NuMicro™ NUC122 Series 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™ NUC122 series are 32-bit microcontrollers with Cortex®-M0 core runs up to 60 MHz, up to 32K/64K-byte embedded flash, 4K/8K-byte embedded SRAM, and 4K-byte loader ROM for the In System Program (ISP) function. It also integrates Timers, Watchdog Timer, RTC, UART, SPI, I^2C, PWM Timer, GPIO, USB 2.0 Full Speed Device, Low Voltage Reset Controller and Brownout Detector.

Product Line	UART	SPI	I^2C	USB	PS/2
NUC122	Y	Y	Y	Y	Y

Table 1-1 Connectivity Supported Table

2 FEATURES

2.1 NuMicro™ NUC122 Features

- Core
  - ARM® Cortex®-M0 core runs up to 60 MHz
  - One 24-bit system timer
  - Support 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
- Wide operating voltage ranges from 2.5 V to 5.5 V
- Flash Memory
  - 32K/64K bytes Flash for program code
  - 4KB Flash for ISP loader
  - Support In System Program (ISP) function to update Application code
  - 512 bytes page erase for Flash
  - 4KB Data Flash
  - Support 2 wire In Circuit Program (ICP) function to update code through SWD/ICE interface
  - Support fast parallel programming mode by external programmer
- SRAM Memory
  - 4K/8K bytes embedded SRAM
- Clock Control
  - Flexible selection from different clock sources
  - Built-in 22.1184 MHz high speed OSC for system operation
    - Trimmed to ± 1 % at +25 ℃ and V_{DD} = 3.3 V
    - Trimmed to ± 5 % at -40 ℃ ~ +85 ℃ and V_{DD} = 2.5 V ~ 5.5 V
  - Built-in 10 KHz low speed OSC for Watchdog Timer and Wake-up operation
  - Support one PLL, up to 60 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 bi-direction
    - Push-Pull output
    - Open-Drain output
    - Input only with high impendence
  - TTL/Schmitt trigger input selectable
  - I/O pin can be configured as interrupt source with edge/level setting
  - High driver and high sink IO mode support
- Timers
  - 4 sets of 32-bit timers with 24-bit counters and one 8-bit prescaler
  - Counter auto reload

- Watchdog Timer
  - Multiple clock sources
  - 8 selectable time-out period from 1.6 ms ~ 26.0 sec (depends on clock source)
  - WDT can wake-up chip from power down or idle mode
  - Interrupt or reset selectable while Watchdog Timer time-out
- RTC
  - Support software compensation by setting frequency compensate register (FCR)
  - Support RTC counter (second, minute, hour) and calendar counter (day, month, year)
  - Support Alarm registers (second, minute, hour, day, month, year)
  - 12-hour or 24-hour mode
  - Automatic leap year recognition
  - Support time tick interrupt
  - Support wake-up function
- PWM/Capture
  - Built-in up to two 16-bit PWM generators provide four PWM outputs or two 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 four 16-bit digital Capture timers (shared with PWM timers) provide four rising/falling capture inputs
  - Support Capture interrupt
- UART
  - Two UART controllers
  - UART ports with flow control (TXD, RXD, CTS and RTS)
  - UART ports with 14-byte FIFO for standard device
  - Support IrDA (SIR) function
  - Support RS-485 9-bit mode and direction control
  - Programmable baud-rate generator up to 1/16 system clock
- SPI
  - Up to two sets of SPI device
  - Master up to 25 MHz, and Slave up to 12 MHz (chip is working @ 5 V)
  - Support SPI master/slave mode
  - Full duplex synchronous serial data transfer
  - Variable length of transfer data from 1 to 32 bits
  - MSB or LSB first data transfer
  - 2 slave/device select lines when it is as the master, and 1 slave/device select line when it is as the slave
  - Byte suspend mode in 32-bit transmission

- I^2C
  - One set 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 allows devices with different bit rates to communicate via one serial bus
  - Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer
  - Programmable clocks allow versatile rate control
  - I^2C-bus controller supports multiple address recognition (four slave address with mask option)
- USB 2.0 Full-Speed Device
  - One set of USB 2.0 FS Device 12 Mbps
  - On-chip USB Transceiver
  - Provide 1 interrupt source with 4 interrupt events
  - Support Control, Bulk In/Out, Interrupt and Isochronous transfers
  - Auto suspend function when no bus signaling for 3 ms
  - Provide 6 programmable endpoints
  - Include 512 bytes internal SRAM as USB buffer
  - Provide remote wake-up capability
- Brownout Detector
  - With 4 levels: 4.5 V/3.8 V/2.7 V/2.2 V
  - Support Brownout Interrupt and Reset options
- One built-in LDO
- Low Voltage Reset
- Operating Temperature: -40 ℃ ~ 85 ℃
- Packages:
  - All Green package (RoHS)
  - LQFP 64-pin (7mmX7mm)
  - LQFP 48-pin
  - QFN 33-pin

3 PARTS INFORMATION LIST AND PIN CONFIGURATION

3.1 NuMicro™ NUC122 Products Selection Guide

Part number	Flash (KB)	ISP ROM (KB)	SRAM (KB)	I/O	Timer	Connectivity						I²S	Comp.	PWM	ADC	RTC	ISP ICP	Package
						UART	SPI	I²C	USB	LIN	PS/2							
NUC122ZD2AN	64 KB	4KB	8 KB	up to 18	4x32-bit	1	2	1	1	-	-	-	-	-	-	-	v	QFN33
NUC122ZC1AN	32 KB	4KB	4 KB	up to 18	4x32-bit	1	2	1	1	-	-	-	-	-	-	-	v	QFN33
NUC122LD2AN	64 KB	4KB	8 KB	up to 30	4x32-bit	2	2	1	1	-	1	-	-	4	-	v	v	LQFP48
NUC122LC1AN	32 KB	4KB	4 KB	up to 30	4x32-bit	2	2	1	1	-	1	-	-	4	-	v	v	LQFP48
NUC122SD2AN	64 KB	4KB	8 KB	up to 41	4x32-bit	2	2	1	1	-	1	-	-	4	-	v	v	LQFP64
NUC122SC1AN	32 KB	4KB	4 KB	up to 41	4x32-bit	2	2	1	1	-	1	-	-	4	-	v	v	LQFP64

3.2 Pin Configuration

3.2.1 NuMicro™ NUC122 Pin Diagram

3.2.1.1 NuMicro™ NUC122 LQFP 64-pin

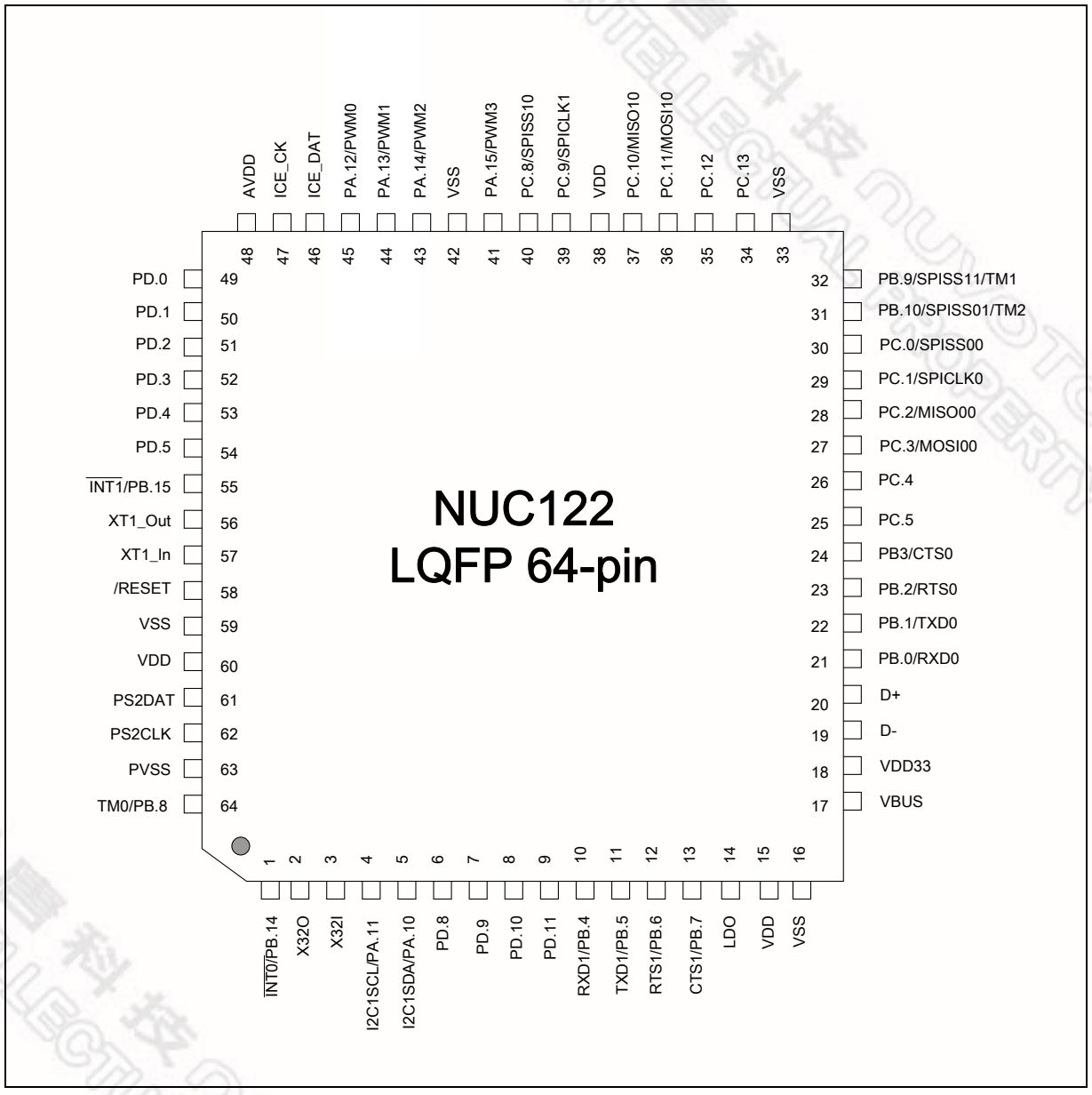


Figure 3-1 NuMicro™ NUC122 LQFP 64-pin Pin Diagram

3.2.1.2 NuMicro™ NUC122 LQFP 48-pin

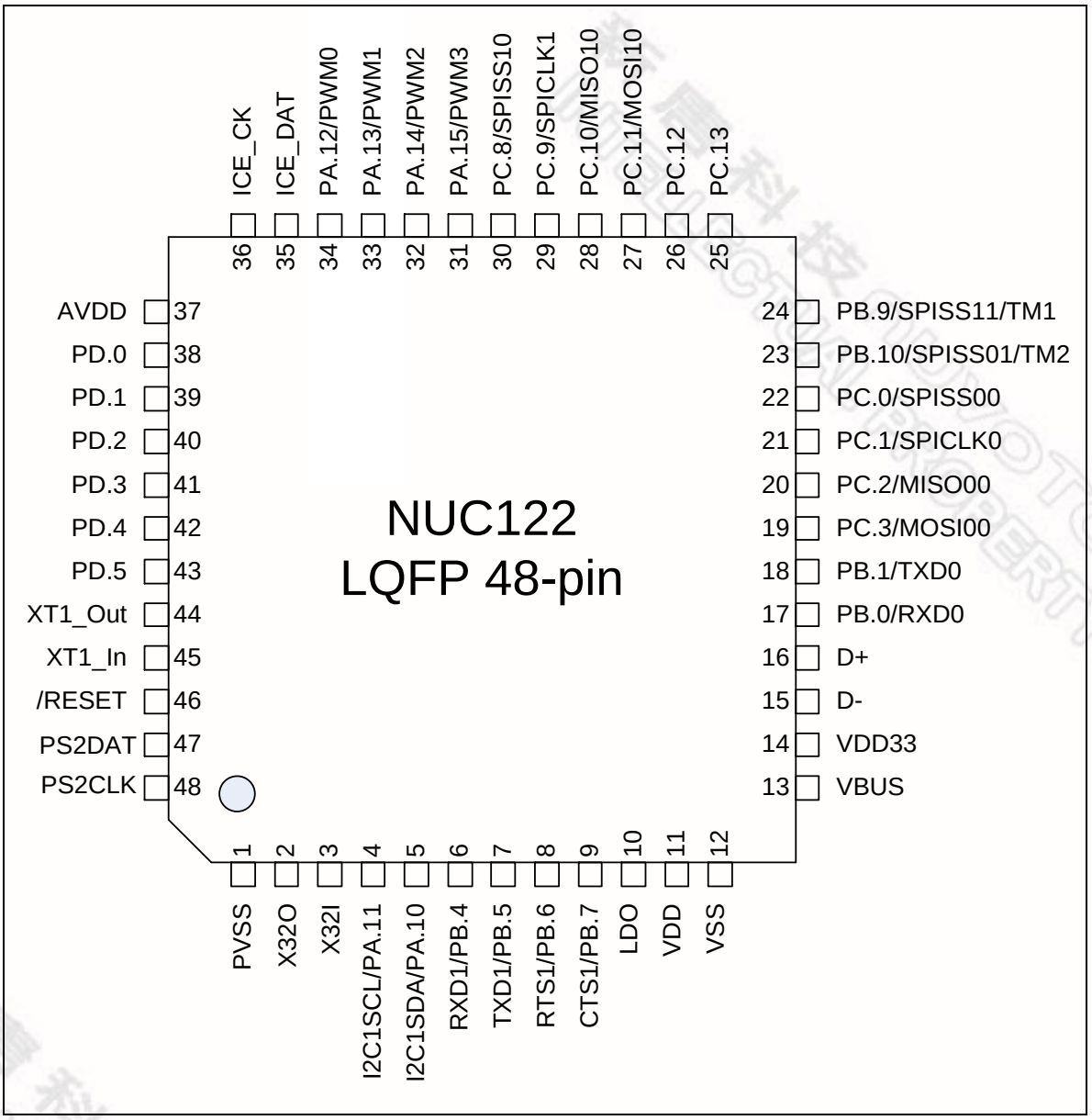


Figure 3-2 NuMicro™ NUC122 LQFP 48-pin Pin Diagram

3.2.1.3 NuMicro™ NUC122 QFN 33-pin

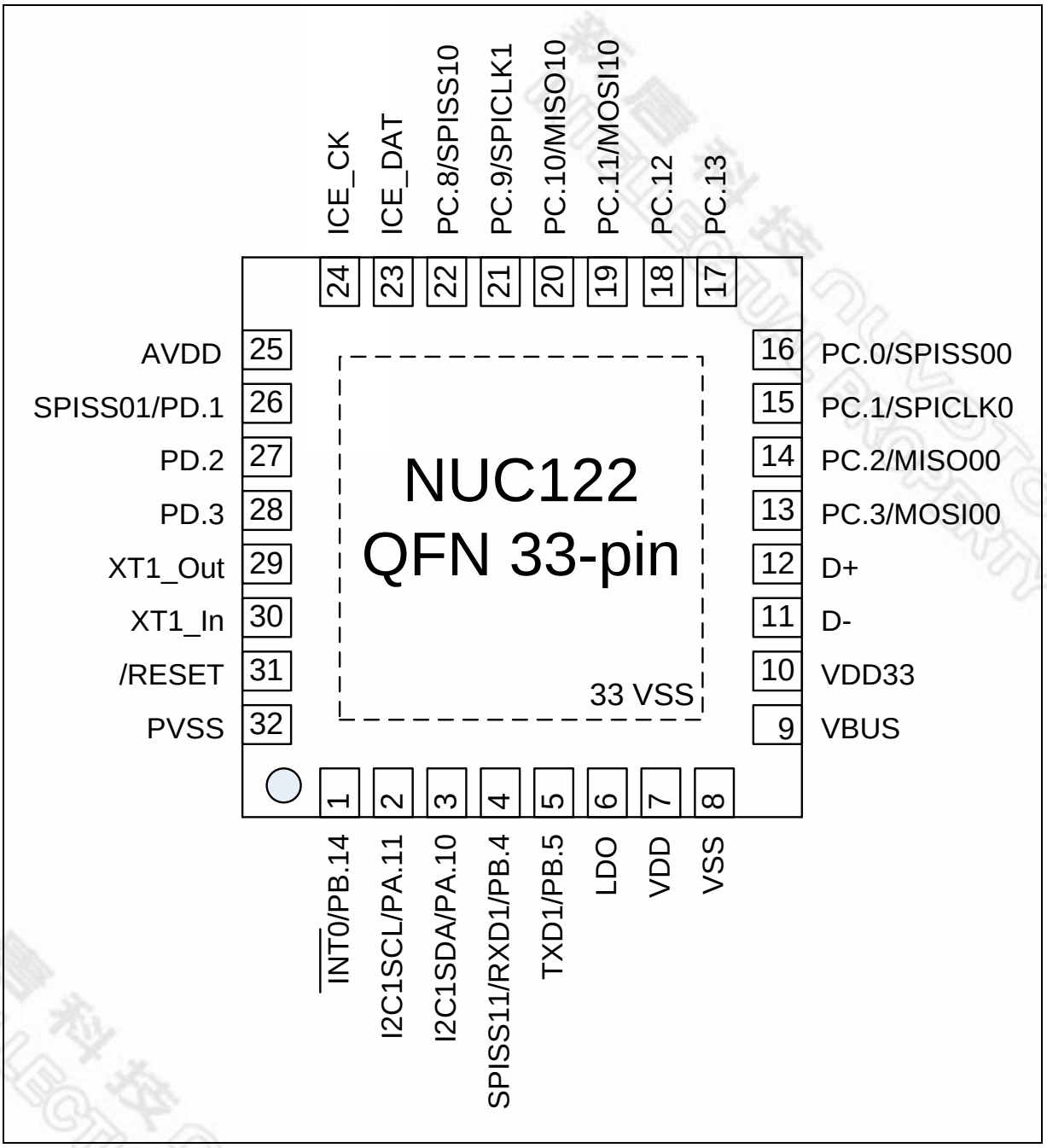


Figure 3-3 NuMicro™ NUC122 QFN 33-pin Pin Diagram

3.3 Pin Description

3.3.1 NuMicro™ NUC122 Pin Description

3.3.1.1 NuMicro™ NUC122 Pin Description for LQFP64/LQFP48/QFN33

Pin No.			Pin Name	Pin Type	Description
LQFP 64	LQFP 48	QFN 33			
1		1	PB.14	I/O	General purpose input/output digital pin
			/INT0	I	/INT0: External interrupt1 input pin
2	2		X32O	O	32.768 KHz low speed crystal output pin
3	3		X32I	I	32.768 KHz low speed crystal input pin
4	4	2	PA.11	I/O	General purpose input/output digital pin
			I2C1SCL	I/O	I2C1SCL: I^2C1 clock pin
5	5	3	PA.10	I/O	General purpose input/output digital pin
			I2C1SDA	I/O	I2C1SDA: I^2C1 data input/output pin
6			PD.8	I/O	General purpose input/output digital pin
7			PD.9	I/O	General purpose input/output digital pin
8			PD.10	I/O	General purpose input/output digital pin
9			PD.11	I/O	General purpose input/output digital pin
10	6	4	PB.4	I/O	General purpose input/output digital pin
			RXD1	I	RXD1: Data receiver input pin for UART1
			SPISS11	I/O	SPISS11: SPI1 2nd slave select pin (for QFN33 only)
11	7	5	PB.5	I/O	General purpose input/output digital pin
			TXD1	O	TXD1: Data transmitter output pin for UART1
12	8		PB.6	I/O	General purpose input/output digital pin
			RTS1	O	RTS1: Request to Send output pin for UART1
13	9		PB.7	I/O	General purpose input/output digital pin
			CTS1	I	CTS1: Clear to Send input pin for UART1
14	10	6	LDO	P	LDO output pin
15	11	7	VDD	P	Power supply for I/O ports and LDO source for internal PLL and digital function
16	12	8	VSS	P	Ground
17	13	9	VBUS	P	POWER SUPPLY: From USB Host or HUB.

Pin No.			Pin Name	Pin Type	Description
LQFP 64	LQFP 48	QFN 33			
18	14	10	VDD33	**P**	Internal Power Regulator Output 3.3 V Decoupling Pin
19	15	11	D-	**USB**	USB Differential Signal D-
20	16	12	D+	**USB**	USB Differential Signal D+
21	17		PB.0	**I/O**	General purpose input/output digital pin
			RXD0	**I**	RXD0: Data Receiver input pin for UART0
22	18		PB.1	**I/O**	General purpose input/output digital pin
			TXD0	**O**	TXD0: Data transmitter output pin for UART0
23			PB.2	**I/O**	General purpose input/output digital pin
			RTS0	**O**	RTS0: Request to Send output pin for UART0
24			PB.3	**I/O**	General purpose input/output digital pin
			CTS0	**I**	CTS0: Clear to Send input pin for UART0
25			PC.5	**I/O**	General purpose input/output digital pin
26			PC.4	**I/O**	General purpose input/output digital pin
27	19	13	PC.3	**I/O**	General purpose input/output digital pin
			MOSI00	**O**	MOSI00: SPI0 MOSI (Master Out, Slave In) pin
28	20	14	PC.2	**I/O**	General purpose input/output digital pin
			MISO00	**I**	MISO00: SPI0 MISO (Master In, Slave Out) pin
29	21	15	PC.1	**I/O**	General purpose input/output digital pin
			SPICLK0	**I/O**	SPICLK0: SPI0 serial clock pin
30	22	16	PC.0	**I/O**	General purpose input/output digital pin
			SPISS00	**I/O**	SPISS00: SPI0 slave select pin
31	23		PB.10	**I/O**	General purpose input/output digital pin
			TM2	**O**	TM2: Timer2 external counter input
			SPISS01	**I/O**	SPISS01: SPI0 2nd slave select pin
32	24		PB.9	**I/O**	General purpose input/output digital pin
			TM1	**O**	TM1: Timer1 external counter input
			SPISS11	**I/O**	SPISS11: SPI1 2nd slave select pin
33			VSS	**P**	Ground
34	25	17	PC.13	**I/O**	General purpose input/output digital pin

Pin No.			Pin Name	Pin Type	Description
LQFP 64	LQFP 48	QFN 33			
35	26	18	PC.12	I/O	General purpose input/output digital pin
36	27	19	PC.11	I/O	General purpose input/output digital pin
			MOSI10	O	MOSI10: SPI1 MOSI (Master Out, Slave In) pin
37	28	20	PC.10	I/O	General purpose input/output digital pin
			MISO10	I	MISO10: SPI1 MISO (Master In, Slave Out) pin
38			VDD	P	Power supply for I/O ports
39	29	21	PC.9	I/O	General purpose input/output digital pin
			SPICLK1	I/O	SPICLK1: SPI1 serial clock pin
40	30	22	PC.8	I/O	General purpose input/output digital pin
			SPISS10	I/O	SPISS10: SPI1 slave select pin
41	31		PA.15	I/O	General purpose input/output digital pin
			PWM3	O	PWM3: PWM output pin
42			VSS	P	Ground
43	32		PA.14	I/O	General purpose input/output digital pin
			PWM2	O	PWM2: PWM output pin
44	33		PA.13	I/O	General purpose input/output digital pin
			PWM1	O	PWM1: PWM output pin
45	34		PA.12	I/O	General purpose input/output digital pin
			PWM0	O	PWM0: PWM output pin
46	35	23	ICE_DAT	I/O	Serial Wired Debugger Data pin
47	36	24	ICE_CK	I	Serial Wired Debugger Clock pin
48	37	25	AVDD	AP	Power supply for internal analog circuit
49	38		PD.0	I/O	General purpose input/output digital pin
50	39	26	PD.1	I/O	General purpose input/output digital pin
			SPISS01	I/O	SPISS01: SPI0 2nd slave select pin (for QFN33 only)
51	40	27	PD.2	I/O	General purpose input/output digital pin
52	41	28	PD.3	I/O	General purpose input/output digital pin
53	42		PD.4	I/O	General purpose input/output digital pin
54	43		PD.5	I/O	General purpose input/output digital pin

Pin No.			Pin Name	Pin Type	Description
LQFP 64	LQFP 48	QFN 33			
55			PB.15	I/O	General purpose input/output digital pin
			/INT1	I	/INT1: External interrupt 1 input pin
56	44	29	XT1_OUT	O	Crystal output pin
57	45	30	XT1_IN	I	Crystal input pin
58	46	31	/RESET	I	External reset input: Low active, set this pin low reset chip to initial state. With internal pull-up.
59		33	VSS	P	Ground
60			VDD	P	Power supply for I/O ports
61	47		PS2DAT	I/O	PS/2 data pin
62	48		PS2CLK	I/O	PS/2 clock pin
63	1	32	PVSS	P	PLL Ground
64			PB.8	I/O	General purpose input/output digital pin
			TM0	O	TM0: Timer0 external counter input

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

4 BLOCK DIAGRAM

4.1 NuMicro™ NUC122 Block Diagram

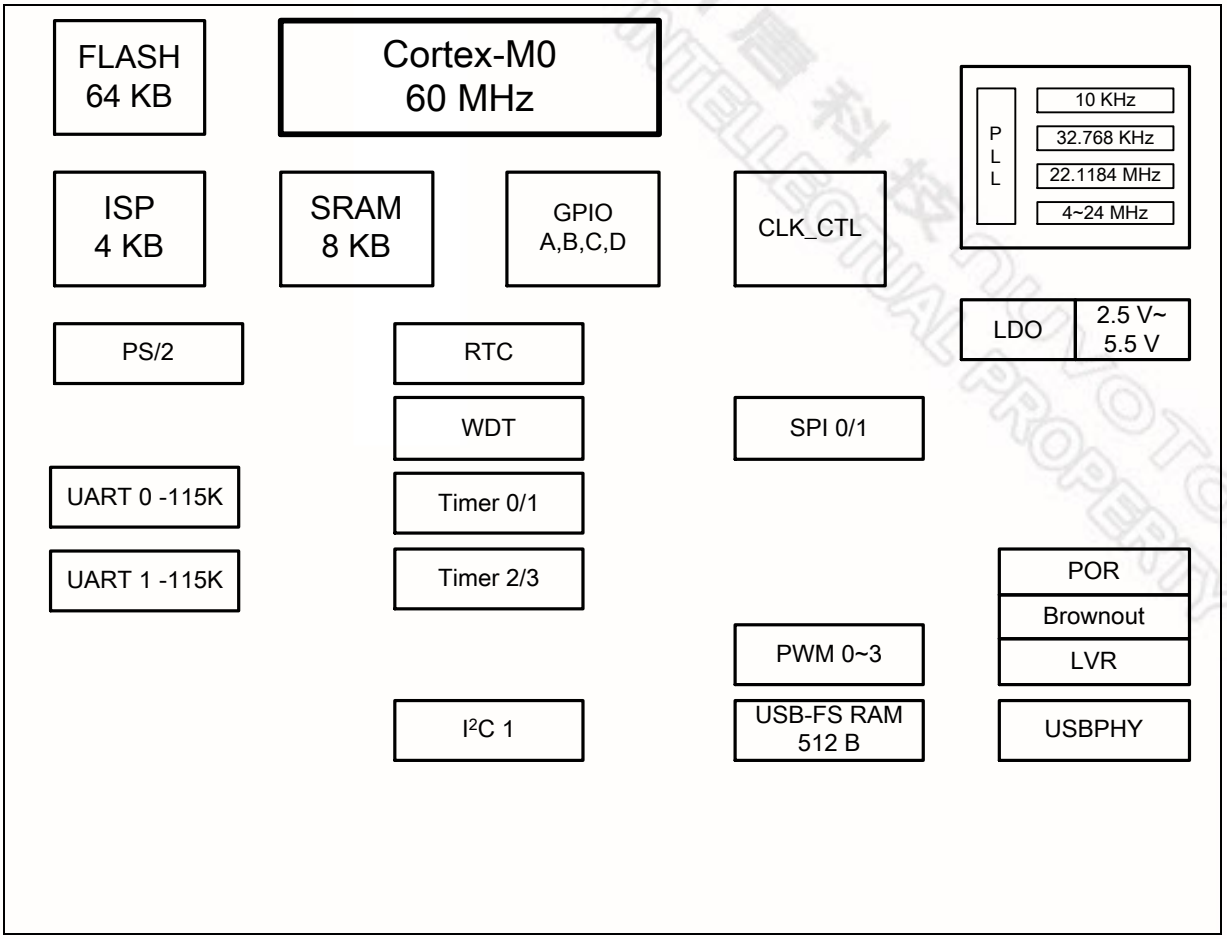


Figure 4-1 NuMicro™ NUC122 Block Diagram

5 FUNCTIONAL DESCRIPTION

5.1 ARM® Cortex®-M0 Core

The Cortex®-M0 processor is a configurable, multistage, 32-bit RISC processor. It 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. Following figure shows the functional controllers of processor.

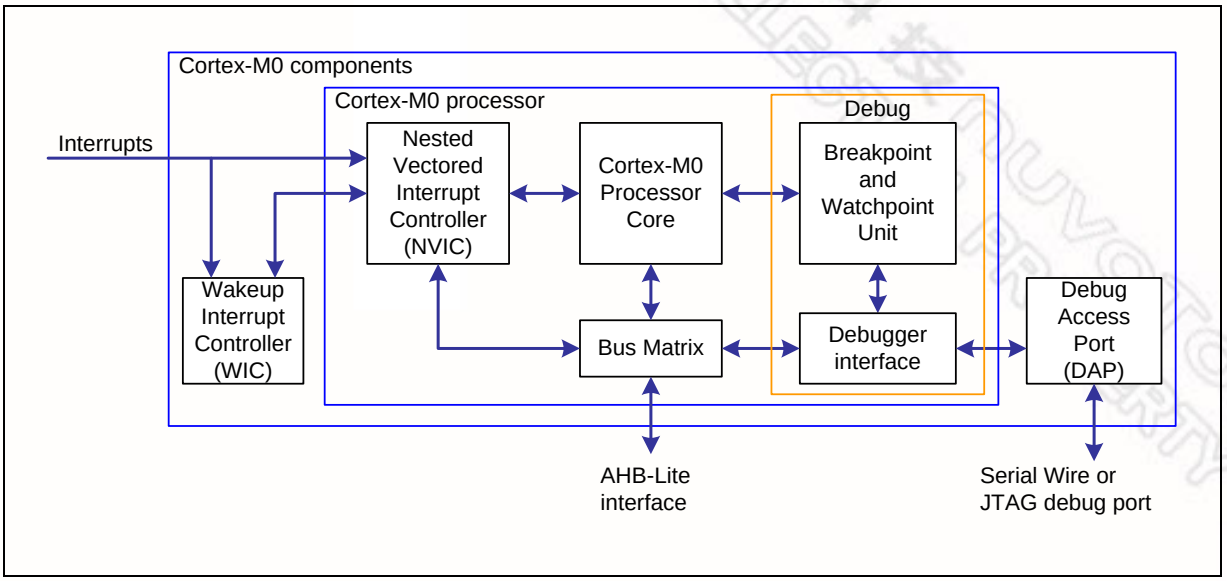


Figure 5-1 Functional Controller Diagram

The implemented device provides:

- A low gate count processor that features:
  - The ARM® v6-M Thumb® instruction set
  - Thumb-2 technology
  - ARM® v6-M compliant 24-bit SysTick timer
  - A 32-bit hardware multiplier
  - The system interface supports little-endian data accesses
  - The ability to have deterministic, fixed-latency, and 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 ARM® v6-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 that features:
  - 32 external interrupt inputs, each with four levels of priority
  - Dedicated Non-Maskable Interrupt (NMI) input.
  - Support for both level-sensitive and pulse-sensitive interrupt lines
  - Wake-Up Interrupt Controller (WIC), providing ultra-low power sleep mode support.

- 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).

5.2 System Manager

5.2.1 Overview

System management includes these following sections:

- System Resets
- System Memory Map
- System management registers for Part Number ID, chip reset and on-chip controllers reset, multi-functional pin control
- System Timer (SysTick)
- Nested Vectored Interrupt Controller (NVIC)
- System Control registers

5.2.2 System Reset

The system reset can be issued by one of the below listed events. These reset event flags can be read from RSTSRC register.

- The Power-On Reset
- The low level on the /RESET pin
- Watchdog Timer Time-Out Reset
- Low Voltage Reset
- Brownout Detector Reset
- Cortex®-M0 Reset
- System Reset

Both System Reset and Power-On Reset can reset the whole chip including all peripherals. The difference between System Reset and Power-On Reset is external Crystal circuit and ISPCON.BS bit. System Reset doesn't reset external Crystal circuit and ISPCON.BS bit, but Power-On Reset does.

5.2.3 System Power Distribution

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

- Analog power from AVDD and AVSS provides the power for analog components operation.
- Digital power from VDD and VSS 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 VBUS offers the power for operating the USB transceiver.

The outputs of internal voltage regulators, LDO and VDD33, require an external capacitor which should be located close to the corresponding pin. Analog power (AVDD) should be the same voltage level of the digital power (VDD). The following diagram shows the power distribution of this chip.

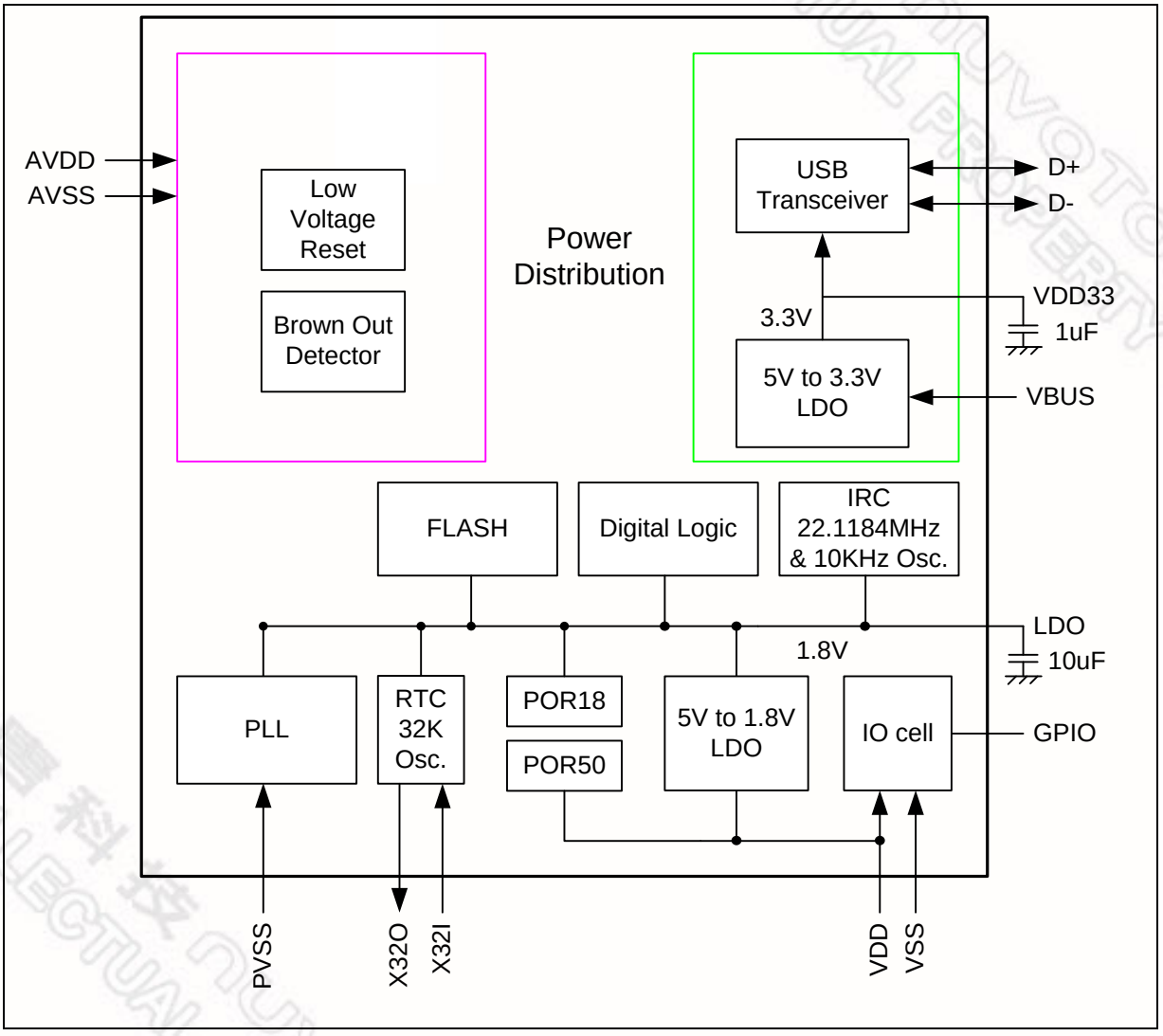


Figure 5-2 NuMicro™ NUC122 Power Distribution Diagram

5.2.4 System Memory Map

NuMicro™ NUC122 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 peripherals. NuMicro™ NUC122 Series only supports little-endian data format.

Address Space	Token	Controllers
Flash & SRAM Memory Space		
0x0000_0000 – 0x0000_FFFF	FLASH_BA	FLASH Memory Space (64KB)
0x2000_0000 – 0x2000_1FFF	SRAM_BA	SRAM Memory Space (8KB)
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_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
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
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
0x4015_0000 – 0x4015_3FFF	UART1_BA	UART1 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 5-1 Address Space Assignments for On-Chip Controller

5.2.5 System Manager Control Registers

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

Register	Offset	R/W	Description	Reset Value
GCR_BA = 0x5000_0000				
PDID	GCR_BA+0x00	R	Part Device Identification Number Register	0x0014_0018[1]
RSTSRC	GCR_BA+0x04	R/W	System Reset Source Register	0x0000_00XX
IPRSTC1	GCR_BA+0x08	R/W	Peripheral Reset Control Register 1	0x0000_0000
IPRSTC2	GCR_BA+0x0C	R/W	Peripheral Reset Control Register 2	0x0000_0000
BODCR	GCR_BA+0x18	R/W	Brownout Detector Control Register	0x0000_008X
PORCR	GCR_BA+0x24	R/W	Power-On Reset Control 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
ALT_MFP	GCR_BA+0x50	R/W	Alternative Multiple Function Pin Control Register	0x0000_0000
REGWRPROT	GCR_BA+0x100	R/W	Register Write Protect Register	0x0000_0000

Note: [1] Dependents on part number.

Part Device ID Code Register (PDID)

Register	Offset	R/W	Description	Reset Value
PDID	GCR_BA+0x00	R	Part Device Identification Number Register	0x0014_0018[1]

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

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

Bits	Descriptions	
[31:0]	**PDID**	**Part Device Identification Number** This register reflects device part number code. S/W 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	Descriptions	
[31:8]	**Reserved**	Reserved
[7]	**RSTS_CPU**	The RSTS_CPU flag is set by hardware if software writes CPU_RST (IPRSTC1[1]) 1 to reset Cortex®-M0 CPU kernel and Flash memory controller (FMC). 1 = The Cortex®-M0 CPU kernel and FMC are reset by software setting CPU_RST to 1. 0 = No reset from CPU Software can write 1 to clear this bit to zero.
[6]	**Reserved**	Reserved
[5]	**RSTS_SYS**	The RSTS_SYS flag is set by the "reset signal" from the Cortex®-M0 kernel to indicate the previous reset source. 1 = The Cortex®-M0 had issued the reset signal to reset the system by software writing 1 to bit SYSRESETREQ(AIRCR[2], Application Interrupt and Reset Control Register, address = 0xE000ED0C) in system control registers of Cortex®-M0 kernel. 0 = No reset from Cortex®-M0 Software can write 1 to clear this bit to zero.
[4]	**RSTS_BOD**	The RSTS_BOD flag is set by the "reset signal" from the Brownout Detector to indicate the previous reset source. 1 = The BOD had issued the reset signal to reset the system 0 = No reset from BOD Software can write 1 to clear this bit to zero.
[3]	**RSTS_LVR**	The RSTS_LVR flag is set by the "reset signal" from the Low-Voltage-Reset controller to

		indicate the previous reset source. 1 = The LVR controller had issued the reset signal to reset the system. 0 = No reset from LVR Software can write 1 to clear this bit to zero.
[2]	**RSTS_WDT**	The RSTS_WDT flag is set by the “reset signal” from the Watchdog Timer to indicate the previous reset source. 1 = The Watchdog Timer had issued the reset signal to reset the system. 0 = No reset from Watchdog Timer Software can write 1 to clear this bit to zero.
[1]	**RSTS_RESET**	The RSTS_RESET flag is set by the “reset signal” from the /RESET pin to indicate the previous reset source. 1 = The Pin /RESET had issued the reset signal to reset the system. 0 = No reset from /RESET pin Software can write 1 to clear this bit to zero.
[0]	**RSTS_POR**	The RSTS_POR flag is set by the “reset signal” from the Power-On Reset (POR) controller or bit CHIP_RST (IPRSTC1[0]) to indicate the previous reset source. 1 = The Power-On Reset (POR) or CHIP_RST had issued the reset signal to reset the system. 0 = No reset from POR or CHIP_RST Software can write 1 to clear this bit to zero.

Peripheral Reset Control Register 1 (IPRSTC1)

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

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

Bits	Descriptions	
[31:2]	**Reserved**	Reserved
[1]	**CPU_RST**	**CPU Kernel One Shot Reset** (write-protection bit) Setting this bit will only reset the CPU kernel and Flash Memory Controller(FMC), and this bit will automatically return to 0 after the 2 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. Reference the register REGWRPROT at address GCR_BA+0x100 1 = CPU one shot reset 0 = CPU normal operation
[0]	**CHIP_RST**	**CHIP One Shot Reset** (write-protection bit) Setting this bit will reset the whole chip, including CPU kernel and all peripherals, and this bit will automatically return to 0 after the 2 clock cycles. The CHIP_RST is same as the POR reset, all the chip controllers is reset and the chip setting from flash are also reload. About 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. Reference the register REGWRPROT at address GCR_BA+0x100 1 = CHIP one shot reset 0 = CHIP normal operation

Peripheral Reset Control Register 2 (IPRSTC2)

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

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

31	30	29	28	27	26	25	24
Reserved				**USBD_RST**	**Reserved**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PS2_RST	**Reserved**	**Reserved**	**PWM03_RST**	**Reserved**	**Reserved**	**UART1_RST**	**UART0_RST**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**Reserved**	**SPI1_RST**	**SPI0_RST**	**Reserved**		**I2C1_RST**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TMR3_RST**	**TMR2_RST**	**TMR1_RST**	**TMR0_RST**	**GPIO_RST**	**Reserved**

Bits	Descriptions	
[31:28]	**Reserved**	Reserved
[27]	**USBD_RST**	**USB Device Controller Reset** 1 = USB device controller reset 0 = USB device controller normal operation
[26:24]	**Reserved**	Reserved
[23]	**PS2_RST**	**PS/2 Controller Reset** 1 = PS/2 controller reset 0 = PS/2 controller normal operation
[22:21]	**Reserved**	Reserved
[20]	**PWM03_RST**	**PWM03 Controller Reset** 1 = PWM03 controller reset 0 = PWM03 controller normal operation
[19:18]	**Reserved**	Reserved
[17]	**UART1_RST**	**UART1 Controller Reset** 1 = UART1 controller reset 0 = UART1 controller normal operation
[16]	**UART0_RST**	**UART0 Controller Reset**

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

Brownout Detector Control Register (BODCR)

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

Register	Offset	R/W	Description	Reset Value
BODCR	GCR_BA+0x18	R/W	Brownout 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_LP**	**BOD_INTF**	**BOD_RSTEN**	**BOD_VL**		**BOD_EN**

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7]	**LVR_EN**	**Low Voltage Reset Enable** (write-protection bit) The LVR function reset the chip when the input power voltage is lower than LVR circuit setting. LVR function is enabled in default. 1 = Enabled Low Voltage Reset function – After enabling the bit, the LVR function will be active with 100uS delay for LVR output stable. (default). 0 = Disabled Low Voltage Reset function 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
[6]	**BOD_OUT**	**Brownout Detector Output Status** 1 = Brownout 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 0 0 = Brownout Detector output status is 0. It means the detected voltage is higher than BOD_VL setting or BOD_EN is 0
[5]	**BOD_LPM**	**Brownout Detector Low Power Mode** (write-protection bit) 1 = Enable the BOD low power mode 0 = BOD operate in normal mode (default) The BOD consumes about 100 uA in normal mode, the low power mode can reduce the current to about 1/10 but slow the BOD response.

		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.
[4]	**BOD_INTF**	**Brownout Detector Interrupt Flag** 1 = When Brownout Detector detects the VDD is dropped down through the voltage of BOD_VL setting or the VDD is raised up through the voltage of BOD_VL setting, this bit is set to 1 and the brownout interrupt is requested if brownout interrupt is enabled. 0 = Brownout Detector does not detect any voltage draft at VDD down through or up through the voltage of BOD_VL setting. Software can write 1 to clear this bit to zero.
[3]	**BOD_RSTEN**	**Brownout Reset Enable** (write-protection bit) 1 = Enable the brownout “RESET” function While the Brownout 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). 0 = Enable the brownout “INTERRUPT” function 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). 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. Reference the register REGWRPROT at address GCR_BA+0x100.
[2:1]	**BOD_VL**	**Brownout Detector Threshold Voltage Selection** (write-protection bits) The default value is set by flash controller user configuration register config0 bit[22:21] 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.

BOV_VL[1]	BOV_VL[0]	Brownout voltage
1	1	4.5 V
1	0	3.8 V
0	1	2.7 V
0	0	2.2 V

[0]	**BOD_EN**	**Brownout Detector Enable** (write-protection bit) The default value is set by flash controller user configuration register config0 bit[23] 1 = Brownout Detector function is enabled 0 = Brownout Detector function is disabled 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.

Power-On Reset Control Register (PORCR)

Register	Offset	R/W	Description	Reset Value
PORCR	GCR_BA+0x24	R/W	Power-On Reset Control 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[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POR_DIS_CODE[7:0]							

Bits	Descriptions	
[31:16]	**Reserved**	Reserved
[15:0]	**POR_DIS_CODE**	**The register is used for the Power-On Reset enable control** (write-protection bits) When power 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 pin, Watchdog Timer Time-Out reset, LVR reset, BOD reset, ICE reset command and the software-chip reset function 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.

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[15:8]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
GPA_TYPE[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
GPA_MFP[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPA_MFP[7:0]							

Bits	Descriptions	
[31:16]	**GPA_TYPEn**	1 = Enable GPIOA[15:0] I/O input Schmitt Trigger function 0 = Disable GPIOA[15:0] I/O input Schmitt Trigger function GPA[9:0] are reserved
[15]	**GPA_MFP15**	**PA.15 Pin Function Selection** The pin function depends on GPA_MFP15 and ALT_MFP[9] ALT_MFP[9] / GPA_MFP[15] / PA.15 function: x / 0 / **GPIO** 0 / 1 / **PWM3** (PWM) 1 / 1 / **Reserved**
[14]	**GPA_MFP14**	**PA.14 Pin Function Selection** The pin function depends on GPA_MFP14 and ALT_MFP[11]. ALT_MFP[11] / GPA_MFP[14] / PA.14 function: x / 0 / **GPIO** 0 / 1 / **PWM2** (PWM) 1 / 1 / **Reserved**
[13]	**GPA_MFP13**	**PA.13 Pin Function Selection** The pin function depends on GPA_MFP13 and ALT_MFP[11]. ALT_MFP[11] / GPA_MFP[13] / PA.13 function:

		x | 0 | **GPIO** 0 | 1 | **PWM1** (PWM) 1 | 1 | **Reserved**
[12]	**GPA_MFP12**	**PA.12 Pin Function Selection** The pin function depends on GPA_MFP12 and ALT_MFP[11]. ALT_MFP[11] | GPA_MFP[12] | PA.12 function x | 0 | **GPIO** 0 | 1 | **PWM0** (PWM) 1 | 1 | **Reserved**
[11]	**GPA_MFP11**	**PA.11 Pin Function Selection** The pin function depends on GPA_MFP11 and ALT_MFP[11]. ALT_MFP[11] | GPA_MFP[11] | PA.11 function x | 0 | **GPIO** 0 | 1 | **SCL1** (I^2C) 1 | 1 | **Reserved**
[10]	**GPA_MFP10**	**PA.10 Pin Function Selection** The pin function depends on GPA_MFP10 and ALT_MFP[11]. ALT_MFP[11] | GPA_MFP[10] | PA.10 function x | 0 | **GPIO** 0 | 1 | **SDA1** (I^2C) 1 | 1 | **Reserved**
[9:0]	**Reserved**	Reserved

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[15:8]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
GPB_TYPE[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
GPB_MFP[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPB_MFP[7:0]							

Bits	Descriptions	
[31:16]	**GPB_TYPEn**	1 = Enable GPIOB[15:0] I/O input Schmitt Trigger function 0 = Disable GPIOB[15:0] I/O input Schmitt Trigger function GPB[13:11],are reserved
[15]	**GPB_MFP15**	**PB.15 Pin Function Selection** The pin function depends on GPB_MFP15 GPB_MFP[15] = 0 → **GPIO** GPB_MFP[15] = 1 → **/INT1**
[14]	**GPB_MFP14**	**PB.14 Pin Function Selection** The pin function depends on GPB_MFP14 and ALT_MFP[3] ALT_MFP[3] = x, GPB_MFP[14] = 0 → **GPIO** ALT_MFP[3] = 0, GPB_MFP[14] = 1 → **/INT0** ALT_MFP[3] = 1, GPB_MFP[14] = 1 → Reserved
[13:11]	Reserved	Reserved
[10]	**GPB_MFP10**	**PB.10 Pin Function Selection** The pin function depends on GPB_MFP10 and PB10_S01 (ALT_MFP[0]).

		PB10_S01 / GPB_MFP[10] / PB.10 function x / 0 / **GPIO** 0 / 1 / **TM2** 1 / 1 / **SPISS01** (SPI0) (the validity of this function is depended on part no)
[9]	**GPB_MFP9**	**PB.9 Pin Function Selection** The pin function depends on GPB_MFP9 and PB9_S11 (ALT_MFP[1]). PB9_S11 / GPB_MFP[9] / PB.9 function x / 0 / **GPIO** 0 / 1 / **TM1** 1 / 1 / **SPISS11** (SPI1) (the validity of this function is depended on part no)
[8]	**GPB_MFP8**	**PB.8 Pin Function Selection** The pin function depends on GPB_MFP8 GPB_MFP[8] / PB.8 function 0 / **GPIO** 1 / **TM0**
[7]	**GPB_MFP7**	**PB.7 Pin Function Selection** The pin function depends on GPB_MFP7 and ALT_MFP[16]. ALT_MFP[16] / GPB_MFP[7] / PB.7 function 0 / 0 / **GPIO** 0 / 1 / **CTS1** (UART1) 1 / x / **Reserved**
[6]	**GPB_MFP6**	**PB.6 Pin Function Selection** The pin function depends on GPB_MFP6 and ALT_MFP[17]. ALT_MFP[17] / GPB_MFP[6] / PB.6 function 0 / 0 / **GPIO** 0 / 1 / **RTS1** (UART1) 1 / x / **Reserved**
[5]	**GPB_MFP5**	**PB. 5 Pin Function Selection** 1 = The UART1 TXD function is selected to the pin PB.5 0 = The GPIOB[5] is selected to the pin PB.5

<table>
<tr><td>[4]</td><td>GPB_MFP4</td><td>PB.4 Pin Function Selection
1 = The UART1 RXD function is selected to the pin PB.4
0 = The GPIOB[4] is selected to the pin PB.4
The pin function depends on GPB_MFP4 and ALT_MFP[15].
<table>
<tr><th>ALT_MFP[15]</th><th>GPB_MFP[4]</th><th>PB.4 function</th></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>RXD (UART1)</td></tr>
<tr><td>1</td><td>1</td><td>SPISS11 (SPI1)
(the validity of this function is depended on part no)</td></tr>
</table></td></tr>
<tr><td>[3]</td><td>GPB_MFP3</td><td>PB.3 Pin Function Selection
The pin function depends on GPB_MFP3 and ALT_MFP[11].
<table>
<tr><th>ALT_MFP[11]</th><th>GPB_MFP[3]</th><th>PB.3 function</th></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>CTS0 (UART0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[2]</td><td>GPB_MFP2</td><td>PB.2 Pin Function Selection
The pin function depends on GPB_MFP2 and ALT_MFP[11].
<table>
<tr><th>ALT_MFP[11]</th><th>GPB_MFP[2]</th><th>PB.2 function</th></tr>
<tr><td>x</td><td>0</td><td>GPIO</td></tr>
<tr><td>0</td><td>1</td><td>RTS0 (UART0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[1]</td><td>GPB_MFP1</td><td>PB.1 Pin Function Selection
1= The UART0 TXD function is selected to the pin PB.1
0= The GPIOB[1] is selected to the pin PB.1</td></tr>
<tr><td>[0]</td><td>GPB_MFP0</td><td>PB.0 Pin Function Selection
1= The UART0 RXD function is selected to the pin PB.0
0= The GPIOB[0] is selected to the pin PB.0</td></tr>
</table>

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[15:8]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
GPC_TYPE[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
GPC_MFP[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPC_MFP[7:0]							

Bits	Descriptions	
[31:16]	**GPC_TYPEn**	1 = Enable GPIOC[15:0] I/O input Schmitt Trigger function 0 = Disable GPIOC[15:0] I/O input Schmitt Trigger function GPC[15:14], GPC[7:6] are reserved
[15:14]	Reserved	Reserved
[13]	**GPC_MFP13**	**PC.13 Pin Function Selection** Both GPC_MFP[13] and ALT_MFP[21] are needed to set 0 for GPIOC[13] function on PC.13.
[12]	**GPC_MFP12**	**PC.12 Pin Function Selection** Both GPC_MFP[12] and ALT_MFP[13] are needed to set 0 for GPIOC[12] function on PC.12.
[11]	**GPC_MFP11**	**PC.11 Pin Function Selection** The pin function depends on GPC_MFP[11] and ALT_MFP[19]. ALT_MFP[19] / GPC_MFP[11] / PC.11 function: x / 0 / **GPIO** 0 / 1 / **MOSI10** (SPI1) 1 / 1 / **Reserved**
[10]	**GPC_MFP10**	**PC.10 Pin Function Selection** The pin function depends on GPC_MFP[10] and ALT_MFP[18]. ALT_MFP[18] / GPC_MFP[10] / PC.10 function: x / 0 / **GPIO**

		0 | 1 | **MISO10** (SPI1) 1 | 1 | **Reserved**
[9]	**GPC_MFP9**	**PC.9 Pin Function Selection** The pin function depends on GPC_MFP[9] and ALT_MFP[17]. ALT_MFP[17] | GPC_MFP[9] | PC.9 function x | 0 | **GPIO** 0 | 1 | **SPICLK1** (SPI1) 1 | 1 | **Reserved**
[8]	**GPC_MFP8**	**PC.8 Pin Function Selection** The pin function depends on GPC_MFP[8] and ALT_MFP[16]. ALT_MFP[16] | GPC_MFP[8] | PC.8 function x | 0 | **GPIO** 0 | 1 | **SPISS10** (SPI1) 1 | 1 | **Reserved**
[7:6]	**Reserved**	Reserved
[5]	**GPC_MFP5**	**PC.5 Pin Function Selection** GPC_MFP[5] is needed to set 0 for GPIOC[5] function on PC.5.
[4]	**GPC_MFP4**	**PC.4 Pin Function Selection** GPC_MFP[4] is needed to set 0 for GPIOC[4] function on PC.4.
[3]	**GPC_MFP3**	**PC.3 Pin Function Selection** ALT_MFP[8] and GPC_MFP[3] determine the PC.3 function. ALT_MFP[8] | GPC_MFP[3] | PC.3 function x | 0 | **GPIO** 0 | 1 | **MOSI00** (SPI0) 1 | 1 | **Reserved**
[2]	**GPC_MFP2**	**PC.2 Pin Function Selection** ALT_MFP[7] and GPC_MFP[2] determine the PC.2 function. ALT_MFP[7] | GPC_MFP[2] | PC.2 function x | 0 | **GPIO** 0 | 1 | **MISO00** (SPI0) 1 | 1 | **Reserved**
[1]	**GPC_MFP1**	**PC.1 Pin Function Selection** ALT_MFP[6] and GPC_MFP[1] determine the PC.1 function.

		ALT_MFP[6]	GPC_MFP[1]	PC.1 function	
		x	0	**GPIO**	
		0	1	**SPICLK0** (SPI0)	
		1	1	**Reserved**	
[0]	**GPC_MFP0**	**PC.0 Pin Function Selection** ALT_MFP[5] and GPC_MFP[0] determine the PC.0 function.			
		ALT_MFP[5]	GPC_MFP[0]	PC.0 function	
		x	0	**GPIO**	
		0	1	**SPISS00** (SPI0)	
		1	1	**Reserved**	

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[15:8]							
23	22	21	20	19	18	17	16
GPD_TYPE[7:0]							
15	14	13	12	11	10	9	8
GPD_MFP[15:8]							
7	6	5	4	3	2	1	0
GPD_MFP[7:0]							

Bits	Descriptions	
[31:16]	GPD_TYPEn	1 = Enable GPIOD[15:0] I/O input Schmitt Trigger function 0 = Disable GPIOD[15:0] I/O input Schmitt Trigger function GPIOD[15:12], GPIOD[7:6] are reserved
[15:12]	Reserved	Reserved
[11]	GPD_MFP11	**PD.11 Pin Function Selection** Both GPD_MFP[11] and ALT_MFP[21] are needed to set 0 for GPIOD[11] function on PD.11
[10]	GPD_MFP10	**PD.10 Pin Function Selection** Both GPD_MFP[10] and ALT_MFP[20] are needed to set 0 for GPIOD[10] function on PD.10
[9]	GPD_MFP9	**PD.9 Pin Function Selection** Both GPD_MFP[9] and ALT_MFP[19] are needed to set 0 for GPIOD[9] function on PD.9
[8]	GPD_MFP8	**PD.8 Pin Function Selection** Both GPD_MFP[8] and ALT_MFP[18] are needed to set 0 for GPIOD[8] function on PD.8
[7:6]	Reserved	Reserved
[5]	GPD_MFP5	**PD.5 Pin Function Selection** GPD_MFP[5] is needed to set 0 for GPIOD[5] function on PD.5
[4]	GPD_MFP4	**PD.4 Pin Function Selection** GPD_MFP[4] is needed to set 0 for GPIOD[4] function on PD.4
[3]	GPD_MFP3	**PD.3 Pin Function Selection**

		GPD_MFP[3] is needed to set 0 for GPIOD[3] function on PD.3
[2]	**GPD_MFP2**	**PD.2 Pin Function Selection** GPD_MFP[2] is needed to set 0 for GPIOD[2] function on PD.2
[1]	**GPD_MFP1**	**PD.1 Pin Function Selection** 1 = The SPI0 **SS01** function is selected to the pin PD.1(the validity of this function is depended on part no) 0 = The GPIOD[1] is selected to the pin PD.1
[0]	**GPD_MFP0**	**PD.0 Pin Function Selection** GPD_MFP[0] is needed to set 0 for GPIOD[0] function on PD.0

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							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**ALT_MFP[21:16]**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ALT_MFP[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ALT_MFP[7:2]						**PB9_S11**	**PB10_S01**

Bits	Descriptions	
[31:22]	**Reserved**	They are necessary to set 0
[21:16]	**ALT_MFP[21:16]**	They are necessary to set 0
[15]	**ALT_MFP[15]**	The PB.4 pin function depends on GPB_MFP4 and ALT_MFP[15]. ALT_MFP[15] / GPB_MFP4 / PB.4 function x / 0 / **GPIO** 0 / 1 / **UART1 RX** 1 / 1 / **SPISS11** (SPI1) Note: For QFN33 only.
[14:2]	**ALT_MFP[14:2]**	They are necessary to set 0

<table>
<tr><td>[1]</td><td>PB9_S11</td><td>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>x</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>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>x</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>

Register Write-Protection Control 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 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**
REGWRPROT[7:1]							**REGWRPROT [0]** **REGPROTDIS**

Bits	Descriptions	
[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) 1 = Write-protection is disabled for writing protected registers 0 = Write-protection is enabled for writing protected registers. Any write to the protected register is ignored. The Protected registers are:

		IPRSTC1: address 0x5000_0008 **BODCR**: address 0x5000_0018 **PORCR**: address 0x5000_0024 **PWRCON**: address 0x5000_0200 (bit[6] is not protected for power wake-up interrupt clear) **APBCLK bit[0]**: address 0x5000_0208 (bit[0] is Watchdog Timer clock enable) **CLK_SEL0**: address 0x5000_0210 (for HCLK and CPU STCLK clock source select) **CLK_SEL1 bit[1:0]**: address 0x5000_0214 (for Watchdog Timer clock source select) **NMI_SEL bit[7]**: address 0x5000_0380 (for interrupt test mode) **ISPCON**: address 0x5000_C000 (Flash ISP Control register) **WTCR**: address 0x4000_4000 **FATCON**: address 0x5000_C018

5.2.6 System Timer (SysTick)

The Cortex®-M0 includes an integrated system timer, SysTick. SysTick provides a simple, 24-bit

clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used in several different ways, for example:

- An RTOS tick timer which fires at a programmable rate (for example 100 Hz) and invokes a SysTick routine.
- A high speed alarm timer using Core clock.
- A variable rate alarm or signal timer – the duration range dependent on the reference clock used and the dynamic range of the counter.
- A simple counter. Software can use this to measure time to completion and time used.
- An internal clock source control based on missing/meeting durations. The COUNTFLAG bit-field in the control and status register can be used to determine if an action completed within a set duration, as part of a dynamic clock management control loop.

When enabled, the timer will count down from the value in the SysTick Current Value Register (SYST_CVR) to zero, 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 zero, 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 zero 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 zero, the timer will be maintained with a current value of zero after it is reloaded with this value. This mechanism can be used to disable the feature independently from the timer enable bit.

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

5.2.6.1 System Timer Control Register Map

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

Register	Offset	R/W	Description	Reset Value
SCS_BA = 0xE000_E000				
SYST_CSR	SCS_BA+0x10	R/W	SysTick Control and Status Register	0x0000_0000
SYST_RVR	SCS_BA+0x14	R/W	SysTick Reload Value Register	0xXXXX_XXXX
SYST_CVR	SCS_BA+0x18	R/W	SysTick Current Value Register	0xXXXX_XXXX

5.2.6.2 System Timer Control Register Description

SysTick Control and Status （SYST_CSR）

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

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

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

SysTick Reload Value Register （SYST_RVR）

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

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

Bits	Descriptions	
[31:24]	Reserved	Reserved
[23:0]	RELOAD	Value to load into the Current Value register when the counter reaches 0.

SysTick Current Value Register （SYST_CVR）

Register	Offset	R/W	Description	Reset Value
SYST_CVR	SCS _BA+0x18	R/W	SysTick Current Value Register	0xXXXX_XXXX

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

Bits	Descriptions	
[31:24]	**Reserved**	Reserved
[23:0]	**CURRENT**	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.

5.2.7 Nested Vectored Interrupt Controller (NVIC)

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

- Nested and Vectored interrupt support
- Automatic processor state saving and restoration
- Dynamic priority changing
- 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 any interrupts 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 documents "ARM® Cortex®-M0 Technical Reference Manual" and "ARM® v6-M Architecture Reference Manual".

5.2.7.1 Exception Model and System Interrupt Map

The following table lists the exception model supported by NuMicro™ NUC122 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 5-2 Exception Model

Vector Number	Interrupt Number (Bit in Interrupt Registers)	Interrupt Name	Source IP	Interrupt description
0 ~ 15	-	-	-	System exceptions
16	0	**BOD_OUT**	Brownout	Brownout 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	**GPCD_INT**	GPIO	External interrupt from PC[15:0]/PD[15:0]
22	6	**PWMA_INT**	PWM0~3	PWM0, PWM1, PWM2 and PWM3 interrupt
23	7	**Reserved**	Reserved	Reserved
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	**UART0_INT**	UART0	UART0 interrupt

29	13	**UART1_INT**	UART1	UART1 interrupt
30	14	**SPI0_INT**	SPI0	SPI0 interrupt
31	15	**SPI1_INT**	SPI1	SPI1 interrupt
32	16	**Reserved**	Reserved	Reserved
33	17	**Reserved**	Reserved	Reserved
34	18	**Reserved**	Reserved	Reserved
35	19	**I2C1_INT**	I^2C1	I^2C1 interrupt
36	20	**Reserved**	Reserved	Reserved
37	21	**Reserved**	Reserved	Reserved
38	22	**Reserved**	Reserved	Reserved
39	23	**USB_INT**	USBD	USB 2.0 FS Device interrupt
40	24	**PS2_INT**	PS/2	PS/2 interrupt
41	25	**Reserved**	Reserved	Reserved
42	26	**Reserved**	Reserved	Reserved
43	27	**Reserved**	Reserved	Reserved
44	28	**PWRWU_INT**	CLKC	Power Down Wake-up interrupt
45	29	**Reserved**	Reserved	Reserved
46	30	**Reserved**	Reserved	Reserved
47	31	**RTC_INT**	RTC	Real time clock interrupt

Table 5-3 System Interrupt Map

5.2.7.2 Vector Table

When any interrupts is accepted, the processor will automatically fetch the starting address of the interrupt service routine (ISR) from a vector table in memory. For ARM® v6-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 5-4 Vector Table Format

5.2.7.3 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.

5.2.7.4 NVIC Control Registers

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

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

IRQ0 ~ IRQ31 Set-Enable Control Register （NVIC_ISER）

Register	Offset	R/W	Description	Reset Value
NVIC_ISER	SCS _BA+0x100	R/W	IRQ0 ~ IRQ31 Set-Enable Control Register	0x0000_0000

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

Bits	Descriptions	
[31:0]	SETENA	Enable one or more interrupts within a group of 32. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Writing 1 will enable the associated interrupt. Writing 0 has 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	SCS _BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000

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

Bits	Descriptions	
[31:0]	**CLRENA**	Disable one or more interrupts within a group of 32. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Writing 1 will disable the associated interrupt. Writing 0 has no effect. 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	SCS _BA+0x200	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000

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

Bits	Descriptions	
[31:0]	SETPEND	Writing 1 to a bit to set pending state of the associated interrupt under software control. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Writing 0 has no effect. 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	SCS _BA+0x280	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000

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

Bits	Descriptions	
[31:0]	CLRPEND	Writing 1 to a bit to remove the pending state of associated interrupt under software control. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Writing 0 has no effect. The register reads back with the current pending state.

IRQ0 ~ IRQ3 Interrupt Priority Register （NVIC_IPR0）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR0	SCS _BA+0x400	R/W	IRQ0 ~ IRQ3 Interrupt 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	Descriptions	
[31:30]	**PRI_3**	**Priority of IRQ3** “0” denotes the highest priority and “3” denotes lowest priority
[29:24]	**Reserved**	Reserved
[23:22]	**PRI_2**	**Priority of IRQ2** “0” denotes the highest priority and “3” denotes lowest priority
[21:16]	**Reserved**	Reserved
[15:14]	**PRI_1**	**Priority of IRQ1** “0” denotes the highest priority and “3” denotes lowest priority
[13:8]	**Reserved**	Reserved
[7:6]	**PRI_0**	**Priority of IRQ0** “0” denotes the highest priority and “3” denotes lowest priority
[5:0]	**Reserved**	Reserved

IRQ4 ~ IRQ7 Interrupt Priority Register （NVIC_IPR1）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR1	SCS _BA+0x404	R/W	IRQ4 ~ IRQ7 Interrupt 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	Descriptions	
[31:30]	**PRI_7**	**Priority of IRQ7** “0” denotes the highest priority and “3” denotes lowest priority
[29:24]	**Reserved**	Reserved
[23:22]	**PRI_6**	**Priority of IRQ6** “0” denotes the highest priority and “3” denotes lowest priority
[21:16]	**Reserved**	Reserved
[15:14]	**PRI_5**	**Priority of IRQ5** “0” denotes the highest priority and “3” denotes lowest priority
[13:8]	**Reserved**	Reserved
[7:6]	**PRI_4**	**Priority of IRQ4** “0” denotes the highest priority and “3” denotes lowest priority
[5:0]	**Reserved**	Reserved

IRQ8 ~ IRQ11 Interrupt Priority Register （NVIC_IPR2）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR2	SCS _BA+0x408	R/W	IRQ8 ~ IRQ11 Interrupt 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	Descriptions	
[31:30]	**PRI_11**	**Priority of IRQ11** “0” denotes the highest priority and “3” denotes lowest priority
[29:24]	**Reserved**	Reserved
[23:22]	**PRI_10**	**Priority of IRQ10** “0” denotes the highest priority and “3” denotes lowest priority
[21:16]	**Reserved**	Reserved
[15:14]	**PRI_9**	**Priority of IRQ9** “0” denotes the highest priority and “3” denotes lowest priority
[13:8]	**Reserved**	Reserved
[7:6]	**PRI_8**	**Priority of IRQ8** “0” denotes the highest priority and “3” denotes lowest priority
[5:0]	**Reserved**	Reserved

IRQ12 ~ IRQ15 Interrupt Priority Register （NVIC_IPR3）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR3	SCS _BA+0x40C	R/W	IRQ12 ~ IRQ15 Interrupt 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	Descriptions	
[31:30]	**PRI_15**	**Priority of IRQ15** “0” denotes the highest priority and “3” denotes lowest priority
[29:24]	**Reserved**	Reserved
[23:22]	**PRI_14**	**Priority of IRQ14** “0” denotes the highest priority and “3” denotes lowest priority
[21:16]	**Reserved**	Reserved
[15:14]	**PRI_13**	**Priority of IRQ13** “0” denotes the highest priority and “3” denotes lowest priority
[13:8]	**Reserved**	Reserved
[7:6]	**PRI_12**	**Priority of IRQ12** “0” denotes the highest priority and “3” denotes lowest priority
[5:0]	**Reserved**	Reserved

IRQ16 ~ IRQ19 Interrupt Priority Register （NVIC_IPR4）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR4	SCS _BA+0x410	R/W	IRQ16 ~ IRQ19 Interrupt 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	Descriptions	
[31:30]	**PRI_19**	**Priority of IRQ19** “0” denotes the highest priority and “3” denotes lowest priority
[29:24]	**Reserved**	Reserved
[23:22]	**PRI_18**	**Priority of IRQ18** “0” denotes the highest priority and “3” denotes lowest priority
[21:16]	**Reserved**	Reserved
[15:14]	**PRI_17**	**Priority of IRQ17** “0” denotes the highest priority and “3” denotes lowest priority
[13:8]	**Reserved**	Reserved
[7:6]	**PRI_16**	**Priority of IRQ16** “0” denotes the highest priority and “3” denotes lowest priority
[5:0]	**Reserved**	Reserved

IRQ20 ~ IRQ23 Interrupt Priority Register （NVIC_IPR5）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR5	SCS _BA+0x414	R/W	IRQ20 ~ IRQ23 Interrupt 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	Descriptions	
[31:30]	**PRI_23**	**Priority of IRQ23** “0” denotes the highest priority and “3” denotes lowest priority
[29:24]	**Reserved**	Reserved
[23:22]	**PRI_22**	**Priority of IRQ22** “0” denotes the highest priority and “3” denotes lowest priority
[21:16]	**Reserved**	Reserved
[15:14]	**PRI_21**	**Priority of IRQ21** “0” denotes the highest priority and “3” denotes lowest priority
[13:8]	**Reserved**	Reserved
[7:6]	**PRI_20**	**Priority of IRQ20** “0” denotes the highest priority and “3” denotes lowest priority
[5:0]	**Reserved**	Reserved

IRQ24 ~ IRQ27 Interrupt Priority Register （NVIC_IPR6）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR6	SCS _BA+0x418	R/W	IRQ24 ~ IRQ27 Interrupt 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	Descriptions	
[31:30]	**PRI_27**	**Priority of IRQ27** “0” denotes the highest priority and “3” denotes lowest priority
[29:24]	**Reserved**	Reserved
[23:22]	**PRI_26**	**Priority of IRQ26** “0” denotes the highest priority and “3” denotes lowest priority
[21:16]	**Reserved**	Reserved
[15:14]	**PRI_25**	**Priority of IRQ25** “0” denotes the highest priority and “3” denotes lowest priority
[13:8]	**Reserved**	Reserved
[7:6]	**PRI_24**	**Priority of IRQ24** “0” denotes the highest priority and “3” denotes lowest priority
[5:0]	**Reserved**	Reserved

IRQ28 ~ IRQ31 Interrupt Priority Register （NVIC_IPR7）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR7	SCS _BA+0x41C	R/W	IRQ28 ~ IRQ31 Interrupt 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	Descriptions	
[31:30]	**PRI_31**	**Priority of IRQ31** “0” denotes the highest priority and “3” denotes lowest priority
[29:24]	**Reserved**	Reserved
[23:22]	**PRI_30**	**Priority of IRQ30** “0” denotes the highest priority and “3” denotes lowest priority
[21:16]	**Reserved**	Reserved
[15:14]	**PRI_29**	**Priority of IRQ29** “0” denotes the highest priority and “3” denotes lowest priority
[13:8]	**Reserved**	Reserved
[7:6]	**PRI_28**	**Priority of IRQ28** “0” denotes the highest priority and “3” denotes lowest priority
[5:0]	**Reserved**	Reserved

5.2.7.5 *Interrupt Source Control Registers*

Besides the interrupt control registers associated with the NVIC, NuMicro™ NUC122 Series also implement some specific control registers to facilitate the interrupt functions, including “interrupt source identification”, ”NMI source selection” and “interrupt test mode”. They are described as below.

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

Register	Offset	R/W	Description	Reset Value
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) 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 (Reserved) 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) 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 (Reserved) Interrupt Source Identity	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (Reserved) Interrupt Source Identity	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (Reserved) 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	IRQ20 (Reserved) Interrupt Source Identity	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (Reserved) Interrupt Source Identity	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (Reserved) 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 (Reserved) Interrupt Source Identity	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (Reserved) Interrupt Source Identity	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	IRQ27 (Reserved) 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 (Reserved) Interrupt Source Identity	0xXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	R	IRQ30 (Reserved) 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 IRQ Number Identity Register	0x0000_0000

Interrupt Source Identity Register (IRQn_SRC)

Register	Offset	R/W	Description	Reset Value
IRQn_SRC	INT_BA+0x00 …….. INT_BA+0x7C	R	IRQ0 (BOD) Interrupt Source Identity : 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[3]	INT_SRC[2:0]		

Bits	Address	INT-Num	Descriptions
[2:0]	INT_BA+0x00	0	Bit2: 0 Bit1: 0 Bit0: BOD_INT
[2:0]	INT_BA+0x04	1	Bit2: 0 Bit1: 0 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
[2:0]	INT_BA+0x14	5	Bit2: 0 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: 0 Bit2: 0 Bit1: 0 Bit0: 0
[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: 0 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: 0
[2:0]	INT_BA+0x44	17	Bit2: 0

			Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x48	18	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x4C	19	Bit2: 0 Bit1: 0 Bit0: I^2C1_INT
[2:0]	INT_BA+0x50	20	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x54	21	Reserved
[2:0]	INT_BA+0x58	22	Reserved
[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: 0
[2:0]	INT_BA+0x68	26	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x6C	27	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x70	28	Bit2: 0 Bit1: 0 Bit0: PWRWU_INT
[2:0]	INT_BA+0x74	29	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x78	30	Reserved
[2:0]	INT_BA+0x7C	31	Bit2: 0 Bit1: 0 Bit0: RTC_INT

NMI Interrupt Source Selection Control Register (NMI_SEL)

Register	Offset	R/W	Description	Reset Value
NMI_SEL	INT_BA+0x80	R/W	NMI Interrupt Source Selection 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**
INT_TEST	**Reserved**		**NMI_SEL[4:0]**				

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7]	**INT_TEST**	Interrupt Test Mode (write-protection bit)
[6:5]	**Reserved**	Reserved
[4:0]	**NMI_SEL**	NMI Interrupt Source Selection The NMI interrupt to Cortex®-M0 can be selected from one of the peripheral interrupts 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[31:24]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
MCU_IRQ[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MCU_IRQ[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MCU_IRQ[7:0]							

Bits	Descriptions	
[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 then 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), set 1 to the MCU_IRQ[n] will clear the interrupt and set MCU_IRQ[n] 0 : no any effect.

5.2.8 System Control Register

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

For more detailed information, please refer to the documents “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_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[7:0]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**PART[3:0]**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PARTNO[11:4]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PARTNO[3:0]				**REVISION[3:0]**			

Bits	Descriptions	
[31:24]	**IMPLEMENTER**	Implementer code assigned by ARM. (ARM = 0x41)
[23:20]	**Reserved**	Reserved
[19:16]	**PART**	Reads as 0xC for ARM® v6-M parts
[15:4]	**PARTNO**	Reads as 0xC20.
[3:0]	**REVISION**	Reads 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[5:4]	
15	14	13	12	11	10	9	8
VECTPENDING[3:0]				Reserved			
7	6	5	4	3	2	1	0
Reserved		VECTACTIVE[5:0]					

Bits	Descriptions	
[31]	**NMIPENDSET**	NMI set-pending bit Write: 0 = no effect 1 = changes NMI exception state to pending. Read: 0 = NMI exception is not pending 1 = NMI exception is pending. Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it detects a write of 1 to this bit. Entering the handler then clears this bit to 0. This means a read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler.
[30:29]	**Reserved**	Reserved
[28]	**PENDSVSET**	PendSV set-pending bit. Write: 0 = no effect 1 = changes PendSV exception state to pending. Read: 0 = PendSV exception is not pending 1 = PendSV exception is pending. 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. 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. 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 is a read only bit.
[22]	**ISRPENDING**	Interrupt pending flag, excluding NMI and Faults: 0 = interrupt not pending 1 = interrupt pending. This is a read only bit.
[21:18]	**Reserved**	Reserved
[17:12]	**VECTPENDING**	Indicates the exception number of the highest priority pending enabled exception: 0 = no pending exceptions Nonzero = the exception number of the highest priority pending enabled exception.
[11:6]	**Reserved**	Reserved
[5:0]	**VECTACTIVE**	Contains the active exception number 0 = Thread mode Nonzero = The exception number of the currently active exception.

Application Interrupt and Reset Control Register (AIRCR)

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

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

Bits	Descriptions	
[31:16]	**VECTORKEY**	When write this register, this field should be 0x05FA, otherwise the write action will be unpredictable.
[15:3]	**Reserved**	Reserved
[2]	**SYSRESETREQ**	Writing this bit 1 will cause a reset signal to be asserted to the chip to indicate a reset is requested. The bit is a write only bit and self-clears as part of the reset sequence.
[1]	**VECTCLRACTIVE**	Set this bit to 1 will clears 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	Descriptions	
[31:5]	**Reserved**	Reserved
[4]	**SEVONPEND**	Send Event on Pending bit: 0 = only enabled interrupts or events can wakeup the processor, disabled interrupts are excluded 1 = enabled events and all interrupts, including disabled interrupts, can wakeup 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	Descriptions	
[31:30]	**PRI_11**	**Priority of system handler 11 – SVCall** “0” denotes the highest priority and “3” denotes 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	Descriptions	
[31:30]	**PRI_15**	**Priority of system handler 15 – SysTick** "0" denotes the highest priority and "3" denotes lowest priority
[29:24]	**Reserved**	Reserved
[23:22]	**PRI_14**	**Priority of system handler 14 – PendSV** "0" denotes the highest priority and "3" denotes lowest priority
[21:0]	**Reserved**	Reserved

5.3 Clock Controller

5.3.1 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 will not enter power down mode until CPU sets the power down enable bit (PWR_DOWN_EN) and Cortex®-M0 core executes the WFI instruction. After that, chip enters power down mode and wait for wake-up interrupt source triggered to leave power down mode. In 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.

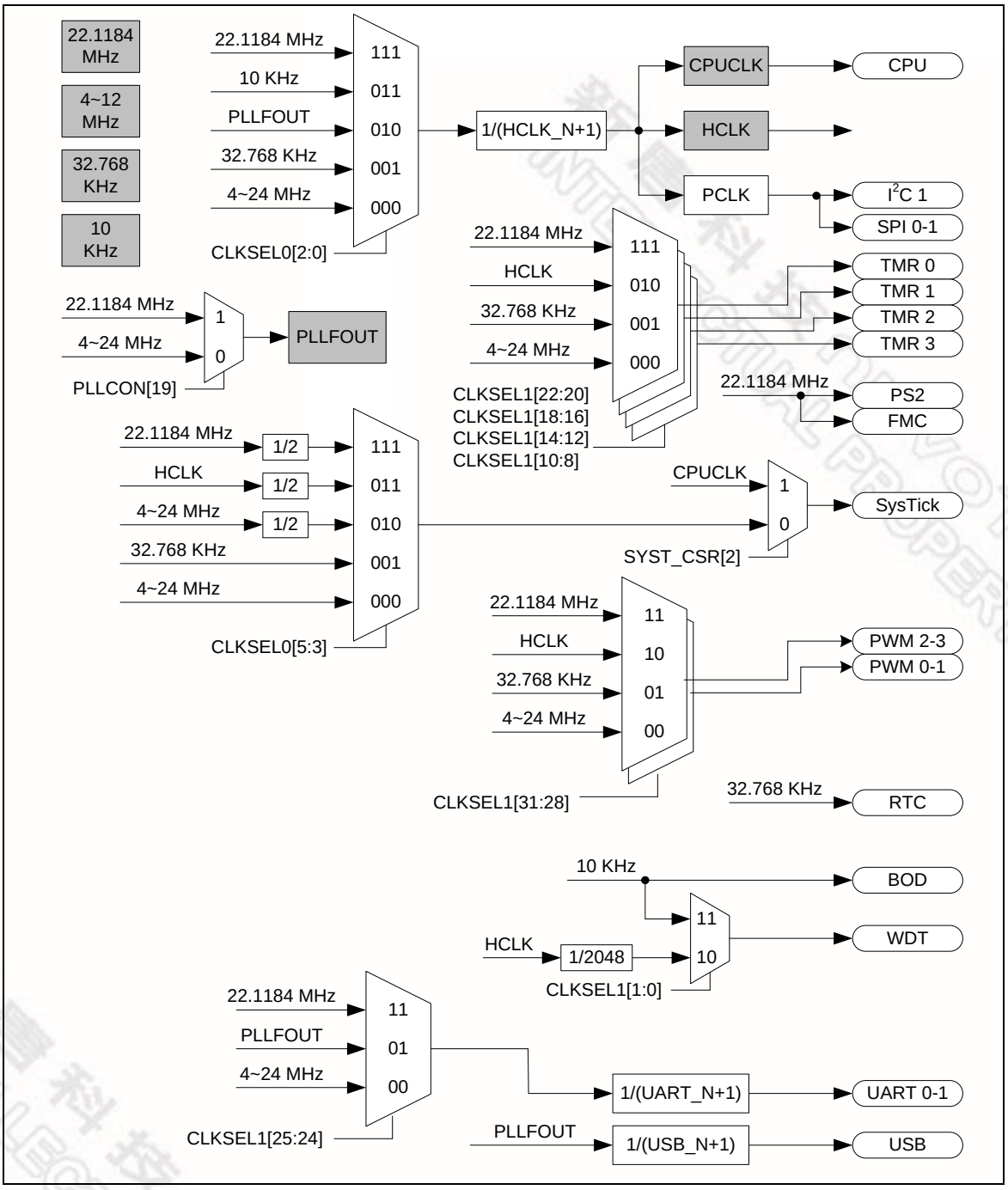


Figure 5-3 Clock Generator Global View Diagram

5.3.2 Clock Generator

The clock generator consists of 5 clock sources which are listed as 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

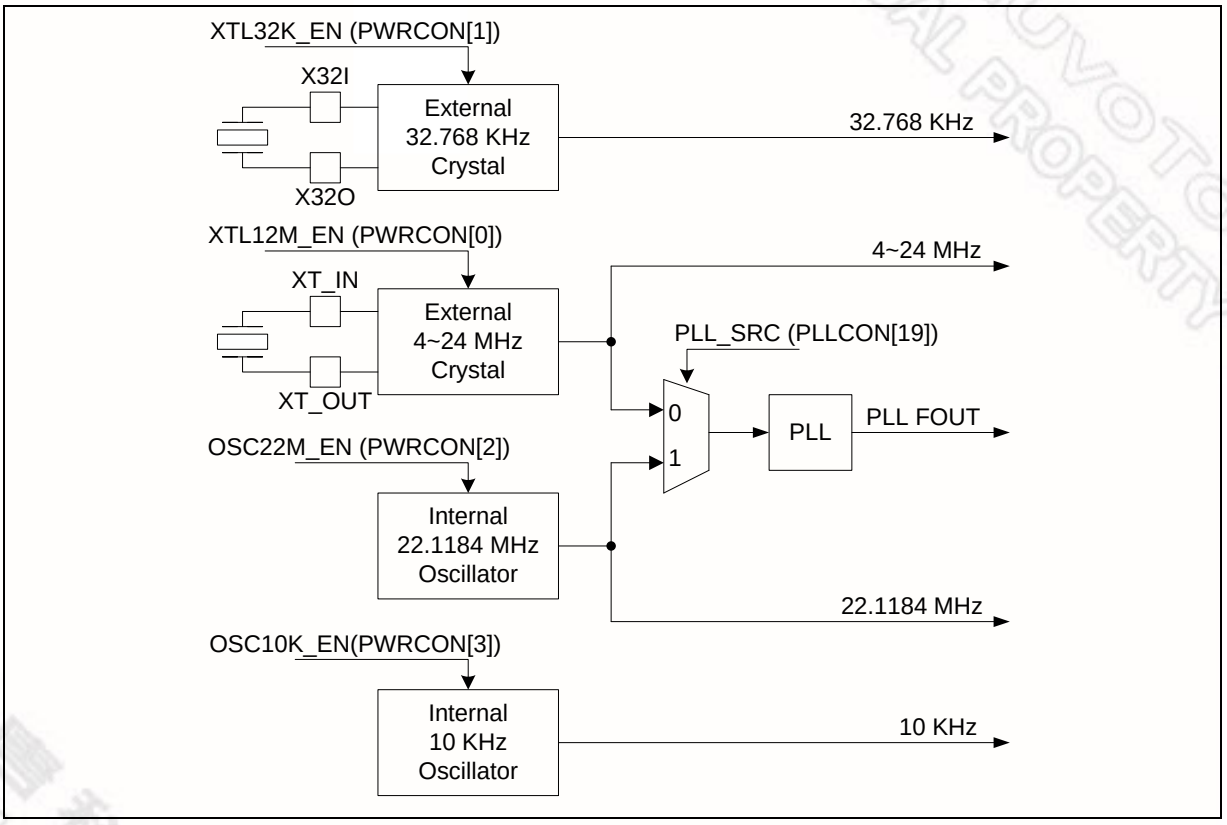


Figure 5-4 Clock Generator Block Diagram

5.3.3 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 listed below.

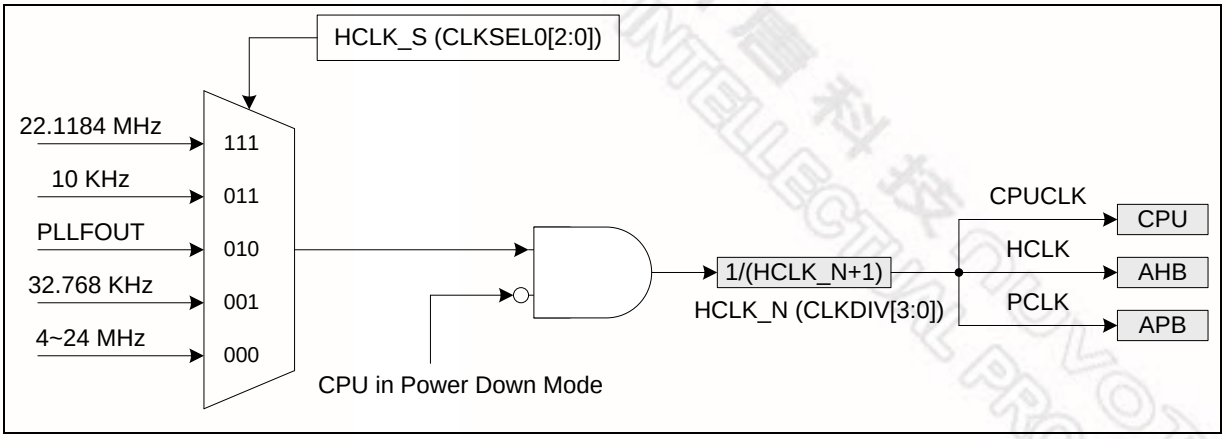


Figure 5-5 System Clock Block Diagram

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 listed below.

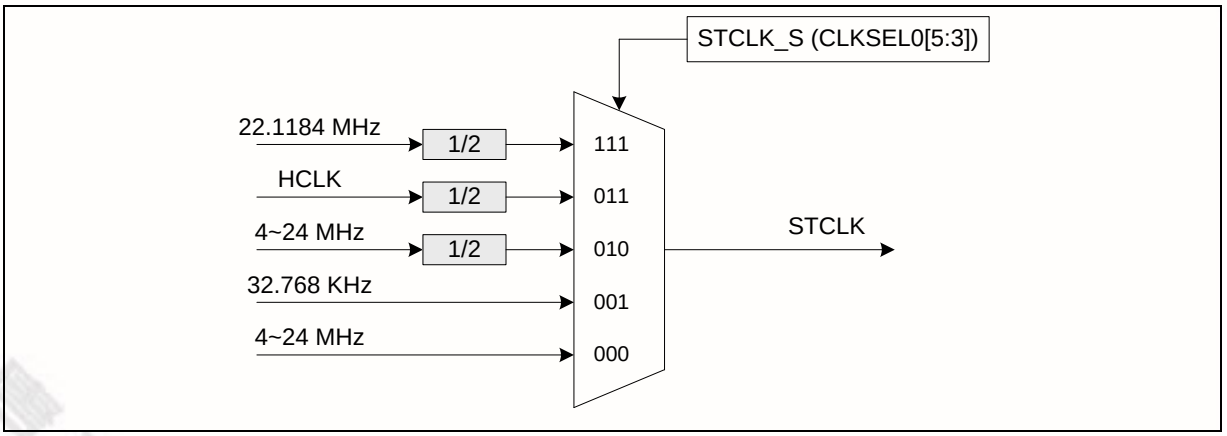


Figure 5-6 SysTick Clock Control Block Diagram

5.3.4 Peripherals Clock

The peripherals clock had different clock source switch setting which depends on the different peripheral. Please refer the CLKSEL1 register description in 5.3.7.

5.3.5 Power Down Mode Clock

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

These clocks which still keep activity that are listed as below:

- Clock Generator
  - Internal 10 KHz low speed oscillator clock
  - External 32.768 KHz low speed crystal clock
- Peripherals Clock (When WDT adopts 10 KHz low speed as clock source and RTC adopts 32.768 KHz low speed as clock source)

5.3.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
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_000D
APBCLK	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register	0x0000_000X
CLKSTATUS	CLK_BA+0x0C	R/W	Clock Status Monitor Register	0x0000_00XX
CLKSEL0	CLK_BA+0x10	R/W	Clock Source Selection Control Register 0	0x0000_003X
CLKSEL1	CLK_BA+0x14	R/W	Clock Source Selection Control Register 1	0xFFFF_FFFF
CLKDIV	CLK_BA+0x18	R/W	Clock Divider Register	0x0000_0000
PLLCON	CLK_BA+0x20	R/W	PLL Control Register	0x0005_C22E

5.3.7 Register Description

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. Reference 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	Descriptions	
[31:9]	**Reserved**	Reserve
[8]	**PD_WAIT_CPU**	**This Bit Control the Power Down Entry Condition** (write-protection bit) 1 = Chip enter power down mode when the both PD_WAIT_CPU and PWR_DOWN_EN bits are set to 1 and CPU run WFI instruction. 0 = Chip entry power down mode when the PWR_DOWN_EN bit is set to 1
[7]	**PWR_DOWN_EN**	**System Power Down Enable** (write-protection bit) When this bit is set to 1, the chip 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 When chip be woken-up from power down mode, this bit is auto cleared. Users need to set this bit again for next power down. When in power down mode, external 4~24 MHz high speed crystal and the 22.1184 MHz high speed OSC will be disabled in this mode, but the 32.768 KHz low speed crystal and 10 KHz low speed OSC is not controlled by power down mode. When in power down mode, the PLL and system clock are disabled, and ignored the clock source selection. The clocks of peripheral are not controlled by power down mode, if the peripheral clock source is from 32.768 KHz low speed crystal or the 10 KHz low

		speed oscillator. 1 = Chip enters the power down mode instant or wait CPU sleep command WFI 0 = Chip is operating normally or chip is in idle mode because of WFI command
[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, BOD or RTC wake-up occurred Write 1 to clear the bit to zero. Note: This bit is effective only if PD_WU_INT_EN (PWRCON[5]) be set to 1.
[5]	**PD_WU_INT_EN**	**Power Down Mode Wake-Up Interrupt Enable** (write-protection bit) 0 = Disable 1 = Enable The interrupt will occur when both PD_WU_STS and PD_WU_INT_EN are high.
[4]	**PD_WU_DLY**	**Wake-Up Delay Counter Enable** (write-protection bit) When the chip be woken-up from power down mode, the clock control will delay certain clock cycles to wait system clock is 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 22.1184 MHz high speed oscillator. 1 = Enable clock cycles delay 0 = Disable clock cycles delay
[3]	**OSC10K_EN**	**Internal 10 KHz Low Speed Oscillator Enable** (write-protection bit) 1 = Enable 10 KHz low speed oscillator 0 = Disable 10 KHz low speed oscillator
[2]	**OSC22M_EN**	**Internal 22.1184 MHz High Speed Oscillator Enable** (write-protection bit) 1 = Enable 22.1184 MHz high speed oscillator 0 = Disable 22.1184 MHz high speed oscillator
[1]	**XTL32K_EN**	**External 32.768 KHz Low Speed Crystal Enable** (write-protection bit) 1 = Enable 32.768 KHz low speed crystal (Normal operation) 0 = Disable 32.768 KHz low speed crystal
[0]	**XTL12M_EN**	**External 4~24 MHz High Speed Crystal Enable** (write-protection bit) 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 1 = Enable external 4~24 MHz high speed crystal 0 = Disable external 4~24 MHz high speed crystal

Register/Instruction / Mode	PWR_DOWN_EN	PD_WAIT_CPU	CPU run WFI instruction	Clock Disable
Normal operation	0	0	NO	All clocks are disabled by control register
Idle mode (CPU in sleep mode)	0	0	YES	Only CPU clock is disabled
Power down mode	1	0	NO	Most clocks are disabled except 10 KHz/32.768 KHz, only RTC/WDT peripheral clocks are still enabled.
Power down mode (CPU in deep sleep mode)	1	1	YES	Most clocks are disabled except 10 KHz/32.768 KHz, only RTC/WDT peripheral clocks are still enabled.

Table 5-5 Power Down Mode Control Table

When chip enter power down mode, user can wake-up chip by some interrupt sources. User should enable related interrupt sources and NVIC IRQ enable bits (NVIC_ISER) before setting PWR_DOWN_EN bit in PWRCON[7] to ensure chip can enter power down and be woken-up successfully.

AHB Devices Clock Enable Control Register （AHBCLK）

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

Register	Offset	R/W	Description	Reset Value
AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_000D

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

Bits	Descriptions	
[31:3]	**Reserved**	Reserved
[2]	**ISP_EN**	**Flash ISP Controller Clock Enable Control** 1 = Enable the Flash ISP engine clock 0 = Disable the Flash ISP engine clock
[1:0]	**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	Reserved	Reserved	Reserved	USBD_EN	Reserved		Reserved
23	22	21	20	19	18	17	16
Reserved	Reserved	PWM23_EN	PWM01_EN	Reserved	Reserved	UART1_EN	UART0_EN
15	14	13	12	11	10	9	8
Reserved	Reserved	SPI1_EN	SPI0_EN	Reserved		I2C1_EN	Reserved
7	6	5	4	3	2	1	0
Reserved	Reserved	TMR3_EN	TMR2_EN	TMR1_EN	TMR0_EN	RTC_EN	WDT_EN

Bits	Descriptions	
[31]	PS2_EN	**PS2 Clock Enable** 1 = Enable PS/2 clock 0 = Disable PS/2 clock
[30:28]	Reserved	Reserved
[27]	USBD_EN	**USB 2.0 FS Device Controller Clock Enable** 1 = Enable USB clock 0 = Disable USB clock
[26:22]	Reserved	Reserved
[21]	PWM23_EN	**PWM_23 Clock Enable** 1 = Enable PWM23 clock 0 = Disable PWM23 clock
[20]	PWM01_EN	**PWM_01 Clock Enable** 1 = Enable PWM01 clock 0 = Disable PWM01 clock
[19:18]	Reserved	Reserved
[17]	UART1_EN	**UART1 Clock Enable** 1 = Enable UART1 clock 0 = Disable UART1 clock
[16]	UART0_EN	**UART0 Clock Enable** 1 = Enable UART0 clock

		0 = Disable UART0 clock
[15:14]	**Reserved**	Reserved
[13]	**SPI1_EN**	**SPI1 Clock Enable** 1 = Enable SPI1 clock 0 = Disable SPI1 clock
[12]	**SPI0_EN**	**SPI0 Clock Enable** 1 = Enable SPI0 clock 0 = Disable SPI0 clock
[11:10]	**Reserved**	Reserved
[9]	**I2C1_EN**	**I^2C1 Clock Enable** 1 = Enable I^2C1 clock 0 = Disable I^2C1 clock
[8:6]	**Reserved**	Reserved
[5]	**TMR3_EN**	**Timer3 Clock Enable** 1 = Enable Timer3 clock 0 = Disable Timer3 clock
[4]	**TMR2_EN**	**Timer2 Clock Enable** 1 = Enable Timer2 clock 0 = Disable Timer2 clock
[3]	**TMR1_EN**	**Timer1 Clock Enable** 1 = Enable Timer1 clock 0 = Disable Timer1 clock
[2]	**TMR0_EN**	**Timer0 Clock Enable** 1 = Enable Timer0 clock 0 = Disable Timer0 clock
[1]	**RTC_EN**	**Real-Time-Clock APB interface Clock Enable** This bit is used to control the RTC APB clock only, The RTC engine clock source is from the 32768 Hz crystal. 1 = Enable RTC clock 0 = Disable RTC clock
[0]	**WDT_EN**	**Watchdog Timer Clock Enable** (write-protection bit) 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. 1 = Enable Watchdog Timer clock 0 = Disable Watchdog Timer clock

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	Descriptions	
[31:8]	**Reserved**	Reserved
[7]	**CLK_SW_FAIL**	**Clock Switching Fail Flag** (write-protection bit) 1 = Clock switching failure 0 = Clock switching success 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. Write 1 to clear the bit to zero.
[6:5]	**Reserved**	Reserved
[4]	**OSC22M_STB**	**Internal 22.1184 MHz High Speed Oscillator Clock Source Stable Flag** 1 = Internal 22.1184 MHz high speed oscillator clock is stable 0 = Internal 22.1184 MHz high speed oscillator clock is not stable or disabled This is read only bit
[3]	**OSC10K_STB**	**Internal 10 KHz Low Speed Oscillator Clock Source Stable Flag** 1 = Internal 10 KHz low speed oscillator clock is stable 0 = Internal 10 KHz low speed oscillator clock is not stable or disabled This is read only bit
[2]	**PLL_STB**	**Internal PLL Clock Source Stable Flag** 1 = Internal PLL clock is stable

		0 = Internal PLL clock is not stable or disabled This is read only bit
[1]	**XTL32K_STB**	**External 32.768 KHz Low Speed Crystal Clock Source Stable Flag** 1 = External 32.768 KHz low speed crystal clock is stable 0 = External 32.768 KHz low speed crystal clock is not stable or disabled This is read only bit
[0]	**XTL12M_STB**	**External 4~24 MHz High Speed Crystal Clock Source Stable Flag** 1 = External 4~24 MHz high speed crystal clock is stable 0 = External 4~24 MHz high speed crystal clock is not stable or disabled This is read only bit

Clock Source Selection Control Register 0 （CLKSEL0）

Register	Offset	R/W	Description	Reset Value
CLKSEL0	CLK_BA+0x10	R/W	Clock Source Selection 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	Descriptions	
[31:6]	**Reserved**	Reserved
[5:3]	**STCLK_S**	**Cortex®-M0 SysTick Clock Source Selection** (write-protection bits) If SYST_CSR[2]=0, SysTick uses listed clock source below 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. 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 Others = reserved
[2:0]	**HCLK_S**	**HCLK Clock Source Selection** (write-protection bits) Note: 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. Reference the register REGWRPROT at address GCR_BA+0x100. 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 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 Others = reserved

Clock Source Selection 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 Selection Control Register 1	0xFFFF_FFFF

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

Bits	Descriptions	
[31:30]	**PWM23_S**	**PWM2 and PWM3 Clock Source Selection** PWM2 and PWM3 uses the same Engine clock source, both of them use the same prescaler 00 = Clock source from external 4~24 MHz high speed crystal clock 01 = Clock source from external 32.768 KHz low speed crystal clock 10 = Clock source from HCLK 11 = Clock source from internal 22.1184 MHz high speed oscillator clock
[29:28]	**PWM01_S**	**PWM0 and PWM1 Clock Source Selection** PWM0 and PWM1 uses the same Engine clock source, both of them use the same prescaler 00 = Clock source from external 4~24 MHz high speed crystal clock 01 = Clock source from external 32.768 KHz low speed crystal clock 10 = Clock source from HCLK 11 = Clock source from internal 22.1184 MHz high 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 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 = Reserved 111 = Clock source from internal 22.1184 MHz high speed oscillator clock
[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 = Reserved 111 = Clock source from internal 22.1184 MHz high speed oscillator clock
[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 = Reserved 111 = Clock source from internal 22.1184 MHz high speed oscillator clock
[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 = Reserved 111 = Clock source from internal 22.1184 MHz high speed oscillator clock
[7:2]	**Reserved**	Reserved
[1:0]	**WDT_S**	**Watchdog Timer Clock Source Selection** (write-protection bits) These bits are protected-bit, program this need to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Reference the register REGWRPROT at address GCR_BA+0x100. 00 = Reserved 01 = Reserved 10 = Clock source from HCLK/2048 clock 11 = Clock source from internal 10 KHz low speed oscillator clock

Clock Divider Register (CLKDIV)

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

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

Bits	Descriptions	
[31:12]	**Reserved**	Reserved
[11:8]	**UART_N**	**UART Clock Divider from UART Clock Source** The UART clock frequency = (UART clock source frequency) / (UART_N + 1)
[7:4]	**USB_N**	**USB Clock Divider from PLL Clock Source** The USB clock frequency = (PLL frequency) / (USB_N + 1)
[3:0]	**HCLK_N**	**HCLK Clock Divider from HCLK Clock Source** The HCLK clock frequency = (HCLK clock source frequency) / (HCLK_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 use 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	Descriptions	
[31:20]	**Reserved**	Reserved
[19]	**PLL_SRC**	**PLL Source Clock Selection** 1 = PLL source clock from internal 22.1184 MHz high speed oscillator 0 = PLL source clock from external 4~24 MHz high speed crystal
[18]	**OE**	**PLL OE (FOUT enable)** 0 = PLL FOUT enable 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 clock input (XTALin)
[16]	**PD**	**Power Down Mode** If set the PWR_DOWN_EN bit 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** Refer to the formulas below the table.
[13:9]	**IN_DV**	**PLL Input Divider**

		Refer to the formulas below the table.
[8:0]	**FB_DV**	**PLL Feedback Divider** Refer to the formulas below the table.

Output Clock Frequency Setting

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

Constrain:

1. $3.2MHz < FIN < 150MHz$
2. $800KHz < \frac{FIN}{2 * NR} < 7.5MHz$
3. $100MHz < FCO = FIN \times \frac{NF}{NR} < 500MHz$

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

48 MHz	50 MHz	60 MHz
0xC22E	0xC230	0xC23A

5.4 FLASH MEMORY CONTROLLER (FMC)

5.4.1 Overview

NuMicro™ NUC122 Series equips with 64/32K bytes on-chip embedded Flash for application program memory (APROM) that can be updated through ISP procedure. In System Programming (ISP) function enables user to update program memory when chip is soldered on PCB. After chip power on, Cortex™-M0 CPU fetches code from APROM or LDROM decided by boot select (CBS) in Config0. By the way, NuMicro™ NUC122 Series also provides additional DATA Flash for user to store some application dependent data before chip power off. For 64K/32K bytes APROM device, the data flash is fixed at 4K bytes.

5.4.2 Features

- Run up to 60 MHz with zero wait state for continuous address read access
- 64/32KB application program memory (APROM)
- 4KB in system programming (ISP) loader program memory (LDROM)
- Fixed 4KB data flash with 512 bytes page erase unit
- In System Program (ISP) to update on chip Flash

5.4.3 Block Diagram

The flash memory controller consist 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 following:

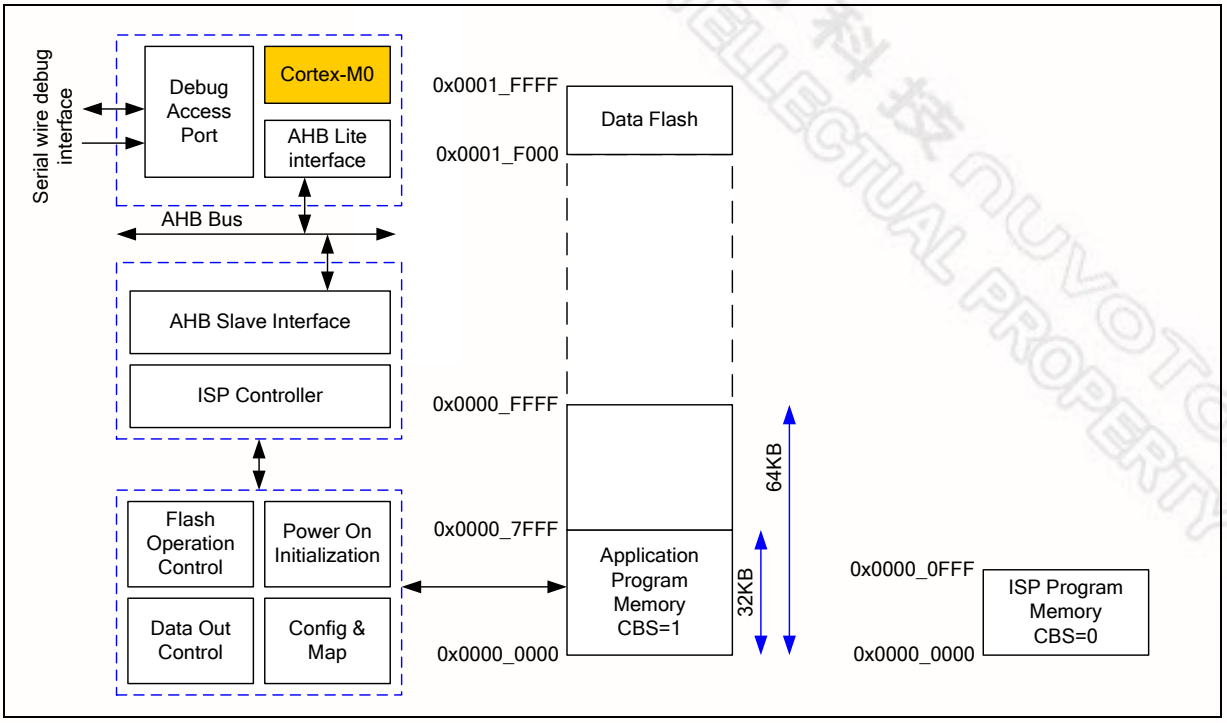


Figure 5-7 Flash Memory Control Block Diagram

5.4.4 Flash Memory Organization

NuMicro™ NUC122 Series flash memory consists of Program memory (64/32 KB), data flash, ISP loader program memory, user configuration. User configuration block provides several bytes to control system logic, like flash security lock, boot select, brownout voltage level, and so on. It works like a fuse for power on setting. It is loaded from flash memory to its corresponding control registers during chip power on. User can set these bits according to application request by writer before chip is mounted on PCB. For 64/32 KB APROM devices, its size of data flash is 4 KB and start address is fixed at 0x0001_F000.

Table 5-1 Memory Address Map

Block Name	Size	Start Address	End Address
AP-ROM	32/64 KB	0x0000_0000	0x0000_7FFF (32 KB) 0x0000_FFFF (64 KB)
Reserved for future use	960 KB	0x0001_0000	0x000F_FFFF
Data Flash	4 KB	0x0001_F000	0x0001_FFFF
LD-ROM	4 KB	0x0010_0000	0x0010_0FFF
User Configuration	1 word	0x0030_0000	0x0030_0000

The Flash memory organization is shown as below:

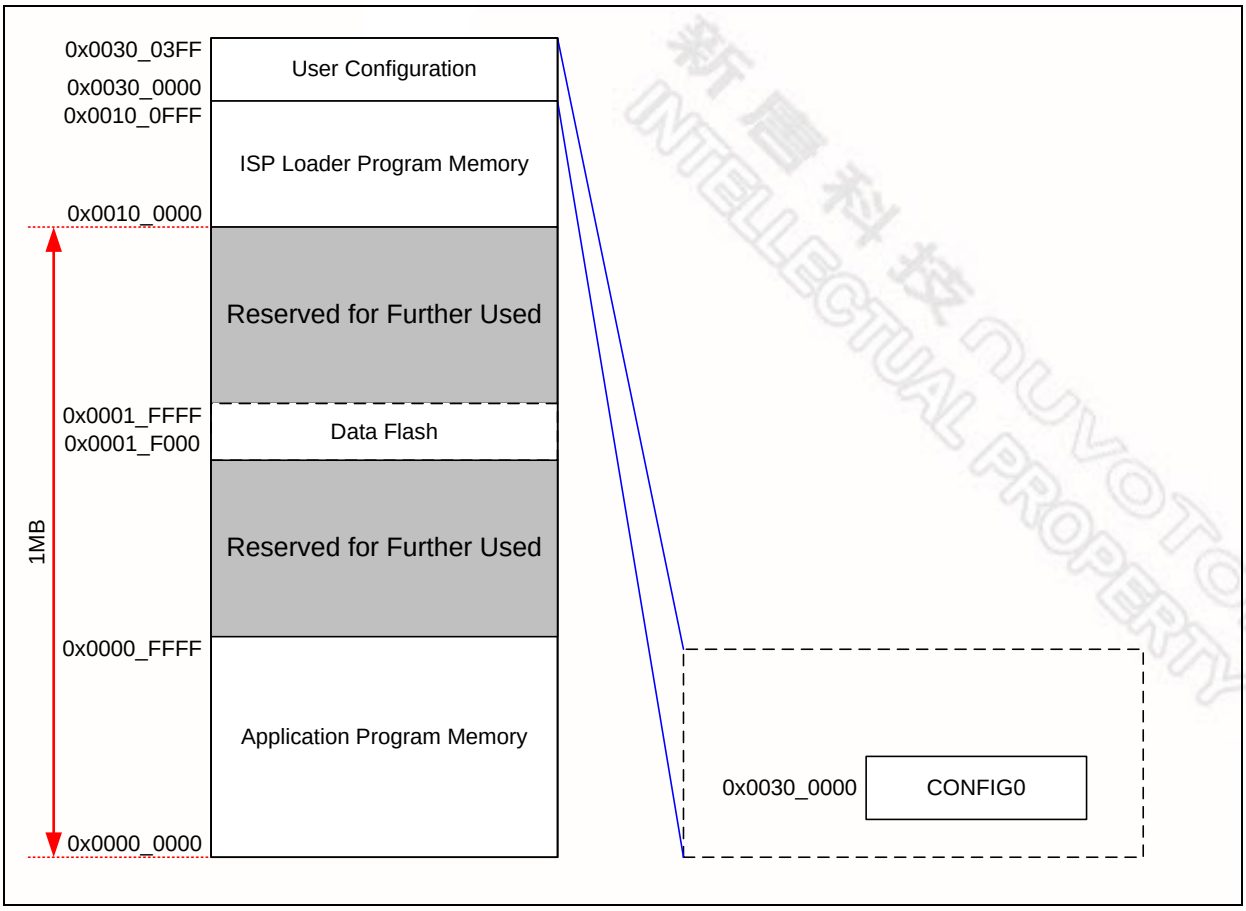


Figure 5-8 Flash Memory Organization

5.4.5 Boot Selection

NuMicro™ NUC122 Series provides in system programming (ISP) feature to enable user to update program memory when chip is mounted on PCB. A dedicated 4 KB program memory is used to store ISP firmware. Users can select to start program fetch from APROM or LDROM by (CBS) in Config0.

5.4.6 Data Flash

NuMicro™ NUC122 Series provides data flash for user to store data. It is read/write through ISP procedure. The size of each erase unit is 512 bytes. When a word will be changed, all 128 words need to be copied to another page or SRAM in advance. For 64/32 KB APROM devices, the size of data flash is 4 KB and start address is fixed at 0x0001_F000.

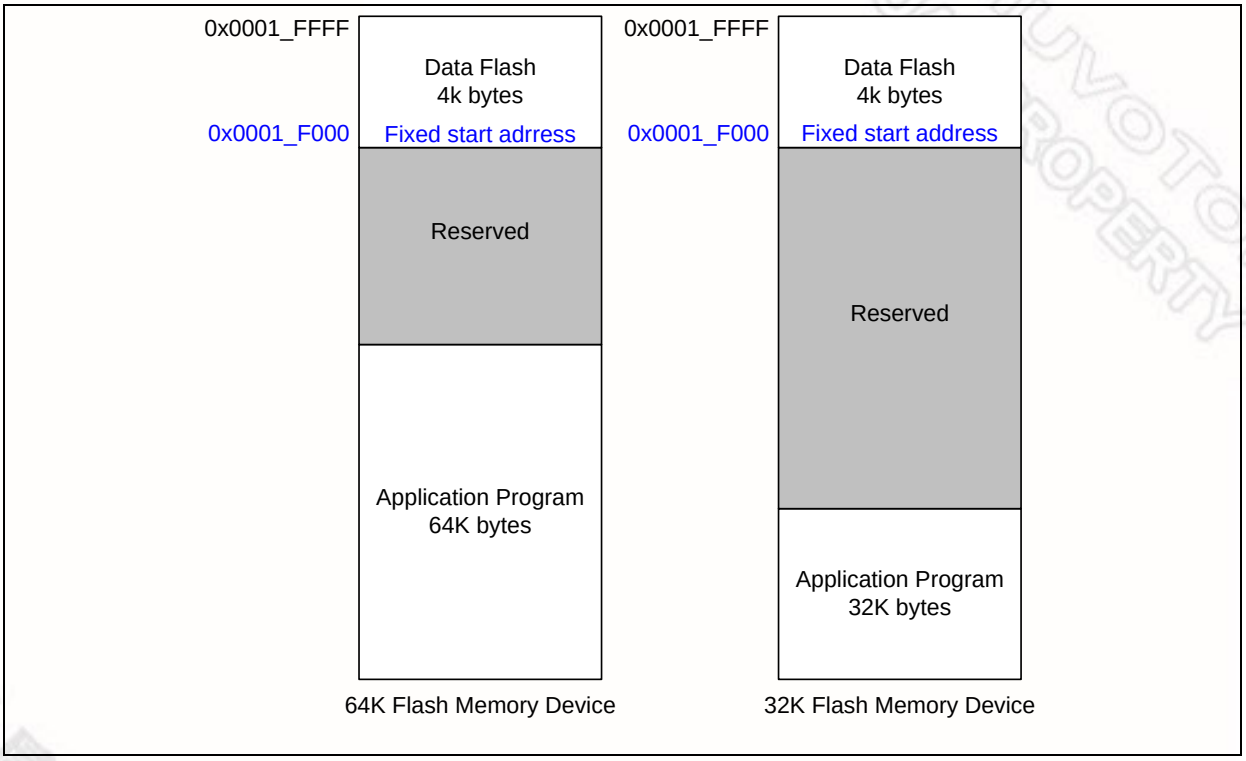


Figure 5-9 Flash Memory Structure

5.4.7 User Configuration

5.4.7.1 Config0 (Address = 0x0030_0000)

31	30	29	28	27	26	25	24
Reserved			CKF	Reserved	CFOSC		
23	22	21	20	19	18	17	16
CBODEN	CBOV1	CBOV0	CBORST	Reserved			
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CBS	Reserved					LOCK	Reserved

Bits	Descriptions	
[31:29]	**Reserved**	Reserved
[28]	**CKF**	**XT1 Clock Filter Enable** 0 = Disable XT1 clock filter 1 = Enable XT1 clock filter
[27]	**Reserved**	Reserved
[26:24]	**CFOSC**	**CPU Clock Source Selection After Reset** FOSC[2:0] / Clock Source 000 / External 4~24 MHz high speed crystal clock 111 / Internal RC 22.1184 MHz high speed oscillator clock Others / Reserved The value of CFOSC will be load to CLKSEL0.HCLK_S[2:0] in system register after any reset occurs.
[23]	**CBODEN**	**Brownout Detector Enable** 0= Enable brownout detecting function after power on 1= Disable brownout detecting function after power on
[22:21]	**CBOV1-0**	**Brownout Voltage Selection** CBOV1 / CBOV0 / Brownout voltage 1 / 1 / 4.5 V 1 / 0 / 3.8 V 0 / 1 / 2.7 V

		0	0	2.2 V	
[20]	**CBORST**	**Brownout Reset Enable** 0 = Enable brownout reset after power on 1 = Disable brownout reset after power on			
[19:8]	**Reserved**	Reserved			
[7]	**CBS**	**Chip Boot Selection** 0 = Chip boot from LDROM 1 = Chip boot from APROM			
[6:2]	**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.			
[0]	**Reserved**	Reserved			

5.4.8 In System Program (ISP)

The program memory and data flash supports both in hardware programming and in system programming (ISP). Hardware programming mode uses gang-writers to reduce programming costs and time to market while the products enter into the mass production state. However, if the product is just under development or the end product needs firmware updating in the hand of an end user, the hardware programming mode will make repeated programming difficult and inconvenient. ISP method makes it easy and possible. NuMicro™ NUC122 Series supports ISP mode allowing a device to be reprogrammed under software control. Furthermore, the capability to update the application firmware makes wide range of applications possible.

ISP is performed without removing the microcontroller from the system. Various interfaces enable LDROM firmware to get new program code easily. The most common method to perform ISP is via UART along with the firmware in LDROM. General speaking, PC transfers the new APROM code through serial port. Then LDROM firmware receives it and re-programs into APROM through ISP commands. Nuvoton provides ISP firmware and PC application program for NuMicro™ NUC122 Series. It makes users quite easy perform ISP through Nuvoton ISP tool.

5.4.8.1 ISP Procedure

NuMicro™ NUC122 Series supports booting from APROM or LDROM initially defined by user configuration bit (CBS). If user wants to update application program in APROM, he can write BS=1 and starts software reset to make chip boot from LDROM. The first step to start ISP function is write ISPEN bit to 1. S/W is required to write REGWRPROT register in Global Control Register (GCR, 0x5000_0100) with 0x59, 0x16 and 0x88 before writing ISPCON register. This procedure is used to protect flash memory from destroying owning to unintended write during power on/off duration.

Several error conditions are checked after software writes ISPGO bit. If error condition occurs, ISP operation is not been started and ISP fail flag will be set instead of. ISPFF flag is cleared by s/w, it will not be over written in next ISP operation. The next ISP procedure can be started even ISPFF bit keeps at 1. It is recommended that s/w to check ISPFF bit and clear it after each ISP operation if it is set to 1.

When ISPGO bit is set, CPU will wait for ISP operation finished, during this period; peripheral still keeps working as usual. If any interrupt request occurs, CPU will not service it till ISP operation is finished.

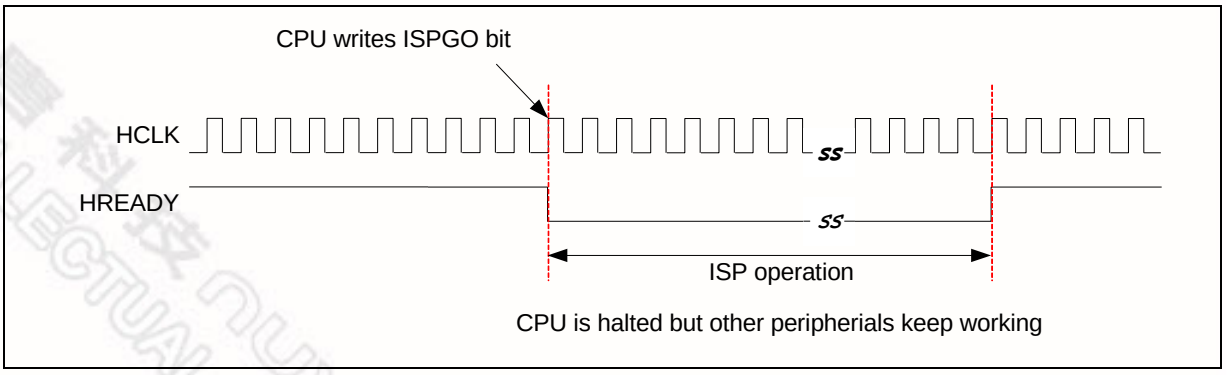


Figure 5-10 ISP Operation Timing

Note that NuMicro™ NUC122 Series allows user to update CONFIG value by ISP.

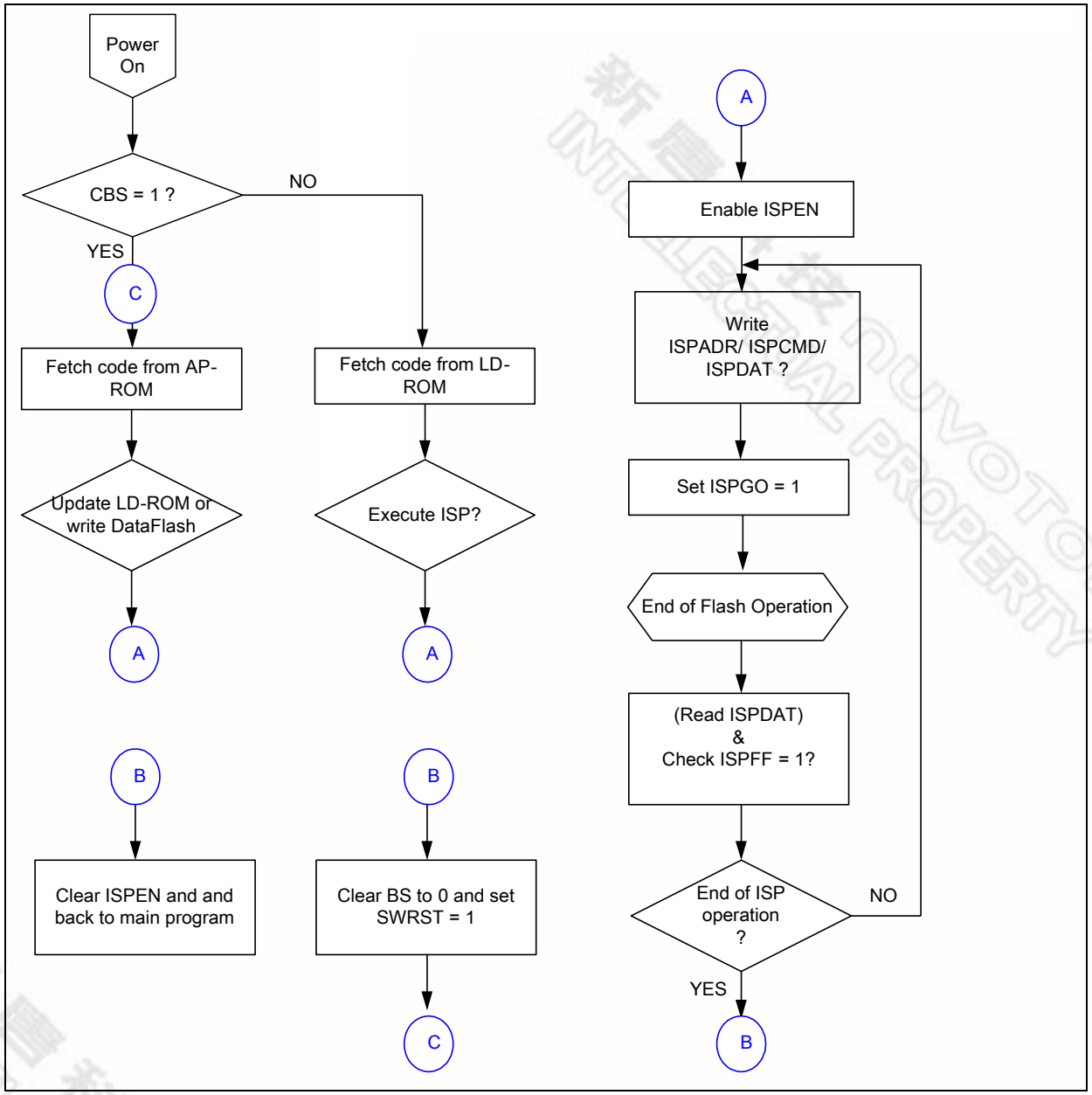


Figure 5-11 ISP Flow Chart

ISP Mode	ISPCMD			ISPADR			ISPDAT
	FOEN	FCEN	FCTRL[3:0]	A21	A20	A[19:0]	D[31:0]
FLASH Page Erase	1	0	0010	0	A20	Address in A[19:0]	x
FLASH Program	1	0	0001	0	A20	Address in A[19:0]	Data in D[31:0]
FLASH Read	0	0	0000	0	A20	Address in A[19:0]	Data out D[31:0]
CONFIG Page Erase	1	0	0010	1	1	Address in A[19:0]	x
CONFIG Program	1	0	0001	1	1	Address in A[19:0]	Data in D[31:0]
CONFIG Read	0	0	0000	1	1	Address in A[19:0]	Data out D[31:0]

Table 5-6 ISP Mode

5.4.9 Register Map

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

Register	Offset	R/W	Description	Reset Value
Base Address (FMC_BA) : 0x5000_C000				
ISPCON	FMC_BA+0x000	R/W	ISP Control Register	0x0000_0000
ISPADR	FMC_BA+0x004	R/W	ISP Address Register	0x0000_0000
ISPDAT	FMC_BA+0x008	R/W	ISP Data Register	0x0000_0000
ISPCMD	FMC_BA+0x00C	R/W	ISP Command Register	0x0000_0000
ISPTRG	FMC_BA+0x010	R/W	ISP Trigger Register	0x0000_0000
FATCON	FMC_BA+0x018	R/W	Flash Access Window Control Register	0x0000_0000

5.4.10 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	ET			Reserved	PT		
7	6	5	4	3	2	1	0
Reserved	ISPFF	LDUEN	CFGUEN	Reserved		BS	ISPEN

Bits	Descriptions	
[31:15]	**Reserved**	Reserved
[14:12]	**ET[2:0]**	**Flash Erase Time** (write-protection bits)

ET[2]	ET[1]	ET[0]	Erase Time (ms)
0	0	0	20 (default)
0	0	1	25
0	1	0	30
0	1	1	35
1	0	0	3
1	0	1	5
1	1	0	10
1	1	1	15

[11]	**Reserved**	Reserved
[8:10]	**PT[2:0]**	**Flash Program Time** (write-protection bits)

PT[2]	PT[1]	PT[0]	Program Time (us)
0	0	0	40
0	0	1	45
0	1	0	50

		0 1 1 55 1 0 0 20 1 0 1 25 1 1 0 30 1 1 1 35
[7]	**Reserved**	Reserved
[6]	**ISPFF**	**ISP Fail Flag** (write-protection bit) This bit is set by hardware when a triggered ISP meets any of the following conditions: (1) APROM writes to itself (2) LDROM writes to itself (3) CONFIG is erased/programmed if CFGUEN is set to 0 (4) Destination address is illegal, such as over an available range Write 1 to clear.
[5]	**LDUEN**	**LDROM Update Enable** (write-protection bit) LDROM update enable bit. 1 = LDROM can be updated when the chip runs in APROM 0 = LDROM can not be updated
[4]	**CFGUEN**	**Enable Config-bits Update by ISP** (write-protection bit) 1 = Enable ISP can update config-bits 0 = Disable ISP can update config-bits
[3:2]	**Reserved**	Reserved
[1]	**BS**	**Boot Select** (write-protection bit) 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. 1 = boot from LDROM 0 = boot from APROM
[0]	**ISPEN**	**ISP Enable** (write-protection bit) ISP function enable bit. Set this bit to enable ISP function. 1 = Enable ISP function 0 = Disable ISP function

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[31:24]							
23	22	21	20	19	18	17	16
ISPADR[23:16]							
15	14	13	12	11	10	9	8
ISPADR[15:8]							
7	6	5	4	3	2	1	0
ISPADR[7:0]							

Bits	Descriptions	
[31:0]	**ISPADR**	**ISP Address** NuMicro™ NUC122 Series equips with a maximum 16Kx32 embedded flash, it 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[31:24]							
23	22	21	20	19	18	17	16
ISPDAT[23:16]							
15	14	13	12	11	10	9	8
ISPDAT[15:8]							
7	6	5	4	3	2	1	0
ISPDAT[7:0]							

Bits	Descriptions	
[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		FOEN	FCEN	FCTRL[3:0]			

Bits	Descriptions	
[31:6]	**Reserved**	Reserved
[5:0]	**FOEN, FCEN, FCTRL**	**ISP Command** ISP command table is showed below:

Operation Mode	FOEN	FCEN	FCTRL[3:0]			
Read	0	0	0	0	0	0
Program	1	0	0	0	0	1
Page Erase	1	0	0	0	1	0

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	Descriptions	
[31:1]	**Reserved**	Reserved
[0]	**ISPGO**	**ISP Start Trigger** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. 1 = ISP is on going 0 = ISP operation is finished

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	MFOM	Reserved	LFOM	Reserved			

Bits	Descriptions	
[31:7]	**Reserved**	Reserved
[6]	**MFOM**	**Middle Frequency Optimization Mode** (write-protection bit) If chip operation frequency is between 20 MHz ~ 40 MHz, chip can work more efficiently when this bit is set to 1. If chip operation frequency is > 40 MHz, both of LFOM and MFOM have to set to zero.
[5]	**Reserved**	Reserved
[4]	**LFOM**	**Low Frequency Optimization Mode** (write-protection bit) If chip operation frequency lower than 20 MHz, chip can work more efficiently when this bit is set to 1. If chip operation frequency is > 40 MHz, both of LFOM and MFOM have to set to zero.
[3:0]	**Reserved**	Reserved

5.5 General Purpose I/O (GPIO)

5.5.1 Overview and Features

NuMicro™ NUC122 Series has up to 41 General Purpose I/O pins can be shared with other function pins; it depends on the chip configuration. These 41 pins are arranged in 4 ports named with GPIOA, GPIOB, GPIOC and GPIOD. Each port equips maximum 16 pins. Each one of the 41 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 type of all pins stay in quasi-bidirectional mode and port data register GPIOx_DOUT[15:0] resets to 0x0000_FFFF. Each I/O pin equips a very weakly individual pull-up resistor which is about 110 KΩ ~ 300 KΩ for V_{DD} is from 5.5 V to 2.5 V.

5.5.2 Function Description

5.5.2.1 Input Mode Explanation

Set GPIOx_PMD (PMDn[1:0]) to 00b 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.

5.5.2.2 Output Mode Explanation

Set GPIOx_PMD (PMDn[1:0]) to 01b the GPIOx port [n] pin is in 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.

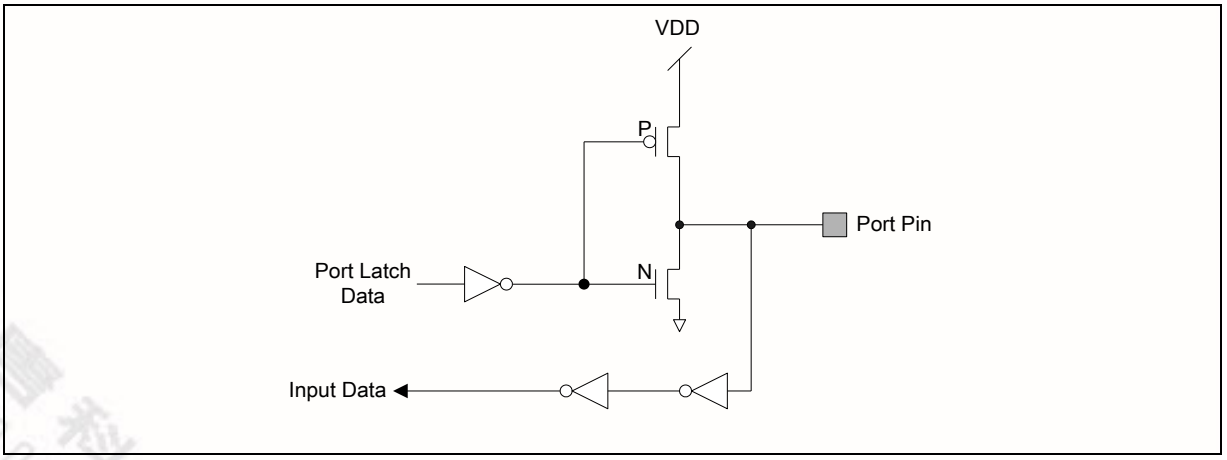


Figure 5-12 Push-Pull Output

5.5.2.3 Open-Drain Mode Explanation

Set GPIOx_PMD (PMDn[1:0]) to 10b 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 resister 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 high resistor.

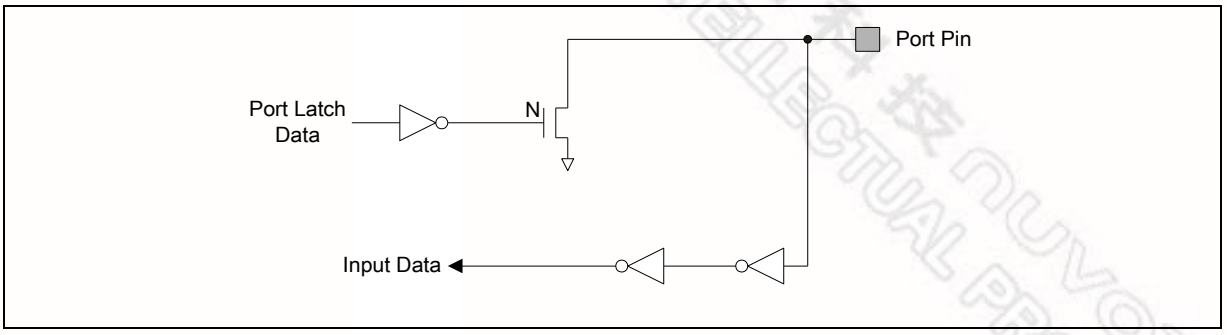


Figure 5-13 Open-Drain Output

5.5.2.4 Quasi-bidirectional Mode Explanation

Set GPIOx_PMD (PMDn[1:0]) to 11b 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 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 VDD is form 5.0 V to 2.5 V.

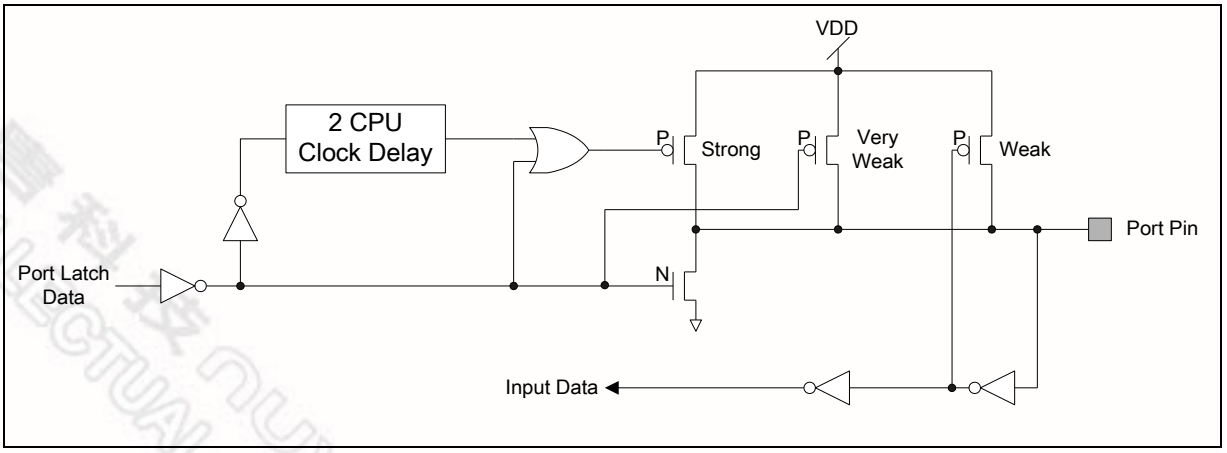


Figure 5-14 Quasi-bidirectional I/O Mode

5.5.3 Register Map

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

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

GPIOC_ISRC	GP_BA+0x0A0	R/W	GPIO Port C Interrupt Source Flag	0xXXXX_XXXX
GPIOD_PMD	GP_BA+0x0C0	R/W	GPIO Port D Pin I/O Mode Control	0xFFFF_FFFF
GPIOD_OFFD	GP_BA+0x0C4	R/W	GPIO Port D Pin OFF Digital Enable	0x0000_0000
GPIOD_DOUT	GP_BA+0x0C8	R/W	GPIO Port D Data Output Value	0x0000_FFFF
GPIOD_DMASK	GP_BA+0x0CC	R/W	GPIO Port D Data Output Write Mask	0x0000_0000
GPIOD_PIN	GP_BA+0x0D0	R	GPIO Port D Pin Value	0x0000_XXXX
GPIOD_DBEN	GP_BA+0x0D4	R/W	GPIO Port D De-bounce Enable	0x0000_0000
GPIOD_IMD	GP_BA+0x0D8	R/W	GPIO Port D Interrupt Mode Control	0x0000_0000
GPIOD_IEN	GP_BA+0x0DC	R/W	GPIO Port D Interrupt Enable	0x0000_0000
GPIOD_ISRC	GP_BA+0x0E0	R/W	GPIO Port D Interrupt Source Flag	0xXXXX_XXXX
DBNCECON	GP_BA+0x180	R/W	De-bounce Cycle Control	0x0000_0020
GPIOA0_DOUT	GP_BA+0x200	R/W	GPIO PA.0 Bit Output/Input Value	0x0000_0001
GPIOA1_DOUT	GP_BA+0x204	R/W	GPIO PA.1 Bit Output/Input Value	0x0000_0001
GPIOA2_DOUT	GP_BA+0x208	R/W	GPIO PA.2 Bit Output/Input Value	0x0000_0001
GPIOA3_DOUT	GP_BA+0x20C	R/W	GPIO PA.3 Bit Output/Input Value	0x0000_0001
GPIOA4_DOUT	GP_BA+0x210	R/W	GPIO PA.4 Bit Output/Input Value	0x0000_0001
GPIOA5_DOUT	GP_BA+0x214	R/W	GPIO PA.5 Bit Output/Input Value	0x0000_0001
GPIOA6_DOUT	GP_BA+0x218	R/W	GPIO PA.6 Bit Output/Input Value	0x0000_0001
GPIOA7_DOUT	GP_BA+0x21C	R/W	GPIO PA.7 Bit Output/Input Value	0x0000_0001
GPIOA8_DOUT	GP_BA+0x220	R/W	GPIO PA.8 Bit Output/Input Value	0x0000_0001
GPIOA9_DOUT	GP_BA+0x224	R/W	GPIO PA.9 Bit Output/Input Value	0x0000_0001
GPIOA10_DOUT	GP_BA+0x228	R/W	GPIO PA.10 Bit Output/Input Value	0x0000_0001
GPIOA11_DOUT	GP_BA+0x22C	R/W	GPIO PA.11 Bit Output/Input Value	0x0000_0001
GPIOA12_DOUT	GP_BA+0x230	R/W	GPIO PA.12 Bit Output/Input Value	0x0000_0001
GPIOA13_DOUT	GP_BA+0x234	R/W	GPIO PA.13 Bit Output/Input Value	0x0000_0001
GPIOA14_DOUT	GP_BA+0x238	R/W	GPIO PA.14 Bit Output/Input Value	0x0000_0001
GPIOA15_DOUT	GP_BA+0x23C	R/W	GPIO PA.15 Bit Output/Input Value	0x0000_0001
GPIOB0_DOUT	GP_BA+0x240	R/W	GPIO PB.0 Bit Output/Input Value	0x0000_0001
GPIOB1_DOUT	GP_BA+0x244	R/W	GPIO PB.1 Bit Output/Input Value	0x0000_0001
GPIOB2_DOUT	GP_BA+0x248	R/W	GPIO PB.2 Bit Output/Input Value	0x0000_0001

GPIOB3_DOUT	GP_BA+0x24C	R/W	GPIO PB.3 Bit Output/Input Value	0x0000_0001
GPIOB4_DOUT	GP_BA+0x250	R/W	GPIO PB.4 Bit Output/Input Value	0x0000_0001
GPIOB5_DOUT	GP_BA+0x254	R/W	GPIO PB.5 Bit Output/Input Value	0x0000_0001
GPIOB6_DOUT	GP_BA+0x258	R/W	GPIO PB.6 Bit Output/Input Value	0x0000_0001
GPIOB7_DOUT	GP_BA+0x25C	R/W	GPIO PB.7 Bit Output/Input Value	0x0000_0001
GPIOB8_DOUT	GP_BA+0x260	R/W	GPIO PB.8 Bit Output/Input Value	0x0000_0001
GPIOB9_DOUT	GP_BA+0x264	R/W	GPIO PB.9 Bit Output/Input Value	0x0000_0001
GPIOB10_DOUT	GP_BA+0x268	R/W	GPIO PB.10 Bit Output/Input Value	0x0000_0001
GPIOB11_DOUT	GP_BA+0x26C	R/W	GPIO PB.11 Bit Output/Input Value	0x0000_0001
GPIOB12_DOUT	GP_BA+0x270	R/W	GPIO PB.12 Bit Output/Input Value	0x0000_0001
GPIOB13_DOUT	GP_BA+0x274	R/W	GPIO PB.13 Bit Output/Input Value	0x0000_0001
GPIOB14_DOUT	GP_BA+0x278	R/W	GPIO PB.14 Bit Output/Input Value	0x0000_0001
GPIOB15_DOUT	GP_BA+0x27C	R/W	GPIO PB.15 Bit Output/Input Value	0x0000_0001
GPIOC0_DOUT	GP_BA+0x280	R/W	GPIO PC.0 Bit Output/Input Value	0x0000_0001
GPIOC1_DOUT	GP_BA+0x284	R/W	GPIO PC.1 Bit Output/Input Value	0x0000_0001
GPIOC2_DOUT	GP_BA+0x288	R/W	GPIO PC.2 Bit Output/Input Value	0x0000_0001
GPIOC3_DOUT	GP_BA+0x28C	R/W	GPIO PC.3 Bit Output/Input Value	0x0000_0001
GPIOC4_DOUT	GP_BA+0x290	R/W	GPIO PC.4 Bit Output/Input Value	0x0000_0001
GPIOC5_DOUT	GP_BA+0x294	R/W	GPIO PC.5 Bit Output/Input Value	0x0000_0001
GPIOC6_DOUT	GP_BA+0x298	R/W	GPIO PC.6 Bit Output/Input Value	0x0000_0001
GPIOC7_DOUT	GP_BA+0x29C	R/W	GPIO PC.7 Bit Output/Input Value	0x0000_0001
GPIOC8_DOUT	GP_BA+0x2A0	R/W	GPIO PC.8 Bit Output/Input Value	0x0000_0001
GPIOC9_DOUT	GP_BA+0x2A4	R/W	GPIO PC.9 Bit Output/Input Value	0x0000_0001
GPIOC10_DOUT	GP_BA+0x2A8	R/W	GPIO PC.10 Bit Output/Input Value	0x0000_0001
GPIOC11_DOUT	GP_BA+0x2AC	R/W	GPIO PC.11 Bit Output/Input Value	0x0000_0001
GPIOC12_DOUT	GP_BA+0x2B0	R/W	GPIO PC.12 Bit Output/Input Value	0x0000_0001
GPIOC13_DOUT	GP_BA+0x2B4	R/W	GPIO PC.13 Bit Output/Input Value	0x0000_0001
GPIOC14_DOUT	GP_BA+0x2B8	R/W	GPIO PC.14 Bit Output/Input Value	0x0000_0001
GPIOC15_DOUT	GP_BA+0x2BC	R/W	GPIO PC.15 Bit Output/Input Value	0x0000_0001
GPIOD0_DOUT	GP_BA+0x2C0	R/W	GPIO PD.0 Bit Output/Input Value	0x0000_0001

GPIOD1_DOUT	GP_BA+0x2C4	R/W	GPIO PD.1 Bit Output/Input Value	0x0000_0001
GPIOD2_DOUT	GP_BA+0x2C8	R/W	GPIO PD.2 Bit Output/Input Value	0x0000_0001
GPIOD3_DOUT	GP_BA+0x2CC	R/W	GPIO PD.3 Bit Output/Input Value	0x0000_0001
GPIOD4_DOUT	GP_BA+0x2D0	R/W	GPIO PD.4 Bit Output/Input Value	0x0000_0001
GPIOD5_DOUT	GP_BA+0x2D4	R/W	GPIO PD.5 Bit Output/Input Value	0x0000_0001
GPIOD6_DOUT	GP_BA+0x2D8	R/W	GPIO PD.6 Bit Output/Input Value	0x0000_0001
GPIOD7_DOUT	GP_BA+0x2DC	R/W	GPIO PD.7 Bit Output/Input Value	0x0000_0001
GPIOD8_DOUT	GP_BA+0x2E0	R/W	GPIO PD.8 Bit Output/Input Value	0x0000_0001
GPIOD9_DOUT	GP_BA+0x2E4	R/W	GPIO PD.9 Bit Output/Input Value	0x0000_0001
GPIOD10_DOUT	GP_BA+0x2E8	R/W	GPIO PD.10 Bit Output/Input Value	0x0000_0001
GPIOD11_DOUT	GP_BA+0x2EC	R/W	GPIO PD.11 Bit Output/Input Value	0x0000_0001
GPIOD12_DOUT	GP_BA+0x2F0	R/W	GPIO PD.12 Bit Output/Input Value	0x0000_0001
GPIOD13_DOUT	GP_BA+0x2F4	R/W	GPIO PD.13 Bit Output/Input Value	0x0000_0001
GPIOD14_DOUT	GP_BA+0x2F8	R/W	GPIO PD.14 Bit Output/Input Value	0x0000_0001
GPIOD15_DOUT	GP_BA+0x2FC	R/W	GPIO PD.15 Bit Output/Input Value	0x0000_0001

5.5.4 Register Description

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

Register	Offset	R/W	Description	Reset Value
GPIOA_PMD	GP_BA+0x000	R/W	GPIO Port A Pin I/O Mode Control	0xFFFF_FFFF
GPIOB_PMD	GP_BA+0x040	R/W	GPIO Port B Pin I/O Mode Control	0xFFFF_FFFF
GPIOC_PMD	GP_BA+0x080	R/W	GPIO Port C Pin I/O Mode Control	0xFFFF_FFFF
GPIOD_PMD	GP_BA+0x0C0	R/W	GPIO Port D Pin I/O Mode Control	0xFFFF_FFFF

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	Descriptions	
[2n+1:2n]	**PMDn**	**GPIOx I/O Pin[n] Mode Control** Determine each I/O type of GPIOx pins. 00 = GPIO port [n] pin is in INPUT mode 01 = GPIO port [n] pin is in OUTPUT mode 10 = GPIO port [n] pin is in Open-Drain mode 11 = GPIO port [n] pin is in Quasi-bidirectional mode

GPIO Port [A/B/C/D] Pin OFF Digital Resistor Enable (GPIOx_OFFD)

Register	Offset	R/W	Description	Reset Value
GPIOA_OFFD	GP_BA+0x004	R/W	GPIO Port A Pin OFF Digital Enable	0x0000_0000
GPIOB_OFFD	GP_BA+0x044	R/W	GPIO Port B Pin OFF Digital Enable	0x0000_0000
GPIOC_OFFD	GP_BA+0x084	R/W	GPIO Port C Pin OFF Digital Enable	0x0000_0000
GPIOD_OFFD	GP_BA+0x0C4	R/W	GPIO Port D Pin OFF Digital Enable	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	Descriptions	
[31:16]	**OFFD**	**GPIOx Pin[n] OFF Digital Input Path Enable** Each of these bits is used to control if the input path of corresponding GPIO pin is disabled. If input is analog signal, users can OFF digital input path to avoid **creepage**, 1 = Disable IO digital input path (digital input tied to low) 0 = Enable IO digital input path
[15:0]	**Reserved**	Reserved

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

Register	Offset	R/W	Description	Reset Value
GPIOA_DOUT	GP_BA+0x008	R/W	GPIO Port A Data Output Value	0x0000_FFFF
GPIOB_DOUT	GP_BA+0x048	R/W	GPIO Port B Data Output Value	0x0000_FFFF
GPIOC_DOUT	GP_BA+0x088	R/W	GPIO Port C Data Output Value	0x0000_FFFF
GPIOD_DOUT	GP_BA+0x0C8	R/W	GPIO Port D Data Output Value	0x0000_FFFF

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

Bits	Descriptions	
[31:16]	**Reserved**	Reserved
[n]	**DOUT[n]**	**GPIOx Pin[n] Output Value** Each of these bits control the status of a GPIO pin when the GPIO pin is configures as output, open-drain and quasi-mode. 1 = GPIO port [A/B/C/D] Pin[n] will drive High if the GPIO pin is configures as output, open-drain and quasi-mode. 0 = GPIO port [A/B/C/D] Pin[n] will drive Low if the GPIO pin is configures as output, open-drain and quasi-mode.

GPIO Port [A/B/C/D] Data Output Write Mask (GPIOx_DMASK)

Register	Offset	R/W	Description	Reset Value
GPIOA_DMASK	GP_BA+0x00C	R/W	GPIO Port A Data Output Write Mask	0xXXXX_0000
GPIOB_DMASK	GP_BA+0x04C	R/W	GPIO Port B Data Output Write Mask	0xXXXX_0000
GPIOC_DMASK	GP_BA+0x08C	R/W	GPIO Port C Data Output Write Mask	0xXXXX_0000
GPIOD_DMASK	GP_BA+0x0CC	R/W	GPIO Port D Data Output Write Mask	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**
DMASK[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMASK[7:0]							

Bits	Descriptions	
[31:16]	**Reserved**	Reserved
[n]	**DMASK[n]**	**Port [A/B/C/D] Data Output Write Mask** These bits are used to protect the corresponding register of GPIOx_DOUT bit[n] . When set the DMASK bit[n] to 1, the corresponding GPIOx_DOUT[n] bit is protected. The write signal is masked, write data to the protect bit is ignored 1 = The corresponding GPIOx_DOUT[n] bit is protected 0 = The corresponding GPIOx_DOUT[n] bit can be updated Note: This function only protect corresponding GPIOx_DOUT[n] bit, and will not protect corresponding bit control register (GPIOAx_DOUT, GPIOBx_DOUT, GPIOCx_DOUT, GPIODx_DOUT).

GPIO Port [A/B/C/D] Pin Value (GPIOx_PIN)

Register	Offset	R/W	Description	Reset Value
GPIOA_PIN	GP_BA+0x010	R	GPIO Port A Pin Value	0x0000_XXXX
GPIOB_PIN	GP_BA+0x050	R	GPIO Port B Pin Value	0x0000_XXXX
GPIOC_PIN	GP_BA+0x090	R	GPIO Port C Pin Value	0x0000_XXXX
GPIOD_PIN	GP_BA+0x0D0	R	GPIO Port D Pin Value	0x0000_XXXX

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

Bits	Descriptions	
[31:16]	**Reserved**	Reserved
[n]	**PIN[n]**	**Port [A/B/C/D] Pin Values** Each bit of the register reflects the actual status of the respective GPIO pin If bit is 1, it indicates the corresponding pin status is high, else the pin status is low

GPIO Port [A/B/C/D] De-bounce Enable (GPIOx_DBEN)

Register	Offset	R/W	Description	Reset Value
GPIOA_DBEN	GP_BA+0x014	R/W	GPIO Port A De-bounce Enable	0xXXXX_0000
GPIOB_DBEN	GP_BA+0x054	R/W	GPIO Port B De-bounce Enable	0xXXXX_0000
GPIOC_DBEN	GP_BA+0x094	R/W	GPIO Port C De-bounce Enable	0xXXXX_0000
GPIOD_DBEN	GP_BA+0x0D4	R/W	GPIO Port D De-bounce Enable	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**
DBEN[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBEN[7:0]							

Bits	Descriptions	
[31:16]	**Reserved**	Reserved
[n]	**DBEN[n]**	**Port [A/B/C/D] Input Signal De-bounce Enable** DBEN[n]used to enable the de-bounce function for each corresponding bit. If the input signal pulse width can't 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 is controlled by DBNCECON[3:0] The DBEN[n] is used for "edge-trigger" interrupt only, and ignored for "level trigger" interrupt 1 = The bit[n] de-bounce function is enabled 0 = The bit[n] de-bounce function is disabled The de-bounce function is valid for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored.

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

Register	Offset	R/W	Description	Reset Value
GPIOA_IMD	GP_BA+0x018	R/W	GPIO Port A Interrupt Mode Control	0xXXXX_0000
GPIOB_IMD	GP_BA+0x058	R/W	GPIO Port B Interrupt Mode Control	0xXXXX_0000
GPIOC_IMD	GP_BA+0x098	R/W	GPIO Port C Interrupt Mode Control	0xXXXX_0000
GPIOD_IMD	GP_BA+0x0D8	R/W	GPIO Port D Interrupt Mode Control	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**
IMD[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IMD[7:0]							

Bits	Descriptions	
[31:16]	**Reserved**	Reserved
[n]	**IMD[n]**	**Port [A/B/C/D] 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. 1 = Level trigger interrupt 0 = Edge trigger interrupt If set pin as the level trigger interrupt, then only one level can be set on the registers GPIOx_IEN. If set both the level to trigger interrupt, the setting is ignored and no interrupt will occur The de-bounce function is valid for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored.

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

Register	Offset	R/W	Description	Reset Value
GPIOA_IEN	GP_BA+0x01C	R/W	GPIO Port A Interrupt Enable	0x0000_0000
GPIOB_IEN	GP_BA+0x05C	R/W	GPIO Port B Interrupt Enable	0x0000_0000
GPIOC_IEN	GP_BA+0x09C	R/W	GPIO Port C Interrupt Enable	0x0000_0000
GPIOD_IEN	GP_BA+0x0DC	R/W	GPIO Port D Interrupt Enable	0x0000_0000

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

Bits	Descriptions	
[n+16]	**IR_EN[n]**	**Port [A/B/C/D] 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 set 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. 1 = Enable the PIN[n] level-high or low-to-high interrupt 0 = Disable the PIN[n] level-high or low-to-high interrupt
[n]	**IF_EN[n]**	**Port [A/B/C/D] Interrupt Enable by Input Falling Edge or Input Level Low** IF_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 set 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. 1 = Enable the PIN[n] state low-level or high-to-low change interrupt 0 = Disable the PIN[n] state low-level or high-to-low change interrupt

GPIO Port [A/B/C/D] Interrupt Trigger Source (GPIOx _ISRC)

Register	Offset	R/W	Description	Reset Value
GPIOA_ISRC	GP_BA+0x020	R/W	GPIO Port A Interrupt Trigger Source Indicator	0x0000_0000
GPIOB_ISRC	GP_BA+0x060	R/W	GPIO Port B Interrupt Trigger Source Indicator	0x0000_0000
GPIOC_ISRC	GP_BA+0x0A0	R/W	GPIO Port C Interrupt Trigger Source Indicator	0x0000_0000
GPIOD_ISRC	GP_BA+0x0E0	R/W	GPIO Port D Interrupt Trigger Source Indicator	0x0000_0000

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

Bits	Descriptions	
[31:16]	**Reserved**	Reserved
[n]	**ISRC[n]**	**Port [A/B/C/D] Interrupt Trigger Source Indicator** Read : 1 = Indicates GPIOx[n] generate an interrupt 0 = No interrupt at GPIOx[n] Write : 1= Clear the correspond pending interrupt 0= No action

Interrupt De-bounce Cycle Control (DBNCECON)

Register	Offset	R/W	Description	Reset Value
DBNCECON	GP_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	Descriptions	
[5]	ICLK_ON	**Interrupt Clock On Mode** Set this bit to 0 will disable the interrupt generate circuit clock, if the pin[n] interrupt is disabled 1 = Interrupt generated circuit clock always enable 0 = Disable the clock if the GPIOA/B/C/D[n] interrupt is disabled
[4]	DBCLKSRC	**De-bounce Counter Clock Source Selection** 1 = De-bounce counter clock source is the internal 10 KHz low speed clock 0 = De-bounce counter clock source is the HCLK
[3:0]	DBCLKSEL	**De-bounce Sampling Cycle Selection**

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

		9	Sample interrupt input once per 2*256 clocks	
		10	Sample interrupt input once per 4*256clocks	
		11	Sample interrupt input once per 8*256 clocks	
		12	Sample interrupt input once per 16*256 clocks	
		13	Sample interrupt input once per 32*256 clocks	
		14	Sample interrupt input once per 64*256 clocks	
		15	Sample interrupt input once per 128*256 clocks	

GPIO Port [A/B/C/D] I/O Bit Output/Input Control (GPIOxx_DOUT)

Register	Offset	R/W	Description	Reset Value
GPIOAx_DOUT	GP_BA+0x200 - GP_BA+0x23C	R/W	GPIO Port A Pin I/O Bit Output/Input Control	0x0000_0001
GPIOBx_DOUT	GP_BA+0x240 - GP_BA+0x27C	R/W	GPIO Port B Pin I/O Bit Output/Input Control	0x0000_0001
GPIOCx_DOUT	GP_BA+0x280 - GP_BA+0x2BC	R/W	GPIO Port C Pin I/O Bit Output/Input Control	0x0000_0001
GPIODx_DOUT	GP_BA+0x2C0 - GP_BA+0x2FC	R/W	GPIO Port D Pin I/O Bit Output/Input Control	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							**GPIOxx_DOUT**

Bits	Descriptions	
[0]	**GPIOxx_DOUT**	**GPIOxx I/O Pin Bit Output/Input Control** Write this bit can control one GPIO pin output value 1 = Set corresponding GPIO pin to high 0 = Set corresponding GPIO pin to low Read this register to get IO pin status. For example: write GPIOA0_DOUT will reflect the written value to bit GPIOA_DOUT[0], read GPIOA0_DOUT will return the value of GPIOA_PIN[0].

5.6 Timer Controller (TMR)

5.6.1 Overview

The timer controller includes four 32-bit timers, TIMER0~TIMER3, which allows user to easily implement a timer control for applications. The timer can perform functions like frequency measurement, event counting, interval measurement, clock generation, delay timing, and so on. The timer can generates an interrupt signal upon timeout, or provide the current value during operation.

5.6.2 Features

- 4 sets of 32-bit timers with 24-bit up-timer and one 8-bit pre-scale 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 pre-scale 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 timer value is readable through TDR (Timer Data Register)
- Support event counting function to count the event from external pin

5.6.3 Block Diagram

Each channel is equipped with an 8-bit pre-scale counter, a 24-bit up-timer, a 24-bit compare register and an interrupt request signal. There are four options of clock sources for each channel. Figure 5-15 Timer Controller Clock Source Diagram illustrates the clock source control function. Refer to Figure 5-16 Timer Controller Block Diagram for the Timer controller block diagram. Software can program the 8-bit pre-scale counter to decide the clock period to 24-bit up timer.

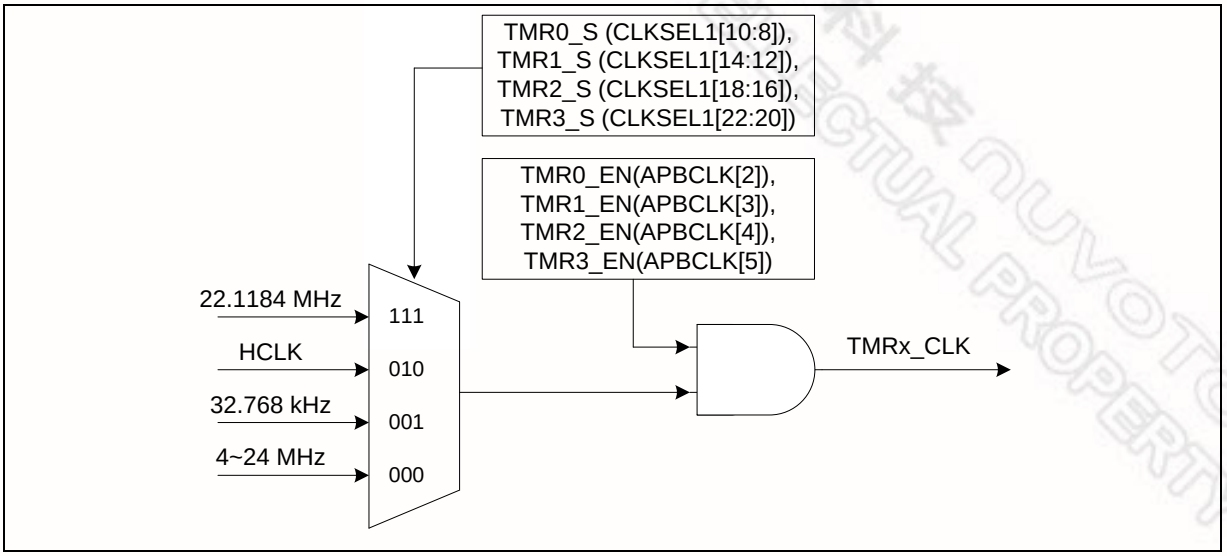


Figure 5-15 Timer Controller Clock Source Diagram

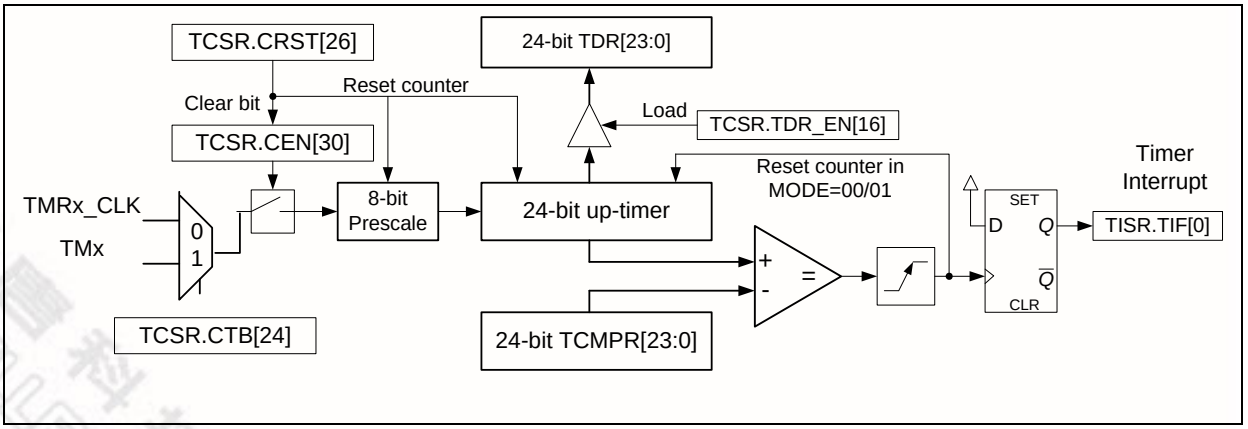


Figure 5-16 Timer Controller Block Diagram

5.6.4 Function Description

Timer controller provides one-shot, period, toggle and continuous counting operation modes. It also provides the event counting function to count the event from external pin. Each operating function mode is shown as following:

5.6.4.1 One –Shot Mode

If timer is operated at one-shot mode and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value reaches timer compare register (TCMPR) value, if IE (TCSR[29] interrupt enable bit) is set to 1, then the timer interrupt flag is set and the interrupt signal is generated and sent to NVIC to inform CPU. It indicates 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 reaches timer compare register (TCMPR) value, the timer counter value goes back to counting initial value and CEN (timer enable bit) is cleared to 0 by timer controller. Timer counting operation stops, once the timer counter value reaches timer compare register (TCMPR) value. 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 (timer enable bit) is set to 1. So, this operating mode is called One-Shot mode.

5.6.4.2 Periodic Mode

If timer is operated at period mode and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value reaches timer compare register (TCMPR) value, if IE (TCSR[29] interrupt enable bit) is set to 1, then the timer interrupt flag is set and the interrupt signal is generated and sent to NVIC to inform CPU. It indicates 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 reaches timer compare register (TCMPR) value, the timer counter value goes back to counting initial value and CEN is kept at 1 (counting enable continuously). The timer counter operates up counting again. If the interrupt flag is cleared by software, once the timer counter value reaches timer compare register (TCMPR) value and IE (interrupt enable bit) is set to 1'b1, then the timer interrupt flag is set and the interrupt signal is generated and sent to NVIC to inform CPU again. That is to say, timer operates timer counting and compares with TCMPR value function periodically. The timer counting operation doesn't stop until the CEN is set to 0. The interrupt signal is also generated periodically. So, this operating mode is called Periodic mode.

5.6.4.3 Toggle Mode

If timer is operated at toggle mode and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value reaches timer compare register (TCMPR) value, if IE (TCSR[29] interrupt enable bit) is set to 1, then the timer interrupt flag is set and the interrupt signal is generated and sent to NVIC to inform CPU. It indicates that the timer counting overflow happens. The associated toggle output (tout) signal is set to 1. In this operating mode, once the timer counter value reaches timer compare register (TCMPR) value, the timer counter value goes back to counting initial value and CEN is kept at 1 (counting enable continuously). The timer counter operates up counting again. If the interrupt flag is cleared by software, once the timer counter value reaches timer compare register (TCMPR) value and IE (interrupt enable bit) is set to 1, then the timer interrupt flag is set and the interrupt signal is generated and sent to NVIC to inform CPU again. The associated toggle output (tout) signal is set to 0. The timer counting operation doesn't stop until the CEN is set to 0. Thus, the toggle output (tout) signal is changing back and forth with 50% duty cycle. So, this operating mode is called Toggle mode.

5.6.4.4 Continuous Counting Mode

If the timer is operated at continuous counting mode and CEN (TCSR[30] timer enable bit) is set to 1, the associated interrupt signal is generated depending on TDR = TCMPR if IE (TCSR[29]

interrupt enable bit) is enabled. User can change different TCMPR value immediately without disabling timer counting and restarting timer counting. For example, TCMPR is set as 80, first. (The TCMPR should be less than 2^{24} and be greater than 1). The timer generates the interrupt if IE is enabled and TIF (timer interrupt flag) will set to 1 then the interrupt signal is generated and sent to NVIC to inform CPU when TDR value is equal to 80. But the CEN is kept at 1 (counting enable continuously) and TDR value will not goes back to 0, it continues to count 81, 82, 83, • • • to 2^{24} -1, 0, 1, 2, 3, • • • to 2^{24} -1 again and again. Next, if user programs TCMPR as 200 and the TIF is cleared to 0, then timer interrupt occurred and TIF is set to 1 then the interrupt signal is generated and sent to NVIC to inform CPU again when TDR value reaches to 200. At last, user programs TCMPR as 500 and clears TIF to 0 again, then timer interrupt occurred and TIF sets to 1 then the interrupt signal is generated and sent to NVIC to inform CPU when TDR value reaches to 500. From application view, the interrupt is generated depending on TCMPR. In this mode, the timer counting is continuous. So, this operation mode is called as continuous counting mode.

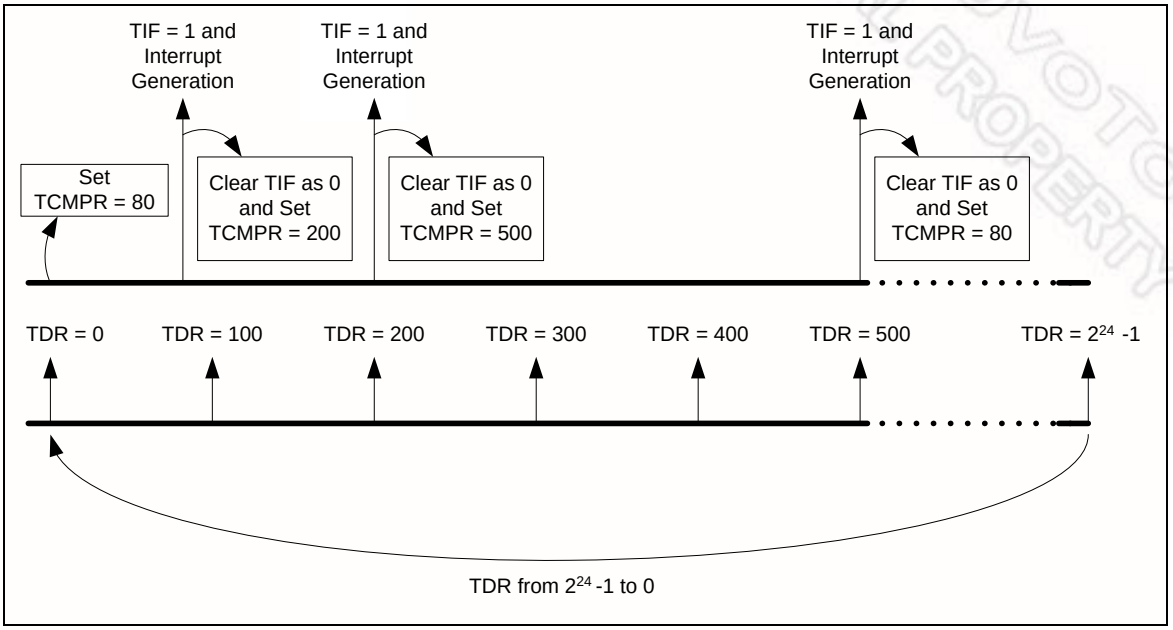


Figure 5-17 Continuous Counting Mode

5.6.4.5 Event Counting Function

It also provides an application which can count the event from TM0~TM3 pins. It is called as event counting mode. In event counting mode, the clock source of timer controller, TMRx_CLK, in Figure 5-15 should be set as HCLK. And, the event count source operating frequency should be less than 1/3 HCLK frequency.

5.6.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
TMR_BA01 = 0x4001_0000 **TMR_BA23 = 0x4011_0000**				
TCSR0	TMR_BA01+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TCMPR0	TMR_BA01+0x04	R/W	Timer0 Compare Register	0x0000_0000
TISR0	TMR_BA01+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TDR0	TMR_BA01+0x0C	R	Timer0 Data Register	0x0000_0000
TCSR1	TMR_BA01+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TCMPR1	TMR_BA01+0x24	R/W	Timer1 Compare Register	0x0000_0000
TISR1	TMR_BA01+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TDR1	TMR_BA01+0x2C	R	Timer1 Data Register	0x0000_0000
TCSR2	TMR_BA23+0x00	R/W	Timer2 Control and Status Register	0x0000_0005
TCMPR2	TMR_BA23+0x04	R/W	Timer2 Compare Register	0x0000_0000
TISR2	TMR_BA23+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TDR2	TMR_BA23+0x0C	R	Timer2 Data Register	0x0000_0000
TCSR3	TMR_BA23+0x20	R/W	Timer3 Control and Status Register	0x0000_0005
TCMPR3	TMR_BA23+0x24	R/W	Timer3 Compare Register	0x0000_0000
TISR3	TMR_BA23+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000
TDR3	TMR_BA23+0x2C	R	Timer3 Data Register	0x0000_0000

5.6.6 Register Description

Timer Control and Status Register (TCSR)

Register	Offset	R/W	Description	Reset Value
TCSR0	TMR_BA01+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TCSR1	TMR_BA01+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TCSR2	TMR_BA23+0x00	R/W	Timer2 Control and Status Register	0x0000_0005
TCSR3	TMR_BA23+0x20	R/W	Timer3 Control and Status Register	0x0000_0005

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

Bits	Descriptions	
[31]	**DBGACK_TMR**	**ICE Debug Mode Acknowledge Disable** (write-protection bit) 0 = ICE debug mode acknowledgement effects TIMER counting. TIMER counter will be held while ICE debug mode acknowledged. 1 = ICE debug mode acknowledgement disabled. TIMER counter will keep going no matter ICE debug mode acknowledged or not.
[30]	**CEN**	**Timer Enable Bit** 1 = Starts counting 0 = Stops/Suspends counting Note1: In stop status, and then set CEN to 1 will enables the 24-bit up-timer keeps up counting from the last stop counting value. Note2: This bit is auto-cleared by hardware in one-shot mode (MODE [28:27] =00) when the associated timer interrupt is generated (IE [29] =1).
[29]	**IE**	**Interrupt Enable Bit** 1 = Enable timer Interrupt 0 = Disable timer Interrupt If timer interrupt is enabled, the timer asserts its interrupt signal when the associated up-timer value is equal to TCMPR.

<table>
<tr><td>[28:27]</td><td>MODE</td><td>Timer Operating Mode
<table>
<tr><th>MODE</th><th>Timer Operating Mode</th></tr>
<tr><td>00</td><td>The timer is operating in the one-shot mode. The associated interrupt signal is generated once (if IE is enabled) and CEN is automatically cleared by hardware.</td></tr>
<tr><td>01</td><td>The timer is operating in the periodic mode. The associated interrupt signal is generated periodically (if IE is enabled).</td></tr>
<tr><td>10</td><td>The timer is operating in the toggle mode. The interrupt signal is generated periodically (if IE is enabled). And the associated signal (tout) is changing back and forth with 50 % duty cycle.</td></tr>
<tr><td>11</td><td>The timer is operating at continuous counting mode. The associated interrupt signal is generated when TDR = TCMPR (if IE is enabled). However, the 24-bit up-timer counts continuously. Please refer to Figure 5-17 for detail description about continuous counting mode operation.</td></tr>
</table></td></tr>
<tr><td>[26]</td><td>CRST</td><td>Timer Reset Bit
Set this bit will reset the 24-bit up-timer, pre-scale and also force CEN to 0.
0 = No effect
1 = Reset Timer’s pre-scale counter, internal 24-bit up-timer and CEN bit</td></tr>
<tr><td>[25]</td><td>CACT</td><td>Timer Active Status Bit (Read only)
This bit indicates the up-timer status.
0 = Timer is not active
1 = Timer is active</td></tr>
<tr><td>[24]</td><td>CTB</td><td>Counter Mode Enable Bit
This bit is the counter mode enable bit. When Timer is used as an event counter, this bit should be set to 1 and Timer will work as an event counter. The event is triggered by rising edge from external pin.
1 = Enable counter mode
0 = Disable counter mode</td></tr>
<tr><td>[23:17]</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>[16]</td><td>TDR_EN</td><td>Data Load Enable
When TDR_EN is set, TDR (Timer Data Register) will be updated continuously with the 24-bit up-timer value as the timer is counting.
1 = Timer Data Register update enable
0 = Timer Data Register update disable</td></tr>
<tr><td>[15:8]</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>[7:0]</td><td>PRESCALE</td><td>Pre-scale Counter
Clock input is divided by PRESCALE+1 before it is fed to the counter. If PRESCALE =0, then there is no scaling.</td></tr>
</table>

Timer Compare Register (TCMPR)

Register	Offset	R/W	Description	Reset Value
TCMPR0	TMR_BA01+0x04	R/W	Timer0 Compare Register	0x0000_0000
TCMPR1	TMR_BA01+0x24	R/W	Timer1 Compare Register	0x0000_0000
TCMPR2	TMR_BA23+0x04	R/W	Timer2 Compare Register	0x0000_0000
TCMPR3	TMR_BA23+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[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TCMP[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TCMP[7:0]							

Bits	Descriptions	
[31:24]	**Reserved**	Reserved
[23:0]	**TCMP**	**Timer Compared Value** TCMP is a 24-bit compared register. When the internal 24-bit up-timer counts and its value is equal to TCMP value, a Timer Interrupt is requested if the timer interrupt is enabled with TCSR.IE[29]=1. The TCMP value defines the timer counting cycle time. Time-out period = (Period of timer clock input) * (8-bit PRESCALE + 1) * (24-bit TCMP) Note1: Never write 0x0 or 0x1 in TCMP, or the core will run into unknown state. Note2: When timer is operating at continuous counting mode, the 24-bit up-timer will count continuously if software writes a new value into TCMP. If timer is operating at other modes, the 24-bit up-timer will restart counting and using newest TCMP value to be the compared value if software writes a new value into TCMP.

Timer Interrupt Status Register (TISR)

Register	Offset	R/W	Description	Reset Value
TISR0	TMR_BA01+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TISR1	TMR_BA01+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TISR2	TMR_BA23+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TISR3	TMR_BA23+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							**TIF**

Bits	Descriptions	
[31:1]	**Reserved**	Reserved
[0]	**TIF**	**Timer Interrupt Flag** This bit indicates the interrupt status of Timer. TIF bit is set by hardware when the up counting value of internal 24-bit timer matches the timer compared value (TCMP). It is cleared by writing 1 to this bit.

Timer Data Register (TDR)

Register	Offset	R/W	Description	Reset Value
TDR0	TMR_BA01+0x0C	R/W	Timer0 Data Register	0x0000_0000
TDR1	TMR_BA01+0x2C	R/W	Timer1 Data Register	0x0000_0000
TDR2	TMR_BA23+0x0C	R/W	Timer2 Data Register	0x0000_0000
TDR3	TMR_BA23+0x2C	R/W	Timer3 Data Register	0x0000_0000

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

Bits	Descriptions	
[31:24]	**Reserved**	Reserved
[23:0]	**TDR**	**Timer Data Register** When TCSR.TDR_EN is set to 1, the internal 24-bit up-timer value will be loaded into TDR. User can read this register for the up-timer value.

5.7 PWM Generator and Capture Timer (PWM)

5.7.1 Overview

NuMicro™ NUC122 only support 1 set of PWM group supports total 2 sets of PWM Generators which can be configured as 4 independent PWM outputs, PWM0~PWM3, or as 2 complementary PWM pairs, (PWM0, PWM1) and (PWM2, PWM3) with 2 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 down-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 which are set by hardware when the corresponding PWM period down counter reaches zero. 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), are controlled by PWM2, timer and Dead-zone generator 2. Refer to figures bellowed 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 zero. The double buffering feature avoids glitch at PWM outputs.

When the 16-bit period down counter reaches zero, the interrupt request is generated. If PWM-timer is set as auto-reload mode, when the down counter reaches zero, 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 zero.

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 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, they are: Read PIIRx to get interrupt source and Read CRLRx/CFLRx(x=0~3) to get capture value and finally write 1 to clear PIIRx to zero. 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 1/900 ns ≈ 1000 KHz

5.7.2 Features

5.7.2.1 PWM function features:

- PWM group has two PWM generators. Each PWM generator supports one 8-bit prescaler, one clock divider, two PWM-timers (down counter), one dead-zone generator and two PWM outputs.
- Up to 16 bits resolution
- PWM Interrupt request synchronized with PWM period
- One-shot or Auto-reload mode PWM
- Support 4 PWM channels or 2 PWM paired channels

5.7.2.2 Capture Function Features:

- Timing control logic shared with PWM Generators
- 4 Capture input channels shared with 4 PWM output channels
- Each channel supports one rising latch register (CRLR), one falling latch register (CFLR) and Capture interrupt flag (CAPIFx)

5.7.3 Block Diagram

The following figures illustrate the architecture of PWM in pair (PWM-Timer 0&1 are in one pair and PWM-Timer 2&3 are in another one).

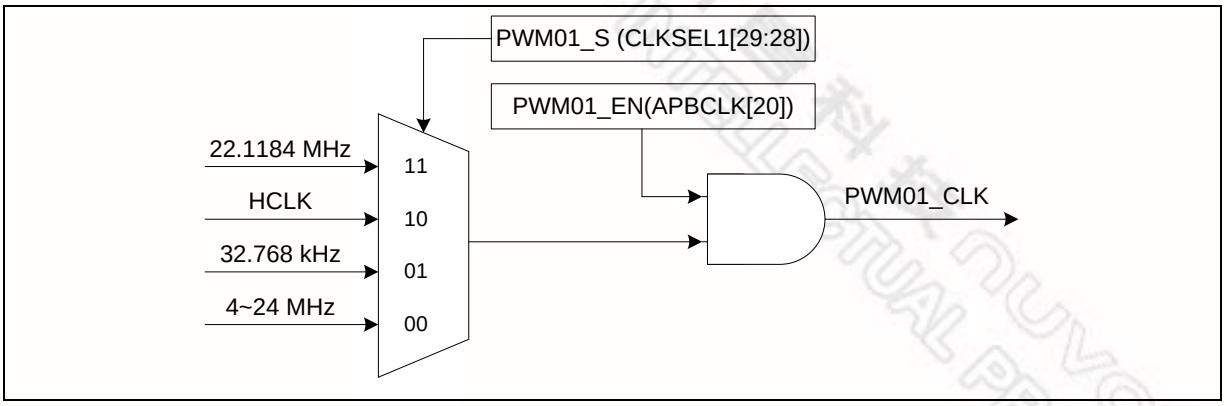


Figure 5-18 PWM Generator 0 Clock Source Control

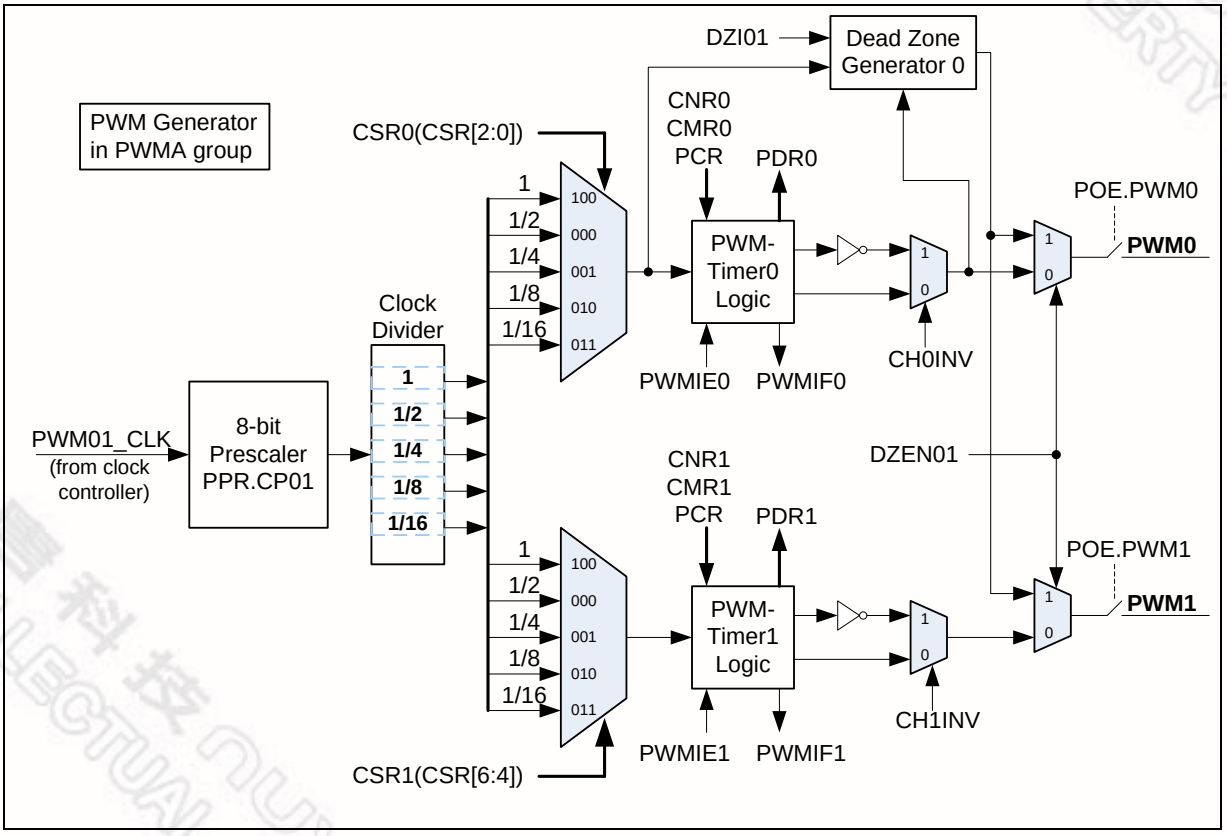


Figure 5-19 PWM Generator 0 Architecture Diagram

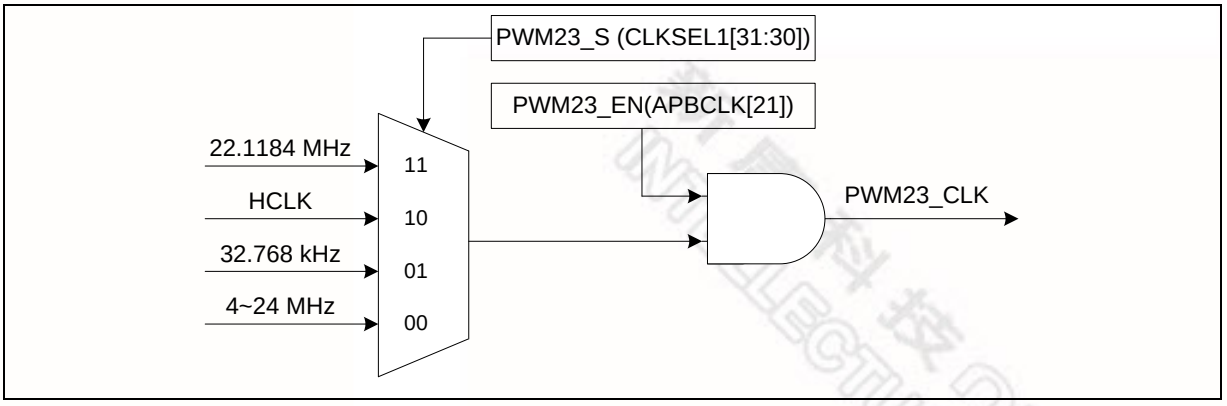


Figure 5-20 PWM Generator 2 Clock Source Control

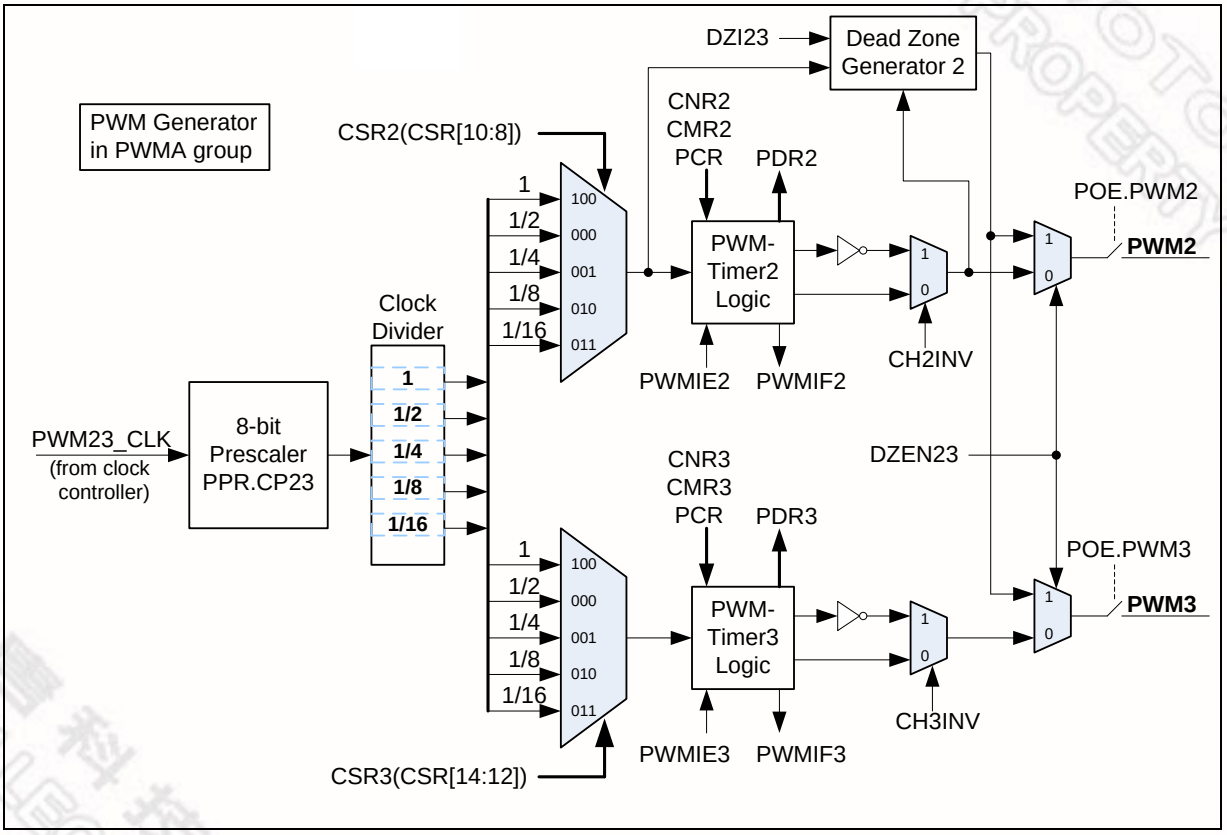


Figure 5-21 PWM Generator 2 Architecture Diagram

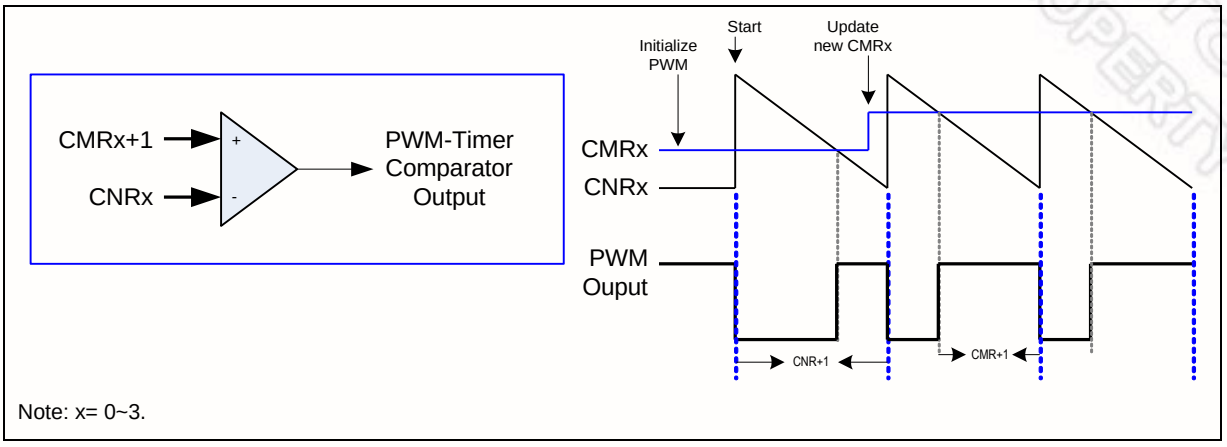
5.7.4 Function Description

5.7.4.1 PWM-Timer Operation

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 5-23. The pulse width modulation follows the formula as below and the legend of PWM-Timer Comparator is shown as Figure 5-22. 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, or 23, 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 5-22 Legend of Internal Comparator Output of PWM-Timer

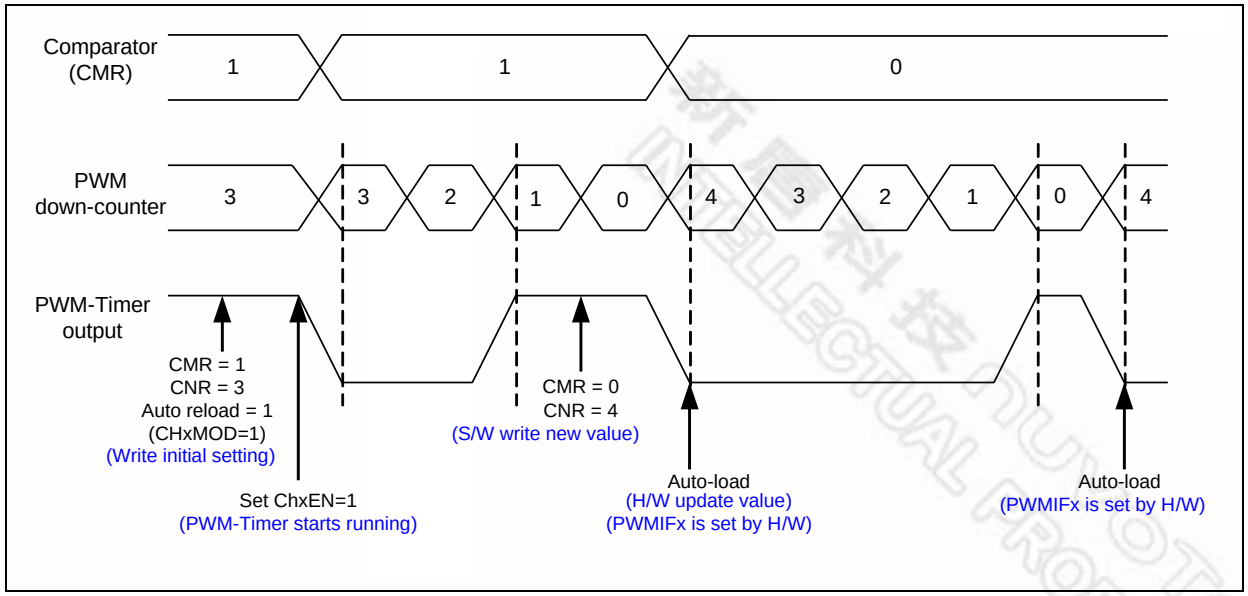


Figure 5-23 PWM-Timer Operation Timing

5.7.4.2 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.

The bit CH0MOD in PWM Control Register (PCR) defines PWM0 operates in auto-reload or one-shot mode If CH0MOD is set to one, the auto-reload operation loads CNR0 to PWM counter when PWM counter reaches zero. If CNR0 are set to zero, PWM counter will be halt when PWM counter counts to zero. If CH0MOD is set as zero, counter will be stopped immediately. PWM1~PWM3 performs the same function as PWM0.

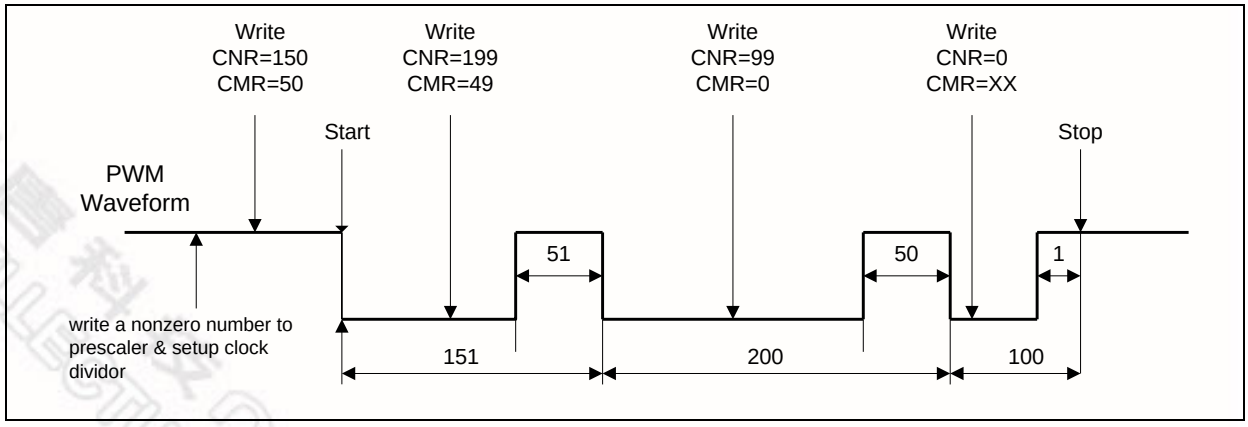


Figure 5-24 PWM Double Buffering Illustration

5.7.4.3 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.

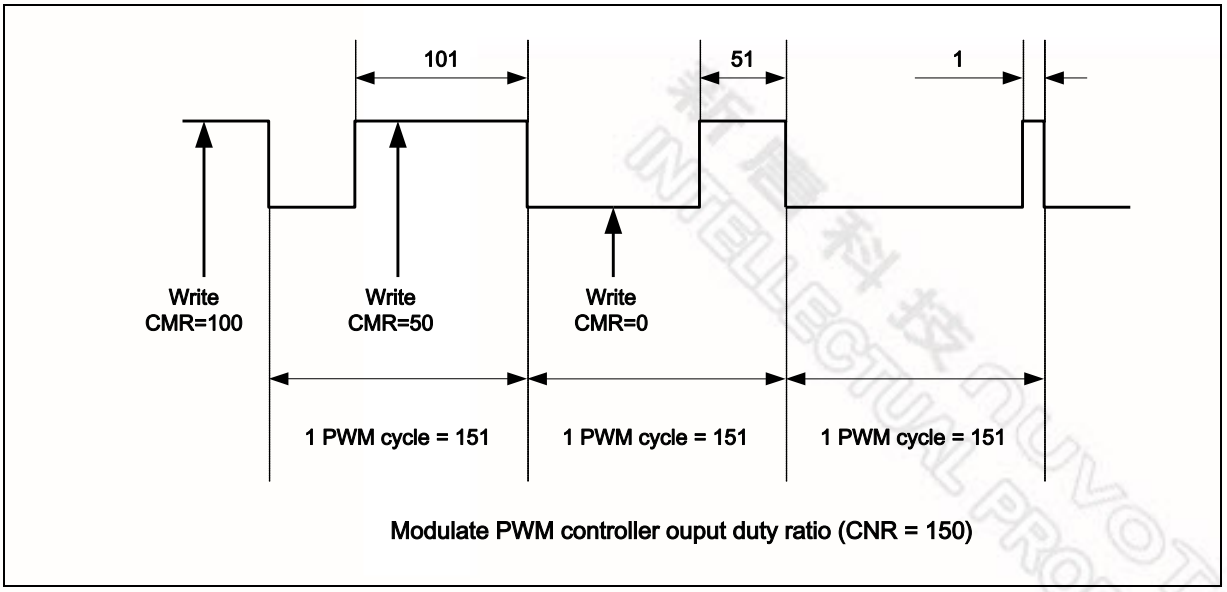


Figure 5-25 PWM Controller Output Duty Ratio

5.7.4.4 Dead-Zone Generator

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.

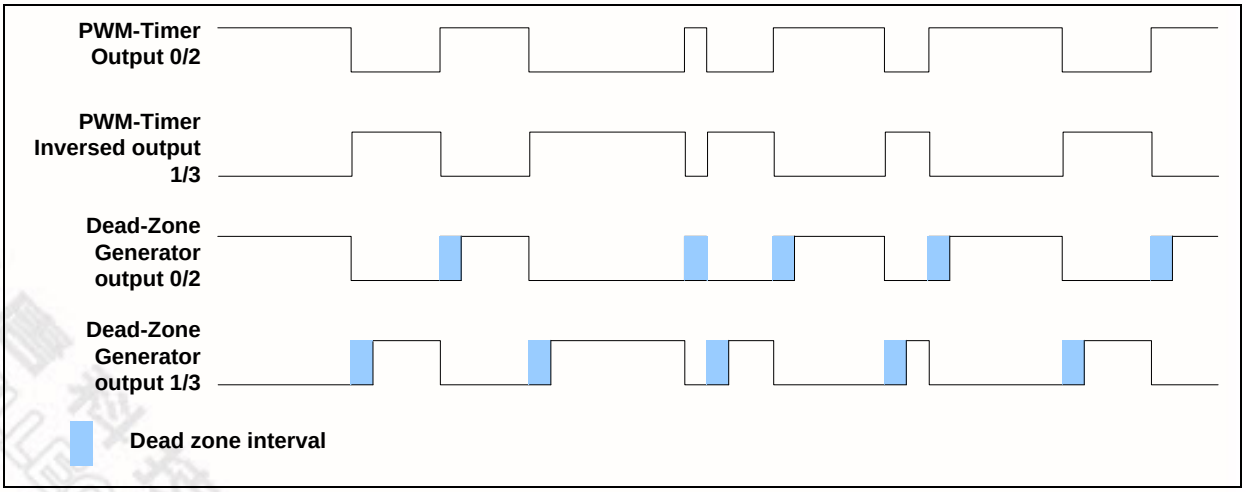


Figure 5-26 Paired-PWM Output with Dead Zone Generation Operation

5.7.4.5 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 (disable POE and enable CAPENR) for the corresponding capture channel.

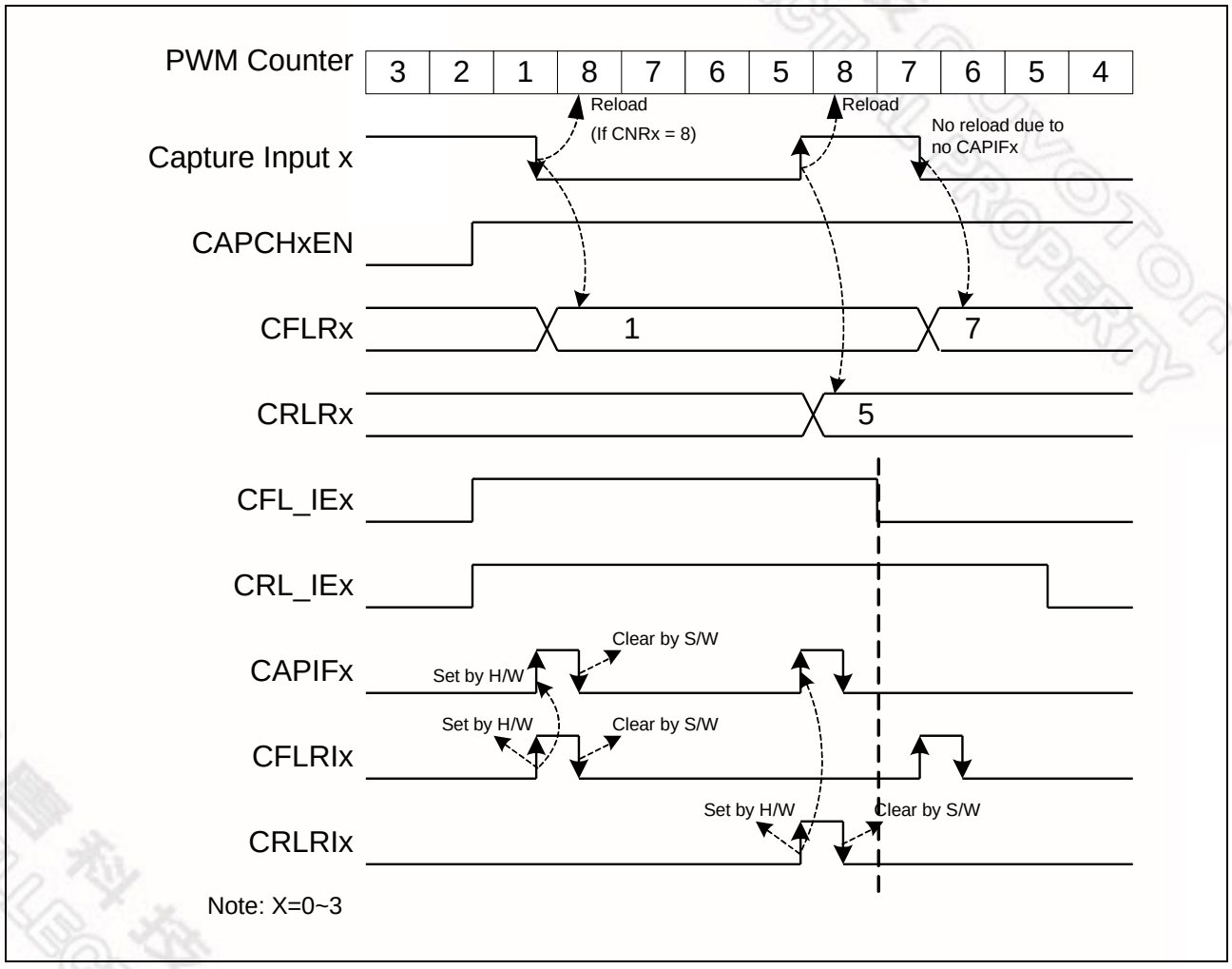


Figure 5-27 Capture Operation Timing

At 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).

5.7.4.6 PWM-Timer Interrupt Architecture

There are four PWM interrupts, PWM0_INT~PWM3_INT, which are combined into PWMA_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 5-28 demonstrates the architecture of PWM-Timer interrupts.

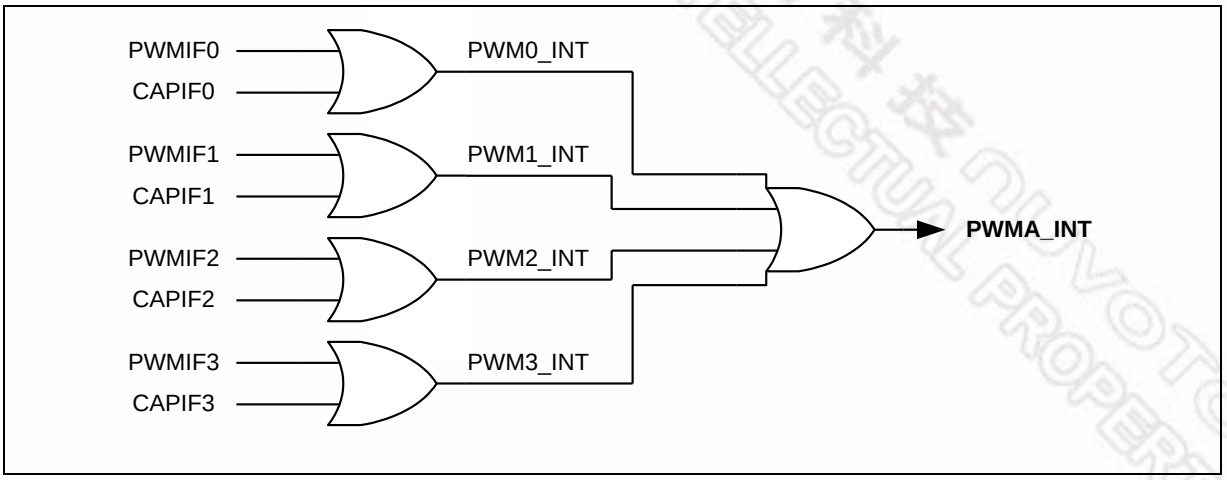


Figure 5-28 PWM Group A PWM-Timer Interrupt Architecture Diagram

5.7.4.7 PWM-Timer Start Procedure

The following procedure is recommended for starting a PWM drive.

1. Setup clock selector (CSR)
2. Setup prescaler (PPR)
3. Setup inverter on/off, dead zone generator on/off, auto-reload/one-shot mode and Stop PWM-timer (PCR)
4. Setup comparator register (CMR) for setting PWM duty.
5. Setup PWM down-counter register (CNR) for setting PWM period.
6. Setup interrupt enable register (PIER)
7. Setup corresponding GPIO pins as PWM function (enable POE and disable CAPENR) for the corresponding PWM channel.
8. Enable PWM timer start running (Set CHxEN = 1 in PCR)

5.7.4.8 PWM-Timer Stop Procedure

Method 1:

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

Method 2:

Set 16-bit down 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

5.7.4.9 Capture Start Procedure

1. Setup clock selector (CSR)
2. Setup prescaler (PPR)
3. Setup channel enabled, rising/falling interrupt enable and input signal inverter on/off (CCR0, CCR2)
4. Setup PWM down-counter (CNR)
5. Setup corresponding GPIO pins as capture function (disable POE and enable CAPENR) for the corresponding PWM channel.
6. Enable PWM timer start running (Set CHxEN = 1 in PCR)

5.7.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
PWMA_BA = 0x4004_0000 (PWM group A)				
PPR	PWMA_BA+0x00	R/W	PWM Group A Pre-scale Register	0x0000_0000
CSR	PWMA_BA+0x04	R/W	PWM Group A Clock Selection Register	0x0000_0000
PCR	PWMA_BA+0x08	R/W	PWM Group A Control Register	0x0000_0000
CNR0	PWMA_BA+0x0C	R/W	PWM Group A Counter Register 0	0x0000_0000
CMR0	PWMA_BA+0x10	R/W	PWM Group A Comparator Register 0	0x0000_0000
PDR0	PWMA_BA+0x14	R	PWM Group A Data Register 0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWM Group A Counter Register 1	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWM Group A Comparator Register 1	0x0000_0000
PDR1	PWMA_BA+0x20	R	PWM Group A Data Register 1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM Group A Counter Register 2	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM Group A Comparator Register 2	0x0000_0000
PDR2	PWMA_BA+0x2C	R	PWM Group A Data Register 2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM Group A Counter Register 3	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM Group A Comparator Register 3	0x0000_0000
PDR3	PWMA_BA+0x38	R	PWM Group A Data Register 3	0x0000_0000
PIER	PWMA_BA+0x40	R/W	PWM Group A Interrupt Enable Register	0x0000_0000
PIIR	PWMA_BA+0x44	R/W	PWM Group A Interrupt Indication Register	0x0000_0000
CCR0	PWMA_BA+0x50	R/W	PWM Group A Capture Control Register 0	0x0000_0000
CCR2	PWMA_BA+0x54	R/W	PWM Group A Capture Control Register 2	0x0000_0000
CRLR0	PWMA_BA+0x58	R/W	PWM Group A Capture Rising Latch Register (Channel 0)	0x0000_0000
CFLR0	PWMA_BA+0x5C	R/W	PWM Group A Capture Falling Latch Register (Channel 0)	0x0000_0000
CRLR1	PWMA_BA+0x60	R/W	PWM Group A Capture Rising Latch Register (Channel 1)	0x0000_0000
CFLR1	PWMA_BA+0x64	R/W	PWM Group A Capture Falling Latch Register (Channel 1)	0x0000_0000
CRLR2	PWMA_BA+0x68	R/W	PWM Group A Capture Rising Latch Register (Channel 2)	0x0000_0000
CFLR2	PWMA_BA+0x6C	R/W	PWM Group A Capture Falling Latch Register (Channel 2)	0x0000_0000
CRLR3	PWMA_BA+0x70	R/W	PWM Group A Capture Rising Latch Register (Channel 3)	0x0000_0000

CFLR3	PWMA_BA+0x74	R/W	PWM Group A Capture Falling Latch Register (Channel 3)	0x0000_0000
CAPENR	PWMA_BA+0x78	R/W	PWM Group A Capture Input 0~3 Enable Register	0x0000_0000
POE	PWMA_BA+0x7C	R/W	PWM Group A Output Enable for Channel 0~3	0x0000_0000

5.7.6 Register Description

PWM Pre-Scale Register (PPR)

Register	Offset	R/W	Description	Reset Value
PPR	PWMA_BA+0x00	R/W	PWM Group A Pre-scale 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	Descriptions	
[31:24]	**DZI23**	**Dead Zone Interval for Pair of Channel2 and Channel3** (PWM2 and PWM3 pair for PWM group A) These 8 bits determine dead zone length. The unit time of dead zone length is received from corresponding CSR bits.
[23:16]	**DZI01**	**Dead Zone Interval for Pair of Channel 0 and Channel 1** (PWM0 and PWM1 pair for PWM group A) These 8 bits determine dead zone length. The unit time of dead zone length is received from corresponding CSR bits.
[15:8]	**CP23**	**Clock Prescaler 2** (PWM-timer2 & 3 for group A) Clock input is divided by (CP23 + 1) before it is fed to the corresponding PWM-timer If CP23=0, then the clock prescaler 2 output clock will be stopped. So corresponding PWM-timer will be stopped also.
[7:0]	**CP01**	**Clock Prescaler 0** (PWM-timer 0 & 1 for group A) 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 be stopped also.

PWM Clock Selector Register (CSR)

Register	Offset	R/W	Description	Reset Value
CSR	PWMA_BA+0x04	R/W	PWM Group A Clock Selection Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	CSR3			Reserved	CSR2		
7	6	5	4	3	2	1	0
Reserved	CSR1			Reserved	CSR0		

Bits	Descriptions	
[31:15]	**Reserved**	Reserved
[14:12]	**CSR3**	**PWM Timer 3 Clock Source Selection** (PWM timer 3 for group A) Select clock input for PWM timer. CSR3 [14:12] / Input clock divided by 100 / 1 011 / 16 010 / 8 001 / 4 000 / 2
[11]	**Reserved**	Reserved
[10:8]	**CSR2**	**PWM Timer 2 Clock Source Selection** (PWM timer 2 for group A) Select clock input for PWM timer. (Table is the same as CSR3)
[7]	**Reserved**	Reserved
[6:4]	**CSR1**	**PWM Timer 1 Clock Source Selection** (PWM timer 1 for group A) Select clock input for PWM timer. (Table is the same as CSR3)
[3]	**Reserved**	Reserved
[2:0]	**CSR0**	**PWM Timer 0 Clock Source Selection** (PWM timer 0 for group A) Select clock input for PWM timer. (Table is the same as CSR3)

PWM Control Register (PCR)

Register	Offset	R/W	Description	Reset Value
PCR	PWMA_BA+0x08	R/W	PWM Group A Control Register (PCR)	0x0000_0000

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

Bits	Descriptions	
[31:28]	**Reserved**	Reserved
[27]	**CH3MOD**	**PWM-Timer 3 Auto-reload/One-Shot Mode** (PWM timer 3 for group A) 1 = Auto-reload Mode 0 = One-Shot Mode Note: If there is a transition at this bit, it will cause CNR3 and CMR3 be clear.
[26]	**CH3INV**	**PWM-Timer 3 Output Inverter Enable** (PWM timer 3 for group A) 1 = Inverter enable 0 = Inverter disable
[25]	**Reserved**	Reserved
[24]	**CH3EN**	**PWM-Timer 3 Enable** (PWM timer 3 for group A) 1 = Enable corresponding PWM-Timer Start Run 0 = Stop corresponding PWM-Timer Running
[23:20]	**Reserved**	Reserved
[19]	**CH2MOD**	**PWM-Timer 2 Auto-reload/One-Shot Mode** (PWM timer 2 for group A) 1 = Auto-reload Mode 0 = One-Shot Mode Note: If there is a transition at this bit, it will cause CNR2 and CMR2 be clear.
[18]	**CH2INV**	**PWM-Timer 2 Output Inverter Enable** (PWM timer 2 for group A) 1 = Inverter enable 0 = Inverter disable

[17]	**Reserved**	Reserved
[16]	**CH2EN**	**PWM-Timer 2 Enable** (PWM timer 2 for group A) 1 = Enable corresponding PWM-Timer Start Run 0 = Stop corresponding PWM-Timer Running
[15:12]	**Reserved**	Reserved
[11]	**CH1MOD**	**PWM-Timer 1 Auto-reload/One-Shot Mode** (PWM timer 1 for group A) 1 = Auto-load Mode 0 = One-Shot Mode Note: If there is a transition at this bit, it will cause CNR1 and CMR1 be clear.
[10]	**CH1INV**	**PWM-Timer 1 Output Inverter Enable** (PWM timer 1 for group A) 1 = Inverter enable 0 = Inverter disable
[9]	**Reserved**	Reserved
[8]	**CH1EN**	**PWM-Timer 1 Enable** (PWM timer 1 for group A) 1 = Enable corresponding PWM-Timer Start Run 0 = Stop corresponding PWM-Timer Running
[7:6]	**Reserved**	Reserved
[5]	**DZEN23**	**Dead-Zone 2 Generator Enable** (PWM2 and PWM3 pair for PWM group A) 1 = Enable 0 = Disable Note: When Dead-Zone Generator is enabled, the pair of PWM2 and PWM3 becomes a complementary pair for PWM group A.
[4]	**DZEN01**	**Dead-Zone 0 Generator Enable** (PWM0 and PWM1 pair for PWM group A) 1 = Enable 0 = Disable Note: When Dead-Zone Generator is enabled, the pair of PWM0 and PWM1 becomes a complementary pair for PWM group A
[3]	**CH0MOD**	**PWM-Timer 0 Auto-reload/One-Shot Mode** (PWM timer 0 for group A) 1 = Auto-reload Mode 0 = One-Shot Mode Note: If there is a transition at this bit, it will cause CNR0 and CMR0 be clear.
[2]	**CH0INV**	**PWM-Timer 0 Output Inverter Enable** (PWM timer 0 for group A 1 = Inverter enable 0 = Inverter disable
[1]	**Reserved**	Reserved
[0]	**CH0EN**	**PWM-Timer 0 Enable** (PWM timer 0 for group A) 1 = Enable corresponding PWM-Timer Start Run 0 = Stop corresponding PWM-Timer Running

PWM Counter Register 3-0 (CNR3-0)

Register	Offset	R/W	Description	Reset Value
CNR0	PWMA_BA+0x0C	R/W	PWM Group A Counter Register 0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWM Group A Counter Register 1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM Group A Counter Register 2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM Group A 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[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNRx[7:0]							

Bits	Descriptions	
[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, or 23, 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; PWM high width = (CMR+1) unit. ● CMR = 0: PWM low width = (CNR) unit; PWM high width = 1 unit (Unit = one PWM clock cycle) Note: Any write to CNR will take effect in next PWM cycle.

PWM Comparator Register 3-0 (CMR3-0)

Register	Offset	R/W	Description	Reset Value
CMR0	PWMA_BA+0x10	R/W	PWM Group A Comparator Register 0	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWM Group A Comparator Register 1	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM Group A Comparator Register 2	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM Group A 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[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMRx[7:0]							

Bits	Descriptions	
[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, or 23, 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; PWM high width = (CMR+1) unit. ● CMR = 0: PWM low width = (CNR) unit; PWM high width = 1 unit (Unit = one PWM clock cycle) Note: Any write to CMR 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	R	PWM Group A Data Register 0	0x0000_0000
PDR1	PWMA_BA+0x20	R	PWM Group A Data Register 1	0x0000_0000
PDR2	PWMA_BA+0x2C	R	PWM Group A Data Register 2	0x0000_0000
PDR3	PWMA_BA+0x38	R	PWM Group A 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[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PDR[7:0]							

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

PWM Interrupt Enable Register (PIER)

Register	Offset	R/W	Description	Reset Value
PIER	PWMA_BA+0x40	R/W	PWM Group A 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				PWMIE 3	PWMIE 2	PWMIE 1	PWMIE0

Bits	Descriptions	
[31:4]	**Reserved**	Reserved
[3]	**PWMIE3**	**PWM Channel 3 Interrupt Enable** 1 = Enable 0 = Disable
[2]	**PWMIE2**	**PWM Channel 2 Interrupt Enable** 1 = Enable 0 = Disable
[1]	**PWMIE1**	**PWM Channel 1 Interrupt Enable** 1 = Enable 0 = Disable
[0]	**PWMIE0**	**PWM Channel 0 Interrupt Enable** 1 = Enable 0 = Disable

PWM Interrupt Indication Register (PIIR)

Register	Offset	R/W	Description	Reset Value
PIIR	PWMA_BA+0x44	R/W	PWM Group A 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				PWMIF3	PWMIF2	PWMIF1	PWMIF0

Bits	Descriptions	
[31:4]	Reserved	Reserved
[3]	PWMIF3	**PWM Channel 3 Interrupt Status** This bit is set by hardware when PWM3 down counter reaches zero and PWM3 interrupt enable bit (PWMIE3) is 1, software can write 1 to clear this bit to zero.
[2]	PWMIF2	**PWM Channel 2 Interrupt Status** This bit is set by hardware when PWM2 down counter reaches zero and PWM2 interrupt enable bit (PWMIE2) is 1, software can write 1 to clear this bit to zero.
[1]	PWMIF1	**PWM Channel 1 Interrupt Status** This bit is set by hardware when PWM1 down counter reaches zero and PWM1 interrupt enable bit (PWMIE1) is 1, software can write 1 to clear this bit to zero.
[0]	PWMIF0	**PWM Channel 0 Interrupt Status** This bit is set by hardware when PWM0 down counter reaches zero and PWM0 interrupt enable bit (PWMIE0) is 1, software can write 1 to clear this bit to zero.

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	R/W	PWM Group A Capture Control Register	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	Descriptions	
[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. Write 1 to clear this bit to zero
[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. Write 1 to clear this bit to zero
[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). Write 1 to clear this bit to zero
[19]	**CAPCH1EN**	**Channel 1 Capture Function Enable** 1 = Enable capture function on PWM group channel 1 0 = Disable capture function on PWM group channel 1 When Enable, Capture latched the PWM-counter and saved to CRLR (Rising latch) and CFLR (Falling latch). When Disable, Capture does not update CRLR and CFLR, and disable PWM group channel 1 Interrupt.

[18]	**CFL_IE1**	**Channel 1 Falling Latch Interrupt Enable** 1 = Enable falling latch interrupt 0 = Disable falling latch interrupt When Enable, if Capture detects PWM group channel 1 has falling transition, Capture issues an Interrupt.
[17]	**CRL_IE1**	**Channel 1 Rising Latch Interrupt Enable** 1 = Enable rising latch interrupt 0 = Disable rising latch interrupt When Enable, if Capture detects PWM group channel 1 has rising transition, Capture issues an Interrupt.
[16]	**INV1**	**Channel 1 Inverter Enable** 1 = Inverter enable. Reverse the input signal from GPIO before fed to Capture timer 0 = Inverter disable
[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. Write 1 to clear this bit to zero.
[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. Write 1 to clear this bit to zero.
[5]	**Reserved**	Reserved
[4]	**CAPIF0**	**Channel 0 Capture Interrupt Indication Flag** If PWM group channel 0 rising latch interrupt is enabled (CRL_IE0=1), a rising transition occurs at PWM group channel 0 will result in CAPIF0 to high; Similarly, a falling transition will cause CAPIF0 to be set high if PWM group channel 0 falling latch interrupt is enabled (CFL_IE0=1). Write 1 to clear this bit to zero
[3]	**CAPCH0EN**	**Channel 0 Capture Function Enable** 1 = Enable capture function on PWM group channel 0. 0 = Disable capture function on PWM group channel 0 When Enable, Capture latched the PWM-counter value and saved to CRLR (Rising latch) and CFLR (Falling latch). When Disable, Capture does not update CRLR and CFLR, and disable PWM group channel 0 Interrupt.
[2]	**CFL_IE0**	**Channel 0 Falling Latch Interrupt Enable** 1 = Enable falling latch interrupt 0 = Disable falling latch interrupt When Enable, if Capture detects PWM group channel 0 has falling transition, Capture

		issues an Interrupt.
[1]	**CRL_IE0**	**Channel 0 Rising Latch Interrupt Enable** 1 = Enable rising latch interrupt 0 = Disable rising latch interrupt When Enable, if Capture detects PWM group channel 0 has rising transition, Capture issues an Interrupt.
[0]	**INV0**	**Channel 0 Inverter Enable** 1 = Inverter enable. Reverse the input signal from GPIO before fed to Capture timer 0 = Inverter disable

Capture Control Register (CCR2)

Register	Offset	R/W	Description	Reset Value
CCR2	PWMA_BA+0x54	R/W	PWM Group A Capture Control Register	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	Descriptions	
[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. Write 1 to clear this bit to zero.
[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. Write 1 to clear this bit to zero.
[21]	**Reserved**	Reserved
[20]	**CAPIF3**	**Channel 3 Capture Interrupt Indication Flag** If PWM group channel 3 rising latch interrupt is enabled (CRL_IE3=1), a rising transition occurs at PWM group channel 3 will result in CAPIF3 to high; Similarly, a falling transition will cause CAPIF3 to be set high if PWM group channel 3 falling latch interrupt is enabled (CFL_IE3=1). Write 1 to clear this bit to zero
[19]	**CAPCH3EN**	**Channel 3 Capture Function Enable** 1 = Enable capture function on PWM group channel 3 0 = Disable capture function on PWM group channel 3 When Enable, Capture latched the PWM-counter and saved to CRLR (Rising latch) and CFLR (Falling latch). When Disable, Capture does not update CRLR and CFLR, and disable PWM group channel 3 Interrupt.

[18]	**CFL_IE3**	**Channel 3 Falling Latch Interrupt Enable** 1 = Enable falling latch interrupt 0 = Disable falling latch interrupt When Enable, if Capture detects PWM group channel 3 has falling transition, Capture issues an Interrupt.
[17]	**CRL_IE3**	**Channel 3 Rising Latch Interrupt Enable** 1 = Enable rising latch interrupt 0 = Disable rising latch interrupt When Enable, if Capture detects PWM group channel 3 has rising transition, Capture issues an Interrupt.
[16]	**INV3**	**Channel 3 Inverter Enable** 1 = Inverter enable. Reverse the input signal from GPIO before fed to Capture timer 0 = Inverter disable
[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. Write 1 to clear this bit to zero.
[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. Write 1 to clear this bit to zero.
[5]	**Reserved**	Reserved
[4]	**CAPIF2**	**Channel 2 Capture Interrupt Indication Flag** If PWM group channel 2 rising latch interrupt is enabled (CRL_IE2=1), a rising transition occurs at PWM group channel 2 will result in CAPIF2 to high; Similarly, a falling transition will cause CAPIF2 to be set high if PWM group channel 2 falling latch interrupt is enabled (CFL_IE2=1). Write 1 to clear this bit to zero
[3]	**CAPCH2EN**	**Channel 2 Capture Function Enable** 1 = Enable capture function on PWM group channel 2 0 = Disable capture function on PWM group channel 2 When Enable, Capture latched the PWM-counter value and saved to CRLR (Rising latch) and CFLR (Falling latch). When Disable, Capture does not update CRLR and CFLR, and disable PWM group channel 2 Interrupt.
[2]	**CFL_IE2**	**Channel 2 Falling Latch Interrupt Enable** 1 = Enable falling latch interrupt 0 = Disable falling latch interrupt When Enable, if Capture detects PWM group channel 2 has falling transition, Capture

		issues an Interrupt.
[1]	**CRL_IE2**	**Channel 2 Rising Latch Interrupt Enable** 1 = Enable rising latch interrupt 0 = Disable rising latch interrupt When Enable, if Capture detects PWM group channel 2 has rising transition, Capture issues an Interrupt.
[0]	**INV2**	**Channel 2 Inverter Enable** 1 = Inverter enable. Reverse the input signal from GPIO before fed to Capture timer 0 = Inverter disable

Capture Rising Latch Register3-0 (CRLR3-0)

Register	Offset	R/W	Description	Reset Value
CRLR0	PWMA_BA+0x58	R	PWM Group A Capture Rising Latch Register (Channel 0)	0x0000_0000
CRLR1	PWMA_BA+0x60	R	PWM Group A Capture Rising Latch Register (Channel 1)	0x0000_0000
CRLR2	PWMA_BA+0x68	R	PWM Group A Capture Rising Latch Register (Channel 2)	0x0000_0000
CRLR3	PWMA_BA+0x70	R	PWM Group A Capture Rising Latch Register (Channel 3)	0x0000_0000

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

Bits	Descriptions	
[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	R	PWM Group A Capture Falling Latch Register (Channel 0)	0x0000_0000
CFLR1	PWMA_BA+0x64	R	PWM Group A Capture Falling Latch Register (Channel 1)	0x0000_0000
CFLR2	PWMA_BA+0x6C	R	PWM Group A Capture Falling Latch Register (Channel 2)	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM Group A 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[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CFLRx[7:0]							

Bits	Descriptions	
[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	R/W	PWM Group A 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				CAPENR			

Bits	Descriptions	
[31:4]	**Reserved**	Reserved
[3:0]	**CAPENR**	**Capture Input Enable Register** There are four capture inputs from pad. Bit0~Bit3 are used to control each input enable or disable. 0 = Disable (PWMx multi-function pin input does not affect input capture function.) 1 = Enable (PWMx multi-function pin input will affect its input capture function.) CAPENR Bit 3210 for PWM group A Bit xxx1 ➔ Capture channel 0 is from pin PA.12 Bit xx1x ➔ Capture channel 1 is from pin PA.13 Bit x1xx ➔ Capture channel 2 is from pin PA.14 Bit 1xxx ➔ Capture channel 3 is from pin PA.15

PWM Output Enable Register (POE)

Register	Offset	R/W	Description	Reset Value
POE	PWMA_BA+0x7C	R/W	PWM Group A Output Enable Register 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				**PWM3**	**PWM2**	**PWM1**	**PWM0**

Bits	Descriptions	
[31:4]	**Reserved**	Reserved
[3]	**PWM3**	**Channel 3 Output Enable** 1 = Enable PWM channel 3 output to pin 0 = Disable PWM channel 3 output to pin Note: The corresponding GPIO pin also must be switched to PWM function
[2]	**PWM2**	**Channel 2 Output Enable** 1 = Enable PWM channel 2 output to pin 0 = Disable PWM channel 2 output to pin Note: The corresponding GPIO pin also must be switched to PWM function
[1]	**PWM1**	**Channel 1 Output Enable** 1 = Enable PWM channel 1 output to pin 0 = Disable PWM channel 1 output to pin Note: The corresponding GPIO pin also must be switched to PWM function
[0]	**PWM0**	**Channel 0 Output Enable** 1 = Enable PWM channel 0 output to pin 0 = Disable PWM channel 0 output to pin Note: The corresponding GPIO pin also must be switched to PWM function

5.8 Watchdog Timer (WDT)

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 another function to wake-up chip from power down mode. The Watchdog Timer includes an 18-bit free running counter with programmable time-out intervals. Table 5-7 show the Watchdog Timer time-out interval selection and Figure 5-29 shows the timing of Watchdog interrupt signal and reset signal.

Setting WTE (WDTCR [7]) enables the watchdog timer and the WDT counter starts counting up. When the counter reaches the selected time-out interval, Watchdog timer interrupt flag WTIF will be set immediately to request a WDT interrupt if the watchdog timer interrupt enable bit WTIE is set, in the meanwhile, a specified delay time (1024 * T_{WDT}) follows the time-out event. User must set WTR (WDTCR [0]) (Watchdog timer reset) high to reset the 18-bit WDT counter to avoid chip from Watchdog timer reset before the delay time expires. WTR bit is cleared automatically by hardware after WDT counter is reset. There are eight time-out intervals with specific delay time which are selected by Watchdog timer interval select bits WTIS (WDTCR [10:8]). If the WDT counter has not been cleared after the specific delay time expires, the watchdog timer will set Watchdog Timer Reset Flag (WTRF) high and reset chip. This reset will last 63 WDT clocks (T_{RST}) then chip restarts executing program from reset vector (0x0000_0000). WTRF will not be cleared by Watchdog reset. User may poll WTRF by software to recognize the reset source. WDT also provides wake-up function. When chip is powered down and the Watchdog Timer Wake-up Function Enable bit (WDTR[4]) is set, if the WDT counter reaches the specific time interval defined by WTIS (WDTCR [10:8]) , the chip is woken-up from power down state. First example, if WTIS is set as 000, the specific time interval for chip to be woken-up from power down state is 2^4 * T_{WDT}. When power down command is set by software, then, chip enters power down state. After 2^4 * T_{WDT} time is elapsed, chip is woken-up from power down state. Second example, if WTIS (WDTCR [10:8]) is set as 111, the specific time interval for chip to be woken-up from power down state is 2^{18} * T_{WDT}. If power down command is set by software, then, chip enters power down state. After 2^{18} * T_{WDT} time is elapsed, chip is woken-up from power down state. Notice if WTRE (WDTCR [1]) is set to 1, after chip is woken-up, software should clear the Watchdog Timer counter by setting WTR(WDTCR [0]) to 1 as soon as possible. Otherwise, if the Watchdog Timer counter is not cleared by setting WTR (WDTCR [0]) to 1 before time starting from waking up to software clearing Watchdog Timer counter is over 1024 * T_{WDT} , the chip is reset by Watchdog Timer.

WTIS	Time-out Interval Selection T_{TIS}	Interrupt Period T_{INT}	WTR Time-out Interval (WDT_CLK=10 KHz) Min. T_{WTR} ~ Max. T_{WTR}
000	2^4 * T_{WDT}	1024 * T_{WDT}	1.6 ms ~ 104 ms
001	2^6 * T_{WDT}	1024 * T_{WDT}	6.4 ms ~ 108.8 ms
010	2^8 * T_{WDT}	1024 * T_{WDT}	25.6 ms ~ 128 ms
011	2^{10} * T_{WDT}	1024 * T_{WDT}	102.4 ms ~ 204.8 ms
100	2^{12} * T_{WDT}	1024 * T_{WDT}	409.6 ms ~ 512 ms
101	2^{14} * T_{WDT}	1024 * T_{WDT}	1.6384 s ~ 1.7408 s
110	2^{16} * T_{WDT}	1024 * T_{WDT}	6.5536 s ~ 6.656 s
111	2^{18} * T_{WDT}	1024 * T_{WDT}	26.2144 s ~ 26.3168 s

Table 5-7 Watchdog Timer Time-out Interval Selection

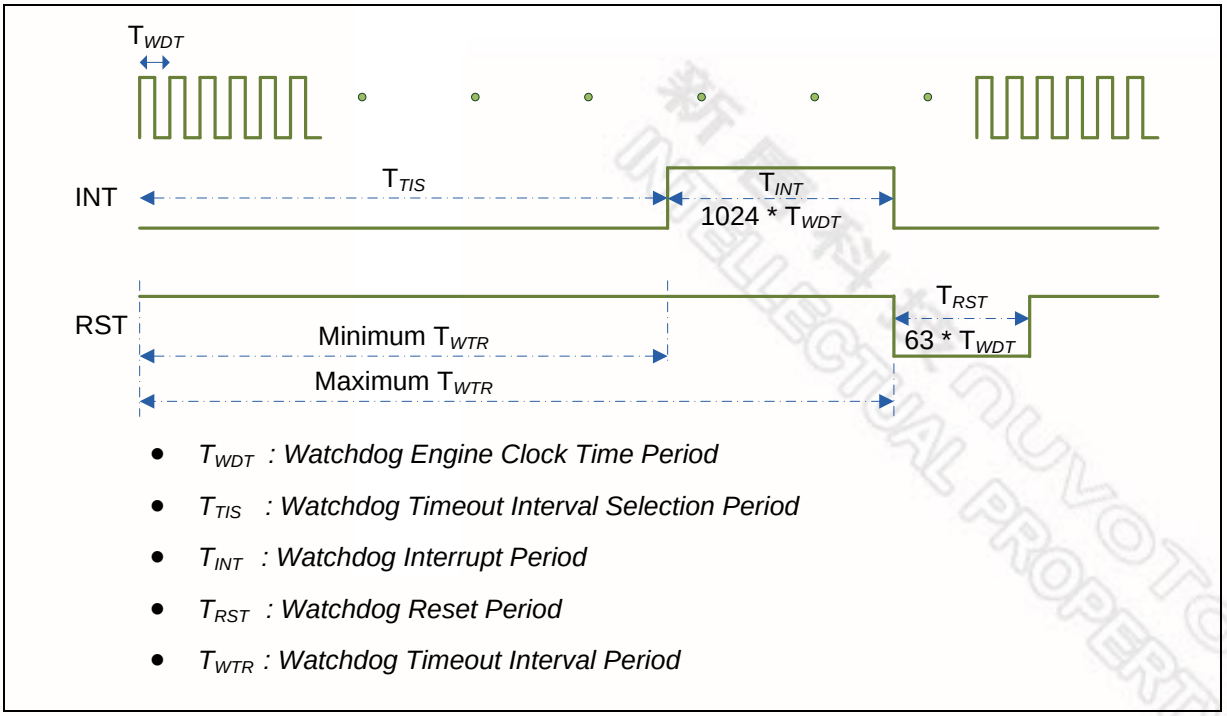


Figure 5-29 Timing of Interrupt and Reset Signals

5.8.1 Features

- 18-bit free running counter to avoid chip from Watchdog Timer reset before the delay time expires.
- Selectable time-out interval (2^4 ~ 2^18) and the time-out interval is 104 ms ~ 26.3168 s (if WDT_CLK = 10 KHz).
- Reset period = (1 / 10 KHz) * 63, if WDT_CLK = 10 KHz.

5.8.2 Block Diagram

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

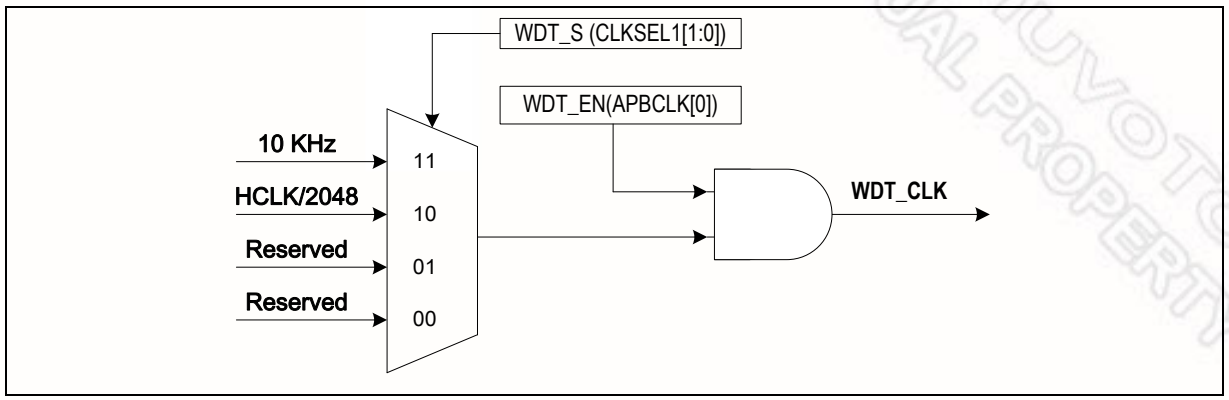


Figure 5-30 Watchdog Timer Clock Source Diagram

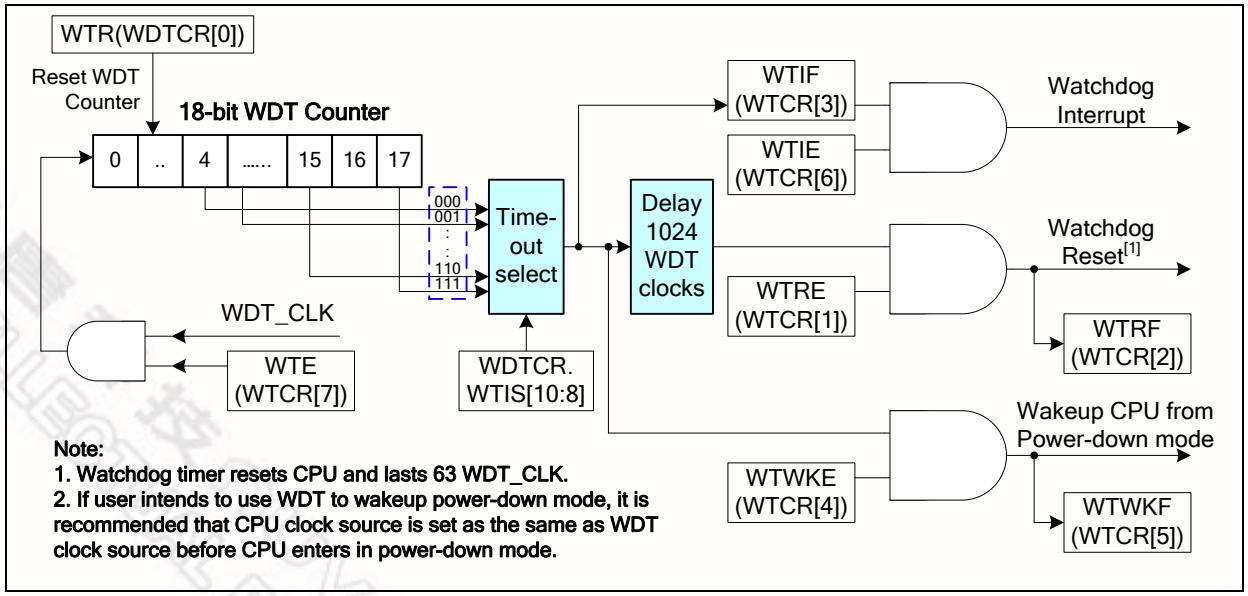


Figure 5-31 Watchdog Timer Block Diagram

5.8.3 Register Map

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

Register	Offset	R/W	Description	Reset Value
WDT_BA = 0x4000_4000				
WTCR	WDT_BA+0x00	R/W	Watchdog Timer Control Register	0x0000_0700

5.8.4 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 can be write in this register are write-protected. To program it needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Reference 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	Descriptions	
[31]	DBGACK_WDT	**ICE Debug Mode Acknowledge Disable** (write-protection bit) 0 = ICE debug mode acknowledgement effects Watchdog Timer counting. Watchdog Timer counter will be held while ICE debug mode acknowledged. 1 = ICE debug mode acknowledgement disabled. **Watchdog Timer counter will keep going no matter ICE debug mode acknowledged or not.**
[30:11]	**Reserved**	Reserved
[10:8]	**WTIS**	**Watchdog Timer Interval Select** (write-protection bits) These three bits select the time-out interval for the Watchdog Timer.

WTIS	Time-out Interval Selection	Interrupt Period	WTR Time-out Interval (WDT_CLK=10 KHz)
000	2^4 * T_{WDT}	1024 * T_{WDT}	1.6 ms ~ 104 ms
001	2^6 * T_{WDT}	1024 * T_{WDT}	6.4 ms ~ 108.8 ms
010	2^8 * T_{WDT}	1024 * T_{WDT}	25.6 ms ~ 128 ms
011	2^{10} * T_{WDT}	1024 * T_{WDT}	102.4 ms ~ 204.8 ms
100	2^{12} * T_{WDT}	1024 * T_{WDT}	409.6 ms ~ 512 ms

		101 | 2^{14} * T_{WDT} | 1024 * T_{WDT} | 1.6384 s ~ 1.7408 s 110 | 2^{16} * T_{WDT} | 1024 * T_{WDT} | 6.5536 s ~ 6.656 s 111 | 2^{18} * T_{WDT} | 1024 * T_{WDT} | 26.2144 s ~ 26.3168 s
[7]	**WTE**	**Watchdog Timer Enable** (write-protection bit) 0 = Disable the Watchdog Timer (This action will reset the internal counter) 1 = Enable the Watchdog Timer
[6]	**WTIE**	**Watchdog Timer Interrupt Enable** (write-protection bit) 0 = Disable the Watchdog Timer interrupt 1 = Enable the Watchdog Timer interrupt
[5]	**WTWKF**	**Watchdog Timer Wake-Up Flag** If Watchdog Timer causes chip be woken-up from power down mode, this bit will be set to high. It must be cleared by software with a write 1 to this bit. 0 = Watchdog Timer does not cause chip be woken-up. 1 = Chip be woken-up from idle or power down mode by Watchdog Timer time-out.
[4]	**WTWKE**	**Watchdog Timer Wake-Up Function Enable bit** (write-protection bit) 0 = Disable Watchdog Timer wake-up chip function. 1 = Enable the wake-up function that Watchdog Timer time-out can wake-up chip from power down mode. Note: Chip can be woken-up by WDT only when WDT clock source is selected from RC10K.
[3]	**WTIF**	**Watchdog Timer Interrupt Flag** If the Watchdog Timer interrupt is enabled, then the hardware will set this bit to indicate that the Watchdog Timer interrupt has occurred. 0 = Watchdog Timer interrupt did not occur 1 = Watchdog Timer interrupt occurs Note: This bit is cleared by writing 1 to this bit.
[2]	**WTRF**	**Watchdog Timer Reset Flag** When the Watchdog Timer initiates a reset, the hardware will set this bit. This flag can be read by software to determine the source of reset. Software is responsible to clear it manually by writing 1 to it. If WTRE is disabled, then the Watchdog Timer has no effect on this bit. 0 = Watchdog Timer reset did not occur 1 = Watchdog Timer reset occurs Note: This bit is cleared by writing 1 to this bit.
[1]	**WTRE**	**Watchdog Timer Reset Enable** (write-protection bit) Setting this bit will enable the Watchdog timer reset function. 0 = Disable Watchdog Timer reset function 1 = Enable Watchdog Timer reset function
[0]	**WTR**	**Clear Watchdog Timer** (write-protection bit)

		Set this bit will clear the Watchdog Timer. 0 = Writing 0 to this bit has no effect 1 = Reset the contents of the Watchdog Timer Note: This bit will be auto cleared by hardware.

5.9 Real Time Clock (RTC)

5.9.1 Overview

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

The RTC controller supports periodic Time Tick and Alarm Match interrupts. The periodic interrupt has 8 period 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]). When RTC counter in TLR and CLR is equal to alarm setting time registers TAR and CAR, the alarm interrupt flag (RIIR.AIF) is set and the alarm interrupt is requested if the alarm interrupt is enabled (RIER.AIER=1). Both RTC Time Tick and Alarm Match can cause chip be woken-up from power down mode if wake-up function is enabled (TWKE (TTR[3])=1).

5.9.2 Features

- There is a time counter (second, minute, hour) and calendar counter (day, month, year) for user to check the time
- Alarm register (second, minute, hour, day, month, year)
- 12-hour or 24-hour mode is selectable
- Leap year compensation automatically
- Day of week counter
- Frequency compensate register (FCR)
- All time and calendar message is expressed in BCD code
- Support 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
- Support RTC Time Tick and Alarm Match interrupt
- Support wake-up chip from power down mode

5.9.3 Block Diagram

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

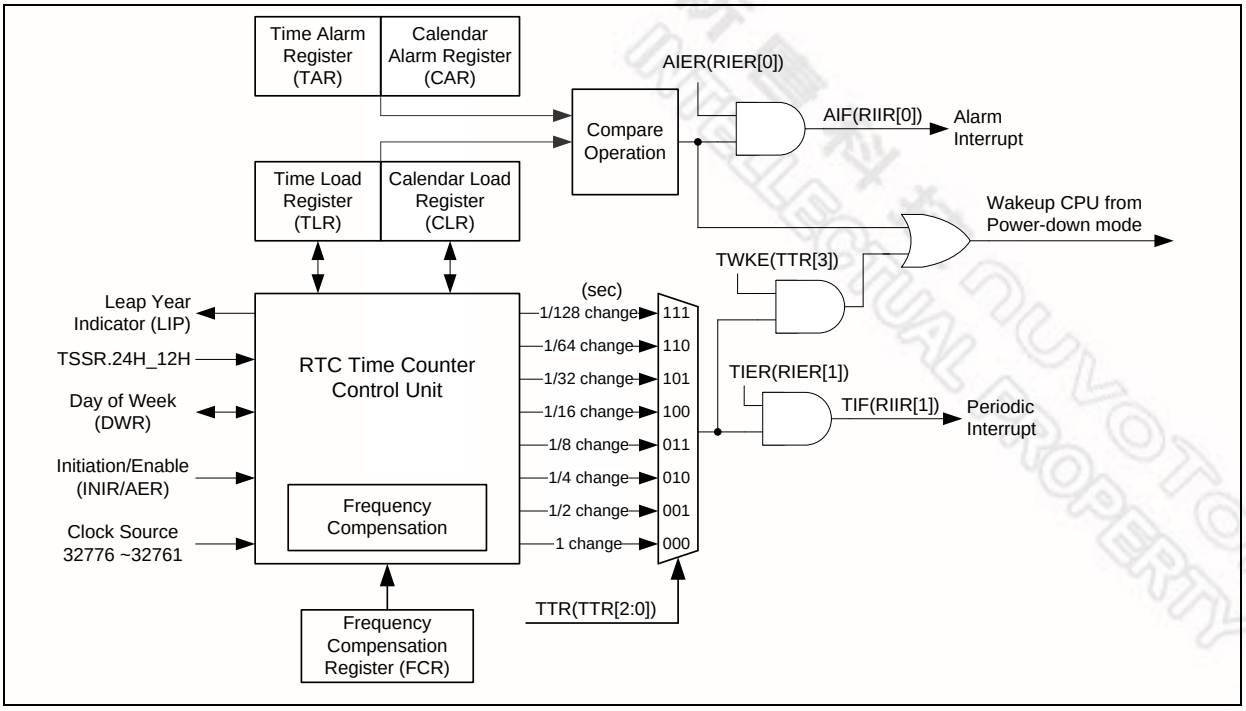


Figure 5-32 RTC Block Diagram

5.9.4 Function Description

5.9.4.1 Access to RTC register

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

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

5.9.4.2 RTC Initiation

When RTC block is power on, RTC is at reset state. User has to write a number (0xa5eb1357) to INIR to make RTC leaving reset state. Once the INIR is written as 0xa5eb1357, the RTC will be in un-reset state permanently.

5.9.4.3 RTC Read/Write Enable

Register AER bit 15~0 is served as RTC read/write password to protect RTC registers. AER bit 15~0 has to be set as 0xA965 to enable access restriction. Once it is set, it will take effect at least 512 RTC clocks (about 15ms). Programmer can read RTC enabled status flag in AER.ENF to check whether if RTC controller starts operating or not.

5.9.4.4 Frequency Compensation

The RTC FCR allows software to make digital compensation to a clock input. The frequency of clock input must be in the range from 32776 Hz to 32761 Hz. User can utilize a frequency counter to measure RTC clock on one of GPIO pin during manufacture, and store the value in Flash memory for retrieval when the product is first power on. Following are the compensation examples for higher or lower frequency clock input.

Example 1:

Frequency counter measurement : 32773.65 Hz (> 32768 Hz)

Integer part: 32773 => 0x8005

FCR.Integer = 0x05 – 0x01 + 0x08 = 0x0c

Fraction part: 0.65 x 60 = 39 => 0x27

FCR.Fraction = 0x27

Example 2

Frequency counter measurement : 32765.27 Hz (≦ 32768 Hz)

Integer part: 32765 => 0x7FFD

FCR.Integer = 0x0A – 0x01 – 0x08 = 0x04

Fraction part: 0.27 x 60 = 16.2=> 0x10

FCR.Fraction = 0x10

5.9.4.5 Time and Calendar counter

TLR and CLR are used to load the time and calendar. TAR and CAR are used for alarm. They are all represented by BCD.

5.9.4.6 12/24 hour Time Scale Selection

The 12/24 hour time scale selection depends on TSSR bit 0.

5.9.4.7 Day of the week counter

The RTC controller provides day of week in Day of the Week Register (DWR). The value is defined

from 0 to 6 to represent Sunday to Saturday respectively.

5.9.4.8 Periodic Time Tick Interrupt

The periodic interrupt has 8 period option 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]. When periodic time tick interrupt is enabled by setting RIER.TIER to 1, the Periodic Time Tick Interrupt is requested periodically in the period selected by TTR register.

5.9.4.9 Alarm interrupt

When RTC counter in TLR and CLR is equal to alarm setting time TAR and CAR the alarm interrupt flag (RIIR.AIF) is set and the alarm interrupt is requested if the alarm interrupt is enabled (RIER.AIER=1).

5.9.4.10 Application note:

1. TAR, CAR, TLR and CLR registers are all BCD counter.
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. Reset state :

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 hr mode)
DWR	6 (Saturday)
RIER	0
RIIR	0
LIR	0
TTR	0

4. In TLR and TAR, only 2 BCD digits are used to express “year”. We assume 2 BCD digits of xY denote 20xY, but not 19xY or 21xY.

5.9.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
RTC_BA = 0x4000_8000				
INIR	RTC_BA+0x00	R/W	RTC Initiation Register	0x0000_0000
AER	RTC_BA+0x04	R/W	RTC Access Enable Register	0x0000_0000
FCR	RTC_BA+0x08	R/W	RTC Frequency Compensation Register	0x0000_0700
TLR	RTC_BA+0x0C	R/W	Time Loading Register	0x0000_0000
CLR	RTC_BA+0x10	R/W	Calendar Loading Register	0x0005_0101
TSSR	RTC_BA+0x14	R/W	Time Scale Selection Register	0x0000_0001
DWR	RTC_BA+0x18	R/W	Day of the Week Register	0x0000_0006
TAR	RTC_BA+0x1C	R/W	Time Alarm Register	0x0000_0000
CAR	RTC_BA+0x20	R/W	Calendar Alarm Register	0x0000_0000
LIR	RTC_BA+0x24	R	Leap Year Indicator Register	0x0000_0000
RIER	RTC_BA+0x28	R/W	RTC Interrupt Enable Register	0x0000_0000
RIIR	RTC_BA+0x2C	R/W	RTC Interrupt Indicator Register	0x0000_0000
TTR	RTC_BA+0x30	R/W	RTC Time Tick Register	0x0000_0000

5.9.6 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							**INIR/Active**

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

RTC Access Enable Register (AER)

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

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

Bits	Descriptions	
[31:17]	**Reserved**	Reserved
[16]	**ENF**	**RTC Register Access Enable Flag (Read only)** 1 = RTC register read/write enable 0 = RTC register read/write disable This bit will be set after AER[15:0] register is load a 0xA965, and be clear automatically 512 RTC clock or AER[15:0] is not 0xA965.

Register \ AER.ENF	1	0
INIR	R/W	R/W
AER	R/W	R/W
FCR	R/W	-
TLR	R/W	R
CLR	R/W	R
TSSR	R/W	R/W
DWR	R/W	R
TAR	R/W	-
CAR	R/W	-
LIR	R	R
RIER	R/W	R/W
RIIR	R/W	R/W

		TTR	R/W	-	
[15:0]	**AER**	**RTC Register Access Enable Password (Write only)** 0xA965 = Enable RTC access Others = Disable RTC access			

RTC Frequency Compensation Register (FCR)

Register	Offset	R/W	Description	Reset Value
FCR	RTC_BA+0x08	R/W	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	Descriptions	
[31:12]	**Reserved**	Reserved
[11:8]	**INTEGER**	**Integer Part** 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
[7:6]	**Reserved**	Reserved
[5:0]	**FRACTION**	**Fraction Part** Formula = (fraction part of detected value) x 60 Note: Digit in FCR must be expressed as hexadecimal number. Refer to 5.9.4.4 for the examples.

Note: This register can be read back after the RTC register access enable bit ENF (AER[16]) is active.

RTC Time Loading Register (TLR)

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

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

Bits	Descriptions	
[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 will not check loaded data.
2. The reasonable value range is listed in the parenthesis.

RTC Calendar Loading Register (CLR)

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

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

Bits	Descriptions	
[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 loaded data.
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	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							24hr/12hr

Bits	Descriptions	
[31:1]	**Reserved**	Reserved
[0]	**24hr/12hr**	**24-Hour / 12-Hour Mode Selection** It indicate that TLR and TAR are in 24-hour mode or 12-hour mode 1 = select 24-hour time scale 0 = select 12-hour time scale with AM and PM indication

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)

RTC Day of the Week Register (DWR)

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

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

Bits	Descriptions	
[31:3]	**Reserved**	Reserved
[2:0]	**DWR**	**Day of the Week Register** Value / Day of the Week 0 Sunday 1 Monday 2 Tuesday 3 Wednesday 4 Thursday 5 Friday 6 Saturday

RTC Time Alarm Register (TAR)

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

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

Bits	Descriptions	
[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. TAR is a BCD digit counter and RTC will not check loaded data.
2. The reasonable value range is listed in the parenthesis.
3. This register can be read back after the RTC register access enable bit ENF (AER[16]) is active.

RTC Calendar Alarm Register (CAR)

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

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

Bits	Descriptions	
[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. CAR is a BCD digit counter and RTC will not check loaded data.
2. The reasonable value range is listed in the parenthesis.
3. This register can be read back after the RTC register access enable bit ENF (AER[16]) is active.

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	Descriptions	
[31:1]	**Reserved**	Reserved
[0]	**LIR**	**Leap Year Indication REGISTER** (Real only). 1 = It indicate that this year is leap year 0 = It indicate that this year is not 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	Descriptions	
[31:2]	**Reserved**	Reserved
[1]	**TIER**	**Time Tick Interrupt Enable** 1 = RTC Time Tick Interrupt is enabled 0 = RTC Time Tick Interrupt is disabled
[0]	**AIER**	**Alarm Interrupt Enable** 1 = RTC Alarm Interrupt is enabled 0 = RTC Alarm Interrupt is disabled

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	Descriptions	
[31:2]	**Reserved**	Reserved
[1]	**TIF**	**RTC Time Tick Interrupt Flag** When RTC Time Tick Interrupt is enabled (RIER.TIER=1), RTC controller will set TIF to high periodically in the period selected by TTR[2:0]. This bit is software clear by writing 1 to it. 1= Indicates RTC Time Tick Interrupt is requested if RIER.TIER=1 0= Indicates RTC Time Tick Interrupt condition never occurred.
[0]	**AIF**	**RTC Alarm Interrupt Flag** When RTC Alarm Interrupt is enabled (RIER.AIER=1), RTC controller will set AIF to high once the RTC real time counters TLR and CLR reach the alarm setting time registers TAR and CAR. This bit is software clear by writing 1 to it. 1= Indicates RTC Alarm Interrupt is requested if RIER.AIER=1 0= Indicates RTC Alarm Interrupt condition never occurred.

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				TWKE	TTR[2:0]		

Bits	Descriptions	
[31:4]	**Reserved**	Reserved
[3]	**TWKE**	**RTC Timer Wake-Up Function Enable Bit** If TWKE is set before chip is in power down mode, chip will be woken-up by RTC controller when a RTC Time Tick occurs, The chip can also be woken-up by alarm match occur. 1 = Enable RTC Timer wake-up function that chip can be woken-up from power down mode by Time Tick or Alarm Match. 0 = Disable RTC Timer wake-up function. Note: Tick timer setting follows TTR[2:0] description.
[2:0]	**TTR**	**Time Tick Register** The RTC time tick period for Periodic Time Tick Interrupt request. TTR[2:0] / Time tick (second) 0 / 1 1 / 1/2 2 / 1/4 3 / 1/8 4 / 1/16 5 / 1/32 6 / 1/64 7 / 1/128 Note: This register can be read back after the RTC register access enable bit ENF (AER[16]) is active.

5.10 UART Interface Controller (UART)

NuMicro™ NUC122 provides two channels of Universal Asynchronous Receiver/Transmitter (UART0/1). UART0 and UART1 perform Normal Speed UART, besides, UART0 and UART1 also support flow control function.

5.10.1 Overview

The Universal Asynchronous Receiver/Transmitter (UART0/1) performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the CPU. The UART controller also supports IrDA SIR Function and RS-485 mode functions. Each UART channel supports seven types of interrupts including 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), receiver buffer time-out interrupt (INT_TOUT), MODEM/Wake-Up status interrupt (INT_MODEM), Buffer error interrupt (INT_BUF_ERR). Interrupt number 13 (vector number is 29) supports UART0/1 interrupt. Refer to Nested Vectored Interrupt Controller chapter for System Interrupt Map.

The UART0/1 are equipped 14-byte transmitter FIFO (TX_FIFO) and 14-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, framing 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). Below table lists the equations in the various conditions and the UART baud rate setting table.

Mode	DIV_X_EN	DIV_X_ONE	Divider X	BRD	Baud rate equation
0	0	0	B	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 5-8 UART Baud Rate Equation

System clock = 22.1184 MHz high speed			
Baud rate	Mode0	Mode1	Mode2
921600	x	A=0,B=11	A=22
460800	A=1	A=1,B=15 A=2,B=11	A=46
230400	A=4	A=4,B=15 A=6,B=11	A=94
115200	A=10	A=10,B=15 A=14,B=11	A=190
57600	A=22	A=22,B=15 A=30,B=11	A=382

38400	A=34	A=62,B=8 A=46,B=11 A=34,B=15	A=574
19200	A=70	A=126,B=8 A=94,B=11 A=70,B=15	A=1150
9600	A=142	A=254,B=8 A=190,B=11 A=142,B=15	A=2302
4800	A=286	A=510,B=8 A=382,B=11 A=286,B=15	A=4606

Table 5-9 UART Baud Rate Setting Table

The UART0/1 controllers support 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 UART and external devices (ex: 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.

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 one start bit, 8 data bits, and 1 stop bit. The maximum data rate is 115.2 Kbps (half duplex). The IrDA SIR block contains an IrDA SIR Protocol encoder/decoder. The IrDA SIR protocol is half-duplex only. So it cannot transmit and receive data at the same time. The IrDA SIR physical layer specifies a minimum 10 ms transfer delay between transmission and reception. This delay feature must be implemented by software.

For NuMicro™ NUC122, another alternate function of UART controllers is RS-485 9-bit mode function, and direction control provided by RTS pin or can program GPIO (PB.2 for RTS0 and PB.6 for RTS1) 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 driver control is implemented using the RTS control signal from an asynchronous serial port to enable the RS-485 driver. In RS-485 mode, many characteristics of the RX and TX are same as UART.

5.10.2 Features

- Full duplex, asynchronous communications
- Separate receive / transmit 14 bytes entry FIFO for data payloads
- Support hardware auto flow control/flow control function (CTS, RTS) and programmable RTS flow control trigger level
- Programmable receiver buffer trigger level
- Support programmable baud-rate generator for each channel individually
- Support CTS wake-up function
- Support 8 bits receiver buffer time-out detection function
- Programmable transmitting data delay time between the last stop and the next start bit by setting UA_TOR [DLY] register
- Support break error, frame error, parity error and receive / transmit buffer overflow detect function
- Fully programmable serial-interface characteristics
  - Programmable number of data bit, 5, 6, 7, 8 bits character
  - Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
  - Programmable stop bit, 1, 1.5, or 2 stop bits generation
- Support IrDA SIR function mode
  - Support for 3/16 bits duration for normal mode
- Support RS-485 function mode.
  - Support RS-485 9-bit mode
  - Support hardware or software direct enable control provided by RTS pin

5.10.3 Block Diagram

The UART clock control and block diagram are shown as Figure 5-33 and Figure 5-34.

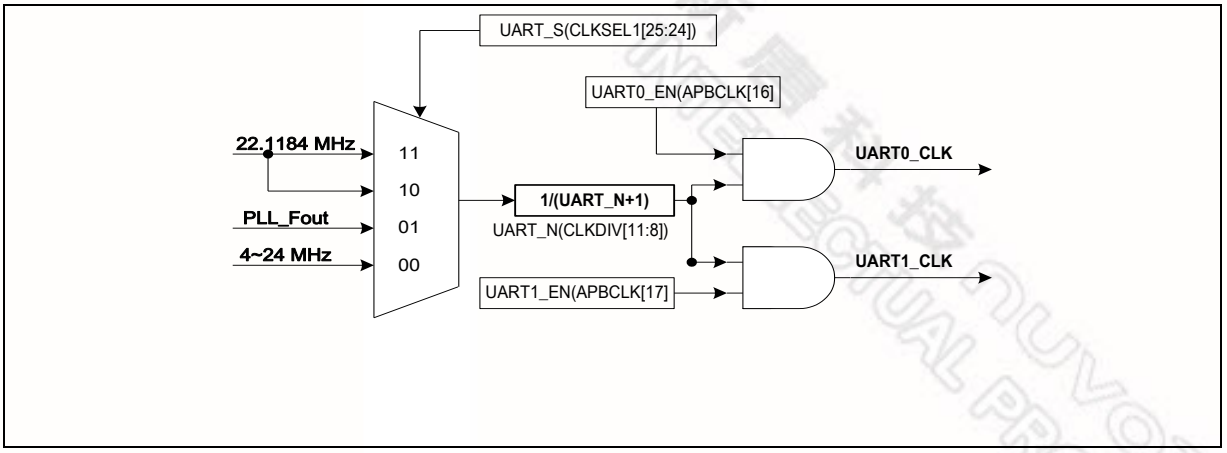


Figure 5-33 UART Clock Source Diagram

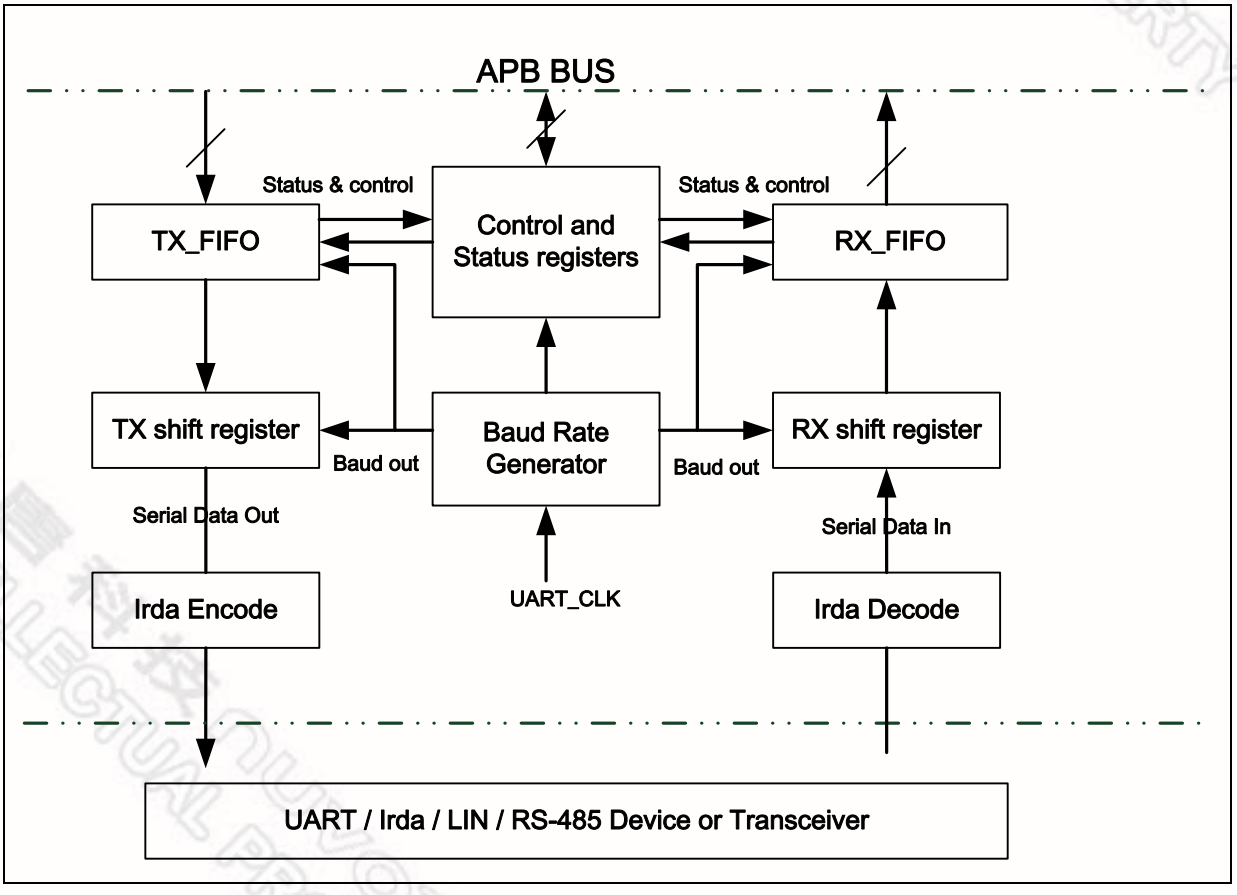


Figure 5-34 UART Block Diagram

TX_FIFO
The transmitter is buffered with a 14-byte FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO
The receiver is buffered with a 14-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 serially control block.

RX shift Register
This block is the shifting the receiving data in serially control block.

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 clock. Refer to baud rate equation.

IrDA Encode
This block is IrDA encode control block.

IrDA Decode
This block is IrDA decode control block.

Control and Status Register
This field is register set that including the FIFO control registers (UA_FCR), FIFO status registers (UA_FSR), and line control register (UA_LCR) for transmitter and receiver. The time-out control register (UA_TOR) identifies the condition of time-out interrupt. This register set also includes the interrupt enable register (UA_IER) and interrupt status register (UA_ISR) to enable or disable the responding interrupt and to identify the occurrence of the responding interrupt. There are seven types of interrupts, transmitter FIFO empty interrupt(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) and Buffer error interrupt (INT_BUF_ERR).

The following diagram demonstrates the auto-flow control block diagram.

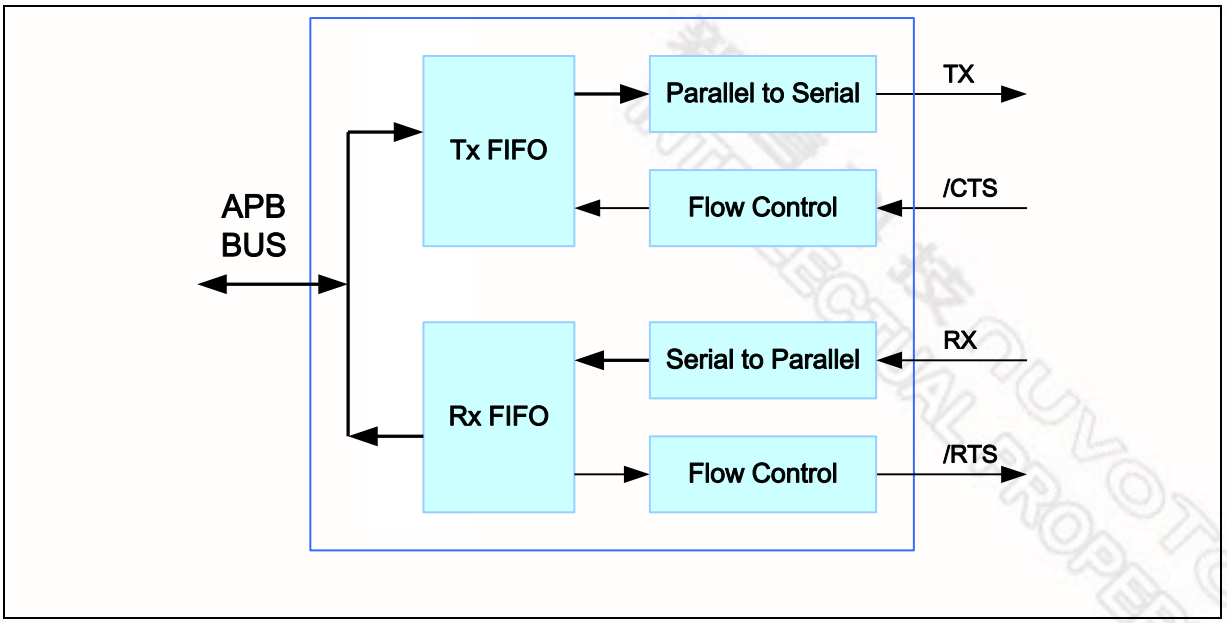


Figure 5-35 Auto Flow Control Block Diagram

5.10.4 IrDA Mode

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

When in IrDA mode, the UA_BAUD [DIV_X_EN] register must disable.

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

The following diagram demonstrates the IrDA control block diagram.

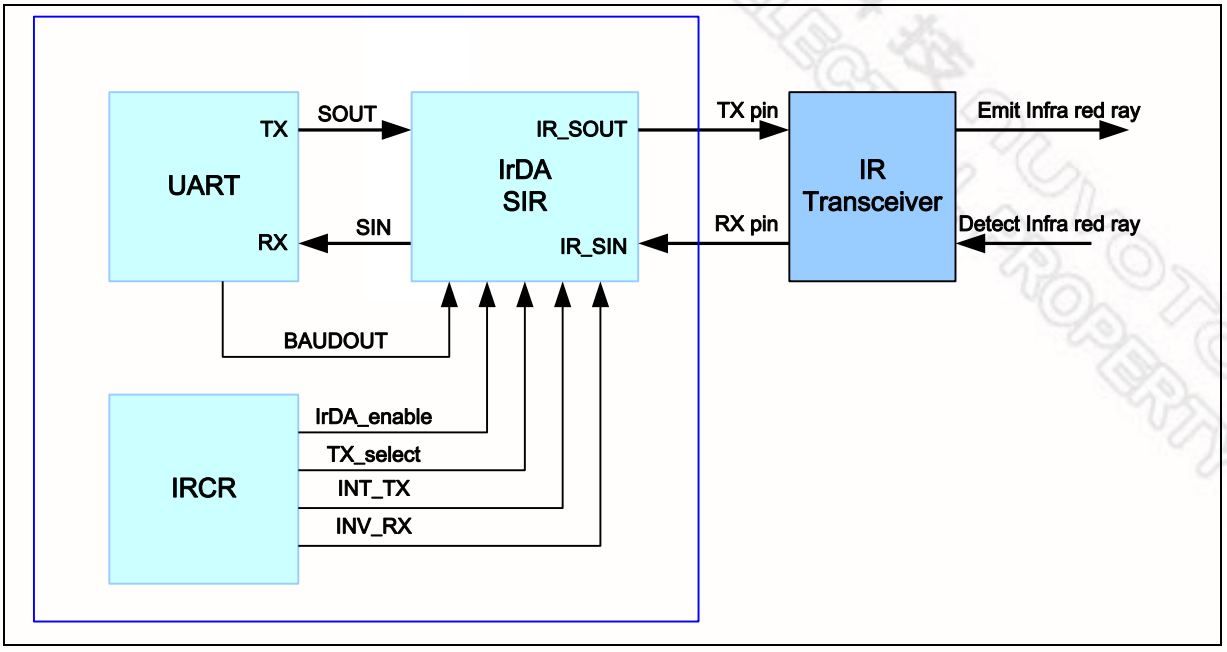


Figure 5-36 IrDA Block Diagram

5.10.4.1 IrDA SIR Transmit Encoder

The IrDA SIR Transmit Encoder modulate Non-Return-to Zero (NRZ) transmit bit stream output from UART. The IrDA SIR physical layer specifies use of Return-to-Zero, Inverted (RZI) modulation scheme which represent 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.

5.10.4.2 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

5.10.4.3 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:

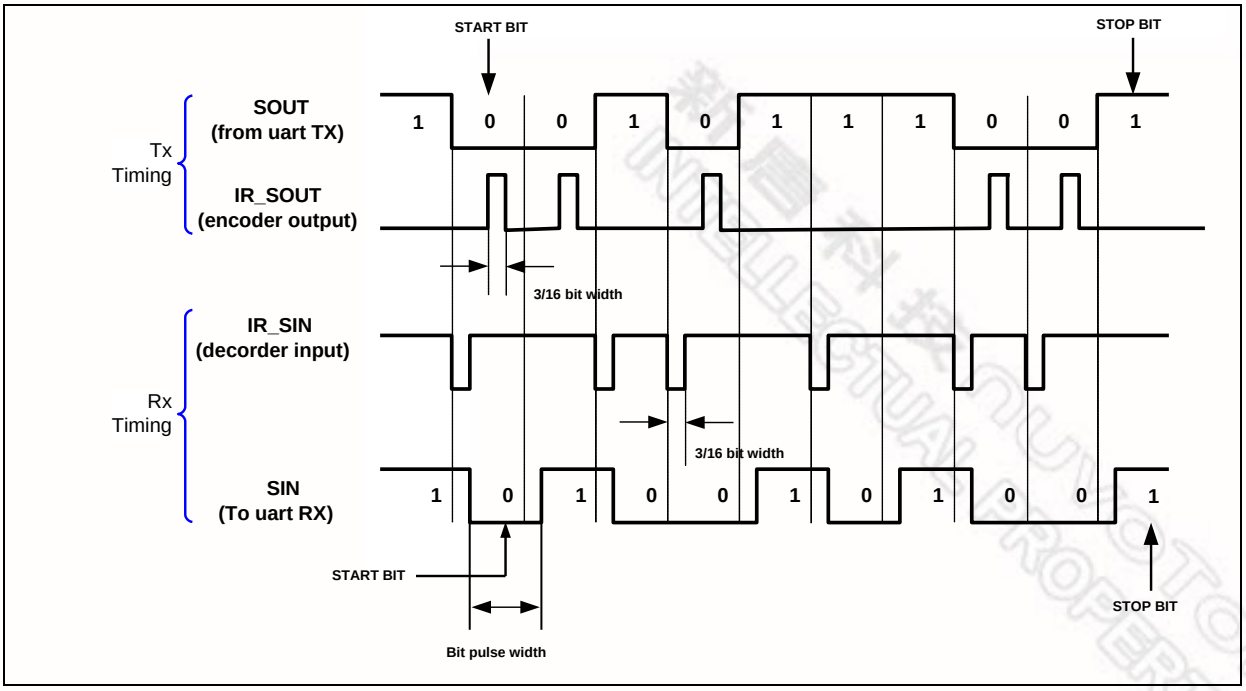


Figure 5-37 IrDA TX/RX Timing Diagram

5.10.5 RS-485 Function Mode

The UART support RS-485 9-bit mode function. The RS-485 mode is selected by setting the UA_FUN_SEL register to select RS-485 function. The RS-485 driver control is implemented using the RTS control signal from an asynchronous serial port to enable the RS-485 driver. In RS-485 mode, many characteristics of the RX and TX are same as UART.

When in RS-485 transfer mode, the controller can be configured as a RS-485 addressable slave mode and the RS-485 master transmitter will identify an address character by setting the parity (bit 9th) to 1. For data characters, the parity is set to 0. Software can program 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 that are RS-485 Normal Multi-drop Operation Mode (NMM), RS-485 Auto Address Detection Operation Mode (AAD) and RS-485 Auto Direction Control Operation Mode (AUD), one of the three operation modes can be selected by program UA_ALT_CSR register, and software can program the transfer delay time between the last stop and the next start bit by setting UA_TOR [DLY] register. Following figure show the structure of RS-485 frame

RS-485 Normal Multi-drop Operation Mode (NMM)

In RS-485 Normal Multi-drop operation mode, 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. Software can decide whether enable or disable receiver to accept the following data byte by setting UA_FCR [RX_DIS]. If the receiver is be enabled, all received byte data will be accepted and stored in the RX-FIFO, and if the receiver is disabled, all received byte data will be ignore until the next address byte be detected. If software disable receiver by setting UA_FCR [RX_DIS] register, when a next address byte be 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 match the UA_ALT_CSR [ADDR_MATCH] value. The address byte data will be stored in the RX-FIFO. The all received byte data will be accepted and stored in the RX-FIFO until an address byte or data byte 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 driver control is implemented using the RTS control signal from an asynchronous serial port to enable the RS-485 driver. The RTS line is connected to the RS-485 driver enable such that setting the RTS line to high (logic 1) enables the RS-485 driver. Setting the RTS line to low (logic 0) puts the driver into the tri-state condition. 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 RS-485 receiver

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.

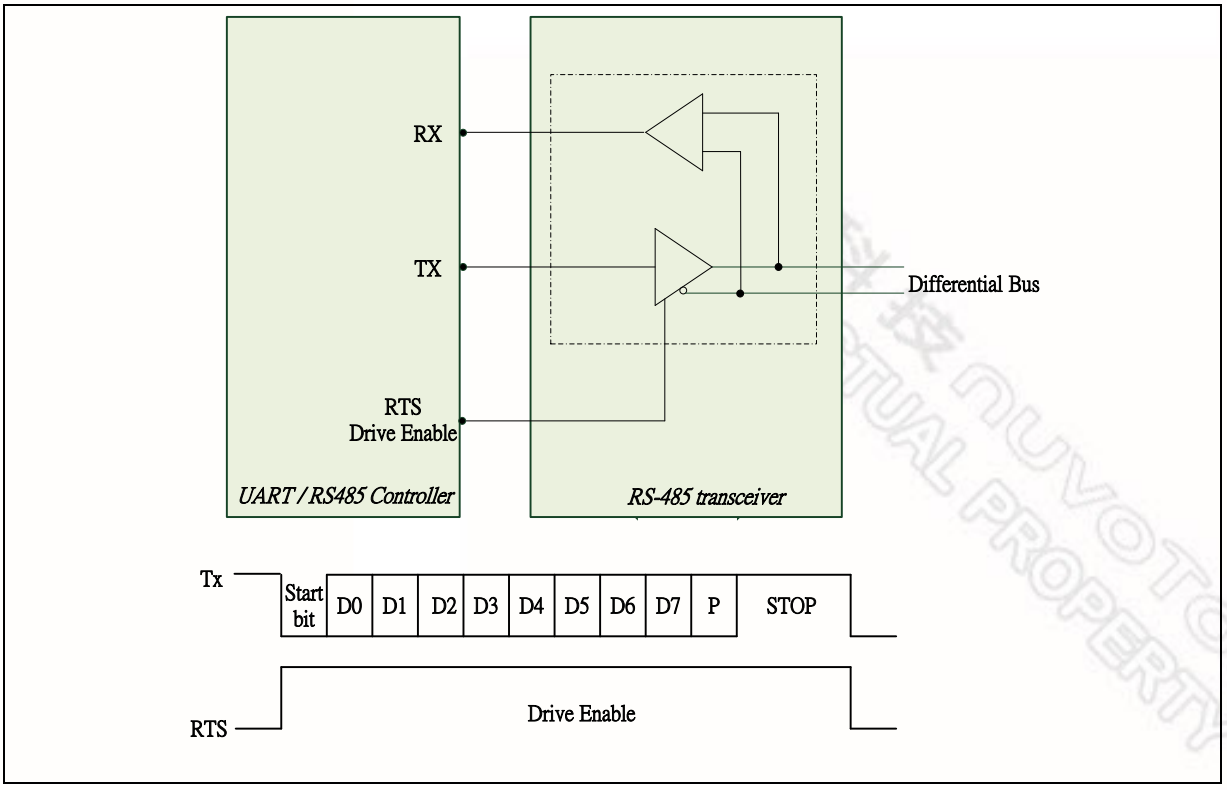


Figure 5-38 Structure of RS-485 Frame

5.10.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UART Base Address : **Channel0 : UART0_BA (Normal Speed)= 0x4005_0000** **Channel1 : UART1_BA (Normal Speed)= 0x4015_0000**				
UA_RBR	UART0_BA+0x00	R	UART0 Receive Buffer Register	Undefined
	UART1_BA+0x00	R	UART1 Receive Buffer Register	Undefined
UA_THR	UART0_BA+0x00	W	UART0 Transmit Holding Register	Undefined
	UART1_BA+0x00	W	UART1 Transmit Holding Register	Undefined
UA_IER	UART0_BA+0x04	R/W	UART0 Interrupt Enable Register	0x0000_0000
	UART1_BA+0x04	R/W	UART1 Interrupt Enable Register	0x0000_0000
UA_FCR	UART0_BA+0x08	R/W	UART0 FIFO Control Register	0x0000_0000
	UART1_BA+0x08	R/W	UART1 FIFO Control Register	0x0000_0000
UA_LCR	UART0_BA+0x0C	R/W	UART0 Line Control Register	0x0000_0000
	UART1_BA+0x0C	R/W	UART1 Line Control Register	0x0000_0000
UA_MCR	UART0_BA+0x10	R/W	UART0 Modem Control Register	0x0000_0200
	UART1_BA+0x10	R/W	UART1 Modem Control Register	0x0000_0200
UA_MSR	UART0_BA+0x14	R/W	UART0 Modem Status Register	0x0000_0110
	UART1_BA+0x14	R/W	UART1 Modem Status Register	0x0000_0110
UA_FSR	UART0_BA+0x18	R/W	UART0 FIFO Status Register	0x1040_4000
	UART1_BA+0x18	R/W	UART1 FIFO Status Register	0x1040_4000
UA_ISR	UART0_BA+0x1C	R/W	UART0 Interrupt Status Register	0x0000_0002
	UART1_BA+0x1C	R/W	UART1 Interrupt Status Register	0x0000_0002
UA_TOR	UART0_BA+0x20	R/W	UART0 Time-Out Register	0x0000_0000
	UART1_BA+0x20	R/W	UART1 Time-Out Register	0x0000_0000
UA_BAUD	UART0_BA+0x24	R/W	UART0 Baud Rate Divisor Register	0x0F00_0000
	UART1_BA+0x24	R/W	UART1 Baud Rate Divisor Register	0x0F00_0000
UA_IRCR	UART0_BA+0x28	R/W	UART0 IrDA Control Register	0x0000_0040
	UART1_BA+0x28	R/W	UART1 IrDA Control Register	0x0000_0040

UA_ALT_CSR	UART0_BA+0x2C	R/W	UART0 Alternate Control/Status Register	0x0000_0000
	UART1_BA+0x2C	R/W	UART1 Alternate Control/Status Register	0x0000_0000
UA_FUN_SEL	UART0_BA+0x30	R/W	UART0 Function Select Register	0x0000_0000
	UART1_BA+0x30	R/W	UART1 Function Select Register	0x0000_0000

5.10.7 Register Description

Receive Buffer Register (UA_RBR)

Register	Offset	R/W	Description	Reset Value
UA_RBR	UART0_BA+0x00	R	UART0 Receive Buffer Register	Undefined
	UART1_BA+0x00	R	UART1 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	Descriptions	
[31:8]	**Reserved**	Reserved
[7:0]	**RBR**	**Receive Buffer Register** (Read only) By reading this register, the UART will return an 8-bit data received from RX pin (LSB first).

Transmit Holding Register (UA_THR)

Register	Offset	R/W	Description	Reset Value
UA_THR	UART0_BA+0x00	W	UART0 Transmit Holding Register	Undefined
	UART1_BA+0x00	W	UART1 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	Descriptions	
[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	UART0_BA+0x04	R/W	UART0 Interrupt Enable Register	0x0000_0000
	UART1_BA+0x04	R/W	UART1 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		AUTO_CTS_EN	AUTO_RTS_EN	TIME_OUT_EN	Reserved		
7	6	5	4	3	2	1	0
Reserved	WAKE_EN	BUF_ERR_IEN	RTO_IEN	MODEM_IEN	RLS_IEN	THRE_IEN	RDA_IEN

Bits	Descriptions	
[31:14]	**Reserved**	Reserved
[13]	**AUTO_CTS_EN**	**CTS Auto Flow Control Enable** 1 = Enable CTS auto flow control 0 = Disable CTS auto flow control 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** 1 = Enable RTS auto flow control 0 = Disable RTS auto flow control 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** 1 = Enable Time-out counter 0 = Disable Time-out counter
[10:7]	**Reserved**	Reserved
[6]	**WAKE_EN**	**UART Wake-Up Function Enable** 0 = Disable UART wake-up function 1 = Enable UART wake-up function, 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** 0 = Mask off INT_BUF_ERR 1 = Enable INT_BUF_ERR
[4]	**RTO_IEN**	**RX Time-Out Interrupt Enable** 0 = Mask off INT_TOUT 1 = Enable INT_TOUT
[3]	**MODEM_IEN**	**Modem Status Interrupt Enable** 0 = Mask off INT_MODEM 1 = Enable INT_MODEM
[2]	**RLS_IEN**	**Receive Line Status Interrupt Enable** 0 = Mask off INT_RLS 1 = Enable INT_RLS
[1]	**THRE_IEN**	**Transmit Holding Register Empty Interrupt Enable** 0 = Mask off INT_THRE 1 = Enable INT_THRE
[0]	**RDA_IEN**	**Receive Data Available Interrupt Enable**. 0 = Mask off INT_RDA 1 = Enable INT_RDA

FIFO Control Register (UA_FCR)

Register	Offset	R/W	Description	Reset Value
UA_FCR	UART0_BA+0x08	R/W	UART0 FIFO Control Register	0x0000_0000
	UART1_BA+0x08	R/W	UART1 FIFO Control Register	0x0000_0000

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	Descriptions	
[31:20]	**Reserved**	Reserved
[19:16]	**RTS_TRI_LEV**	**RTS Trigger Level for Auto-flow Control** RTS_TRI_LEV / Trigger Level (Bytes) 0000 / 01 0001 / 04 0010 / 08 0011 / 14 others / Reserved Note: This field is used for auto RTS flow control.
[15:9]	**Reserved**	Reserved
[8]	**RX_DIS**	**Receiver Disable** The receiver is disabled or not (set 1 is disable receiver) 1 = Disable Receiver 0 = Enable Receiver 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 then the RDA_IF will be set (if UA_IER [RDA_IEN] is enable, an interrupt will generated).

		<table><tr><th>RFITL</th><th>INTR_RDA Trigger Level (Bytes)</th></tr><tr><td>0000</td><td>01</td></tr><tr><td>0001</td><td>04</td></tr><tr><td>0010</td><td>08</td></tr><tr><td>0011</td><td>14</td></tr><tr><td>others</td><td>Reserved</td></tr></table>
[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 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit will reset the TX internal state machine and pointers. Note: This bit will auto clear needs at least 3 UART engine 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 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit will reset the RX internal state machine and pointers. Note: This bit will auto clear needs at least 3 UART engine clock cycles.
[0]	**Reserved**	Reserved

Line Control Register (UA_LCR)

Register	Offset	R/W	Description	Reset Value
UA_LCR	UART0_BA+0x0C	R/W	UART0 Line Control Register	0x0000_0000
	UART1_BA+0x0C	R/W	UART1 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	

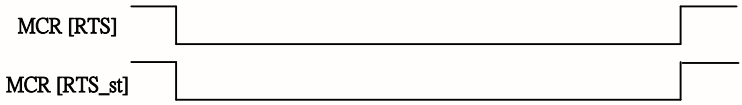
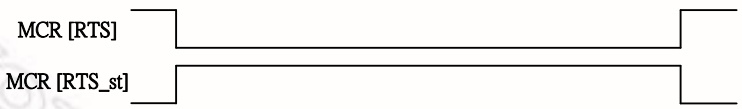
Bits	Descriptions	
[31:7]	**Reserved**	Reserved
[6]	**BCB**	**Break Control Bit** When this bit is set to logic 1, the serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX and has no effect on the transmitter logic.
[5]	**SPE**	**Stick Parity Enable** 1 = If bit 3 and 4 are logic 1, the parity bit is transmitted and checked as logic 0. If bit 3 is 1 and bit 4 is 0 then the parity bit is transmitted and checked as 1 0 = Stick parity disabled
[4]	**EPE**	**Even Parity Enable** 1 = Even number of logic 1's is transmitted and checked in each word 0 = Odd number of logic 1's is transmitted and checked in each word This bit has effect only when bit 3 (parity bit enable) is set.
[3]	**PBE**	**Parity Bit Enable** 1 = Parity bit is generated on each outgoing character and is checked on each incoming data. 0 = No parity bit.
[2]	**NSB**	**Number of "STOP bit"** 1 = One and a half " STOP bit" is generated in the transmitted data when 5-bit word length is selected; 0 = One " STOP bit" is generated in the transmitted data

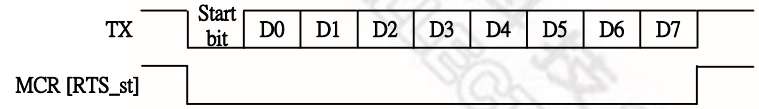
		Two “STOP bit” is generated when 6-, 7- and 8-bit word length is selected.
[1:0]	**WLS**	**Word Length Select** **WLS[1:0]** / **Character length** 00 / 5 bits 01 / 6 bits 10 / 7 bits 11 / 8 bits

MODEM Control Register (UA_MCR)

Register	Offset	R/W	Description	Reset Value
UA_MCR	UART0_BA+0x10	R/W	UART0 Modem Control Register	0x0000_0200
	UART1_BA+0x10	R/W	UART1 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	Descriptions	
[31:14]	**Reserved**	Reserved
[13]	**RTS_ST**	**RTS Pin State** (Read only) This bit is the output pin status of RTS.
[12:10]	**Reserved**	Reserved
[9]	**LEV_RTS**	**RTS Trigger Level** This bit can change the RTS trigger level. 0= low level triggered 1= high level triggered *UART Mode : MCR[Lev_RTS] = 1*  *UART Mode : MCR[Lev_RTS] = 0* 

		RS-485 Mode : MCR[Lev_RTS] = 1  *RS-485 Mode : MCR[Lev_RTS] = 0* 
[8:2]	**Reserved**	Reserved
[1]	**RTS**	**RTS (Request-To-Send) Signal** 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)

Register	Offset	R/W	Description	Reset Value
UA_MSR	UART0_BA+0x14	R/W	UART0 Modem Status Register	0x0000_0110
	UART1_BA+0x14	R/W	UART1 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	Descriptions	
[31:9]	**Reserved**	Reserved
[8]	**LEV_CTS**	**CTS polarity setting** This bit can select the polarity of CTS active level to send TX_FIFO data. According to CTS pin and LEV_CTS setting, the four cases are described as following: **1. CTS_pin =0 and LEV_CTS = 0 case:** When the **CTS** pin input is low, the CTS function is not active if the **LEV_CTS** is set to 0. **2. CTS_pin =1 and LEV_CTS = 0 case:** When the **CTS** pin input is high, the CTS function is active if the **LEV_CTS** is set to 0. **3. CTS_pin =0 and LEV_CTS = 1 case:** When the **CTS** pin input is low, the CTS function is active if the **LEV_CTS** is set to 1. **4. CTS_pin =1 and LEV_CTS = 1 case:** When the **CTS** pin input is high, the CTS function is not active if the **LEV_CTS** is set to 1.
[7:5]	**Reserved**	Reserved
[4]	**CTS_ST**	**CTS Pin Status** (Read only) This bit is the pin status of CTS.
[3:1]	**Reserved**	Reserved
[0]	**DCTSF**	**Detect CTS State Change Flag** (Read only) 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. Software can write 1 to clear this bit to zero

FIFO Status Register (UA_FSR)

Register	Offset	R/W	Description	Reset Value
UA_FSR	UART0_BA+0x18	R/W	UART0 FIFO Status Register	0x1040_4000
	UART1_BA+0x18	R/W	UART1 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	RS-485_ADD_DETF	Reserved		RX_OVER_IF

Bits	Descriptions	
[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 full or not. This bit is set when the number of usage in TX FIFO Buffer is more than 14, otherwise is cleared by hardware.
[22]	**TX_EMPTY**	**Transmitter FIFO Empty** (Read only) This bit indicates TX FIFO 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, TX_POINTER increases one. When one byte of TX FIFO is transferred to Transmitter Shift Register, TX_POINTER decreases one.
[15]	**RX_FULL**	**Receiver FIFO Full** (Read only) This bit initiates RX FIFO full or not. This bit is set when the number of usage in RX FIFO Buffer is more than 14, 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, RX_POINTER increases one. When one byte of RX FIFO is read by CPU, RX_POINTER decreases one.
[7]	**Reserved**	Reserved
[6]	**BIF**	**Break Interrupt Flag** (Read only) This bit is set to a 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 RFR in UA_FCR register.
[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 a 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 RFR in UA_FCR register.
[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 RFR in UA_FCR register.
[3]	**RS-485_ADD_DETF**	**RS-485 Address Byte Detection Flag** (Read only) This bit is set to logic 1 and set UA_ALT_CSR [RS-485_ADD_EN] whenever in RS-485 mode the receiver detect any address byte received address byte character (bit9 = ‘1’) bit", and it is reset whenever the CPU writes 1 to this bit. 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, 14 bytes of UART0/UART1, this bit will be set. Note: This bit is read only, but can be cleared by writing ‘1’ to it.

Interrupt Status Register (UA_ISR)

Register	Offset	R/W	Description	Reset Value
UA_ISR	UART0_BA+0x1C	R/W	UART0 Interrupt Status Register	0x0000_0002
	UART1_BA+0x1C	R/W	UART1 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		BUF_ERR_INT	TOUT_INT	MODEM_INT	RLS_INT	THRE_INT	RDA_INT
7	6	5	4	3	2	1	0
Reserved		BUF_ERR_IF	TOUT_IF	MODEM_IF	RLS_IF	THRE_IF	RDA_IF

Bits	Descriptions	
[31:14]	**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. 1 = The buffer error interrupt is generated 0 = No 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. 1 = The Tout interrupt is generated 0 = No Tout interrupt is generated
[11]	**MODEM_INT**	**MODEM Status Interrupt Indicator** (Read only). This bit is set if MODEM_IEN and MODEM_IF are both set to 1. 1 = The Modem interrupt is generated 0 = No 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. 1 = The RLS interrupt is generated 0 = No 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. 1 = The THRE interrupt is generated 0 = No 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. 1 = The RDA interrupt is generated 0 = No RDA interrupt is generated
[7:6]	**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 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.
[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) 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, framing 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: When 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.
[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	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	RTO_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)	Write ‘1’ to BIF/FEF/PEF
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 5-10 UART Interrupt Sources and Flags Table In Software Mode

Time-out Register (UA_TOR)

Register	Offset	R/W	Description	Reset Value
UA_TOR	UART0_BA+0x20	R/W	UART0 Time-Out Register	0x0000_0000
	UART1_BA+0x20	R/W	UART1 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	Descriptions	
[31:16]	**Reserved**	Reserved
[15:8]	**DLY**	**TX Delay Time Value** This field is use 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 [RTO_IEN]. A new incoming data word or RX FIFO empty clears 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	UART0_BA+0x24	R/W	UART0 Baud Rate Divisor Register	0x0F00_0000
	UART1_BA+0x24	R/W	UART1 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	Descriptions	
[31:30]	**Reserved**	Reserved
[29]	**DIV_X_EN**	**Divider X Enable** 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 = Disable divider X (the equation of M = 16) 1 = Enable divider X (the equation of M = X+1, but DIVIDER_X [27:24] must >= 8). Refer to the table below for more information. Note: When in IrDA mode, this bit must disable.
[28]	**DIV_X_ONE**	**Divider X Equal 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 the table below 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 indicated the baud rate divider

Mode	DIV_X_EN	DIV_X_ONE	DIVIDER X	BRD	Baud rate equation
0	Disable	0	B	A	UART_CLK / [16 * (A+2)]
1	Enable	0	B	A	UART_CLK / [(B+1) * (A+2)] , B must >= 8
2	Enable	1	Don't care	A	UART_CLK / (A+2), A must >=3

IrDA Control Register (IRCR)

Register	Offset	R/W	Description	Reset Value
UA_IRCR	UART0_BA+0x28	R/W	UART0 IrDA Control Register	0x0000_0040
	UART1_BA+0x28	R/W	UART1 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	Descriptions	
[31:7]	**Reserved**	Reserved
[6]	**INV_RX**	**INV_RX** 1= Inverse RX input signal 0= No inversion
[5]	**INV_TX**	**INV_TX** 1= Inverse TX output signal 0= No inversion
[4:2]	**Reserved**	Reserved
[1]	**TX_SELECT**	**TX_SELECT** 1= Enable IrDA transmitter 0= Enable IrDA receiver
[0]	**Reserved**	Reserved

Note: When in IrDA mode, the UA_BAUD [DIV_X_EN] register must disable (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	UART0_BA+0x2C	R/W	UART0 Alternate Control/Status Register	0x0000_0000
	UART1_BA+0x2C	R/W	UART1 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
RS-485_ADD_EN	Reserved				RS-485_AUD	RS-485_AAD	RS-485_NMM
7	6	5	4	3	2	1	0
Reserved							

Bits	Descriptions	
[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]	**RS-485_ADD_EN**	**RS-485 Address Detection Enable** This bit is use to enable RS-485 address detection mode. 1 = Enable address detection mode 0 = Disable address detection mode Note: This field is used for RS-485 any operation mode.
[14:11]	**Reserved**	Reserved
[10]	**RS-485_AUD**	**RS-485 Auto Direction Mode (AUD)** 1 = Enable RS-485 Auto Direction Operation Mode (AUO) 0 = Disable RS-485 Auto Direction Operation Mode (AUO) Note: It can be active with RS-485_AAD or RS-485_NMM operation mode.
[9]	**RS-485_AAD**	**RS-485 Auto Address Detection Operation Mode (AAD)** 1 = Enable RS-485 Auto Address Detection Operation Mode (AAD) 0 = Disable RS-485 Auto Address Detection Operation Mode (AAD) **Note:** It can't be active with RS-485_NMM operation mode.

[8]	**RS-485_NMM**	**RS-485 Normal Multi-drop Operation Mode (NMM)** 1 = Enable RS-485 Normal Multi-drop Operation Mode (NMM) 0 = Disable RS-485 Normal Multi-drop Operation Mode (NMM) Note: It can't be active with RS-485_AAD operation mode.
[7:0]	**Reserved**	Reserved

UART Function Select Register (UA_FUN_SEL)

Register	Offset	R/W	Description	Reset Value
UA_FUN_SEL	UART0_BA+0x30	R/W	UART0 Function Select Register	0x0000_0000
	UART1_BA+0x30	R/W	UART1 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	Descriptions	
[31:2]	**Reserved**	Reserved
[1:0]	**FUN_SEL**	**Function Select Enable** 00 = UART Function 01 = Reserved 10 = Enable IrDA Function 11 = Enable RS-485 Function

5.11 PS/2 Device Controller (PS2D)

5.11.1 Overview

PS/2 device controller provides 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 received/transmit code needs to be translated as meaningful code by firmware. The device controller generates the CLK signal after receiving a request to send, but host has ultimate control over communication. DATA sent from the host to the device is read on the rising edge and DATA sent from device to the host is change after rising edge. A 16 bytes FIFO is used to reduce CPU intervention. S/W can select 1 to 16 bytes for a continuous transmission.

5.11.2 Features

- 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
- S/W override bus

5.11.3 Block Diagram

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

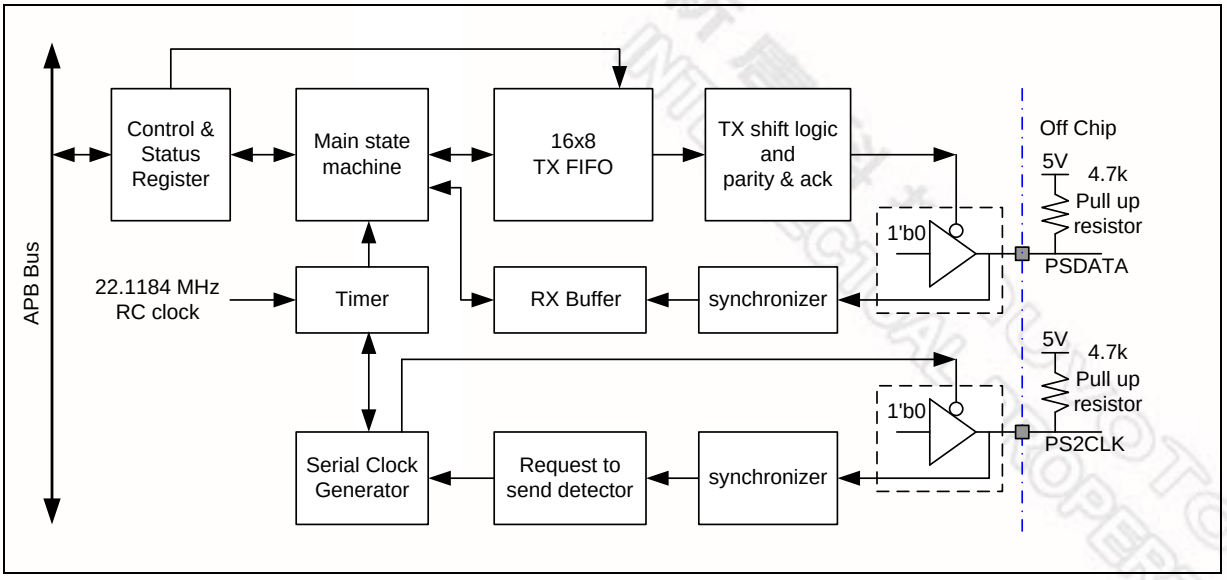


Figure 5-39 PS/2 Device Block Diagram

5.11.4 Functional Description

5.11.4.1 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 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 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. This is always 0
- 8 DATA bits, least significant bit first
- 1 parity bit (odd parity)
- 1 stop bit. This 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. The numbers of 1's in the data bits plus the parity bit always add up to an odd number set to 1. This is used for error detection. 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 microseconds. If a transmission is inhibited before the 11th clock pulse, the device must abort the current transmission and prepare to retransmit the current data when host releases Clock. In order to reserve enough time for s/w to decode host command, the transmit logic is blocked by RXINT bit, S/W must clear RXINT bit to start retransmit. S/W can write CLRFIFO to 1 to reset FIFO pointer if need.

Device-to-Host

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

- 1 start bit. This is always 0
- 8 DATA bits, least significant bit first
- 1 parity bit (odd parity)
- 1 stop bit. This 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. Figures in the following illustrate this.

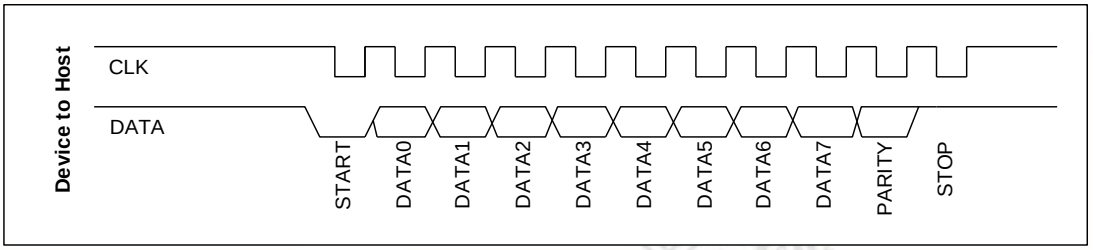


Figure 5-40 Data Format of Device-to-Host

Host-to-Device:

First of all, the PS/2 device always generates the CLK signal. If the host wants to send 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 microseconds
- Apply "Request-to-send" by pulling DATA low, then release CLK

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

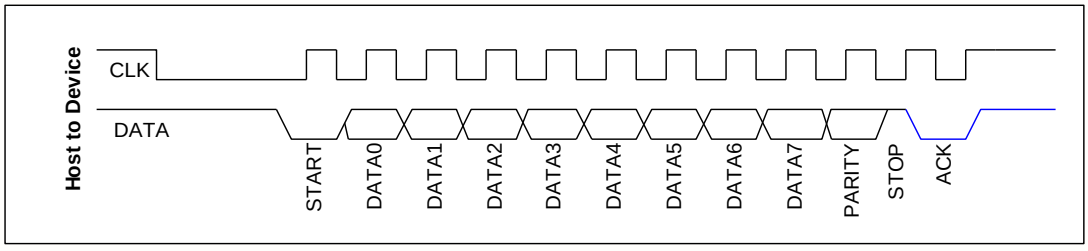


Figure 5-41 Data Format of Host-to-Device

The host and the device DATA and CLK detailed timing for communication is shown as below:

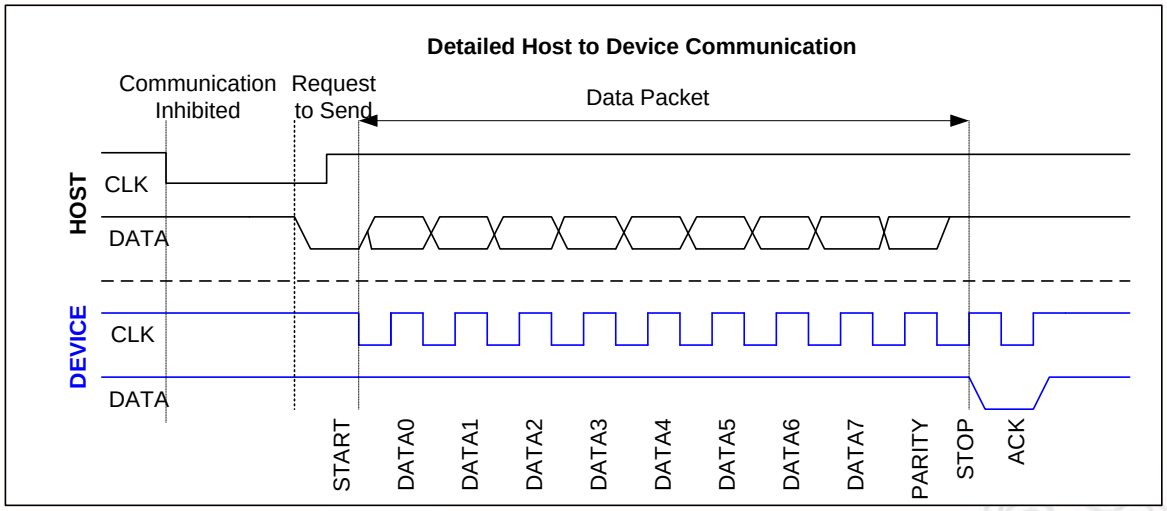


Figure 5-42 PS/2 Bit Data Format

5.11.4.2 PS/2 Bus Timing Specification

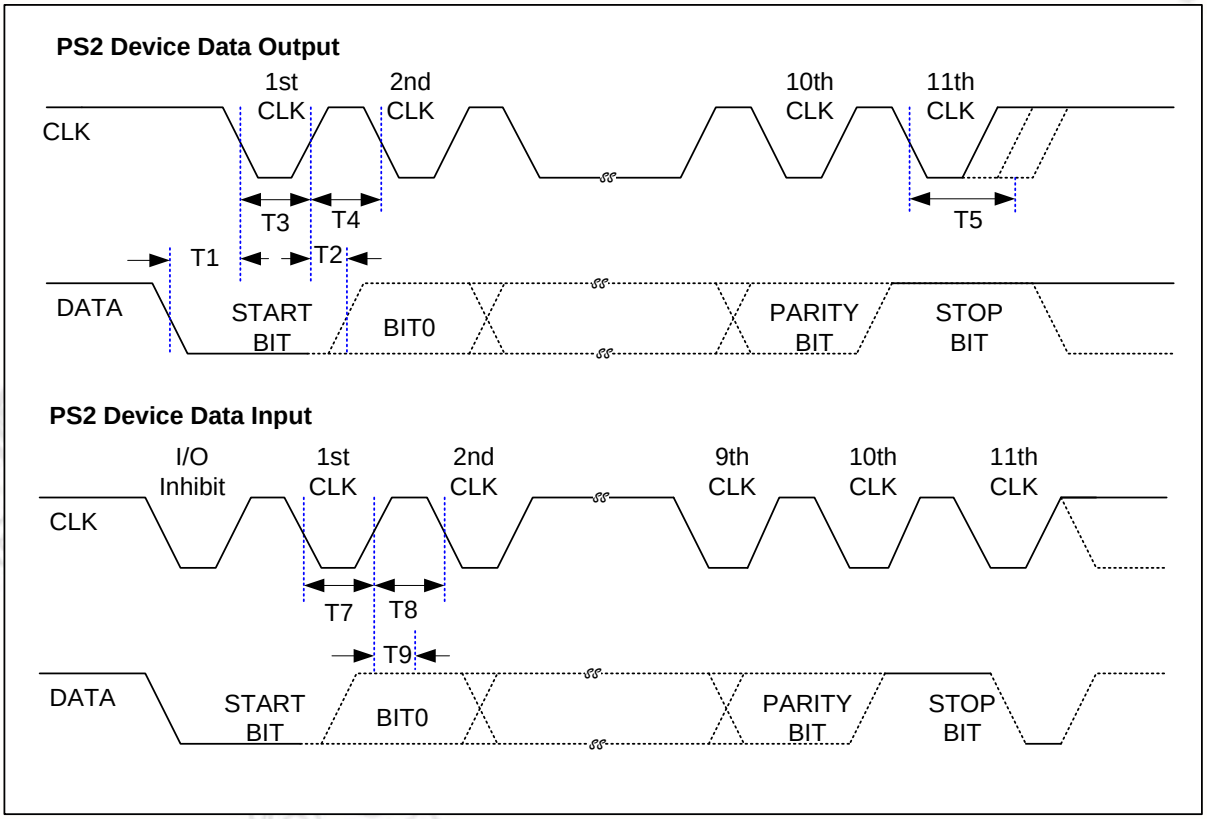


Figure 5-43 PS/2 Bus Timing

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

5.11.4.3 TX FIFO Operation

Writing PS2TXDATA0 register starts device to host communication. S/W is required to define TXFIFO depth before writing transmission data to TX FIFO. 1st START bit is sent to PS/2 bus 100us after S/W writes TX FIFO, if there is more than 4 bytes data need to be sent, S/W can write residual data to PS2TXDATA1-3 before 4th byte transmit complete. A time delay 100us is added between two consecutive bytes.

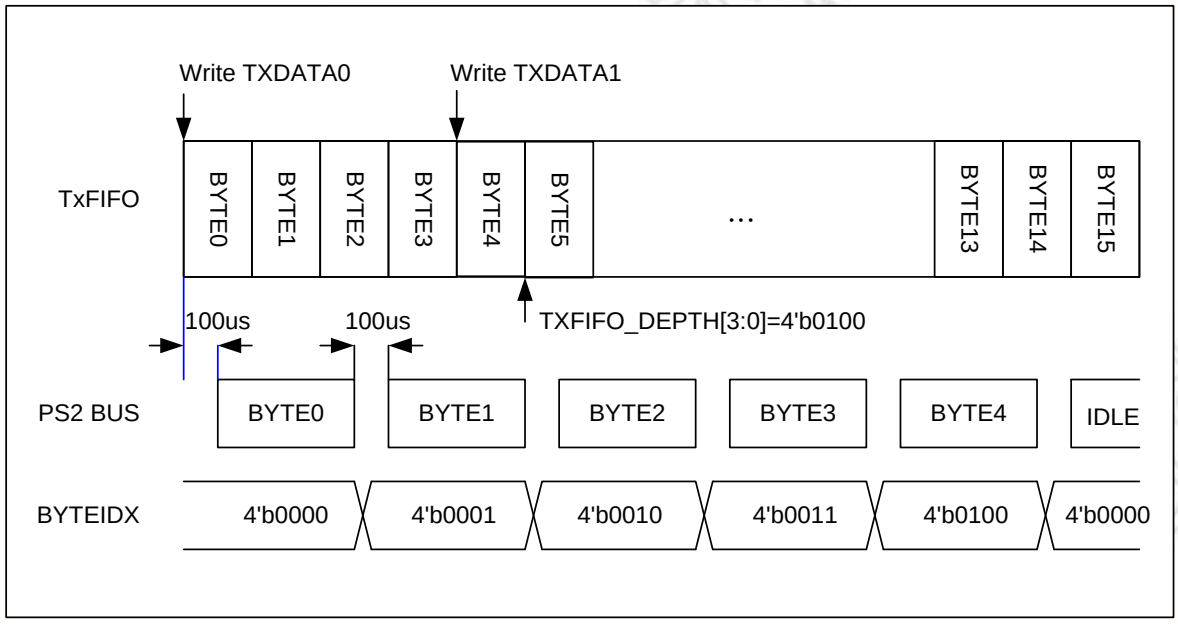


Figure 5-44 PS/2 Data Format

5.11.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
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

5.11.6 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	Descriptions	
[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 high. 1 = Force PS2DATA high 0 = Force PS2DATA low
[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 high. 1 = Force PS2CLK line high 0 = Force PS2CLK line low
[9]	**OVERRIDE**	**Software Override PS/2 CLK/DATA Pin State** 1 = PS2CLK and PS2DATA pins are controlled by S/W 0 = PS2CLK and PS2DATA pins are controlled by internal state machine.
[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. 1 = Clear FIFO 0 = Not active

[7]	**ACK**	**Acknowledge Enable** 1 = If parity error or stop bit is not received correctly, acknowledge bit will not be sent to host at 12th clock 0 = Always send acknowledge to host at 12th clock for host to device communication.
[6:3]	**TXFIFODIPTH**	**Transmit Data FIFO Depth** There is 16 bytes buffer for data transmit. S/W can define the FIFO depth from 1 to 16 bytes depends on application. 0 = 1 byte 1 = 2 bytes … 14 = 15 bytes 15 = 16 bytes
[2]	**RXINTEN**	**Enable Receive Interrupt** 1 = Enable data receive complete interrupt 0 = Disable data receive complete interrupt
[1]	**TXINTEN**	**Enable Transmit Interrupt** 1 = Enable data transmit complete interrupt 0 = Disable data transmit complete interrupt
[0]	**PS2EN**	**Enable PS/2 Device** Enable PS/2 device controller 1 = Enable 0 = Disable

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[31:24]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PS2TXDATAx[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PS2TXDATAx[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PS2TXDATAx[7:0]							

Bits	Descriptions	
[31:0]	**PS2TXDATAx**	**Transmit Data** Write data to this register starts device to host communication if bus is in IDLE state. S/W 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[7:0]							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7:0]	**PS2RXDATA**	**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_0000

31	30	29	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[3:0]			
7	6	5	4	3	2	1	0
TXEMPTY	RXOVF	TXBUSY	RXBUSY	RXPARTY	FRAMERR	PS2DATA	PS2CLK

Bits	Descriptions	
[31:12]	**Reserved**	Reserved
[11:8]	**BYTEIDX**	**Byte Index** It indicates which data byte in transmit data shift register. When all data in FIFO is transmitted and it will be cleared to 0. It is a read only bit.
[7]	**TXEMPTY**	**TX FIFO Empty** When S/W writes any 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. 1 = FIFO is empty 0 = There is data to be transmitted

BYTEIDX	DATA Transmit	BYTEIDX	DATA Transmit
0000	TXDATA0[7:0]	1000	TXDATA2[7:0]
0001	TXDATA0[15:8]	1001	TXDATA2[15:8]
0010	TXDATA0[23:16]	1010	TXDATA2[23:16]
0011	TXDATA0[31:24]	1011	TXDATA2[31:24]
0100	TXDATA1[7:0]	1100	TXDATA3[7:0]
0101	TXDATA1[15:8]	1101	TXDATA3[15:8]
0110	TXDATA1[23:16]	1110	TXDATA3[23:16]
0111	TXDATA1[31:24]	1111	TXDATA3[31:24]

		Read only bit.
[6]	**RXOVF**	**RX Buffer Overwrite** 1 = Data in PS2RXDATA register is overwritten by new received data 0 = No overwrite Write 1 to clear this bit.
[5]	**TXBUSY**	**Transmit Busy** This bit indicates that the PS/2 device is currently sending data. 0 = Idle 1 = Currently sending data Read only bit.
[4]	**RXBUSY**	**Receive Busy** This bit indicates that the PS/2 device is currently receiving data. 0 = Idle 1 = Currently receiving data Read only bit.
[3]	**RXPARITY**	**Received Parity** This bit reflects the parity bit for the last received data byte (odd parity). Read only bit.
[2]	**FRAMERR**	**Frame Error** For host to device communication, if STOP bit (logic 1) is not received it is a frame error. If frame error occurs, DATA line may keep at low state after 12th clock. At this moment, S/W overrides PS2CLK to send clock till PS2DATA release to high state. After that, device sends a “Resend” command to host. 1 = Frame error occur 0 = No frame error Write 1 to clear this bit.
[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	Descriptions	
[31:3]	**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. 1 = Transmit interrupt occurs 0 = No interrupt Write 1 to clear this bit to 0.
[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. 1 = Receive interrupt occurs 0 = No interrupt Write 1 to clear this bit to 0.

5.12 I²C Serial Interface Controller (Master/Slave) (I²C)

5.12.1 Overview

I²C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. The I²C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously.

Data is transferred between a Master and a Slave synchronously to SCL on the SDA line on a byte-by-byte basis. Each data byte is 8 bits long. There is one SCL clock pulse for each data bit with the MSB being transmitted first. 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 the following figure for more detail I²C BUS Timing.

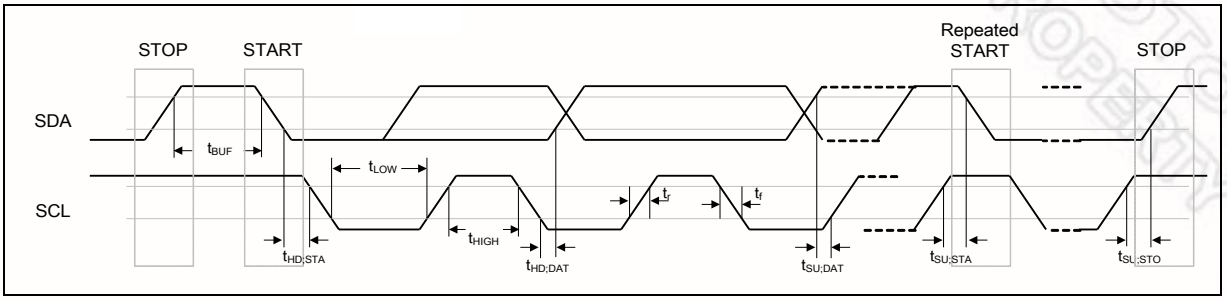


Figure 5-45 I²C Bus Timing

The device's on-chip I²C logic provides the serial interface that meets the I²C bus standard mode specification. The I²C port handles byte transfers autonomously. To enable this port, the bit ENS1 in I2CON should be set to '1'. The I²C H/W interfaces to the I²C bus via two pins: SDA (PA10, serial data line) and SCL (PA11, serial clock line). Pull up resistor is needed for Pin PA10 and PA11 for I²C operation as these are open drain pins. When the I/O pins are used as I²C port, user must set the pins function to I²C in advance.

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

- 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 allows devices with different bit rates to communicate via one serial bus
- Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer
- Built-in a 14-bit time-out counter will request the I²C interrupt if the I²C bus hangs up and timer-out counter overflows.

- External pull-up are needed for high output
- Programmable clocks allow versatile rate control
- Supports 7-bit addressing mode
- I²C-bus controllers support multiple address recognition (Four slave address with mask option)

5.12.1.1 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

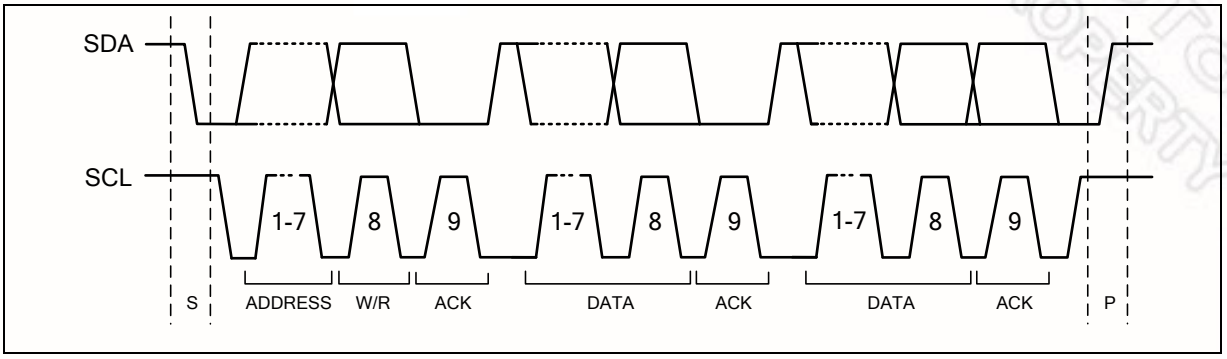


Figure 5-46 I²C Protocol

5.12.1.2 Data transfer on the I²C-bus

A master-transmitter addressing a slave receiver with a 7-bit address

The transfer direction is not changed

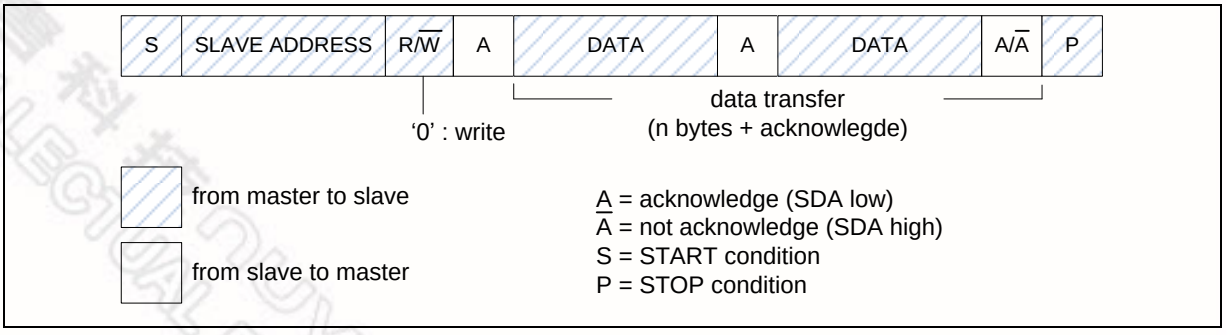


Figure 5-47 Master Transmits Data to Slave

A master reads a slave immediately after the first byte (address)

The transfer direction is changed

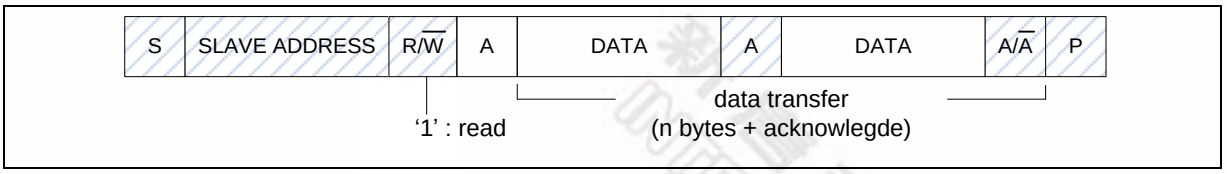


Figure 5-48 Master Reads Data from Slave

5.12.1.3 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 transfer.

A Repeated START (Sr) is no 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.

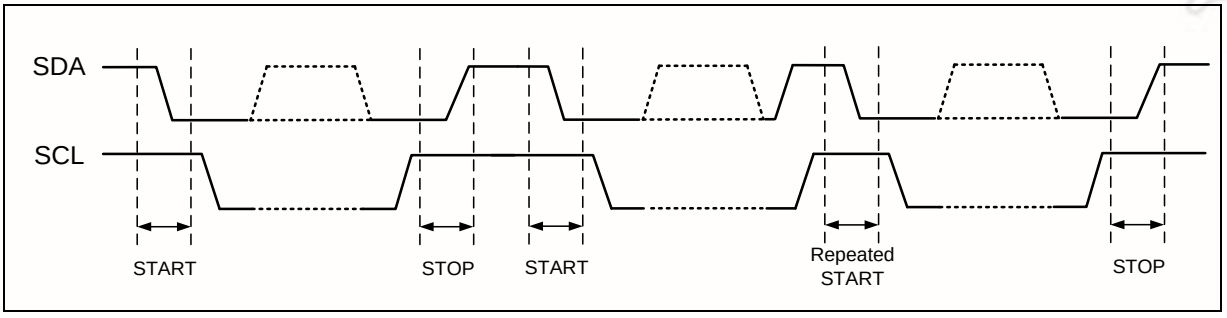


Figure 5-49 START and STOP Condition

5.12.1.4 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-bits 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.

5.12.1.5 Data Transfer

Once successful slave addressing has been achieved, the data transfer can proceed on a byte-by-byte basis in the direction specified by the RW bit sent by the master. 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 the 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.

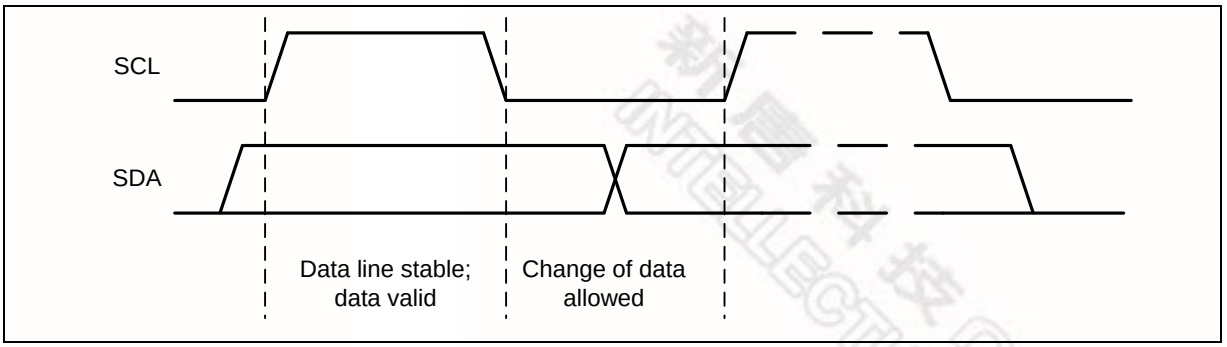


Figure 5-50 Bit Transfer on the I²C bus

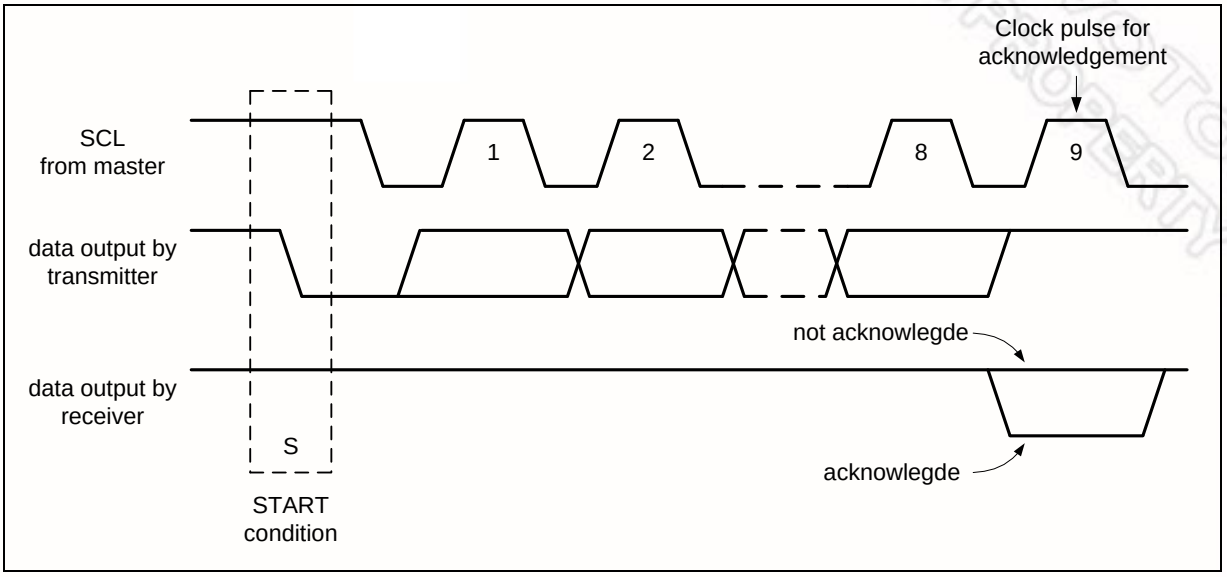


Figure 5-51 Acknowledge on the I²C bus

5.12.2 Protocol Registers

The CPU interfaces 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 zero if read back.

When 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 high 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 4012_0000.

5.12.2.1 Address Registers (I2CADDR)

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 the 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 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.

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 zero, that means the received corresponding register bit should be exact the same as address register.

5.12.2.2 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 read from or write to this 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 as long as SI bit is set. While data is being shifted out, data on the bus is simultaneously being shifted in; I2CDAT [7:0] always contains the last data byte present on the bus. Thus, in the event of arbitration lost, the transition from master transmitter to slave receiver is made with the correct data in I2CDAT [7:0].

I2CDAT [7:0] and the acknowledge bit form a 9-bit shift register, the acknowledge bit is controlled by the I²C hardware and cannot be accessed by the CPU. Serial data is shifted through the acknowledge bit 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. Serial data is shifted out from I2CDAT [7:0] on the falling edges of SCL clock pulses, and is shifted into I2CDAT [7:0] on the rising edges of SCL clock pulses.

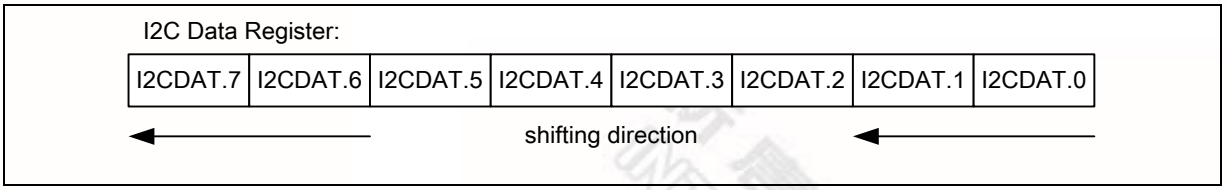


Figure 5-52 I²C Data Shifting Direction

5.12.2.3 Control Register (I2CON)

The CPU can read from and write to this 8-bit field of I2CON [7:0] directly. Two bits are affected by hardware: the SI bit is set when the I²C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when ENS1 = 0.

EI	Enable Interrupt.
ENS1	Set to enable I²C serial function controller. When ENS1=1 the I²C serial function enables. The Multi Function pin function of SDA and SCL must be set to I²C function.
STA	I²C START Control Bit. Setting STA to logic 1 to enter master mode, the I²C hardware sends a START or repeat START condition to bus when the bus is free.
STO	I²C STOP Control Bit. In master mode, setting STO to transmit a STOP condition to bus then I²C hardware will check the bus condition if a STOP condition is detected this flag will be cleared by hardware automatically. In a slave mode, setting STO resets I²C hardware to the defined “not addressed” slave mode. This means it is NO LONGER in the slave receiver mode to receive data from the master transmit device.
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. 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 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.

5.12.2.4 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] contain the status code. There are 26 possible status codes, All states are listed in section 5.6.6. When I2CSTATUS [7:0] contains F8H, no serial interrupt is requested. All other I2CSTATUS [7:3] values correspond to defined I²C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2CSTATUS[7:3] one cycle after SI is set by hardware and is still present one cycle after SI has been reset by software.

In addition, state 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the format frame. Examples of illegal positions are during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I²C from bus error, STO should be set and SI should be clear to enter not addressed slave mode. Then clear STO to release bus and to wait new communication. I²C bus can not recognize stop condition during this action when bus error occurs.

5.12.2.5 I²C Clock Baud Rate Bits (I2CLK)

The data baud rate of I²C is determines by I2CLK [7:0] register when I²C is in a master mode. It is not important when I²C is in a slave mode. In the slave modes, I²C will automatically synchronize with any clock frequency up to 1 MHz from master I²C device.

The data baud rate of I²C setting is Data Baud Rate of I²C = (system clock) / (4x (I2CLK [7:0] +1)). If system clock = 16 MHz, the I2CLK [7:0] = 40 (28H), so data baud rate of I²C = 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec.

5.12.2.6 The I²C Time-out Counter Register (I2CTOC)

There is a 14-bit time-out counter which can be used to deal with the I²C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows (TIF=1) and generates I²C interrupt to CPU or stops counting by clearing ENTI to 0. When time-out counter is enabled, setting flag SI to high will reset counter and re-start up counting after SI is cleared. If I²C 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 I²C interrupt. Refer to the following figure for the 14-bit time-out counter. User may write 1 to clear TIF to zero.

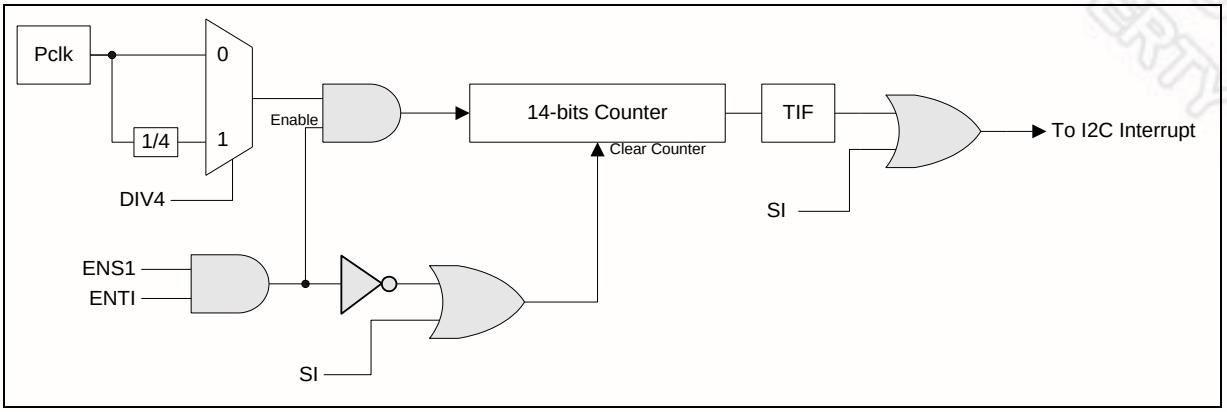


Figure 5-53 I²C Time-out Counter Block Diagram

5.12.3 Register Map

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

Register	Offset	R/W	Description	Reset Value
I2C_BA = 0x4012_0000				
I2CON	I2C_BA+0x00	R/W	I^2C Control Register	0x0000_0000
I2CADDR0	I2C_BA+0x04	R/W	I^2C Slave Address Register 0	0x0000_0000
I2CDAT	I2C_BA+0x08	R/W	I^2C DATA Register	0x0000_0000
I2CSTATUS	I2C_BA+0x0C	R	I^2C Status Register	0x0000_00F8
I2CLK	I2C_BA+0x10	R/W	I^2C Clock Divider Register	0x0000_0000
I2CTOC	I2C_BA+0x14	R/W	I^2C Time-Out Control Register	0x0000_0000
I2CADDR1	I2C_BA+0x18	R/W	I^2C Slave Address Register 1	0x0000_0000
I2CADDR2	I2C_BA+0x1C	R/W	I^2C Slave Address Register 2	0x0000_0000
I2CADDR3	I2C_BA+0x20	R/W	I^2C Slave Address Register 3	0x0000_0000
I2CADM0	I2C_BA+0x24	R/W	I^2C Slave Address Mask Register 0	0x0000_0000
I2CADM1	I2C_BA+0x28	R/W	I^2C Slave Address Mask Register 1	0x0000_0000
I2CADM2	I2C_BA+0x2C	R/W	I^2C Slave Address Mask Register 2	0x0000_0000
I2CADM3	I2C_BA+0x30	R/W	I^2C Slave Address Mask Register 3	0x0000_0000

5.12.4 Register Description

I²C Control Register (I2CON)

Register	Offset	R/W	Description	Reset Value
I2CON	I2C_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	Descriptions	
[31:8]	**Reserved**	Reserved
[7]	**EI**	**Enable Interrupt** 1 = Enable I²C interrupt 0 = Disable I²C interrupt
[6]	**ENS1**	**I²C Controller Enable Bit** 1 = Enable 0 = Disable Set to enable I²C serial function controller. When ENS1=1 the I²C serial function enables. The multi-function pin function of SDA and SCL must set to I²C function first.
[5]	**STA**	**I²C START Control Bit** Setting STA to logic 1 to enter master mode, the I²C hardware sends a START or repeat START condition to bus when the bus is free.
[4]	**STO**	**I²C STOP Control Bit** In master mode, setting STO to transmit a STOP condition to bus then I²C hardware will check the bus condition if a STOP condition is detected this bit will be cleared by hardware automatically. In a slave mode, setting STO resets I²C hardware to the defined “not addressed” slave mode. This means it is NO LONGER in the slave receiver mode to receive data from the master transmit device.
[3]	**SI**	**I²C Interrupt Flag** When a new I²C state is present in the I2CSTATUS register, the SI flag is set by hardware, and if bit EI (I2CON [7]) is set, the I²C interrupt is requested. SI must be

		cleared by software. Clear SI 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

I²C Data Register (I2CDAT)

Register	Offset	R/W	Description	Reset Value
I2CDAT	I2C_BA+0x08	R/W	I²C Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CDAT[7:0]							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7:0]	**I2CDAT**	**I²C Data Register** Bit [7:0] is located with the 8-bit transferred data of I²C serial port.

I²C Status Register (I2CSTATUS)

Register	Offset	R/W	Description	Reset Value
I2CSTATUS	I2C_BA+0x0C	R/W	I²C Status Register	0x0000_00F8

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CSTATUS[7:3]					**0**	**0**	**0**

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7:0]	**I2CSTATUS**	**I²C Status Register** The status register of I²C: 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 I²C 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 illegal position in the formation frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit.

I²C Clock Divider Register (I2CLK)

Register	Offset	R/W	Description	Reset Value
I2CLK	I2C_BA+0x10	R/W	I²C Clock Divider Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CLK[7:0]							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7:0]	**I2CLK**	**I²C Clock Divider Register** The I²C clock rate bits: Data Baud Rate of I²C = (system clock) / (4x (I2CLK+1)).

I²C Time-Out Counter Register (I2CTOC)

Register	Offset	R/W	Description	Reset Value
I2CTOC	I2C_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	Descriptions	
[31:3]	**Reserved**	Reserved
[2]	**ENTI**	**Time-out Counter Enable** 1 = Enable 0 = Disable When Enable, the 14 bit time-out counter will start counting when SI is clear. Setting flag SI to high will reset counter and re-start up counting after SI is cleared.
[1]	**DIV4**	**Time-Out Counter Input Clock is Divided by 4** 1 = Enable 0 = Disable When Enable, The time-Out period is extend 4 times.
[0]	**TIF**	**Time-Out Flag** This bit is set by H/W when I²C time-out happened and it can interrupt CPU if I²C interrupt enable bit (EI) is set to 1. S/W can write 1 to clear this bit.

I²C Slave Address Register (I2CADDRx)

Register	Offset	R/W	Description	Reset Value
I2CADDR0	I2C_BA+0x04	R/W	I²C Slave Address Register 0	0x0000_0000
I2CADDR1	I2C_BA+0x18	R/W	I²C Slave Address Register 1	0x0000_0000
I2CADDR2	I2C_BA+0x1C	R/W	I²C Slave Address Register 2	0x0000_0000
I2CADDR3	I2C_BA+0x20	R/W	I²C Slave Address 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**
I2CADDR[7:1]							**GC**

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7:1]	**I2CADDR**	**I²C Address Register** The content of this register is irrelevant when I²C is in master mode. In the slave mode, the seven most significant bits must be loaded with the chip's own address. The I²C hardware will react if either of the address is matched.
[0]	**GC**	**General Call Function** 0 = Disable General Call Function. 1 = Enable General Call Function.

I²C Slave Address Mask Register (I2CADMx)

Register	Offset	R/W	Description	Reset Value
I2CADM0	I2C_BA+0x24	R/W	I²C Slave Address Mask Register 0	0x0000_0000
I2CADM1	I2C_BA+0x28	R/W	I²C Slave Address Mask Register 1	0x0000_0000
I2CADM2	I2C_BA+0x2C	R/W	I²C Slave Address Mask Register 2	0x0000_0000
I2CADM3	I2C_BA+0x30	R/W	I²C Slave Address Mask 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**
I2CADMx[7:1]							**Reserved**

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7:1]	**I2CADMx**	**I²C Address Mask Register** 1 = Mask enable (the received corresponding address bit is don't care.) 0 = Mask disable (the received corresponding register bit should be exact the same as address register.) I²C bus controllers support multiple address recognition with four address mask register. When the bit in the address mask register is set to one, it means the received corresponding address bit is don't-care. If the bit is set to zero, that means the received corresponding register bit should be exact the same as address register.
[0]	**Reserved**	Reserved

5.12.5 Modes of Operation

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 the slave mode, the I^2C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master(by setting the AA bit), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, the hardware waits until the bus is free before the master mode is entered so that a possible slave action didn't be interrupted. If bus arbitration is lost in the master mode, I^2C port switches to the slave mode immediately and can detect its own slave address in the same serial transfer.

5.12.5.1 Master Transmitter Mode

Serial data output through SDA while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) 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 the flow diagrams. Thus the first byte transmitted is SLA+W. Serial data is transmitted 8 bits 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.

5.12.5.2 Master Receiver Mode

In this case the data direction bit (R/W) will be logic 1, and it is represented by "R" in the flow diagrams. 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 bits 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.

5.12.5.3 Slave Receiver 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.

5.12.5.4 Slave Transmitter Mode

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.

5.12.6 Data Transfer Flow in Five Operating Modes

The five operating modes are: Master/Transmitter, Master/Receiver, Slave/Transmitter, Slave/Receiver and GC Call. 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.

Data transfers in each mode are shown in the following figures.

*** Legend for the following five figures:

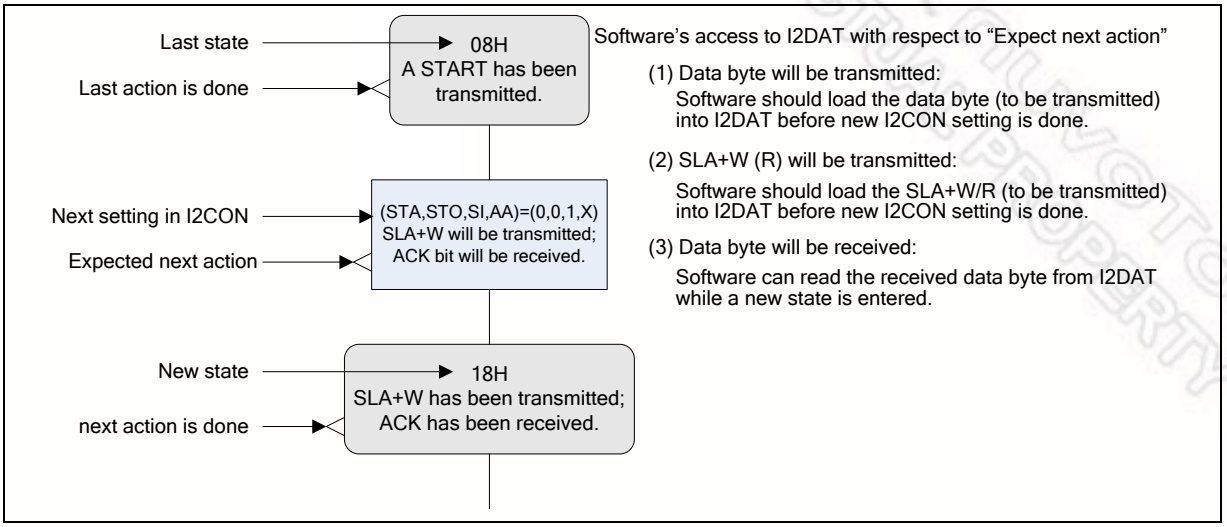


Figure 5-54 Legend for the following four figures

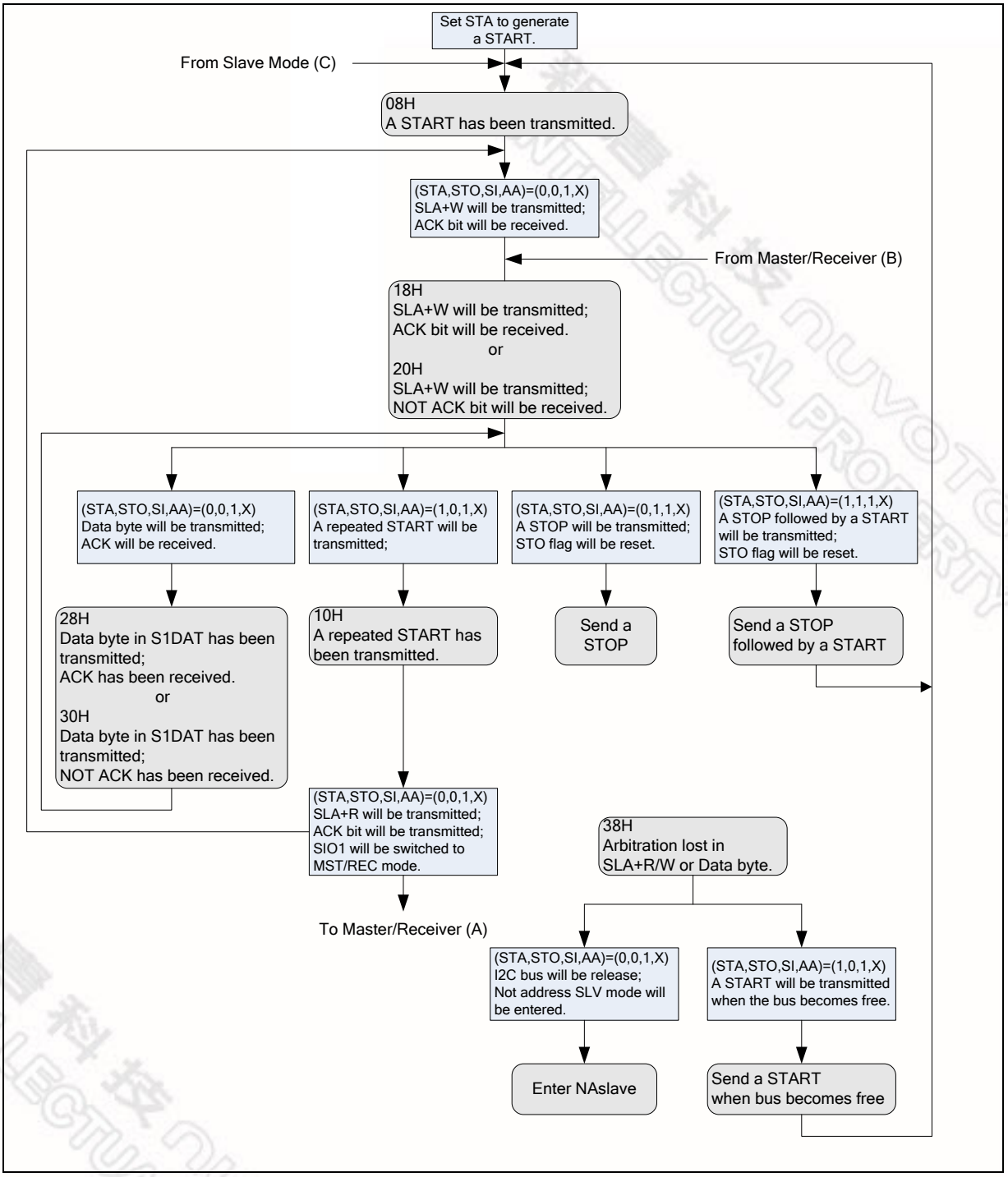


Figure 5-55 Master Transmitter Mode

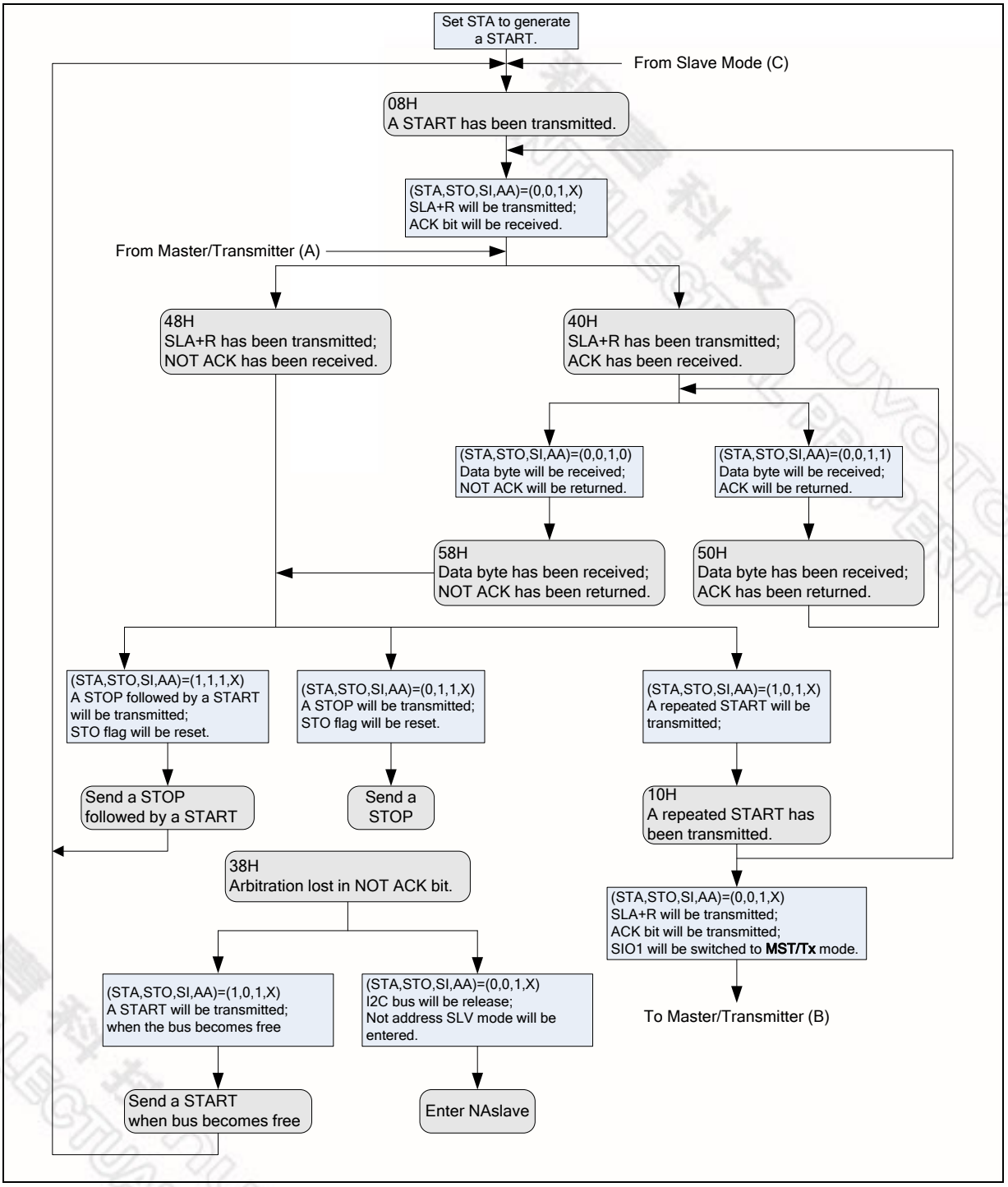


Figure 5-56 Master Receiver Mode

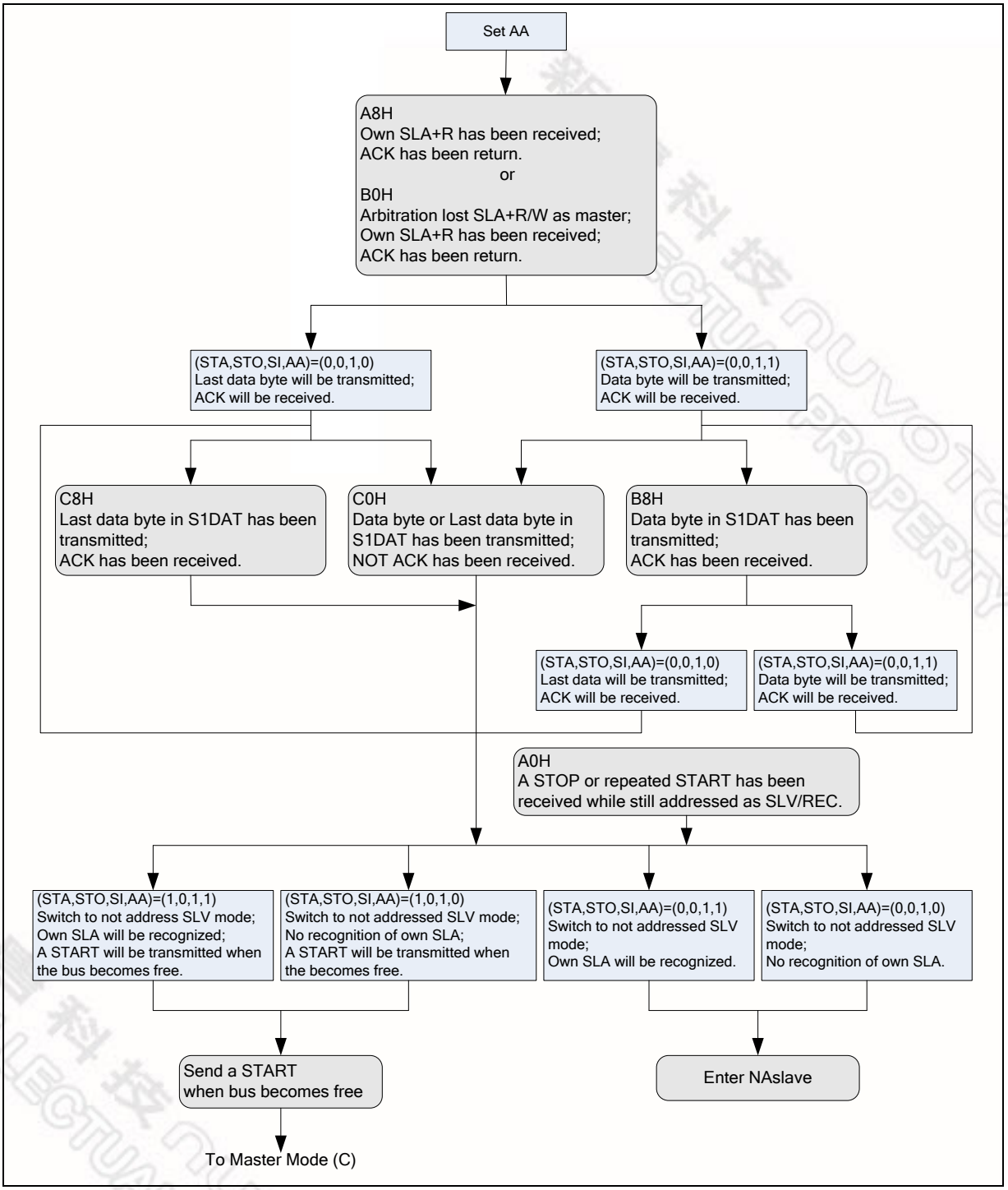


Figure 5-57 Slave Transmitter Mode

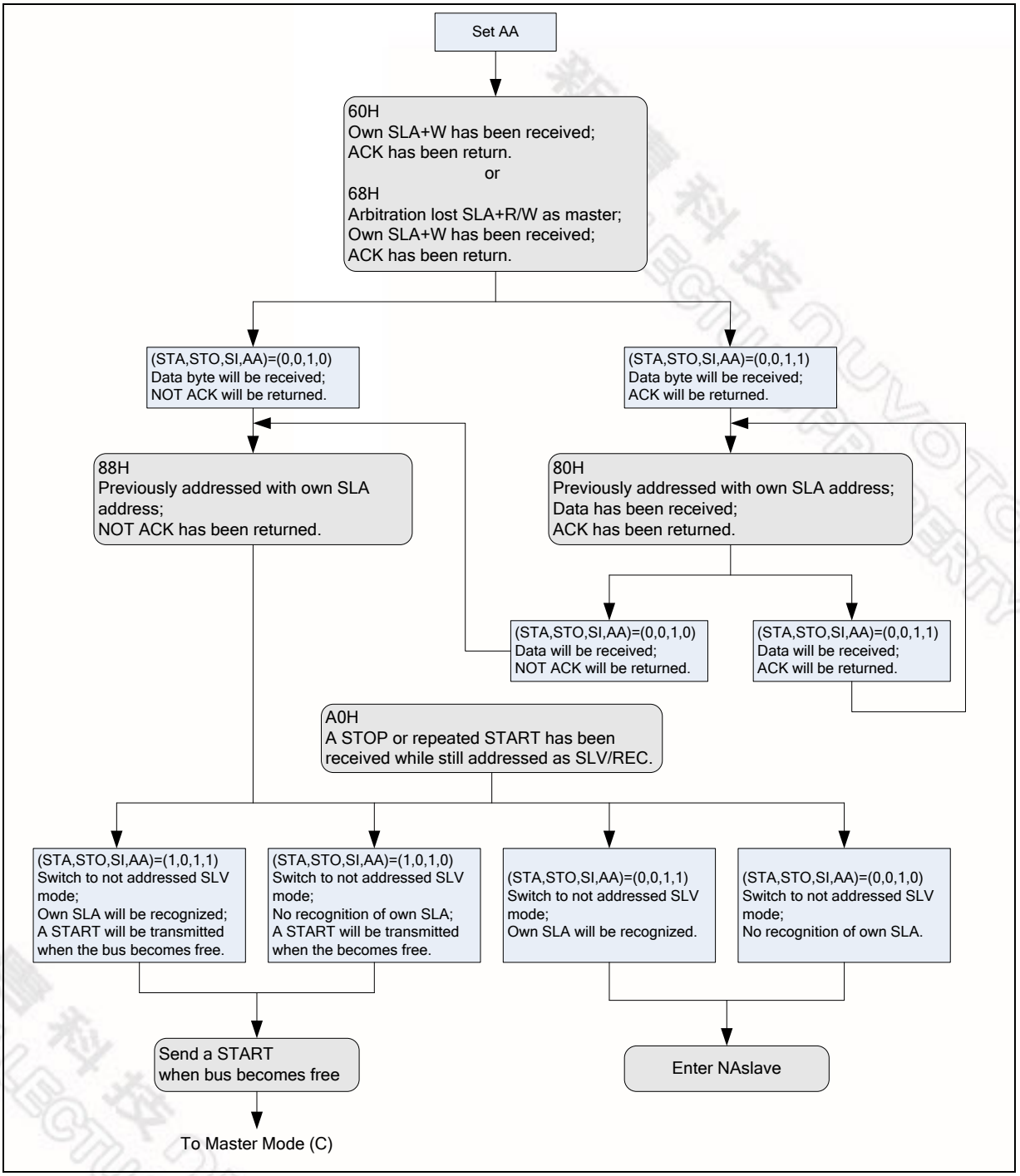


Figure 5-58 Slave Receiver Mode

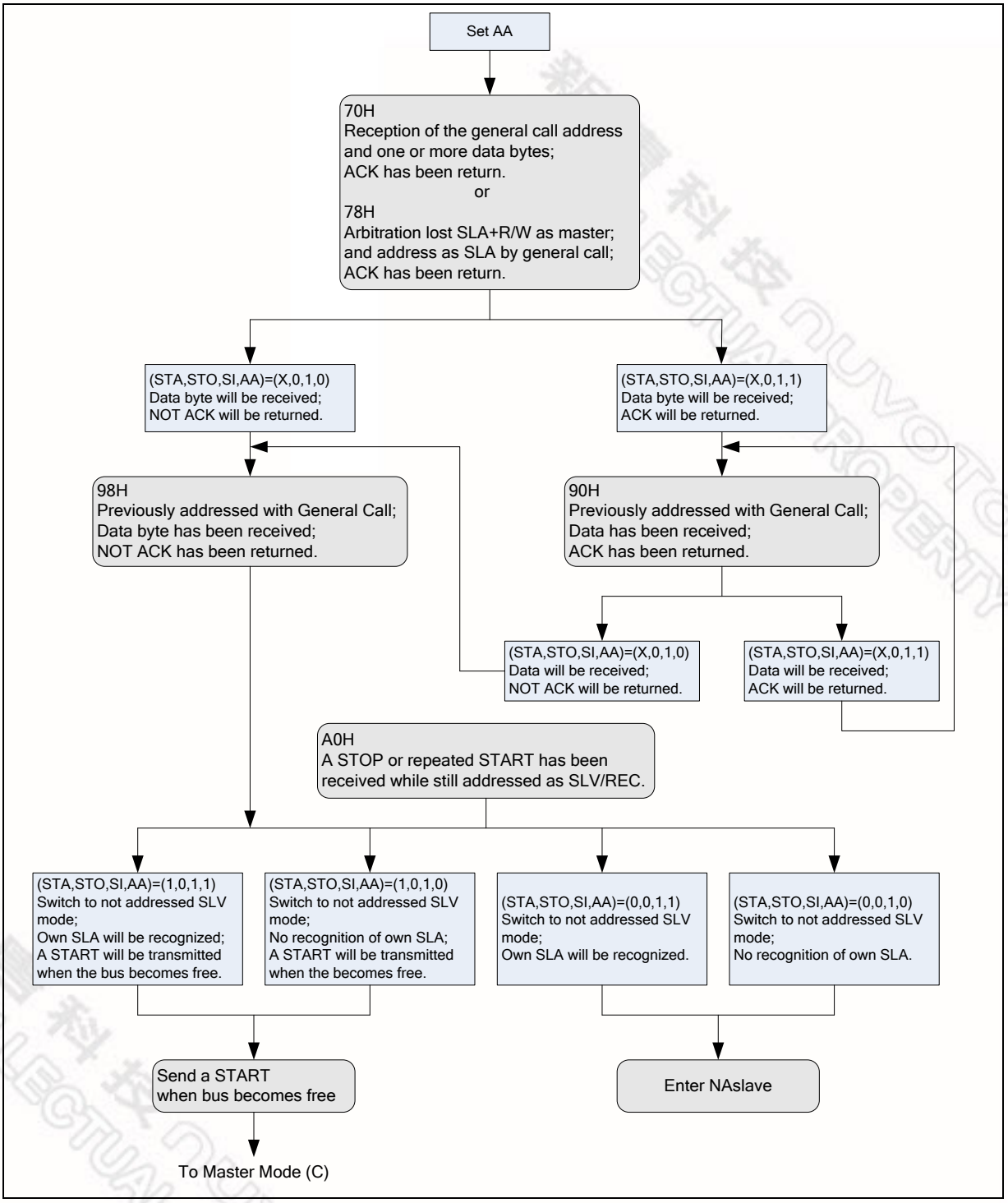


Figure 5-59 GC Mode

5.13 Serial Peripheral Interface (SPI)

5.13.1 Overview

The Serial Peripheral Interface (SPI) is a synchronous serial data communication protocol which operates in full duplex mode. Devices communicate in master/slave mode with 4-wire bi-direction interface. The NuMicro™ NUC122 contains up to two sets of SPI controller performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. Each set of SPI controller can be set as a master that can drive up to 2 external peripheral slave devices; it also can be configured as a slave device controlled by an off-chip master device.

This controller supports a variable serial clock for special application.

5.13.2 Features

- Up to two sets of SPI controller for NuMicro™ NUC122
- Support master or slave mode operation
- Support 1-bit transfer mode
- Configurable bit length up to 32 bits of a transfer word and configurable word numbers up to 2 of a transaction, so the maximum bit length is 64 bits for each data transfer
- Provide burst mode operation, transmit/receive can be transferred up to two times word transaction in one transfer
- Support MSB or LSB first transfer
- 2 device/slave select lines in master mode, but 1 device/slave select line in slave mode
- Support byte reorder in data register
- Support byte or word suspend mode
- Variable output serial clock frequency in master mode
- Support two programmable serial clock frequencies in master mode

5.13.3 Block Diagram

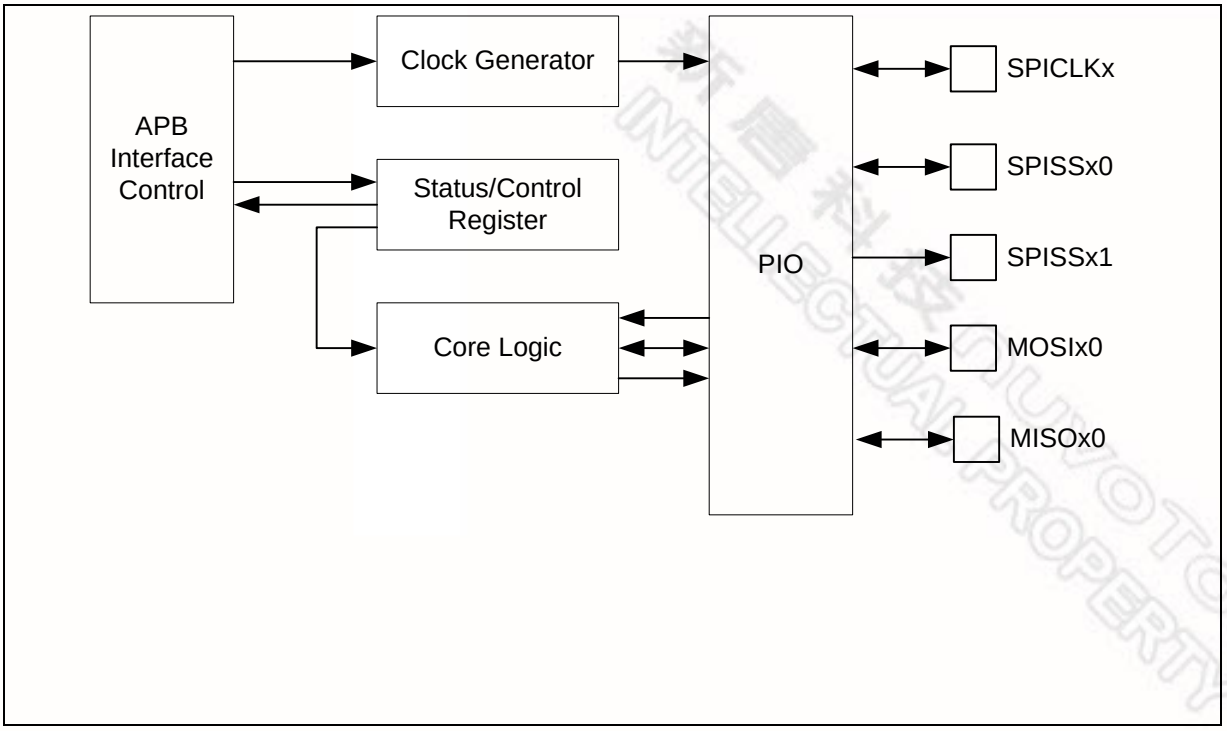


Figure 5-60 SPI Block Diagram

5.13.4 Function Description

Master/Slave Mode

This 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 as below. This SPI controller does not support multi-slave in SPI bus if the controller is set as slave mode. In slave mode, the SPI clock pin must be kept at idle state when the slave select pin is at inactive state.

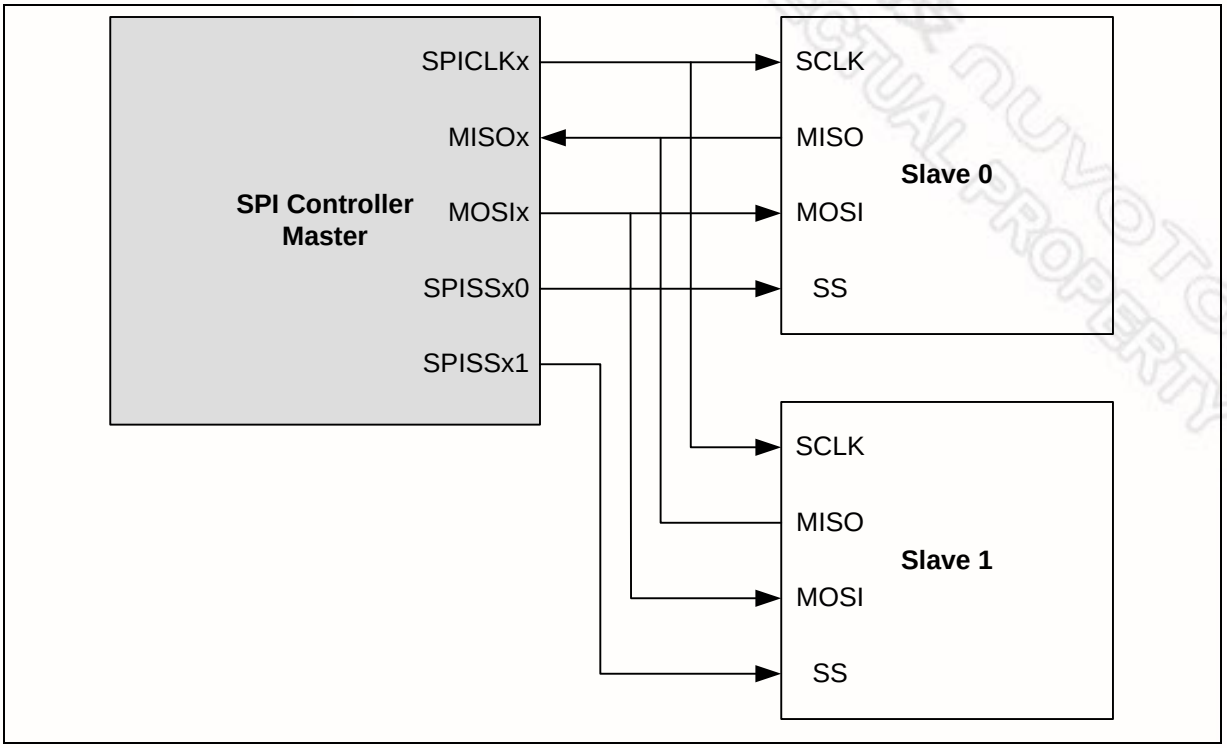


Figure 5-61 SPI Master Mode Application Block Diagram

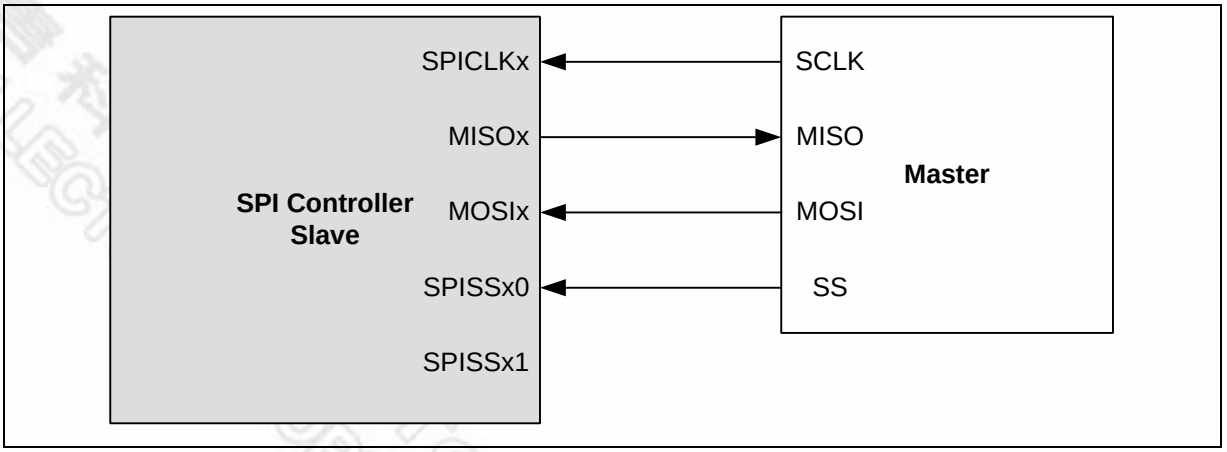


Figure 5-62 SPI Slave Mode Application Block Diagram

Slave Select

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 active 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 trigger or edge trigger. The selection of trigger condition 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 both the received number and received bits met the requirement which defines in TX_NUM and TX_BIT_LEN among one transaction done (the transaction done means the slave select has deactivated or the SPI controller has finished one data transfer).

Level-trigger / Edge-trigger

In slave mode, the slave select signal can be configured as level-trigger or edge-trigger. In edge-trigger, the data transfer starts from an active edge and ends on an inactive edge. If master does not send an inactive edge to slave, the transfer procedure will not be completed and the interrupt flag of slave will not be set. In level-trigger, the following two conditions will terminate the transfer procedure and the interrupt flag of slave will be set. The first condition, if master set the slave select pin to inactive level, it will force slave device to terminate the current transfer no matter how many bits have been transferred and the interrupt flag will be set. User can read the status of LTRIG_FLAG bit to check if the data has been completely transferred. The second condition is that if the number of transferred bits matches the settings of TX_NUM and TX_BIT_LEN, the interrupt flag of slave will be set.

Automatic Slave Select

In master mode, if the bit AUTOSS (SPI_SSR[3]) is set, the slave select signals will be generated automatically and output to SPISSx0 and SPISSx1 pins according to SSR[0] (SPI_SSR[0]) and SSR[1] (SPI_SSR[1]) whether be enabled or not. It means that the slave select signals, which are selected in SSR[1:0], will be asserted by the SPI controller when transmit/receive 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]).

Serial Clock

In master mode, set the DIVIDER1 bits (SPI_DIVIDER[15:0]) to program the output frequency of serial clock to the SPICLK output port. It also supports a variable serial clock if the VARCLK_EN bit (SPI_CTL[23]) is enabled. In this case, the output frequency of serial clock can be programmed as one of the two different frequencies which depend on the value of DIVIDER1 (SPI_DIVIDER[15:0]) and DIVIDER2 (SPI_DIVIDER[31:16]). The serial clock rate of each cycle is depended on the setting of the SPI_VARCLK register.

In slave mode, the off-chip master device drives the serial clock through the SPICLK input port to this SPI controller.

Variable Serial Clock Frequency

In master mode, the output of serial clock can be programmed as variable frequency pattern if the Variable Clock Enable bit VARCLK_EN (SPI_CNTRL[23]) is enabled. The frequency pattern format is defined in VARCLK (SPI_VARCLK[31:0]) register. If the bit content of VARCLK is '0' the output frequency is according with the DIVIDER (SPI_DIVIDER[15:0]) and if the bit content of VARCLK is '1', the output frequency is according to the DIVIDER2 (SPI_DIVIDER[31:16]). The following figure is the timing relationship among the serial clock (SPICLK), the VARCLK, the DIVIDER and the DIVIDER2 registers. A two-bit combination in the VARCLK defines one clock cycle. The bit field VARCLK[31:30] defines the first clock cycle of SPICLK. The bit field VARCLK[29:28] defines the second clock cycle of SPICLK and so on. The clock source selections are defined in VARCLK and it must be set 1 cycle before the next clock option. For example, if there are 5 CLK1 cycle in SPICLK, the VARCLK shall set 9 '0' in the MSB of VARCLK. The 10th shall be set as '1' in order to switch the next clock source is CLK2. Note that when enable the VARCLK_EN bit, the setting of TX_BIT_LEN must be programmed as 0x10 (16 bits mode only).

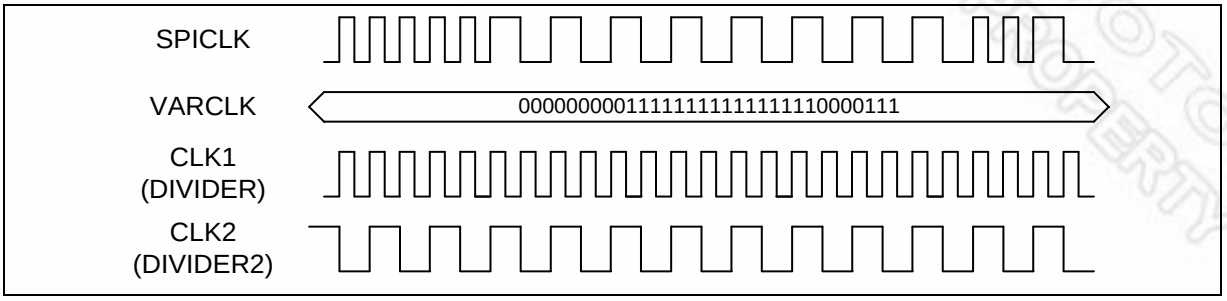


Figure 5-63 Variable Serial Clock Frequency

Clock Polarity

The CLKP bit (SPI_CTL[11]) defines the serial clock idle state. If CLKP = 1, the output SPICLK is idle at high state, otherwise it is at low state if CLKP = 0.

Transmit/Receive Bit Length

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

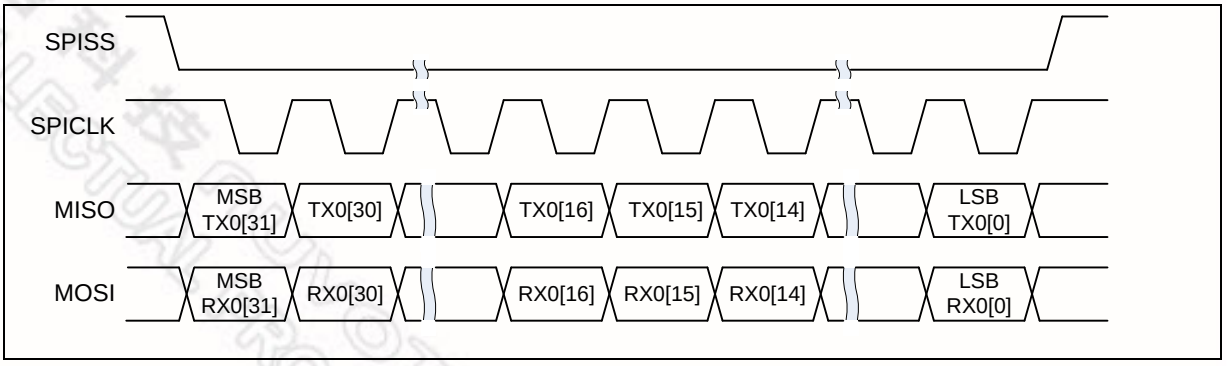


Figure 5-64 32-Bit in one Transaction

Burst Mode

SPI controller can switch to burst mode by setting TX_NUM bit field (SPI_CNTRL[9:8]) to 0x01. In burst mode, SPI can transmit/receive two transactions in one transfer. The SPI burst mode waveform is showed below:

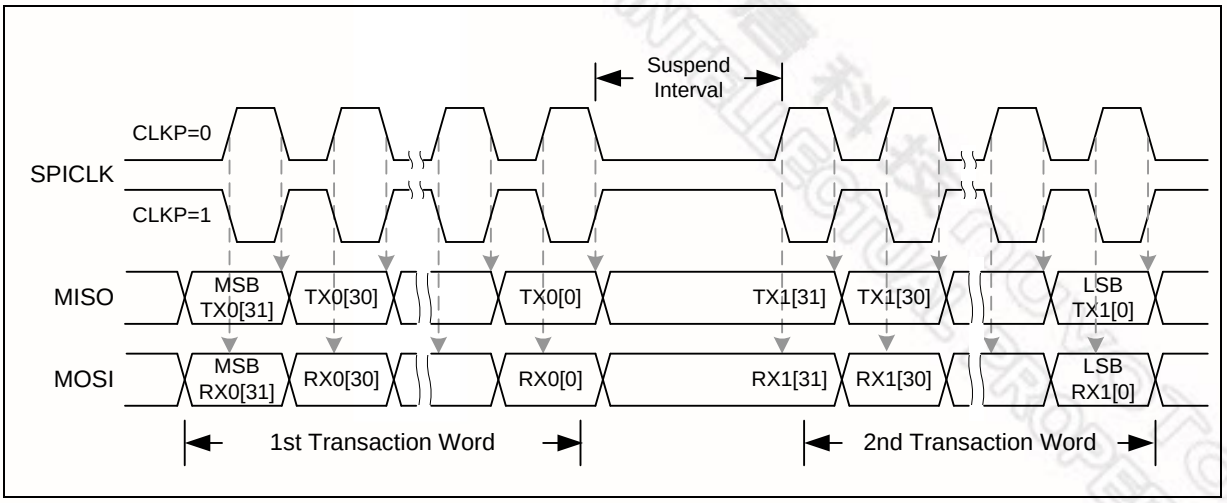


Figure 5-65 Two Transactions in One Transfer (Burst Mode)

LSB First

The LSB bit (SPI_CNTRL[10]) defines the data transmission either from LSB or MSB firstly to start to transmit/receive data.

Transmit Edge

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

Receive Edge

The Rx_NEG bit (SPI_CNTRL[1]) defines the data received in either at negative edge or at positive edge of serial clock SPICLK.

Note: the settings of TX_NEG and RX_NEG are mutual exclusive. In other words, don't transmit and receive data at the same clock edge.

Word Suspend

These four bits field of SP_CYCLE (SPI_CNTRL[15:12]) provide a configurable suspend interval 2 ~ 17 serial clock periods between two successive transaction words in master mode. The suspend interval is from the last falling clock edge of the preceding transaction word to the first rising clock edge of the following transaction word if CLKP = 0. If CLKP = 1, the interval is from the rising clock edge of the preceding transaction word to the falling clock edge of the following transaction word. The default value of SP_CYCLE is 0x0 (2 serial clock cycles), but set these bits field has no any effects on data transaction process if TX_NUM = 0x00.

Byte Reorder

When the transfer is set as MSB first (LSB = 0) and the REORDER is enabled, the data stored in the TX buffer and RX buffer will be rearranged in the order as [BYTE0, BYTE1, BYTE2, BYTE3] in TX_BIT_LEN = 32-bit mode, and the sequence of transmitted/received data will be BYTE0, BYTE1, BYTE2, and then BYTE3. If the TX_BIT_LEN is set as 24-bit 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.

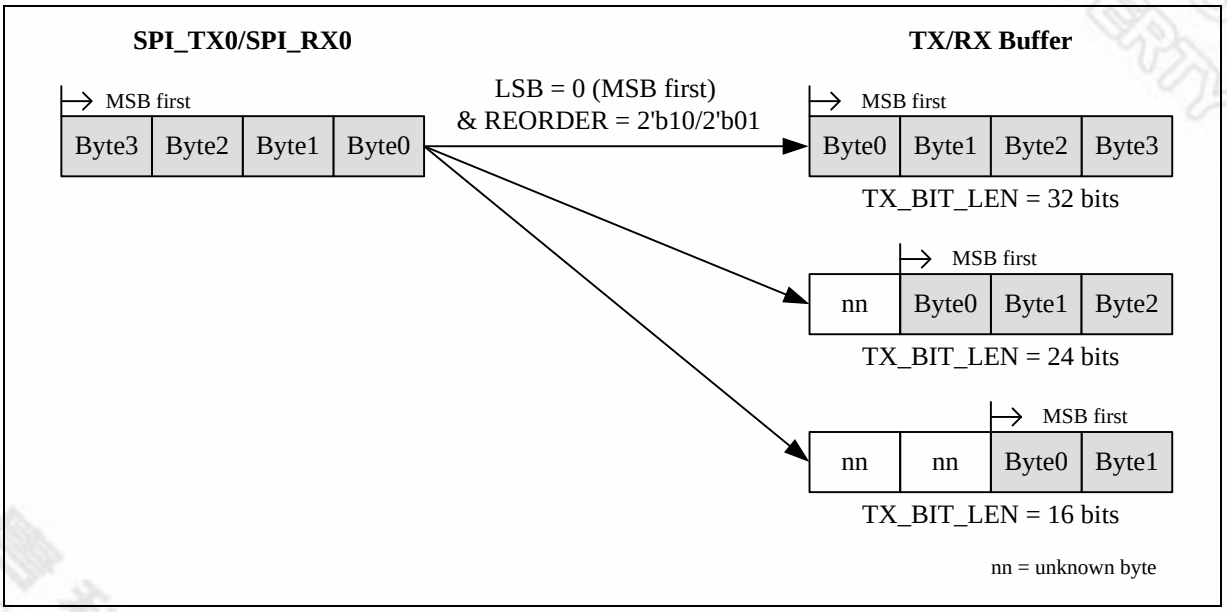


Figure 5-66 Byte Reorder

Byte Suspend

In master mode, if SPI_CNTRL[19] is set to 1, the hardware will insert a suspend interval 2 ~ 17 serial clock periods between two successive bytes in a transaction word. Both settings of byte suspend and word suspend are configured in SP_CYCLE. Note that when enable the byte suspend function, the setting of TX_BIT_LEN must be programmed as 0x00 only (32 bits per transaction word).

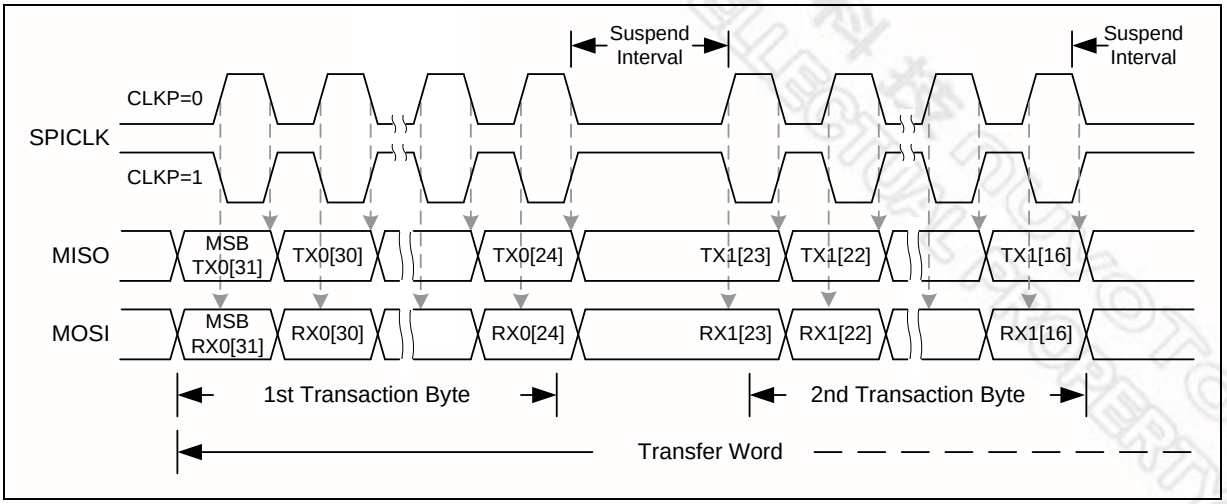


Figure 5-67 Timing Waveform for Byte Suspend

REORDER	Description
00	Disable both byte reorder function and byte suspend interval.
01	Enable byte reorder function and insert a byte suspend internal (2~17 SPICLK) among each byte. The setting of TX_BIT_LEN must be configured as 0x00 (32 bits/ word)
10	Enable byte reorder function but disable byte suspend function
11	Disable byte reorder function, but insert a suspend interval (2~17 SPICLK) among each byte. The setting of TX_BIT_LEN must be configured as 0x00 (32 bits/ word)

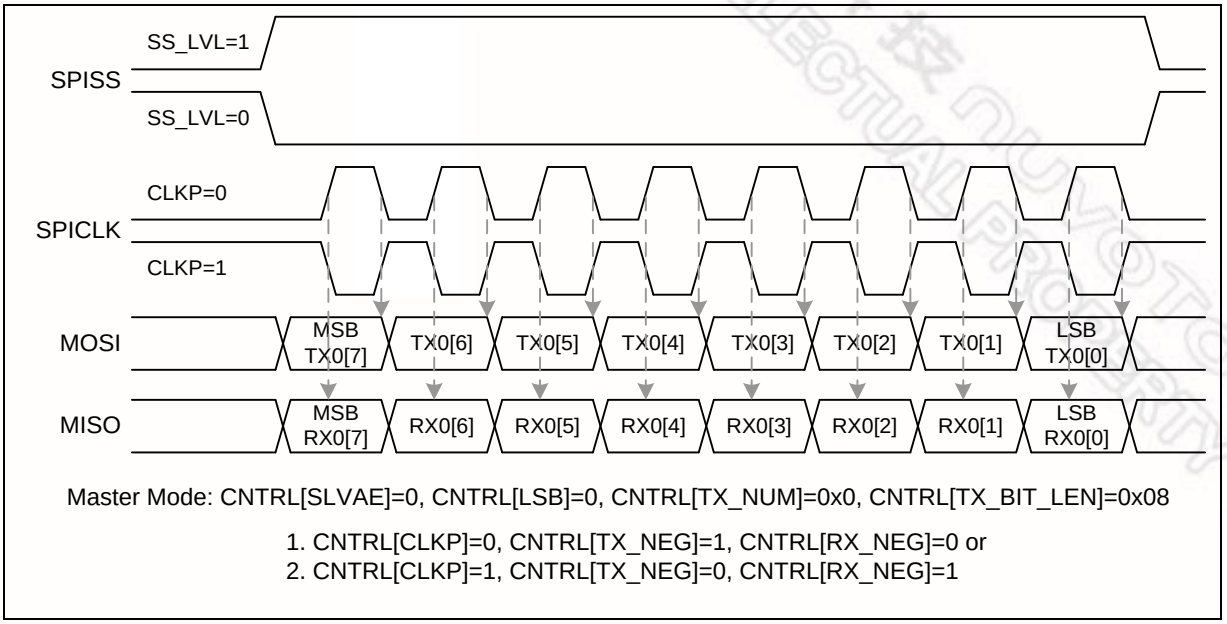
Table 5-11 Byte Order and Byte Suspend Conditions

Interrupt

Each SPI controller can generates an individual interrupt when data transfer is finished and the respective interrupt event flag IF (SPI_CNTRL[16]) will be set. The interrupt event flag will generates an interrupt to CPU if the interrupt enable bit IE (SPI_CNTRL[17]) is set. The interrupt event flag IF can be cleared only by writing 1 to it.

Timing Diagram

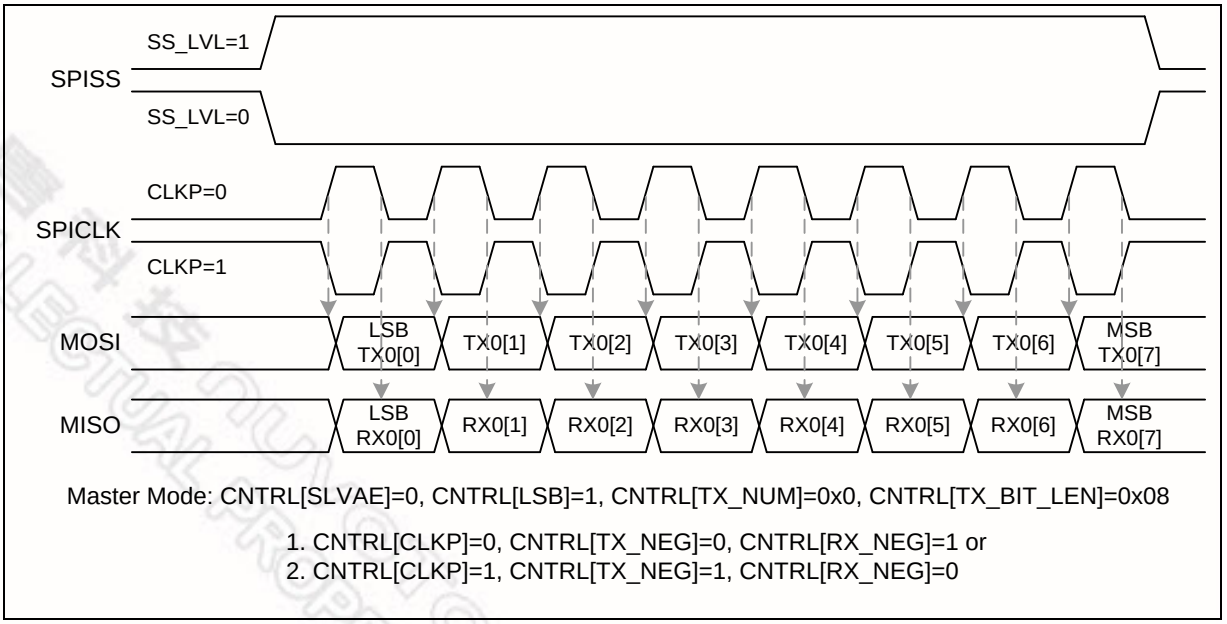
The active state of slave select signal can be defined by the settings of SS_LVL bit (SPI_SSR[2]) and SS_LTRIG bit (SPI_SSR[4]). The serial clock (SPICLK) idle state can be configured as high state or low state by setting the CLKP bit (SPI_CNTRL[11]). It also provides the bit length of a transaction word in TX_BIT_LEN (SPI_CNTRL[7:3]), the transfer number in TX_NUM (SPI_CNTRL[8]), and transmit/receive data from MSB or LSB first in LSB bit (SPI_CNTRL[10]). Users also can select which edge of serial clock to transmit/receive data in TX_NEG/RX_NEG (SPI_CNTRL[2:1]) registers. Four SPI timing diagrams for master/slave operations and the related settings are shown as below.



Master Mode: CNTRL[SLVAE]=0, CNTRL[LSB]=0, CNTRL[TX_NUM]=0x0, CNTRL[TX_BIT_LEN]=0x08

1. CNTRL[CLKP]=0, CNTRL[TX_NEG]=1, CNTRL[RX_NEG]=0 or
2. CNTRL[CLKP]=1, CNTRL[TX_NEG]=0, CNTRL[RX_NEG]=1

Figure 5-68 SPI Timing in Master Mode



Master Mode: CNTRL[SLVAE]=0, CNTRL[LSB]=1, CNTRL[TX_NUM]=0x0, CNTRL[TX_BIT_LEN]=0x08

1. CNTRL[CLKP]=0, CNTRL[TX_NEG]=0, CNTRL[RX_NEG]=1 or
2. CNTRL[CLKP]=1, CNTRL[TX_NEG]=1, CNTRL[RX_NEG]=0

Figure 5-69 SPI Timing in Master Mode (Alternate Phase of SPICLK)

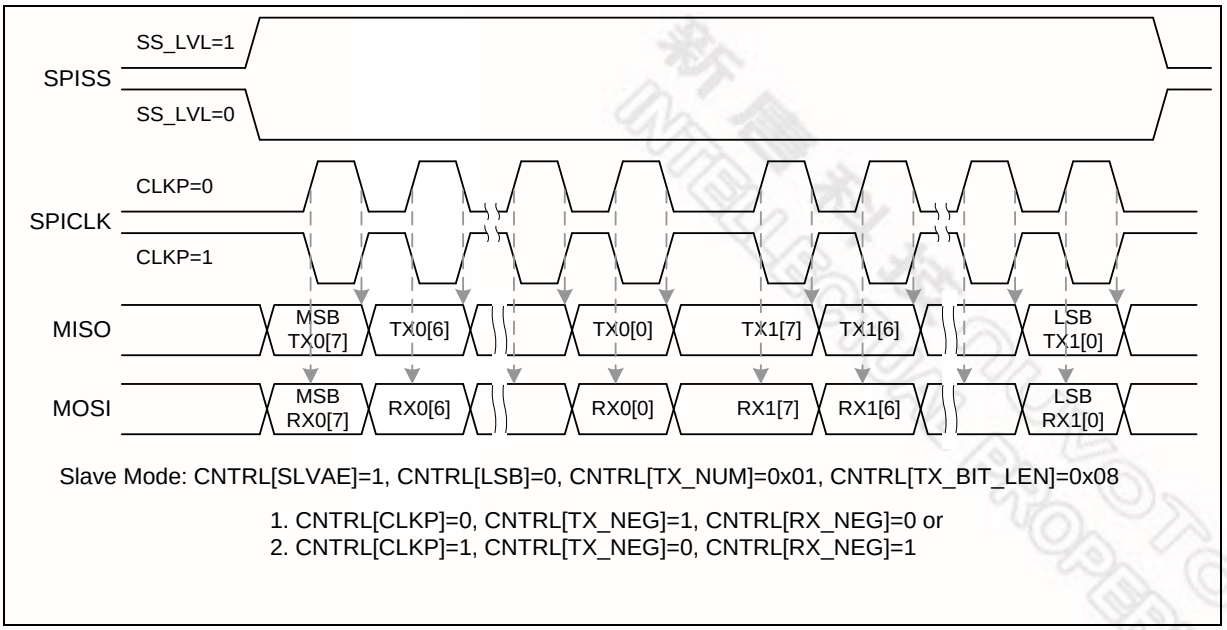


Figure 5-70 SPI Timing in Slave Mode

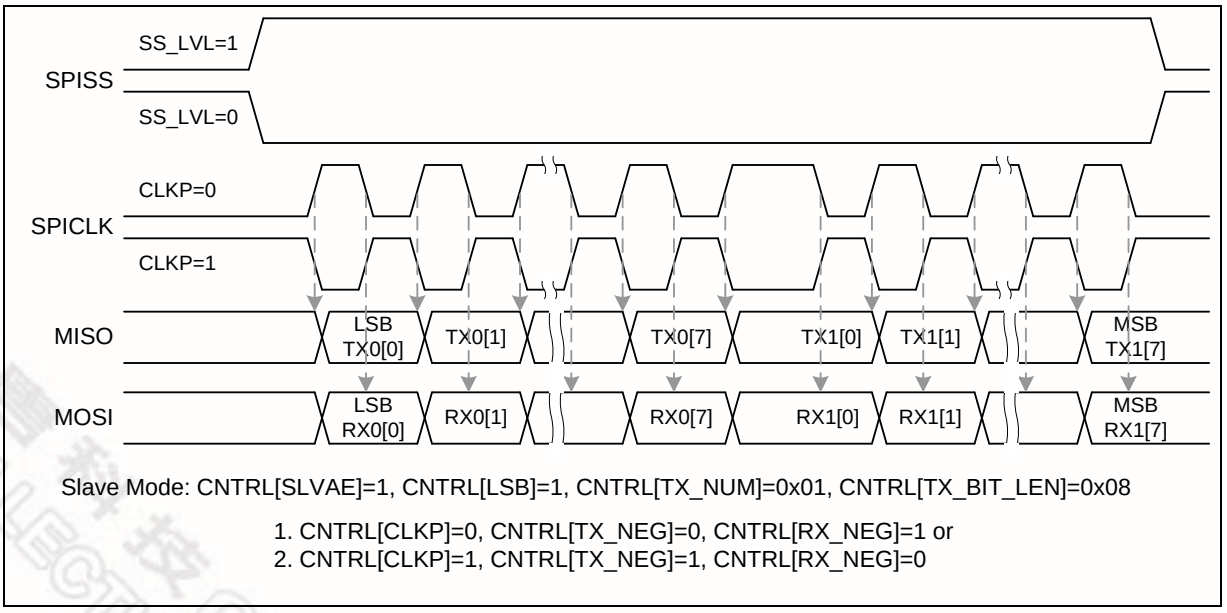


Figure 5-71 SPI Timing in Slave Mode (Alternate Phase of SPICLK)

5.13.5 Programming Examples

Example 1, SPI controller is set as a master to access an off-chip slave device with following specifications:

- Data bit is latched on positive edge of serial clock
- Data bit is driven on negative edge of serial 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
- Use the first SPI slave select pin to connect with an off-chip slave device. Slave select signal is active low

The operation flow is as follows.

1) Set the DIVIDER (SPI_DIVIDER [15:0]) register to determine the output frequency of serial clock.
2) 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)
    3. Select slave select signal to be output active at the IO pin by setting the Slave Select Register bits SSR[0] (SPI_SSR[0]) to active the off-chip slave devices
3) Write the related settings into the SPI_CNTRL register to control this SPI master actions
    1. Set this SPI controller as master device in SLAVE bit (SPI_CNTRL[18] = 0)
    2. Force the serial clock idle state at low in CLKP bit (SPI_CNTRL[11] = 0)
    3. Select data transmitted at negative edge of serial clock in TX_NEG bit (SPI_CNTRL[2] = 1)
    4. Select data latched at positive edge of serial 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 only one time of word transfer in TX_NUM (SPI_CNTRL[9:8] = 0x0)
    7. Set MSB transfer first in MSB bit (SPI_CNTRL[10] = 0), and don't care the SP_CYCLE bit field (SPI_CNTRL[15:12]) due to it's not in burst mode in this case
4) If this SPI master will transmits (writes) one byte data to the off-chip slave device, write the byte data that will be transmitted into the TX0[7:0] (SPI_TX0[7:0]) register.
5) If this SPI master just only receives (reads) one byte data from the off-chip slave device, you don't need to care what data will be transmitted and just write 0xFF into the SPI_TX0[7:0] register.
6) Enable the GO_BUSY bit (SPI_CNTRL [0] = 1) to start the data transfer at the SPI interface.
7) Waiting for SPI interrupt occurred (if the Interrupt Enable IE bit is set) or just polling the GO_BUSY bit till it is cleared to 0 by hardware automatically.
8) Read out the received one byte data from RX0 [7:0] (SPI_RX0[7:0]) register.
9) Go to 4) to continue another data transfer or set SSR [0] to 0 to inactivate the off-chip slave

devices.

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 serial clock
- Data bit is driven on negative edge of serial 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 this SPI controller as slave device in SLAVE bit (SPI_CNTRL[18] = 1)
   2. Select the serial clock idle state at high in CLKP bit (SPI_CNTRL[11] = 1)
   3. Select data transmitted at negative edge of serial clock in TX_NEG bit (SPI_CNTRL[2] = 1)
   4. Select data latched at positive edge of serial 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 only one time of word transfer in TX_NUM (SPI_CNTRL[9:8] = 0x0)
   7. Set LSB transfer first in LSB bit (SPI_CNTRL[10] = 1), and don't care the SP_CYCLE bit field (SPI_CNTRL[15:12]) due to not burst mode in this case.

3) If this SPI slave will transmits (be read) one byte data to the off-chip master device, write the byte data that will be transmitted into the TX0 [7:0] (SPI_TX0[7:0]) register.

4) If this SPI slave just only receives (be written) one byte data from the off-chip master device, you don't care what data will be transmitted and just write 0xFF into the SPI_TX0[7:0] register.

5) Enable the GO_BUSY bit (SPI_CNTRL[0] = 1) to wait for the slave select trigger input and serial clock input from the off-chip master device to start the data transfer at the SPI interface.

   Waiting for SPI interrupt occurred (if the Interrupt Enable IE bit is set), or just polling the GO_BUSY bit till it is cleared to 0 by hardware automatically.

6) Read out the received one byte data from RX[7:0] (SPI_RX0[7:0]) register

Go to 3) to continue another data transfer or disable the GO_BUSY bit to stop data transfer.

5.13.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI0_BA = 0x4003_0000 **SPI1_BA = 0x4003_4000**				
SPI_CNTRL	SPIx_BA+0x00	R/W	Control and Status Register	0x0500_0004
SPI_DIVIDER	SPIx_BA+0x04	R/W	Clock Divider Register	0x0000_0000
SPI_SSR	SPIx_BA+0x08	R/W	Slave Select Register	0x0000_0000
SPI_RX0	SPIx_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPI_RX1	SPIx_BA+0x14	R	Data Receive Register 1	0x0000_0000
SPI_TX0	SPIx_BA+0x20	W	Data Transmit Register 0	0x0000_0000
SPI_TX1	SPIx_BA+0x24	W	Data Transmit Register 1	0x0000_0000
SPI_VARCLK	SPIx_BA+0x34	R/W	Variable Clock Pattern Register	0x007F_FF87

Note: When software programs CNTRL, the GO_BUSY bit should be written last.

5.13.7 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_0004

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
VARCLK_EN	SBZ		REORDER		SLAVE	IE	IF
15	14	13	12	11	10	9	8
SP_CYCLE				CLKP	LSB	TX_NUM	
7	6	5	4	3	2	1	0
TX_BIT_LEN					TX_NEG	RX_NEG	GO_BUSY

Bits	Descriptions	
[31:24]	**Reserved**	Reserved
[23]	**VARCLK_EN**	**Variable Clock Enable** (Master only) 1 = The serial clock output frequency is variable. The output frequency is decided by the value of VARCLK, DIVIDER, and DIVIDER2. 0 = The serial clock output frequency is fixed and decided only by the value of DIVIDER. Note that when enable this VARCLK_EN bit, the setting of TX_BIT_LEN must be programmed as 0x10 (16 bits mode)
[22:21]	**SBZ**	**Note:** This bit must always be kept 0. If set to 1, the result is unpredictable
[20:19]	**REORDER**	**Reorder Mode Select** 00 = Disable both byte reorder and byte suspend functions. 01 = Enable byte reorder function and insert a byte suspend interval (2~17 SPICLK cycles) among each byte. The setting of TX_BIT_LEN must be configured as 0x00. (32 bits/word) 10 = Enable byte reorder function, but disable byte suspend function. 11 = Disable byte reorder function, but insert a suspend interval (2~17 SPICLK cycles) among each byte. The setting of TX_BIT_LEN must be configured as 0x00. (32 bits/word) 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, if the byte suspend function is enabled, the slave select pin must be kept at active state during the successive four bytes transfer.

[18]	**SLAVE**	**Slave Mode Indication** 1 = Slave mode 0 = Master mode
[17]	**IE**	**Interrupt Enable** 1 = Enable SPI Interrupt 0 = Disable SPI Interrupt
[16]	**IF**	**Interrupt Flag** 1 = It indicates that the transfer is done. 0 = It indicates that the transfer dose not finish yet. Note: This bit is cleared by writing 1 to itself.
[15:12]	**SP_CYCLE**	**Suspend Interval** (Master only) These four bits provide configurable suspend interval between two successive transmit/receive transaction in a transfer. The suspend interval is from the last falling clock edge of the current transaction to the first rising clock edge of the successive transaction if CLKP = 0. If CLKP = 1, the interval is from the rising clock edge to the falling clock edge. The default value is 0x0. When TX_NUM = 00b, setting this field has no effect on transfer. The desired suspend interval is obtained according to the following equation: Suspend interval for byte suspend and burst mode suspend: (SP_CYCLE[3:0] + 2) * period of SPICLK Ex: SP_CYCLE = 0x0 … 2 SPICLK clock cycle SP_CYCLE = 0x1 … 3 SPICLK clock cycle …… SP_CYCLE = 0xE … 16 SPICLK clock cycle SP_CYCLE = 0xF … 17 SPICLK clock cycle
[11]	**CLKP**	**Clock Polarity** 1 = SPICLK idle high 0 = SPICLK idle low
[10]	**LSB**	**LSB First** 1 = The LSB is sent first on the line (bit 0 of SPI_TX0/1), and the first bit received from the line will be put in the LSB position in the RX register (bit 0 of SPI_RX0/1). 0 = The MSB is transmitted/received first (which bit in SPI_TX0/1 and SPI_RX0/1 register that is depends on the TX_BIT_LEN field).
[9:8]	**TX_NUM**	**Numbers of Transmit/Receive Word** This field specifies how many transmit/receive word numbers should be executed in one transfer. 00 = Only one transmit/receive word will be executed in one transfer. 01 = Two successive transmit/receive words will be executed in one transfer. (burst mode) 10 = Reserved.

		11 = Reserved. Note: In slave mode with level-trigger configuration, if TX_NUM is set to 01, the slave select pin must be kept at active state during the successive data transfer.
[7:3]	**TX_BIT_LEN**	**Transmit Bit Length** This field specifies how many bits are transmitted in one transaction. Up to 32 bits can be transmitted. TX_BIT_LEN = 0x01 … 1 bit TX_BIT_LEN = 0x02 … 2 bits …… TX_BIT_LEN = 0x1F … 31 bits TX_BIT_LEN = 0x00 … 32 bits
[2]	**TX_NEG**	**Transmit At Negative Edge** 1 = The transmitted data output signal is changed at the falling edge of SPICLK 0 = The transmitted data output signal is changed at the rising edge of SPICLK
[1]	**RX_NEG**	**Receive At Negative Edge** 1 = The received data input signal is latched at the falling edge of SPICLK 0 = The received data input signal is latched at the rising edge of SPICLK
[0]	**GO_BUSY**	**Go and Busy Status** 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. 0 = Writing 0 to this bit to stop data transfer if SPI is transferring. During the data transfer, this bit keeps the value of 1. As the transfer is finished, this bit will be cleared automatically. Note: 1. All registers should be set before writing 1 to this GO_BUSY bit.

SPI Divider Register (SPI_DIVIDER)

Register	Offset	R/W	Description	Reset Value
SPI_DIVIDER	SPIx_BA+0x04	R/W	Clock Divider Register (Master only)	0x0000_0000

31	30	29	28	27	26	25	24
DIVIDER2[15:8]							
23	22	21	20	19	18	17	16
DIVIDER2[7:0]							
15	14	13	12	11	10	9	8
DIVIDER[15:8]							
7	6	5	4	3	2	1	0
DIVIDER[7:0]							

Bits	Descriptions	
[31:16]	DIVIDER2	**Clock Divider 2 Register** (Master only) The value in this field is the 2nd frequency divider for generating the serial clock on the output SPICLK. The desired frequency is obtained according to the following equation: $f_{sclk} = \frac{f_{pclk}}{(DIVIDER2+1)*2}$ If VARCLK_EN is cleared to 0, this setting is unmeaning.
[15:0]	DIVIDER	**Clock Divider Register** (Master only) The value in this field is the frequency divider for generating the serial clock on the output SPICLK. The desired frequency is obtained according to the following equation: $f_{sclk} = \frac{f_{pclk}}{(DIVIDER+1)*2}$ In slave mode, the period of SPI clock driven by a master shall equal or over 5 times the period of PCLK. In other words, the maximum frequency of SPI clock is the fifth of the frequency of slave's PCLK.

SPI Slave Select Register (SPI_SSR)

Register	Offset	R/W	Description	Reset Value
SPI_SSR	SPI0_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	Descriptions	
[31:6]	**Reserved**	Reserved
[5]	**LTRIG_FLAG**	**Level Trigger Flag** When the SS_LTRIG bit is set in slave mode, this bit can be read to indicate the received bit number is met the requirement or not. 1 = The transaction number and the transferred bit length met the specified requirements which defined in TX_NUM and TX_BIT_LEN. 0 = The transaction number or the transferred bit length of one transaction doesn't meet the specified requirements. Note: This bit is READ only
[4]	**SS_LTRIG**	**Slave Select Level Trigger** (Slave only) 1 = The slave select signal will be level-trigger. It depends on SS_LVL to decide the signal is active low or active high. 0 = The input slave select signal is edge-trigger. This is the default value. It depends on SS_LVL to decide the signal is active at falling-edge or rising-edge.
[3]	**AUTOSS**	**Automatic Slave Select** (Master only) 1 = If this bit is set, SPISSx0/1 signals will be generated automatically. It means that device/slave select signal, which is set in SSR[1:0], will be asserted by the SPI controller when transmit/receive is started by setting GO_BUSY, and will be de-asserted after each transmit/receive is finished. 0 = If this bit is cleared, slave select signals will be asserted and de-asserted by setting and clearing related bits in SSR[1:0].
[2]	**SS_LVL**	**Slave Select Active Level** It defines the active state of slave select signal (SPISSx0/1). 1 = The slave select signal SPISSx0/1 is active at high-level/rising-edge.

		0 = The slave select signal SPISSx0/1 is active at low-level/falling-edge.
[1:0]	**SSR**	**Slave Select Register** (Master only) If AUTOSS bit is cleared, writing 1 to any bit location of this field sets the proper SPISSx0/1 line to an active state and writing 0 sets the line back to inactive state. If 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 the corresponding 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 also defined as 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[31:24]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RX[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RX[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX[7:0]							

Bits	Descriptions	
[31:0]	**RX**	**Data Receive Register** The Data Receive Registers hold the value of received data of the last executed transfer. Valid bits depend on the transmit bit length field in the SPI_CNTRL register. For example, if TX_BIT_LEN is set to 0x08 and TX_NUM is set to 0x0, bit RX0[7:0] holds the received data. The values of the other bits are unknown. Note: The Data Receive Registers are read only registers.

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[31:24]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TX[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TX[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX[7:0]							

Bits	Descriptions	
[31:0]	**TX**	**Data Transmit Register** The Data Transmit Registers hold the data to be transmitted in the next transfer. Valid bits depend on the transmit bit length field in the CNTRL register. For example, if TX_BIT_LEN is set to 0x08 and the TX_NUM is set to 0x0, the bit TX0[7:0] will be transmitted in next transfer. If TX_BIT_LEN is set to 0x00 and TX_NUM is set to 0x1, the SPI controller will perform two 32-bit transmit/receive successive using the same setting. The transmission sequence is TX0[31:0] first and then TX1[31:0].

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 (Master only)	0x007F_FF87

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

Bits	Descriptions	
[31:0]	**VARCLK**	**Variable Clock Pattern** (Master only) The value in this field is the frequency patterns of the SPI clock. If the bit pattern of VARCLK is '0', the output frequency of SPICLK is according the value of DIVIDER. If the bit patterns of VARCLK are '1', the output frequency of SPICLK is according the value of DIVIDER2. Refer to register SPI_DIVIDER. Refer to Variable Clock paragraph for more detailed descriptions.

5.14 USB Device Controller (USB)

5.14.1 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 support 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. Users need to set the effective starting address of SRAM for each endpoint buffer through “buffer segmentation register (BUFSEGx)”.

There are six 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 support 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.

Reference: Universal Serial Bus Specification Revision 1.1

5.14.2 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 listing of this USB.

- Compliant with USB 2.0 Full-Speed specification
- Provide 1 interrupt vector with 4 different interrupt events (WAKEUP, FLDET, USB and BUS)
- Support Control/Bulk/Interrupt/Isochronous transfer type
- Support suspend function when no bus activity existing for 3 ms
- Provide 6 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 512 bytes buffer size
- Provide remote wake-up capability

5.14.3 Block Diagram

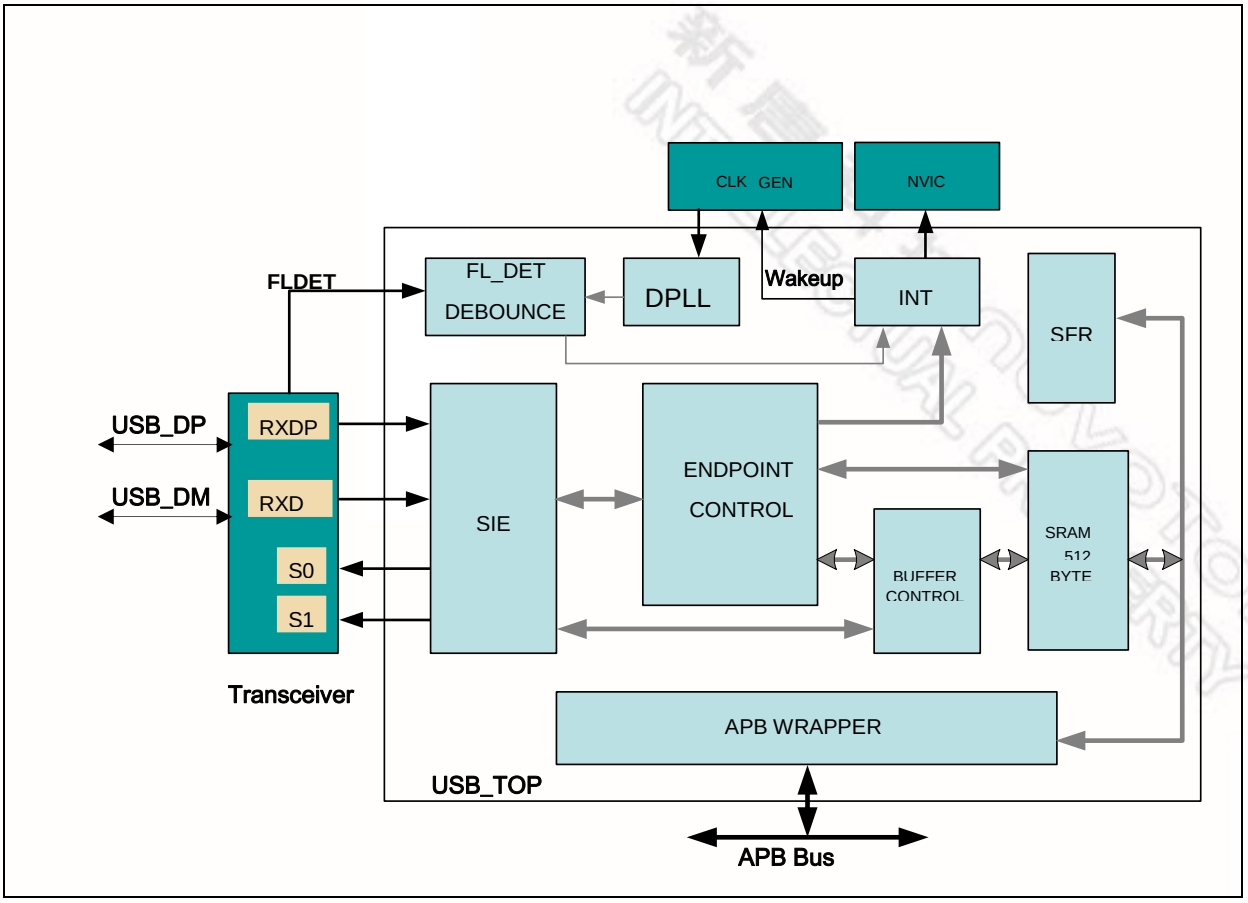


Figure 5-72 USB Block Diagram

5.14.4 Function Description

5.14.4.1 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

5.14.4.2 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.

5.14.4.3 Digital Phase Lock Loop

The bit rate of USB data is 12 MHz. The DPLL use 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.

5.14.4.4 Floating De-bounce

A USB device may be plug-in or plug-out from the USB host. In order 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. A user can acknowledge USB plug-in/plug-out by reading 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 plug-in the USB. If the user polling this flag to check USB state, he/she must add software de-bounce if necessary.

5.14.4.5 Interrupt

This USB provides 1 interrupt vector with 4 interrupt events (WAKEUP, FLDET, USB and BUS). The WAKEUP event is used to wake-up the system clock when the 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, like IN ACK, OUT ACK etc., and the BUS event notifies users of some bus events, like suspend, resume, etc. User must set related bits in the interrupt enable register (USB_INTEN) of USB Device Controller to enable USB interrupts.

Wake-up interrupt is only present when the chip entered 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 20 ms. The following figure is the control flow of wake-up interrupt.

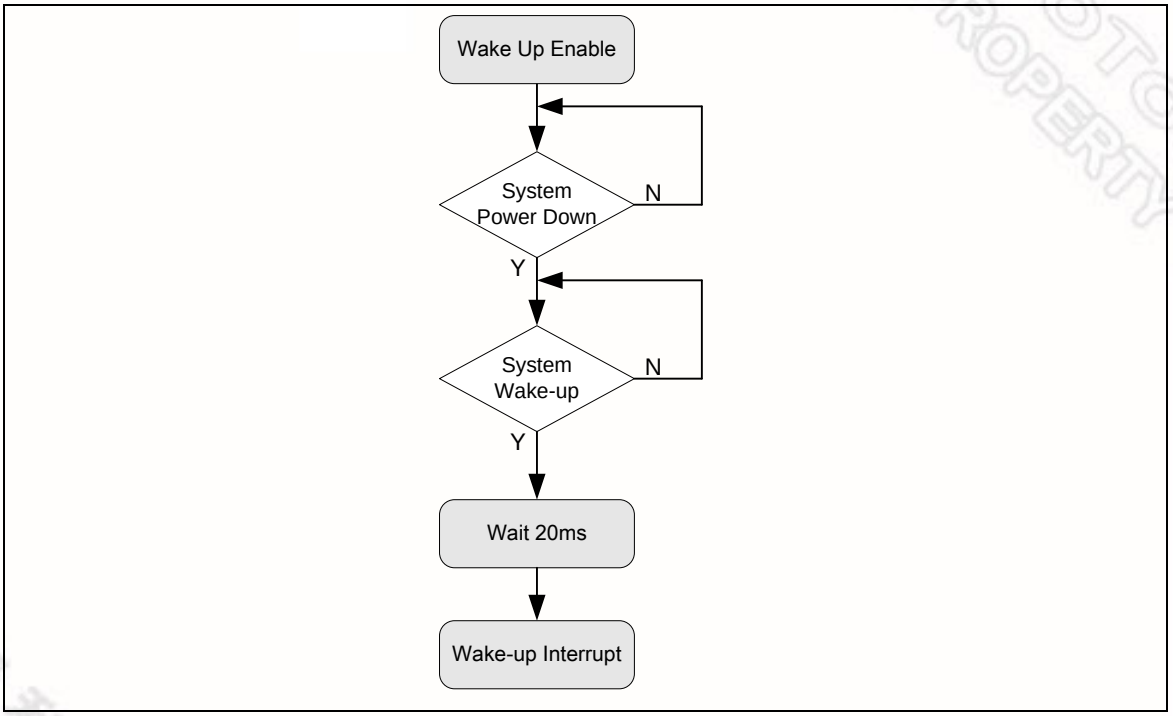


Figure 5-73 Wake-up Interrupt Operation Flow

USB interrupt is used to notify users of any USB event on the bus, and a 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.

5.14.4.6 Power Saving

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.

5.14.4.7 Buffer Control

There is 512 bytes SRAM in the controller and the 6 endpoints share this buffer. The 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.

The following figure depicts the starting address for each endpoint according the content of BUFSEG and MXPLD registers. If the BUFSEG0 is programmed as 0x08h and MXPLD0 is set as 0x40h, the SRAM size of endpoint 0 is start from USB_BASE + 0x108h and end in USB_BASE + 0x148h. (Note: the USB SRAM base is USB_BASE + 0x100h).

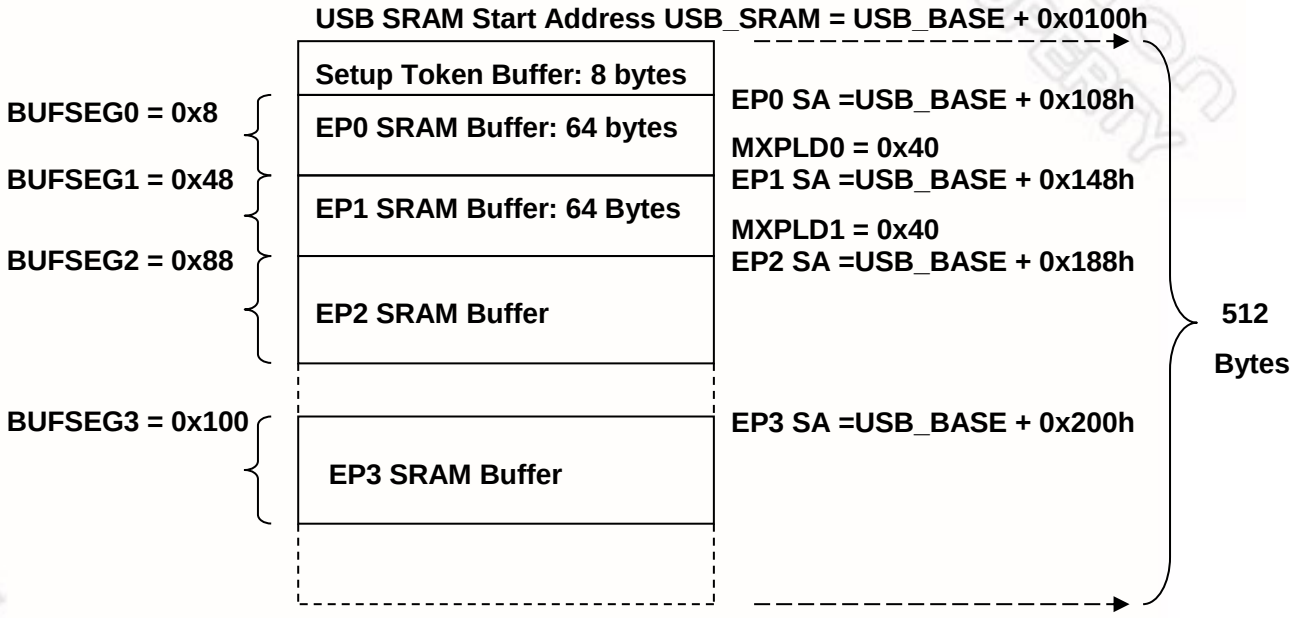


Figure 5-74 Endpoint SRAM Structure

5.14.4.8 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 enable.

When USB host has requested data from device controller, users need to prepare related data into the specified endpoint buffer in advance. After buffering the required data, users need 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.

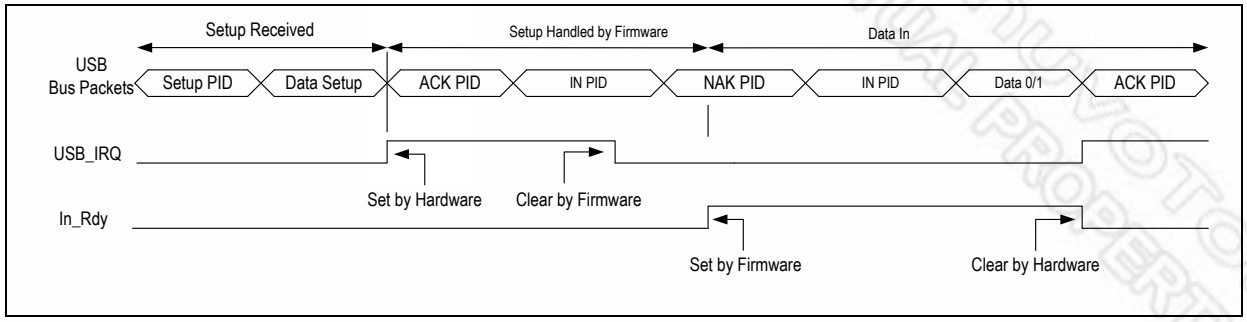


Figure 5-75 Setup Transaction Followed by Data in Transaction

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 users move out 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 next transaction.

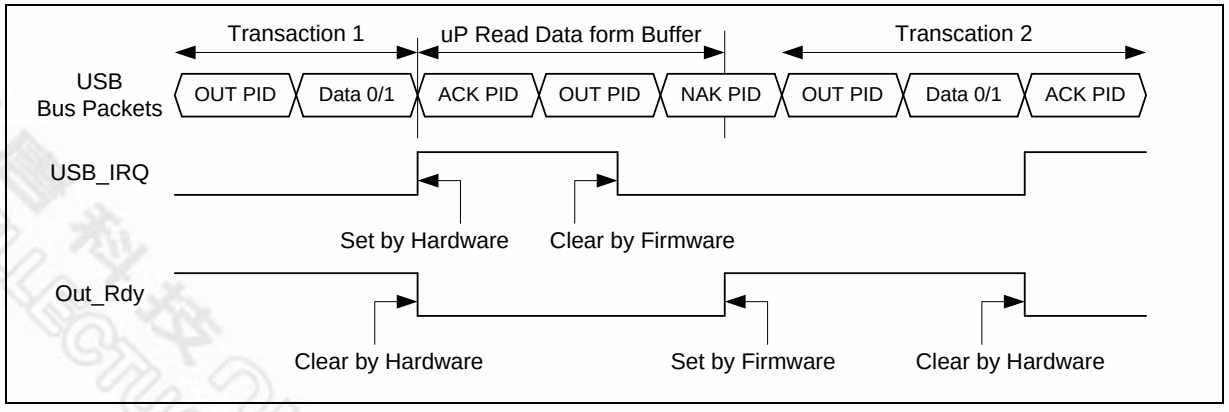


Figure 5-76 Data Out Transfer

5.14.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
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_00x0
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_BUFSEG	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+0x100h~ USB_BA+0x2FFh	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.

5.14.6 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	Descriptions	
[31:16]	**Reserved**	Reserved
[15]	**INNAK_EN**	**Active NAK Function and its Status in IN Token** 1 = The 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 enable the interrupt event when the device responds NAK after receiving IN token 0 = The NAK status doesn't be updated into the endpoint status register when it was set to 0. It also disable the interrupt event when device responds NAK after receiving IN token
[14:9]	**Reserved**	Reserved
[8]	**WAKEUP_EN**	**Wake-Up Function Enable** 1 = Enable USB wake-up function 0 = Disable USB wake-up function
[7:4]	**Reserved**	Reserved
[3]	**WAKEUP_IE**	**USB Wake-Up Interrupt Enable** 1 = Enable USB wake-up Interrupt 0 = Disable USB wake-up Interrupt
[2]	**FLDET_IE**	**Floating Detected Interrupt Enable** 1 = Enable Floating detect Interrupt 0 = Disable Floating detect Interrupt
[1]	**USB_IE**	**USB Event Interrupt Enable**

		1 = Enable USB event interrupt 0 = Disable USB event interrupt
[0]	**BUS_IE**	**Bus Event Interrupt Enable** 1 = Enable BUS event interrupt 0 = Disable BUS event interrupt

USB Interrupt Event Status Register (USB_INTSTS)

This register is USB Interrupt Event Status register; clear write '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	Descriptions	
[31]	**SETUP**	**Setup Event Status** 1 = Setup event occurred, cleared by write 1 to USB_INTSTS[31] 0 = No Setup event
[30:22]	**Reserved**	Reserved
[21]	**EPEVT5**	**Endpoint 5's USB Event Status** 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] 0 = No event occurred in endpoint 5
[20]	**EPEVT4**	**Endpoint 4's USB Event Status** 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] 0 = No event occurred in endpoint 4
[19]	**EPEVT3**	**Endpoint 3's USB Event Status** 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] 0 = No event occurred in endpoint 3
[18]	**EPEVT2**	**Endpoint 2's USB Event Status** 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] 0 = No event occurred in endpoint 2

[17]	**EPEVT1**	**Endpoint 1's USB Event Status** 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] 0 = No event occurred in endpoint 1
[16]	**EPEVT0**	**Endpoint 0's USB Event Status** 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] 0 = No event occurred in endpoint 0
[15:4]	**Reserved**	Reserved
[3]	**WAKEUP_STS**	**Wake-Up Interrupt Status** 1 = Wake-up event occurred, cleared by write 1 to USB_INTSTS[3] 0 = No wake-up event is occurred
[2]	**FLDET_STS**	**Floating Detected Interrupt Status** 1 = There is attached/detached event in the USB bus and it is cleared by write 1 to USB_INTSTS[2]. 0 = There is not attached/detached event in the USB
[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. 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 = No any USB event is occurred
[0]	**BUS_STS**	**BUS Interrupt Status** The BUS event means that there is one of the suspense or the resume function in the bus. 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]. 0 = No any BUS event is occurred

USB Device Function Address Register (USB_FADDR)

A seven-bit value uses 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	Descriptions	
[31:7]	**Reserved**	Reserved
[6:0]	**FADDR**	USB device's 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[2:1]	
23	22	21	20	19	18	17	16
EPSTS5[0]	EPSTS4[2:0]			EPSTS3[2:0]			EPSTS2[2]
15	14	13	12	11	10	9	8
EPSTS2[1:0]		EPSTS1[2:0]			EPSTS0[2:0]		
7	6	5	4	3	2	1	0
OVERRUN	Reserved						

Bits	Descriptions	
[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. 1 = It indicates that the Out Data more than the Max Payload in MXPLD register or the Setup Data more than 8 Bytes 0 = No overrun
[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	Descriptions	
[31:11]	**Reserved**	Reserved
[10]	**BYTEM**	**CPU access USB SRAM Size Mode Selection** 1 = Byte Mode: The size of the transfer from CPU to USB SRAM can be Byte only. 0 = Word Mode: The size of the transfer from CPU to USB SRAM can be Word only.
[9]	**PWRDN**	**Power Down PHY Transceiver (low active)** 1 = Turn-on related circuit of PHY transceiver 0 = power down related circuit of PHY transceiver
[8]	**DPPU_EN**	**Pull-up Resistor on USB_DP Enable** 1 = The pull-up resistor in USB_DP bus active 0 = Disable the pull-up resistor in USB_DP bus
[7]	**USB_EN**	**USB Controller Enable** 1 = Enable USB Controller 0 = Disable USB Controller
[6]	**Reserved**	Reserved
[5]	**RWAKEUP**	**Remote Wake-Up** 1 = Force USB bus to K (USB_DP low, USB_DM: high) state, used for remote wake-up 0 = Release the USB bus from K state
[4]	**PHY_EN**	**PHY Transceiver Function Enable** 1 = Enable PHY transceiver function 0 = Disable PHY transceiver function

[3]	**TIMEOUT**	**Time-Out Status** 1 = Bus no any response more than 18 bits time 0 = No time-out It is a read only bit.
[2]	**RESUME**	**Resume Status** 1 = Resume from suspend 0 = No bus resume It is a read only bit.
[1]	**SUSPEND**	**Suspend Status** 1 = Bus idle more than 3 ms, either cable is plugged off or host is sleeping 0 = Bus no suspend It is a read only bit.
[0]	**USBRST**	**USB Reset Status** 1 = Bus reset when SE0 (single-ended 0) more than 2.5us 0 = Bus no reset It is a read only bit.

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	Descriptions	
[31:1]	**Reserved**	Reserved
[0]	**FLDET**	**Device Floating Detected** 1 = When the controller is attached into the BUS, this bit will be set as 1 0 = The controller didn't attached into the USB host

Buffer Segmentation Register (USB_BUFSEG)

For Setup token only.

Register	Offset	R/W	Description	Reset Value
USB_BUFSEG	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							**BUFSEG[8]**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BUFSEG[7:3]					**Reserved**		

Bits	Descriptions	
[31:9]	**Reserved**	Reserved
[8:3]	**BUFSEG**	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 + { BUFSEG[8:3], 3'b000} Where the USB_SRAM address = USB_BASE + 0x100h. Note: It is used for Setup token only.
[2:0]	**Reserved**	Reserved

Endpoint Buffer Segmentation Register (USB_BUFSEGx) x = 0~5

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[8]x**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BUFSEG[7:3]x					**Reserved**		

Bits	Descriptions	
[31:9]	**Reserved**	Reserved
[8:3]	**BUFSEGx**	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_BASE + 0x100h. Refer to section 5.4.4.7 for the endpoint SRAM structure and its description.
[2:0]	**Reserved**	Reserved

Maximal Payload Register (USB_MXPLDx) x = 0~5

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[8]**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MXPLD[7:0]							

Bits	Descriptions	
[31:9]	**Reserved**	Reserved
[8:0]	**MXPLD**	**Maximal Payload** It is used to 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 the data length be transmitted to host For OUT token, the value of MXPLD is indicated the actual data length receiving from host. Note that once MXPLD is written, the data packets will be transmitted/received immediately after IN/OUT token arrived.

Configuration Register (USB_CFGx) x = 0~5

Register	Offset	R/W	Description	Reset Value
USB_CFG0	USB_BA+0x028	R/W	Endpoint 0's Configuration Register	0x0000_0000
USB_CFG1	USB_BA+0x038	R/W	Endpoint 1's Configuration Register	0x0000_0000
USB_CFG2	USB_BA+0x048	R/W	Endpoint 2's Configuration Register	0x0000_0000
USB_CFG3	USB_BA+0x058	R/W	Endpoint 3's Configuration Register	0x0000_0000
USB_CFG4	USB_BA+0x068	R/W	Endpoint 4's Configuration Register	0x0000_0000
USB_CFG5	USB_BA+0x078	R/W	Endpoint 5's 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	Descriptions	
[31:10]	**Reserved**	Reserved
[9]	**CSTALL**	**Clear STALL Response** 1 = Clear the device to response STALL handshake in setup stage 0 = Disable the device to clear the STALL handshake in setup stage
[8]	**Reserved**	Reserved
[7]	**DSQ_SYNC**	**Data Sequence Synchronization** 1 = DATA1 PID 0 = DATA0 PID It is used to specify the DATA0 or DATA1 PID in the following IN token transaction. H/W 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. 1 = Isochronous endpoint 0 = No 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) x = 0~5

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	Descriptions	
[31:2]	**Reserved**	Reserved
[1]	**SSTALL**	**Set STALL** 1 = Set the device to respond STALL automatically 0 = Disable the device to response STALL
[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' is used to clear the IN token had ready to transmit the data to USB. For OUT token, write '1' is used to clear the OUT token had ready to receive the data from USB. This bit is write 1 only and it is always 0 when it was read back.

USB Drive SE0 Register (USB_DRVSE0)

Register	Offset	R/W	Description	Reset Value
USB_DRVSE0	USB_BA+0x090	R/W	Force USB PHY to Drive SE0	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	Descriptions	
[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. 1 = Force USB PHY transceiver to drive SE0 0 = None

6 ELECTRICAL CHARACTERISTICS

6.1 Absolute Maximum Ratings

SYMBOL	PARAMETER	MIN.	MAX.	UNIT
DC Power Supply	V_{DD}–V_{SS}	-0.3	+7.0	V
Input Voltage	VIN	V_{SS}-0.3	V_{DD}+0.3	V
Oscillator Frequency	$1/t_{CLCL}$	4	24	MHz
Operating Temperature	TA	-40	+85	°C
Storage Temperature	TST	-55	+150	°C
Maximum Current into V_{DD}		-	120	mA
Maximum Current out of V_{SS}			120	mA
Maximum Current sunk by a I/O pin			35	mA
Maximum Current sourced by a I/O pin			35	mA
Maximum Current sunk by total I/O pins			100	mA
Maximum Current sourced by total I/O pins			100	mA

Note: Exposure to conditions beyond those listed under absolute maximum ratings may adversely affects the lift and reliability of the device.

6.2 DC Electrical Characteristics

6.2.1 NuMicro™ NUC122 DC Electrical Characteristics

(V_{DD}-V_{SS}=3.3 V, TA = 25 °C, FOSC = 60 MHz unless otherwise specified.)

PARAMETER	SYM.	SPECIFICATION				TEST CONDITIONS
		MIN.	TYP.	MAX.	UNIT	
Operation voltage	V_{DD}	2.5		5.5	V	V_{DD} =2.5 V ~ 5.5 V up to 60 MHz
LDO Output Voltage	V_{LDO}	1.6	1.8	2.1	V	V_{DD} ≥ 2.5 V
Analog Operating Voltage	AV_{DD}	2.1		V_{DD}	V	
Operating Current Normal Run Mode @ 60 MHz	I_{DD1}		26		mA	V_{DD} = 5.5 V @ 60 MHz, enable all IP and PLL, XTAL=12 MHz
	I_{DD2}		21		mA	V_{DD} = 5.5 V @ 60 MHz, disable all IP and enable PLL, XTAL=12 MHz
	I_{DD3}		24		mA	V_{DD} = 3.3 V @ 60 MHz, enable all IP and PLL, XTAL=12 MHz
	I_{DD4}		19		mA	V_{DD} = 3.3 V @ 60 MHz, disable all IP and enable PLL, XTAL=12 MHz
Operating Current Normal Run Mode @ 12 MHz	I_{DD5}		6.5		mA	V_{DD} = 5.5 V @ 12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I_{DD6}		5		mA	V_{DD} = 5.5 V @ 12 MHz, disable all IP and PLL, XTAL=12 MHz
	I_{DD7}		4.5		mA	V_{DD} = 3.3 V @ 12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I_{DD8}		3.5		mA	V_{DD} = 3.3 V @ 12 MHz, disable all IP and PLL, XTAL=12 MHz
Operating Current Normal Run Mode @ 4 MHz	I_{DD9}		3.5		mA	V_{DD} = 5.5 V @ 4 MHz, enable all IP and disable PLL, XTAL=4 MHz
	I_{DD10}		3		mA	V_{DD} = 5.5 V @ 4 MHz, disable all IP and PLL, XTAL=4 MHz

PARAMETER	SYM.	SPECIFICATION				TEST CONDITIONS
		MIN.	TYP.	MAX.	UNIT	
	I_{DD11}		3		mA	V_{DD} = 3.3 V @ 4 MHz, enable all IP and disable PLL, XTAL=4 MHz
	I_{DD12}		2		mA	V_{DD} = 3.3 V @ 4 MHz, disable all IP and PLL, XTAL=4 MHz
Operating Current Idle Mode @ 60 MHz	I_{IDLE1}		17		mA	V_{DD} = 5.5 V @ 60 MHz, enable all IP and PLL, XTAL=12 MHz
	I_{IDLE2}		12		mA	V_{DD} = 5.5 V @ 60 MHz, disable all IP and enable PLL, XTAL=12 MHz
	I_{IDLE3}		15		mA	V_{DD} = 3.3 V @ 60 MHz, enable all IP and PLL, XTAL=12 MHz
	I_{IDLE4}		11		mA	V_{DD} = 3.3 V @ 60 MHz, disable all IP and enable PLL, XTAL=12 MHz
Operating Current Idle Mode @ 12 MHz	I_{IDLE5}		4.5		mA	V_{DD} = 5.5 V @ 12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I_{IDLE6}		3.5		mA	V_{DD} = 5.5 V @ 12 MHz, disable all IP and PLL, XTAL=12 MHz
	I_{IDLE7}		3		mA	V_{DD} = 3.3 V @ 12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I_{IDLE8}		2		mA	V_{DD} = 3.3 V @ 12 MHz, disable all IP and PLL, XTAL=12 MHz
Operating Current Idle Mode @ 4 MHz	I_{IDLE9}		3		mA	V_{DD} = 5.5 V @ 4 MHz, enable all IP and disable PLL, XTAL=4 MHz
	I_{IDLE10}		2.5		mA	V_{DD} = 5.5 V @ 4 MHz, disable all IP and PLL, XTAL=4 MHz
	I_{IDLE11}		2		mA	V_{DD} = 3.3 V @ 4 MHz, enable all IP and disable PLL, XTAL=4 MHz
	I_{IDLE12}		1		mA	V_{DD} = 3.3 V @ 4 MHz, disable all IP and PLL, XTAL=4 MHz
Standby Current	I_{PWD1}		13		μA	V_{DD} = 5.5 V, RTC OFF, No load

PARAMETER	SYM.	SPECIFICATION				TEST CONDITIONS
		MIN.	TYP.	MAX.	UNIT	
Power Down Mode						@ Disable BOV function
	I_{PWD2}		12		μA	V_{DD} = 3.3 V, RTC OFF, No load @ Disable BOV function
	I_{PWD3}		15		μA	V_{DD} = 5.5 V, RTC run , No load @ Disable BOV function
	I_{PWD4}		13		μA	V_{DD} = 3.3 V, RTC run , No load @ Disable BOV function
Input Current PA, PB, PC, PD (Quasi-bidirectional mode)	I_{IN1}	-60	-	+15	μA	V_{DD} = 5.5 V, V_{IN} = 0 V or V_{IN}=V_{DD}
Input Current at /RESET[1]	I_{IN2}	-55	-45	-30	μA	V_{DD} = 3.3 V, V_{IN} = 0.45 V
Input Leakage Current PA, PB, PC, PD	I_{LK}	-2	-	+2	μA	V_{DD} = 5.5 V, 0<V_{IN}<V_{DD}
Logic 1 to 0 Transition Current PA~PD (Quasi-bidirectional mode)	I_{TL} [3]	-650	-	-200	μA	V_{DD} = 5.5 V, V_{IN}<2.0 V
Input Low Voltage PA, PB, PC, PD (TTL input)	V_{IL1}	-0.3	-	0.8	V	V_{DD} = 4.5 V
		-0.3	-	0.6		V_{DD} = 2.5 V
Input High Voltage PA, PB, PC, PD(TTL input)	V_{IH1}	2.0	-	V_{DD} +0.2	V	V_{DD} = 5.5 V
		1.5	-	V_{DD} +0.2		V_{DD} = 3.0 V
Input Low Voltage PA, PB, PC, PD (Schmitt input)	V_{IL2}	-0.5		0.4 V_{DD}	V	
Input High Voltage PA, PB, PC, PD(Schmitt input)	V_{IH2}	0.6 V_{DD}		V_{DD}+0.5	V	
Hysteresis voltage of PA~PD (Schmitt input)	V_{HY}		0.2 V_{DD}		V	
Negative going threshold (Schmitt input), /RESET	V_{ILS}	-0.5	-	0.3 V_{DD}	V	
Positive going threshold (Schmitt input), /RESET	V_{IHS}	0.7 V_{DD}	-	V_{DD}+0.5	V	
Source Current PA, PB, PC, PD (Quasi-bidirectional Mode)	I_{SR11}	-300	-370	-450	μA	V_{DD} = 4.5 V, V_S = 2.4 V
	I_{SR12}	-50	-70	-90	μA	V_{DD} = 2.7 V, V_S = 2.2 V
	I_{SR12}	-40	-60	-80	μA	V_{DD} = 2.5 V, V_S = 2.0 V
Source Current PA, PB, PC, PD (Push-pull Mode)	I_{SR21}	-22	-28	-32	mA	V_{DD} = 4.5 V, V_S = 2.4 V
	I_{SR22}	-4	-6	-8	mA	V_{DD} = 2.7 V, V_S = 2.2 V
	I_{SR22}	-3	-5	-7	mA	V_{DD} = 2.5 V, V_S = 2.0 V
Sink Current PA, PB, PC,	I_{SK1}	10	17	20	mA	V_{DD} = 4.5 V, V_S = 0.45 V

PARAMETER	SYM.	SPECIFICATION				TEST CONDITIONS
		MIN.	TYP.	MAX.	UNIT	
PD(Quasi-bidirectional and Push-pull Mode)	I_{SK1}	7	10	13	mA	V_{DD} = 2.7 V, V_S = 0.45 V
	I_{SK1}	6	9	12	mA	V_{DD} = 2.5 V, V_S = 0.45 V
Brownout voltage with BOV_VL [1:0] =00b	$V_{BO2.2}$	2.1	2.2	2.3	V	
Brownout voltage with BOV_VL [1:0] =01b	$V_{BO2.7}$	2.6	2.7	2.8	V	
Brownout voltage with BOV_VL [1:0] =10b	$V_{BO3.8}$	3.6	3.75	3.9	V	
Brownout voltage with BOV_VL [1:0] =11b	$V_{BO4.5}$	4.2	4.4	4.6	V	
Hysteresis range of BOD voltage	V_{BH}	30	-	150	mV	V_{DD} = 2.5 V ~ 5.5 V

Note:

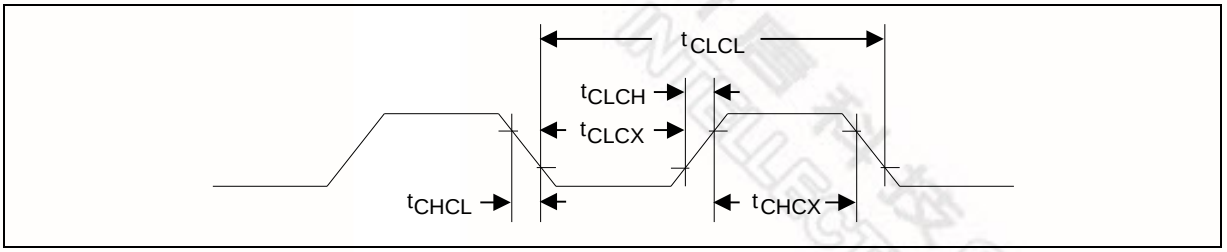
1. /RESET pin is a Schmitt trigger input.

2. Crystal Input is a CMOS input.

3. Pins of PA, PB, PC and PD can source a transition current when they are being externally driven from 1 to 0. In the condition of V_{DD}=5.5 V, the transition current reaches its maximum value when V_{IN} approximates to 2 V.

6.3 AC Electrical Characteristics

6.3.1 External 4~24 MHz High Speed Crystal AC Electrical Characteristics



Note: Duty cycle is 50 %.

SYMBOL	PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
t_{CHCX}	Clock High Time		20	-	-	nS
t_{CLCX}	Clock Low Time		20	-	-	nS
t_{CLCH}	Clock Rise Time		-	-	10	nS
t_{CHCL}	Clock Fall Time		-	-	10	nS

6.3.2 External 4~24 MHz High Speed Crystal

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Input clock frequency	External crystal	4	12	24	MHz
Temperature	-	-40	-	85	℃

6.3.2.1 Typical Crystal Application Circuits

CRYSTAL	C1	C2	R
4 MHz ~ 24 MHz	without	without	without

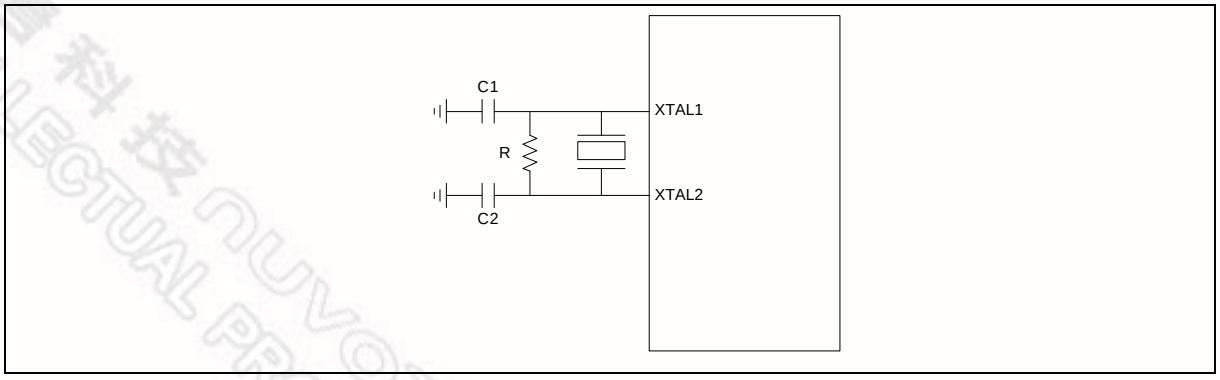


Figure 6-1 Typical Crystal Application Circuit

6.3.3 External 32.768 KHz Low Speed Crystal

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Input clock frequency	External crystal	-	32.768	-	KHz
Temperature	-	-40	-	85	℃

6.3.4 Internal 22.1184 MHz High Speed Oscillator

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Center Frequency	-	-	22.1184	-	MHz
Calibrated Internal Oscillator Frequency	+25 ℃; V_{DD} = 3.3 V	-1	-	+1	%
	-40 ℃ ~ +85 ℃; V_{DD} = 2.5 V ~ 5.5 V	-5	-	+5	%

6.3.5 Internal 10 KHz Low Speed Oscillator

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Center Frequency	-	-	10	-	KHz
Calibrated Internal Oscillator Frequency	+25 ℃; V_{DD} = 5 V	-30	-	+30	%
	-40 ℃ ~ +85 ℃; V_{DD} = 2.5 V ~ 5.5 V	-50	-	+50	%

6.4 Analog Characteristics

6.4.1 Specification of LDO and Power Management

PARAMETER	MIN.	TYP.	MAX.	UNIT	NOTE
Input Voltage	2.5	5	5.5	V	V_{DD} input voltage
Output Voltage	1.6	1.8	2.1	V	$V_{DD} \geq 2.5$ V
Temperature	-40	25	85	℃	
Quiescent Current (PD=0)	-	100	-	μA	
Quiescent Current (PD=1)	-	5	-	μA	
Iload (PD=0)	-	-	100	mA	
Iload (PD=1)	-	-	100	μA	
Cbp	-	4.7	-	μF	Resr=1 ohm

Note:

1. It is recommended that a 10 μF or higher capacitor and a 100 nF bypass capacitor are connected between VDD and the closest VSS pin of the device.

2. For ensuring power stability, a 4.7 μF or higher capacitor must be connected between LDO pin and the closest VSS pin of the device.

6.4.2 Specification of Low Voltage Reset

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Quiescent current	V_{DD}=5.5 V	-	-	5	μA
Temperature	-	-40	25	85	℃
Threshold voltage	Temperature=25℃	1.7	2.0	2.3	V
	Temperature=-40℃	-		-	V
	Temperature=85℃	-		-	V
Hysteresis	-	0	0	0	V

6.4.3 Specification of Brownout Detector

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Quiescent current	AV_{DD}=5.5 V	-	-	140	μA
Temperature	-	-40	25	85	℃
Brownout voltage	BOV_VL[1:0]=11	4.2	4.4	4.6	V
	BOV_VL [1:0]=10	3.6	3.75	3.9	V
	BOV_VL [1:0]=01	2.6	2.7	2.8	V
	BOV_VL [1:0]=00	2.1	2.2	2.3	V
Hysteresis	-	30	-	150	mV

6.4.4 Specification of Power-On Reset

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Temperature	-	-40	25	85	℃
Reset voltage	V+	-	2	-	V
Quiescent current	Vin>reset voltage	-	1	-	nA

6.4.5 Specification of USB PHY

6.4.5.1 USB DC Electrical Characteristics

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
V_{IH}	Input high (driven)		2.0			V
V_{IL}	Input low				0.8	V
V_{DI}	Differential input sensitivity	|PADP-PADM|	0.2			V
V_{CM}	Differential common-mode range	Includes V_{DI} range	0.8		2.5	V
V_{SE}	Single-ended receiver threshold		0.8		2.0	V
	Receiver hysteresis			200		mV
V_{OL}	Output low (driven)		0		0.3	V
V_{OH}	Output high (driven)		2.8		3.6	V
V_{CRS}	Output signal cross voltage		1.3		2.0	V
R_{PU}	Pull-up resistor		1.425		1.575	kΩ
R_{PD}	Pull-down resistor		14.25		15.75	kΩ
V_{TRM}	Termination Voltage for upstream port pull up (RPU)		3.0		3.6	V
Z_{DRV}	Driver output resistance	Steady state drive*		10		Ω
C_{IN}	Transceiver capacitance	Pin to GND			20	pF

*Driver output resistance doesn't include series resistor resistance.

6.4.5.2 USB Full-Speed Driver Electrical Characteristics

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
T_{FR}	Rise Time	C_L=50p	4		20	ns
T_{FF}	Fall Time	C_L=50p	4		20	ns
T_{FRFF}	Rise and fall time matching	$T_{FRFF}=T_{FR}/T_{FF}$	90		111.11	%

6.4.5.3 USB Power Dissipation

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
I_{VDDREG} (Full Speed)	V_{DDD} and V_{DDREG} Supply Current (Steady State)	Standby		50		μA
		Input mode				μA
		Output mode				μA

6.5 SPI Dynamic Characteristics

6.5.1 Dynamic Characteristics of Data Input and Output Pin

SYMBOL	PARAMETER	MIN.	TYP.	MAX.	UNIT
SPI MASTER MODE (VDD = 4.5 V ~ 5.5 V, 30 PF LOADING CAPACITOR)					
t_{DS}	Data setup time	16	10	-	ns
t_{DH}	Data hold time	0	-	-	ns
t_V	Data output valid time	-	5	8	ns
SPI Master Mode (VDD = 3.0 V ~ 3.6 V, 30 pF loading Capacitor)					
t_{DS}	Data setup time	20	13	-	ns
t_{DH}	Data hold time	0	-	-	ns
t_V	Data output valid time	-	7	14	ns
SPI Slave Mode (VDD = 4.5 V ~ 5.5 V, 30 pF loading Capacitor)					
t_{DS}	Data setup time	0	-	-	ns
t_{DH}	Data hold time	2*PCLK+4	-	-	ns
t_V	Data output valid time	-	2*PCLK+11	2*PCLK+20	ns
SPI Slave Mode (VDD = 3.0 V ~ 3.6 V, 30 pF loading Capacitor)					
t_{DS}	Data setup time	0	-	-	ns
t_{DH}	Data hold time	2*PCLK+8	-	-	ns
t_V	Data output valid time	-	2*PCLK+20	2*PCLK+32	ns

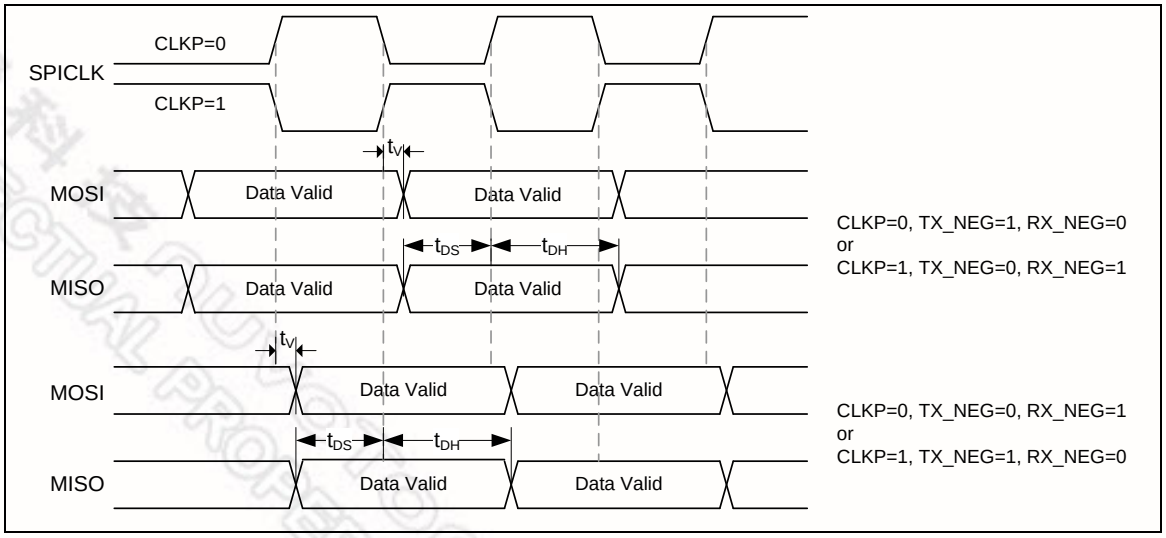


Figure 6-2 SPI Master Mode Timing

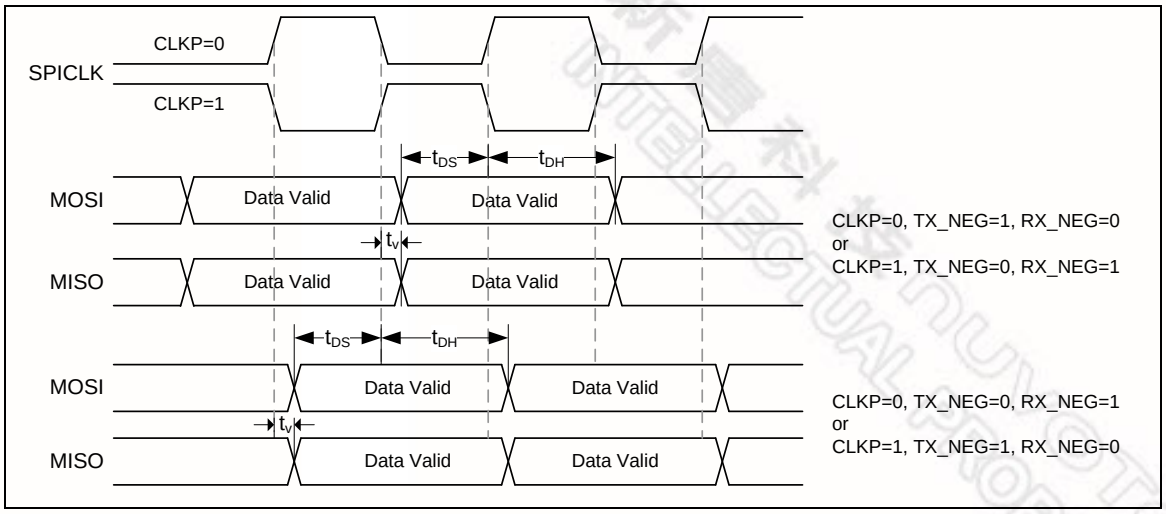
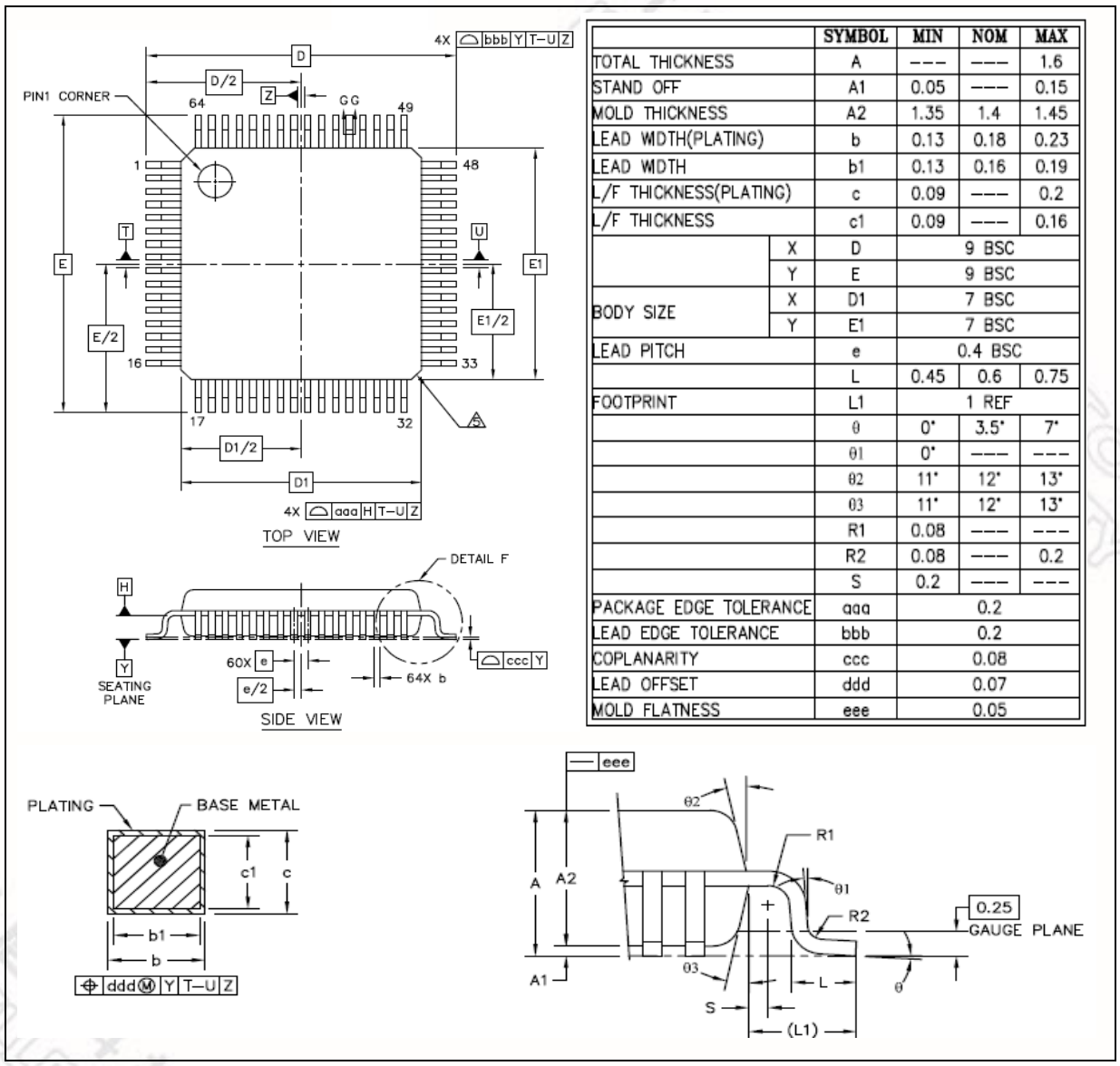


Figure 6-3 SPI Slave Mode Timing

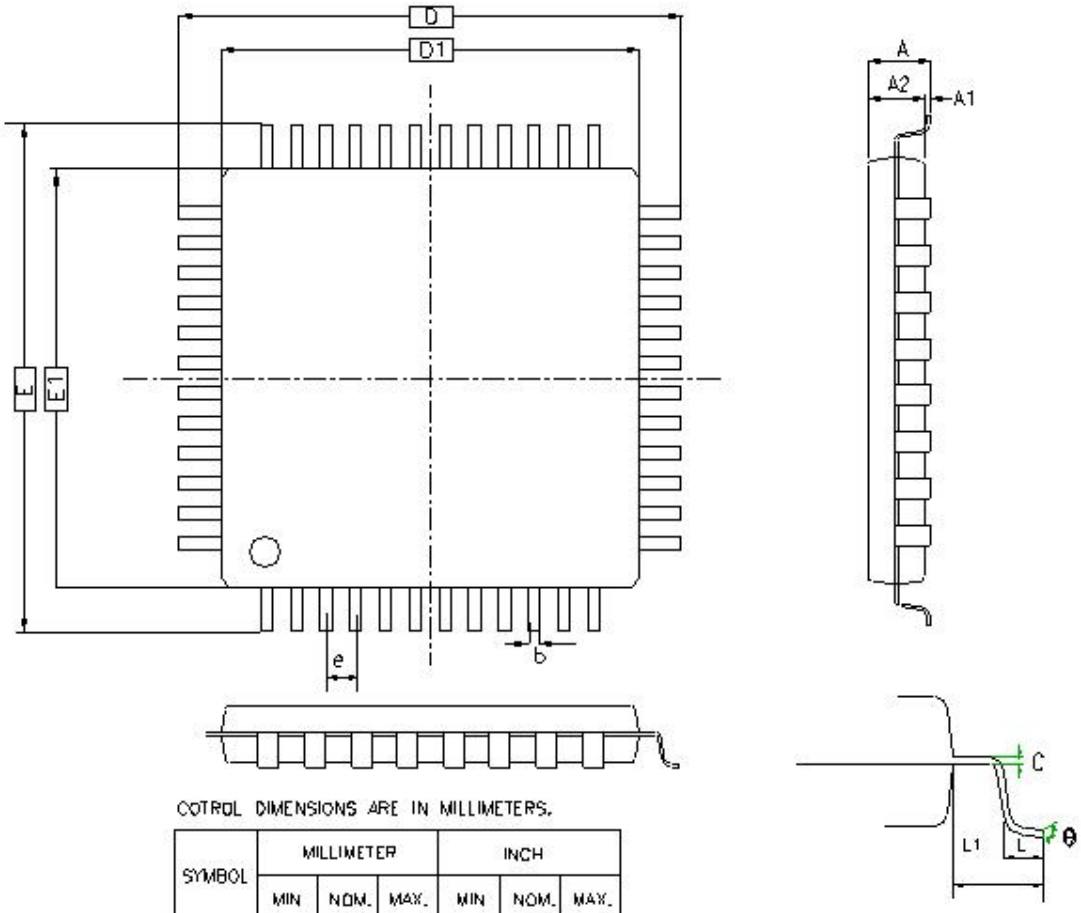
7 PACKAGE DIMENSIONS

7.1 64L LQFP (7x7x1.4mm footprint 2.0 mm)



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	---	---	1.6
STAND OFF		A1	0.05	---	0.15
MOLD THICKNESS		A2	1.35	1.4	1.45
LEAD WIDTH(PLATING)		b	0.13	0.18	0.23
LEAD WIDTH		b1	0.13	0.16	0.19
L/F THICKNESS(PLATING)		c	0.09	---	0.2
L/F THICKNESS		c1	0.09	---	0.16
BODY SIZE	X	D	9 BSC		
	Y	E	9 BSC		
	X	D1	7 BSC		
	Y	E1	7 BSC		
LEAD PITCH		e	0.4 BSC		
		L	0.45	0.6	0.75
FOOTPRINT		L1	1 REF		
		θ	0°	3.5°	7°
		θ1	0°	---	---
		θ2	11°	12°	13°
		θ3	11°	12°	13°
		R1	0.08	---	---
		R2	0.08	---	0.2
		S	0.2	---	---
PACKAGE EDGE TOLERANCE		aaa	0.2		
LEAD EDGE TOLERANCE		bbb	0.2		
COPLANARITY		ccc	0.08		
LEAD OFFSET		ddd	0.07		
MOLD FLATNESS		eee	0.05		

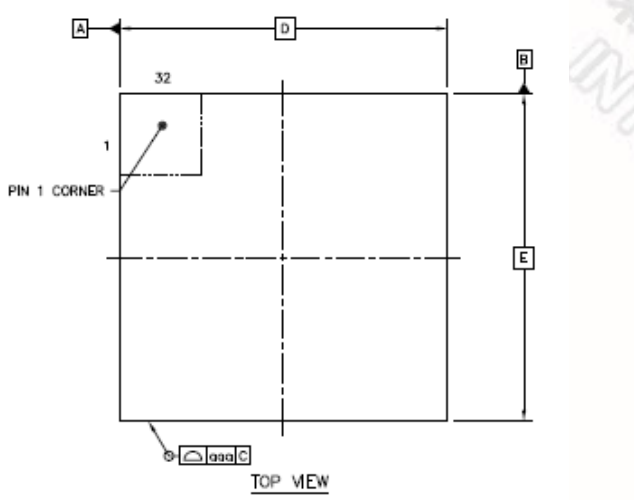
7.2 48L LQFP (7x7x1.4mm footprint 2.0 mm)

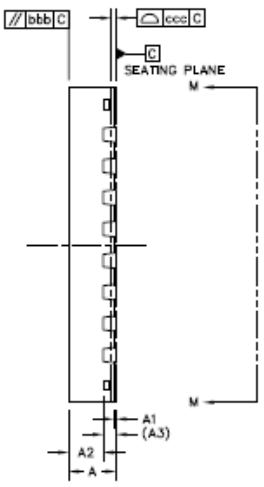


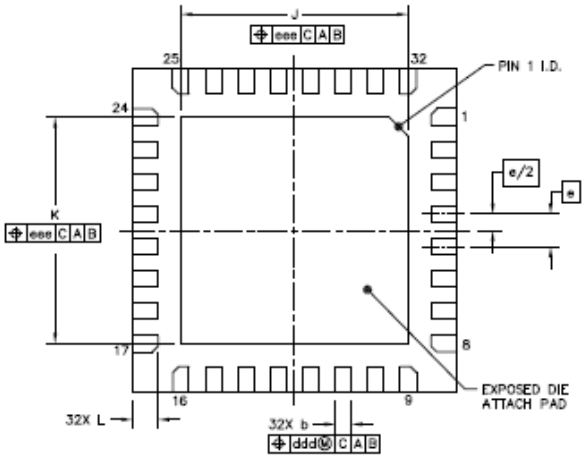
CONTROL DIMENSIONS ARE IN MILLIMETERS.

SYMBOL	MILLIMETER			INCH		
	MIN	NOM.	MAX.	MIN	NOM.	MAX.
A	—	—	1.60	—	—	0.063
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
D1	6.90	7.00	7.10	0.272	0.276	0.280
E1	6.90	7.00	7.10	0.272	0.276	0.280
e	0.35	0.50	0.65	0.014	0.020	0.280
D	8.9	9.00	9.10	0.350	0.354	0.358
E	8.9	9.00	9.10	0.350	0.354	0.358
L	0.45	0.60	0.75	0.018	0.024	0.030
L1	—	1.00	—	—	0.039	—
C	0.09	—	0.20	0.0035	—	0.0079
θ	0°	—	7°	0°	—	7°
b	0.17	0.22	0.27	0.007	0.0087	0.011

7.3 33L QFN (5x5x0.8mm)







BOTTOM VIEW
VIEW M–M

		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.7	0.75	0.8
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.55	0.57
L/F THICKNESS		A3		0.203 REF	
LEAD WIDTH		b	0.2	0.25	0.3
BODY SIZE	X	D		5 BSC	
	Y	E		5 BSC	
LEAD PITCH		e		0.5 BSC	
EP SIZE	X	J	3.4	3.5	3.6
	Y	K	3.4	3.5	3.6
LEAD LENGTH		L	0.35	0.4	0.45
PACKAGE EDGE TOLERANCE		aaa		0.1	
MOLD FLATNESS		bbb		0.1	
COPLANARITY		ccc		0.08	
LEAD OFFSET		ddd		0.1	
EXPOSED PAD OFFSET		eee		0.1	

8 REVISION HISTORY

VERSION	DATE	PAGE/ CHAP.	DESCRIPTION
V1.00	Jan. 18, 2011		Preliminary version initial issued
V1.01	Feb. 11, 2011	Chap. 5	Correct the TMRx_S descriptions in the CLKSEL1 register.
V1.02	March 14, 2011	Chap. 3 Chap. 5 Chap .7 Chap. 8	1. Added the LQFP 64-pin part number for 7x7x1.4mm package. (NUC122SD2AN, NUC122SC1AN) 2. Corrected the LQFP 64-pin Pin Diagram. 3. Corrected the clock source block diagram of Timer in the bit field of CLKSEL1. 4. Removed the RCADJ control register. 5. Corrected the SPI related descriptions of functions and control registers. 6. Updated DC and AC Electrical Characteristics. 7. Updated LQFP 48-pin package dimensions.
V1.03	March 31, 2011	Chap. 2 Chap. 3 Chap. 4 Chap. 5 Chap. 8	1. Removed the LQFP 64-pin part number for 10x10x1.4mm package. 2. Replaced “12 MHz” with “4~24 MHz” in some contents and block diagrams. 3. Added ICE debug ACK control bits in TCSR[31] and WTCR[31] control registers.
V1.04	Apr. 29, 2011	Chap. 1 Chap. 2 Chap. 3 Chap. 5 Chap. 6 Chap. 7	1. Corrected the GPIOxn_DOUT description. (x=A~D, n=0~15) 2. Updated the RTC Block Diagram. 3. Updated the table of specification of LDO and Power Management. 4. Removed the LIN function from UART controller. 5. Corrected “PS2DAT” to “PS2CLK” in FPS2CLK register. 6. Corrected the “five” options to “four” options in the description for the clock source block diagram of Timer Controller. 7. Corrected the reset value in I2CSTATUS register. 8. Corrected the “PWM_CRLx/PWM_CFLx(x=0~3)” to “CRLRx/CFLRx(x=0~3)” in the Overview of PWM Generator and Capture Timer chapter. 9. Corrected the descriptions of SPE, EPE, PBE and NSB bits in UA_LCR register. 10. Corrected the PIIR, RIIR and TTR registers as R/W. 11. Corrected the Programming Examples of SPI. 12. Corrected the “1xx” to “111” in System Clock and SysTick Clock Control Block Diagram. 13. Added the note in the GPIOx_DMASK registers. (x=A~D) 14. Added the Clock Generator Global View Diagram.

			15. Added the Function Description of Timer Controller. 16. Corrected the description of ICSR and SCR registers. 17. Corrected the “RX0/1” and “TX0/1” to “RXD0/1” and “TXD0/1” in Pin Configuration and Pin Description. 18. Corrected the description of bit field TOIC in UA_TOR register. 19. Corrected the description of FATCON register. 20. Corrected the five status flow diagrams of I^2C operation mode.
V1.05	May 30, 2011	Chap. 3 Chap. 5	1. Corrected the Pin Description of pins 17 and 18 for LQFP 48-pin. 2. Corrected the description of PIIR register.
V1.06	June 8, 2011	Chap. 2 Chap. 7	1. Corrected the trimmed condition for the internal 22.1184 MHz high speed oscillator in the “Clock Control” item of Feature list. 2. Corrected the specification of the “Internal 22.1184 MHz High Speed Oscillator”.
V1.07	June 21, 2011	Chap. 2 Chap. 5	1. Added the condition and corrected the speed of SPI in Master/Slave mode in the “SPI” item of Feature list. 2. Corrected the descriptions of CH0MOD, CH1MOD, CH2MOD CH3MOD, TIF and CURRENT registers. 3. Corrected the descriptions of some SPI functions, GO_BUSY and SSR registers.
V1.08	Dec 5, 2011	Chap. 8	1. Corrected QFN33 package dimension.
V1.09	May 16, 2014	Chap. 3 Chap. 5	1. Added the PF.2 and PF.3 function on PS2DAT and PS2CLK in Pin Diagram and Pin Description. . 2. Added GPF_MFP register.
V1.10	Dec. 22, 2014	Chap. 5	1. Removed SPI FIFO mode. 2. Rearranged the chapter 5 session sequence.
V1.11	Jan. 09, 2015	Chap. 2 Chap. 5	1. Corrected UART FIFO to 14-byte and the description of UA_FSR register. 2. Corrected the typo EP_NUM in USB_CFGx registers, x=0~5. 3. Removed the GPIO PF.2 and PF.3 in Pin Diagram and Pin Description and the GPF_MFP register.

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.