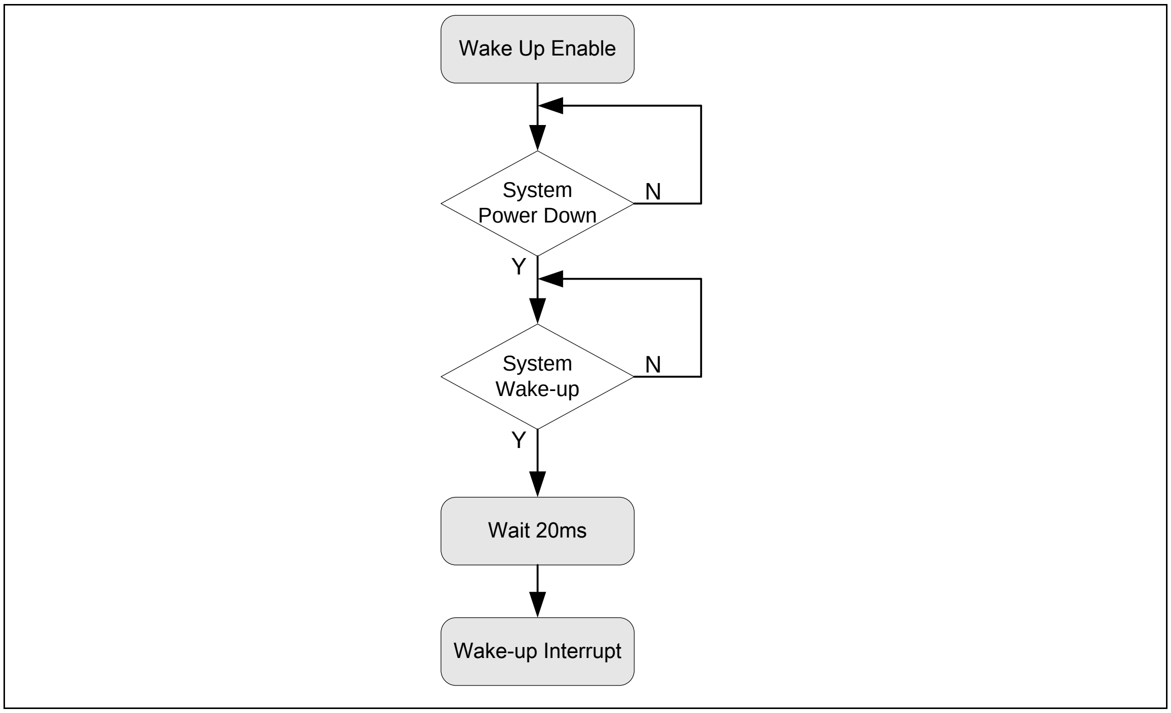


Figure 6-132 Wake-up Interrupt Operation Flow

USB interrupt is used to notify users of any USB event on the bus, and user can read EPSTS (USB_EPSTS[25:8]) and EPEVT5~0 (USB_INTSTS[21:16]) to know what kind of request is to which endpoint and 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.

6.19.4.6 Power Saving

The USB turns off PHY transceiver automatically to save power while this chip enters Power-down mode. Furthermore, a user can write 0 into USB_ATTR[4] to turn off PHY under special circumstances like suspend to save power.

6.19.4.7 Buffer Control

There is 512 bytes SRAM in the controller and the 6 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 MXPLD register.

Figure 6-133 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 USB_BA+0x108h and end in USB_BA+0x148h. (**Note:** The USB SRAM base is USB_BA+0x100h).

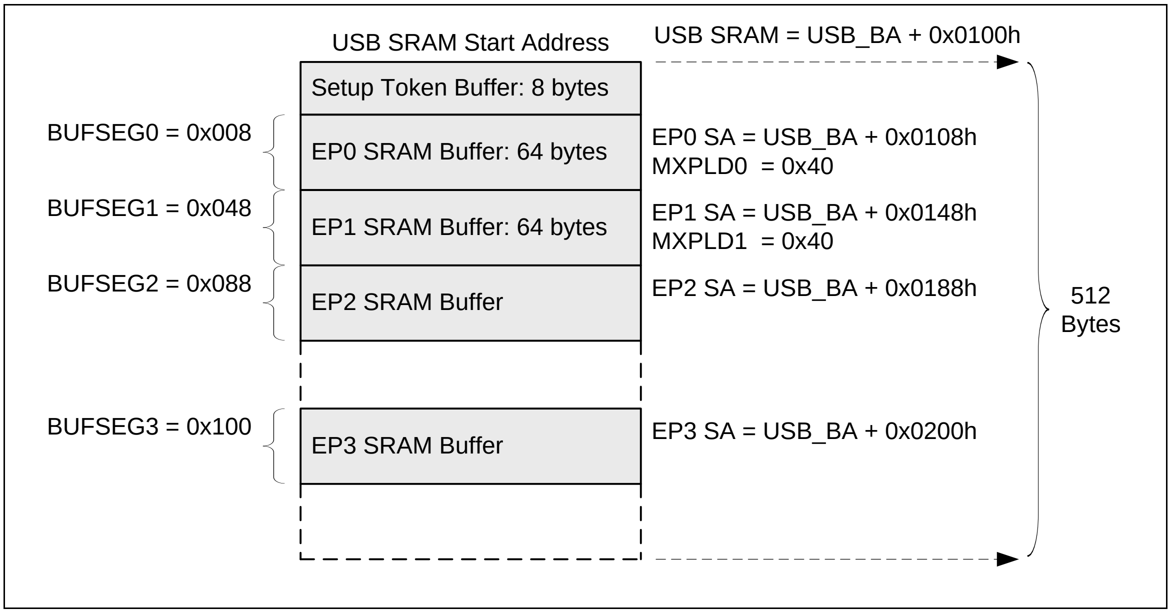


Figure 6-133 Endpoint SRAM Structure

6.19.4.8 Handling Transactions with USB Device Peripheral

User can use interrupt or polling 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 polling 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 MAXPLD register. Once this register is written, the internal signal "In_Rdy" will be asserted and the buffering data will be transmitted immediately after receiving associated IN token from Host. Note that after transferring the specified data, the signal "In_Rdy" will de-assert automatically by hardware.

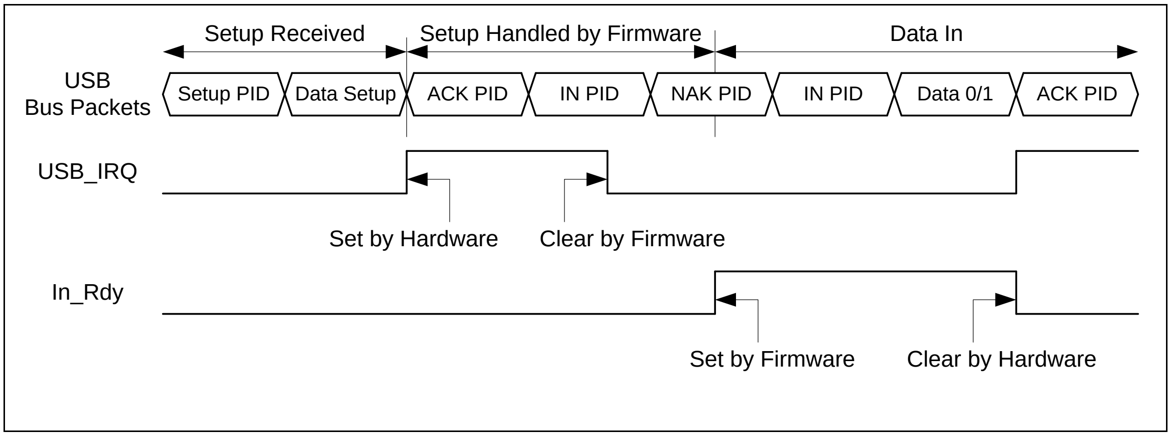


Figure 6-134 Setup Transaction followed by Data in Transaction

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 related MAXPLD register and de-assert the 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 related register “MAXPLD” needs to be written by firmware to assert the signal “Out_Rdy” again to accept the next transaction.

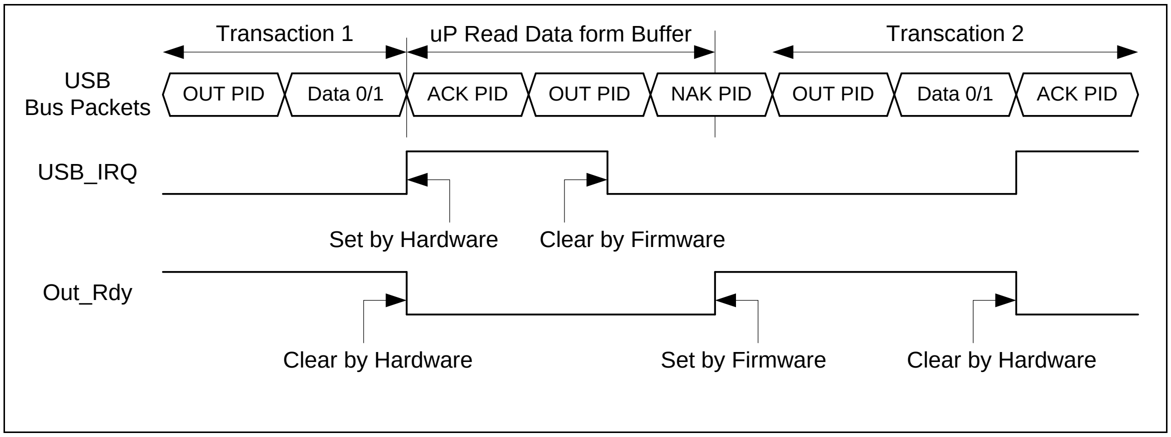


Figure 6-135 Data Out Transfer

6.19.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
USB Base Address: **USB_BA = 0x4006_0000**				
USB_INTEN	USB_BA+0x000	R/W	USB Interrupt Enable Register	0x0000_0000
USB_INTSTS	USB_BA+0x004	R/W	USB Interrupt Event Status Register	0x0000_0000
USB_FADDR	USB_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000
USB_EPSTS	USB_BA+0x00C	R	USB Endpoint Status Register	0x0000_0000
USB_ATTR	USB_BA+0x010	R/W	USB Bus Status and Attribution Register	0x0000_0040
USB_FLDET	USB_BA+0x014	R	USB Floating Detected Register	0x0000_0000
USB_STBUFSEG	USB_BA+0x018	R/W	Setup Token Buffer Segmentation Register	0x0000_0000
USB_BUFSEG0	USB_BA+0x020	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USB_MXPLD0	USB_BA+0x024	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USB_CFG0	USB_BA+0x028	R/W	Endpoint 0 Configuration Register	0x0000_0000
USB_CFGP0	USB_BA+0x02C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG1	USB_BA+0x030	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USB_MXPLD1	USB_BA+0x034	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USB_CFG1	USB_BA+0x038	R/W	Endpoint 1 Configuration Register	0x0000_0000
USB_CFGP1	USB_BA+0x03C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG2	USB_BA+0x040	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USB_MXPLD2	USB_BA+0x044	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USB_CFG2	USB_BA+0x048	R/W	Endpoint 2 Configuration Register	0x0000_0000
USB_CFGP2	USB_BA+0x04C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG3	USB_BA+0x050	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USB_MXPLD3	USB_BA+0x054	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USB_CFG3	USB_BA+0x058	R/W	Endpoint 3 Configuration Register	0x0000_0000
USB_CFGP3	USB_BA+0x05C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG4	USB_BA+0x060	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USB_MXPLD4	USB_BA+0x064	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USB_CFG4	USB_BA+0x068	R/W	Endpoint 4 Configuration Register	0x0000_0000

USB_CFGP4	USB_BA+0x06C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG5	USB_BA+0x070	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USB_MXPLD5	USB_BA+0x074	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USB_CFG5	USB_BA+0x078	R/W	Endpoint 5 Configuration Register	0x0000_0000
USB_CFGP5	USB_BA+0x07C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_DRVSE0	USB_BA+0x090	R/W	USB Drive SE0 Control Register	0x0000_0001

Memory Type	**Address**	**Size**	**Description**
USB_BA = 0x4006_0000			
SRAM	USB_BA+0x100 ~ USB_BA+0x2FF	512 Bytes	The SRAM is used for the entire endpoints buffer. Refer to section 5.4.4.7 for the endpoint SRAM structure and its description.

6.19.6 Register Description

USB Interrupt Enable Register (USB_INTEN)

Register	Offset	R/W	Description	Reset Value
USB_INTEN	USB_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 = NAK status is not updated into the endpoint status register when it was set to 0. It also disable the interrupt event when device responds to NAK after receiving IN token. 1 = NAK status is updated into the endpoint status register, USB_EPSTS, when it is set to 1 and there is NAK response in IN token. It also enables the interrupt event 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 Detected Interrupt Enable Bit** 0 = Floating detect Interrupt Disabled. 1 = Floating detect 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)

The USB Interrupt Event Status register is cleared by writing ‘1’ to the corresponding bit.

Register	Offset	R/W	Description	Reset Value
USB_INTSTS	USB_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**
Reserved		EPEVT5	EPEVT4	EPEVT3	EPEVT2	EPEVT1	EPEVT0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				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].
[30:22]	**Reserved**	Reserved.
[21]	**EPEVT5**	**Endpoint 5’s USB Event Status** 0 = No event occurred in 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 in 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 in 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 in 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 in 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 in 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].
[15:4]	**Reserved**	Reserved.
[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 Detected 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 EPSTS0~5[2:0] to know which kind of USB event was occurred, cleared by write 1 to USB_INTSTS[1] or EPSTS0~5 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	USB_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	USB_BA+0x00C	R	USB Endpoint Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						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	
[31:26]	**Reserved**	Reserved.
[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	USB_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_DP Enable Bit** 0 = Pull-up resistor in USB_DP bus Disabled. 1 = Pull-up resistor in USB_DP bus Active.
[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_DP low, USB_DM: 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 (Read Only)** 0 = No time-out. 1 = No Bus response more than 18 bits time.
[2]	**RESUME**	**Resume Status (Read Only)** 0 = No bus resume.

		1 = Resume from suspend.
[1]	**SUSPEND**	**Suspend Status (Read Only)** 0 = Bus no suspend. 1 = Bus idle more than 3ms, either cable is plugged off or host is sleeping.
[0]	**USBRST**	**USB Reset Status (Read Only)** 0 = Bus no reset. 1 = Bus reset when SE0 (single-ended 0) more than 2.5us.

Floating detection Register (USB_FLDET)

Register	Offset	R/W	Description	Reset Value
USB_FLDET	USB_BA+0x014	R	USB Floating Detected Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**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	USB_BA+0x018	R/W	Setup Token Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:3]	**STBUFSEG**	**Buffer Segmentation** It is used to indicate the offset address for the setup token with the usb sram starting address the effective starting address is: USB_SRAM address + {STBUFSEG[8:3], 3'b000} Where the USB_SRAM address = USB_BA+0x100h. **Note:** It is used for Setup token only.
[2:0]	**Reserved**	Reserved.

Buffer Segmentation Register (USB_BUFSEGx)

Register	Offset	R/W	Description	Reset Value
USB_BUFSEG0	USB_BA+0x020	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG1	USB_BA+0x030	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG2	USB_BA+0x040	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG3	USB_BA+0x050	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG4	USB_BA+0x060	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG5	USB_BA+0x070	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**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 = USB_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	USB_BA+0x024	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USB_MXPLD1	USB_BA+0x034	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USB_MXPLD2	USB_BA+0x044	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USB_MXPLD3	USB_BA+0x054	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USB_MXPLD4	USB_BA+0x064	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USB_MXPLD5	USB_BA+0x074	R/W	Endpoint 5 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	USB_BA+0x028	R/W	Endpoint 0 Configuration Register	0x0000_0000
USB_CFG1	USB_BA+0x038	R/W	Endpoint 1 Configuration Register	0x0000_0000
USB_CFG2	USB_BA+0x048	R/W	Endpoint 2 Configuration Register	0x0000_0000
USB_CFG3	USB_BA+0x058	R/W	Endpoint 3 Configuration Register	0x0000_0000
USB_CFG4	USB_BA+0x068	R/W	Endpoint 4 Configuration Register	0x0000_0000
USB_CFG5	USB_BA+0x078	R/W	Endpoint 5 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	USB_BA+0x02C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP1	USB_BA+0x03C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP2	USB_BA+0x04C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP3	USB_BA+0x05C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP4	USB_BA+0x06C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP5	USB_BA+0x07C	R/W	Endpoint 5 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 MXPLD register is set by user, it means that the endpoint is ready to transmit or receive data. If the user wants to turn off this transaction before the transaction start, users can set this bit to 1 to turn it off and it is auto clear to 0. For IN token, write '1' to clear the IN token had ready to transmit the data to USB. For OUT token, write '1' to clear the OUT token had ready to receive the data from USB. This bit is 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	USB_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_DP and USB_DM) are being pulled low. 0 = None. 1 = Force USB PHY transceiver to drive SE0.

6.20 Analog-to-Digital Converter (ADC)

6.20.1 Overview

The NuMicro® NUC100 series contains one 12-bit successive approximation analog-to-digital converters (SAR A/D converter) with 8 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.

6.20.2 Features

- Analog input voltage range: 0~V_{REF}
- 12-bit resolution and 10-bit accuracy is guaranteed
- Up to 8 single-end analog input channels or 4 differential analog input channels
- Up to 760 kSPS conversion rate as ADC clock frequency is 16 MHz (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 through software
  - PWM Center-aligned trigger
  - External pin STADC
- Conversion results are held in data registers for each channel with valid and overrun indicators
- 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

6.20.3 Block Diagram

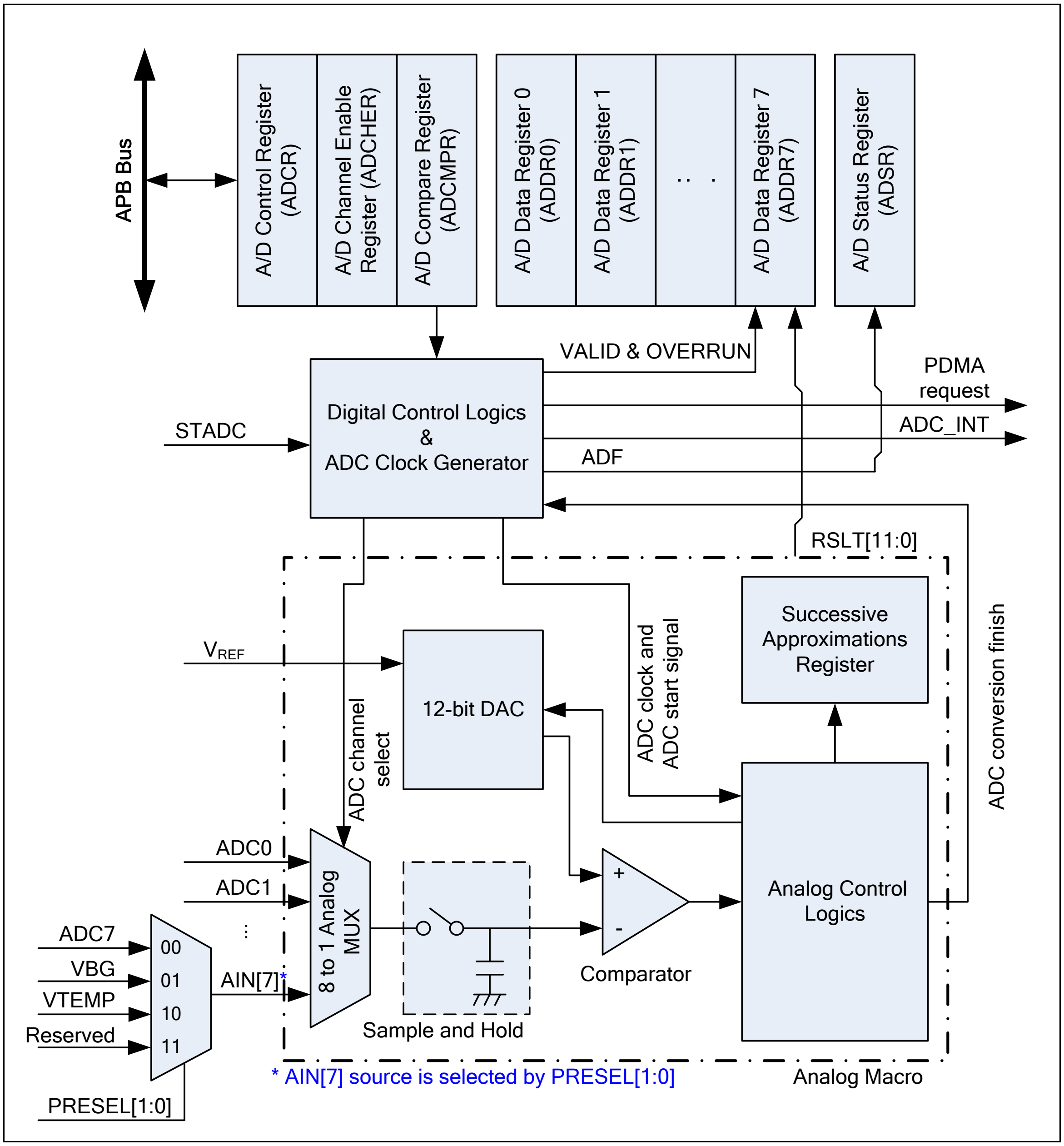


Figure 6-136 ADC Controller Block Diagram

6.20.4 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 to 0 in ADCR register.

6.20.4.1 ADC Clock Generator

The maximum sampling rate is up to 760 kSPS. 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+1);

where the 8-bit ADC_N is located in register CLKDIV[23:16].

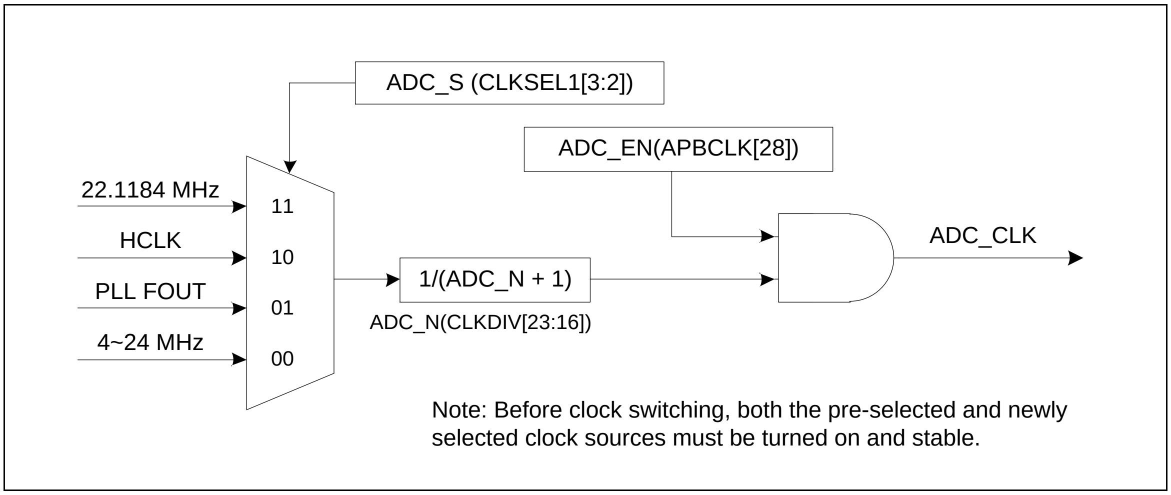


Figure 6-137 ADC Clock Control

6.20.4.2 Single Mode

In single mode, A/D conversion is performed only once on the specified single channel. The operations are as follows:

1. A/D conversion will be started when the ADST bit of ADCR 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 of ADSR register will be set to 1. If the ADIE bit of ADCR register is set to 1, the ADC interrupt will be asserted.
4. The ADST bit remains 1 during A/D conversion. When A/D conversion ends, the ADST bit is automatically cleared to 0 and the A/D converter enters idle state.

Note: 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.

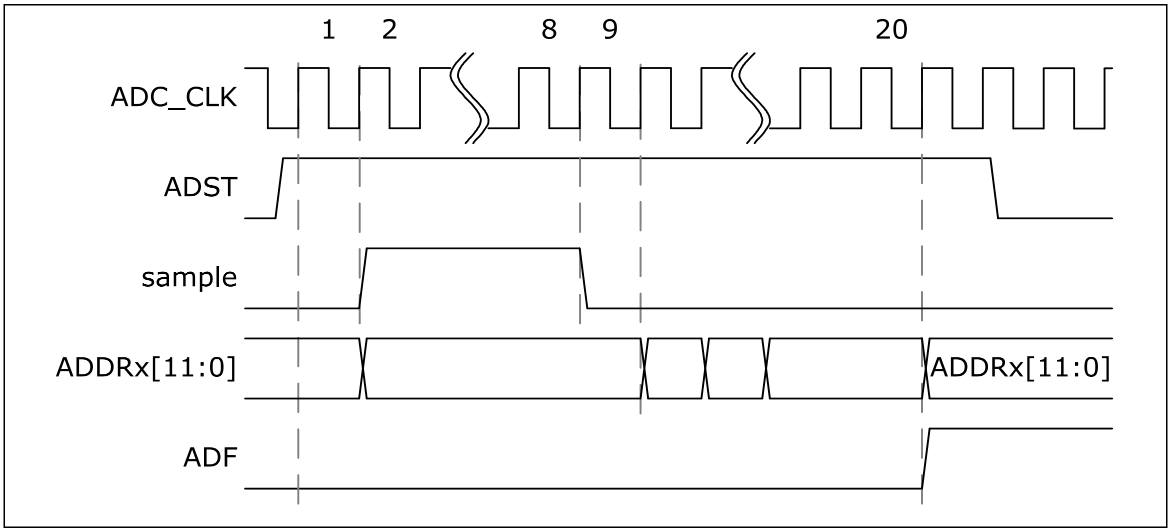


Figure 6-138 Single Mode Conversion Timing Diagram

6.20.4.3 Single-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 of ADCR 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 in ADSR 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 7) is shown below:

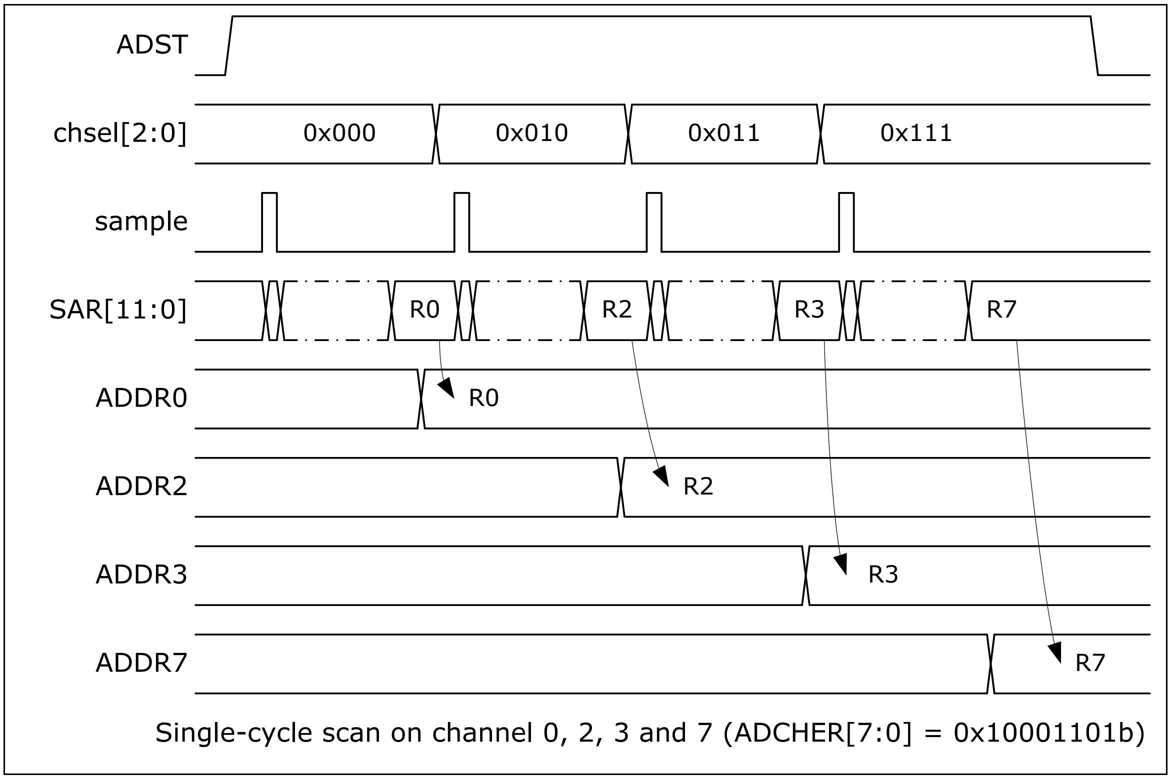


Figure 6-139 Single-Cycle Scan on Enabled Channels Timing Diagram

6.20.4.4 Continuous Scan Mode

In continuous scan mode, A/D conversion is performed sequentially on the specified channels that enabled by CHEN bits in ADCHER register (maximum 8 channels for ADC). The operations are as follows:

1. When the ADST bit in ADCR 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 7) is shown below:

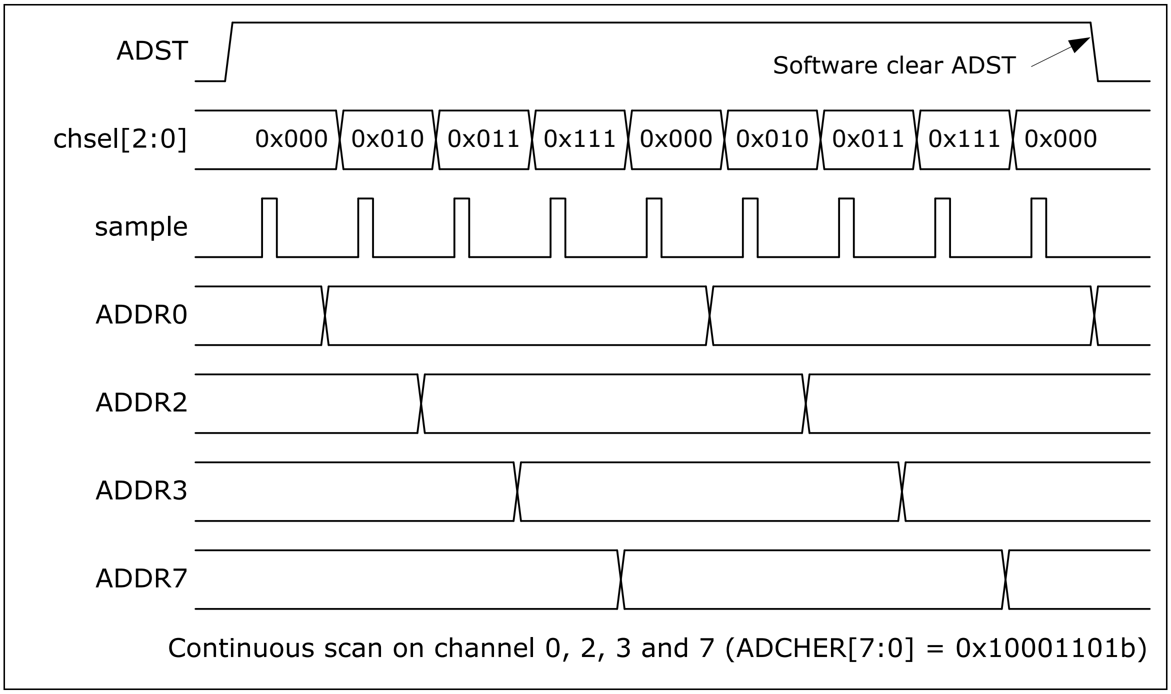


Figure 6-140 Continuous Scan on Enabled Channels Timing Diagram

6.20.4.5 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} (for LQFP 48-pin and LQFP 64-pin package) or V_{REF} (for LQFP 100-pin package) with following formula.

AV_{DD} or V_{REF} = ((2 ^ N) / R) * V_{BG}

N: ADC resolution

R: A/D conversion result

V_{BG}: Band-gap voltage

The block diagram is shown as Figure 6-141 for LQFP 48-pin and LQFP 64-pin package and Figure 6-142 for LQFP 100-pin package.

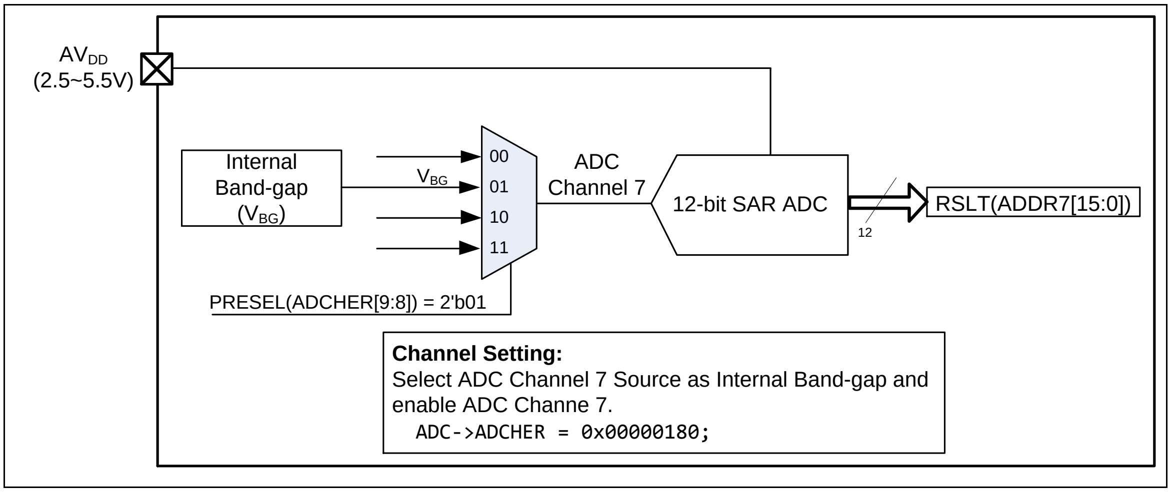


Figure 6-141 V_{BG} for Measuring AV_{DD} Application Block Diagram (for LQFP 48-pin and LQFP 64-pin Package)

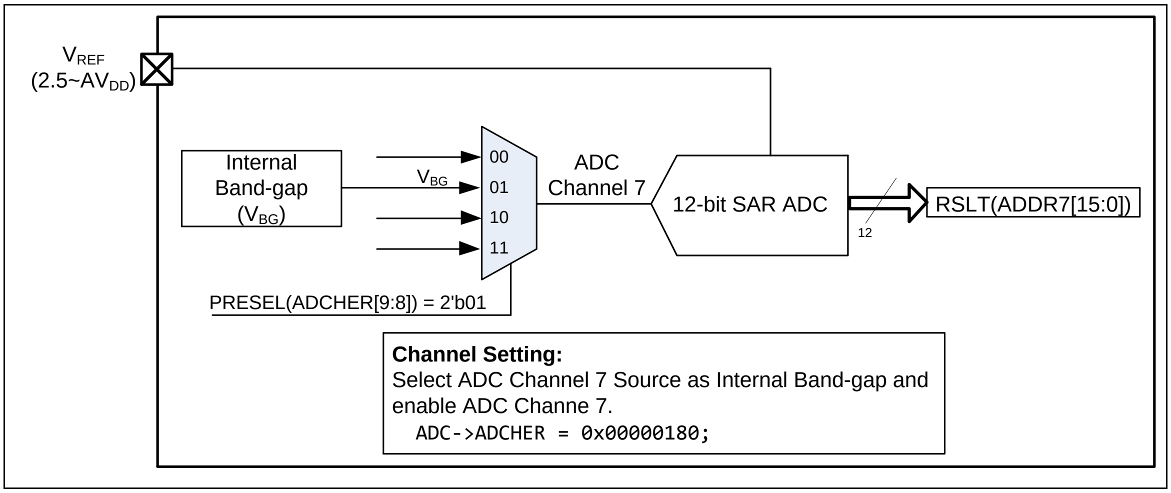


Figure 6-142 V_{BG} for Measuring V_{REF} Application Block Diagram (for LQFP 100-pin Package)

For example, the V_{BG} typical value is 1.26 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 1720:

N = 12

R = 1720

VBG = 1.25 V

AV_{DD} or V_{REF} = ((2 ^ 12) / 1720) * 1.26 = (4096 / 1720) * 1.26 = 3 V

If the A/D conversion result is 2064:

AV_{DD} or V_{REF} = ((2 ^ 12) / 2064) * 1.26 = (4096 / 2064) * 1.26 = 2.5 V

6.20.4.6 External 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 ADCR.TRGEN is set to high to enable ADC external trigger function, setting the TRGS[1:0] bits to 00b is to select external trigger input from the STADC pin. Software can set TRGCOND[1:0] 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.

6.20.4.7 PWM 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.

6.20.4.8 Conversion Result Monitor by Compare Function

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 6-143. Software can select which channel to be monitored by set CMPCH(ADCMPRx[5:0]) and CMPCOND bit is used to check conversion result is less than specify value or greater than (equal to) value specified in CMPD[11:0]. 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. When counter value reach the setting of (CMPMATCNT+1) then CMPF bit will be set to 1, if CMPIE bit 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:

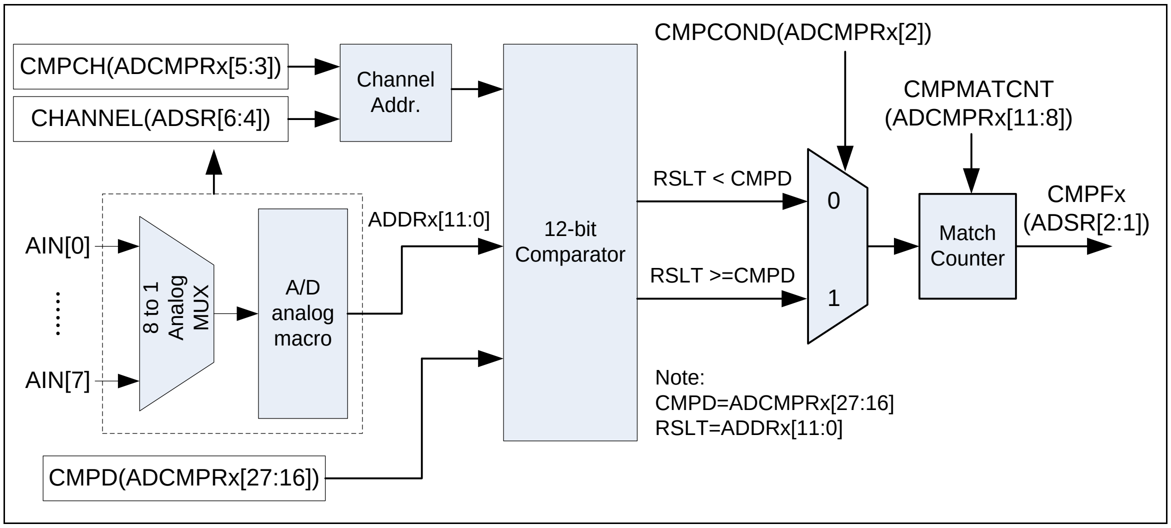


Figure 6-143 A/D Conversion Result Monitor Logics Diagram

6.20.4.9 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 and CMPF1 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, CMPF0 and CMPF1, is set to 1 and the corresponding interrupt enable bit, ADIE of ADCR and CMPIE of ADCMPR0/1, is set to 1, the ADC interrupt will be asserted. Software can clear the flag to revoke the interrupt request.

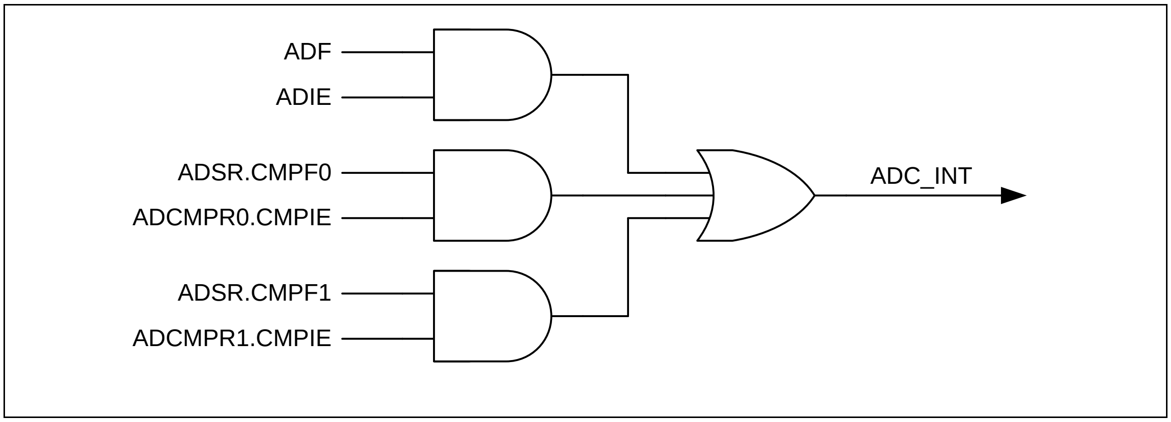


Figure 6-144 A/D Controller Interrupt

6.20.4.10 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 of ADCR 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 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.

6.20.5 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

6.20.6 Register Description

ADC Data Registers (ADDR0 ~ ADDR7)

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

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**Valid Flag (Read Only)** 0 = Data in RSLT[15:0] bits is not valid. 1 = Data in RSLT[15:0] bits is valid. **Note:** This bit is set to 1 when corresponding channel analog input conversion is completed and cleared by hardware after ADDR register is read.
[16]	**OVERRUN**	**Overrun Flag (Read Only)** 0 = Data in RSLT[15:0] is recent conversion result. 1 = Data in RSLT[15:0] is overwritten. **Note:** If converted data in RSLT[15:0] 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.

[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[11:0] and zero will be filled in RSLT[15:12]. When DMOF bit (ADCR[31]) set to 1, 12-bit ADC conversion result with 2'complement format will be filled in RSLT[11:0] and signed bits to will be filled in RSLT[15:12].

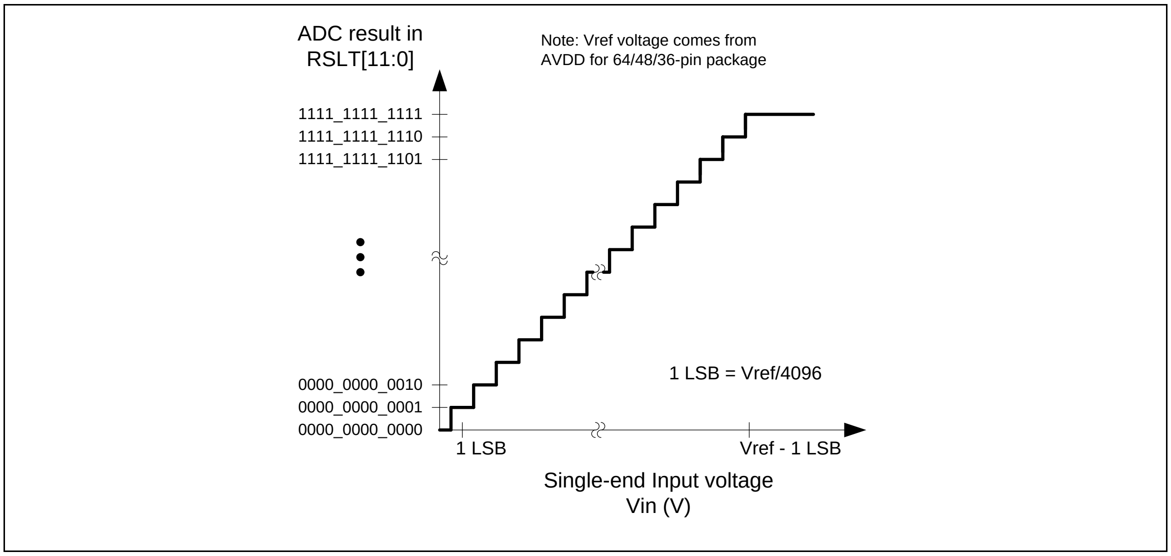


Figure 6-145 Conversion Result Mapping Diagram of Single-end Input

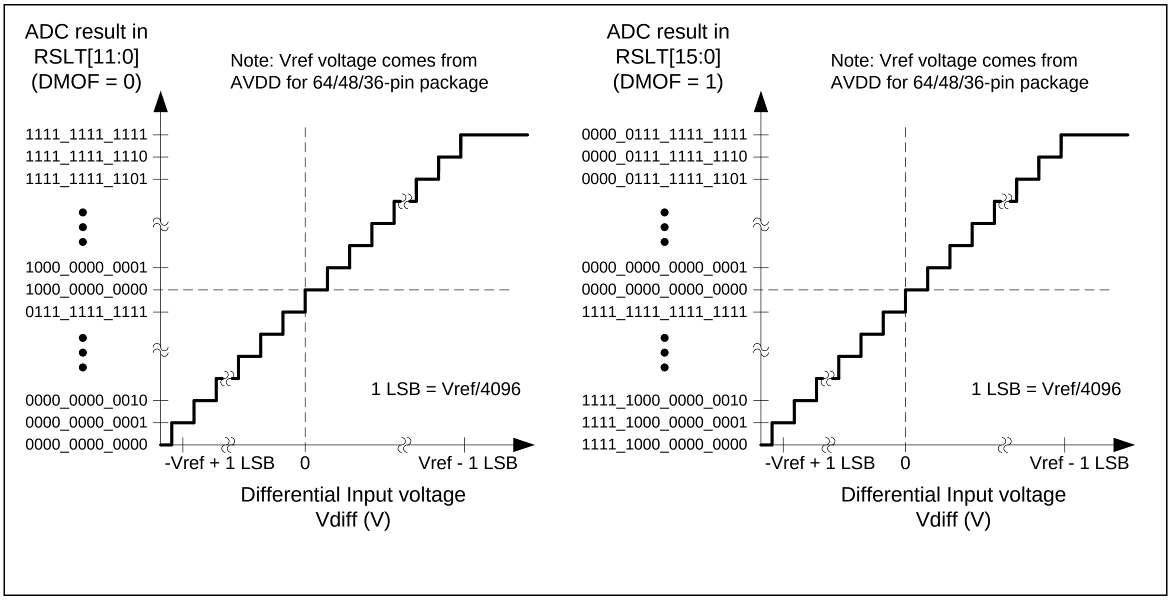


Figure 6-146 Conversion Result Mapping Diagram of Differential Input

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 Enable Bit** 0 = Single-end analog input mode. 1 = Differential analog input mode. **Differential input Paired Channel:** Channel 0: V_{plus} = ADC0, V_{minus} = ADC1. Channel 1: V_{plus} = ADC2, V_{minus} = ADC3. Channel 2: V_{plus} = ADC4, V_{minus} = ADC5. Channel 3: V_{plus} = ADC6, V_{minus} = ADC7. 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 even 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 0~7 Enabled. When A/D conversion is completed, the converted data is loaded into ADDR 0~7, software can enable this bit to generate a PDMA data transfer request. When PTEN=1, software must set ADIE=0 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 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 and ADST 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 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 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						**PRESEL**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHEN							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**PRESEL**	**Analog Input Channel 7 Select** 00 = External analog input. 01 = Internal band-gap voltage. 10 = Internal temperature sensor. 11 = Reserved. **Note:** When software select the band-gap voltage as the analog input source of ADC channel 7, ADC clock rate needs to be limited to slower than 300 kHz.
[7:0]	**CHEN**	**Analog Input Channel Enable Bit** Set CHEN[7:0] to enable the corresponding analog input channel 7 ~ 0. If DIFFEN bit 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			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPD							
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 is set to 0, ADC comparator compares CMPD with conversion result with unsigned format. CMPD should be filled in unsigned format. When DMOF bit 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[2], the internal match counter will increase 1. When the internal counter reaches the value to (CMPMATCNT +1), the CMPFx bit will be set.
[7:6]	**Reserved**	Reserved.
[5:3]	**CMPCH**	**Compare Channel Selection** 000 = Channel 0 conversion result is selected to be compared. 001 = Channel 1 conversion result is selected to be compared. 010 = Channel 2 conversion result is selected to be compared. 011 = Channel 3 conversion result is selected to be compared. 100 = Channel 4 conversion result is selected to be compared. 101 = Channel 5 conversion result is selected to be compared. 110 = Channel 6 conversion result is selected to be compared. 111 = Channel 7 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 (ADCMPRx[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 (ADCMPRx[27:16]), the internal match counter will increase one. **Note:** When the internal counter reaches the value to (CMPMATCNT +1), the CMPFx bit will be set.
[1]	**CMPIE**	**Compare Interrupt Enable Bit** 0 = Compare function interrupt Disabled. 1 = Compare function interrupt Enabled. **Note:** If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMATCNT, CMPF bit will be asserted, in the meanwhile, if CMPIE is set to 1, a compare interrupt request is generated.
[0]	**CMPEN**	**Compare Enable Bit** 0 = Compare function Disabled. 1 = Compare function Enabled. **Note:** Set this bit to 1 to enable ADC controller to compare CMPD[11:0] 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
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
OVERRUN							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VALID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	CHANNEL			BUSY	CMPF1	CMPF0	ADF

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**OVERRUN**	**Overrun Flag (Read Only)** It is a mirror to OVERRUN bit in ADDRx.
[15:8]	**VALID**	**Data Valid Flag (Read Only)** It is a mirror of VALID bit in ADDRx.
[7]	**Reserved**	Reserved.
[6:4]	**CHANNEL**	**Current Conversion Channel (Read Only)** This field reflects the current conversion channel when BUSY = 1. When BUSY = 0, it shows the number of the next converted channel.
[3]	**BUSY**	**BUSY/IDLE (Read Only)** 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion. This bit is mirror of as ADST bit in ADCR.
[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. **Note:** This bit can be cleared by writing ‘1’ to it.

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 (Read Only)** When PDMA transferring, read this register can monitor current PDMA transfer data. Current PDMA transfer data is the content of ADDR0 ~ ADDR7.

6.21 Analog Comparator (ACMP)

6.21.1 Overview

The NuMicro® NUC100 series contains two comparators which can be used in a number of different configurations. The comparator output is logic 1 when positive input voltage is greater than negative input voltage; otherwise the output is logic 0. Each comparator can be configured to cause an interrupt when the comparator output value changes. The block diagram is shown in Figure 6-147.

6.21.2 Features

- Analog input voltage range: 0~ V_{DDA}
- Supports Hysteresis function
- Supports optional internal reference voltage input at negative end for each comparator

6.21.3 Block Diagram

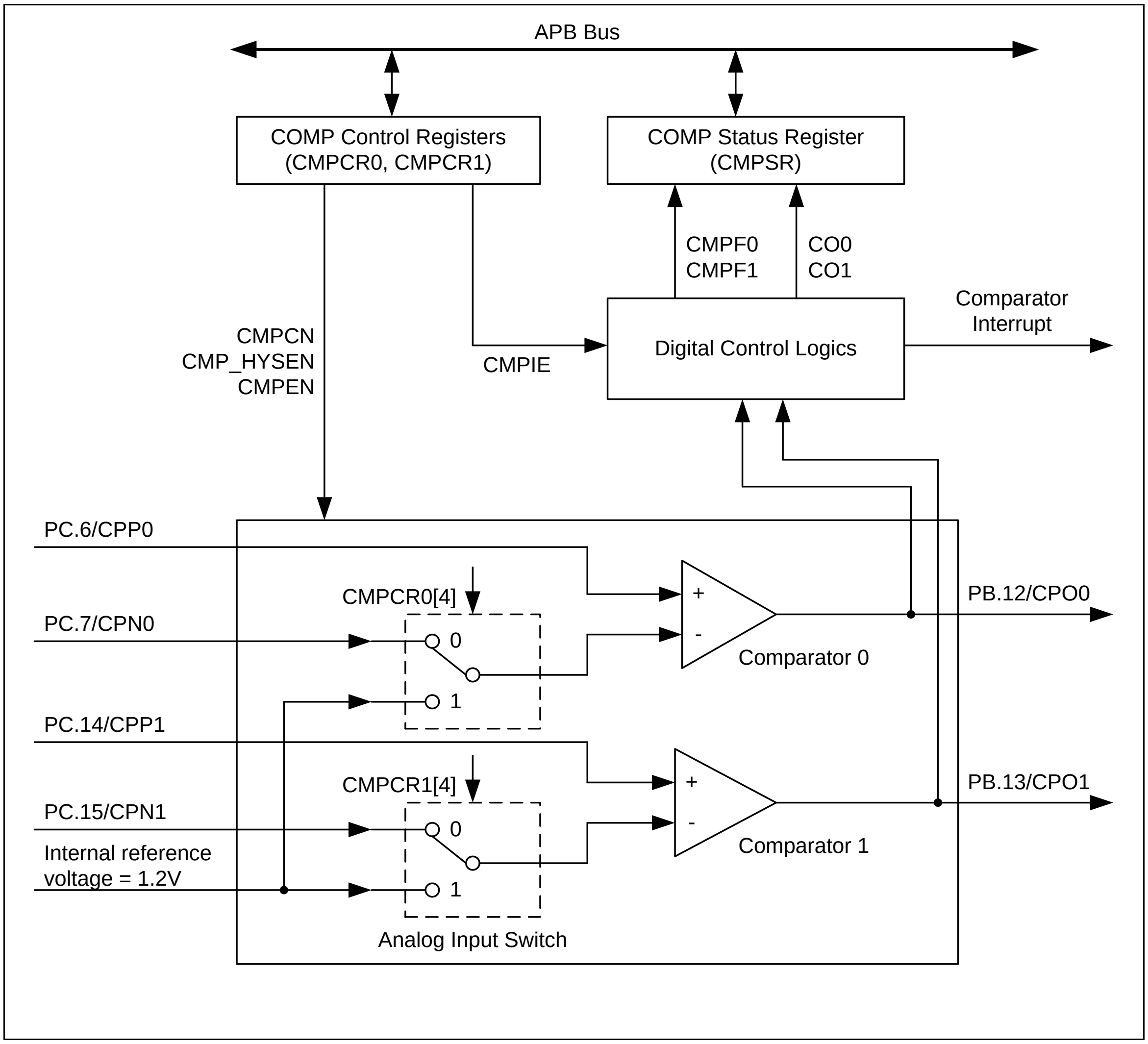


Figure 6-147 Analog Comparator Block Diagram

6.21.4 Functional Description

6.21.4.1 Interrupt Sources

The output of comparators are sampled by PCLK and reflected at CO1 and CO2 of CMPSR register. If CMP0IE/CMP1IE of CMP0CR/CMP1CR is set to 1, the comparator interrupt will be enabled. As the output state of comparator is changed, the comparator interrupt will be asserted and the corresponding flag, CMPF0 or CMPF1, will be set. Software can clear the flag to 0 by writing 1 to it.

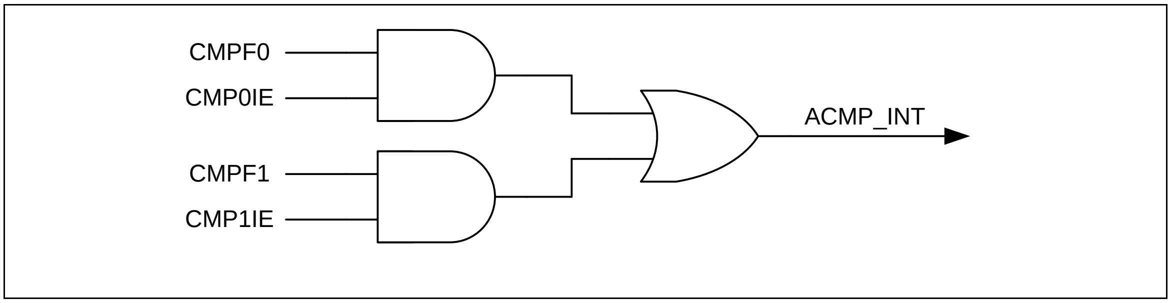


Figure 6-148 Comparator Controller Interrupt Sources

6.21.4.2 Hysteresis Function

The analog comparator provides hysteresis function to make the comparator output transition more stable. If comparator output is 0, it will not change to 1 until the positive input voltage exceeds the negative input voltage by a positive hysteresis voltage. Similarly, if comparator output is 1, it will not change to 0 until the positive input voltage drops below the negative input voltage by a negative hysteresis voltage.

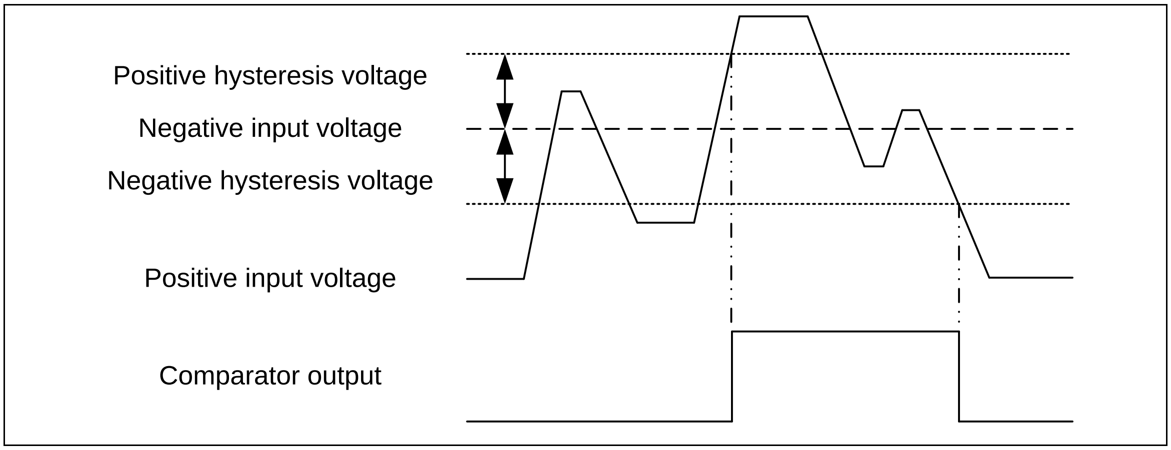


Figure 6-149 Comparator Hysteresis Function

6.21.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
ACMP Base Address: **ACMP_BA = 0x400D_0000**				
CMPCR0	ACMP_BA+0x00	R/W	Comparator 0 Control Register	0x0000_0000
CMPCR1	ACMP_BA+0x04	R/W	Comparator 1 Control Register	0x0000_0000
CMPSR	ACMP_BA+0x08	R/W	Comparator Status Register	0x0000_0000

6.21.6 Register Description

CMP Control Register 0/1 (CMPCR0/1)

Register	Offset	R/W	Description	Reset Value
CMPCR0	ACMP_BA+0x00	R/W	Comparator 0 Control Register	0x0000_0000
CMPCR1	ACMP_BA+0x04	R/W	Comparator 1 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			CMPCN	Reserved	CMP_HYSEN	CMPIE	CMPEN

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**CMPCN**	**Comparator Negative Input Selection** 0 = The source of the negative comparator input is from CPNx pin (x = 0. 1). 1 = Internal band-gap reference voltage is selected as the source of negative comparator input.
[3]	**Reserved**	Reserved.
[2]	**CMP_HYSEN**	**Comparator Hysteresis Enable Bit** 0 = Hysteresis function Disabled (Default). 1 = Hysteresis function Enabled. The typical range is 20mV.
[1]	**CMPIE**	**Comparator Interrupt Enable Bit** 0 = Interrupt function Disabled. 1 = Interrupt function Enabled.
[0]	**CMPEN**	**Comparator Enable Bit** 0 = Comparator Disabled. 1 = Comparator Enabled. **Note:** Comparator output needs to wait 2 us stable time after CMPEN is set.

CMP Status Register (CMPSR)

Register	Offset	R/W	Description	Reset Value
CMPSR	ACMP_BA+0x08	R/W	Comparator 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				**CO1**	**CO0**	**CMPF1**	**CMPF0**

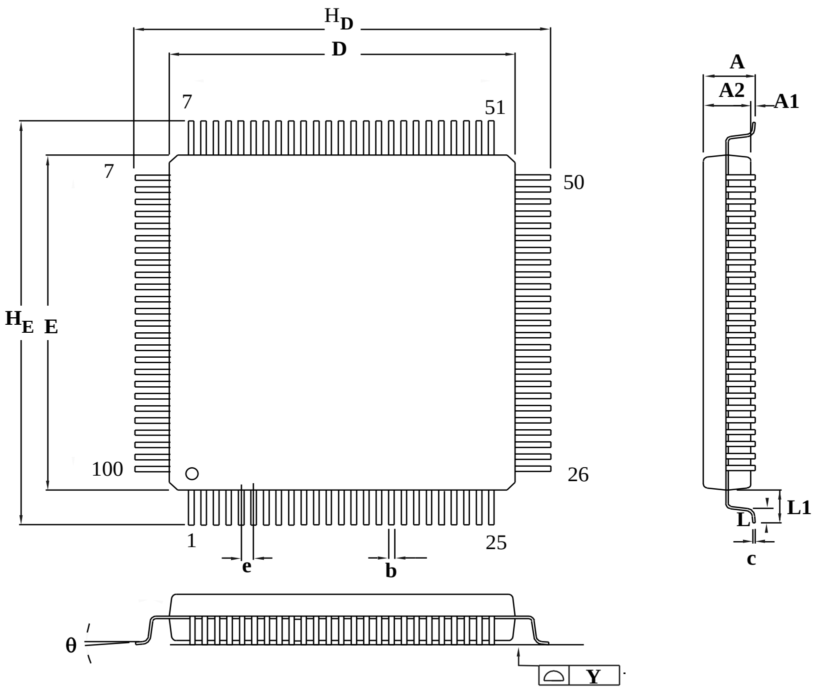
Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**CO1**	**Comparator 1 Output** Synchronized to the APB clock to allow reading by software. Cleared when the comparator 1 is disabled (CMPCR1[0] = 0).
[2]	**CO0**	**Comparator 0 Output** Synchronized to the APB clock to allow reading by software. Cleared when the comparator 0 is disabled (CMPCR0[0] = 0).
[1]	**CMPF1**	**Comparator 1 Flag** This bit is set by hardware whenever the comparator 1 output changes state. This will cause an interrupt if CMPCR1[1] is set to 1. **Note:** This bit can be cleared by writing ‘1’ to it.
[0]	**CMPF0**	**Comparator 0 Flag** This bit is set by hardware whenever the comparator 0 output changes state. This will cause an interrupt if CMPCR0[1] is set to 1. **Note:** This bit can be cleared by writing ‘1’ to it.

7 ELECTRICAL CHARACTERISTICS

For information on NuMicro® NUC100/120xxxDN electrical characteristics, please refer to NuMicro® NUC100 Series NUC100/120xxxDN Datasheet.

8 PACKAGE DIMENSIONS

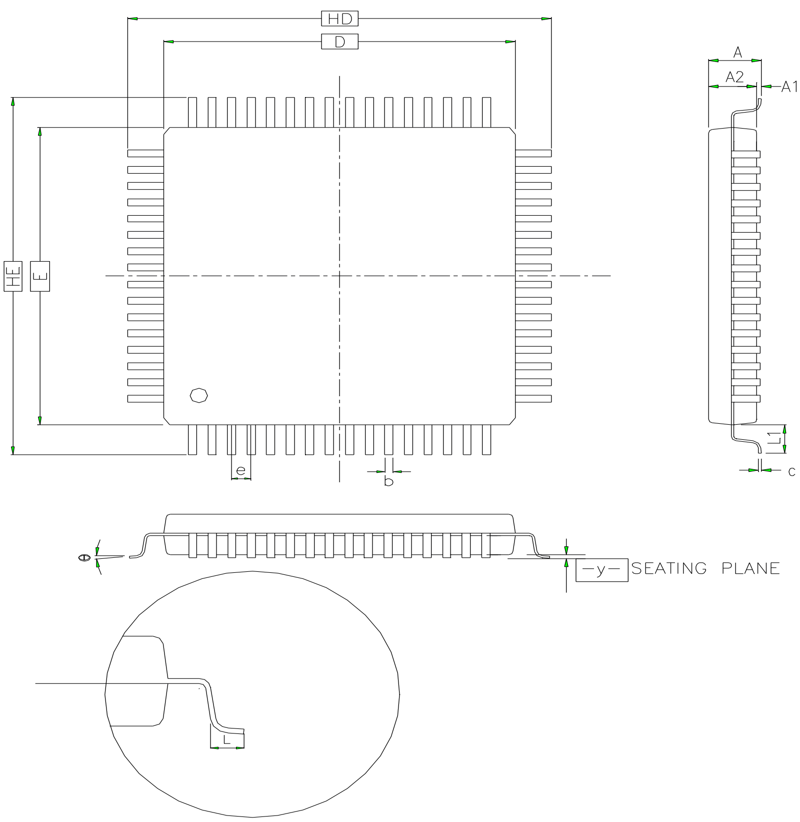
8.1 100-pin LQFP (14x14x1.4 mm footprint 2.0 mm)



Controlling Dimension : Millimeters

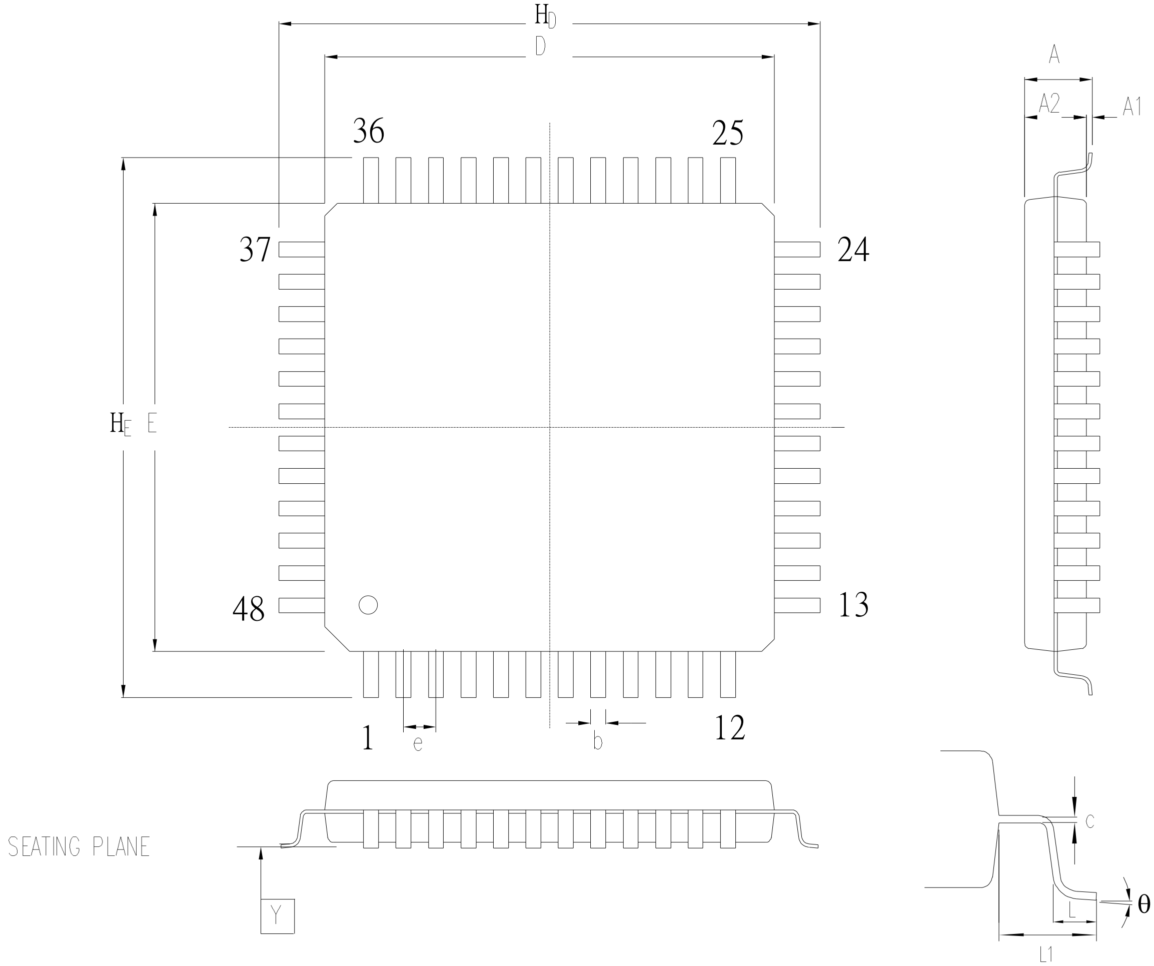
Symbol	Dimension in inch			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	—	—	0.063	—	—	1.60
A1	0.002	—	—	0.05	—	—
A	0.053	0.055	0.057	1.35	1.40	1.45
b	0.007	0.009	0.011	0.17	0.22	0.27
c	0.004	0.006	0.008	0.10	0.15	0.20
D	0.547	0.551	0.556	13.90	14.00	14.10
E	0.547	0.551	0.556	13.90	14.00	14.10
e	—	0.020	—	—	0.50	—
H_D	0.622	0.630	0.638	15.80	16.00	16.20
H_E	0.622	0.630	0.638	15.80	16.00	16.20
L	0.018	0.024	0.030	0.45	0.60	0.75
L1	—	0.039	—	—	1.00	—
y	—	—	0.004	—	—	0.10
θ	0°	—	7°	0°	—	7°

8.2 64-pin LQFP (10x10x1.4 mm footprint 2.0 mm)



Symbol	Dimension in inch			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	—	—	0.063	—	—	1.60
A_1	0.002	—	0.006	0.05	—	0.15
A_2	0.053	0.055	0.057	1.35	1.40	1.45
b	0.007	0.008	0.011	0.17	0.20	0.27
c	0.004	—	0.008	0.09	—	0.20
D	—	0.393	—	—	10.00	—
E	—	0.393	—	—	10.00	—
e	—	0.020	—	—	0.50	—
H_D	—	0.472	—	—	12.00	—
H_E	—	0.472	—	—	12.00	—
L	0.018	0.024	0.030	0.45	0.60	0.75
L_1	—	0.039	—	—	1.00	—
y	—	0.004	—	—	0.10	—
θ	0°	3.5°	7°	0°	3.5°	7°

8.3 48-pin LQFP (7x7x1.4 mm footprint 2.0 mm)



Controlling dimension : Millimeters

Symbol	Dimension in inch			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	—	—	—	—	—	—
A_1	0.002	0.004	0.006	0.05	0.10	0.15
A_2	0.053	0.055	0.057	1.35	1.40	1.45
b	0.006	0.008	0.010	0.15	0.20	0.25
c	0.004	0.006	0.008	0.10	0.15	0.20
D	0.272	0.276	0.280	6.90	7.00	7.10
E	0.272	0.276	0.280	6.90	7.00	7.10
e	0.014	0.020	0.026	0.35	0.50	0.65
H_D	0.350	0.354	0.358	8.90	9.00	9.10
H_E	0.350	0.354	0.358	8.90	9.00	9.10
L	0.018	0.024	0.030	0.45	0.60	0.75
L_1	—	0.039	—	—	1.00	—
Y	—	—	0.004	—	—	0.10
θ	0	—	7	0	—	7

9 REVISION HISTORY

Date	Revision	Description
2014.05.13	1.00	1. Preliminary version.
2015.08.31	1.01	1. Reorganized the chapter sequence. 2. Added a note in all clock source block diagrams of all peripheral sections that "Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable." 3. Revised package size of 64-pin LQFP (10x10x1.4 mm footprint 2.0 mm) in section 8.2.
2016.07.12	1.02	1. Added Section 6.2.5 Register Protection. 2. Added Section 6.20.4.5 Internal Reference Voltage.
2017.03.02	1.03	1. Updated section 4.1 NuMicro® NUC100/120xxxDN Selection Guide. 2. Fixed the reset value of register AHBCLK.
2020.04.08	1.04	1. Added notes about the hardware reference design for ICE_DAT, ICE_CLK and nRESET pins in section 4.3.

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.