nuvoTon

ARM Cortex™-M0

32-BIT MICROCONTROLLER

NuMicro M051™ Series
Technical Reference Manual

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 GENERAL DESCRIPTION

The NuMicro M051™ series is a 32-bit microcontroller with embedded ARM® Cortex™-M0 core for industrial control and applications which need rich communication interfaces. The Cortex™-M0 is the newest ARM embedded processor with 32-bit performance and at a cost equivalent to traditional 8-bit microcontroller. The NuMicro M051™ series includes M052, M054, M058 and M0516 families.

The NuMicro M051™ series can run up to 50 MHz. Thus it can afford to support a variety of industrial control and applications which need high CPU performance. The NuMicro M051™ series has 8K/16K/32K/64K-byte embedded flash, 4K-byte data flash, 4K-byte flash for the ISP, and 4K-byte embedded SRAM.

Many system level peripheral functions, such as I/O Port, EBI (External Bus Interface), Timer, UART, SPI, I2C, PWM, ADC, Watchdog Timer and Brownout Detector, have been incorporated into the NuMicro M051™ series in order to reduce component count, board space and system cost. These useful functions make the NuMicro M051™ series powerful for a wide range of applications.

Additionally, the NuMicro M051™ series is equipped with ISP (In-System Programming) and ICP (In-Circuit Programming) functions, which allow the user to update the program memory without removing the chip from the actual end product.

2 FEATURES

- Core
  - ARM® Cortex™-M0 core runs up to 50 MHz.
  - One 24-bit system timer.
  - Supports low power sleep-mode.
  - A single-cycle 32-bit hardware multiplier.
  - NVIC for the 32 interrupt inputs, each with 4-levels of priority.
  - Supports Serial Wire Debug (SWD) interface and 2 watchpoints/4 breakpoints.
- Built-in LDO for Wide Operating Voltage Range: 2.5V to 5.5V
- Memory
  - 8KB/16KB/32KB/64KB Flash memory for program memory (APROM)
  - 4KB Flash memory for data memory (DataFlash)
  - 4KB Flash memory for loader (LDROM)
  - 4KB SRAM for internal scratch-pad RAM (SRAM)
- Clock Control
  - Programmable system clock source
  - 4~24 MHz external crystal input
  - 22.1184 MHz internal oscillator (trimmed to 1% accuracy)
  - 10 kHz low-power oscillator for Watchdog Timer and wake-up in sleep mode
  - PLL allows CPU operation up to the maximum 50MHz
- I/O Port
  - Up to 40 general-purpose I/O (GPIO) pins for LQFP-48 package

- 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
- Supports high driver and high sink IO mode

- Timer
  - Provides four channel 32-bit timers, one 8-bit pre-scale counter with 24-bit up-timer for each timer.
  - Independent clock source for each timer.
  - 24-bit timer value is readable through TDR (Timer Data Register)
  - Provides one-shot, periodic and toggle operation modes.
- Watchdog Timer
  - Multiple clock sources
  - Supports wake up from power down or sleep mode
  - Interrupt or reset selectable on watchdog time-out
- PWM
  - Built-in up to four 16-bit PWM generators; providing eight PWM outputs or four complementary paired PWM outputs
  - Individual clock source, clock divider, 8-bit pre-scalar and dead-zone generator for each PWM generator
  - PWM interrupt synchronized to PWM period
  - 16-bit digital Capture timers (shared with PWM timers) with rising/falling capture inputs

- Supports capture interrupt

- UART
  - Up to two sets of UART device
  - Programmable baud-rate generator
  - Buffered receiver and transmitter, each with 15 bytes FIFO
  - Optional flow control function (CTS and RTS)
  - Supports IrDA(SIR) function
  - Supports RS485 function
- SPI
  - Up to two sets of SPI device.
  - Supports master/slave mode
  - Master mode clock rate up to 20 MHz, and slave mode clock rate up to 10 MHz
  - Full duplex synchronous serial data transfer
  - Variable length of transfer data from 1 to 32 bits
  - MSB or LSB first data transfer
  - Rx latching data can be either at rising edge or at falling edge of serial clock
  - Tx sending data can be either at rising edge or at falling edge of serial clock
  - Supports Byte suspend mode in 32-bit transmission
- I2C
  - Supports 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.
- Supports multiple address recognition (four slave address with mask option)

- ADC
  - 12-bit SAR ADC with 600k SPS
  - Up to 8-ch single-ended input or 4-ch differential input
  - Supports single mode/burst mode/single-cycle scan mode/continuous scan mode
  - Each channel with an individual result register
  - Supports conversion value monitoring (or comparison) for threshold voltage detection
  - Conversion can be started either by software trigger or external pin trigger
- EBI (External Bus Interface) for external memory-mapped device access
  - Accessible space: 64KB in 8-bit mode or 128KB in 16-bit mode
  - Supports 8-bit/16-bit data width
- In-System Programming (ISP) and In-Circuit Programming (ICP)
- Brownout Detector
  - With 4 levels: 4.5V/3.8V/2.7V/2.2V
  - Supports brownout interrupt and reset option
- LVR (Low Voltage Reset)
  - Threshold voltage levels: 2.0V
- Operating Temperature: -40℃~85℃
- Packages:
  - Green package (RoHS)

- 48-pin LQFP, 33-pin QFN

3 BLOCK DIAGRAM

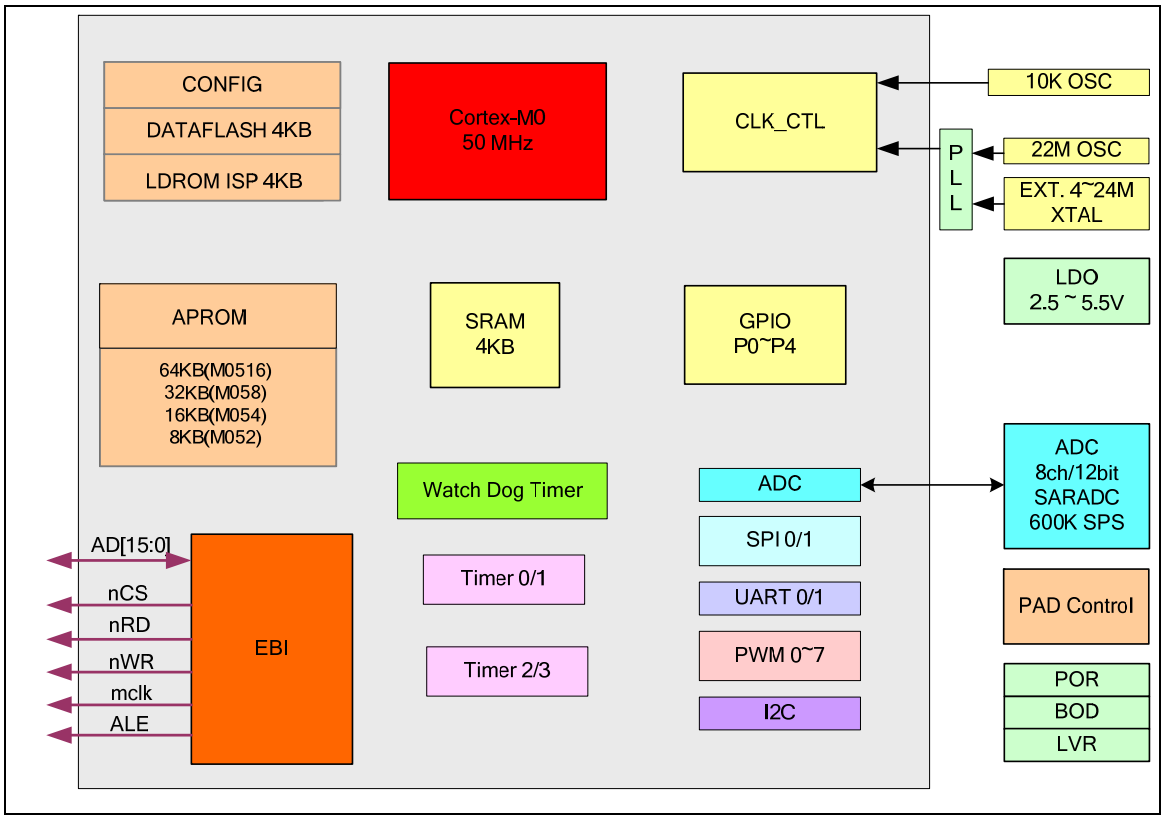


Figure 5.1-1 NuMicro™ M051 Series Block Diagram

4 SELECTION TABLE

NuMicro M051™ Series Selection Guide

Part No.	APROM	RAM	Data Flash	LDROM	I/O	Timer	Connectivity			PWM	ADC	EBI	ISP ICP	Package
							UART	SPI	I2C					
M052LAN	8KB	4KB	4KB	4KB	40	4x32-bit	2	2	1	8	8x12-bit	v	v	LQFP48
M052ZAN	8KB	4KB	4KB	4KB	24	4x32-bit	2	1	1	5	5x12-bit		v	QFN 33
M054LAN	16KB	4KB	4KB	4KB	40	4x32-bit	2	2	1	8	8x12-bit	v	v	LQFP48
M054ZAN	16KB	4KB	4KB	4KB	24	4x32-bit	2	1	1	5	5x12-bit		v	QFN 33
M058LAN	32KB	4KB	4KB	4KB	40	4x32-bit	2	2	1	8	8x12-bit	v	v	LQFP48
M058ZAN	32KB	4KB	4KB	4KB	24	4x32-bit	2	1	1	5	5x12-bit		v	QFN 33
M0516LAN	64KB	4KB	4KB	4KB	40	4x32-bit	2	2	1	8	8x12-bit	v	v	LQFP48
M0516ZAN	64KB	4KB	4KB	4KB	24	4x32-bit	2	1	1	5	5x12-bit		v	QFN 33

Table 5.1-1 NuMicro™ M051 Series Product Selection Guide

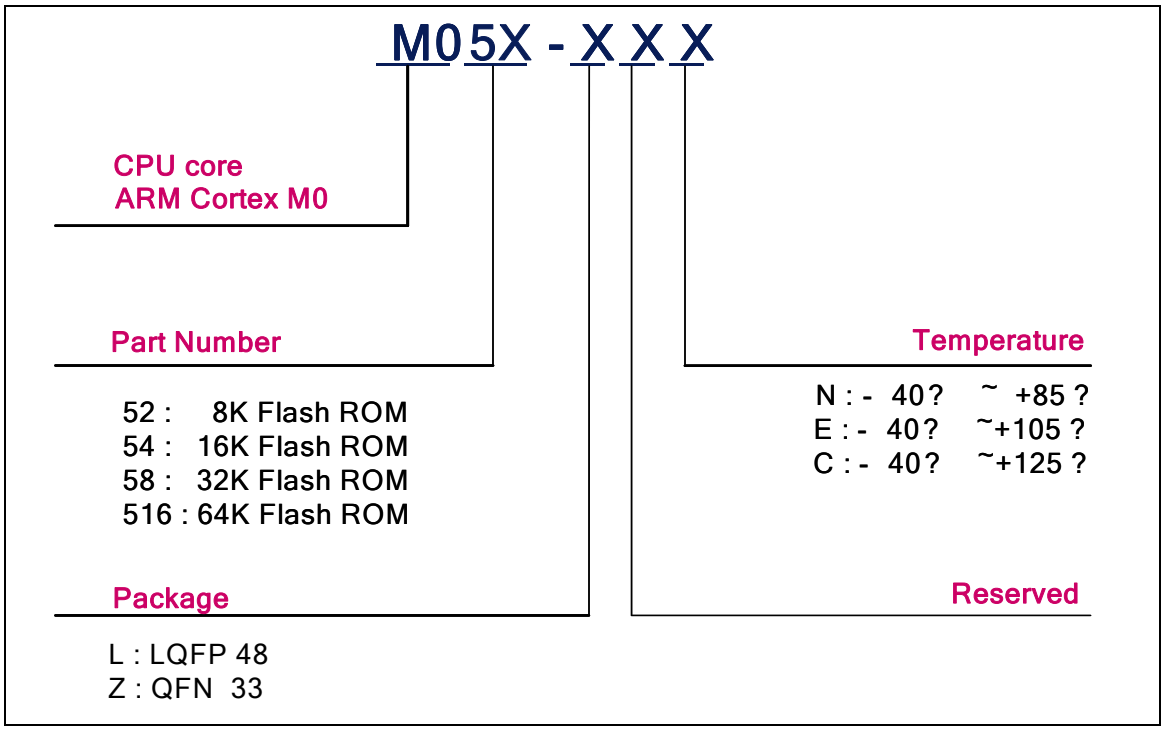


Figure 5.1-1 NuMicro™ Naming Rule

5 PIN CONFIGURATION

5.1 QFN 33 pin

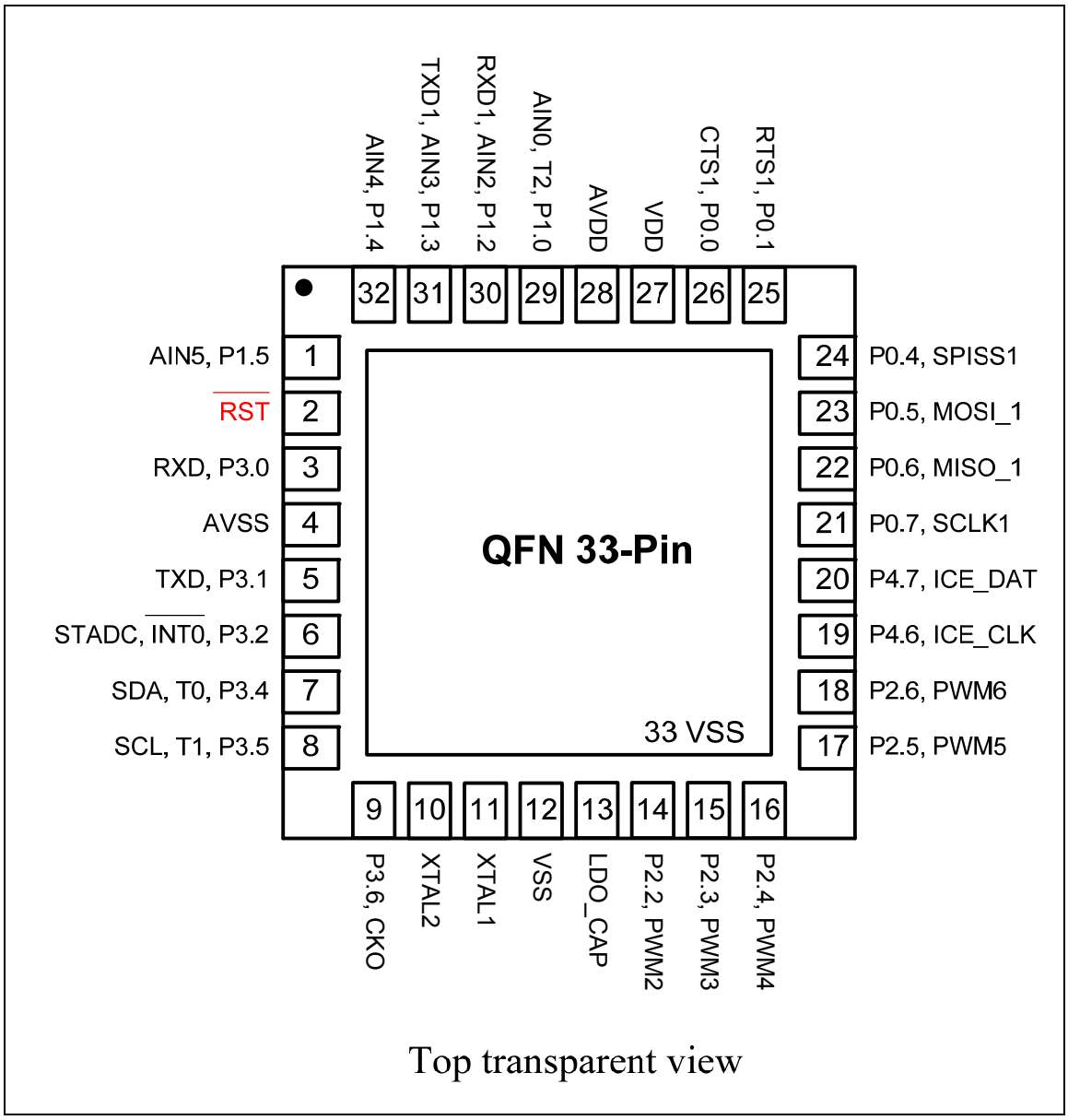


Figure 5.1-1 NuMicro™ M051 Series QFN33 Pin Diagram

5.2 LQFP 48 pin

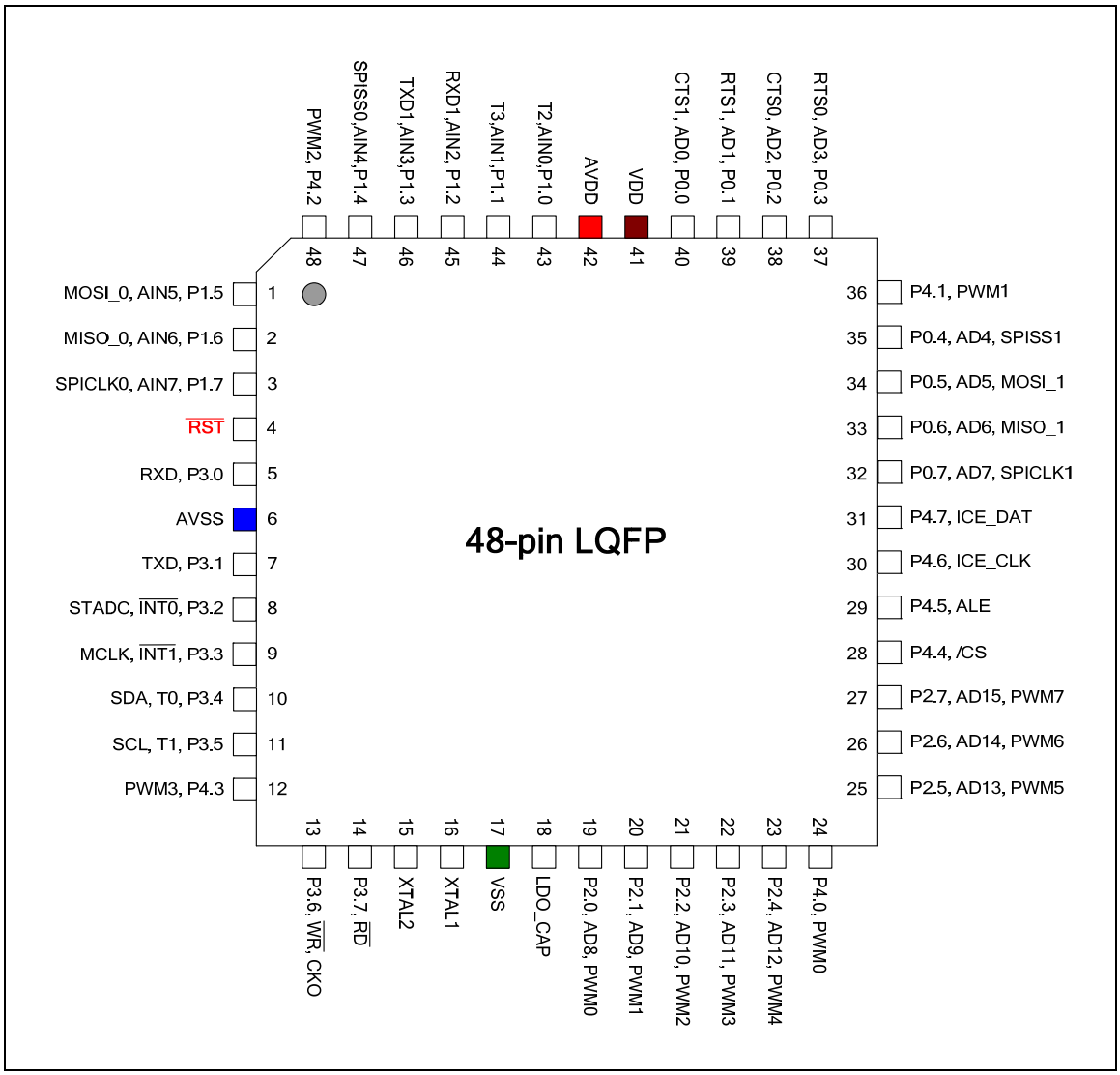


Figure 5.2-1 NuMicro™ M051 Series LQFP-48 Pin Diagram

5.3 Pin Description

Pin number		Symbol	Alternate Function		Type[1]	Description
QFN33	LQFP48		1	2		
11	16	XTAL1			I (ST)	**CRYSTAL1:** This is the input pin to the internal inverting amplifier. The system clock is from external crystal or resonator when FOSC[1:0] (CONFIG3[1:0]) are both logic 1 by default.
10	15	XTAL2			O	**CRYSTAL2:** This is the output pin from the internal inverting amplifier. It emits the inverted signal of XTAL1.
27	41	VDD			P	**POWER SUPPLY:** Supply voltage **Digital** V_{DD} for operation.
12 33	17	VSS			P	**GROUND: Digital** Ground potential.
28	42	AVDD			P	**POWER SUPPLY:** Supply voltage **Analog** AV_{DD} for operation.
4	6	AVSS			P	**GROUND: Analog** Ground potential.
13	18	LDO_CAP			P	**LDO:** LDO output pin **Note: It needs to be connected with a 10uF capacitor.**
2	4	/RST			I (ST)	**RESET:** /RST pin is a Schmitt trigger input pin for hardware device reset. A "**Low"** on this pin for 768 clock counter of Internal RC 22.1184 MHz while the system clock is running will reset the device. /RST pin has an internal pull-up resistor allowing power-on reset by simply connecting an external capacitor to GND.
26	40	P0.0	CTS1	AD0	D, I/O	**PORT0:** Port 0 is an 8-bit four mode output pin and two mode input. Its multifunction pins are for CTS1, RTS1, CTS0, RTS0, SPISS1, MOSI_1, MISO_1, and SPICLK1. P0 has an alternative function as AD[7:0] while external memory interface (EBI) is enabled. These pins which are SPISS1, MOSI_1, MISO_1, and SPICLK1 for the SPI function used. CTS0/1: Clear to Send input pin for UART0/1
25	39	P0.1	RTS1	AD1	D, I/O	
NC	38	P0.2	CTS0	AD2	D, I/O	
NC	37	P0.3	RTS0	AD3	D, I/O	
24	35	P0.4	SPISS1	AD4	D, I/O	

Pin number		Symbol	Alternate Function		Type[1]	Description
QFN33	LQFP48		1	2		
23	34	P0.5	MOSI_1	AD5	D, I/O	RTS0/1: Request to Send output pin for UART0/1
22	33	P0.6	MISO_1	AD6	D, I/O	
21	32	P0.7	SPICLK1	AD7	D, I/O	
29	43	P1.0	T2	AIN0	I/O	**PORT1:** Port 1 is an 8-bit four mode output pin and two mode input. Its multifunction pins are for T2, T3, RXD1, TXD1, SPISS0, MOSI_0, MISO_0, and SPICLK0. T2: Timer2 external input T3: Timer3 external input These pins which are SPISS0, MOSI_0, MISO_0, and SPICLK0 for the SPI function used. These pins which are AIN0~AIN7for the 12 bits ADC function used. The RXD1/TXD1 pins are for UART1 function used.
NC	44	P1.1	T3	AIN1	I/O	
30	45	P1.2	RXD1	AIN2	I/O	
31	46	P1.3	TXD1	AIN3	I/O	
32	47	P1.4	SPISS0	AIN4	I/O	
1	1	P1.5	MOSI_0	AIN5	I/O	
NC	2	P1.6	MISO_0	AIN6	I/O	
NC	3	P1.7	SPICLK0	AIN7	I/O	
NC	19	P2.0	PWM0	AD8	D, I/O	**PORT2:** Port 2 is an 8-bit four mode output pin and two mode input. It has an alternative function P2 has an alternative function as AD[15:8] while external memory interface (EBI) is enabled. These pins which are PWM0~PWM7 for the PWM function.
NC	20	P2.1	PWM1	AD9	D, I/O	
14	21	P2.2	PWM2	AD10	D, I/O	
15	22	P2.3	PWM3	AD11	D, I/O	
16	23	P2.4	PWM4	AD12	D, I/O	
17	25	P2.5	PWM5	AD13	D, I/O	
18	26	P2.6	PWM6	AD14	D, I/O	
NC	27	P2.7	PWM7	AD15	D, I/O	
3	5	P3.0	RXD		I/O	**PORT3:** Port 3 is an 8-bit four mode output pin and two

Pin number		Symbol	Alternate Function		Type[1]	Description
QFN33	LQFP48		1	2		
5	7	P3.1	TXD		I/O	mode input. Its multifunction pins are for RXD, TXD, $\overline{INT0}$, $\overline{INT1}$, T0, T1, $\overline{WR}$, and $\overline{RD}$. T0: Timer0 external input T1: Timer1 external input The RXD/TXD pins are for UART0 function used. The SDA/SCL pins are for I2C function used. MCLK: EBI clock output pin. CKO: HCLK clock output The STADC pin is for ADC external trigger input.
6	8	P3.2	$\overline{INT0}$	STADC	I/O	
NC	9	P3.3	$\overline{INT1}$	MCLK	I/O	
7	10	P3.4	T0	SDA	I/O	
8	11	P3.5	T1	SCL	I/O	
9	13	P3.6	$\overline{WR}$	CKO	I/O	
NC	14	P3.7	$\overline{RD}$		I/O	
NC	24	P4.0	PWM0		I/O	**PORT4:** Port 4 is an 8-bit four mode output pin and two mode input. Its multifunction pins are for /CS, ALE, ICE_CLK and ICE_DAT. /CS for EBI (External Bus Interface) used. ALE (Address Latch Enable) is used to enable the address latch that separates the address from the data on Port 0 and Port 2. The ICE_CLK/ICE_DAT pins are for JTAG-ICE function used. PWM0-3 can be used from P4.0-P4.3 when EBI is active.
NC	36	P4.1	PWM1		I/O	
NC	48	P4.2	PWM2		I/O	
NC	12	P4.3	PWM3		I/O	
NC	28	P4.4	/CS		I/O	
NC	29	P4.5	ALE		I/O	
19	30	P4.6	ICE_CLK		I/O	
20	31	P4.7	ICE_DAT		I/O	

Table 5.3-1 NuMicro™ M051 Series Pin Description

[1] I/O type description. I: input, O: output, I/O: quasi bi-direction, D: open-drain, P: power pins, ST: Schmitt trigger.

6 FUNCTIONAL DESCRIPTION

6.1 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. The profile supports two modes -Thread and Handler modes. Handler mode is entered as a result of an exception. An exception return can only be issued in Handler mode. Thread mode is entered on Reset, and can be entered as a result of an exception return.

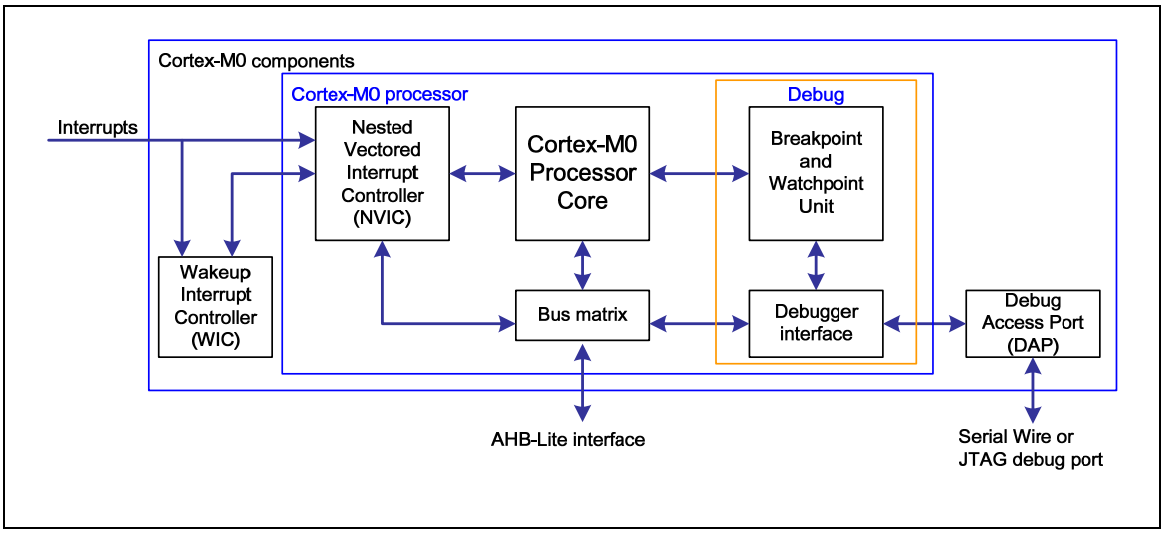


Figure 6.1-1 Functional Block Diagram

The implemented device provides:

A low gate count processor the features:

- The ARMv6-M Thumb® instruction set.
- Thumb-2 technology.
- ARMv6-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, interrupt handling.
- Load/store-multiples and multicycle-multiplies that can be abandoned and restarted to facilitate rapid interrupt handling.
- C Application Binary Interface compliant exception model.
  This is the ARMv6-M, C Application Binary Interface(C-ABI) compliant exception model that enables the use of pure C functions as interrupt handlers.
- Low power sleep-mode entry using Wait For Interrupt (WFI), Wait For Event(WFE) instructions, or the return from interrupt sleep-on-exit feature.

NVIC 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), supports ultra-low power sleep mode.

Debug support:

- Four hardware breakpoints.
- Two watchpoints.
- Program Counter Sampling Register (PCSR) for non-intrusive code profiling.
- Single step and vector catch capabilities.

Bus interfaces:

- Single 32-bit AMBA-3 AHB-Lite system interface that provides simple integration to all system peripherals and memory.
- Single 32-bit slave port that supports the DAP (Debug Access Port).

6.2 System Manager

6.2.1 Overview

The following functions are included in system manager section

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

6.2.2 System Reset

The system reset includes one of the list below event occurs. For these reset event flags can be read by RSTRC register.

- The Power-On Reset (POR)
- The low level on the /RESET pin
- Watchdog Time Out Reset (WDT)
- Low Voltage Reset (LVR)
- Brown-Out-Detected Reset (BOD)
- CPU Reset
- System Reset

6.2.3 System Power Architecture

In this device, the power architecture is divided into three segments.

- Analog power from AVDD and AVSS provides the power for analog module operation.

- Digital power from VDD and VSS supplies the power to the internal regulator which provides a fixed 2.5V power for digital operation and I/O pins.

The outputs of internal voltage regulator, which is LDO, require an external capacitor which should be located close to the corresponding pin. The Figure 6.2-1 shows the power architecture of this device.

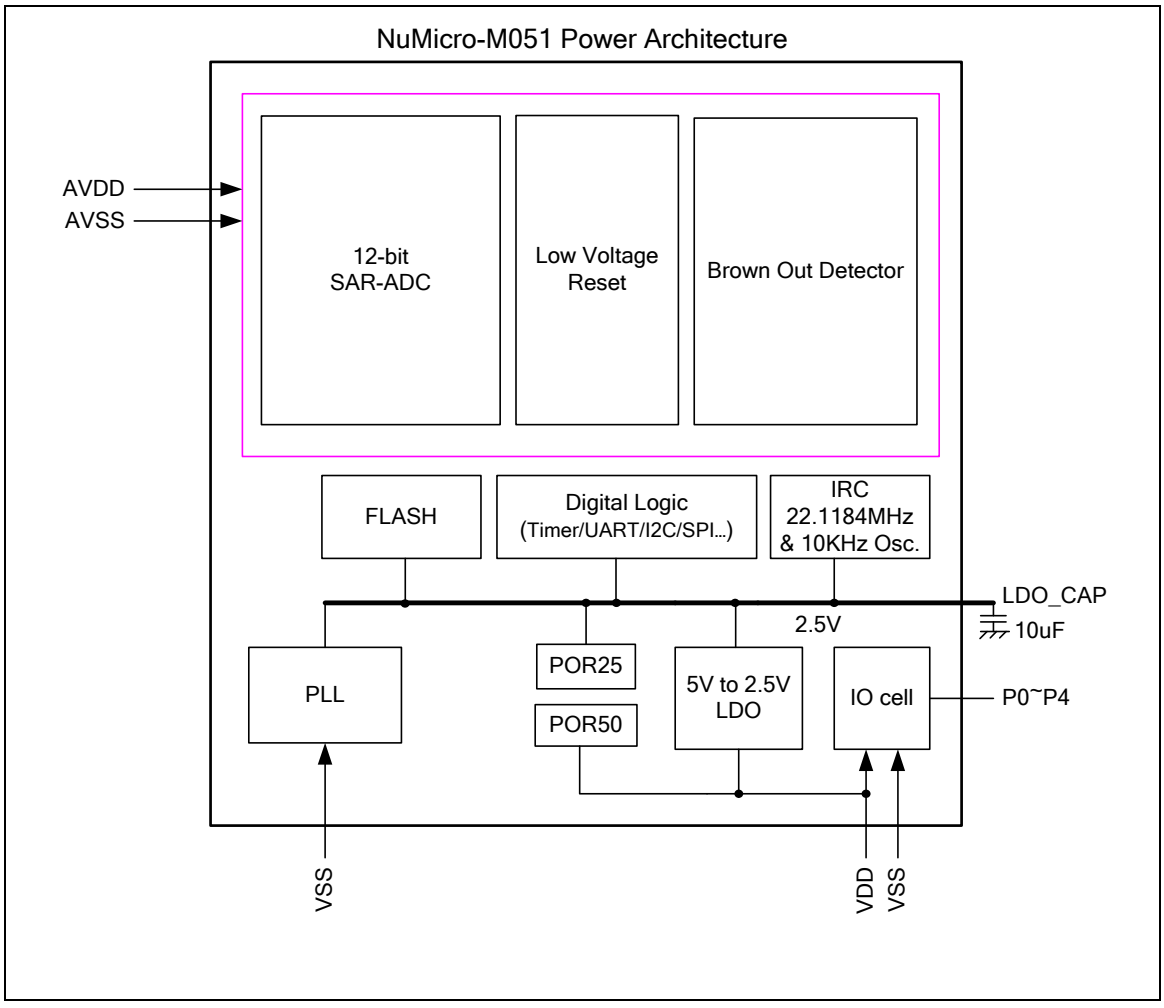


Figure 6.2-1 NuMicro M051™ Series Power Architecture Diagram

6.2.4 Whole System Memory Map

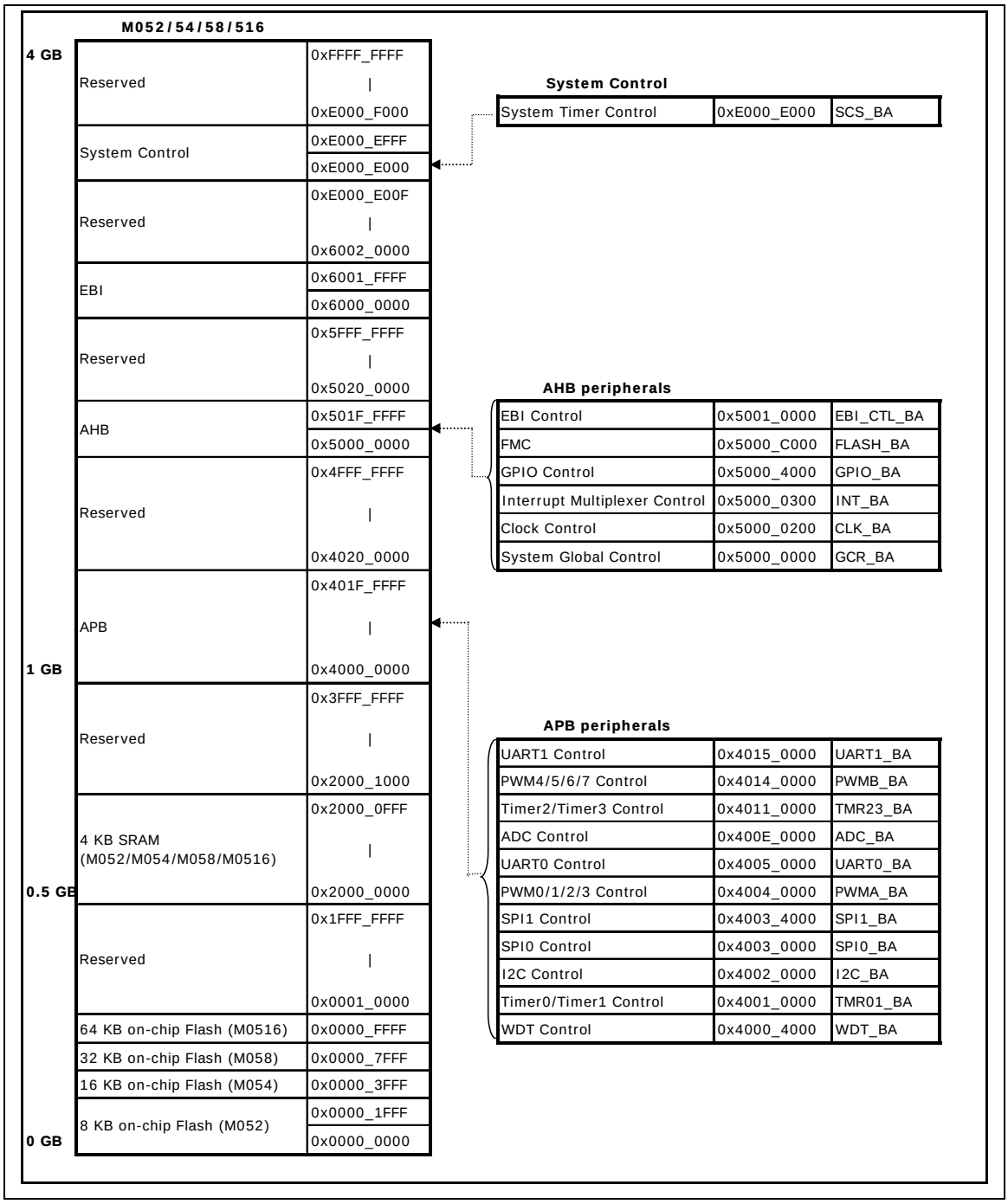
NuMicro M051™ series provides a 4G-byte address space. The memory locations assigned to each on-chip modules are shown in Table 6.2-1. The detailed register memory addressing and programming will be described in the following sections for individual on-chip peripherals. NuMicro M051™ series only supports little-endian data format.

Address Space	Token	Modules
Flash & SRAM Memory Space		
0x0000_0000 – 0x0000_FFFF	FLASH_BA	FLASH Memory Space (64KB)
0x2000_0000 – 0x2000_0FFF	SRAM_BA	SRAM Memory Space (4KB)
AHB Modules 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 (P0~P4) Control Registers
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash Memory Control Registers
0x5001_0000 – 0x5001_3FFF	EBI_CTL_BA	EBI Control Registers (128KB)
EBI Space (0x6000_0000 ~ 0x6001_FFFF)		
0x6000_0000 – 0x6001_FFFF	EBI_BA	EBI Space
APB Modules Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4000_4000 – 0x4000_7FFF	WDT_BA	Watch-Dog Timer Control Registers
0x4001_0000 – 0x4001_3FFF	TMR01_BA	Timer0/Timer1 Control Registers
0x4002_0000 – 0x4002_3FFF	I2C_BA	I2C Interface Control Registers
0x4003_0000 – 0x4003_3FFF	SPI0_BA	SPI0 with master/slave function Control Registers
0x4003_4000 – 0x4003_7FFF	SPI1_BA	SPI1 with master/slave function Control Registers

0x4004_0000 – 0x4004_3FFF	PWMA_BA	PWM0/1/2/3 Control Registers
0x4005_0000 – 0x4005_3FFF	UART0_BA	UART0 Control Registers
0x400E_0000 – 0x400E_FFFF	ADC_BA	Analog-Digital-Converter (ADC) Control Registers
0x4011_0000 – 0x4011_3FFF	TMR23_BA	Timer2/Timer3 Control Registers
0x4014_0000 – 0x4014_3FFF	PWMB_BA	PWM4/5/6/7 Control Registers
0x4015_0000 – 0x4015_3FFF	UART1_BA	UART1 Control Registers
System Control Space (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SCS_BA	System Timer Control Registers
0xE000_E100 – 0xE000_ECFF	SCS_BA	External Interrupt Controller Control Registers
0xE000_ED00 – 0xE000_ED8F	SCS_BA	System Control Registers

Table 6.2-1 Address Space Assignments for On-Chip Modules

6.2.5 Whole System Memory Mapping Table



6.2.6 System Manager Controller Registers Map

Register	Offset	R/W	Description	Reset Value
GCR_BA = 0x5000_0000				
PDID	GCR_BA+0x00	R	Part Device Identification number Register	0x0000_5200
RSTSRC	GCR_BA+0x04	R/W	System Reset Source Register	0x0000_00XX
IPRSTC1	GCR_BA+0x08	R/W	Peripheral Reset Control Resister1	0x0000_0000
IPRSTC2	GCR_BA+0x0C	R/W	Peripheral Reset Control Resister2	0x0000_0000
BODCR	GCR_BA+0x18	R/W	Brown Out Detector Control Register	0x0000_008X
PORCR	GCR_BA+0x24	R/W	Power-On-Reset Controller Register	0x0000_00xx
P0_MFP	GCR_BA+0x30	R/W	P0 multiple function and input type control register	0x0000_0000
P1_MFP	GCR_BA+0x34	R/W	P1 multiple function and input type control register	0x0000_0000
P2_MFP	GCR_BA+0x38	R/W	P2 multiple function and input type control register	0x0000_0000
P3_MFP	GCR_BA+0x3C	R/W	P3 multiple function and input type control register	0x0000_0000
P4_MFP	GCR_BA+0x40	R/W	P4 input type control register	0x0000_00C0
REGWRPROT	GCR_BA+0x100	R/W	Register Write Protect register	0x0000_0000

Part Device ID Code Register (PDID)

Register	Offset	R/W	Description	Reset Value
PDID	GCR_BA+0x00	R	Part Device Identification number Register	0x0000_5200[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. For example, M052LAN PDID code is 0x0000_5200.

NuMicro M051™ series	Part Device Identification Number
M052LAN	0x00005200
M054LAN	0x00005400
M058LAN	0x00005800
M0516LAN	0x00005A00
M052ZAN	0x00005203
M054ZAN	0x00005403
M058ZAN	0x00005803

M0516ZAN	0x00005A03

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+04	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_MCU	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]) with a "1" to rest Cortex-M0 CPU kernel and Flash memory controller(FMC). 1= The Cortex-M0 CPU kernel and FMC are reset by software set CPU_RST to 1. 0= No reset from CPU This bit is cleared by writing 1 to itself.
[6]	**Reserved**	Reserved
[5]	**RSTS_MCU**	The **RSTS_MCU** flag is set by the "reset signal" from the MCU Cortex_M0 kernel to indicate the previous reset source. 1= The MCU Cortex_M0 had issued the reset signal to reset the system by software writing 1 to bit SYSRESTREQ(AIRCR[2], Application Interrupt and Reset Control Register) in system control registers of Cortex_M0 kernel.

		0= No reset from MCU This bit is cleared by writing 1 to itself.
[4]	**RSTS_BOD**	The **RSTS_BOD** flag is set by the "reset signal" from the Brown-Out-Detector module to indicate the previous reset source. 1= The Brown-Out-Detector module had issued the reset signal to reset the system. 0= No reset from BOD This bit is cleared by writing 1 to itself.
[3]	**RSTS_LVR**	The **RSTS_LVR** flag is set by the "reset signal" from the Low-Voltage-Reset module to indicate the previous reset source. 1= The LVR module had issued the reset signal to reset the system. 0= No reset from LVR This bit is cleared by writing 1 to itself.
[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 This bit is cleared by writing 1 to itself.
[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 Pin /RESET This bit is cleared by writing 1 to itself.
[0]	**RSTS_POR**	The **RSTS_POR** flag is set by the "reset signal", which is from the Power-On Reset(POR) module or bit CHIP_RST (IPRSTC1[0]) is set, to indicate the previous reset source. 1= The Power-On-Reset(POR) or CHIP_RST=1 had issued the reset signal to reset the system. 0= No reset from POR This bit is cleared by writing 1 to itself.

Peripheral Reset Control Register1 (IPRSTC1)

Register	Offset	R/W	Description	Reset Value
IPRSTC1	GCR_BA+08	R/W	Peripheral Reset Control Resister 1	0x0000_0000

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

Bits	Descriptions	
[31:4]	**Reserved**	Reserved
[3]	**EBI_RST**	**EBI Controller Reset** Set these bit “1” will generate a reset signal to the EBI. User need to set this bit to “0” to release from the reset state This bit is the protected bit. It means programming this needs to write “59h”, “16h”, “88h” to address 0x5000_0100 to disable register protection. Reference the register REGWRPROT at address GCR_BA+0x100. 0= Normal operation 1= EBI IP reset
[2]	**Reserved**	Reserved
[1]	**CPU_RST**	**CPU kernel one shot reset.** Set this bit will reset the Cortex-M0 CPU kernel and Flash memory controller (FMC). This bit will automatically return to “0” after the 2 clock cycles 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. 0= Normal 1= Reset CPU
[0]	**CHIP_RST**	**CHIP one shot reset.** Set this bit will reset the 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 module is reset and the chip setting from flash are also reload 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. 0= Normal 1= Reset CHIP

Peripheral Reset Control Register2 (IPRSTC2)

Set these bit "1" will generate asynchronous reset signal to the correspond IP. User need to set bit to "0" to release IP from the reset state

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

31	30	29	28	27	26	25	24
Reserved			**ADC_RST**	**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**PWM47_RST**	**PWM03_RST**	**Reserved**		**UART1_RST**	**UART0_RST**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**SPI1_RST**	**SPI0_RST**	**Reserved**			**I2C_RST**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TMR3_RST**	**TMR2_RST**	**TMR1_RST**	**TMR0_RST**	**GPIO_RST**	**Reserved**

Bits	Descriptions	
[31:29]	**Reserved**	Reserved
[28]	**ADC_RST**	ADC Controller Reset 0= ADC controller normal operation 1= ADC controller reset
[27:22]	**Reserved**	Reserved
[21]	**PWM47_RST**	PWM4~7 controller Reset 0= PWM4~7 controller normal operation 1= PWM4~7 controller reset
[20]	**PWM03_RST**	PWM0~3 controller Reset 0= PWM0~3 controller normal operation

		1= PWM0~3 controller reset
[19:18]	**Reserved**	Reserved
[17]	**UART1_RST**	UART1 controller Reset 0= UART1 controller normal operation 1= UART1 controller reset
[16]	**UART0_RST**	UART0 controller Reset 0= UART0 controller normal operation 1= UART0 controller reset
[15:14]	**Reserved**	Reserved
[13]	**SPI1_ RST**	SPI1 controller Reset 0= SPI1 controller normal operation 1= SPI1 controller reset
[12]	**SPI0_ RST**	SPI0 controller Reset 0= SPI0 controller normal operation 1= SPI0 controller reset
[11:9]	**Reserved**	Reserved
[8]	**I2C_RST**	I2C controller Reset 0= I2C controller normal operation 1= I2C controller reset
[7:6]	**Reserved**	Reserved
[5]	**TMR3_RST**	Timer3 controller Reset 0= Timer3 controller normal operation 1= Timer3 controller reset
[4]	**TMR2_RST**	Timer2 controller Reset 0= Timer2 controller normal operation 1= Timer2 controller reset
[3]	**TMR1_RST**	Timer1 controller Reset 0= Timer1 controller normal operation 1= Timer1 controller reset
[2]	**TMR0_RST**	Timer0 controller Reset 0= Timer0 controller normal operation

		1= Timer0 controller reset
[1]	**GPIO_RST**	GPIO (P0~P4) controller Reset 0= GPIO controller normal operation 1= GPIO controller reset
[0]	**Reserved**	Reserved

Brown-Out Detector Control Register (BODCR)

Partial of the BODCR control registers values are initiated by the flash configuration and write-protected. Programming these protected 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+18	R/W	Brown Out Detector Control Register	0x0000_008X

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

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7]	**LVR_EN**	Low Voltage Reset Enable **(write-protected bit)** The LVR function reset the chip when the input power voltage is lower than LVR circuit setting. LVR function is enabled in default. The typical value of LVR is about 2.0V. 1= Enabled Low Voltage Reset function – After enable the bit, the LVR function will active with 100uS delay for LVR output stable.**(default)**. 0= Disabled Low Voltage Reset function
[6]	**BOD_OUT**	The status for Brown Out Detector output state 1= Brown Out Detector status output is 1, the detected voltage is lower than BOD_VL setting. If the BOD_EN is “0”(disabled), this bit always response “0” 0= Brown Out Detector status output is 0, the detected voltage is higher than BOD_VL setting

<table>
<tr><td>[5]</td><td>BOD_LPM</td><td>Brown Out Detector Low power Mode (write-protected bit)
1= Enable the BOD low power mode
0= BOD operate in normal mode (default)
The BOD consumes about 100uA in normal mode, the low power mode can reduce the current to about 1/10 but slow the BOD response.</td></tr>
<tr><td>[4]</td><td>BOD_INTF</td><td>Brown Out Detector Interrupt Flag
1= When Brown Out Detector detects the V_{DD} is dropped through the voltage of BOD_VL setting or the V_{DD} is raised up through the voltage of BOD_VL setting, this bit is set to “1” and the brown out interrupt is requested if brown out interrupt is enabled.
0= Brown Out Detector does not detect any voltage draft at V_{DD} down through or up through the voltage of BOD_VL setting.</td></tr>
<tr><td>[3]</td><td>BOD_RSTEN</td><td>Brown Out Reset Enable (initiated & write-protected bit)
1= Enable the Brown Out “RESET” function, when the Brown Out Detector function is enable and the detected voltage is lower than the threshold then assert a signal to reset the chip
The default value is set by flash controller user configuration register config0 bit[20]
0= Enable the Brown Out “INTERRUPT” function, when the Brown Out Detector function is enable and the detected voltage is lower than the threshold then assert a signal to interrupt the MCU Cortex-M0
When the BOD_EN is enabled and the interrupt is assert, the interrupt will keep till to the BOD_EN set to “0”. The interrupt for CPU can be blocked by disable the NVIC in CPU for BOD interrupt or disable the interrupt source by disable the BOD_EN and then re-enable the BOD_EN function if the BOD function is required</td></tr>
<tr><td>[2:1]</td><td>BOD_VL</td><td>Brown Out Detector Threshold Voltage Selection (initiated & write-protected bit)
The default value is set by flash controller user configuration register config0 bit[22:21]
<table>
<tr><td>BOV_VL[1]</td><td>BOV_VL[0]</td><td>Brown out voltage</td></tr>
<tr><td>1</td><td>1</td><td>4.5V</td></tr>
<tr><td>1</td><td>0</td><td>3.8V</td></tr>
<tr><td>0</td><td>1</td><td>2.7V</td></tr>
<tr><td>0</td><td>0</td><td>2.2V</td></tr>
</table></td></tr>
<tr><td>[0]</td><td>BOD_EN</td><td>Brown Out Detector Enable (initiated & write-protected bit)
The default value is set by flash controller user configuration register config0 bit[23]
1= Brown Out Detector function is enabled
0= Brown Out Detector function is disabled</td></tr>
</table>

Power-On-Reset Control Register (PORCR)

Register	Offset	R/W	Description	Reset Value
PORCR	GCR_BA+0x24	R/W	Power-On-Reset Controller Register	0x0000_00xx

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
POR_DIS_CODE[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. 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. If set the POR_DIS_CODE equal to 0x5AA5, the POR reset function will be disabled and the POR function will re-active till the power voltage is lower to set the POR_DIS_CODE to another value or reset by chip other reset function. Include: /RESET, Watch dog, 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 Port0 Control Register (P0_MFP)

Register	Offset	R/W	Description	Reset Value
P0_MFP	GCR_BA+30	R/W	P0 multiple function and input type control register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
P0_TYPE[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
P0_ALT[7:0]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P0_MFP[7:0]							

Bits	Descriptions	
[31:24]	**Reserved**	Reserved
[23:16]	**P0_TYPEn**	**P0[7:0] input Schmitt Trigger function Enable** 1= P0[7:0] I/O input Schmitt Trigger function enable 0= P0[7:0] I/O input Schmitt Trigger function disable
[15]	**P0_ ALT[7]**	P0.7 alternate function Selection The pin function of P0.7 is depend on P0_MFP[7] and P0_ALT[7]. P0_ALT[7] / P0_MFP[7] / P0.7 function 0 / 0 / P0.7 0 / 1 / AD7(EBI) 1 / 0 / SPICLK1(SPI1) 1 / 1 / Reserved

<table>
<tr>
<td>[14]</td>
<td>P0_ ALT[6]</td>
<td>P0.6 alternate function Selection
The pin function of P0.6 depends on P0_MFP[6] and P0_ALT[6].
<table>
<tr><td>P0_ALT[6]</td><td>P0_MFP[6]</td><td>P0.6 function</td></tr>
<tr><td>0</td><td>0</td><td>P0.6</td></tr>
<tr><td>0</td><td>1</td><td>AD6(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>MISO_1(SPI1)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td>
</tr>
<tr>
<td>[13]</td>
<td>P0_ ALT[5]</td>
<td>P0.5 alternate function Selection
The pin function of P0.5 is depend on P0_MFP[5] and P0_ALT[5].
<table>
<tr><td>P0_ALT[5]</td><td>P0_MFP[5]</td><td>P0.5 function</td></tr>
<tr><td>0</td><td>0</td><td>P0.5</td></tr>
<tr><td>0</td><td>1</td><td>AD5(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>MOSI_1(SPI1)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td>
</tr>
<tr>
<td>[12]</td>
<td>P0_ ALT[4]</td>
<td>P0.4 alternate function Selection
The pin function of P0.4 depends on P0_MFP[4] and P0_ALT[4].
<table>
<tr><td>P0_ALT[4]</td><td>P0_MFP[4]</td><td>P0.4function</td></tr>
<tr><td>0</td><td>0</td><td>P0.4</td></tr>
<tr><td>0</td><td>1</td><td>AD4(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>SPISS1(SPI1)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td>
</tr>
<tr>
<td>[11]</td>
<td>P0_ ALT[3]</td>
<td>P0.3 alternate function Selection
The pin function of P0.3 depends on P0_MFP[3] and P0_ALT[3].
<table>
<tr><td>P0_ALT[3]</td><td>P0_MFP[3]</td><td>P0.3function</td></tr>
<tr><td>0</td><td>0</td><td>P0.3</td></tr>
<tr><td>0</td><td>1</td><td>AD3(EBI)</td></tr>
</table>
</td>
</tr>
</table>

<table>
<tr><td></td><td></td><td>
<table>
<tr><td>1</td><td>0</td><td>RTS0(UART0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[10]</td><td>P0_ ALT[2]</td><td>P0.2 alternate function Selection
The pin function of P0.2 depends on P0_MFP[2] and P0_ALT[2].
<table>
<tr><td>P0_ALT[2]</td><td>P0_MFP[2]</td><td>P0.2function</td></tr>
<tr><td>0</td><td>0</td><td>P0.2</td></tr>
<tr><td>0</td><td>1</td><td>AD2(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>CTS0(UART0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[9]</td><td>P0_ ALT[1]</td><td>P0.1 alternate function Selection
The pin function of P0.1 depends on P0_MFP[1] and P0_ALT[1].
<table>
<tr><td>P0_ALT[1]</td><td>P0_MFP[1]</td><td>P0.1function</td></tr>
<tr><td>0</td><td>0</td><td>P0.1</td></tr>
<tr><td>0</td><td>1</td><td>AD1(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>RTS1(UART1)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[8]</td><td>P0_ ALT[0]</td><td>P0.0 alternate function Selection
The pin function of P0.0 depends on P0_MFP[0] and P0_ALT[0].
<table>
<tr><td>P0_ALT[0]</td><td>P0_MFP[0]</td><td>P0.0function</td></tr>
<tr><td>0</td><td>0</td><td>P0.0</td></tr>
<tr><td>0</td><td>1</td><td>AD0(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>CTS1(UART1)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[7:0]</td><td>P0_MFP[7:0]</td><td>P0 multiple function Selection
The pin function of P0 depends on P0_MFP and P0_ALT.</td></tr>
</table>

		Refer to P0_ALT for details descriptions.

Multiple Function Port1 Control Register (P1_MFP)

Register	Offset	R/W	Description	Reset Value
P1_MFP	GCR_BA+34	R/W	P1 multiple function and input type control register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
P1_TYPE[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
P1_ALT[7:0]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P1_MFP[7:0]							

Bits	Descriptions	
[31:24]	Reserved	Reserved
[23:16]	P1_TYPEn	**P1[7:0] input Schmitt Trigger function Enable** 1= P1[7:0] I/O input Schmitt Trigger function enable 0= P1[7:0] I/O input Schmitt Trigger function disable
[15]	P1_ ALT[7]	**P1.7 alternate function Selection** The pin function of P1.7 depends on P1_MFP[7] and P1_ALT[7].

P1_ALT[7]	P1_MFP[7]	P1.7 function
0	0	P1.7
0	1	AIN7(ADC)
1	0	SPICLK0(SPI0)
1	1	Reserved

<table>
<tr>
<td>[14]</td>
<td>P1_ ALT[6]</td>
<td>P1.6 alternate function Selection
The pin function of P1.6 depends on P1_MFP[6] and P1_ALT[6].
<table>
<tr><th>P1_ALT[6]</th><th>P1_MFP[6]</th><th>P1.6 function</th></tr>
<tr><td>0</td><td>0</td><td>P1.6</td></tr>
<tr><td>0</td><td>1</td><td>AIN6(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>MISO_0(SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td>
</tr>
<tr>
<td>[13]</td>
<td>P1_ ALT[5]</td>
<td>P1.5 alternate function Selection
The pin function of P1.5 depends on P1_MFP[5] and P1_ALT[5].
<table>
<tr><th>P1_ALT[5]</th><th>P1_MFP[5]</th><th>P1.5 function</th></tr>
<tr><td>0</td><td>0</td><td>P1.5</td></tr>
<tr><td>0</td><td>1</td><td>AIN5(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>MOSI_0(SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td>
</tr>
<tr>
<td>[12]</td>
<td>P1_ ALT[4]</td>
<td>P1.4 alternate function Selection
The pin function of P1.4 depends on P1_MFP[4] and P1_ALT[4].
<table>
<tr><th>P1_ALT[4]</th><th>P1_MFP[4]</th><th>P1.4function</th></tr>
<tr><td>0</td><td>0</td><td>P1.4</td></tr>
<tr><td>0</td><td>1</td><td>AIN4(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>SPISS0(SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td>
</tr>
<tr>
<td>[11]</td>
<td>P1_ ALT[3]</td>
<td>P1.3 alternate function Selection
The pin function of P1.3 depends on P1_MFP[3] and P1_ALT[3].
<table>
<tr><th>P1_ALT[3]</th><th>P1_MFP[3]</th><th>P1.3function</th></tr>
<tr><td>0</td><td>0</td><td>P1.3</td></tr>
<tr><td>0</td><td>1</td><td>AIN3(ADC)</td></tr>
</table>
</td>
</tr>
</table>

<table>
<tr><td></td><td></td><td>
<table>
<tr><td>1</td><td>0</td><td>TXD1(UART1)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[10]</td><td>P1_ ALT[2]</td><td>P1.2 alternate function Selection
The pin function of P1.2 depends on P1_MFP[2] and P1_ALT[2].
<table>
<tr><td>P1_ALT[2]</td><td>P1_MFP[2]</td><td>P1.2function</td></tr>
<tr><td>0</td><td>0</td><td>P1.2</td></tr>
<tr><td>0</td><td>1</td><td>AIN2(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>RXD1(UART1)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[9]</td><td>P1_ ALT[1]</td><td>P1.1 alternate function Selection
The pin function of P1.1 depends on P1_MFP[1] and P1_ALT[1].
<table>
<tr><td>P1_ALT[1]</td><td>P1_MFP[1]</td><td>P1.1function</td></tr>
<tr><td>0</td><td>0</td><td>P1.1</td></tr>
<tr><td>0</td><td>1</td><td>AIN1(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>T3(Timer3)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[8]</td><td>P1_ ALT[0]</td><td>P1.0 alternate function Selection
The pin function of P1.0 depends on P1_MFP[0] and P1_ALT[0].
<table>
<tr><td>P1_ALT[0]</td><td>P1_MFP[0]</td><td>P1.0function</td></tr>
<tr><td>0</td><td>0</td><td>P1.0</td></tr>
<tr><td>0</td><td>1</td><td>AIN0(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>T2(Timer2)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[7:0]</td><td>P1_MFP[7:0]</td><td>P1 multiple function Selection
The pin function of P1 is depending on P1_MFP and P1_ALT.</td></tr>
</table>

		Refer to P1_ALT for details descriptions.

Multiple Function Port2 Control Register (P2_MFP)

Register	Offset	R/W	Description	Reset Value
P2_MFP	GCR_BA+38	R/W	P2 multiple function and input type control register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
P2_TYPE[7:0]							
15	14	13	12	11	10	9	8
P2_ALT[7:0]							
7	6	5	4	3	2	1	0
P2_MFP[7:0]							

Bits	Descriptions	
[31:24]	Reserved	Reserved
[23:16]	P2_TYPEn	**P2[7:0] input Schmitt Trigger function Enable** 1= P2[7:0] I/O input Schmitt Trigger function enable 0= P2[7:0] I/O input Schmitt Trigger function disable
[15]	P2_ ALT[7]	P2.7 alternate function Selection The pin function of P2.7 depends on P2_MFP[7] and P2_ALT[7]. P2_ALT[7] / P2_MFP[7] / P2.7 function 0 / 0 / P2.7 0 / 1 / AD15(EBI) 1 / 0 / PWM7(PWM generator 6)

<table>
<tr><td></td><td></td><td><table>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[14]</td><td>P2_ ALT[6]</td><td>P2.6 alternate function Selection
The pin function of P2.6 depends on P2_MFP[6] and P2_ALT[6].<table>
<tr><td>P2_ALT[6]</td><td>P2_MFP[6]</td><td>P2.6 function</td></tr>
<tr><td>0</td><td>0</td><td>P2.6</td></tr>
<tr><td>0</td><td>1</td><td>AD14(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM6(PWM generator 6)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[13]</td><td>P2_ ALT[5]</td><td>P2.5 alternate function Selection
The pin function of P2.5 depends on P2_MFP[5] and P2_ALT[5].<table>
<tr><td>P2_ALT[5]</td><td>P2_MFP[5]</td><td>P2.5 function</td></tr>
<tr><td>0</td><td>0</td><td>P2.5</td></tr>
<tr><td>0</td><td>1</td><td>AD13(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM5(PWM generator 4)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[12]</td><td>P2_ ALT[4]</td><td>P2.4 alternate function Selection
The pin function of P2.4 depends on P2_MFP[4] and P2_ALT[4].<table>
<tr><td>P2_ALT[4]</td><td>P2_MFP[4]</td><td>P2.4function</td></tr>
<tr><td>0</td><td>0</td><td>P2.4</td></tr>
<tr><td>0</td><td>1</td><td>AD12(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM4(PWM generator 4)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[11]</td><td>P2_ ALT[3]</td><td>P2.3 alternate function Selection
The pin function of P2.3 depends on P2_MFP[3] and P2_ALT[3].</td></tr>
</table>

<table>
<tr><td></td><td></td><td>
<table>
<tr><td>P2_ALT[3]</td><td>P2_MFP[3]</td><td>P2.3function</td></tr>
<tr><td>0</td><td>0</td><td>P2.3</td></tr>
<tr><td>0</td><td>1</td><td>AD11(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM3(PWM generator 2)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[10]</td><td>P2_ ALT[2]</td><td>P2.2 alternate function Selection
The pin function of P2.2 depends on P2_MFP[2] and P2_ALT[2].
<table>
<tr><td>P2_ALT[2]</td><td>P2_MFP[2]</td><td>P2.2function</td></tr>
<tr><td>0</td><td>0</td><td>P2.2</td></tr>
<tr><td>0</td><td>1</td><td>AD10(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM2(PWM generator 2)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[9]</td><td>P2_ ALT[1]</td><td>P2.1 alternate function Selection
The pin function of P2.1 depends on P2_MFP[1] and P2_ALT[1].
<table>
<tr><td>P2_ALT[1]</td><td>P2_MFP[1]</td><td>P2.1function</td></tr>
<tr><td>0</td><td>0</td><td>P2.1</td></tr>
<tr><td>0</td><td>1</td><td>AD9(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM1(PWM generator 0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[8]</td><td>P2_ ALT[0]</td><td>P2.0 alternate function Selection
The pin function of P2.0 depends on P2_MFP[0] and P2_ALT[0].
<table>
<tr><td>P2_ALT[0]</td><td>P2_MFP[0]</td><td>P2.0function</td></tr>
<tr><td>0</td><td>0</td><td>P2.0</td></tr>
<tr><td>0</td><td>1</td><td>AD8(EBI)</td></tr>
</table>
</td></tr>
</table>

		1	0	PWM0(PWM generator 0)
		1	1	Reserved
[7:0]	P2_MFP[7:0]	P2 multiple function Selection The pin function of P2 depends on P2_MFP and P2_ALT. Refer to P2_ALT for details descriptions.		

Multiple Function Port3 Control Register (P3_MFP)

Register	Offset	R/W	Description	Reset Value
P3_MFP	GCR_BA+3C	R/W	P3 multiple function and input type control register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
P3_TYPE[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
P3_ALT[7:0]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P3_MFP[7:0]							

Bits	Descriptions	
[31:24]	Reserved	Reserved
[23:16]	P3_TYPEn	**P3[7:0] input Schmitt Trigger function Enable** 1= P3[7:0] I/O input Schmitt Trigger function enable 0= P3[7:0] I/O input Schmitt Trigger function disable
[15]	P3_ ALT[7]	P3.7 alternate function Selection The pin function of P3.7 is depend on P3_MFP[7] and P3_ALT[7]. P3_ALT[7] / P3_MFP[7] / P3.7 function: 0 / 0 / P3.7 0 / 1 / RD(EBI) 1 / x / Reserved
[14]	P3_ ALT[6]	P3.6 alternate function Selection The pin function of P3.6 depends on P3_MFP[6] and P3_ALT[6].

		P3_ALT[6] / P3_MFP[6] / P3.6 function 0 / 0 / P3.6 0 / 1 / WR(EBI) 1 / 0 / CKO(Clock Driver output) 1 / 1 / Reserved
[13]	P3_ ALT[5]	P3.5 alternate function Selection The pin function of P3.5 depends on P3_MFP[5] and P3_ALT[5]. P3_ALT[5] / P3_MFP[5] / P3.5 function 0 / 0 / P3.5 0 / 1 / T1(Timer1) 1 / 0 / SCL(I2C) 1 / 1 / Reserved
[12]	P3_ ALT[4]	P3.4 alternate function Selection The pin function of P3.4 depends on P3_MFP[4] and P3_ALT[4]. P3_ALT[4] / P3_MFP[4] / P3.4function 0 / 0 / P3.4 0 / 1 / T0(Timer0) 1 / 0 / SDA(I2C) 1 / 1 / Reserved
[11]	P3_ ALT[3]	P3.3 alternate function Selection The pin function of P3.3 depends on P3_MFP[3] and P3_ALT[3]. P3_ALT[3] / P3_MFP[3] / P3.3function 0 / 0 / P3.3 0 / 1 / /INT1 1 / 0 / MCLK(EBI) 1 / x / Reserved

<table>
<tr><td>[10]</td><td>P3_ ALT[2]</td><td>P3.2 alternate function Selection
The pin function of P3.2 depends on P3_MFP[2] and P3_ALT[2].
<table>
<tr><td>P3_ALT[2]</td><td>P3_MFP[2]</td><td>P3.2function</td></tr>
<tr><td>0</td><td>0</td><td>P3.2</td></tr>
<tr><td>0</td><td>1</td><td>/INT0</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[9]</td><td>P3_ ALT[1]</td><td>P3.1 alternate function Selection
The pin function of P3.1 depends on P3_MFP[1] and P3_ALT[1].
<table>
<tr><td>P3_ALT[1]</td><td>P3_MFP[1]</td><td>P3.1function</td></tr>
<tr><td>0</td><td>0</td><td>P3.1</td></tr>
<tr><td>0</td><td>1</td><td>TXD(UART0)</td></tr>
<tr><td>1</td><td>x</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[8]</td><td>P3_ ALT[0]</td><td>P3.0 alternate function Selection
The pin function of P3.0 depends on P3_MFP[0] and P3_ALT[0].
<table>
<tr><td>P3_ALT[0]</td><td>P3_MFP[0]</td><td>P3.0function</td></tr>
<tr><td>0</td><td>0</td><td>P3.0</td></tr>
<tr><td>0</td><td>1</td><td>RXD(UART0)</td></tr>
<tr><td>1</td><td>x</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[7:0]</td><td>P3_MFP[7:0]</td><td>P3 multiple function Selection
The pin function of P3 is depending on P3_MFP and P3_ALT.
Refer to P3_ALT for details descriptions.</td></tr>
</table>

Multiple Function Port4 Control Register (P4_MFP)

Register	Offset	R/W	Description	Reset Value
P4_MFP	GCR_BA+40	R/W	P4 multiple function and input type control register	0x0000_00C0

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
P4_TYPE[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
P4_ALT[7:0]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P4_MFP[7:0]							

Bits	Descriptions	
[31:24]	Reserved	Reserved
[23:16]	P4_TYPEn	**P4[7:0] input Schmitt Trigger function Enable** 1= P4[7:0] I/O input Schmitt Trigger function enable 0= P4[7:0] I/O input Schmitt Trigger function disable
[15]	P4_ ALT[7]	P4.7 alternate function Selection The pin function of P4.7 depends on P4_MFP[7] and P4_ALT[7]. P4_ALT[7] / P4_MFP[7] / P4.7 function 0 / 0 / P4.7 0 / 1 / ICE_DAT(ICE) 1 / x / Reserved
[14]	P4_ ALT[6]	P4.6 alternate function Selection

<table>
<tr><td></td><td></td><td>The pin function of P4.6 depends on P4_MFP[6] and P4_ALT[6].<table><tr><td>P4_ALT[6]</td><td>P4_MFP[6]</td><td>P4.6 function</td></tr><tr><td>0</td><td>0</td><td>P4.6</td></tr><tr><td>0</td><td>1</td><td>ICE_CLK(ICE)</td></tr><tr><td>1</td><td>x</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[13]</td><td>P4_ ALT[5]</td><td>P4.5 alternate function Selection
The pin function of P4.5 depends on P4_MFP[5] and P4_ALT[5].<table><tr><td>P4_ALT[5]</td><td>P4_MFP[5]</td><td>P4.5 function</td></tr><tr><td>0</td><td>0</td><td>P4.5</td></tr><tr><td>0</td><td>1</td><td>ALE(EBI)</td></tr><tr><td>1</td><td>x</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[12]</td><td>P4_ ALT[4]</td><td>P4.4 alternate function Selection
The pin function of P4.4 depends on P4_MFP[4] and P4_ALT[4].<table><tr><td>P4_ALT[4]</td><td>P4_MFP[4]</td><td>P4.4function</td></tr><tr><td>0</td><td>0</td><td>P4.4</td></tr><tr><td>0</td><td>1</td><td>/CS(EBI)</td></tr><tr><td>1</td><td>x</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[11]</td><td>P4_ ALT[3]</td><td>P4.3 alternate function Selection
The pin function of P4.3 depends on P4_MFP[3] and P4_ALT[3].<table><tr><td>P4_ALT[3]</td><td>P4_MFP[3]</td><td>P4.3function</td></tr><tr><td>0</td><td>0</td><td>P4.3</td></tr><tr><td>0</td><td>1</td><td>PWM3(PWM generator 2)</td></tr><tr><td>1</td><td>x</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[10]</td><td>P4_ ALT[2]</td><td>P4.2 alternate function Selection
The pin function of P4.2 depends on P4_MFP[2] and P4_ALT[2].</td></tr>
</table>

<table>
<tr><td></td><td></td><td>
<table>
<tr><td>P4_ALT[2]</td><td>P4_MFP[2]</td><td>P4.2function</td></tr>
<tr><td>0</td><td>0</td><td>P4.2</td></tr>
<tr><td>0</td><td>1</td><td>PWM2(PWM generator 2)</td></tr>
<tr><td>1</td><td>x</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[9]</td><td>P4_ ALT[1]</td><td>P4.1 alternate function Selection
The pin function of P4.1 depends on P4_MFP[1] and P4_ALT[1].
<table>
<tr><td>P4_ALT[1]</td><td>P4_MFP[1]</td><td>P4.1function</td></tr>
<tr><td>0</td><td>0</td><td>P4.1</td></tr>
<tr><td>0</td><td>1</td><td>PWM1(PWM generator 0)</td></tr>
<tr><td>1</td><td>x</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[8]</td><td>P4_ ALT[0]</td><td>P4.0 alternate function Selection
The pin function of P4.0 depends on P4_MFP[0] and P4_ALT[0].
<table>
<tr><td>P4_ALT[0]</td><td>P4_MFP[0]</td><td>P4.0function</td></tr>
<tr><td>0</td><td>0</td><td>P4.0</td></tr>
<tr><td>0</td><td>1</td><td>PWM0(PWM generator 0)</td></tr>
<tr><td>1</td><td>x</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[7:0]</td><td>P4_MFP[7:0]</td><td>P4 multiple function Selection
The pin function of P4 is depending on P4_MFP and P4_ALT.
Refer to P4_ALT for details descriptions.</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, “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.

Writing to this register to disable/enable register protection ,and reading it to get the REGPROTDIS status

Register	Offset	R/W	Description	Reset Value
REGWRPROT	GCR_BA+100	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

<table>
<tr><td>[31:16]</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>[7:0]</td><td>REGWRPROT</td><td>Register Write-Protected Code (Write Only)
Some write-protected registers have to be disabled 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-protected registers can be normal write.</td></tr>
<tr><td>[0]</td><td>REGPROTDIS</td><td>Register Write–Protected Disable index (Read only)
1 = Protection is disabled for writing protected registers
0 = Protection is enabled for writing protected registers. Any write to the protected register is ignored.
The Write-Protected registers list are below table:
<table>
<tr><th>Registers</th><th>Address</th><th>Note</th></tr>
<tr><td>IPRSTC1</td><td>0x5000_0008</td><td></td></tr>
<tr><td>BODCR</td><td>0x5000_0018</td><td></td></tr>
<tr><td>PORCR</td><td>0x5000_001C</td><td></td></tr>
<tr><td>PWRCON</td><td>0x5000_0200</td><td>bit[6] is not protected for power wake-up interrupt clear</td></tr>
<tr><td>APBCLK bit[0]</td><td>0x5000_0208</td><td>bit[0] is watch dog clock enable</td></tr>
<tr><td>CLKSEL0</td><td>0x5000_0210</td><td>HCLK and CPU STCLK clock source select</td></tr>
<tr><td>CLK_SEL1 bit[1:0]</td><td>0x5000_0214</td><td>Watch dog clock source select</td></tr>
<tr><td>ISPCON</td><td>0x5000_C000</td><td>Flash ISP Control register</td></tr>
<tr><td>WTCR</td><td>0x4000_4000</td><td></td></tr>
<tr><td>FATCON</td><td>0x5000_C018</td><td></td></tr>
</table></td></tr>
</table>

6.2.7 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 as a Real Time Operating System (RTOS) tick timer or as a simple counter.

When system timer is enabled, it will count down from the value in the SysTick Current Value Register (SYST_CVR) to zero, and reload (wrap) to the value in the SysTick Reload Value Register (SYST_RVR) on the next clock edge, 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”.

6.2.7.1 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+010	R/W	SysTick Control and Status	0x0000_0004
SYST_RVR	SCS_BA+014	R/W	SysTick Reload value	0xXXXX_XXXX
SYST_CVR	SCS_BA+018	R/W	SysTick Current value	0xXXXX_XXXX

SysTick Control and Status （SYST_CSR）

Register	Offset	R/W	Description	Reset Value
SYST_CSR	SCS_BA+0x10	R/W	SysTick Control and Status	0x0000_0004

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, refer to STCLK_S.
[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+014	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+ 018	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. Unsupported bits RAZ (see SysTick Reload Value register).

6.2.8 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".

6.2.8.1 Exception Model and System Interrupt Map

The Table 6.2-2 lists the exception model supported by NuMicro M051™ 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 Number	Vector Address	Interrupt Number (Bit in Interrupt Registers)	Interrupt Name	Source IP	Interrupt description	Power Down Wakeup
1-15	0x00-0x3C	-	-	-	System exceptions	
16	0x40	0	BOD_OUT	Brown-Out	Brownout low voltage detected interrupt	Yes
17	0x44	1	WDT_INT	WDT	Watch Dog Timer interrupt	Yes
18	0x48	2	EINT0	GPIO	External signal interrupt from P3.2 pin	Yes
19	0x4C	3	EINT1	GPIO	External signal interrupt from P3.3 pin	Yes
20	0x50	4	GP01_INT	GPIO	External signal interrupt from P0[7:0] / P1[7:0]	Yes
21	0x54	5	GP234_INT	GPIO	External interrupt from P2[7:0]/P3[7:0]/P4[7:0], except P32 and P33	Yes
22	0x58	6	PWMA_INT	PWM0~3	PWM0, PWM1, PWM2 and PWM3 interrupt	No
23	0x5C	7	PWMB_INT	PWM4~7	PWM4, PWM5, PWM6 and PWM7 interrupt	No
24	0x60	8	TMR0_INT	TMR0	Timer 0 interrupt	No
25	0x64	9	TMR1_INT	TMR1	Timer 1 interrupt	No
26	0x68	10	TMR2_INT	TMR2	Timer 2 interrupt	No
27	0x6C	11	TMR3_INT	TMR3	Timer 3 interrupt	No
28	0x70	12	UART0_INT	UART0	UART0 interrupt	Yes

29	0x74	13	UART1_INT	UART1	UART1 interrupt	Yes
30	0x78	14	SPI0_INT	SPI0	SPI0 interrupt	No
31	0x7C	15	SPI1_INT	SPI1	SPI1 interrupt	No
32-33	0x80-0x84	16-17	-	-	-	-
34	0x88	18	I2C_INT	I2C	I2C interrupt	No
35-43	0x8C-0xAC	19-27	-	-	-	-
44	0xB0	28	PWRWU_INT	CLKC	Clock controller interrupt for chip wake up from power-down state	Yes
45	0xB4	29	ADC_INT	ADC	ADC interrupt	No
46-47	0xB8-0xBC	30-31	-	-	-	

Table 6.2-2 Exception Model

Exception Name	Vector Number	Priority
Reset	**1**	**-3**
NMI	**2**	**-2**
Hard Fault	**3**	**-1**
Reserved	**4 ~ 10**	**Reserved**
SVCall	**11**	**Configurable**
Reserved	**12 ~ 13**	**Reserved**
PendSV	**14**	**Configurable**
SysTick	**15**	**Configurable**
Interrupt (IRQ0 ~ IRQ31)	**16 ~ 47**	**Configurable**

Table 6.2-3 System Interrupt Map

6.2.8.2 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 ARMv6-M, the vector table base address is fixed at 0x00000000. The vector table contains the initialization value for the stack pointer on reset, and the entry point addresses for all exception handlers. The vector number on previous page defines the order of entries in the vector table associated with exception handler entry as illustrated in previous section.

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

Table 6.2-4 Vector Table Format

6.2.8.3 Operation Description

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

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

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

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

6.2.8.4 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+100	R/W	IRQ0 ~ IRQ31 Set-Enable Control Register	0x0000_0000
NVIC_ICER	SCS_BA+180	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000
NVIC_ISPR	SCS_BA+200	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000
NVIC_ICPR	SCS_BA+280	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000
NVIC_IPR0	SCS_BA+400	R/W	IRQ0 ~ IRQ3 Priority Control Register	0x0000_0000
NVIC_IPR1	SCS_BA+404	R/W	IRQ4 ~ IRQ7 Priority Control Register	0x0000_0000
NVIC_IPR2	SCS_BA+408	R/W	IRQ8 ~ IRQ11 Priority Control Register	0x0000_0000
NVIC_IPR3	SCS_BA+40C	R/W	IRQ12 ~ IRQ15 Priority Control Register	0x0000_0000

NVIC_IPR4	SCS_BA+410	R/W	IRQ16 ~ IRQ19 Priority Control Register	0x0000_0000
NVIC_IPR5	SCS_BA+414	R/W	IRQ20 ~ IRQ23 Priority Control Register	0x0000_0000
NVIC_IPR6	SCS_BA+418	R/W	IRQ24 ~ IRQ27 Priority Control Register	0x0000_0000
NVIC_IPR7	SCS_BA+41C	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+100	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+180	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+200	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 pends 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+280	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 un-pends 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 ~ IRQ3 Interrupt Priority Register （NVIC_IPR0）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR0	SCS_BA+400	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 & “3” denotes lowest priority
[23:22]	**PRI_2**	Priority of IRQ2 “0” denotes the highest priority & “3” denotes lowest priority
[15:14]	**PRI_1**	Priority of IRQ1 “0” denotes the highest priority & “3” denotes lowest priority
[7:6]	**PRI_0**	Priority of IRQ0 “0” denotes the highest priority & “3” denotes lowest priority

IRQ4 ~ IRQ7 Interrupt Priority Register （NVIC_IPR1）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR1	SCS_BA+404	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 & “3” denotes lowest priority
[23:22]	**PRI_6**	Priority of IRQ6 “0” denotes the highest priority & “3” denotes lowest priority
[15:14]	**PRI_5**	Priority of IRQ5 “0” denotes the highest priority & “3” denotes lowest priority
[7:6]	**PRI_4**	Priority of IRQ4 “0” denotes the highest priority & “3” denotes lowest priority

IRQ8 ~ IRQ11 Interrupt Priority Register （NVIC_IPR2）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR2	SCS_BA+408	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 & “3” denotes lowest priority
[23:22]	**PRI_10**	Priority of IRQ10 “0” denotes the highest priority & “3” denotes lowest priority
[15:14]	**PRI_9**	Priority of IRQ9 “0” denotes the highest priority & “3” denotes lowest priority
[7:6]	**PRI_8**	Priority of IRQ8 “0” denotes the highest priority & “3” denotes lowest priority

IRQ12 ~ IRQ15 Interrupt Priority Register （NVIC_IPR3）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR3	SCS_BA+40C	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 & “3” denotes lowest priority
[23:22]	**PRI_14**	Priority of IRQ14 “0” denotes the highest priority & “3” denotes lowest priority
[15:14]	**PRI_13**	Priority of IRQ13 “0” denotes the highest priority & “3” denotes lowest priority
[7:6]	**PRI_12**	Priority of IRQ12 “0” denotes the highest priority & “3” denotes lowest priority

IRQ16 ~ IRQ19 Interrupt Priority Register （NVIC_IPR4）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR4	SCS_BA+ 410	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 & “3” denotes lowest priority
[23:22]	**PRI_18**	Priority of IRQ18 “0” denotes the highest priority & “3” denotes lowest priority
[15:14]	**PRI_17**	Priority of IRQ17 “0” denotes the highest priority & “3” denotes lowest priority
[7:6]	**PRI_16**	Priority of IRQ16 “0” denotes the highest priority & “3” denotes lowest priority

IRQ20 ~ IRQ23 Interrupt Priority Register （NVIC_IPR5）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR5	SCS _BA + 414	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 & “3” denotes lowest priority
[23:22]	**PRI_22**	Priority of IRQ22 “0” denotes the highest priority & “3” denotes lowest priority
[15:14]	**PRI_21**	Priority of IRQ21 “0” denotes the highest priority & “3” denotes lowest priority
[7:6]	**PRI_20**	Priority of IRQ20 “0” denotes the highest priority & “3” denotes lowest priority

IRQ24 ~ IRQ27 Interrupt Priority Register （NVIC_IPR6）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR6	SCS_BA+418	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 & “3” denotes lowest priority
[23:22]	**PRI_26**	Priority of IRQ26 “0” denotes the highest priority & “3” denotes lowest priority
[15:14]	**PRI_25**	Priority of IRQ25 “0” denotes the highest priority & “3” denotes lowest priority
[7:6]	**PRI_24**	Priority of IRQ24 “0” denotes the highest priority & “3” denotes lowest priority

IRQ28 ~ IRQ31 Interrupt Priority Register （NVIC_IPR7）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR7	SCS_BA+41C	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 & “3” denotes lowest priority
[23:22]	**PRI_30**	Priority of IRQ30 “0” denotes the highest priority & “3” denotes lowest priority
[15:14]	**PRI_29**	Priority of IRQ29 “0” denotes the highest priority & “3” denotes lowest priority
[7:6]	**PRI_28**	Priority of IRQ28 “0” denotes the highest priority & “3” denotes lowest priority

6.2.8.5 Interrupt Source Control Registers

Besides the interrupt control registers associated with the NVIC, NuMicro M051™ series also implement some specific control registers to facilitate the interrupt functions, including “interrupt source identify”, ”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 (P0/1) interrupt source identity	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (P2/3/4) interrupt source identity	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWMA) interrupt source identity	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (PWMB) interrupt source identity	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) interrupt source identity	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) interrupt source identity	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) interrupt source identity	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) interrupt source identity	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (URT0) interrupt source identity	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (URT1) 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	Reserved	Reserved	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	Reserved	Reserved	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I2C) interrupt source identity	0xXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	Reserved	Reserved	0xXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	Reserved	Reserved	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	Reserved	Reserved	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	Reserved	Reserved	0xXXXX_XXXX
IRQ23_SRC	INT_BA+0x5C	Reserved	Reserved	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	Reserved	Reserved	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	Reserved	Reserved	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	Reserved	Reserved	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	Reserved	Reserved	0xXXXX_XXXX
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) interrupt source identity	0xXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) interrupt source identity	0xXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	Reserved	Reserved	0xXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	Reserved	Reserved	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	MCU IRQ0 (BOD) interrupt source identity : MCU IRQ31 (Reserved) 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]		

Address	INT-Num	Bits	Descriptions
INT_BA+0x00	0	[2:0]	Bit2 = 0 Bit1 = 0 Bit0 : BOD_INT
INT_BA+0x04	1	[2:0]	Bit2 = 0 Bit1 = 0 Bit0 : WDT_INT
INT_BA+0x08	2	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: EINT0 – external interrupt 0 from P3.2
INT_BA+0x0C	3	[2:0]	Bit2 = 0 Bit1 = 0

			Bit0: EINT1 – external interrupt 1 from P3.3
INT_BA+0x10	4	[2:0]	Bit2 = 0 Bit1: P1_INT Bit0: P0_INT
INT_BA+0x14	5	[2:0]	Bit2: P4_INT Bit1: P3_INT Bit0: P2_INT
INT_BA+0x18	6	[3:0]	Bit3: PWM3_INT Bit2: PWM2_INT Bit1: PWM1_INT Bit0: PWM0_INT
INT_BA+0x1C	7	[3:0]	Bit3: PWM7_INT Bit2: PWM6_INT Bit1: PWM5_INT Bit0: PWM4_INT
INT_BA+0x20	8	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: TMR0_INT
INT_BA+0x24	9	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: TMR1_INT
INT_BA+0x28	10	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: TMR2_INT
INT_BA+0x2C	11	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: TMR3_INT
INT_BA+0x30	12	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: URT0_INT

INT_BA+0x34	13	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: URT1_INT
INT_BA+0x38	14	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: SPI0_INT
INT_BA+0x3C	15	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: SPI1_INT
INT_BA+0x40	16	[2:0]	Reserved
INT_BA+0x44	17	[2:0]	Reserved
INT_BA+0x48	18	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: I2C_INT
INT_BA+0x4C	19	[2:0]	Reserved
INT_BA+0x50	20	[2:0]	Reserved
INT_BA+0x54	21	[2:0]	Reserved
INT_BA+0x58	22	[2:0]	Reserved
INT_BA+0x5C	23	[2:0]	Reserved
INT_BA+0x60	24	[2:0]	Reserved
INT_BA+0x64	25	[2:0]	Reserved
INT_BA+0x68	26	[2:0]	Reserved
INT_BA+0x6C	27	[2:0]	Reserved
INT_BA+0x70	28	[2:0]	Bit2 = 0 Bit1 = 0

			Bit0: PWRWU_INT
INT_BA+0x74	29	[2:0]	Bit2 = 0 Bit1 = 0 Bit0: ADC_INT
INT_BA+0x78	30	[2:0]	Reserved
INT_BA+0x7C	31	[2:0]	Reserved

NMI Interrupt Source Select Control Register (NMI_SEL)

Register	Offset	R/W	Description	Reset Value
NMI_SEL	INT_BA+0x80	R/W	NMI source interrupt select control register	0x0000_0000

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

Bits	Descriptions	
[31:5]	**Reserved**	Reserved
[4:0]	**NMI_SEL**	The NMI interrupt to Cortex-M0 can be selected from one of the interrupt[31:0] The NMI_SEL bit[4:0] used to select the NMI interrupt source

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 core. 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_bit[n] will clear the interrupt and set MCU_IRQ[n] “0” : no any effect

6.2.9 System Controller Registers Map

Cortex-M0 status and operating mode control are managed System Control Registers. Including CPUID, Cortex-M0 interrupt priority and Cortex-M0power 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 Base Register	0x0000_0000
ICSR	SCS_BA+ 0xD04	R/W	Interrupt Control 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 Base Register (CPUID)

Register	Offset	R/W	Description	Reset Value
CPUID	SCS_BA + 0xD00	R	CPUID Register	0x410CC200

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 ARMv6-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 State Register	0x00000000

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[8:4]				
15	14	13	12	11	10	9	8
VECTPENDING[3:0]				Reserved			VECTACTIVE[8]
7	6	5	4	3	2	1	0
VECTACTIVE[7:0]							

Bits	R/W	Descriptions	
[31]	R/W	NMIPENDSET	Setting this bit will activate an NMI. Since NMI is the highest priority exception, it will activate as soon as it is registered. Reads back with current state (1 if Pending, 0 if not).
[28]	R/W	PENDSVSET	Set a pending PendSV interrupt. This is normally used to request a context switch. Reads back with current state (1 if Pending, 0 if not).
[27]	W	PENDSVCLR	Write 1 to clear a pending PendSV interrupt.
[26]	R/W	PENDSTSET	Set a pending SysTick. Reads back with current state (1 if Pending, 0 if not).
[25]	W	PENDSTCLR	Write 1 to clear a pending SysTick.

[23]	R	ISRPREEMPT	If set, a pending exception will be serviced on exit from the debug halt state.
[22]	R	ISRPENDING	Indicates if an external configurable (NVIC generated) interrupt is pending.
[20:12]	R	VECTPENDING	Indicates the exception number for the highest priority pending exception. The pending state includes the effect of memory-mapped enable and mask registers. It does not include the PRIMASK special-purpose register qualifier. A value of zero indicates no pending exceptions.
[8:0]	R	VECTACTIVE	0= Thread mode value > 1, the exception number for the current executing 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					**SYSRESET REQ**	**VECTCLKA CTIVE**	**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	0x00000000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	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	
[4]	SEVONPEND	When enabled, interrupt transitions from Inactive to Pending are included in the list of wakeup events for the WFE instruction.
[2]	SLEEPDEEP	A qualifying hint that indicates waking from sleep might take longer.
[1]	SLEEPONEXIT	When set to 1, the core can enter a sleep state on an exception return to Thread mode. This is the mode and exception level entered at reset, the base level of execution.

System Handler Priority Register 2 (SHPR2)

Register	Offset	R/W	Description	Reset Value
SHPR2	SCS_BA + 0xD1C	R/W	System Handler Priority Register 2	0x00000000

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 & "3" denotes lowest priority

System Handler Priority Register 3 (SHPR3)

Register	Offset	R/W	Description	Reset Value
SHPR3	SCS_BA + 0xD20	R/W	System Handler Priority Register 3	0x00000000

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 & "3" denotes lowest priority
[23:22]	PRI_14	**Priority of system handler 14 – PendSV** "0" denotes the highest priority & "3" denotes lowest priority

6.3 Clock Controller

6.3.1 Overview

The clock controller generates the clocks for the whole chip, including system clocks and all peripheral clocks. The clock controller also implements the power control function with the individually clock ON/OFF control, clock source selection and a 4-bit 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 enter 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 crystal and internal 22.1184 MHz oscillator to reduce the overall system power consumption.

6.3.2 Clock Generator Block Diagram

The clock generator consists of 4 sources which list below:

- One external 4~24 MHz crystal
- One internal 22.1184 MHz RC oscillator
- One programmable PLL FOUT(PLL source consists of external 4~24 MHz crystal and internal 22.1184M)
- One internal 10 kHz oscillator

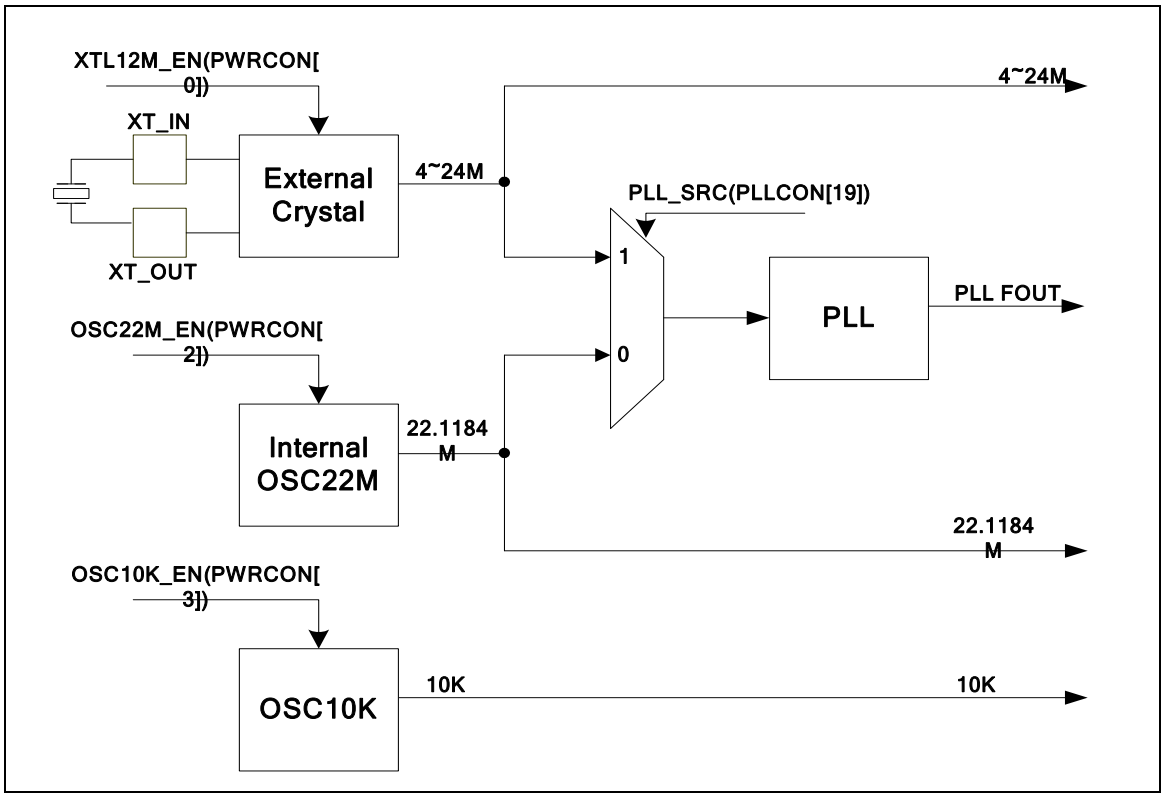


Figure 6.3-1 Clock generator block diagram

6.3.3 System Clock & SysTick Clock

The system clock has 4 clock sources which were generated from clock generator block. The clock source switch depends on the register HCLK_S(CLKSEL0[2:0]). The block diagram is shown in the Figure 6.3-2.

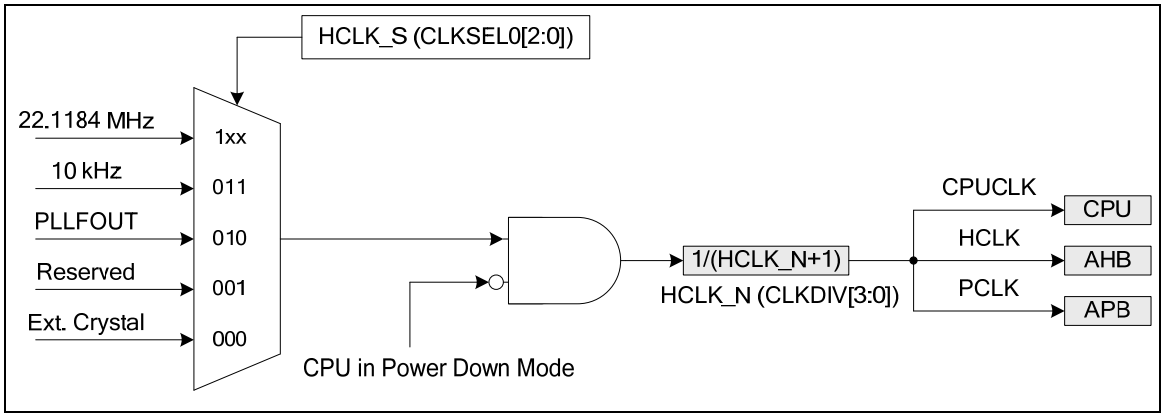


Figure 6.3-2 System Clock Block Diagram

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 4 clock sources. The clock source switch depends on the setting of the register STCLK_S (CLKSEL0[5:3]. The block diagram is shown in the Figure 6.3-3.

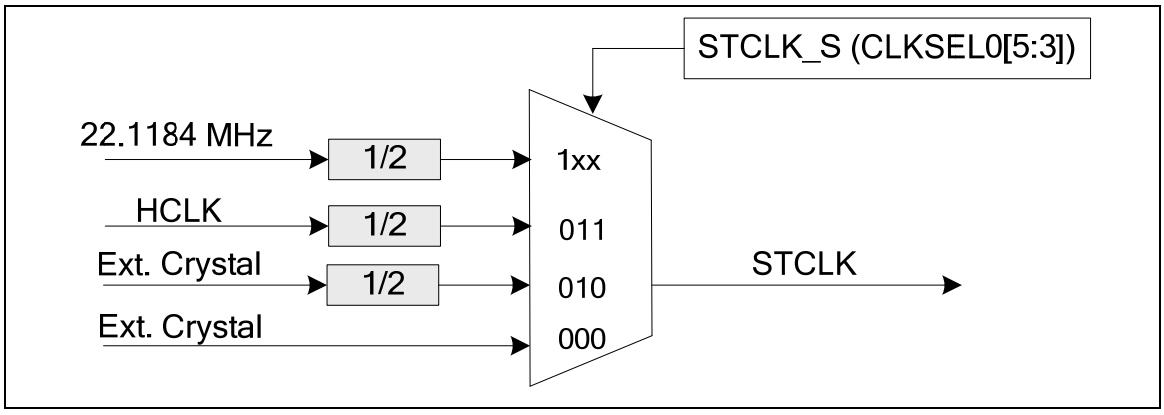


Figure 6.3-3 SysTick clock Control Block Diagram

6.3.4 AHB Clock Source Select

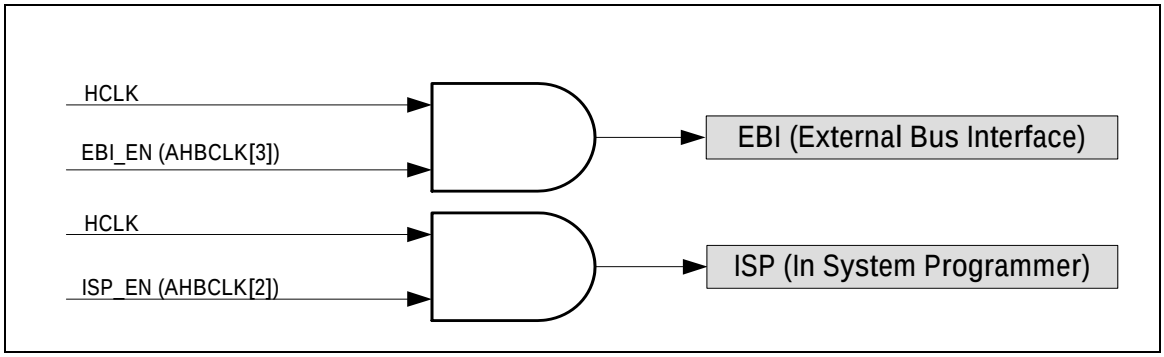


Figure 6.3-4 AHB Clock Source for HCLK

6.3.5 Peripherals Clock Source Select

The peripherals clock had different clock source switch setting which depends on the different peripheral. Please refer the CLKSEL1 & APBCLK register description in chapter 6.3.9.

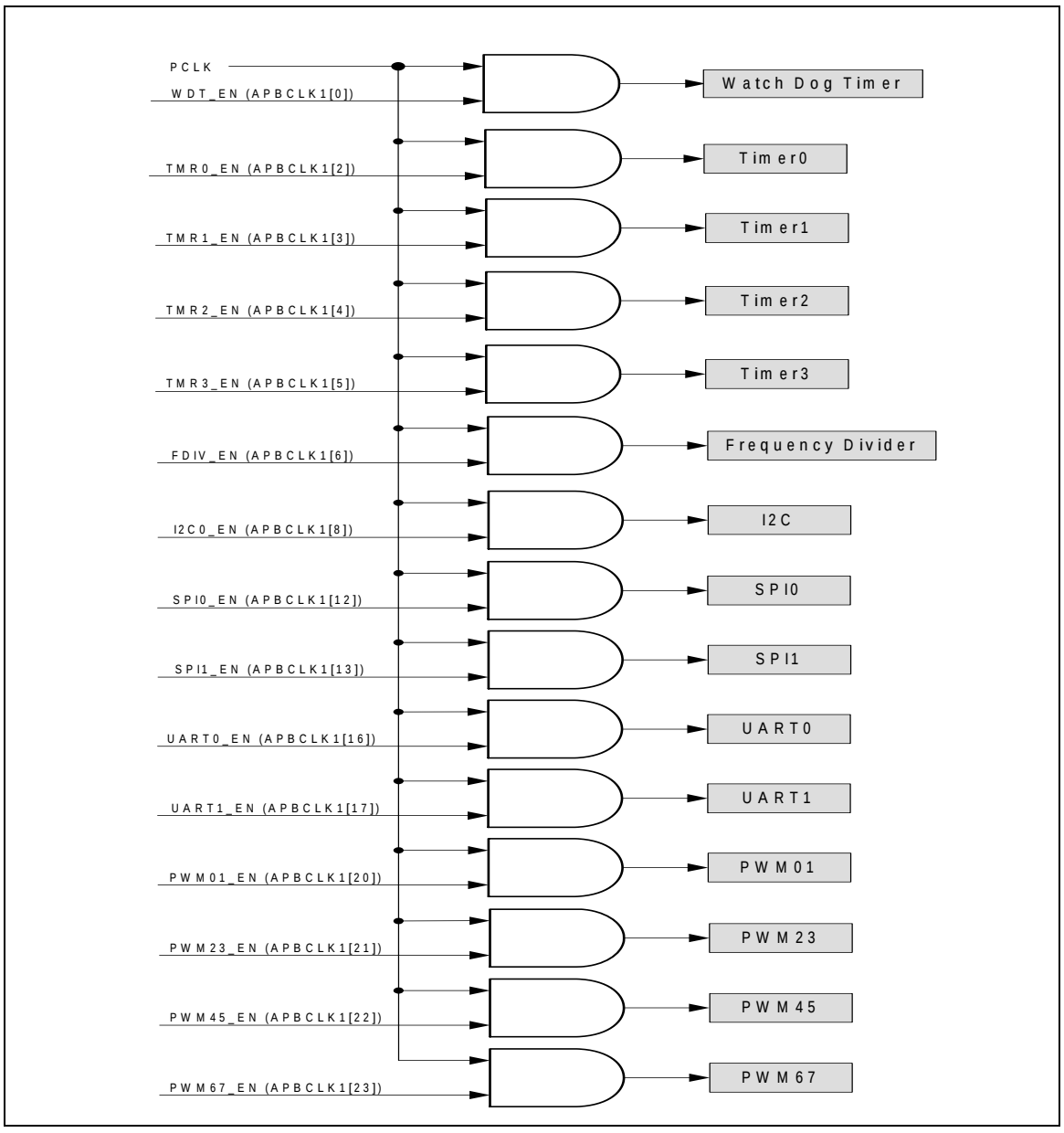


Figure 6.3-5 Peripherals Clock Source Select for PCLK

6.3.6 Power Down Mode (Deep Sleep Mode) Clock

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

For theses clocks which still keep active list below:

Clock Generator

- Internal 10 kHz oscillator clock

Peripherals Clock (When these IP adopt 10 kHz as clock source)

- Watch Dog Clock
- Timer 0/1/2/3 Clock
- PWM Clock

6.3.7 Frequency Divider Output

This device is equipped a power-of-2 frequency divider which is composed by16 chained divide-by-2 shift registers. One of the 16 shift register outputs selected by a sixteen to one multiplexer is reflected to P3.6. Therefore there are 16 options of power-of-2 divided clocks with the frequency from $F_{in}/2^1$ to $F_{in}/2^{17}$ where Fin is input clock frequency to the clock divider.

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

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

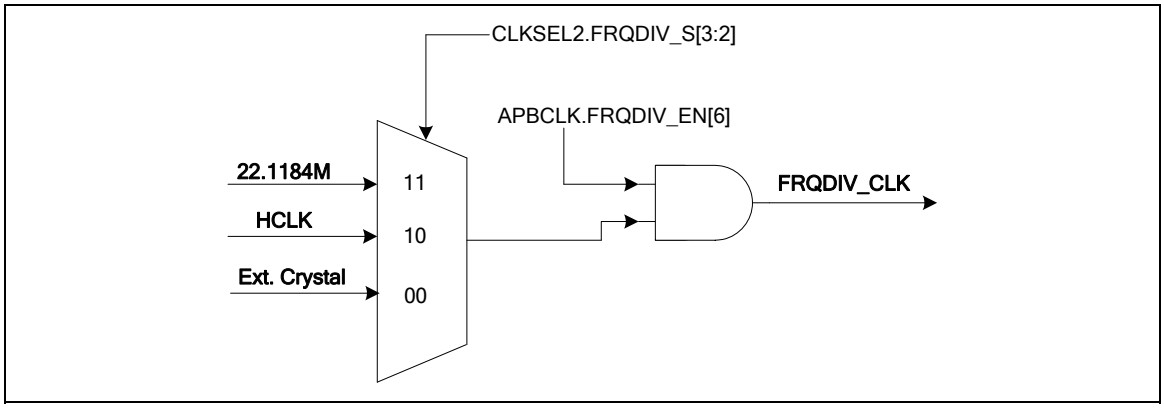


Figure 6.3-6 Clock Source of Frequency Divider

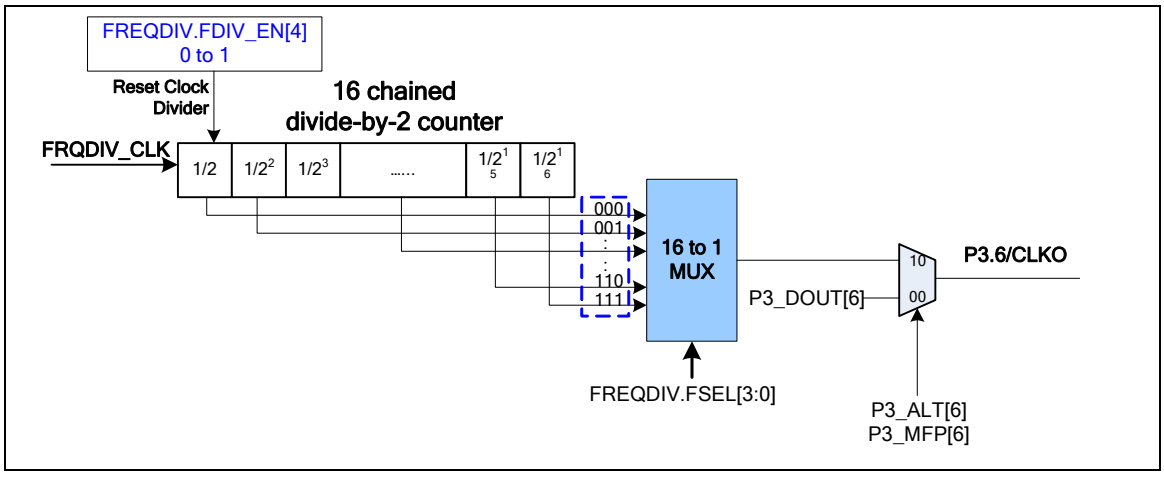


Figure 6.3-7 Block Diagram of Frequency Divider

6.3.8 Clock Controller Registers Map

R: read only, W: write only

Register	Offset	R/W	Description	Reset Value
PWRCON	CLK_BA + 00	R/W	System Power Down Control Register	0x0000_000X
AHBCLK	CLK_BA + 04	R/W	AHB Devices Clock Enable Control Register	0x0000_0001
APBCLK	CLK_BA + 08	R/W	APB Devices Clock Enable Control Register	0x0000_000x
CLKSTATUS	CLK_BA + 0C	R/W	Clock status monitor Register	0x0000_00XX
CLKSEL0	CLK_BA + 10	R/W	Clock Source Select Control Register 0	0xFFFF_FFFX
CLKSEL1	CLK_BA + 14	R/W	Clock Source Select Control Register 1	0xFFFF_FFFX
CLKSEL2	CLK_BA + 1C	R/W	Clock Source Select Control Register 2	0xFFFF_FFFX
CLKDIV	CLK_BA_+ 18	R/W	Clock Divider Number Register	0x0000_0000
PLLCON	CLK_BA + 20	R/W	PLL Control Register	0x0005_C22E
FRQDIV	CLK_BA + 24	R/W	Frequency Divider Control Register	0x0000_0000

6.3.9 Clock Controller Registers Description

Power Down Control Register （PWRCON）

All bits except bit6 (PD_WU_STS) are protected. Programming these protected 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
PWRCON	CLK_BA+00	R/W	System Power Down Control Register	0x0000_000X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							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	Reserved	XTL12M_EN

Bits	Descriptions	
[31:9]	Reserved	Reserve
[8]	PD_WAIT_CPU	This bit control the power down entry condition 1 = Chip enter power down mode when the both PWR_DOWN_EN bit is 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 bit** When CPU sets this bit “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 wakes 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 crystal (4~ 24MHz) and the 22.1184 MHz OSC will be disabled in this mode, but the 10 kHz 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 10 kHz oscillator. 1 = Chip enter the power down mode instant or wait CPU sleep command WFI 0 = Chip operate in normal mode or CPU in idle mode (sleep mode) because of WFI command
[6]	**PD_WU_STS**	**Chip power down wake up status flag** Set by "power down wake up", it indicates that resume from power down mode" The flag is set if the GPIO(P0~P4), and UART wakeup Write 1 to clear this bit to zero.
[5]	**PD_WU_INT_EN**	**Power down mode wake Up Interrupt Enable** 0 = Disable 1 = Enable. The interrupt will occur when Power down mode(Deep Sleep Mode) wakeup.
[4]	**PD_WU_DLY**	**Enable the wake up delay counter.** When the chip wakes up from power down mode, the clock control will delay certain clock cycles to wait system clock stable. The delayed clock cycle is 4096 clock cycles when chip work at external crystal (4 ~ 24MHz), and 256 clock cycles when chip work at 22.1184 MHz oscillator. 1 = Enable clock cycles delay 0 = Disable clock cycles delay
[3]	**OSC10K_EN**	Internal 10 kHz Oscillator Control 1 = 10 kHz Oscillation enable 0 = 10 kHz Oscillation disable
[2]	**OSC22M_EN**	Internal 22.1184 MHz Oscillator Control 1 = 22.1184 MHz Oscillation enable 0 = 22.1184 MHz Oscillation disable
[1]	**Reserved**	Reserve
[0]	**XTL12M_EN**	External Crystal Oscillator Control

		The bit default value is set by flash controller user configuration register config0 [26:24]. When the default clock source is from external crystal, the bit is automatically set to “1” 1 = Crystal oscillation enable 0 = Crystal oscillation disable

Register / Instruction Mode	PWR_DOWN_EN	PD_WAIT_CPU	CPU run WFE/WFI instruction	Clock Gating
Normal Running Mode	0	0	NO	All Clock Gating by control register
IDLE Mode **(CPU entry Sleep Mode)**	0	0	YES	Only CPU clock gating
Power_down Mode	1	0	NO	Most Clock are gating except 10 kHz and some WDT/Timer/PWM/ADC peripheral clock are still active.
Power_down Mode **(CPU entry deep sleep mode)**	1	1	YES	Most Clock are gating except 10 kHz and some WDT/Timer/PWM/ADC peripheral clock are still active.

Table 6.3-1 Power Down Mode Control Table

AHB Devices Clock Enable Control Register （AHBCLK）

These bits for AHBCLK register are used to enable/disable system and AHB engine clock.

Register	Offset	R/W	Description	Reset Value
AHBCLK	CLK_BA + 04	R/W	AHB Devices Clock Enable Control Register	0x0000_0001

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

Bits	Descriptions	
[31:4]	**Reserved**	Reserved
[3]	**EBI_EN**	**EBI Controller Clock Enable Control.** 1 = Enable the EBI controller clock. 0 = Disable the EBI controller clock.
[2]	**ISP_EN**	**Flash ISP Controller Clock Enable Control.** 1 = To enable the Flash ISP controller clock. 0 = To disable the Flash ISP controller clock.
[1:0]	**Reserved**	Reserve

APB Devices Clock Enable Control Register （APBCLK）

These bits of APBCLK register are used to enable/disable clock for APB engine and peripherals.

Register	Offset	R/W	Description	Reset Value
APBCLK	CLK_BA+ 08	R/W	APB Devices Clock Enable Control Register	0x0000_000X

31	30	29	28	27	26	25	24
Reserved			ADC_EN	Reserved			
23	22	21	20	19	18	17	16
PWM67_EN	PWM45_EN	PWM23_EN	PWM01_EN	Reserved		UART1_EN	UART0_EN
15	14	13	12	11	10	9	8
Reserved		SPI1_EN	SPI0_EN	Reserved			I2C_EN
7	6	5	4	3	2	1	0
Reserved	FDIV_EN	TMR3_EN	TMR2_EN	TMR1_EN	TMR0_EN	Reserved	WDT_EN

Bits	Descriptions	
[31:29]	**Reserved**	Reserved
[28]	**ADC_EN**	**Analog-Digital-Converter (ADC) Clock Enable** 1 = Enable ADC clock 0 = Disable ADC clock
[27:24]	**Reserved**	Reserved
[23]	**PWM67_EN**	**PWM_67 Clock Enable** 1 = Enable PWM67 clock 0 = Disable PWM67 clock
[22]	**PWM45_EN**	**PWM_45 Clock Enable** 1 = Enable PWM45 clock

		0 = Disable PWM45 clock
[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
[11:9]	**Reserved**	Reserved
[8]	**I2C_EN**	**I2C Clock Enable** 1 = Enable I2C Clock 0 = Disable I2C Clock
[7]	**Reserved**	Reserved
[6]	**FDIV_EN**	**Clock Divider Clock Enable** 1 = Enable FDIV Clock 0 = Disable FDIV Clock

[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]	**Reserved**	Reserved
[0]	**WDT_EN**	**Watchdog Timer Clock Enable.** 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. The bit default value is set by flash controller. User configuration register congig0 bit[31] 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 if clock switching failed.

Register	Offset	R/W	Description	Reset Value
CLKSTATUS	CLK_BA+0C	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	Reserved	XTL12M_STB

Bits	Descriptions	
[31:8]	Reserved	-
[7]	CLK_SW_FAIL	**Clock switch fail flag** 1 = Clock switch if fail 0 = Clock switch if success This bit will be set when target switch clock source is not stable. Write 1 to clear this bit to zero.
[6:5]	Reserved	-
[4]	OSC22M_STB	**OSC22M (Internal 22.1184 MHz) clock source stable flag (Read Only)** 1 = OSC22M clock is stable 0 = OSC22M clock is not stable or disable

[3]	**OSC10K_STB**	**OSC10K clock source stable flag (Read Only)** 1 = OSC10K clock is stable 0 = OSC10K clock is not stable or disable
[2]	**PLL_STB**	**PLL clock source stable flag (Read Only)** 1 = PLL clock is stable 0 = PLL clock is not stable or disable
[1]	**Reserved**	-
[0]	**XTL12M_STB**	**External Crystal clock source stable flag (Read Only)** 1 = External Crystal clock is stable 0 = External Crystal clock is not stable or disable

Clock Source Select Control Register 0 （CLKSEL0）

Register	Offset	R/W	Description	Reset Value
CLKSEL0[1]	CLK_BA + 10	R/W	Clock Source Select Control Register 0	0xFFFF_FFFX[2]

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**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**	**MCU Cortex_M0 SysTick clock source select.** 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. 000 = Clock source from external crystal clock (4 ~ 24MHz) 001 = Reserved 010 = Clock source from external crystal clock/2 (4 ~ 24MHz) 011 = Clock source from HCLK/2 1xx = Clock source from internal 22.1184 MHz oscillator clock/2
[2:0]	**HCLK_S**	**HCLK clock source select.** 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 crystal clock (4 ~ 24MHz) 001 = Reserved 010 = Clock source from PLL clock 011 = Clock source from internal 10 kHz oscillator clock 1xx = Clock source from internal 22.1184 MHz oscillator clock Others = reserved

Clock Source Select Control Register 1（CLKSEL1）

Before clock switching the related clock sources (current and new) must be turned on.

Register	Offset	R/W	Description	Reset Value
CLKSEL1	CLK_BA + 14	R/W	Clock Source Select Control Register 1	0xFFFF_FFFX

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				ADC_S		WDT_S	

Bits	Descriptions	
[31:30]	PWM23_S	**PWM2 and PWM3 clock source select.** PWM2 and PWM3 uses the same Engine clock source, both of them use the same pre-scalar 00 = Clock source from external external crystal clock (4 ~ 24MHz) 01 = Reserved 10 = Clock source from HCLK 11 = Clock source from internal 22.1184 MHz oscillator clock
[29:28]	PWM01_S	**PWM0 and PWM1 clock source select.** PWM0 and PWM1 uses the same Engine clock source, both of them use the same pre-scalar 00 = Clock source from external external crystal clock (4 ~ 24MHz) 01 = Reserved

		10 = Clock source from HCLK 11 = Clock source from internal 22.1184 MHz oscillator clock
[27:26]	**Reserved**	Reserved
[25:24]	**UART_S**	**UART clock source select.** 00 = Clock source from external crystal clock (4 ~ 24MHz) 01 = Clock source from PLL clock 1x = Clock source from internal 22.1184 MHz oscillator clock
[23]	**Reserved**	Reserved
[22:20]	**TMR3_S**	**TIMER3 clock source select.** 000 = Clock source from external crystal clock (4 ~ 24MHz) 001 = Reserved 010 = Clock source from HCLK 011 = Clock source from external trigger 1xx = Clock source from internal 22.1184 MHz oscillator clock
[19]	**Reserved**	Reserved
[18:16]	**TMR2_S**	**TIMER2 clock source select.** 000 = Clock source from external crystal clock (4 ~ 24MHz) 001 = Reserved 010 = Clock source from HCLK 011 = Clock source from external trigger 1xx = Clock source from internal 22.1184 MHz oscillator clock
[15]	**Reserved**	Reserved
[14:12]	**TMR1_S**	**TIMER1 clock source select.** 000 = Clock source from external crystal clock (4 ~ 24MHz) 001 = Reserved 010 = Clock source from HCLK 011 = Clock source from external trigger 1xx = Clock source from internal 22.1184 MHz oscillator clock
[11]	**Reserved**	Reserved

[10:8]	**TMR0_S**	**TIMER0 clock source select.** 000 = Clock source from external crystal clock (4 ~ 24 MHz) 001 = Reserved 010 = Clock source from HCLK 011 = Clock source from external trigger 1xx = Clock source from internal 22.1184 MHz oscillator clock
[7:4]	**Reserved**	Reserved
[3:2]	**ADC_S**	**ADC clock source select.** 00 = Clock source from external crystal clock (4 ~ 24MHz) 01 = Clock source from PLL clock 1x = Clock source from internal 22.1184 MHz oscillator clock
[1:0]	**WDT_S**	**WDT clock source select.** 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. 00 = Clock source from external crystal clock. (4 ~ 24MHz) 01 = Reserved 10 = Clock source from HCLK/2048 clock 11 = Clock source from internal 10 kHz oscillator clock

Clock Source Select Control Register （CLKSEL2）

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

Register	Offset	R/W	Description	Reset Value
CLKSEL2	CLK_BA + 1C	R/W	Clock Source Select Control Register 2	0xFFFF_FFFX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PWM67_S		**PWM45_S**		**FRQDIV_S**		**Reserved**	

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7:6]	**PWM67_S**	**PWM6 and PWM7 clock source select** PWM6 and PWM7 used the same Engine clock source, both of them use the same pre-scalar 00 = Clock source from external crystal clock (4 ~ 24MHz) 01 = Reserved 10 = Clock source from HCLK 11 = Clock source from internal 22.1184 MHz oscillator clock
[5:4]	**PWM45_S**	**PWM4 and PWM5 clock source select** PWM4 and PWM5 used the same Engine clock source, both of them use the same pre-scalar

		00 = Clock source from external crystal clock (4 ~ 24 MHz) 01 = Reserved 10 = Clock source from HCLK 11 = Clock source from internal 22.1184 MHz oscillator clock
[3:2]	**FRQDIV_S**	**Clock Divider Clock Source Select** 00 = Clock source from external crystal clock (4 ~ 24 MHz) 01 = Reserved 10 = Clock source from HCLK 11 = Clock source from internal 22.1184 MHz oscillator clock
[1:0]	**Reserved**	Reserved

Clock Divider Register (CLKDIV)

Register	Offset	R/W	Description	Reset Value
CLKDIV	CLK_BA_+ 18	R/W	Clock Divider Number Register	0x0000_0000

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

Bits	Descriptions	
[31:24]	**Reserved**	Reserved
[23:16]	**ADC_N**	**ADC clock divide number from ADC clock source** The ADC clock frequency = (ADC clock source frequency) / (ADC_N + 1)
[15:12]	**Reserved**	Reserved
[11:8]	**UART_N**	**UART clock divide number from UART clock source** The UART clock frequency = (UART clock source frequency) / (UART_N + 1)
[7:4]	**Reserved**	Reserved
[3:0]	**HCLK_N**	**HCLK clock divide number 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 crystal clock input (4 ~ 24 MHz) or from the internal 22.1184 MHz oscillator. This register is used to control the PLL output frequency and PLL operating mode

Register	Offset	R/W	Description	Reset Value
PLLCON	CLK_BA + 20	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	
[19]	PLL_SRC	PLL Source Clock Select 1 = PLL source clock from 22.1184 MHz oscillator 0 = PLL source clock from externa; crystal (4 ~ 24 MHz)
[18]	OE	PLL OE (FOUT enable) pin Control 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 IDLE bit “1” in PWRCON register, the PLL will enter power down mode too 0 = PLL is in normal mode (default) 1 = PLL is in power-down mode
[15:14]	**OUT_DV**	PLL Output Divider Control Pins (PLL_OD[1:0])
[13:9]	**IN_DV**	PLL Input Divider Control Pins (PLL_R[4:0])
[8:0]	**FB_DV**	PLL Feedback Divider Control Pins (PLL_F[8:0])

PLL Output Clock Frequency Setting

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

Constrain:

1. $3.2MHz < FIN < 150MHz$

2. $800KHz < \frac{FIN}{2 * NR} < 8MHz$

3. $800KHz < \frac{FIN}{2 * NR} < 8MHz, 120MHz < FCO$ is preferred.

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

Default PLL Frequency Setting:

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

Frequency Divider Control Register (FRQDIV)

Register	Offset	R/W	Description	Reset Value
FRQDIV	CLK_BA+ 24	R/W	Frequency Divider Control Register	0x0000_0000

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

Bits	Descriptions	
[31:5]	**Reserved**	Reserved
[4]	**DIVIDER_EN**	**Frequency Divider Enable Bit** 0 = Disable Frequency Divider 1 = Enable Frequency Divider
[3:0]	**FSEL**	**Divider Output Frequency Selection Bits** The formula of output frequency is $F_{out} = F_{in}/2^{(N+1)}$, F_{in} is the input clock frequency F_{out} is the frequency of divider output clock N is the 4-bit value of FSEL[3:0].

6.4 General Purpose I/O

6.4.1 Overview

There are 40 General Purpose I/O pins shared with special feature functions in this MCU. The 40 pins are arranged in 5 ports named with P0, P1, P2, P3 and P4. Each port equips maximum 8 pins. Each one of the 40 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 software configured individually as input, output, open-drain or quasi-bidirectional mode. The all pins of I/O type stay in quasi-bidirectional mode and port data register Px_DOUT[7:0] resets to 0x000_00FF. Each I/O pin equips a very weakly individual pull-up resistor which is about 110KΩ~300KΩ for V_{DD} is from 5.0V to 2.5V.

6.4.1.1 Input Mode Explanation

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

6.4.1.2 Output Mode Explanation

Set Px_PMD(PMDn[1:0]) to 2'b01 the Px[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 Px_DOUT is driven on the pin.

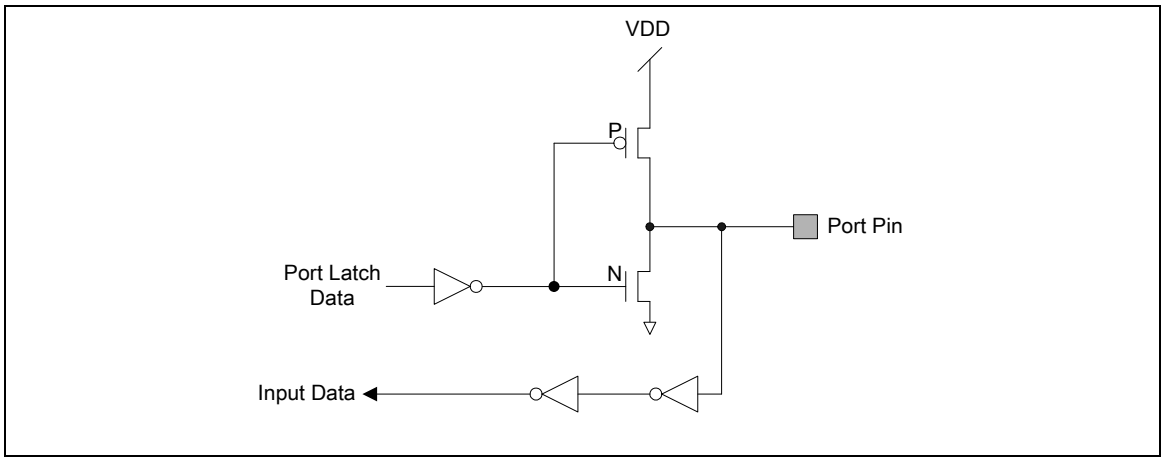


Figure 6.4-1 Push-Pull Output

6.4.1.3 Open-Drain Mode Explanation

Set Px_PMD(PMDn[1:0]) to 2'b10 the Px[n] pin is in Open-Drain mode and the I/O pin supports digital output function but only with sink current capability, an additional pull-up resister is needed for driving high state. If the bit value in the corresponding bit [n] of Px_DOUT is "0", the pin drive a "low" output on the pin. If the bit value in the corresponding bit [n] of Px_DOUT is "1", the pin output drives high that is controlled by the internal pull-up resistor or the external pull high resistor.

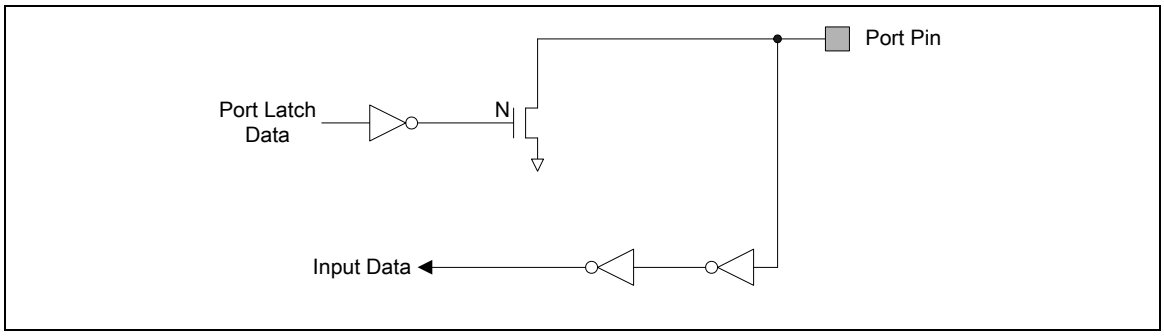


Figure 6.4-2 Open-Drain Output

6.4.1.4 Quasi-bidirectional Mode Explanation

Set Px_PMD(PMDn[1:0]) to 2'b11 the Px[n] pin is in Quasi-bidirectional mode and the I/O pin supports digital output and input function at the same time but the source current is only up to hundreds uA. Before the digital input function is performed the corresponding bit in Px_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 Px_DOUT is "0", the pin drive a "low" output on the pin. If the bit value in the corresponding bit [n] of Px_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 200uA to 30uA for VDD is form 5.0V to 2.5V

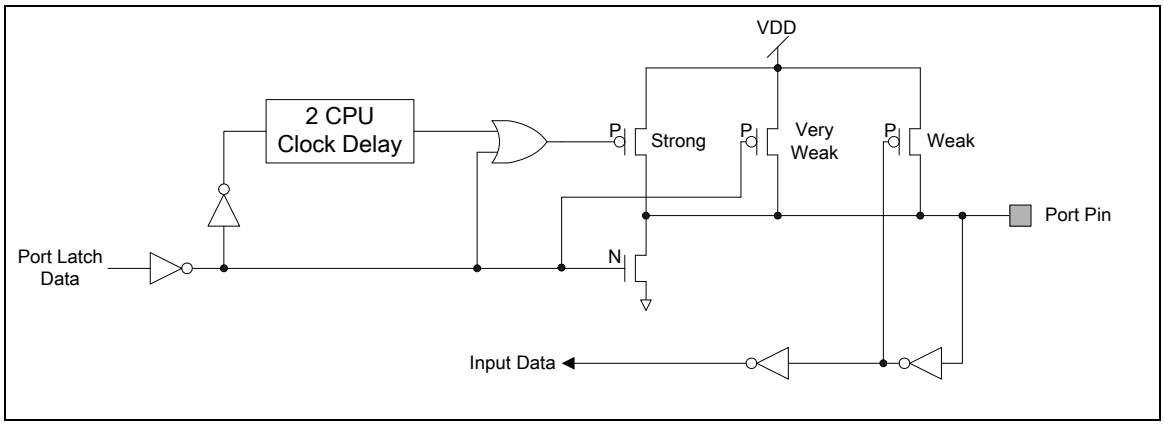


Figure 6.4-3 Quasi-bidirectional I/O Mode

6.4.2 Port 0-4 Controller Registers Map

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

Register	Offset	R/W	Description	Reset Value
GP_BA = 0x5000_4000				
P0_PMD	GP_BA+0x000	R/W	P0 Bit Mode Control	0x0000_FFFF
P0_OFFD	GP_BA+0x004	R/W	P0 Bit OFF Digital Enable	0x0000_0000
P0_DOUT	GP_BA+0x008	R/W	P0 Data Output Value	0x0000_00FF
P0_DMASK	GP_BA+0x00C	R/W	P0 Data Output Write Mask	0x0000_0000
P0_PIN	GP_BA+0x010	R	P0 Pin Value	0x0000_00XX
P0_DBEN	GP_BA+0x014	R/W	P0 De-bounce Enable	0x0000_0000
P0_IMD	GP_BA+0x018	R/W	P0 Interrupt Mode Control	0x0000_0000
P0_IEN	GP_BA+0x01C	R/W	P0 Interrupt Enable	0x0000_0000
P0_ISRC	GP_BA+0x020	R/WC	P0 Interrupt Source Flag	0xXXXX_XXXX
P1_PMD	GP_BA+0x040	R/W	P1 Bit Mode Enable	0x0000_FFFF
P1_OFFD	GP_BA+0x044	R/W	P1 Bit OFF Digital Enable	0x0000_0000
P1_DOUT	GP_BA+0x048	R/W	P1 Data Output Value	0x0000_00FF
P1_DMASK	GP_BA+0x04C	R/W	P1 Data Output Write Mask	0x0000_0000
P1_PIN	GP_BA+0x050	R	P1 Pin Value	0x0000_00XX
P1_DBEN	GP_BA+0x054	R/W	P1 De-bounce Enable	0x0000_0000
P1_IMD	GP_BA+0x058	R/W	P1 Interrupt Mode Control	0x0000_0000
P1_IEN	GP_BA+0x05C	R/W	P1 Interrupt Enable	0x0000_0000
P1_ISRC	GP_BA+0x060	R/WC	P1 Interrupt Source Flag	0xXXXX_XXXX
P2_PMD	GP_BA+0x080	R/W	P2 Bit Mode Enable	0x0000_FFFF
P2_OFFD	GP_BA+0x084	R/W	P2 Bit OFF digital Enable	0x0000_0000

P2_DOUT	GP_BA+0x088	R/W	P2 Data Output Value	0x0000_00FF
P2_DMASK	GP_BA+0x08C	R/W	P2 Data Output Write Mask	0x0000_0000
P2_PIN	GP_BA+0x090	R	P2 Pin Value	0x0000_00XX
P2_DBEN	GP_BA+0x094	R/W	P2 De-bounce Enable	0x0000_0000
P2_IMD	GP_BA+0x098	R/W	P2 Interrupt Mode Control	0x0000_0000
P2_IEN	GP_BA+0x09C	R/W	P2 Interrupt Enable	0x0000_0000
P2_ISRC	GP_BA+0x0A0	R/WC	P2 Interrupt Source Flag	0xXXXX_XXXX
P3_PMD	GP_BA+0x0C0	R/W	P3 Bit Mode Enable	0x0000_FFFF
P3_OFFD	GP_BA+0x0C4	R/W	P3 Bit OFF Digital Enable	0x0000_0000
P3_DOUT	GP_BA+0x0C8	R/W	P3 Data Output Value	0x0000_00FF
P3_DMASK	GP_BA+0x0CC	R/W	P3 Data Output Write Mask	0x0000_0000
P3_PIN	GP_BA+0x0D0	R	P3 Pin Value	0x0000_00XX
P3_DBEN	GP_BA+0x0D4	R/W	P3 De-bounce Enable	0x0000_0000
P3_IMD	GP_BA+0x0D8	R/W	P3 Interrupt Mode Control	0x0000_0000
P3_IEN	GP_BA+0x0DC	R/W	P3 Interrupt Enable	0x0000_0000
P3_ISRC	GP_BA+0x0E0	R/WC	P3 Interrupt Source Flag	0xXXXX_XXXX
P4_PMD	GP_BA+0x100	R/W	P4 Bit Mode Enable	0x0000_FFFF
P4_OFFD	GP_BA+0x104	R/W	P4 Bit OFF Digital Enable	0x0000_0000
P4_DOUT	GP_BA+0x108	R/W	P4 Data Output Value	0x0000_00FF
P4_DMASK	GP_BA+0x10C	R/W	P4 Data Output Write Mask	0x0000_0000
P4_PIN	GP_BA+0x110	R	P4 Pin Value	0x0000_00XX
P4_DBEN	GP_BA+0x114	R/W	P4 De-bounce Enable	0x0000_0000
P4_IMD	GP_BA+0x118	R/W	P4 Interrupt Mode Control	0x0000_0000
P4_IEN	GP_BA+0x11C	R/W	P4 Interrupt Enable	0x0000_0000
P4_ISRC	GP_BA+0x120	R/WC	P4 Interrupt Source Flag	0xXXXX_XXXX

DBNCECON	GP_BA+0x180	R/W	De-bounce Cycle Control	0x0000_0020
P00_DOUT	GP_BA+0x200	R/W	P0.0 Data Output Value	0x0000_0001
P01_DOUT	GP_BA+0x204	R/W	P0.1 Data Output Value	0x0000_0001
P02_DOUT	GP_BA+0x208	R/W	P0.2 Data Output Value	0x0000_0001
P03_DOUT	GP_BA+0x20C	R/W	P0.3 Data Output Value	0x0000_0001
P04_DOUT	GP_BA+0x210	R/W	P0.4 Data Output Value	0x0000_0001
P05_DOUT	GP_BA+0x214	R/W	P0.5 Data Output Value	0x0000_0001
P06_DOUT	GP_BA+0x218	R/W	P0.6 Data Output Value	0x0000_0001
P07_DOUT	GP_BA+0x21C	R/W	P0.7 Data Output Value	0x0000_0001
P10_DOUT	GP_BA+0x220	R/W	P1.0 Data Output Value	0x0000_0001
P11_DOUT	GP_BA+0x224	R/W	P1.1 Data Output Value	0x0000_0001
P12_DOUT	GP_BA+0x228	R/W	P1.2 Data Output Value	0x0000_0001
P13_DOUT	GP_BA+0x22C	R/W	P1.3 Data Output Value	0x0000_0001
P14_DOUT	GP_BA+0x230	R/W	P1.4 Data Output Value	0x0000_0001
P15_DOUT	GP_BA+0x234	R/W	P1.5 Data Output Value	0x0000_0001
P16_DOUT	GP_BA+0x238	R/W	P1.6 Data Output Value	0x0000_0001
P17_DOUT	GP_BA+0x23C	R/W	P1.7 Data Output Value	0x0000_0001
P20_DOUT	GP_BA+0x240	R/W	P2.0 Data Output Value	0x0000_0001
P21_DOUT	GP_BA+0x244	R/W	P2.1 Data Output Value	0x0000_0001
P22_DOUT	GP_BA+0x248	R/W	P2.2 Data Output Value	0x0000_0001
P23_DOUT	GP_BA+0x24C	R/W	P2.3 Data Output Value	0x0000_0001
P24_DOUT	GP_BA+0x250	R/W	P2.4 Data Output Value	0x0000_0001
P25_DOUT	GP_BA+0x254	R/W	P2.5 Data Output Value	0x0000_0001
P26_DOUT	GP_BA+0x258	R/W	P2.6 Data Output Value	0x0000_0001

P27_DOUT	GP_BA+0x25C	R/W	P2.7 Data Output Value	0x0000_0001
P30_DOUT	GP_BA+0x260	R/W	P3.0 Data Output Value	0x0000_0001
P31_DOUT	GP_BA+0x264	R/W	P3.1 Data Output Value	0x0000_0001
P32_DOUT	GP_BA+0x268	R/W	P3.2 Data Output Value	0x0000_0001
P33_DOUT	GP_BA+0x26C	R/W	P3.3 Data Output Value	0x0000_0001
P34_DOUT	GP_BA+0x270	R/W	P3.4 Data Output Value	0x0000_0001
P35_DOUT	GP_BA+0x274	R/W	P3.5 Data Output Value	0x0000_0001
P36_DOUT	GP_BA+0x278	R/W	P3.6 Data Output Value	0x0000_0001
P37_DOUT	GP_BA+0x27C	R/W	P3.7 Data Output Value	0x0000_0001
P40_DOUT	GP_BA+0x280	R/W	P4.0 Data Output Value	0x0000_0001
P41_DOUT	GP_BA+0x284	R/W	P4.1 Data Output Value	0x0000_0001
P42_DOUT	GP_BA+0x288	R/W	P4.2 Data Output Value	0x0000_0001
P43_DOUT	GP_BA+0x28C	R/W	P4.3 Data Output Value	0x0000_0001
P44_DOUT	GP_BA+0x290	R/W	P4.4 Data Output Value	0x0000_0001
P45_DOUT	GP_BA+0x294	R/W	P4.5 Data Output Value	0x0000_0001
P46_DOUT	GP_BA+0x298	R/W	P4.6 Data Output Value	0x0000_0001
P47_DOUT	GP_BA+0x29C	R/W	P4.7 Data Output Value	0x0000_0001

6.4.3 Port 0-4 Controller Registers Description

Port 0-4 I/O Mode Control (Px_PMD)

Register	Offset	R/W	Description	Reset Value
P0_PMD	GP_BA+0x000	R/W	P0 Pin I/O Mode Control	0x0000_FFFF
P1_PMD	GP_BA+0x040	R/W	P1 Pin I/O Mode Control	0x0000_FFFF
P2_PMD	GP_BA+0x080	R/W	P2 Pin I/O Mode Control	0x0000_FFFF
P3_PMD	GP_BA+0x0C0	R/W	P3 Pin I/O Mode Control	0x0000_FFFF
P4_PMD	GP_BA+0x100	R/W	P4 Pin I/O Mode Control	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**
PMD7		**PMD6**		**PMD5**		**PMD4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PMD3		**PMD2**		**PMD1**		**PMD0**	

Bits	Descriptions	
[31:16]	**Reserved**	Reserved
[2n+1 :2n]	**PMDn**	**Px I/O Pin[n] Mode Control** Determine each I/O type of Px pins 00 = Px [n] pin is in INPUT mode. 01 = Px [n] pin is in OUTPUT mode. 10 = Px [n] pin is in Open-Drain mode. 11 = Px [n] pin is in Quasi-bidirectional mode.

		x=0~4, n = 0~7

Port 0-4 Bit OFF Digital Resistor Enable (Px_OFFD)

Register	Offset	R/W	Description	Reset Value
P0_OFFD	GP_BA+0x004	R/W	P0 Pin OFF Digital Enable	0x0000_0000
P1_OFFD	GP_BA+0x044	R/W	P1 Pin OFF Digital Enable	0x0000_0000
P2_OFFD	GP_BA+0x084	R/W	P2 Pin OFF Digital Enable	0x0000_0000
P3_OFFD	GP_BA+0x0C4	R/W	P3 Pin OFF Digital Enable	0x0000_0000
P4_OFFD	GP_BA+0x104	R/W	P4 Pin OFF Digital Enable	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
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	
[31:24]	**Reserved**	Reserved
[23:16]	**OFFD**	**OFFD: Px Pin[n] OFF digital input path Enable** 1 = Disable IO digital input path (digital input tied to low) 0 = Enable IO digital input path x=0~4, n = 0~7
[15:0]	**Reserved**	Reserved

Port 0-4 Data Output Value (Px_DOUT)

Register	Offset	R/W	Description	Reset Value
P0_DOUT	GP_BA+0x008	R/W	P0 Data Output Value	0x0000_00FF
P1_DOUT	GP_BA+0x048	R/W	P1 Data Output Value	0x0000_00FF
P2_DOUT	GP_BA+0x088	R/W	P2 Data Output Value	0x0000_00FF
P3_DOUT	GP_BA+0x0C8	R/W	P3 Data Output Value	0x0000_00FF
P4_DOUT	GP_BA+0x108	R/W	P4 Data Output Value	0x0000_00FF

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

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[n]	**DOUT[n]**	**Px Pin[n] Output Value** Each of these bits control the status of a Px pin when the Px pin is configures as output, open-drain and quasi-mode. 1 = Px Pin[n] will drive High if the corresponding output mode enabling bit is set. 0 = Px Pin[n] will drive Low if the corresponding output mode enabling bit is set. x=0~4, n = 0~7

Port0-4 Data Output Write Mask (Px_DMASK)

Register	Offset	R/W	Description	Reset Value
P0_DMASK	GP_BA+0x00C	R/W	P0 Data Output Write Mask	0xXXXX_XX00
P1_DMASK	GP_BA+0x04C	R/W	P1 Data Output Write Mask	0xXXXX_XX00
P2_DMASK	GP_BA+0x08C	R/W	P2 Data Output Write Mask	0xXXXX_XX00
P3_DMASK	GP_BA+0x0CC	R/W	P3 Data Output Write Mask	0xXXXX_XX00
P4_DMASK	GP_BA+0x10C	R/W	P4 Data Output Write Mask	0xXXXX_XX00

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

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[n]	**DMASK[n]**	**Px Data Output Write Mask** These bits are used to protect the corresponding register of Px_DOUT bit[n] . When set the DMASK bit[n] to "1", the corresponding Px_DOUTn bit is protected. The write signal is masked, write data to the protect bit is ignored 0 = The corresponding Px_DOUT[n] bit is not masked 1 = The corresponding Px_DOUT[n] bit is masked x=0~4, n = 0~7

Port 0-4 Pin Value (Px_PIN)

Register	Offset	R/W	Description	Reset Value
P0_PIN	GP_BA+0x010	R	P0 Pin Value	0x0000_00XX
P1_PIN	GP_BA+0x050	R	P1 Pin Value	0x0000_00XX
P2_PIN	GP_BA+0x090	R	P2 Pin Value	0x0000_00XX
P3_PIN	GP_BA+0x0D0	R	P3 Pin Value	0x0000_00XX
P4_PIN	GP_BA+0x110	R	P4 Pin Value	0x0000_00XX

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

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[n]	**PIN[n]**	**Px Pin Values** The value read from each of these bit reflects the actual status of the respective Px pin x=0~4, n = 0~7

Port 0-4 De-bounce Enable (Px_DBEN)

Register	Offset	R/W	Description	Reset Value
P0_DBEN	GP_BA+0x014	R/W	P0 De-bounce Enable	0xXXXX_XX00
P1_DBEN	GP_BA+0x054	R/W	P1 De-bounce Enable	0xXXXX_XX00
P2_DBEN	GP_BA+0x094	R/W	P2 De-bounce Enable	0xXXXX_XX00
P3_DBEN	GP_BA+0x0D4	R/W	P3 De-bounce Enable	0xXXXX_XX00
P4_DBEN	GP_BA+0x114	R/W	P4 De-bounce Enable	0xXXXX_XX00

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

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[n]	**DBEN[n]**	**Px 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 DBEN[n] is used for "edge-trigger" interrupt only, and ignored for "level trigger" interrupt 0 = The bit[n] de-bounce function is disabled 1 = The bit[n] de-bounce function is enabled The de-bounce function is valid for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored.

		x=0~4, n = 0~7

Port 0-4 Interrupt Mode Control (Px_IMD)

Register	Offset	R/W	Description	Reset Value
P0_IMD	GP_BA+0x018	R/W	P0 Interrupt Mode Control	0xXXXX_XX00
P1_IMD	GP_BA+0x058	R/W	P1 Interrupt Mode Control	0xXXXX_XX00
P2_IMD	GP_BA+0x098	R/W	P2 Interrupt Mode Control	0xXXXX_XX00
P3_IMD	GP_BA+0x0D8	R/W	P3 Interrupt Mode Control	0xXXXX_XX00
P4_IMD	GP_BA+0x118	R/W	P4 Interrupt Mode Control	0xXXXX_XX00

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

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[n]	**IMD[n]**	**Port 0-4 Interrupt Mode Control** IMD[n] used to control the interrupt is by level trigger or by edge trigger. If the interrupt is by edge trigger, the trigger source is control de-bounce. If the interrupt is by level trigger, the input source is sampled by one clock and the generate the interrupt 0 = Edge trigger interrupt 1 = Level trigger interrupt If set pin as the level trigger interrupt, then only one level can be set on the registers Px_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.

		x=0~4, n = 0~7

Port 0-4 Interrupt Enable Control (Px_IEN)

Register	Offset	R/W	Description	Reset Value
P0_IEN	GP_BA+0x01C	R/W	P0 Interrupt Enable	0x0000_0000
P1_IEN	GP_BA+0x05C	R/W	P1 Interrupt Enable	0x0000_0000
P2_IEN	GP_BA+0x09C	R/W	P2 Interrupt Enable	0x0000_0000
P3_IEN	GP_BA+0x0DC	R/W	P3 Interrupt Enable	0x0000_0000
P4_IEN	GP_BA+0x11C	R/W	P4 Interrupt Enable	0x0000_0000

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

Bits	Descriptions	
[31:24]	**Reserved**	Reserved
[n+16]	**IR_EN[n]**	**Port 0-4 Interrupt Enable by Input Rising Edge or Input Level High** IR_EN[n] used to enable the interrupt for each of the corresponding input Px[n]. Set bit "1" also enable the pin wakeup function When set the IR_EN[n] bit "1": If the interrupt is level mode trigger, the input Px[n] state at level "high" will generate the interrupt. If the interrupt is edge mode trigger, the input Px[n] state change from "low-to-high" will generate the interrupt. 1 = Enable the Px[n] level-high or low-to-high interrupt 0 = Disable the Px[n] level-high or low-to-high interrupt.

		x=0~4, n = 0~7
[15:8]	**Reserved**	Reserved
[n]	**IF_EN[n]**	**Port 0-4 Interrupt Enable by Input Falling Edge or Input Level Low** IF_EN[n] used to enable the interrupt for each of the corresponding input Px[n]. Set bit "1" also enable the pin wakeup function When set the IF_EB[n] bit "1": If the interrupt is level mode trigger, the input Px[n] state at level "low" will generate the interrupt. If the interrupt is edge mode trigger, the input Px[n] state change from "high-to-low" will generate the interrupt. 1 = Enable the Px[n] state low-level or high-to-low change interrupt 0 = Disable the Px[n] state low-level or high-to-low change interrupt x=0~4, n = 0~7

Port 0-4 Interrupt Trigger Source (Px_ISRC)

Register	Offset	R/W	Description	Reset Value
P0_ISRC	GP_BA+0x020	R/WC	P0 Interrupt Trigger Source Indicator	0x0000_0000
P1_ISRC	GP_BA+0x060	R/WC	P1 Interrupt Trigger Source Indicator	0x0000_0000
P2_ISRC	GP_BA+0x0A0	R/WC	P2 Interrupt Trigger Source Indicator	0x0000_0000
P3_ISRC	GP_BA+0x0E0	R/WC	P3 Interrupt Trigger Source Indicator	0x0000_0000
P4_ISRC	GP_BA+0x120	R/WC	P4 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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IF_ ISRC[7:0]							

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[n]	**ISRC[n]**	**Port 0-4 Interrupt Trigger Source Indicator** Read : 1 = Indicates Px[n] generate an interrupt 0 = No interrupt at Px[n] Write : 1= Clear the correspond pending interrupt 0= No action x=0~4, n = 0~7

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 "0" will disable the interrupt generate circuit clock, if the pin[n] interrupt is disabled 0 = disable the clock if the P0/1/2/3/4[n] interrupt is disabled 1 = interrupt generated circuit clock always enable n=0~7
[4]	DBCLKSRC	**De-bounce counter clock source select** 1 = De-bounce counter clock source is the internal 10kHz 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 [P0/P1/P2/P3/P4] I/O Bit Output Control (Pxx_DOUT)

Register	Offset	R/W	Description	Reset Value
P0x_DOUT	GP_BA+0x200 - GP_BA+0x21C	R/W	P0 Pin I/O Bit Output Control	0x0000_0001
P1x_DOUT	GP_BA+0x220 - GP_BA+0x23C	R/W	P1 Pin I/O Bit Output Control	0x0000_0001
P2x_DOUT	GP_BA+0x240 - GP_BA+0x25C	R/W	P2 Pin I/O Bit Output Control	0x0000_0001
P3x_DOUT	GP_BA+0x260 - GP_BA+0x27C	R/W	P3 Pin I/O Bit Output Control	0x0000_0001
P4x_DOUT	GP_BA+0x280 - GP_BA+0x29C	R/W	P4 Pin I/O Bit Output 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							**Pxx_DOUT**

Bits	Descriptions

[0]	**Pxx_DOUT**	**Pxx I/O Pin Bit Output Control** Set this bit can control one GPIO pin output value 1 = set corresponding GPIO bit to high 0 = set corresponding GPIO bit to low

6.5 I2C Serial Interface Controller (Master/Slave)

6.5.1 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 Figure 6.5-1 for more detail I2C BUS Timing.

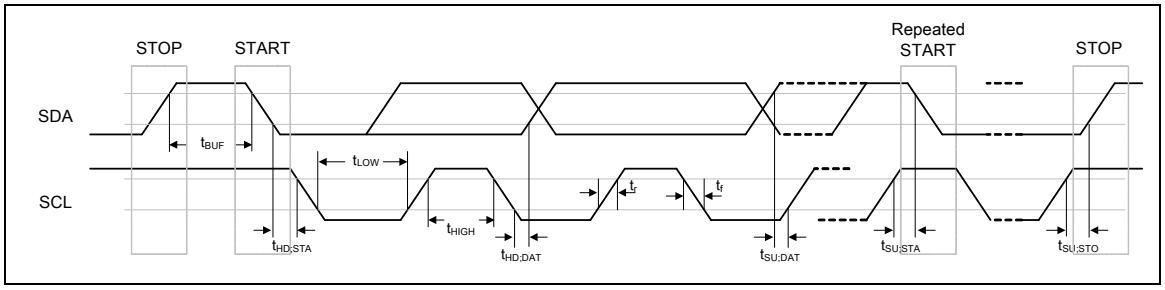


Figure 6.5-1 I2C Bus Timing

The device's on-chip I2C provides the serial interface that meets the I2C bus standard mode specification. The I2C port handles byte transfers autonomously. 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 (Px.y, serial data line) and SCL (Px.y, serial clock line). Pull up resistor is needed for Pin Px.y and Px.y 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.

6.5.2 Features

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

- Support Master and 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)

6.5.3 Function Description

6.5.3.1 I2C Protocol

Normally, a standard communication consists of four parts:

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

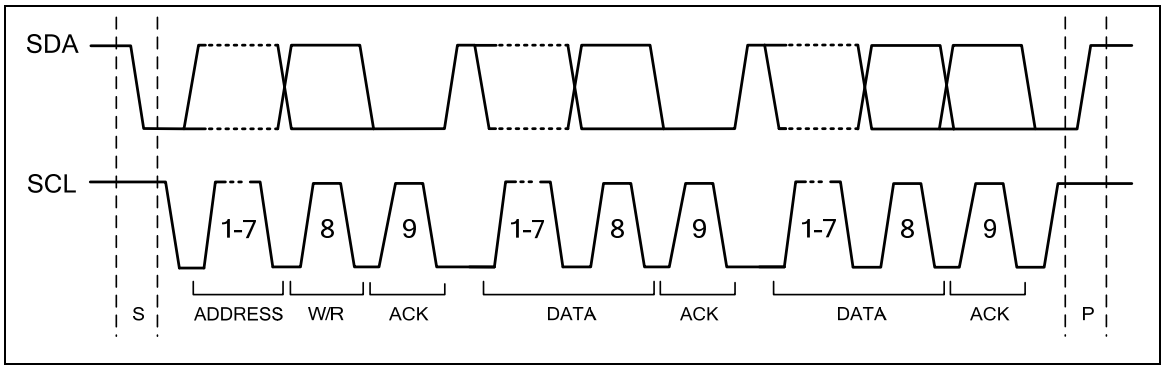


Figure 6.5-2 I2C Protocol

6.5.3.2 Data transfer on the I2C-bus

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

The transfer direction is not changed

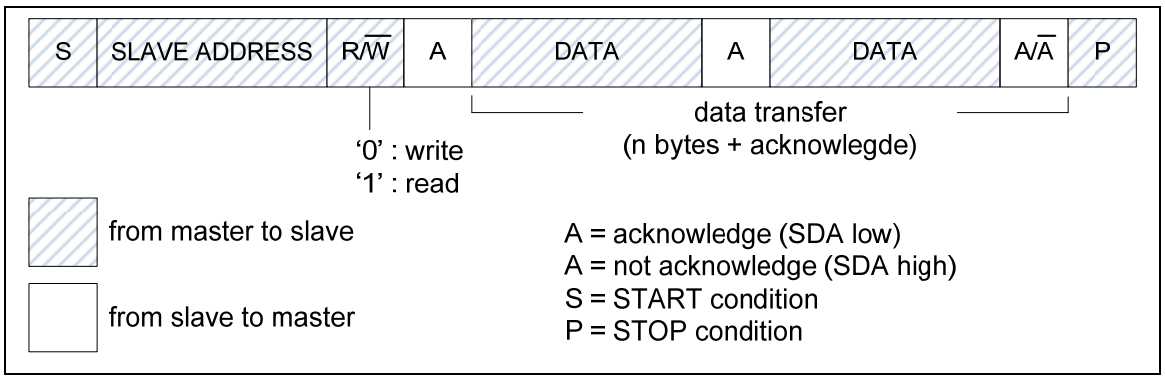


Figure 6.5-3 Master Transmits Data to Slave

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

The transfer direction is changed

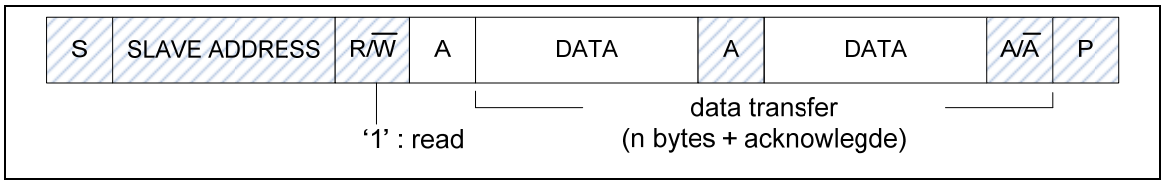


Figure 6.5-4 Master Reads Data from Slave

6.5.3.3 START or Repeated START signal

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

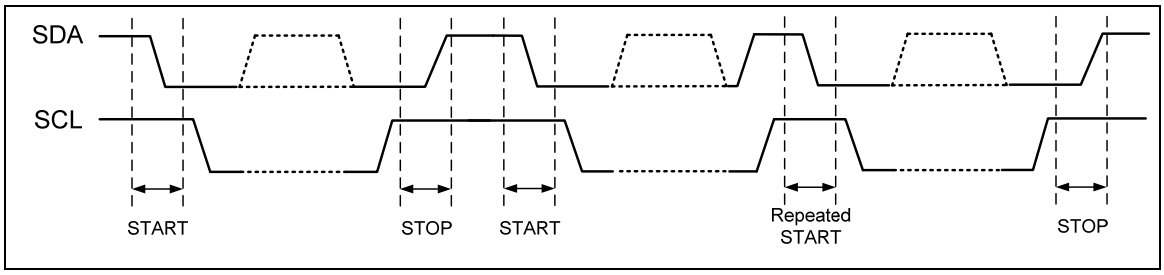


Figure 6.5-5 START and STOP condition

6.5.3.4 Slave Address Transfer

The first byte of data transferred by the master immediately after the START signal is the slave address. This is a 7-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.

6.5.3.5 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.

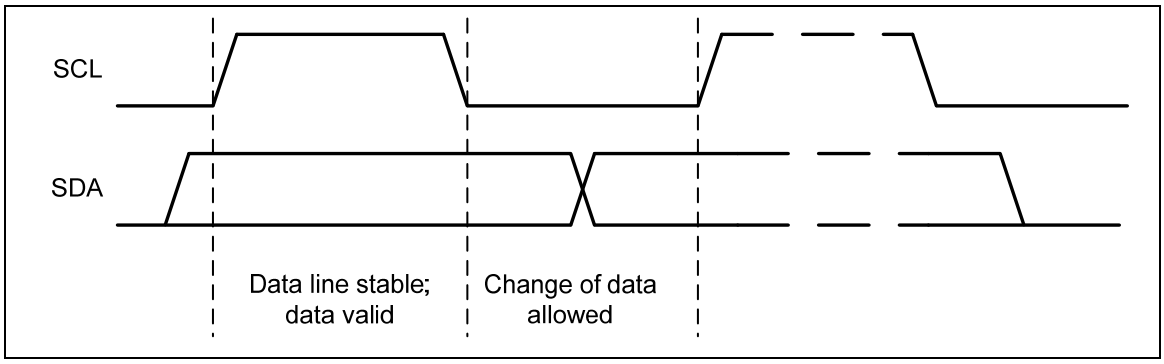


Figure 6.5-6 Bit Transfer on the I2C bus

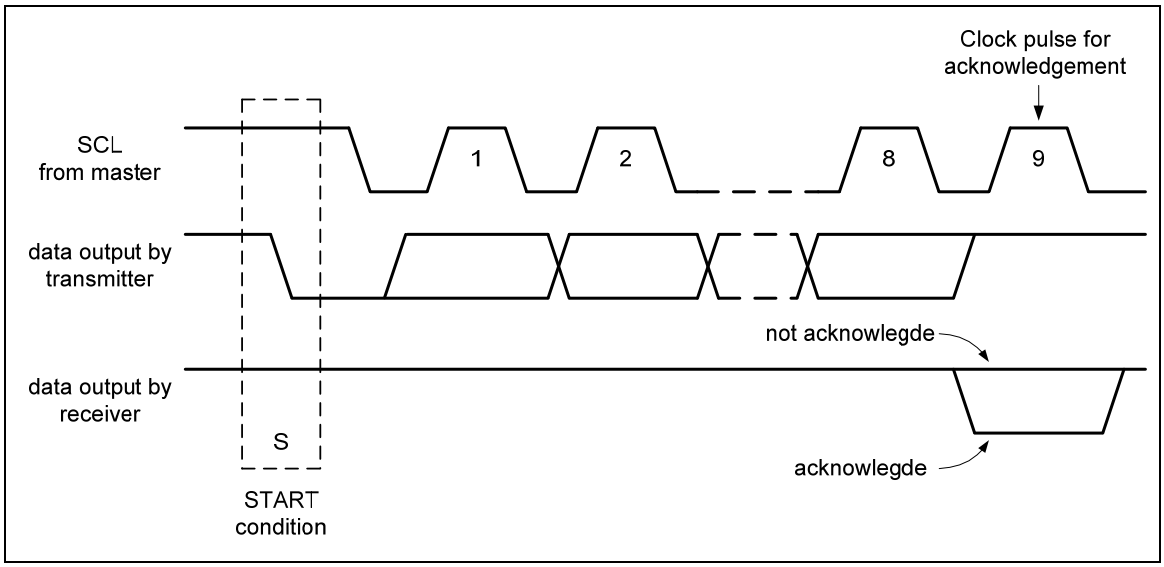


Figure 6.5-7 Acknowledge on the I2C bus

6.5.4 I2C Protocol Registers

The CPU interfaces to the SIO 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 of I2C is 4002_0000.

6.5.4.1 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 MCU's own slave address. The I2C hardware will react if the contents of I2CADDR 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.

6.5.4.2 Data Register (I2CDAT)

This register contains a byte of serial data to be transmitted or a byte which just has been received. The CPU can 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.

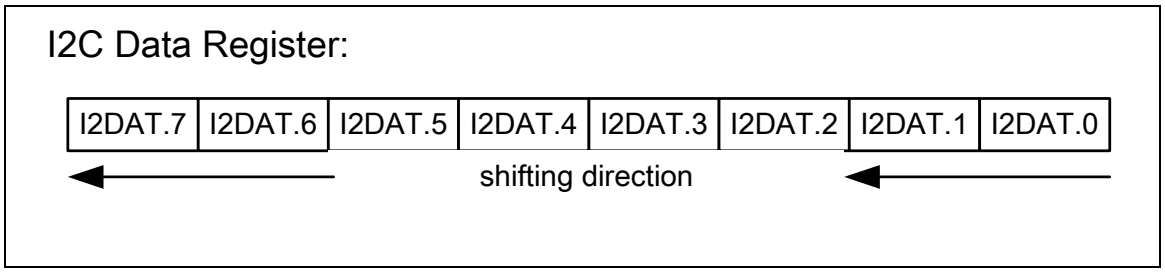


Figure 6.5-8 I2C Data Shifting Direction

6.5.4.3 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.

ENSI Set to enable I2C serial function block. When ENSI=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 SIO 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 one to this bit.

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.

6.5.4.4 Status Register (I2CSTATUS)

I2CSTATUS [7:0] is an 8-bit read-only register. The three least significant bits are always 0. The bit field I2CSTATUS [7:3] contain the status code. There are 26 possible status codes, All states are listed in section 6.5.6. When I2CSTATUS [7:0] contains F8H, no serial interrupt is requested. All other I2CSTATUS [7:3] values correspond to defined SIO 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.

6.5.4.5 I2C Clock Baud Rate Bits (I2CLK)

The data baud rate of I2C is determines by I2CLK [7:0] register when SIO is in a master mode. It is not important when SIO is in a slave mode. In the slave modes, SIO 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 = APBCLK / (4x (I2CLK [7:0] +1)). If PCLK=16 MHz, the I2CLK [7:0] = 40 (28H), so data baud rate of I2C = 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec.

6.5.4.6 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 Figure 6.5-9 for the 14-bit time-out counter. User can clear TIF by writing 1 to this bit.

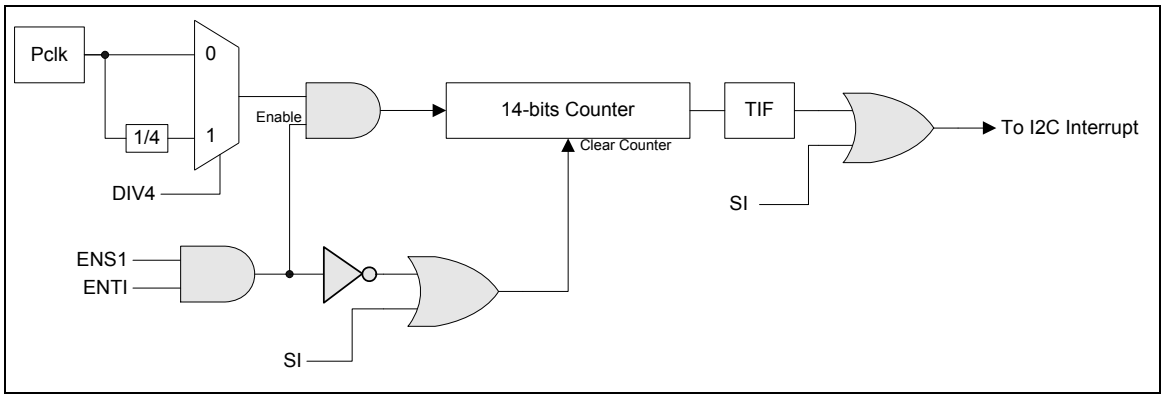


Figure 6.5-9: I2C Time-out Count Block Diagram

6.5.5 I2C Controller Registers Map

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

Register	Offset	R/W	Description	Reset Value
I2C_BA = 0x4002_0000				
I2CON	I2C_BA+0x00	R/W	I2C Control Register	0x0000_0000
I2CADRR0	I2C_BA+0x04	R/W	I2C Slave address Register0	0x0000_0000
I2CDAT	I2C_BA+0x08	R/W	I2C DATA Register	0x0000_0000
I2CSTATUS	I2C_BA+0x0C	R	I2C Status Register	0x0000_00F8
I2CLK	I2C_BA+0x10	R/W	I2C clock divided Register	0x0000_0000
I2CTOC	I2C_BA+0x14	R/W	I2C Time out control Register	0x0000_0000
I2CADDR1	I2C_BA+0x18	R/W	Slave address Register1	0x0000_0000
I2CADDR2	I2C_BA+0x1C	R/W	Slave address Register2	0x0000_0000
I2CADDR3	I2C_BA+0x20	R/W	Slave address Register3	0x0000_0000
I2CADM0	I2C_BA+0x24	R/W	Slave address Mask Register0	0x0000_0000
I2CADM1	I2C_BA+0x28	R/W	Slave address Mask Register1	0x0000_0000
I2CADM2	I2C_BA+0x2C	R/W	Slave address Mask Register2	0x0000_0000
I2CADM3	I2C_BA+0x30	R/W	Slave address Mask Register3	0x0000_0000

6.5.6 I2C Controller Registers 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	**ENSI**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7]	**EI**	**Enable Interrupt** 1 = Enable I2C interrupt 0 = Disable I2C interrupt
[6]	**ENSI**	**I2C Controller Enable Bit** 1 = Enable 0 = Disable Set to enable I2C serial function block. When ENSI=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 SIO 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 one 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_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**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[7:0]							

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 BAUD RATE CONTROL REGISTER (I2CLK)

Register	Offset	R/W	Description	Reset Value
I2CLK	I2C_BA+0x10	R/W	I2C clock divided Register	0x0000_0000

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

Bits	Descriptions	
[31:8]	**Reserved**	Reserved
[7:0]	**I2CLK**	**I2C clock divided Register** The I2C clock rate bits: Data Baud Rate of I2C = PCLK / (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 is enabled/disable** 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** 1 = Time-Out flag is set by H/W. It can interrupt CPU. 0 = S/W can clear the flag.

I2C SLAVE ADDRESS REGISTER (I2CADDRx)

Register	Offset	R/W	Description	Reset Value
I2CADDR0	I2C_BA+0x04	R/W	I2C slave Address Register0	0x0000_0000
I2CADDR1	I2C_BA+0x18	R/W	I2C slave Address Register1	0x0000_0000
I2CADDR2	I2C_BA+0x1C	R/W	I2C slave Address Register2	0x0000_0000
I2CADDR3	I2C_BA+0x20	R/W	I2C slave Address Register3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CADDR[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 MCU'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 Register0	0x0000_0000
I2CADM1	I2C_BA+0x28	R/W	I2C slave Address Mask Register1	0x0000_0000
I2CADM2	I2C_BA+0x2C	R/W	I2C slave Address Mask Register2	0x0000_0000
I2CADM3	I2C_BA+0x30	R/W	I2C slave Address Mask Register3	0x0000_0000

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

6.5.7 Modes of Operation

The on-chip I2C ports support five operation modes, Master transmitter, Master receiver, Slave transmitter, Slave receiver, and GC call.

In a given application, I2C port may operate as a master or as a slave. In the slave mode, the I2C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master(by setting the AA bit), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, 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, I2C port switches to the slave mode immediately and can detect its own slave address in the same serial transfer.

6.5.7.1 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 Figure 6.5-11. 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.

6.5.7.2 Master Receiver Mode

In this case the data direction bit (R/W) will be logic 1, and it is represented by "R" in the Figure 6.5-12. 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.

6.5.7.3 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.

6.5.7.4 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.

6.5.8 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 SIO 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 I2C 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 Figure 6.5-10.

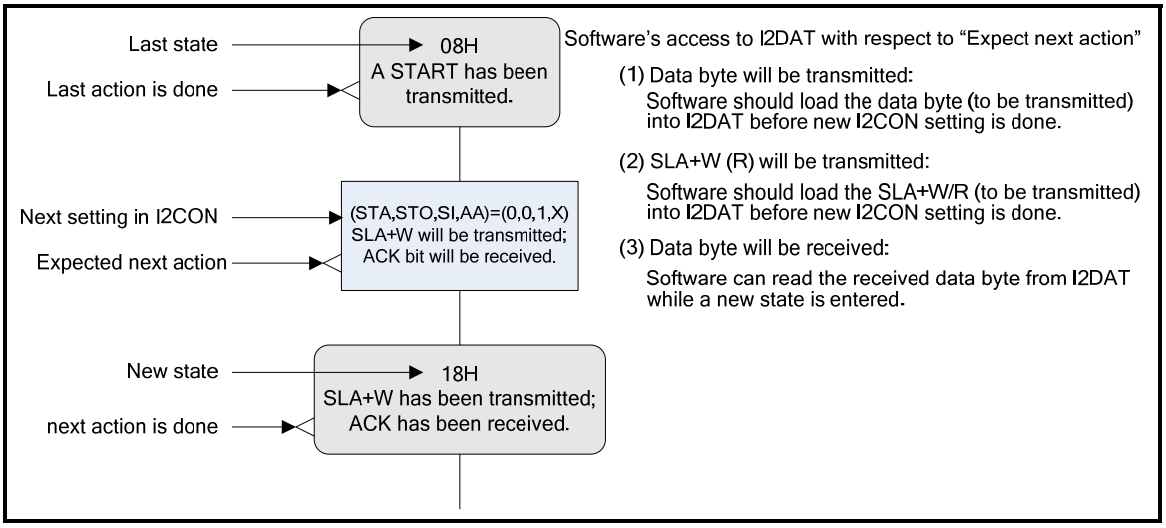


Figure 6.5-10 Legend for the following five figures

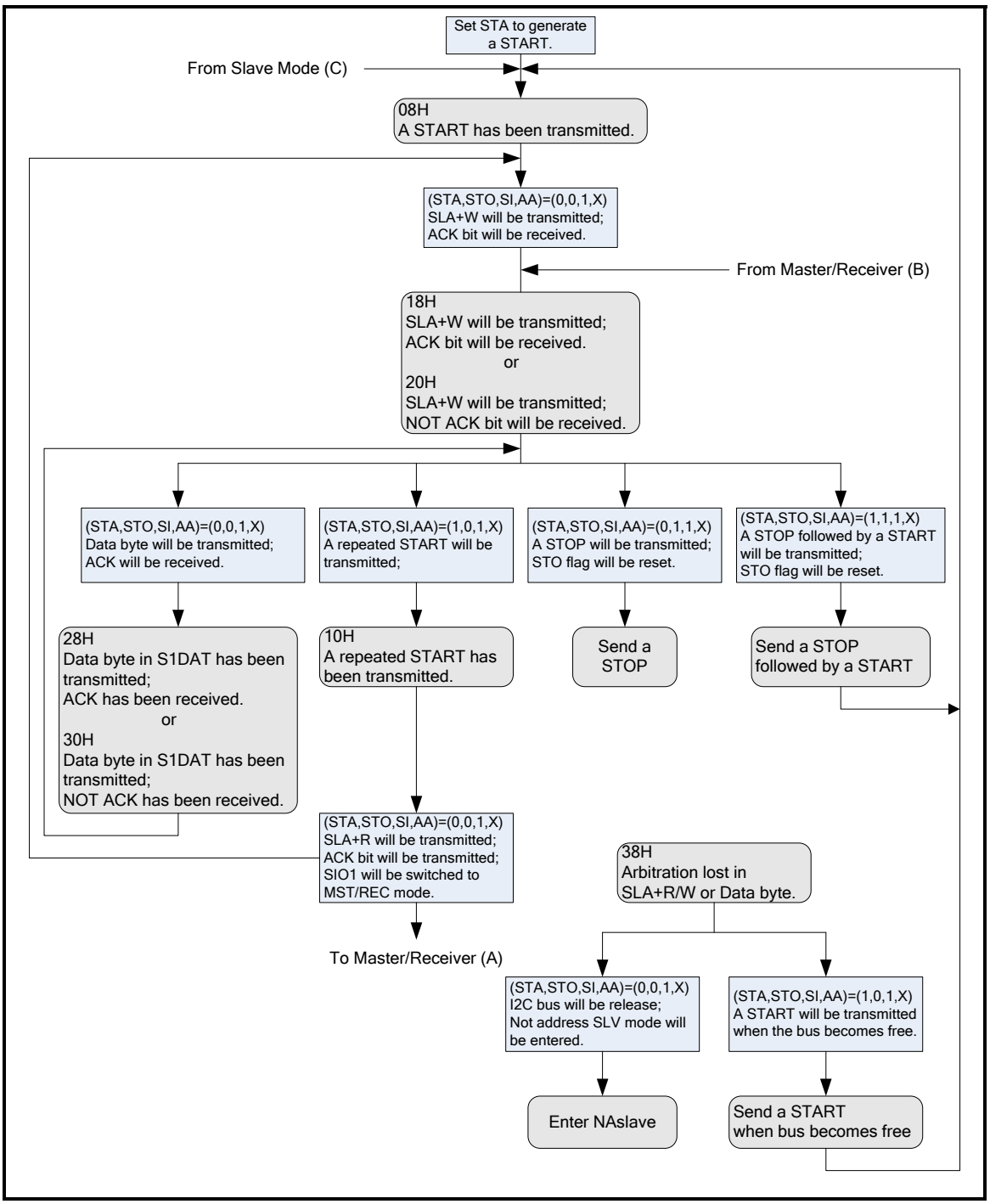


Figure 6.5-11 Master Transmitter Mode

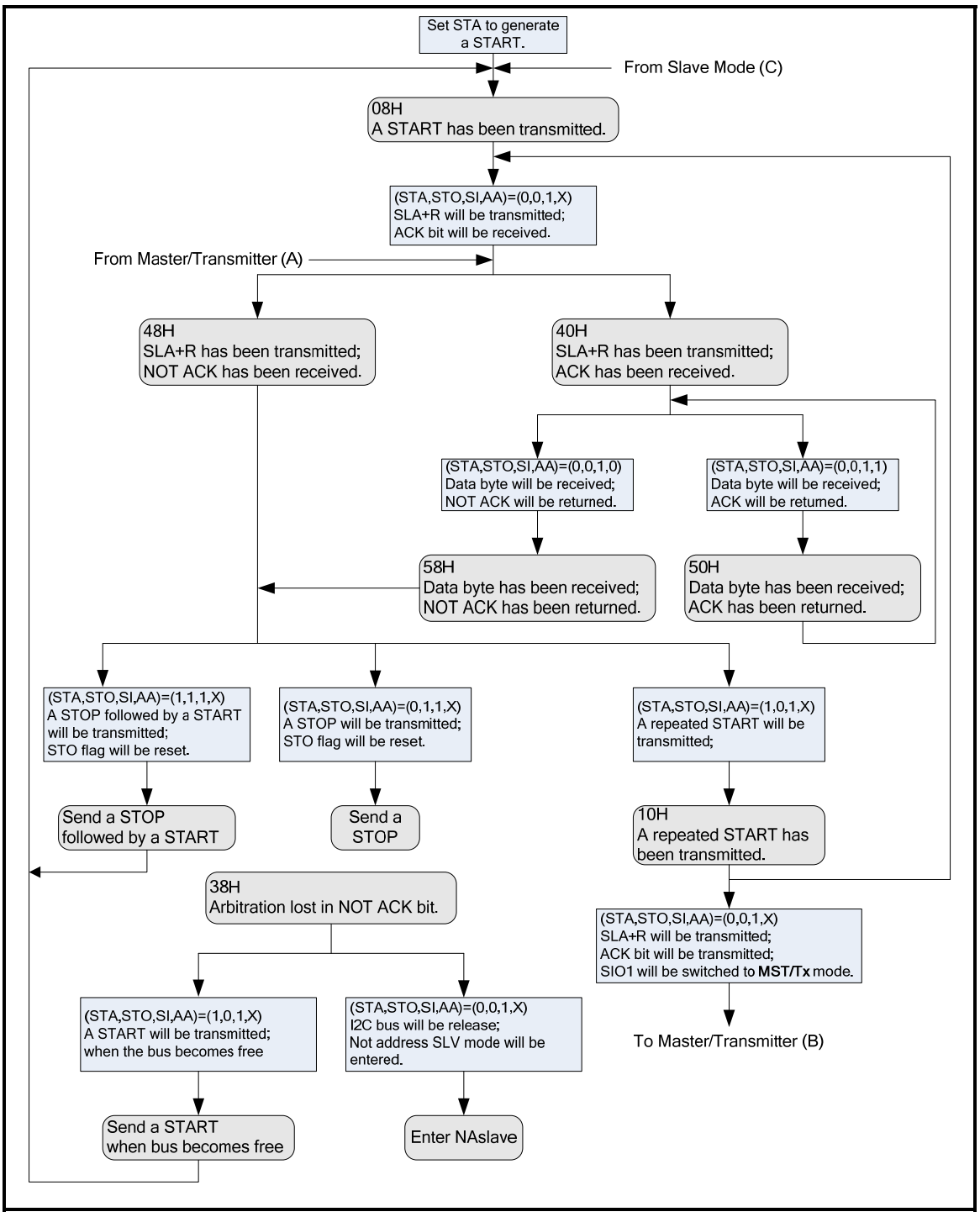


Figure 6.5-12 Master Receiver Mode

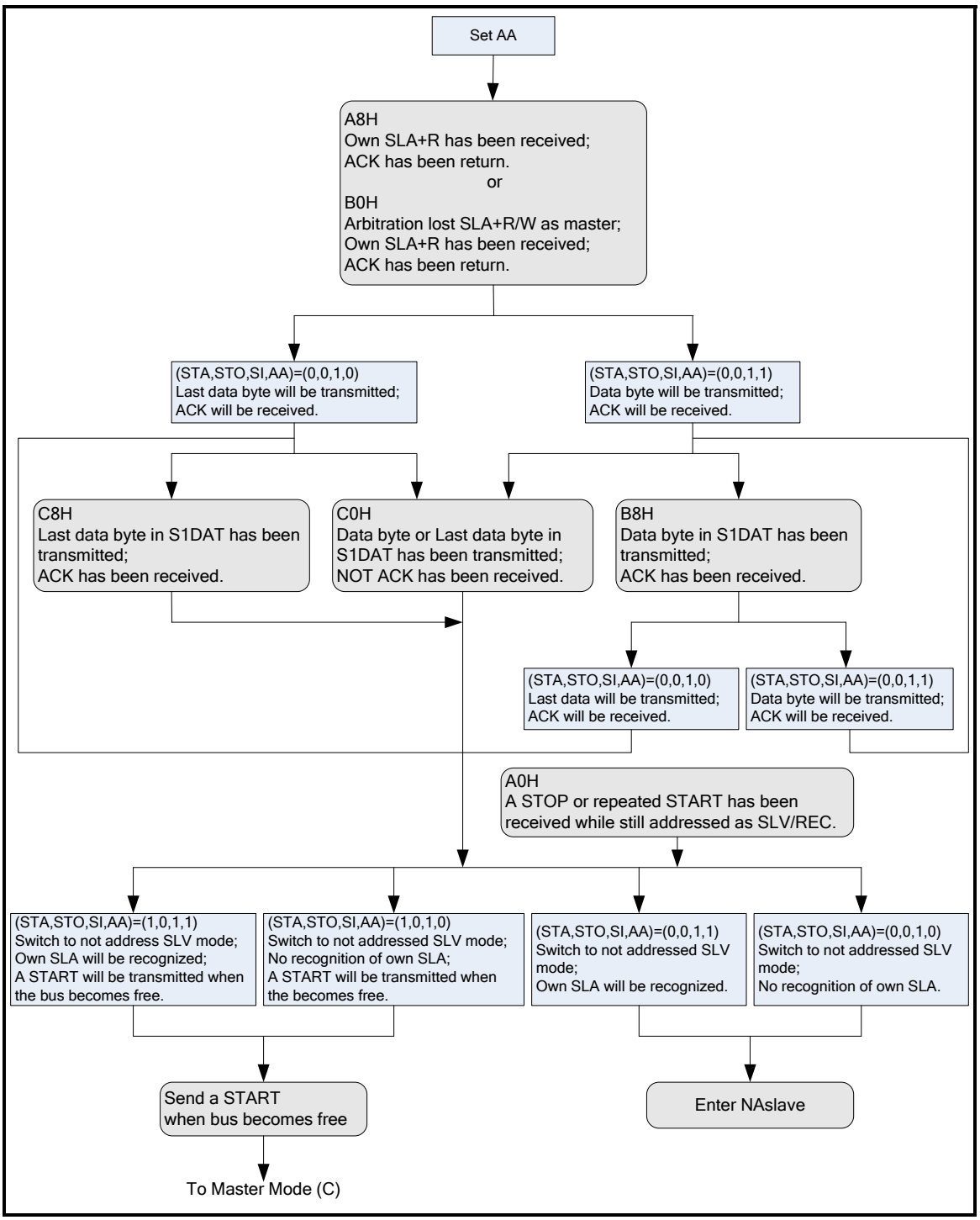


Figure 6.5-13 Slave Transmitter Mode

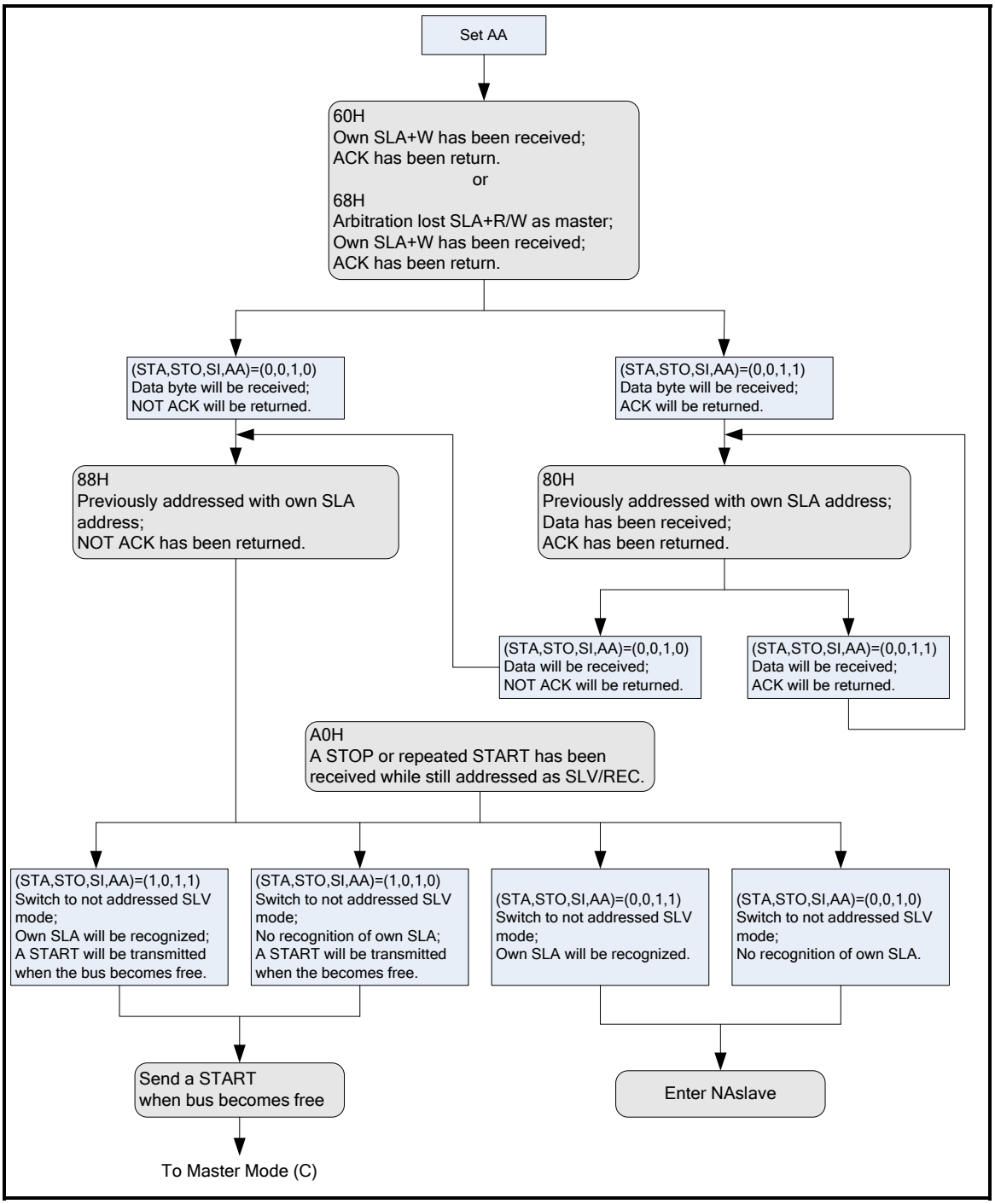


Figure 6.5-14 Slave Receiver Mode

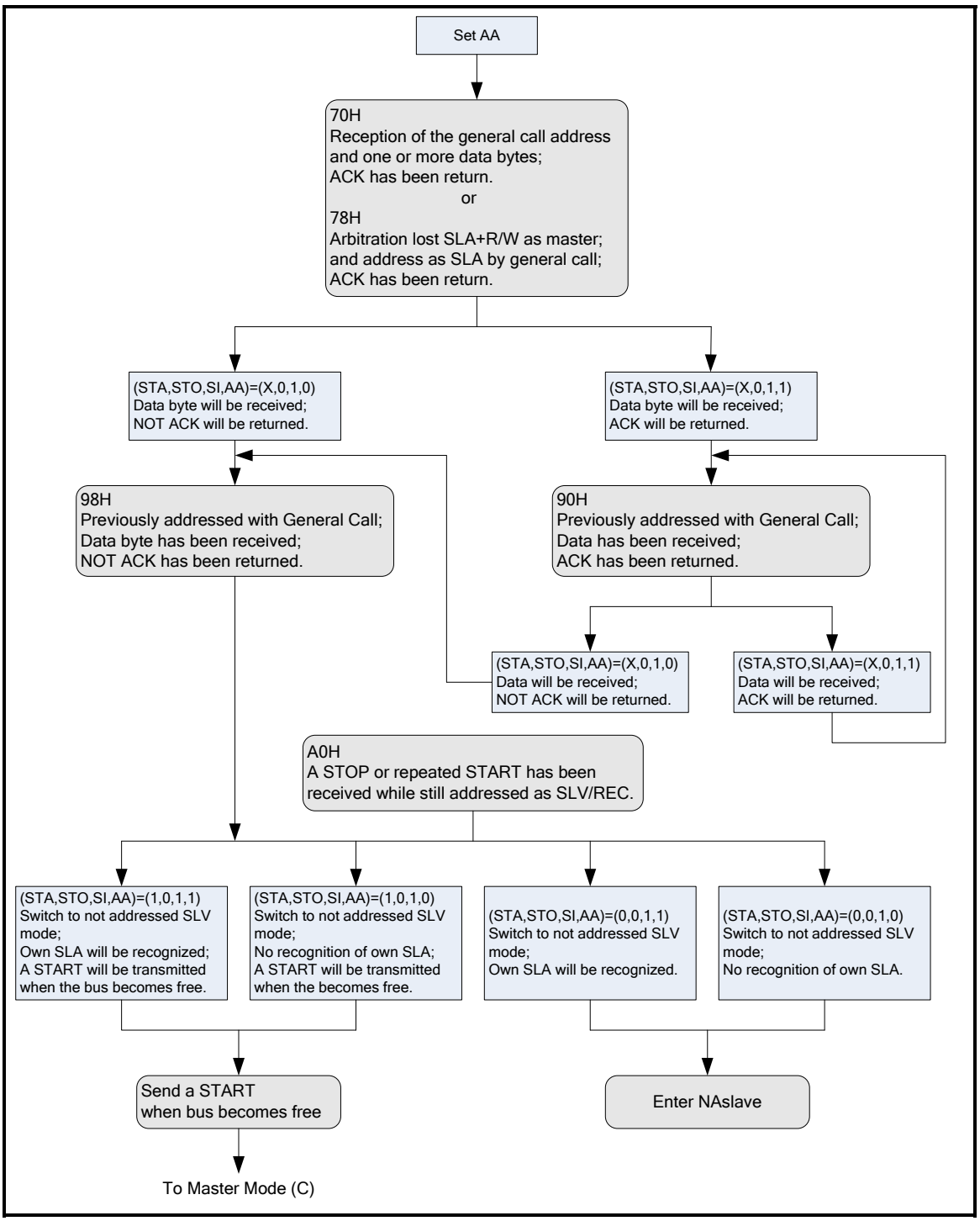


Figure 6.5-15 GC Mode

6.6 PWM Generator and Capture Timer

6.6.1 Overview

NuMicro M051™ series has 2 sets of PWM group supports 4 sets of PWM Generators which can be configured as 8 independent PWM outputs, PWM0~PWM7, or as 4 complementary PWM pairs, (PWM0, PWM1), (PWM2, PWM3), (PWM4, PWM5) and (PWM6, PWM7) with 4 programmable dead-zone generators.

Each PWM Generator has one 8-bit prescaler, one clock divider with 5 divided frequencies (1, 1/2, 1/4, 1/8, 1/16), two PWM Timers including two clock selectors, two 16-bit PWM 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 timing, period, duty and dead-time are determined by PWM0 timer and Dead-zone generator 0. Similarly, the complementary PWM pairs of (PWM2, PWM3), (PWM4, PWM5) and (PWM6, PWM7) are controlled by PWM2, PWM4 and PWM6 timers and Dead-zone generator 2, 4 and 6, respectively. Refer to 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 PWM 0; 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 0 to channel 3 on each group have the same feature by setting the corresponding control bits in CCR0 and 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 PWM_CRLx/PWM_CFLx(x=0 and 3) to get capture value and finally write 1 to clear PIIRx. 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 is 1/900ns ≈ 1000 kHz

6.6.2 Features

6.6.2.1 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
- Up to 2 PWM group (PWMA/PWMB) to support 8 PWM channels

6.6.2.2 Capture Function Features:

- Timing control logic shared with PWM Generators
- 8 capture input channels shared with 8 PWM output channels
- Each channel supports one rising latch register (CRLR), one falling latch register (CFLR) and Capture interrupt flag (CAPIFx)

6.6.3 Block Diagram

The Figure 6.6-1 illustrate the architecture of PWM in pair (Timer 0&1 are in one pair and timer 2&3 are in another one, and so on.).

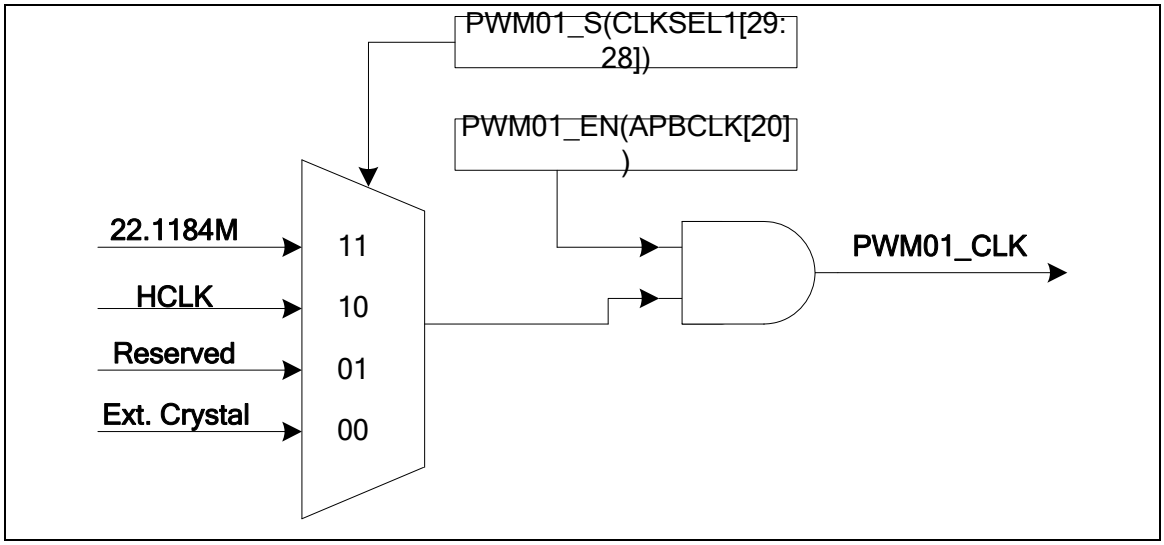


Figure 6.6-1 PWM Generator 0 Clock Source Control

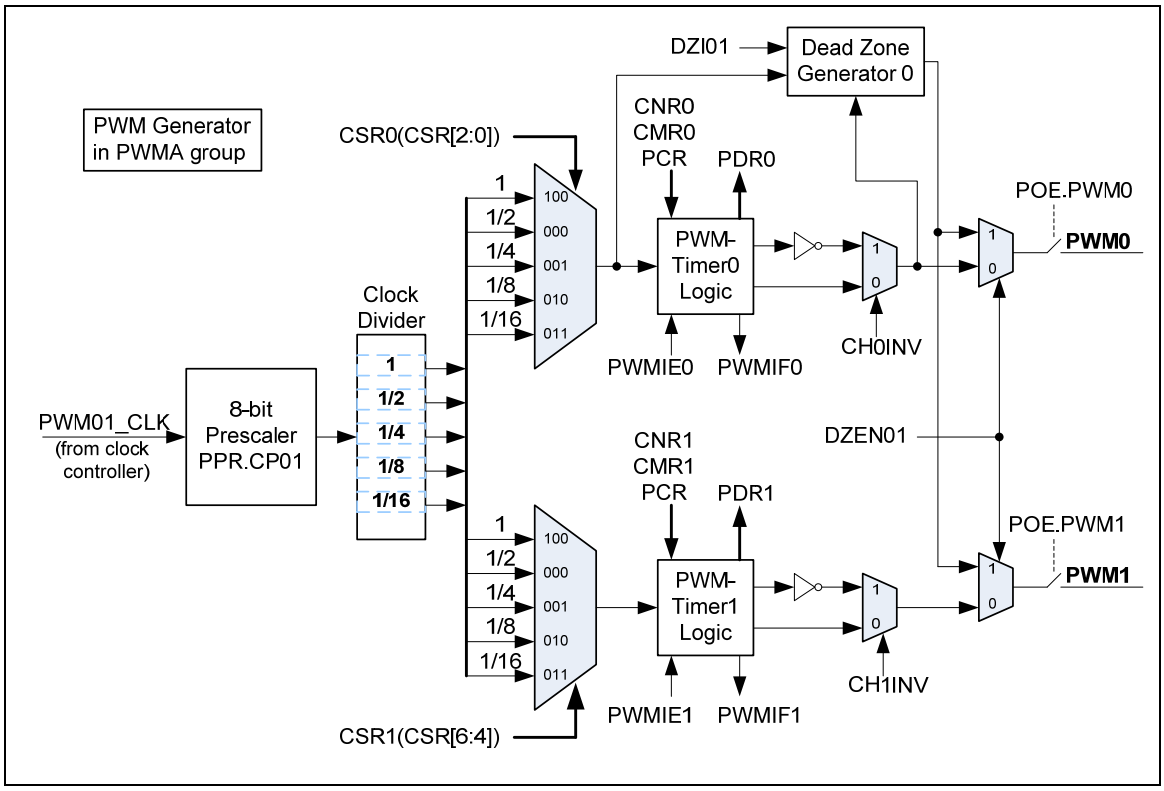


Figure 6.6-2 PWM Generator 0 Architecture Diagram

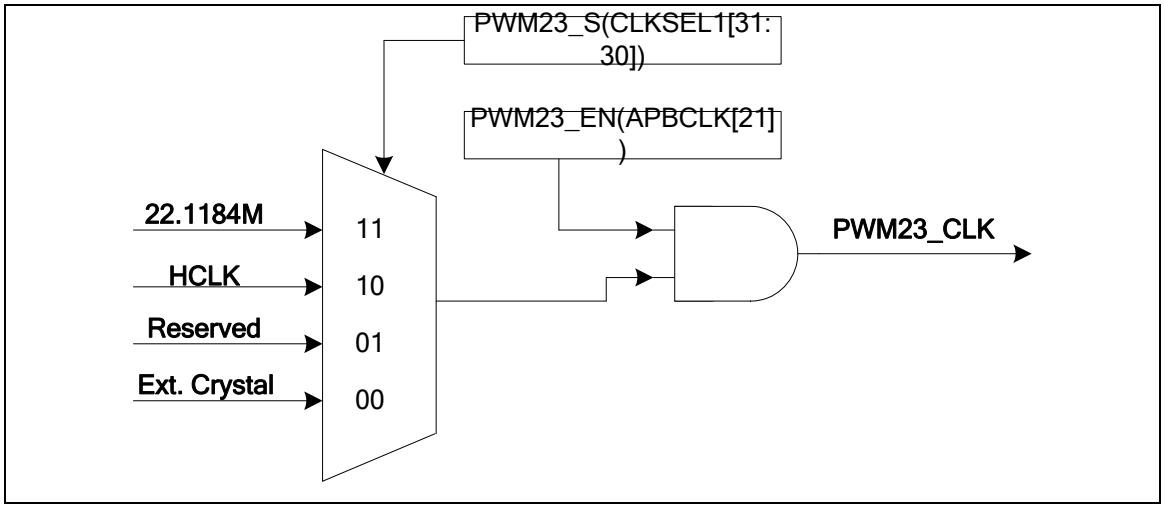


Figure 6.6-3 PWM Generator 2 Clock Source Control

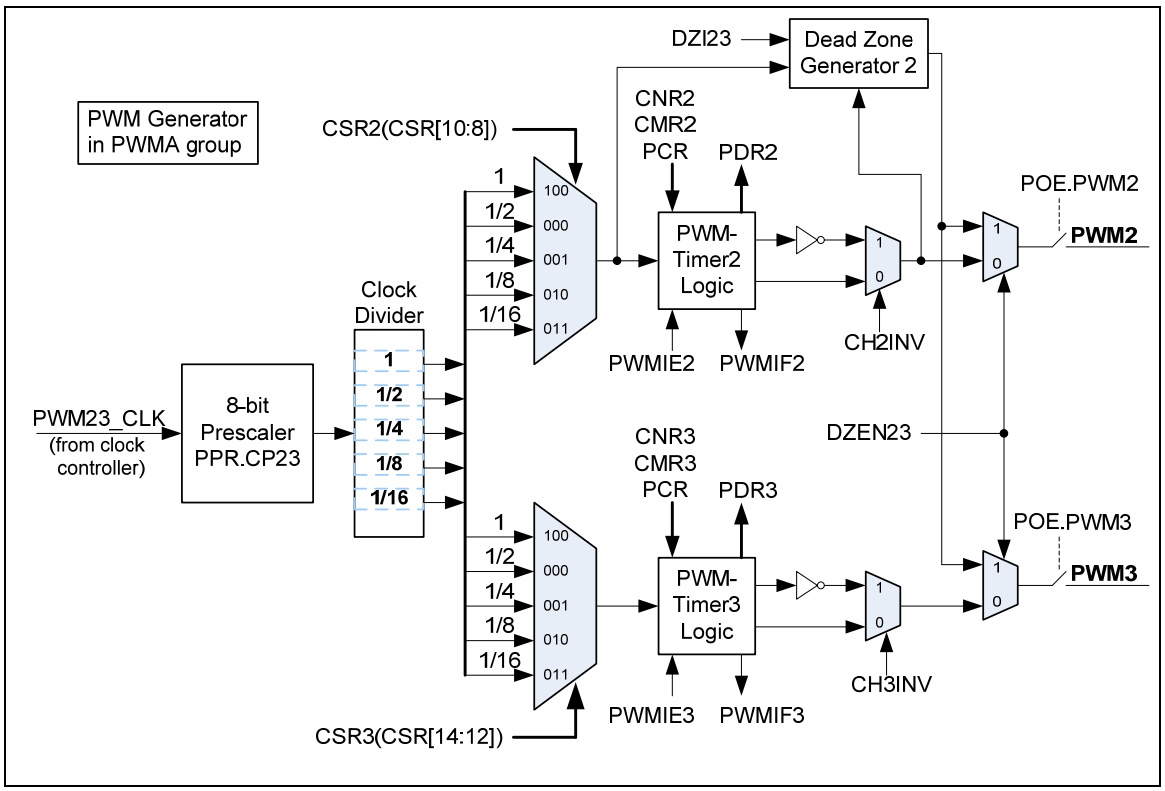


Figure 6.6-4 PWM Generator 2 Architecture Diagram

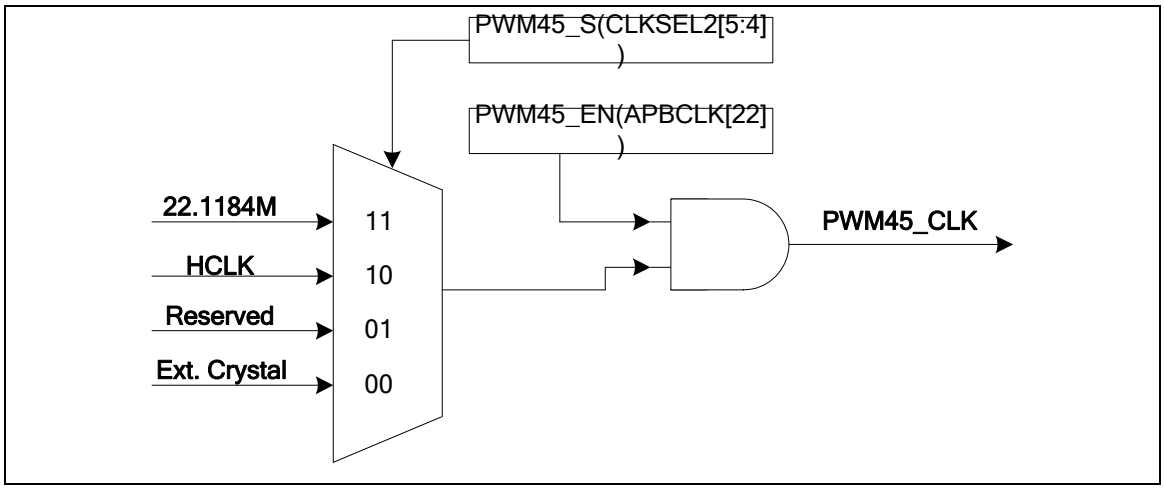


Figure 6.6-5 PWM Generator 4 Clock Source Control

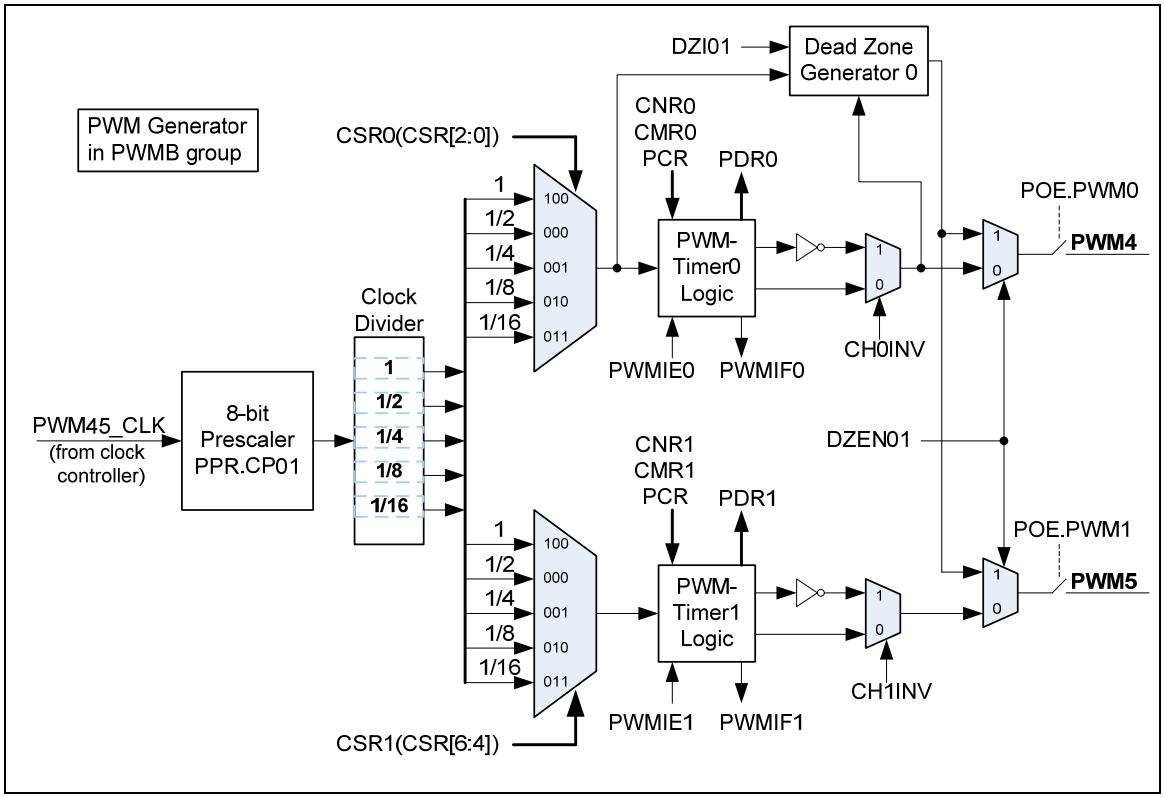


Figure 6.6-6 PWM Generator 4 Architecture Diagram

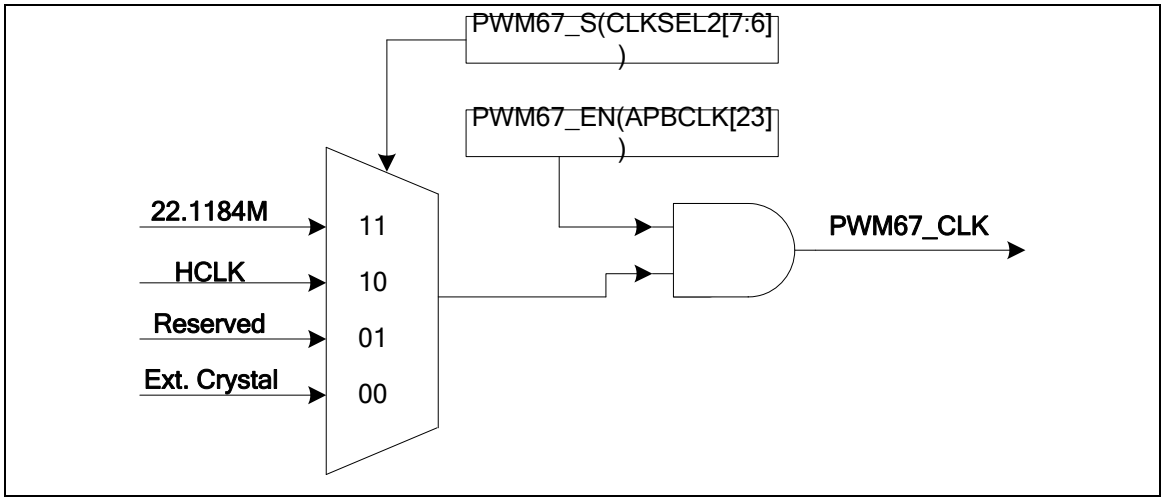


Figure 6.6-7 PWM Generator 6 Clock Source Control

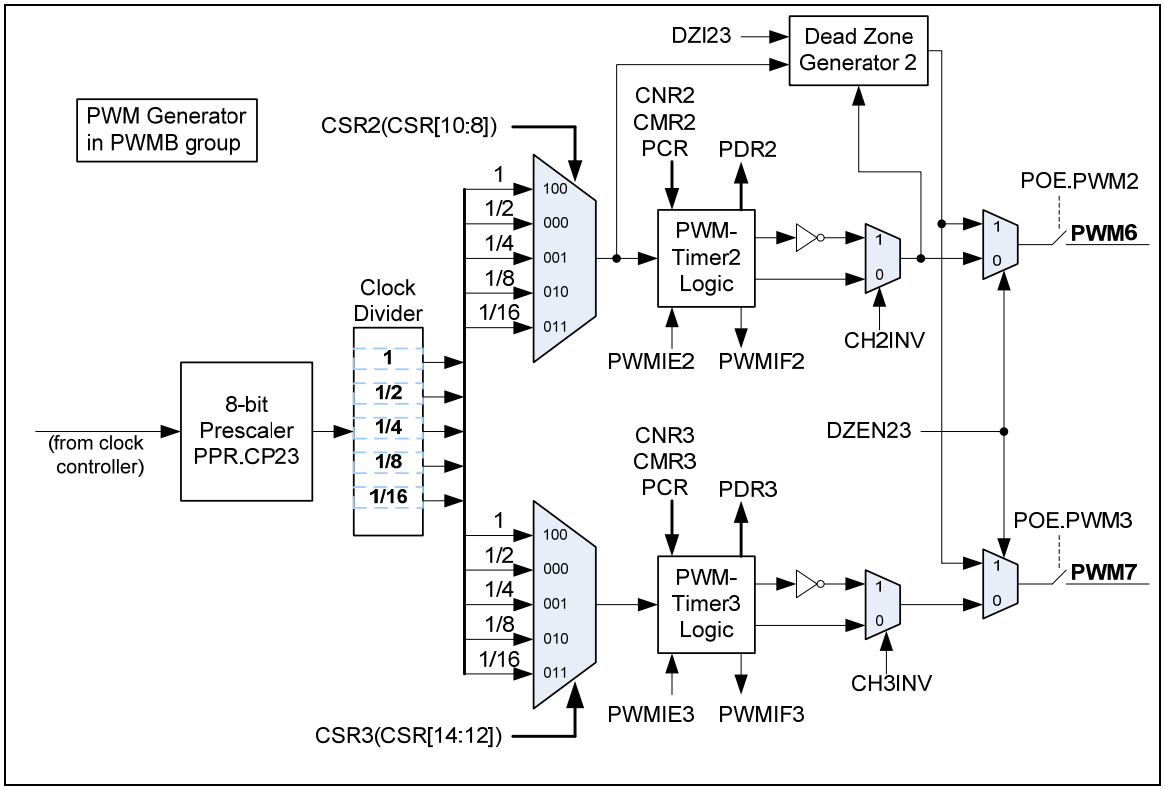


Figure 6.6-8 PWM Generator 6 Architecture Diagram

6.6.4 Function Description

6.6.4.1 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 the Figure 6.6-10. The pulse width modulation follows the formula as below and the legend of PWM-Timer Comparator is shown in the Figure 6.6-9 note that the corresponding GPIO pins must be configured as PWM function (enable POE and disable CAPENR) for the corresponding PWM channel.

PWM frequency = PWMxy_CLK/(prescale+1)*(clock divider)/(CNR+1); where xy, could be 01, 23, 45 or 67, depends on selected PWM channel.

- Duty ratio = (CMR+1)/(CNR+1)
- CMR >= CNR: PWM output is always high
- CMR < CNR: PWM low width= (CNR-CMR) unit[1]; PWM high width = (CMR+1) unit
- CMR = 0: PWM low width = (CNR) unit; PWM high width = 1 unit

Note: 1. Unit = one PWM clock cycle.

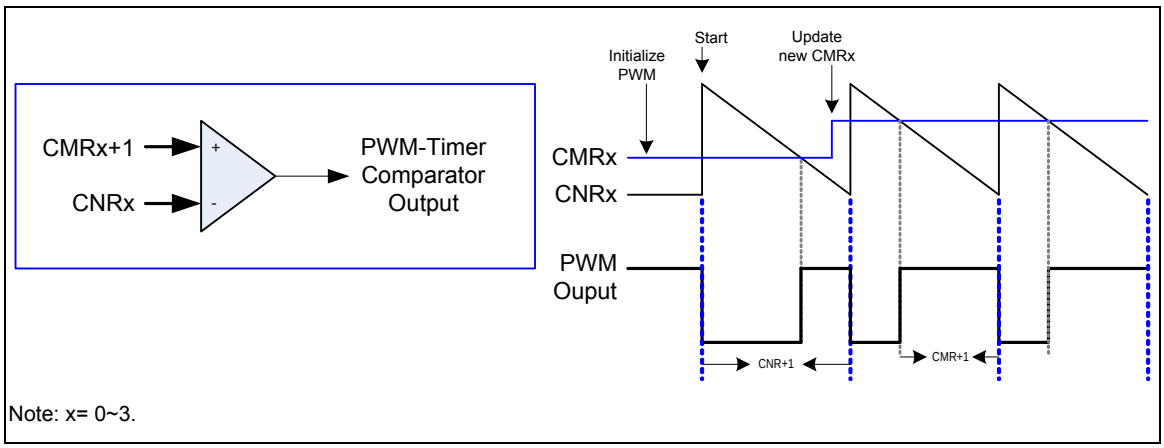


Figure 6.6-9 Legend of Internal Comparator Output of PWM-Timer

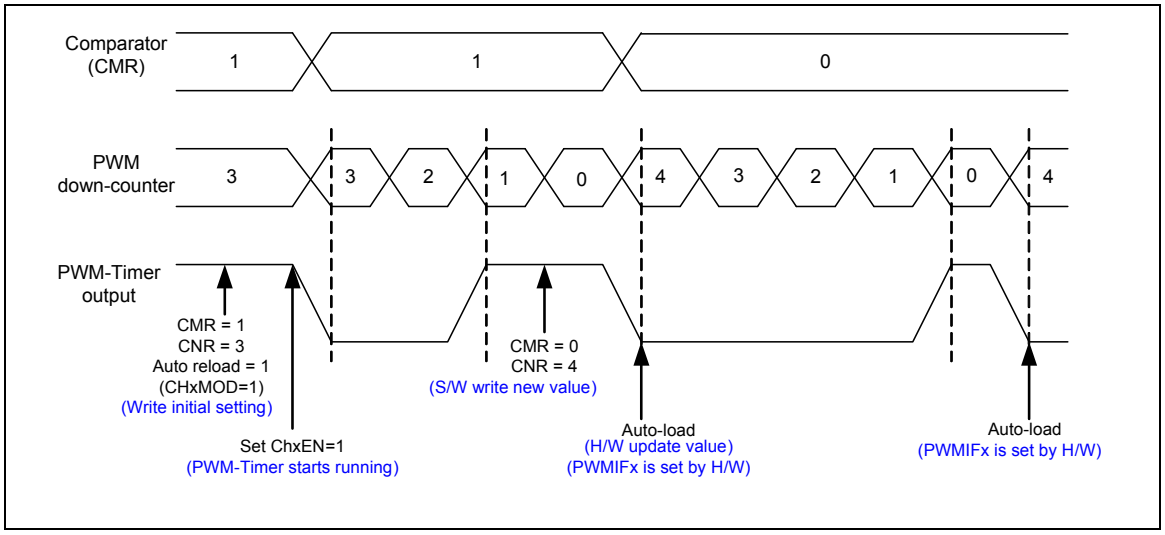


Figure 6.6-10 PWM-Timer Operation Timing

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

NuMicro M051™ series 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~PWM7 performs the same function as PWM0.

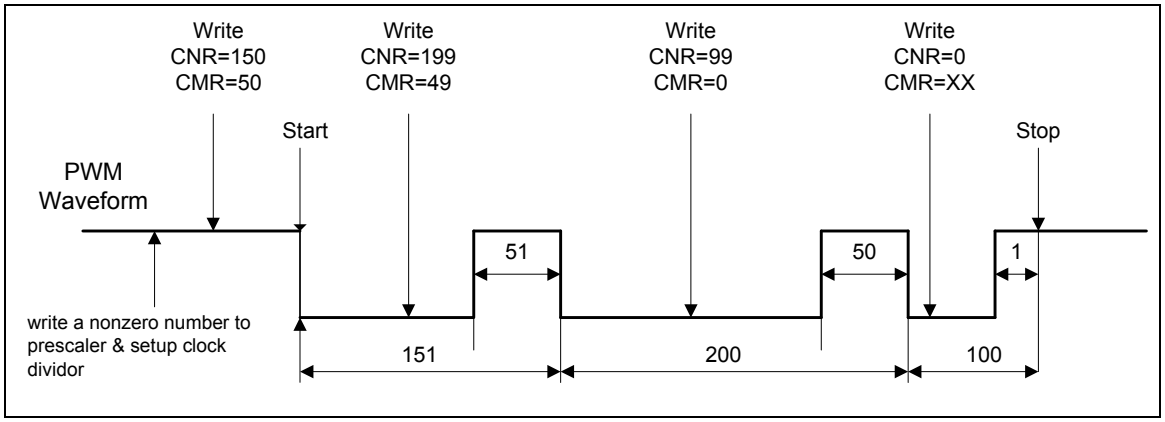


Figure 6.6-11 PWM Double Buffering Illustration

6.6.4.3 Modulate Duty Ratio

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

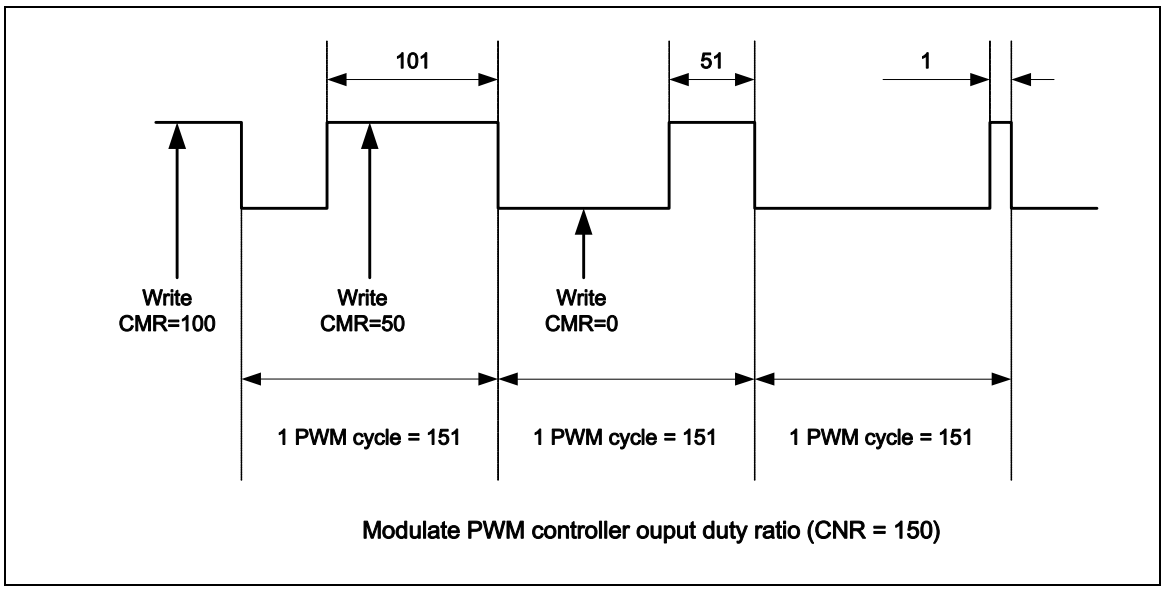


Figure 6.6-12 PWM Controller Output Duty Ratio

6.6.4.4 Dead-Zone Generator

NuMicro M051™ series PWM is implemented with Dead Zone generator. They are built for power device protection. This function generates a programmable time gap to delay PWM rising output. User can program PPRx.DZI to determine the Dead Zone interval.

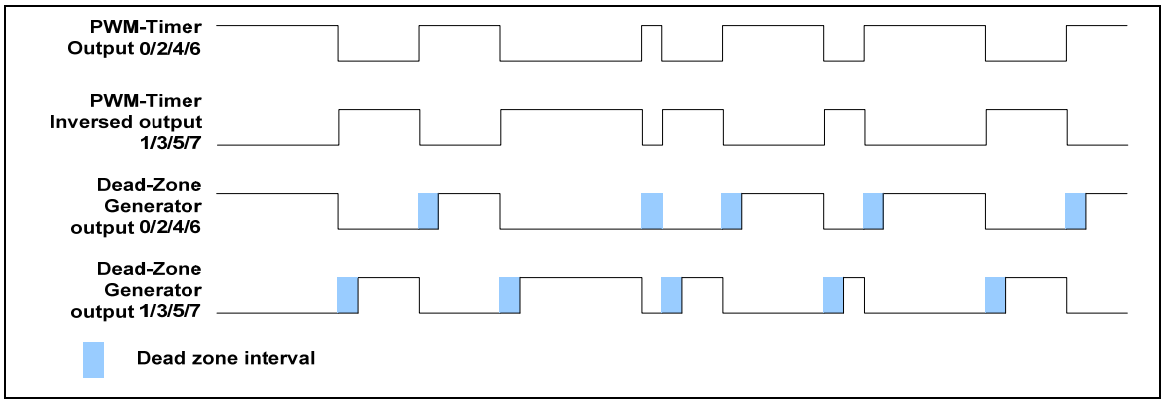


Figure 6.6-13 Paired-PWM Output with Dead Zone Generation Operation

6.6.4.5 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 module 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.

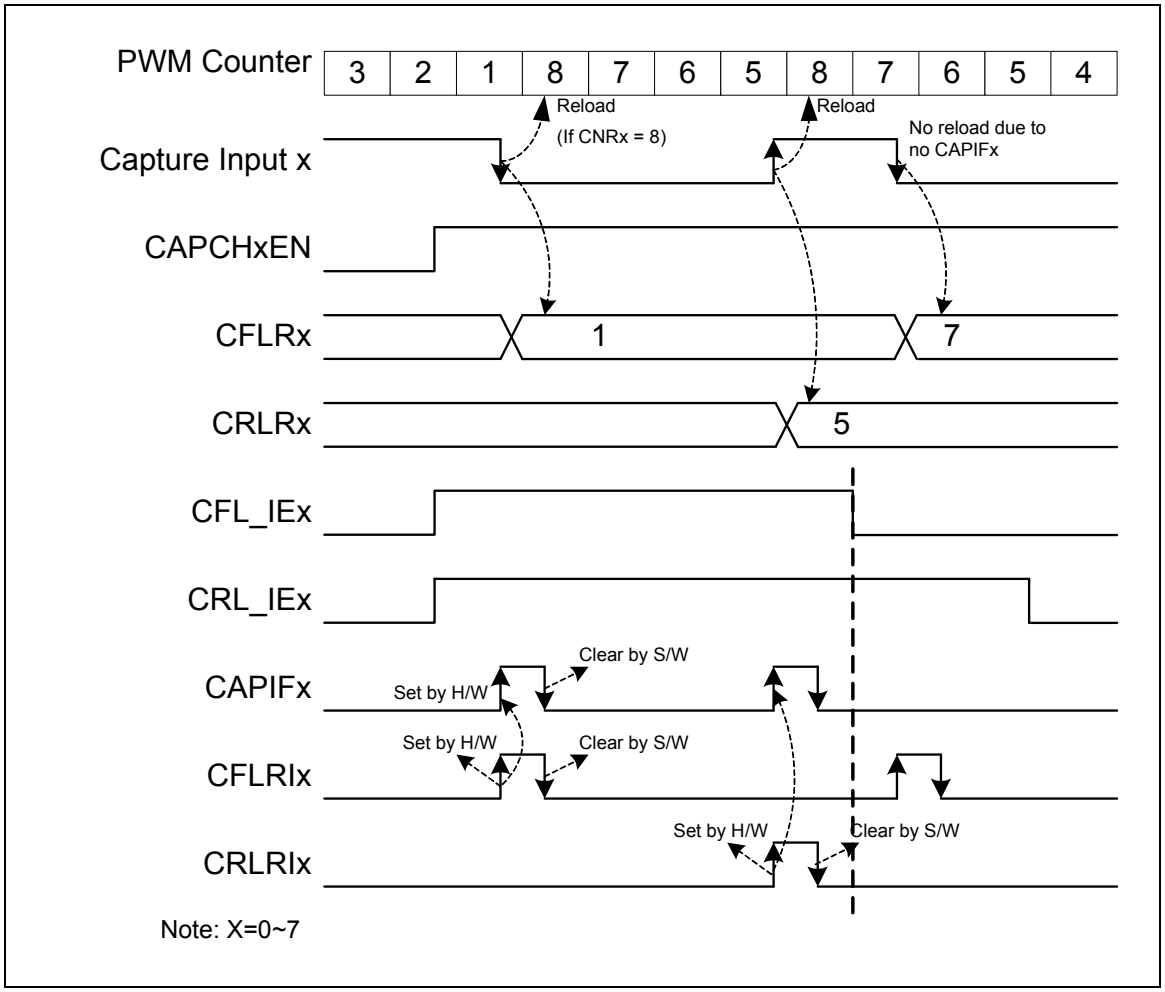


Figure 6.6-14 Capture Operation Timing

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).

6.6.4.6 PWM-Timer Interrupt Architecture

There are eight PWM interrupts, PWM0_INT~PWM7_INT, which are divided into PWMA_INT and PWMB_INT for Advanced Interrupt Controller (AIC). PWM 0 and Capture 0 share one interrupt, PWM1 and Capture 1 share the same interrupt and so on. Therefore, PWM function and Capture function in the same channel cannot be used at the same time. The Figure 6.6-15 demonstrates the architecture of PWM-Timer interrupts.

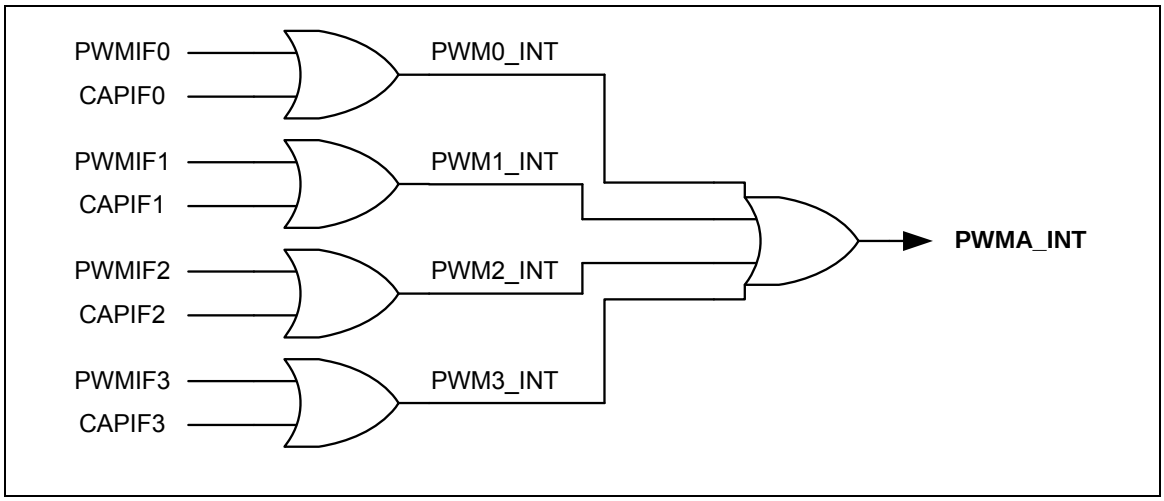


Figure 6.6-15 PWM Group A PWM-Timer Interrupt Architecture Diagram

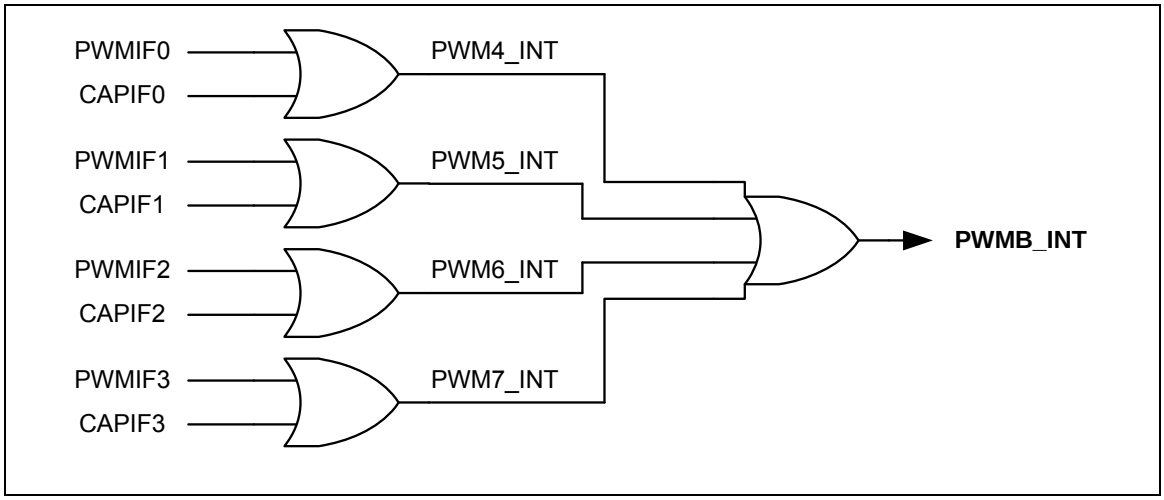


Figure 6.6-16 PWM Group B PWM-Timer Interrupt Architecture Diagram

6.6.4.7 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)

6.6.4.8 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 happens, disable PWM-Timer (CHxEN in PCR). *(Recommended)*

Method 3:

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

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

6.6.4.9 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, CCR1)
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)

6.6.5 Controller Registers 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) **PWMB_BA = 0x4014_0000 (PWM group B)**				
PPR	PWMA_BA+0x00	R/W	PWM Group A Prescaler Register	0x0000_0000
	PWMB_BA+0x00	R/W	PWM Group B Prescaler Register	0x0000_0000
CSR	PWMA_BA+0x04	R/W	PWM Group A Clock Select Register	0x0000_0000
	PWMB_BA+0x04	R/W	PWM Group B Clock Select Register	0x0000_0000
PCR	PWMA_BA+0x08	R/W	PWM Group A Control Register	0x0000_0000
	PWMB_BA+0x08	R/W	PWM Group B Control Register	0x0000_0000
CNR0	PWMA_BA+0x0C	R/W	PWM Group A Counter Register 0	0x0000_0000
	PWMB_BA+0x0C	R/W	PWM Group B Counter Register 0	0x0000_0000
CMR0	PWMA_BA+0x10	R/W	PWM Group A Comparator Register 0	0x0000_0000
	PWMB_BA+0x10	R/W	PWM Group B Comparator Register 0	0x0000_0000
PDR0	PWMA_BA+0x14	R	PWM Group A Data Register 0	0x0000_0000
	PWMB_BA+0x14	R	PWM Group B Data Register 0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWM Group A Counter Register 1	0x0000_0000
	PWMB_BA+0x18	R/W	PWM Group B Counter Register 1	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWM Group A Comparator Register 1	0x0000_0000
	PWMB_BA+0x1C	R/W	PWM Group B Comparator Register 1	0x0000_0000
PDR1	PWMA_BA+0x20	R	PWM Group A Data Register 1	0x0000_0000
	PWMB_BA+0x20	R	PWM Group B Data Register 1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM Group A Counter Register 2	0x0000_0000
	PWMB_BA+0x24	R/W	PWM Group B Counter Register 2	0x0000_0000

CMR2	PWMA_BA+0x28	R/W	PWM Group A Comparator Register 2	0x0000_0000
	PWMB_BA+0x28	R/W	PWM Group B Comparator Register 2	0x0000_0000
PDR2	PWMA_BA+0x2C	R	PWM Group A Data Register 2	0x0000_0000
	PWMB_BA+0x2C	R	PWM Group B Data Register 2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM Group A Counter Register 3	0x0000_0000
	PWMB_BA+0x30	R/W	PWM Group B Counter Register 3	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM Group A Comparator Register 3	0x0000_0000
	PWMB_BA+0x34	R/W	PWM Group B Comparator Register 3	0x0000_0000
PDR3	PWMA_BA+0x38	R	PWM Group A Data Register 3	0x0000_0000
	PWMB_BA+0x38	R	PWM Group B Data Register 3	0x0000_0000
PIER	PWMA_BA+0x40	R/W	PWM Group A Interrupt Enable Register	0x0000_0000
	PWMB_BA+0x40	R/W	PWM Group B Interrupt Enable Register	0x0000_0000
PIIR	PWMA_BA+0x44	R/C	PWM Group A Interrupt Indication Register	0x0000_0000
	PWMB_BA+0x44	R/C	PWM Group B Interrupt Indication Register	0x0000_0000
CCR0	PWMA_BA+0x50	R/W	PWM Group A Capture Control Register 0	0x0000_0000
	PWMB_BA+0x50	R/W	PWM Group B Capture Control Register 0	0x0000_0000
CCR2	PWMA_BA+0x54	R/W	PWM Group A Capture Control Register 2	0x0000_0000
	PWMB_BA+0x54	R/W	PWM Group B Capture Control Register 2	0x0000_0000
CRLR0	PWMA_BA+0x58	R/W	PWM Group A Capture Rising Latch Register (Channel 0)	0x0000_0000
	PWMB_BA+0x58	R/W	PWM Group B 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
	PWMB_BA+0x5C	R/W	PWM Group B 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
	PWMB_BA+0x60	R/W	PWM Group B 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
	PWMB_BA+0x64	R/W	PWM Group B 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
	PWMB_BA+0x68	R/W	PWM Group B 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
	PWMB_BA+0x6C	R/W	PWM Group B 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
	PWMB_BA+0x70	R/W	PWM Group B 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
	PWMB_BA+0x74	R/W	PWM Group B 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
	PWMB_BA+0x78	R/W	PWM Group B 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
	PWMB_BA+0x7C	R/W	PWM Group B Output Enable for channel 0~3	0x0000_0000

6.6.6 Controller Registers 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
	PWMB_BA+0x00	R/W	PWM Group B 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 register for pair of channel2 and channel3** (PWM2 and PWM3 pair for PWM group A, PWM6 and PWM7 pair for PWM group B) 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 register for pair of channel 0 and channel 1** (PWM0 and PWM1 pair for PWM group A, PWM4 and PWM5 pair for PWM group B) These 8 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 counter 2 & 3 for group A and PWM counter 6 & 7 for group B) Clock input is divided by (CP23 + 1) before it is fed to the corresponding PWM counter If CP23=0, then the clock prescaler 2 output clock will be stopped. So corresponding

		PWM counter will be stopped also.
[7:0]	**CP01**	**Clock prescaler 0** (PWM counter 0 & 1 for group A and PWM counter 4 & 5 for group B) Clock input is divided by (CP01 + 1) before it is fed to the corresponding PWM counter If CP01=0, then the clock prescaler 0 output clock will be stopped. So corresponding PWM counter 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 Selector Register	0x0000_0000
	PWMB_BA+0x04	R/W	PWM Group B Clock Selector Register	0x0000_0000

31	30	29	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**	**Timer 3 Clock Source Selection** (PWM timer 3 for group A and PWM timer 7 for group B) 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**	**Timer 2 Clock Source Selection** (PWM timer 2 for group A and PWM timer 6 for group B) Select clock input for PWM timer. (Table is the same as CSR3)
[7]	**Reserved**	Reserved
[6:4]	**CSR1**	**Timer 1 Clock Source Selection** (PWM timer 1 for group A and PWM timer 5 for group B) Select clock input for PWM timer. (Table is the same as CSR3)
[3]	**Reserved**	Reserved
[2:0]	**CSR0**	**Timer 0 Clock Source Selection** (PWM timer 0 for group A and PWM timer 4 for group B) 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
	PWMB_BA+0x08	R/W	PWM Group B 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 and PWM timer 7 for group B) 1 = Auto-reload Mode 0 = One-Shot Mode Note: If there is a rising transition at this bit, it will cause CNR3 and CMR3 be clear.
[26]	**CH3INV**	**PWM-Timer 3 Output Inverter ON/OFF** (PWM timer 3 for group A and PWM timer 7 for group B) 1 = Inverter ON 0 = Inverter OFF
[25]	**Reserved**	Reserved

[24]	**CH3EN**	**PWM-Timer 3 Enable/Disable Start Run** (PWM timer 3 for group A and PWM timer 7 for group B) 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 and PWM timer 6 for group B) 1 = Auto-reload Mode 0 = One-Shot Mode Note: If there is a rising transition at this bit, it will cause CNR2 and CMR2 be clear.
[18]	**CH2INV**	**PWM-Timer 2 Output Inverter ON/OFF** (PWM timer 2 for group A and PWM timer 6 for group B) 1 = Inverter ON 0 = Inverter OFF
[17]	**Reserved**	Reserved
[16]	**CH2EN**	**PWM-Timer 2 Enable/Disable Start Run** (PWM timer 2 for group A and PWM timer 6 for group B) 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 and PWM timer 5 for group B) 1 = Auto-load Mode 0 = One-Shot Mode Note: If there is a rising transition at this bit, it will cause CNR1 and CMR1 be clear.
[10]	**CH1INV**	**PWM-Timer 1 Output Inverter ON/OFF** (PWM timer 1 for group A and PWM timer 5 for group B) 1 = Inverter ON 0 = Inverter OFF
[9]	**Reserved**	Reserved

[8]	**CH1EN**	**PWM-Timer 1 Enable/Disable Start Run** (PWM timer 1 for group A and PWM timer 5 for group B) 1 = Enable corresponding PWM-Timer Start Run 0 = Stop corresponding PWM-Timer Running
[7:6]	**Reserved**	Reserved
[5]	**DZEN23**	**Dead-Zone 2 Generator Enable/Disable** (PWM2 and PWM3 pair for PWM group A, PWM6 and PWM7 pair for PWM group B) 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 and the pair of PWM6 and PWM7 becomes a complementary pair for PWM group B.
[4]	**DZEN01**	**Dead-Zone 0 Generator Enable/Disable** (PWM0 and PWM1 pair for PWM group A, PWM4 and PWM5 pair for PWM group B) 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 and the pair of PWM4 and PWM5 becomes a complementary pair for PWM group B.
[3]	**CH0MOD**	**PWM-Timer 0 Auto-reload/One-Shot Mode** (PWM timer 0 for group A and PWM timer 4 for group B) 1 = Auto-reload Mode 0 = One-Shot Mode Note: If there is a rising transition at this bit, it will cause CNR0 and CMR0 be clear.
[2]	**CH0INV**	**PWM-Timer 0 Output Inverter ON/OFF** (PWM timer 0 for group A and PWM timer 4 for group B) 1 = Inverter ON 0 = Inverter OFF
[1]	**Reserved**	Reserved
[0]	**CH0EN**	**PWM-Timer 0 Enable/Disable Start Run** (PWM timer 0 for group A and PWM timer 4 for group B) 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
	PWMB_BA+0x0C	R/W	PWM Group B Counter Register 0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWM Group A Counter Register 1	0x0000_0000
	PWMB_BA+0x18	R/W	PWM Group B Counter Register 1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM Group A Counter Register 2	0x0000_0000
	PWMB_BA+0x24	R/W	PWM Group B Counter Register 2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM Group A Counter Register 3	0x0000_0000
	PWMB_BA+0x30	R/W	PWM Group B 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 Counter/Timer Loaded Value** CNR determines the PWM period. ● PWM frequency = PWMxy_CLK/(prescale+1)*(clock divider)/(CNR+1); where xy, could be 01, 23, 45 or 67, depends on selected PWM

		channel. ● 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
	PWMB_BA+0x10	R/W	PWM Group B Comparator Register 0	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWM Group A Comparator Register 1	0x0000_0000
	PWMB_BA+0x1C	R/W	PWM Group B Comparator Register 1	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM Group A Comparator Register 2	0x0000_0000
	PWMB_BA+0x28	R/W	PWM Group B Comparator Register 2	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM Group A Comparator Register 3	0x0000_0000
	PWMB_BA+0x34	R/W	PWM Group B 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, 23, 45 or 67, depends on selected PWM channel.

		● Duty ratio = (CMR+1)/(CNR+1). ● CMR >= CNR: PWM output is always high. ● CMR < CNR: PWM low width = (CNR-CMR) unit; 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_BA0+0x14	R	PWM Group A Data Register 0	0x0000_0000
	PWMB_BA0+0x14	R	PWM Group B Data Register 0	0x0000_0000
PDR1	PWMA_BA0+0x20	R	PWM Group A Data Register 1	0x0000_0000
	PWMB_BA0+0x20	R	PWM Group B Data Register 1	0x0000_0000
PDR2	PWMA_BA0+0x2C	R	PWM Group A Data Register 2	0x0000_0000
	PWMB_BA0+0x2C	R	PWM Group B Data Register 2	0x0000_0000
PDR3	PWMA_BA0+0x38	R	PWM Group A Data Register 3	0x0000_0000
	PWMB_BA0+0x38	R	PWM Group B 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
	PWMB_BA+0x40	R/W	PWM Group B 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				PWMIE3	PWMIE2	PWMIE1	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
	PWMB_BA+0x44	R/W	PWM Group B 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** Flag is set by hardware when PWM3 down counter reaches zero, software can clear this bit by writing a one to it.
[2]	**PWMIF2**	**PWM channel 2 Interrupt Status** Flag is set by hardware when PWM2 down counter reaches zero, software can clear this bit by writing a one to it.
[1]	**PWMIF1**	**PWM channel 1 Interrupt Status** Flag is set by hardware when PWM1 down counter reaches zero, software can clear this bit by writing a one to it.
[0]	**PWMIF0**	**PWM channel 0 Interrupt Status** Flag is set by hardware when PWM0 down counter reaches zero, software can clear this bit by writing a one to it.

Note: User can clear each interrupt flag by writing an one 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
	PWMB_BA+0x50	R/W	PWM Group B 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	FL_IE1	RL_IE1	INV1
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CFLRI0	CRLRI0	Reserved	CAPIF0	CAPCH0EN	FL_IE0	RL_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. Clear this bit by writing a one to it.
[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. Clear this bit by writing a one to it.
[5]	**Reserved**	Reserved
[20]	**CAPIF1**	**Capture1 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). This flag is clear by software with a write 1 to itself.

[19]	**CAPCH1EN**	**Capture PWM Group Channel 1 transition Enable/Disable** 1 = Enable capture function on PWM group channel 1. 0 = Disable capture function on PWM group channel 1 When Enable, Capture latched the PMW-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**	**PWM Group 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**	**PWM Group 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**	**PWM Group Channel 1 Inverter ON/OFF** 1 = Inverter ON. Reverse the input signal from GPIO before fed to Capture timer 0 = Inverter OFF
[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. Clear this bit by writing a one to it.
[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. Clear this bit by writing a one to it.
[5]	**Reserved**	Reserved
[4]	**CAPIF0**	**Capture0 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). This flag is clear by software with a write 1 to

		itself.
[3]	**CAPCH0EN**	**Capture Channel 0 transition Enable/Disable** 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**	**PWM Group Channel 0 Falling Latch Interrupt Enable ON/OFF** 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**	**PWM Group Channel 0 Rising Latch Interrupt Enable ON/OFF** 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**	**PWM Group Channel 0 Inverter ON/OFF** 1 = Inverter ON. Reverse the input signal from GPIO before fed to Capture timer 0 = Inverter OFF

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
	PWMB_BA+0x54	R/W	PWM Group B 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	FL_IE3	RL_IE3	INV3
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CFLRI2	CRLRI2	Reserved	CAPIF2	CAPCH2EN	FL_IE2	RL_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**	**Capture3 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**	**Capture Channel 3 transition Enable/Disable** 1 = Enable capture function on PWM group channel 3 0 = Disable capture function on PWM group channel 3 When Enable, Capture latched the PMW-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**	**PWM Group 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**	**PWM Group 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**	**PWM Group Channel 3 Inverter ON/OFF** 1 = Inverter ON. Reverse the input signal from GPIO before fed to Capture timer 0 = Inverter OFF
[15:8]	**Reserved**	Reserved
[7]	**CFLRI2**	**CFLR2 Latched Indicator Bit** When PWM group input channel 2 has a falling transition, CFLR2 was latched with the value of PWM down-counter and this bit is set by hardware. **Note:** 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. **Note:** Write 1 to clear this bit to zero.
[5]	**Reserved**	Reserved
[4]	**CAPIF2**	**Capture2 Interrupt Indication Flag** If PWM group channel 2 rising latch interrupt is enabled (CRL_IE2=1), a rising transition occurs at PWM group channel 2 will result in CAPIF2 to high; Similarly, a falling transition will cause CAPIF2 to be set high if PWM group channel 2 falling latch interrupt is enabled (CFL_IE2=1). **Note:** Write 1 to clear this bit to zero.

[3]	**CAPCH2EN**	**Capture Channel 2 transition Enable/Disable** 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**	**PWM Group Channel 2 Falling Latch Interrupt Enable ON/OFF** 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**	**PWM Group Channel 2 Rising Latch Interrupt Enable ON/OFF** 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**	**PWM Group Channel 2 Inverter ON/OFF** 1 = Inverter ON. Reverse the input signal from GPIO before fed to Capture timer 0 = Inverter OFF

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
	PWMB_BA+0x58	R	PWM Group B Capture Rising Latch Register (channel 0)	0x0000_0000
CRLR1	PWMA_BA+0x60	R	PWM Group A Capture Rising Latch Register (channel 1)	0x0000_0000
	PWMB_BA+0x60	R	PWM Group B Capture Rising Latch Register (channel 1)	0x0000_0000
CRLR2	PWMA_BA+0x68	R	PWM Group A Capture Rising Latch Register (channel 2)	0x0000_0000
	PWMB_BA+0x68	R	PWM Group B Capture Rising Latch Register (channel 2)	0x0000_0000
CRLR3	PWMA_BA+0x70	R	PWM Group A Capture Rising Latch Register (channel 3)	0x0000_0000
	PWMB_BA+0x70	R	PWM Group B Capture Rising Latch Register (channel 3)	0x0000_0000

Note: If CPU clock is slower than PWM/Capture clock, a write to CRLRx is not guaranteed.

31	30	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
	PWMB_BA+0x5C	R	PWM Group B Capture Falling Latch Register (channel 0)	0x0000_0000
CFLR1	PWMA_BA+0x64	R	PWM Group A Capture Falling Latch Register (channel 1)	0x0000_0000
	PWMB_BA+0x64	R	PWM Group B Capture Falling Latch Register (channel 1)	0x0000_0000
CFLR2	PWMA_BA+0x6C	R	PWM Group A Capture Falling Latch Register (channel 2)	0x0000_0000
	PWMB_BA+0x6C	R	PWM Group B Capture Falling Latch Register (channel 2)	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM Group A Capture Falling Latch Register (channel 3)	0x0000_0000
	PWMB_BA+0x74	R	PWM Group B Capture Falling Latch Register (channel 3)	0x0000_0000

Note: If CPU clock is slower than PWM/Capture clock, a write to CFLRx is not guaranteed.

31	30	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 01/2/3 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
	PWMB_BA+0x78	R/W	PWM Group B 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	
[3:0]	CAPENR	**Capture Input Enable Register** There are four capture inputs from pad. Bit0~Bit3 are used to control each inputs ON or OFF. 0 = OFF (PWMx multi-function pin input does not affect input capture function.) 1 = ON (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 P2 [0] Bit xx1x ➔ Capture channel 1 is from P2 [1] Bit x1xx ➔ Capture channel 2 is from P2 [2] Bit 1xxx ➔ Capture channel 3 is from P2 [3] Bit 3210 for PWM group B Bit xxx1 ➔ Capture channel 0 is from P2 [4] Bit xx1x ➔ Capture channel 1 is from P2 [5] Bit x1xx ➔ Capture channel 2 is from P2 [6] Bit 1xxx ➔ Capture channel 3 is from P2 [7]

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
	PWMB_BA+0x7C	R/W	PWM Group B 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	
[3]	**PWM3**	**PWM Channel 3 Output Enable Register** 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**	**PWM Channel 2 Output Enable Register** 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**	**PWM Channel 1 Output Enable Register** 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**	**PWM Channel 0 Output Enable Register** 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

6.7 Serial Peripheral Interface (SPI) Controller

6.7.1 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. NuMicro M051™ series 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; it also can be configured as a slave device controlled by an off-chip master device.

6.7.2 Features

- Up to two sets of SPI controller
- Support master or slave mode operation
- 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
- Byte or word Suspend Mode
- Variable output serial clock frequency in master mode
- Support two programmable serial clock frequencies in master mode

6.7.3 SPI Block Diagram

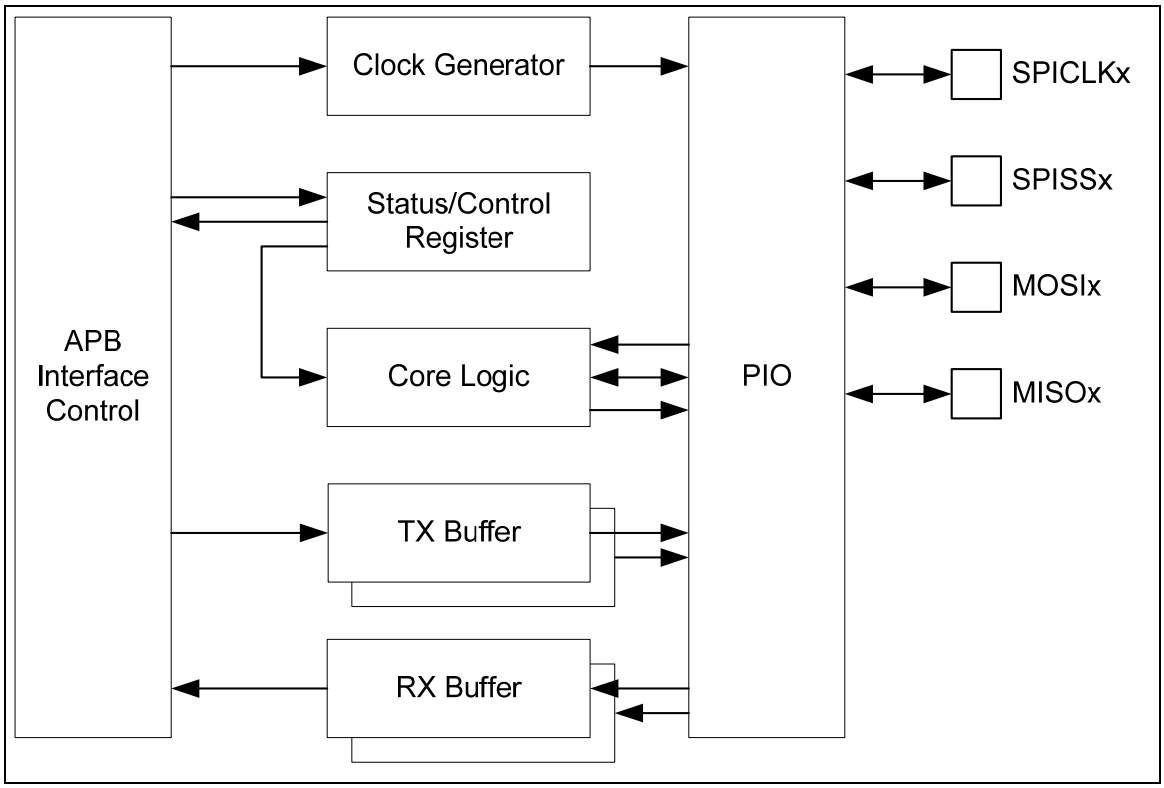


Figure 6.7-1 SPI Block Diagram

6.7.4 SPI Function Descriptions

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 in the both Figure 6.7-2 and Figure 6.7-3.

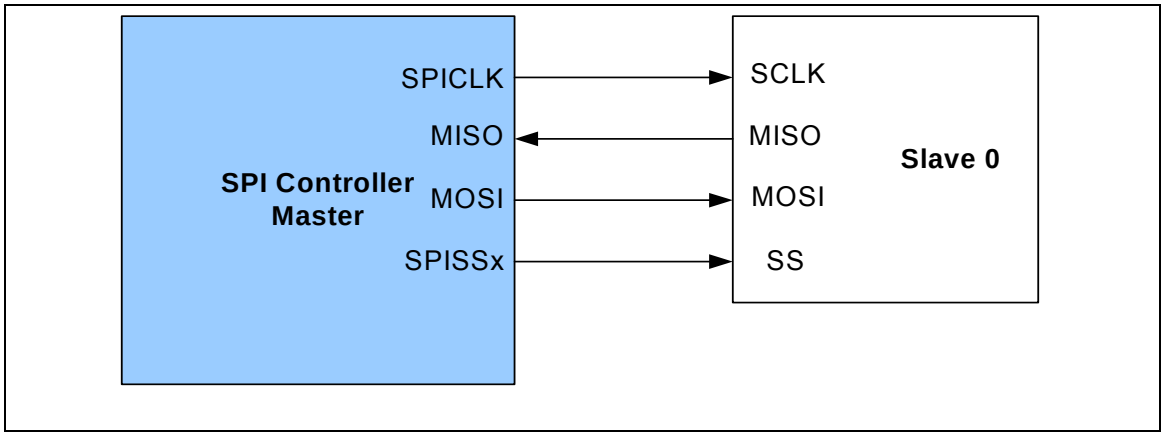


Figure 6.7-2 SPI Master Mode Application Block Diagram

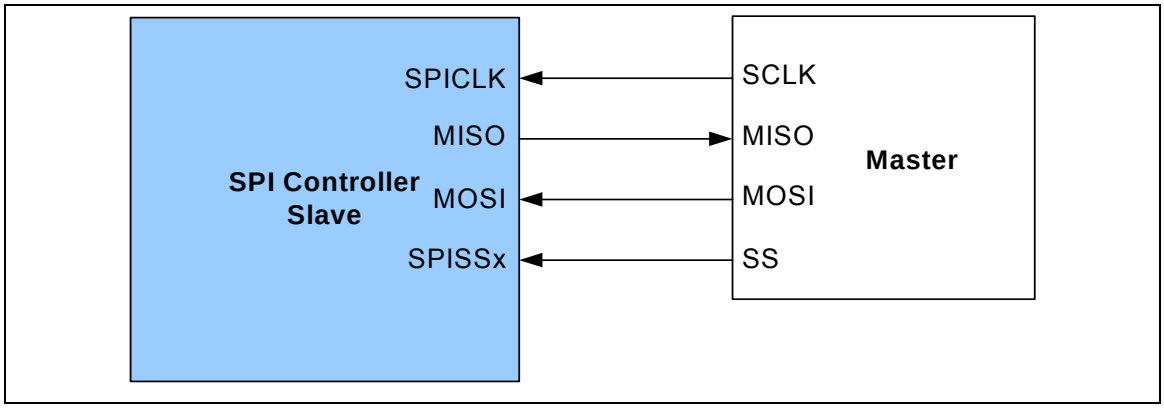


Figure 6.7-3 SPI Slave Mode Application Block Diagram

Slave Select

In master mode, this SPI controller can drive one off-chip slave device through the slave select output pin SPISS. In slave mode, the off-chip master device drives the slave select signal from the SPISS input port to this SPI controller. In master/slave mode, the active level 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]) define the slave select signal SPISS is level trigger or edge trigger. The selection of trigger condition depends on what type of peripheral slave/master device is connected.

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 signal will be generated automatically and output to SPISS pin according to **SSR[0]** (SPI_SSR[0]) whether be enabled or not. It means that the slave select signal, which is enabled in **SSR [0]** register is asserted by the SPI controller when transmit/receive is started by setting the **GO_BUSY** bit (SPI_CNTRL [0]) and is de-asserted after the data transfer is finished. If the AUTOSS bit is cleared, the slave select output signal is asserted and de-asserted by manual setting and clearing the related bit in SPI_SSR [0] register. The active level of the slave select output signal is specified in **SS_LVL** bit (SPI_SSR [2]).

Serial Clock

In master mode, set the DIVIDER (SPI_DIVIDER [15:0]) register to program the output frequency of serial clock to the SPICLK output port. It also supports the variable frequency function if the VARCLK_EN bit (SPI_CNTRL[23]) is enabled, in this case the each bit output frequency of serial clock can be programmed at one frequency of two different frequencies which depend on the DIVIDER and DIVIDER2 (SPI_DIVIDER[31:16]) settings. The decision of the variable frequency for each bit is defined in VARCLK (SPI_VARCLK [31:0]) register. In slave mode, the off-chip master device drives the serial clock through the SPICLK input port to this SPI controller.

Clock Polarity

The CLKP bit (SPI_CNTRL [11]) defines the serial clock idle state in master mode only. If CLKP = 1, the output SPICLK is idle at high state, otherwise it is at low state if CLKP = 0. For variable serial clock, it works in CLKP = 0 only.

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 transfer word for transmitting and receiving.

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 shown below:

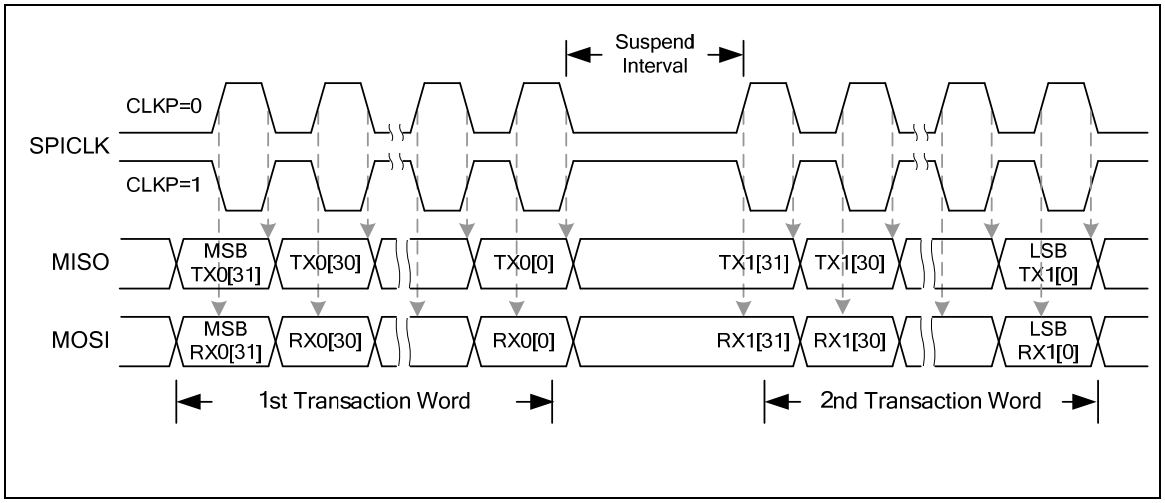


Figure 6.7-4 Two Transactions in One Transfer (Burst Mode)

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.

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 transfer 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 bits 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-bits mode, the data in TX buffer and RX buffer will be rearranged as [unknown byte, BYTE0, BYTE1, BYTE2] and the BYTE0, BYTE1, and BYTE2 will be transmitted/received data step by step in MSB first. The rule of 16-bits mode is the same as above.

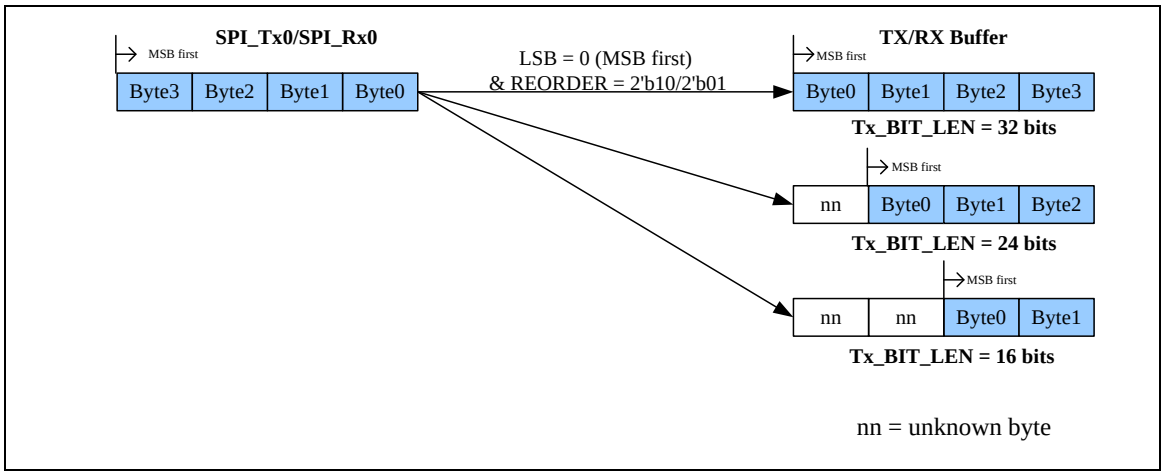


Figure 6.7-5 Byte Reorder

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. The byte suspend setting is the same as the word that using the common bit field of SP_CYCLE register. 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).

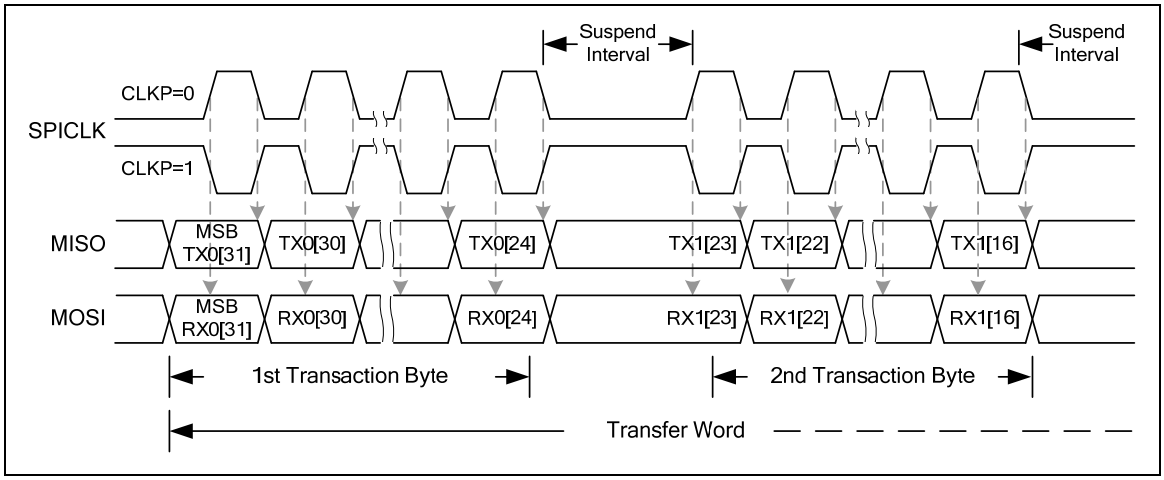


Figure 6.7-6 Timing Waveform for Byte Suspend

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 11-1 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.

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 Figure 6.7-7 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).

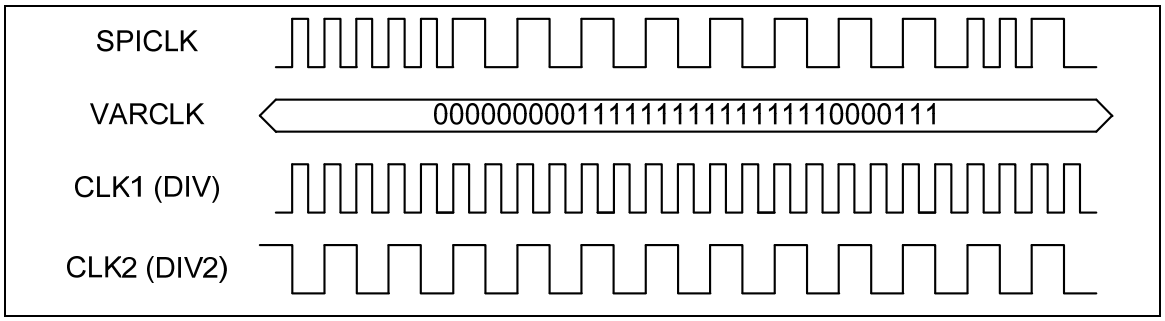


Figure 6.7-7 Variable Serial Clock Frequency

6.7.5 SPI Timing Diagram

In master/slave mode, the active level of device/slave select (SPISS) signal can be programmed to low active or high active in SS_LVL bit (SPI_SSR [2]), but the SPISS0/1 is level trigger or edge trigger which is defined in 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 transfer 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 from Figure 6.7-8 to Figure 6.7-11.

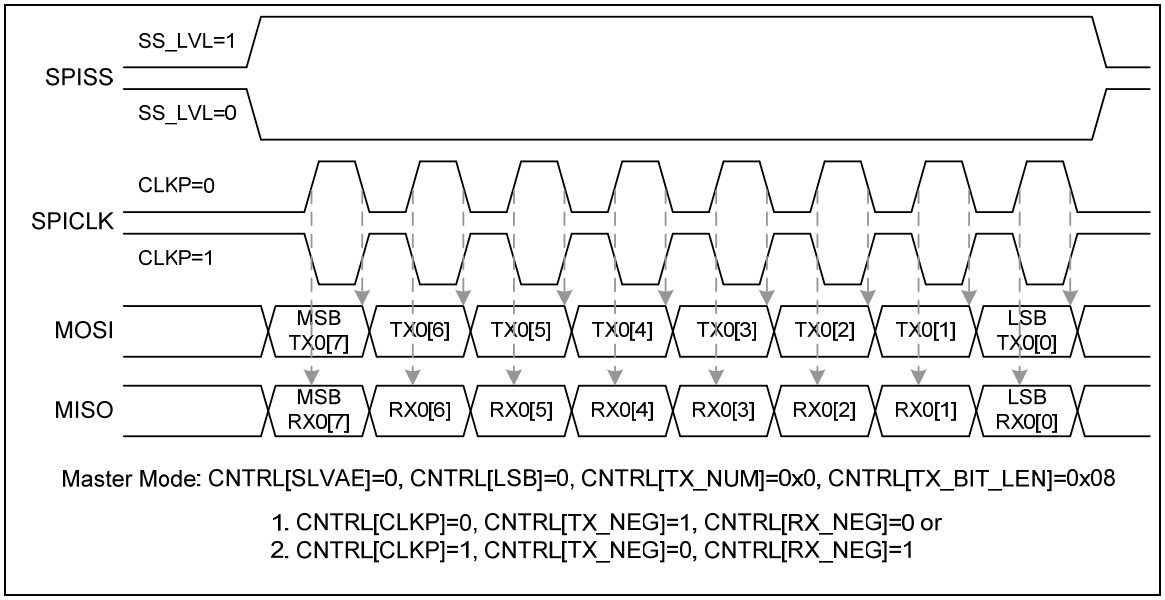


Figure 6.7-8 SPI Timing in Master Mode

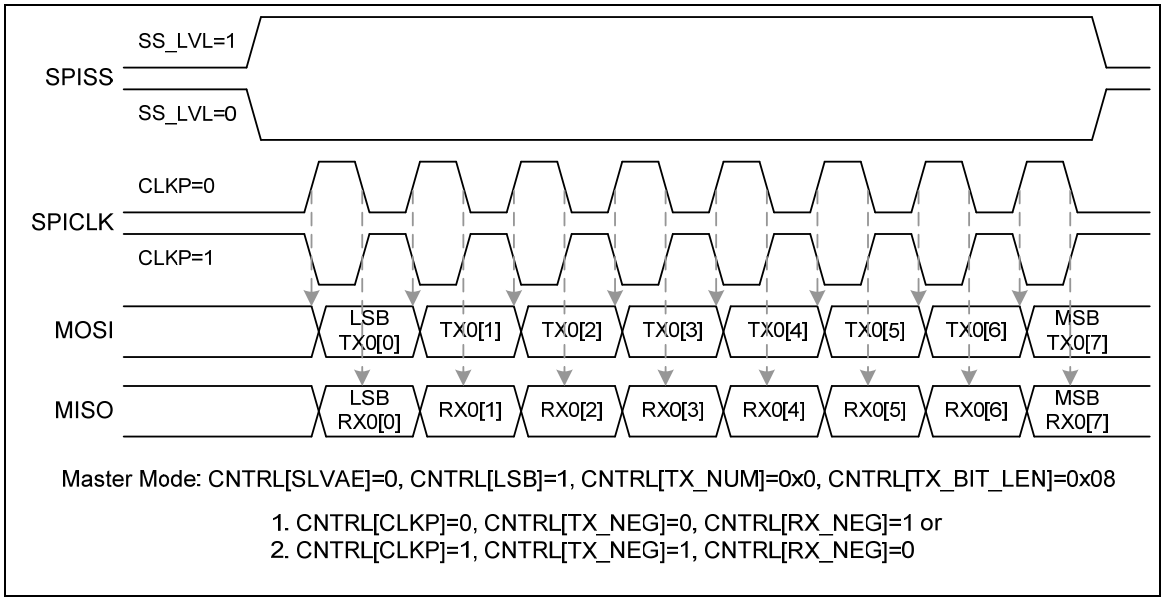


Figure 6.7-9 SPI Timing in Master Mode (Alternate Phase of SPICLK)

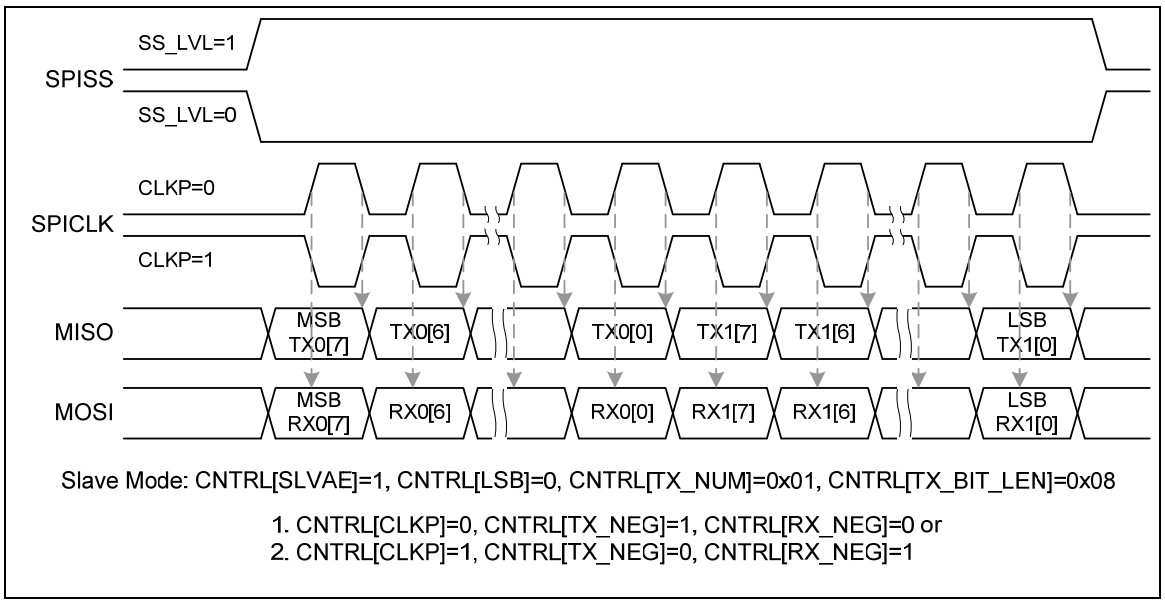


Figure 6.7-10 SPI Timing in Slave Mode

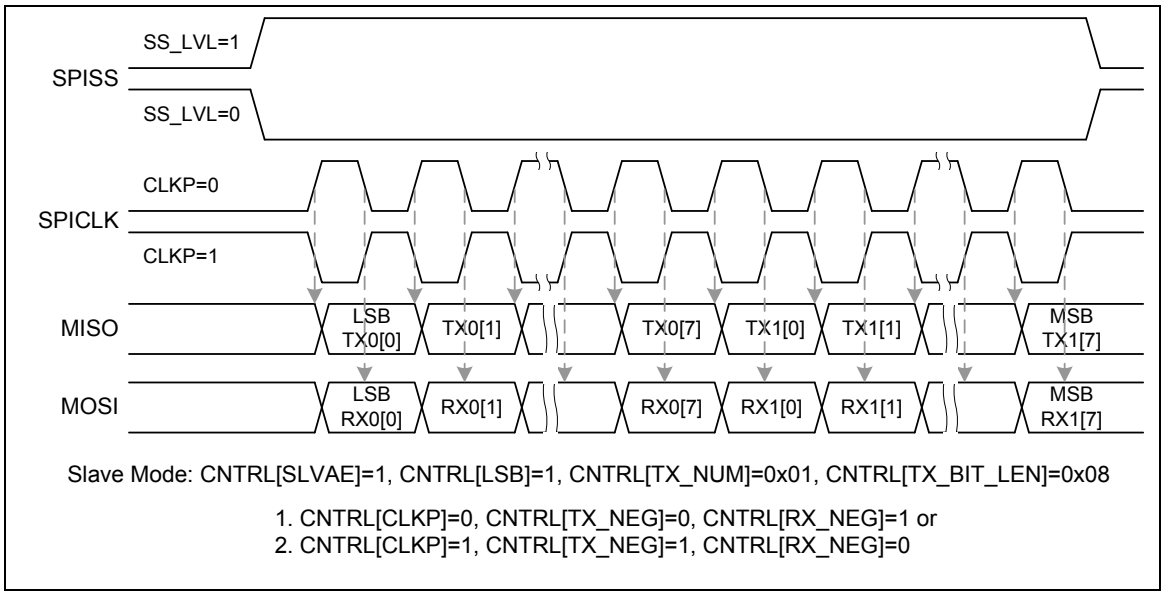


Figure 6.7-11 SPI Timing in Slave Mode (Alternate Phase of SPICLK)

6.7.6 SPI 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 transfer
- Slave select signal is active low

Basically, the specification of the connected off-chip slave device should be referred in details before the following steps:

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 respective 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 bits 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 not 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 be 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, SPI controller is set as a slave device that are controlled by an off-chip master device, and supposes the off-chip master device to access 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 transfer
- Slave select signal is high level trigger

Basically, the specification of the connected off-chip master device should be configured detailed before the following steps

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 bits 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 be cleared to 0 by hardware automatically.

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

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

6.7.7 SPI Controller Registers 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	0x0000_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 1: When software programs CNTRL, the GO_BUSY bit should be written last.

6.7.8 SPI Controller Registers 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	0x0000_0004

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
VARCLK_EN	Reserved		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)** 0 = The serial clock output frequency is fixed and decided only by the value of DIVIDER. 1 = The serial clock output frequency is variable. The output frequency is decided by the value of VARCLK, DIVIDER, and DIVIDER2. Note that when enable this VARCLK_EN bit, the setting of TX_BIT_LEN must be programmed as 0x10 (16 bits mode)
[22:21]	**Reserved**	Reserved
[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 serial clock 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 serial clock cycles) among each byte. The setting of TX_BIT_LEN must be configured as 0x00. (32 bits/word)
[18]	**SLAVE**	**Slave Mode Indication** 0 = Master mode. 1 = Slave mode.
[17]	**IE**	**Interrupt Enable** 0 = Disable SPI/MICROWIRE Interrupt. 1 = Enable SPI/MICROWIRE Interrupt.
[16]	**IF**	**Interrupt Flag** 0 = It indicates that the transfer dose not finish yet. 1 = It indicates that the transfer is done. The interrupt flag is set if it was enable. **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: (SP_CYCLE[3:0] + 2) * period of SPICLK 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** 0 = SPICLK idle low. 1 = SPICLK idle high.
[10]	**LSB**	**LSB First** 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). 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).
[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 word will be executed in one transfer. 10 = Reserved. 11 = Reserved.
[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** 0 = The transmitted data output signal is changed at the rising edge of SPICLK. 1 = The transmitted data output signal is changed at the falling edge of SPICLK.
[1]	**RX_NEG**	**Receive At Negative Edge** 0 = The received data input signal is latched at the rising edge of SPICLK. 1 = The received data input signal is latched at the falling edge of SPICLK.
[0]	**GO_BUSY**	**Go and Busy Status** 0 = Writing 0 to this bit to stop data transfer if SPI is transferring. 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. During the data transfer, this bit keeps the value of 1. As the transfer is finished, this bit will be cleared automatically. NOTE: All registers should be set before writing 1 to this GO_BUSY bit. When a transfer is in progress, writing to any register of the SPI/MICROWIRE master/slave core has no effect.

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 of the system clock, PCLK, to generate the serial clock on the output SPICLK. The desired frequency is obtained according to the following equation: $f_{sclk} = \frac{f_{psclk}}{(DIVIDER2+1)*2}$
[15:0]	**DIVIDER**	**Clock Divider Register** (master only) The value in this field is the frequency divider of the system clock, PCLK, to generate the serial clock on the output SPICLK. The desired frequency is obtained according to the following equation: $f_{sclk} = \frac{f_{psclk}}{(DIVIDER+1)*2}$ 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	Reserved	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 does not meet the specified requirements. Note: This bit is READ only
[4]	**SS_LTRIG**	**Slave Select Level Trigger (Slave only)** 0 = The input slave select signal is edge-trigger. This is the default value. 1 = The slave select signal will be level-trigger. It depends on SS_LVL to decide the signal is active low or active high.
[3]	**AUTOSS**	**Automatic Slave Select (Master only)** 0 = If this bit is cleared, slave select signal is asserted and de-asserted by setting and clearing SSR[0].

		1 = If this bit is set, SPISS0/1 signal is generated automatically. It means that device/slave select signal, which is set in SSR[0] register is asserted by the SPI controller when transmit/receive is started by setting GO_BUSY, and is de-asserted after each transmit/receive is finished.
[2]	**SS_LVL**	**Slave Select Active Level** It defines the active level of slave select signal (SPISS0/1). 0 = The slave select signal SPISS0/1 is active at low-level/falling-edge. 1 = The slave select signal SPISS0/1 is active at high-level/rising-edge.
[1]	Reserved	Reserved
[0]	**SSR**	**Slave Select Register (Master only)** If AUTOSS bit is cleared, writing 1 to this bit sets the SPISSx line to an active state and writing 0 sets the line back to inactive state. If AUTOSS bit is set, writing 1 to this bit will select the SPISSx 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 level of SPISSx is specified in SS_LVL). **Note:** SPISSx is always defined as device/slave select input signal 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. **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 core will perform two 32-bit transmit/receive successive using the same setting (the order is TX0[31:0], 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	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** 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 Figure 6.7-7 for Variable Clock timing diagram. Note: It is used for CLKP = 0 only.

6.8 Timer Controller

6.8.1 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, interval measurement, clock generation, delay timing, and so on. The timer can generates an interrupt signal upon timeout, or provide the current value of count during operation.

6.8.2 Features:

- Provides four channels of 32-bit timers with one 8-bit pre-scale counter with four 24-bit up-timer
- Independent clock source for each timer.
- 24-bit timer value is readable through TDR (Timer Data Register)
- Provides one-shot, periodic and toggle operation modes.

6.8.3 Timer Controller 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. Refer to Figure 6.8-1 for the timer controller block diagram. There are three options of clock sources for each channel. Figure 6.8-2 illustrates the clock source control function. Software can program the 8-bit pre-scale counter to decide the clock period to 24-bit up timer.

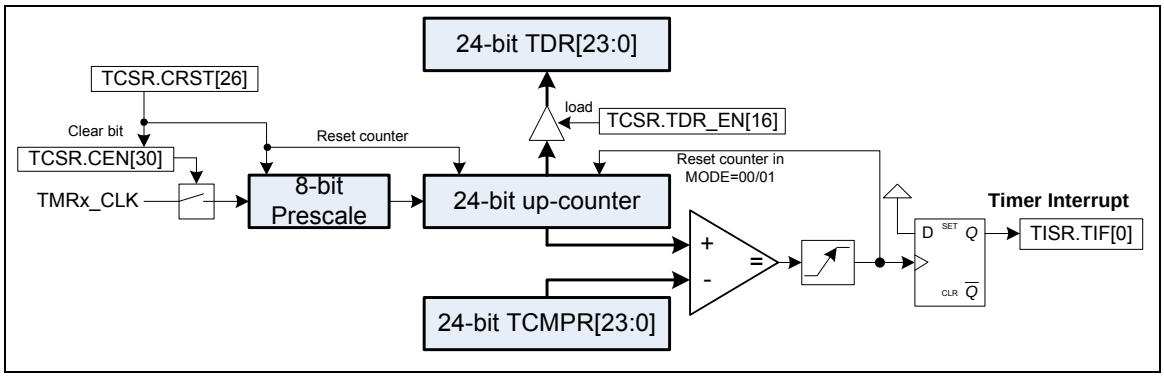


Figure 6.8-1 Timer Controller Block Diagram

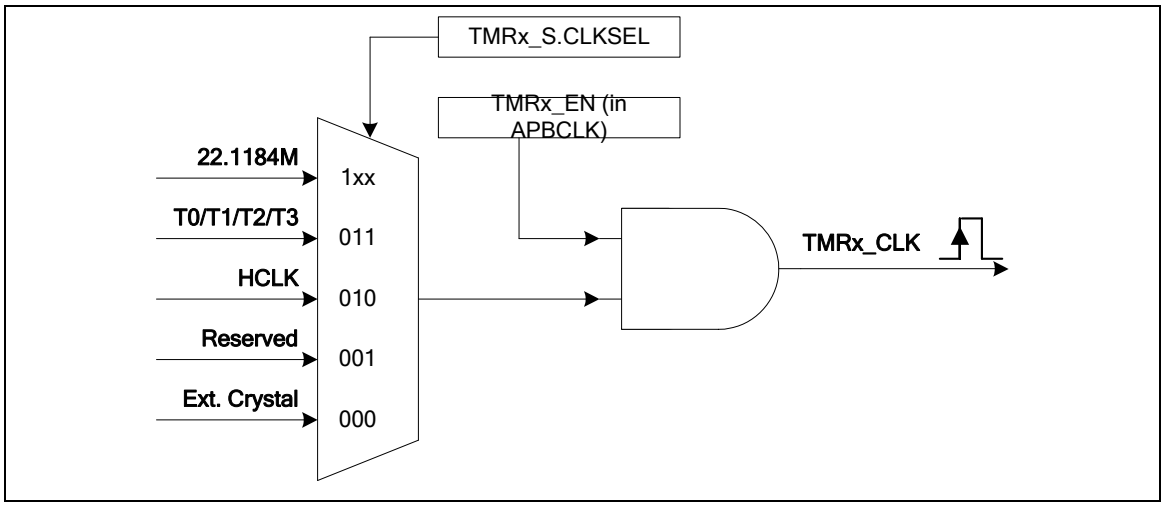


Figure 6.8-2 Clock Source of Timer Controller

6.8.4 Timer Operation Mode

Timer controller provides one-shot, period and toggle modes operation. Each operating function mode is shown as following:

6.8.4.1 One –Shot mode

If timer is operated at one-shot mode and CEN (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 (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. It indicates that the timer counting overflow happens. If IE (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.

6.8.4.2 Periodic mode

If timer is operated at period mode and CEN (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 (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. It indicates that the timer counting overflow happens. If IE (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.

6.8.4.3 Toggle mode

If timer is operated at toggle mode and CEN (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 (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. 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.

6.8.5 Timer Controller Registers 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+00	R/W	Timer0 Control and Status Register	0x0000_0005
TCMPR0	TMR_BA01+04	R/W	Timer0 Compare Register	0x0000_0000
TISR0	TMR_BA01+08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TDR0	TMR_BA01+0C	R	Timer0 Data Register	0x0000_0000
TCSR1	TMR_BA01+20	R/W	Timer1 Control and Status Register	0x0000_0005
TCMPR1	TMR_BA01+24	R/W	Timer1 Compare Register	0x0000_0000
TISR1	TMR_BA01+28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TDR1	TMR_BA01+2C	R	Timer1 Data Register	0x0000_0000
TCSR2	TMR_BA23+00	R/W	Timer2 Control and Status Register	0x0000_0005
TCMPR2	TMR_BA23+04	R/W	Timer2 Compare Register	0x0000_0000
TISR2	TMR_BA23+08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TDR2	TMR_BA23+0C	R	Timer2 Data Register	0x0000_0000
TCSR3	TMR_BA23+20	R/W	Timer3 Control and Status Register	0x0000_0005
TCMPR3	TMR_BA23+24	R/W	Timer3 Compare Register	0x0000_0000
TISR3	TMR_BA23+28	R/W	Timer3 Interrupt Status Register	0x0000_0000
TDR3	TMR_BA23+2C	R	Timer3 Data Register	0x0000_0000

Timer Control Register (TCSR)

Register	Offset	R/W	Description	Reset Value
TCSR0	TMR_BA01+000	R/W	Timer0 Control and Status Register	0x0000_0005
TCSR1	TMR_BA01+020	R/W	Timer1 Control and Status Register	0x0000_0005
TCSR2	TMR_BA23+000	R/W	Timer2 Control and Status Register	0x0000_0005
TCSR3	TMR_BA23+020	R/W	Timer3 Control and Status Register	0x0000_0005

31	30	29	28	27	26	25	24
Reserved	CEN	IE	MODE[1:0]		CRST	CACT	Reserved
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]	**Reserved**	Reserved
[30]	**CEN**	**Timer Enable Bit** 0 = Stops/Suspends counting 1 = Starts counting **Note1**: In stop status, and then set CEN to 1 will enables the 24-bit 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** 0 = Disable timer Interrupt 1 = Enable timer Interrupt If timer interrupt is enabled, the timer asserts its interrupt signal when the associated timer is equal to TCMPR.
[28:27]	**MODE**	**Timer Operating Mode**

		MODE / **Timer Operating Mode** 00 — 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. 01 — The timer is operating in the periodic mode. The associated interrupt signal is generated periodically (if IE is enabled). 10 — 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. 11 — Reserved
[26]	**CRST**	**Timer Reset Bit** Set this bit will reset the timer, prescale and also force CEN to 0. 0 = No effect 1 = Reset Timer's prescale counter, internal 24-bit up-timer and CEN bit
[25]	**CACT**	**Timer Active Status Bit (Read only)** This bit indicates the status of timer. 0 = Timer is **not** active 1 = Timer is **in** active
[24:17]	**Reserved**	Reserved
[16]	**TDR_EN**	**Data Load Enable** When TDR_EN is set, TDR (Timer Data Register) will be updated continuously with the 24-bit up-timer value. 1 = **Timer Data Register** update enable 0 = **Timer Data Register** update disable
[15:8]	**Reserved**	Reserved
[7:0]	**PRESCALE**	**Pre-scale Counter** Clock input is divided by PRESCALE+1 before it is fed to the timer. If PRESCALE =0, then there is no scaling.

Timer Compare Register (TCMPR)

Register	Offset	R/W	Description	Reset Value
TCMPR0	TMR_BA01+004	R/W	Timer0 Compare Register	0x0000_0000
TCMPR1	TMR_BA01+024	R/W	Timer1 Compare Register	0x0000_0000
TCMPR2	TMR_BA23+004	R/W	Timer2 Compare Register	0x0000_0000
TCMPR3	TMR_BA23+024	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:** No matter CEN is 0 or 1, whenever software write a new value into this register, TIMER will restart counting using this new value and abort previous count.

Timer Interrupt Status Register (TISR)

Register	Offset	R/W	Description	Reset Value
TISR0	TMR_BA01+08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TISR1	TMR_BA01+28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TISR2	TMR_BA23+08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TISR3	TMR_BA23+28	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+0C	R/W	Timer0 Data Register	0x0000_0000
TDR1	TMR_BA01+2C	R/W	Timer1 Data Register	0x0000_0000
TDR2	TMR_BA23+0C	R/W	Timer2 Data Register	0x0000_0000
TDR3	TMR_BA23+2C	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.

6.9 Watchdog Timer (WDT)

6.9.1 Overview

The purpose of Watchdog Timer is to perform a system reset when system runs into an unknown state. This prevents system from hanging for an infinite period of time. Besides, this Watchdog Timer supports another function to wakeup CPU from power-down mode. The watchdog timer includes a 18-bit free running counter with programmable time-out intervals. Table 6.9-1 shows the watchdog timeout interval selection and Figure 6.9-1 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 CPU 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 CPU. This reset will last 63 WDT clocks (T_{RST}) then CPU restarts executing program from reset vector (0x0000 0000). WTRF will not be cleared by Watchdog reset. User may poll WTFR by software to recognize the reset source. WDT also provides wakeup function. When chip is powered down and the Watchdog Timer Wakeup Function Enable bit (WDTR[4]) is set, if the WDT counter has not been cleared after the specific delay time expires, the chip will be waken up from power down state.

WTIS	Timeout Interval Selection T_{TIS}	Interrupt Period T_{INT}	WTR Timeout Interval (WDT_CLK=12 MHz) Min. T_{WTR} ~ Max. T_{WTR}
000	2^4 * T_{WDT}	1024 * T_{WDT}	1.33 us ~ 86.67 us
001	2^6 * T_{WDT}	1024 * T_{WDT}	5.33 us ~ 90.67 us
010	2^8 * T_{WDT}	1024 * T_{WDT}	21.33 us ~ 106.67 us
011	2^{10} * T_{WDT}	1024 * T_{WDT}	85.33 us ~ 170.67 us
100	2^{12} * T_{WDT}	1024 * T_{WDT}	341.33 us ~ 426.67 us
101	2^{14} * T_{WDT}	1024 * T_{WDT}	1.36 ms ~ 1.45 ms
110	2^{16} * T_{WDT}	1024 * T_{WDT}	5.46 ms ~ 5.55 ms
111	2^{18} * T_{WDT}	1024 * T_{WDT}	21.84 ms ~ 21.93 ms

Table 6.9-1 Watchdog Timeout Interval Selection

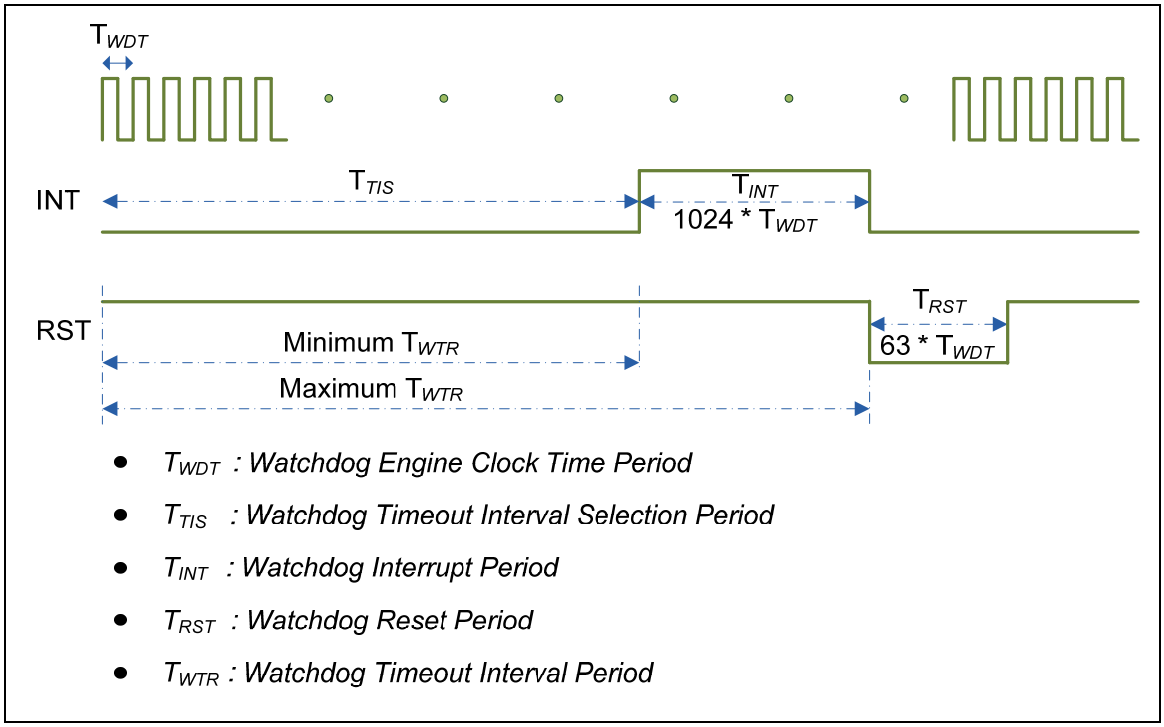


Figure 6.9-1 Timing of Interrupt and Reset Signal

6.9.2 Features

- 18-bit free running counter to avoid CPU from Watchdog timer reset before the delay time expires.
- Selectable time-out interval (2^4 ~ 2^18) and the time out interval is 86.67 us ~ 21.93 ms (if WDT_CLK = 12 MHz).
- Reset period = WDT_CLK * 63, if WDT_CLK = 12 MHz.

6.9.3 WDT Block Diagram

The Watchdog Timer clock control and block diagram is shown in the Figure 6.9-2.

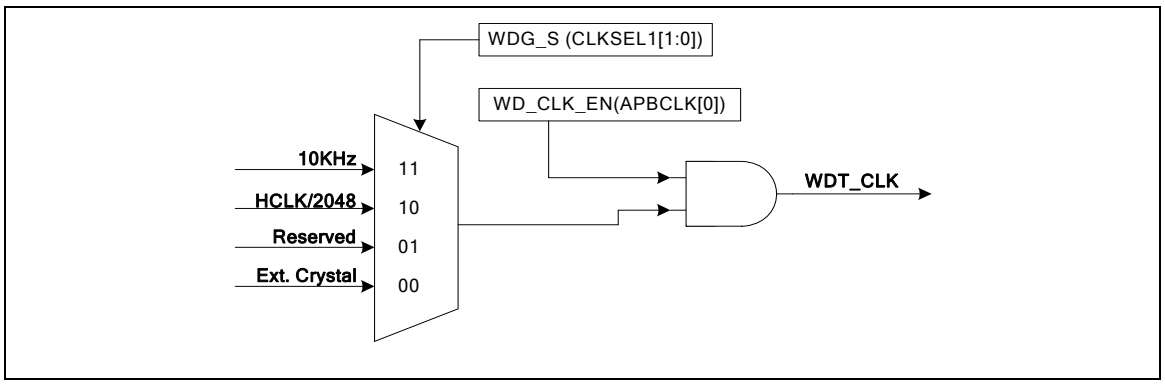


Figure 6.9-2 Watchdog Timer Clock Control

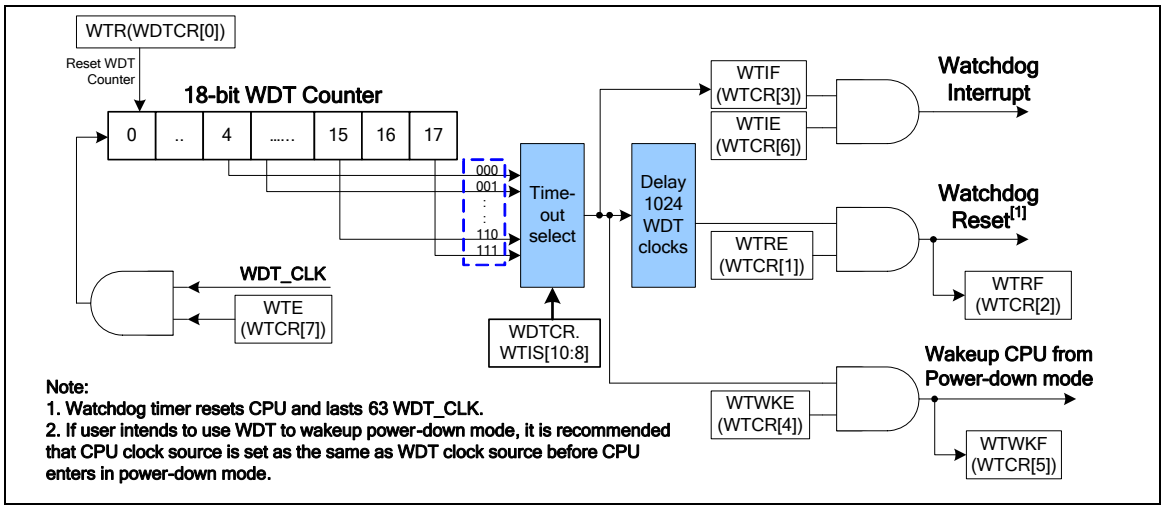


Figure 6.9-3 Watchdog Timer Block Diagram

6.9.4 WDT Controller Registers 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+00	R/W	Watchdog Timer Control Register	0x0000_0700

Watchdog Timer Control Register (WTCR)

Register	Offset	R/W	Description	Reset Value
WTCR	WDT_BA+000	R/W	Watchdog Timer Control Register	0x0000_0700

Note: All bits in this register are write-protected. To program it needs an open lock sequence, by sequentially writing “59h”, “16h”, and “88h” to register REGWRPROT at address GCR_BA + **0x100**

31	30	29	28	27	26	25	24
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	
[31:11]	**Reserved**	Reserved
[10:8]	**WTIS**	**Watchdog Timer Interval Select** (write protection bits) These three bits select the timeout interval for the Watchdog timer.

WTIS	Timeout Interval Selection	Interrupt Period	WTR Timeout Interval (WDT_CLK=12 MHz)
000	$2^4 * T_{WDT}$	$(2^4 + 1024) * T_{WDT}$	1.33 us ~ 86.67 us

		001	2^6 * T_{WDT}	(2^6 + 1024) * T_{WDT}	5.33 us ~ 90.67 us
		010	2^8 * T_{WDT}	(2^8 + 1024) * T_{WDT}	21.33 us ~ 106.67 us
		011	2^{10} * T_{WDT}	(2^{10} + 1024) * T_{WDT}	85.33 us ~ 170.67 us
		100	2^{12} * T_{WDT}	(2^{12} + 1024) * T_{WDT}	341.33 us ~ 426.67 us
		101	2^{14} * T_{WDT}	(2^{14} + 1024) * T_{WDT}	1.36 ms ~ 1.45 ms
		110	2^{16} * T_{WDT}	(2^{16} + 1024) * T_{WDT}	5.46 ms ~ 5.55 ms
		111	2^{18} * T_{WDT}	(2^{18} + 1024) * T_{WDT}	21.84 ms ~ 21.93 ms
[7]	**WTE**	**Watchdog Timer Enable** (write protection bits) 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 bits) 0 = Disable the Watchdog timer interrupt 1 = Enable the Watchdog timer interrupt			
[5]	**WTWKF**	**Watchdog Timer Wakeup Flag** If Watchdog timer causes CPU wakes up from power-down mode, this bit will be set to high. 0 = Watchdog timer does not cause CPU wakeup. 1 = CPU wake up from sleep or power-down mode by Watchdog timeout. **NOTE**: Write 1 to clear this bit to zero.			
[4]	**WTWKE**	**Watchdog Timer Wakeup Function Enable bit** 0 = Disable Watchdog timer Wakeup CPU function. 1 = Enable the Wakeup function that Watchdog timer timeout can wake up CPU from power-down mode.			
[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**: Write 1 to clear this bit to zero.			
[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: Write 1 to clear this bit to zero.
[1]	**WTRE**	**Watchdog Timer Reset Enable** 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** 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**: Write 1 to clear this bit to zero.

6.10 UART Interface Controller

NuMicro M051™ series provides up to two channels of Universal Asynchronous Receiver/Transmitters (UART). UART0~1 performs Normal Speed UART, and support flow control function.

6.10.1 Overview

The Universal Asynchronous Receiver/Transmitter (UART) performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the CPU. The UART controller also supports IrDA SIR Function, and RS-485 mode functions. Each UART channel supports five 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), and MODEM/Wakeup status interrupt (INT_MODEM). Interrupt number 12 (vector number is 28) supports UART0 interrupt. Interrupt number 13 (vector number is 29) supports UART1 interrupt. Refer to Nested Vectored Interrupt Controller chapter for System Interrupt Map.

The UART0~1 are equipped 15-bytes transmitter FIFO (TX_FIFO) and 15-bytes 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 3 error conditions (parity error, framing error, and break interrupt) 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). The Table 6.10-1 and Table 6.10-2 list 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 6.10-1 UART Baud Rate Equation

System clock = 22.1184 MHz			
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 6.10-2 UART Baud Rate Setting Table

The UART0 and UART1 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[1:0]) 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 10ms transfer

delay between transmission and reception. This delay feature must be implemented by software.

Another alternate function of UART controllers is RS-485 9 bit mode function, and direction control provided by RTS pin or can program GPIO (P0.3 for RTS0 and P0.1 for RTS 1) 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.

6.10.2 Features

- Full duplex, asynchronous communications
- Separate receive / transmit 15 bytes (UART0/UART1) entry FIFO for data payloads
- Support hardware auto flow control/flow control function (CTS, RTS) and programmable RTS flow control trigger level (UART0 and UART1 support)
- Programmable receiver buffer trigger level
- Support programmable baud-rate generator for each channel individually
- Support CTS wake up function (UART0 and UART1 support)
- Support 7 bit 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, and parity error detect function
- Fully programmable serial-interface characteristics
- Programmable number of data bit, 5, 6, 7, 8 bit character
- Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
- Programmable stop bit, 1, 1.5, or 2 stop bit generation
- Support IrDA SIR function mode
- Support for 3/16 bit duration for normal mode
- Support RS-485 function mode.
- Support RS-485 9bit mode
- Support hardware or software direct enable control provided by RTS pin

6.10.3 UART Block Diagram

The UART clock control and block diagram are shown in the both Figure 6.10-1 and Figure 6.10-2.

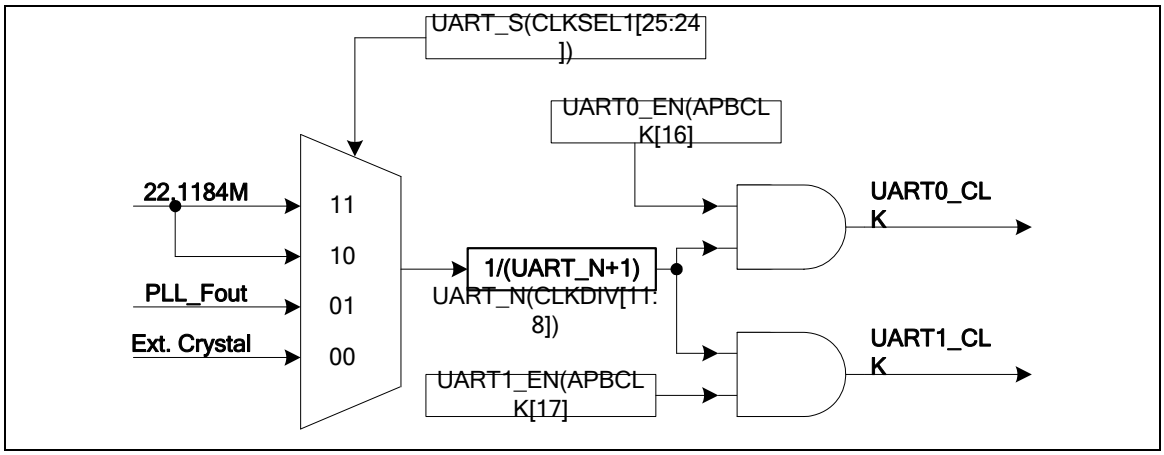


Figure 6.10-1 UART Clock Control Diagram

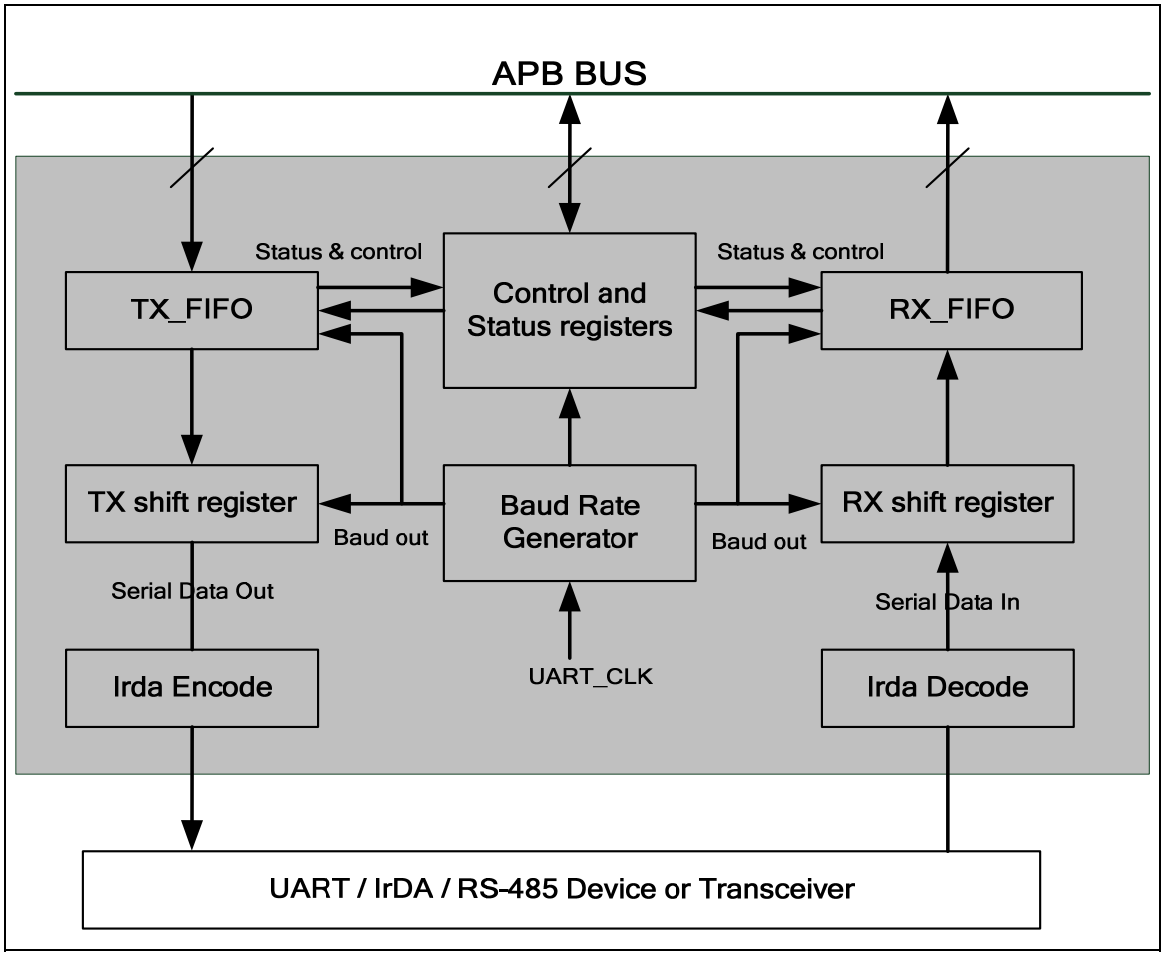


Figure 6.10-2 UART Block Diagram

TX_FIFO
The transmitter is buffered with a 15 bytes FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO
The receiver is buffered with a 15 bytes 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/Wakeup status interrupt (INT_MODEM), and Buffer error interrupt (INT_BUF_ERR).

The Figure 6.10-3 demonstrates the auto-flow control block diagram.

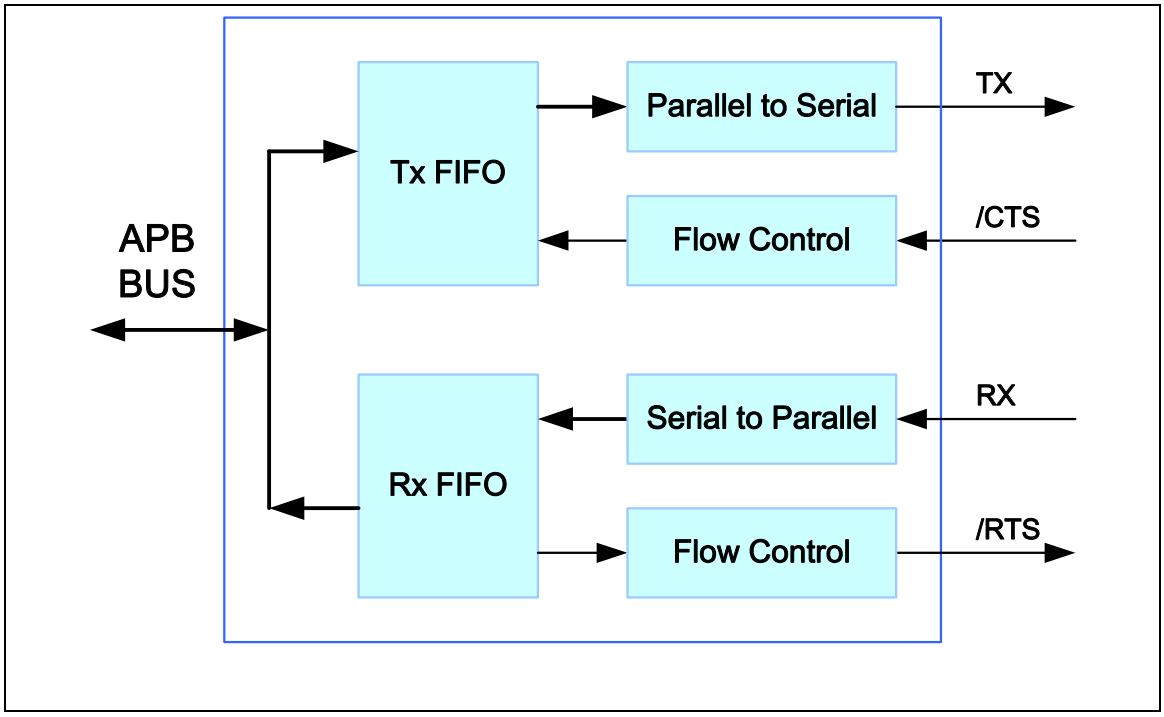


Figure 6.10-3 Auto Flow Control Block Diagram

6.10.4 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 Figure 6.10-4 demonstrates the IrDA control block diagram.

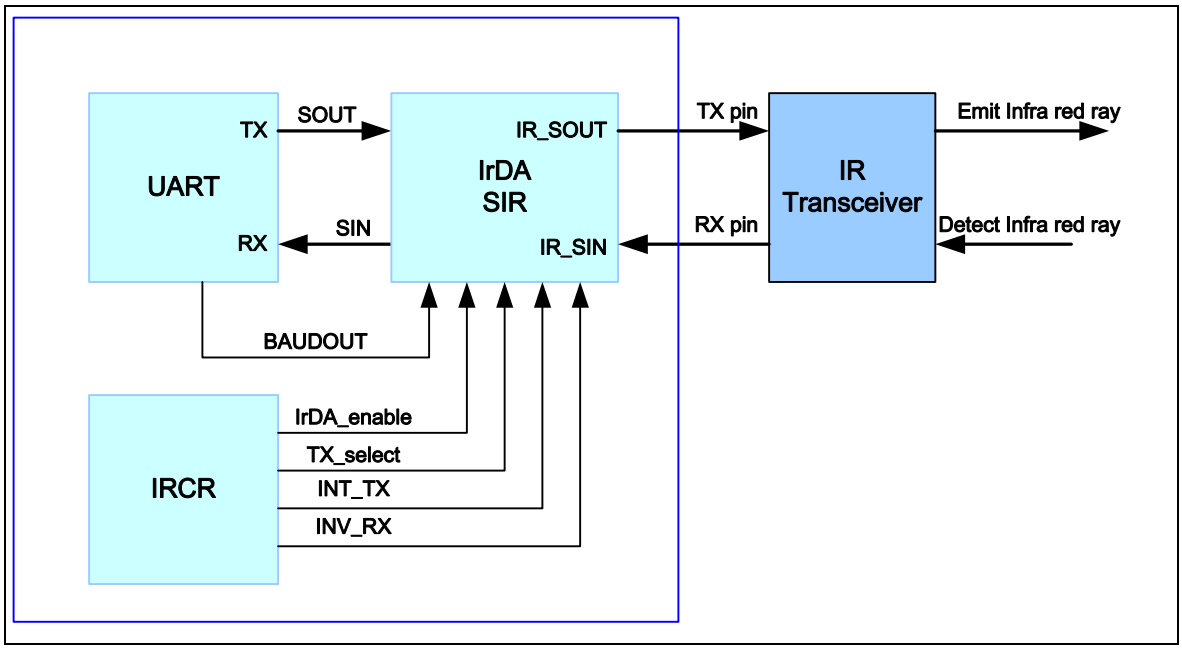


Figure 6.10-4 IrDA Block Diagram

6.10.4.1 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.

6.10.4.2 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

6.10.4.3 IrDA SIR Operation

The IrDA SIR Encoder/decoder provides functionality which converts between UART data stream and half duplex serial SIR interface. The Figure 6.10-5 is IrDA encoder/decoder waveform:

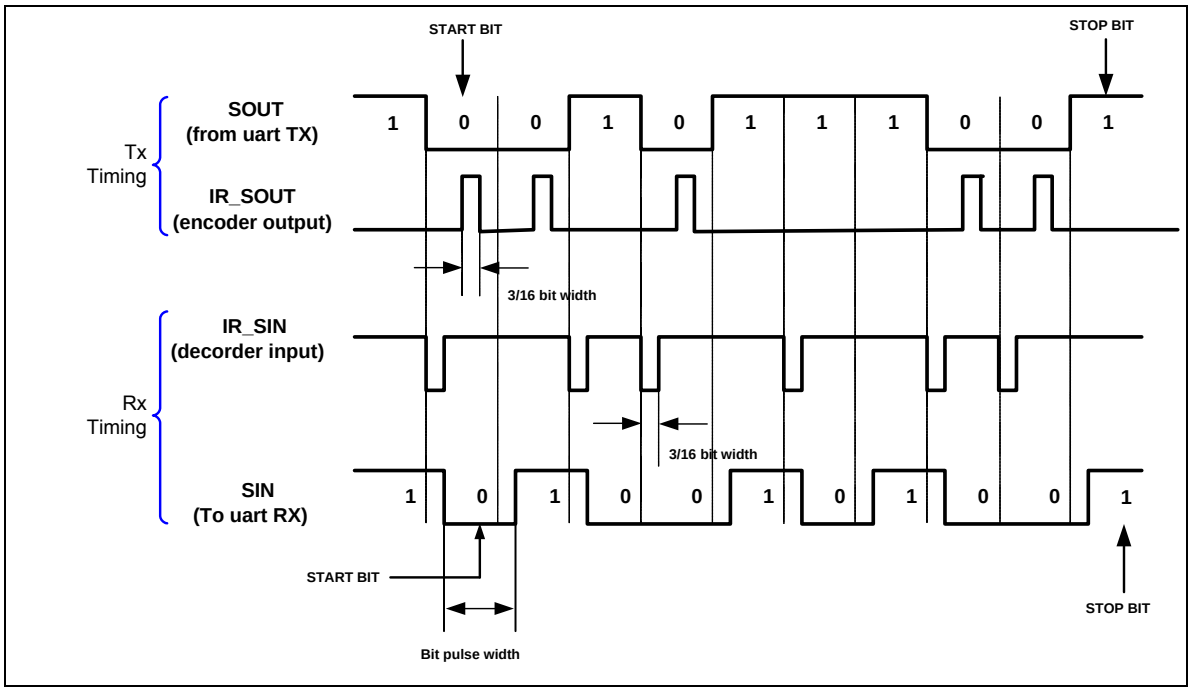


Figure 6.10-5 IrDA TX/RX Timing Diagram

6.10.5 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 mode, the controller can configuration of it as an RS-485 addressable slave and the RS-485 master transmitter will identify an address character by setting the parity (9^{th} bit) to 1. For data characters, the parity is set to 0. Software can use UA_LCR register to control the 9-th bit (When the PBE , EPE and SPE are set, the 9-th bit is transmitted 0 and when PBE and SPE are set and EPE is cleared, the 9-th bit is transmitted 1). The Controller support three operation mode that is RS-485 Normal Multidrop Operation Mode (NMM), RS-485 Auto Address Detection Operation Mode (AAD) and RS-485 Auto Direction Control Operation Mode (AUD), software can choose any operation mode by programming UA_RS-485_CSR register, and software can driving the transfer delay time between the last stop bit leaving the TX-FIFO and the de-assertion of by setting UA_TOR [DLY] register.

RS-485 Normal Multidrop Operation Mode (NMM)

In RS-485 Normal Multidrop operation mode, in first, software must decided the data which before the address byte be detected will be stored in RX-FIFO or not. If software want to ignore any data before address byte detected, the flow is set UART_FCR[RS485_RX_DIS] then enable UA_RS-485[RS485_NMM] and the receiver will ignore any data until an address byte is detected (bit9 =1) and the address byte data will be stored in the RX-FIFO. If software wants to receive any data before address byte detected, the flow is disable UART_FCR [RS485_RX_DIS] then enable UA_RS-485[RS485_NMM] and the receiver will received any data. If an address byte is detected (bit9 =1), it will generator an interrupt to CPU and software can decide whether enable or disable receiver to accept the following data byte by setting UA_RS-485_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_RS-485_FCR [RX_DIS] register, when a next address byte be detected, the controller will clear the UA_RS-485_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_RS-485[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 address doesn't match the UA_RS-485[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 setting 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.

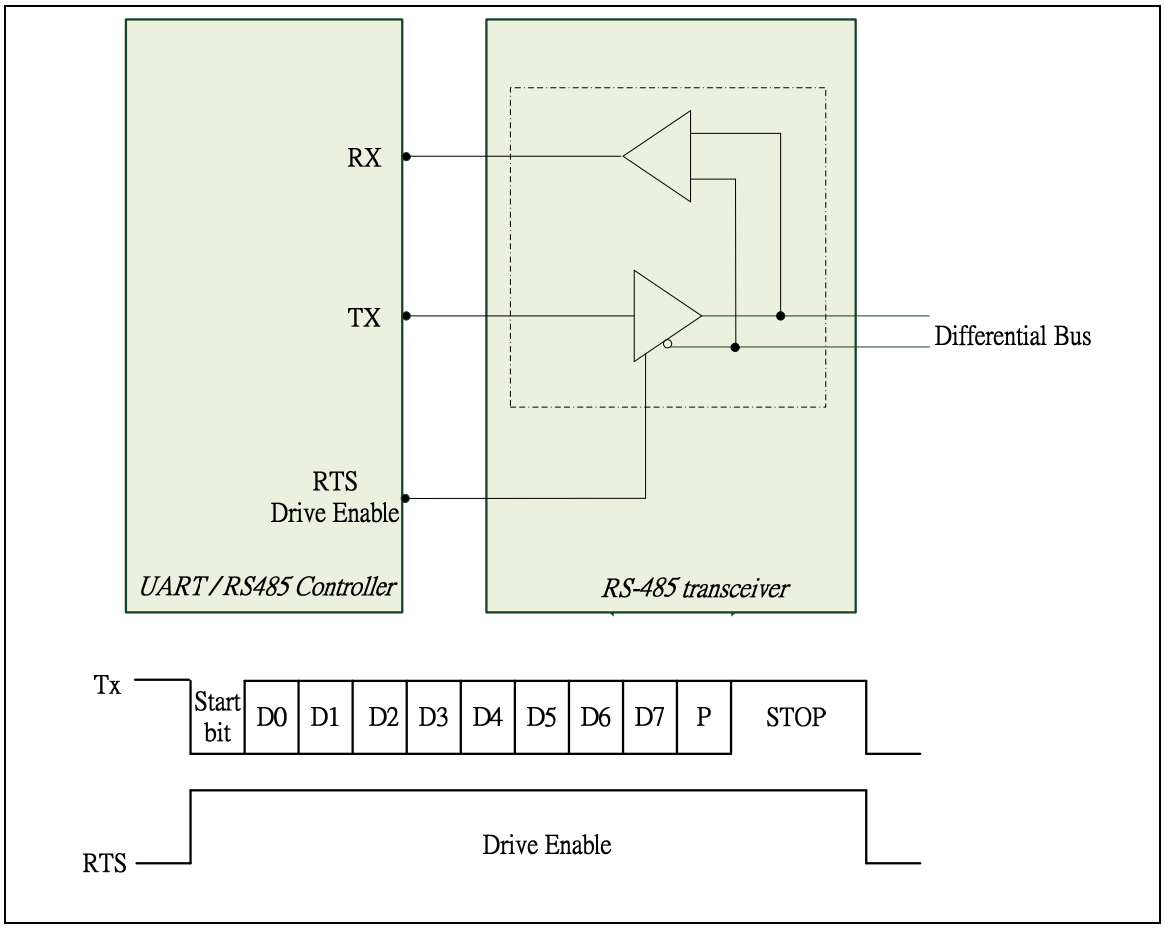


Figure 6.10-6 Structure of RS-485 Frame

6.10.6 UART Interface Controller Registers 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 = 0x4005_0000** **Channel1 : UART1_BA = 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_0101
	UART1_BA+0x08	R/W	UART1 FIFO Control Register	0x0000_0101
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_0001
	UART1_BA+0x14	R/W	UART1 Modem Status Register	0x0000_0001
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_000A
	UART1_BA+0x1C	R/W	UART1 Interrupt Status Register	0x0000_000A
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

6.10.7 UART Interface Controller Registers 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	Reserved	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**	**Wake Up CPU Function Enable**

		0 = Disable UART wake up CPU function 1 = Enable wake up function, when the system is in deep sleep mode, an external /CTS change will wake up CPU from deep sleep mode.
[5]	**Reserved**	Reserved
[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_0101
	UART1_BA+0x08	R/W	UART1 FIFO Control Register	0x0000_0101

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

Bits	Descriptions	
[31:20]	**Reserved**	Reserved
[19:16]	**RTS_TRI_LEV**	**RTS Trigger Level for Auto-flow Control Use** RTS_TRI_LEV / Trigger Level (Bytes) 0000 / 01 0001 / 04 0010 / 08 others / 14 **Note:** This field is used for auto RTS flow control.
[15:9]	**Reserved**	Reserved
[8]	**RX_DIS**	**Receiver Disable register.** The receiver is disabled or not (set 1 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.

<table>
<tr><td>[7:4]</td><td>RFITL</td><td>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>others</td><td>14</td></tr></table></td></tr>
<tr><td>[3]</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>[2]</td><td>TFR</td><td>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.</td></tr>
<tr><td>[1]</td><td>RFR</td><td>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.</td></tr>
<tr><td>[0]</td><td>Reserved</td><td>Reserved</td></tr>
</table>

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** 0 = Disable stick parity 1 = When bits PBE , EPE and SPE are set, the parity bit is transmitted and checked as cleared. When PBE and SPE are set and EPE is cleared, the parity bit is transmitted and checked as set.
[4]	**EPE**	**Even Parity Enable** 0 = Odd number of logic 1's are transmitted or checked in the data word and parity bits. 1 = Even number of logic 1's are transmitted or checked in the data word and parity bits. This bit has effect only when bit 3 (parity bit enable) is set.
[3]	**PBE**	**Parity Bit Enable** 0 = Parity bit is not generated (transmit data) or checked (receive data) during transfer.

<table>
<tr><td></td><td></td><td>1 = Parity bit is generated or checked between the last data bit and the stop bit.</td></tr>
<tr><td>[2]</td><td>NSB</td><td>Number of “STOP bit”
0= One “ STOP bit” is generated in the transmitted data
1= One and a half “ STOP bit” is generated in the transmitted data when 5-bit word length is selected;
Two “STOP bit” is generated when 6-, 7- and 8-bit word length is selected.</td></tr>
<tr><td>[1:0]</td><td>WLS</td><td>Word Length Select
<table>
<tr><th>WLS[1:0]</th><th>Character length</th></tr>
<tr><td>00</td><td>5 bits</td></tr>
<tr><td>01</td><td>6 bits</td></tr>
<tr><td>10</td><td>7 bits</td></tr>
<tr><td>11</td><td>8 bits</td></tr>
</table></td></tr>
</table>

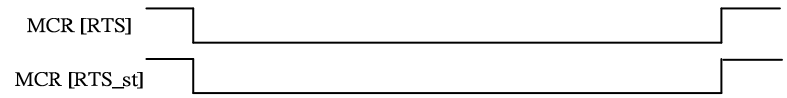
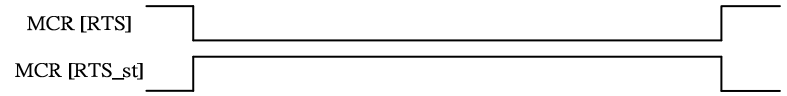
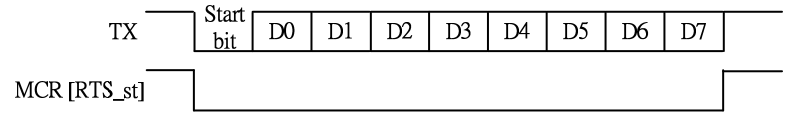
MODEM Control Register (UA_MCR)

Register	Offset	R/W	Description	Reset Value
UA_MCR	UART0_BA+0x10	R/W	UART0 Modem Control Register	0x0000_2000
	UART1_BA+0x10	R/W	UART1 Modem Control Register	0x0000_2000

31	30	29	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:**

Input1	Input0	Output
LEV_RTS (MCR.BIT9)	RTS (MCR.BIT1)	RTS_ST (MCR.BIT13, RTS Pin)
0	0	1
0	1	0
1	0	0

		1 | 1 | 1 *UART Mode : MCR[Lev_RTS] = 1*  *UART Mode : MCR[Lev_RTS] = 0*  **RS-485 Mode:** Input1 | Input0 | Output LEV_RTS (MCR.BIT9) | Tx | RTS_ST (MCR.BIT13, RTS Pin) 0 | x | 0 1 | x | 1 *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_0001
	UART1_BA+0x14	R/W	UART1 Modem Status Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							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 Trigger Level** This bit can change the CTS trigger level to send TX_FIFO data. 0= high level triggered 1= low level triggered
[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. **NOTE**: This bit is read only, but can be cleared by writing '1' to it.

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			
23	22	21	20	19	18	17	16
TX_OVER	TX_EMPTY	TX_POINTER					
15	14	13	12	11	10	9	8
RX_OVERL	RX_EMPTY	RX_POINTER					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	RS-485_ADD_DETF	Reserved		

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.
[27:24]	**Reserved**	Reserved
[23]	**TX_OVER**	**Transmitter FIFO Over (Read Only)** This bit indicates TX FIFO overflowing or not. If the number of bytes of transmitting data is greater than TX_FIFO (UA_RBR) size, 15 bytes of UART0/UART1, this bit will be set. 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_OVER**	**Receiver FIFO Over (Read Only)** This bit indicates RX FIFO overrunning or not. If the number of bytes of received data is greater than RX_FIFO (UA_RBR) size, 15 bytes of UART0/UART1, this bit will be set. 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 it.
[5]	**FEF**	**Framing Error Flag (Read Only)** This bit is set to logic 1 whenever the received character does not have a valid "stop bit" (that is, the stop bit following the last data bit or parity bit is detected as 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 it.
[4]	**PEF**	**Parity Error Flag (Read Only)** This bit is set to logic 1 whenever the received character does not have a valid "parity bit", and is reset whenever the CPU writes 1 to this bit. **NOTE**: This bit is read only, but can be cleared by writing '1' to it.
[3]	**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.
[2:0]	**Reserved**	Reserved

Interrupt Status Control Register (UA_ISR)

Register	Offset	R/W	Description	Reset Value
UA_ISR	UART0_BA+0x1C	R/W	UART0 Interrupt Status Register	0x0000_000A
	UART1_BA+0x1C	R/W	UART1 Interrupt Status Register	0x0000_000A

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			TOUT_INT	MODEM_INT	RLS_INT	THRE_INT	RDA_INT
7	6	5	4	3	2	1	0
Reserved			TOUT_IF	MODEM_IF	RLS_IF	THRE_IF	RDA_IF

Bits	Descriptions	
[31:13]	**Reserved**	Reserved
[12]	**TOUT_INT**	**Time Out Interrupt Indicator To Interrupt Controller (Read Only)** An AND output with inputs of RTO_IEN and TOUT_IF
[11]	**MODEM_INT**	**MODEM Status Interrupt Indicator To Interrupt Controller (Read Only).** An AND output with inputs of MODEM_IEN and MODEM_IF
[10]	**RLS_INT**	**Receive Line Status Interrupt Indicator To Interrupt Controller (Read Only).** An AND output with inputs of RLS_IEN and RLS_IF
[9]	**THRE_INT**	**Transmit Holding Register Empty Interrupt Indicator To Interrupt Controller (Read Only).** An AND output with inputs of THRE_IEN and THRE_IF
[8]	**RDA_INT**	**Receive Data Available Interrupt Indicator To Interrupt Controller (Read Only).** An AND output with inputs of RDA_IEN and RDA_IF
[7:5]	**Reserved**	Reserved
[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:** Write 1 to clear this bit to zero.
[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:** Write 1 to clear this bit to zero.
[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).

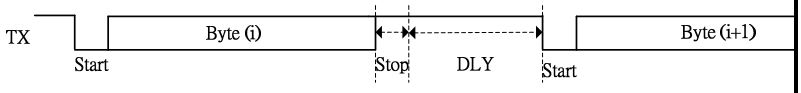
UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator to Interrupt Controller	Interrupt Flag	Flag Cleared by
RX Timeout Interrupt INT_TOUT	RTO_IEN	TOUT_INT	**TOUT_IF**	Read UA_RBR
Modem Status Interrupt INT_MODEM	MODEM_IEN	MODEM_INT	**MODEM_IF =** (DCTSF)	Write '1' to DCTSF
Receive Line Status 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 6.10-3 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
Reserved	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. 
[6:0]	**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.

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 6.10-4 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 6.10-4 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

Table 6.10-4 Baud rate equation table

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

6.11 Analog-to-Digital Converter (ADC)

6.11.1 Overview

NuMicro M051™ series contain one 12-bit successive approximation analog-to-digital converters (SAR A/D converter) with 8 input channels. The A/D converter supports four operation modes: single, burst, single-cycle scan and continuous scan mode. The A/D converters can be started by software and external STADC/P3.2 pin.

6.11.2 Features

- Analog input voltage range: 0~AVDD (Max to 5.0V).
- 12-bit resolution and 10-bit accuracy is guaranteed.
- Up to 8 single-end analog input channels or 4 differential analog input channels.
- Maximum ADC clock frequency is 16 MHz.
- Up to 600k SPS conversion rate.
- Four operating modes
  - Single mode: A/D conversion is performed one time on a specified channel.
  - Single-cycle scan mode: A/D conversion is performed one cycle on all specified channels with the sequence from the lowest numbered channel to the highest numbered channel.
  - Continuous scan mode: A/D converter continuously performs Single-cycle scan mode until software stops A/D conversion.
  - Burst mode: A/D conversion will sample and convert the specified single channel and sequentially store in FIFO.
- An A/D conversion can be started by
  - Software Write 1 to ADST bit
  - External pin STADC
- Conversion results are held in data registers for each channel with valid and overrun indicators.
- Conversion result can be compared with specify value and user can select whether to generate an interrupt when conversion result matches the compare register setting.

- Channel 7 supports 2 input sources: external analog voltage and internal fixed bandgap voltage.
- Support Self-calibration to minimize conversion error.

6.11.3 ADC Block Diagram

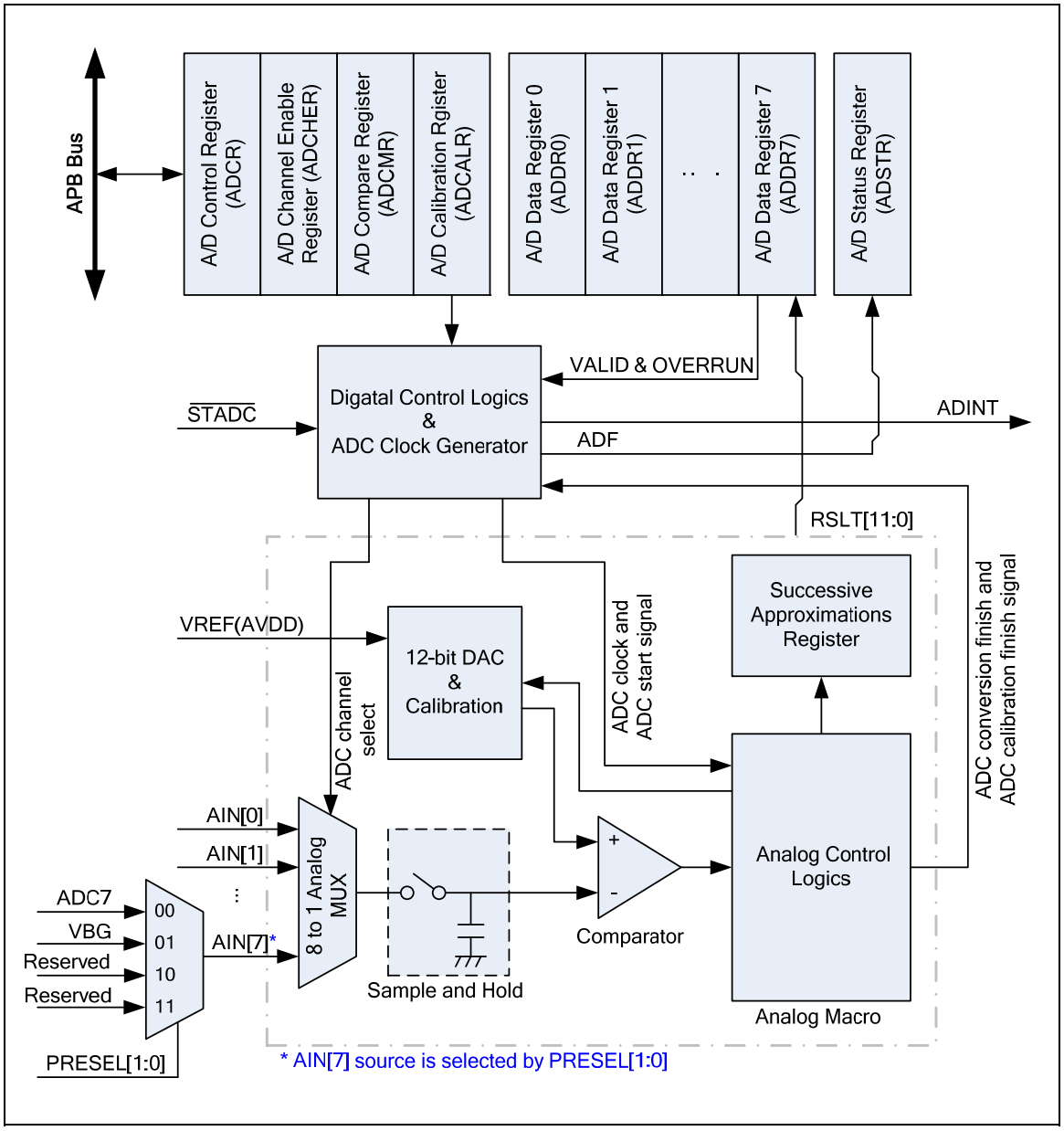


Figure 6.11-1 ADC Controller Block Diagram

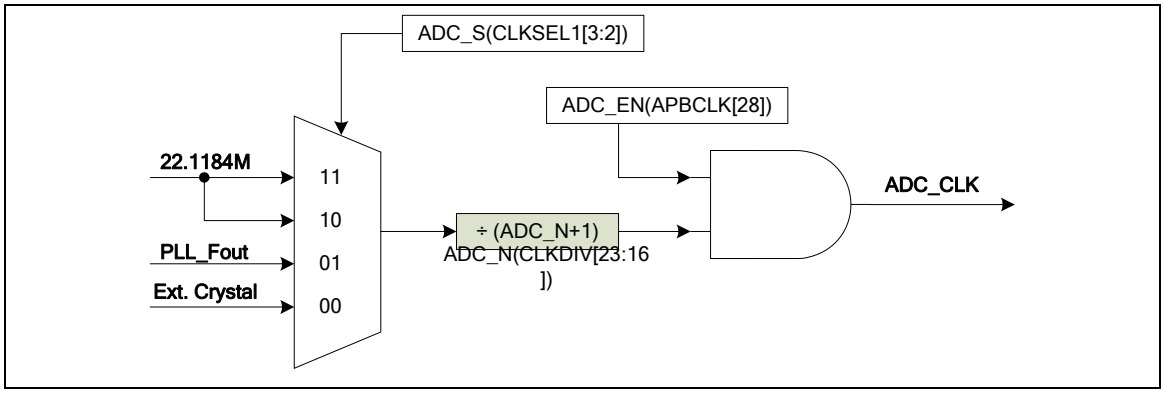


Figure 6.11-2 ADC Clock Control

6.11.4 ADC Operation Procedure

The A/D converter operates by successive approximation with 12-bit resolution. This A/D converter equips with self calibration function to minimize conversion error, user can write 1 to CALEN bit in ADCALR register to enable calibration function, while internal calibration is finished, the CAL_DONE bit will assert. The ADC has four operation modes: single, burst, single-cycle scan mode and continuous scan mode. When changing the operating mode or analog input channel enable, in order to prevent incorrect operation, software must clear ADST bit to 0 in ADCR register.

6.11.4.1 Self-Calibration

When chip power on or switch conversion mode between single-end mode and differential mode, it needs to do ADC self calibration to minimize the conversion error. User can write 1 to CALEN bit in ADCALR register to enable self calibration. The operation is process internally and it needs 127 ADC clocks to complete calibration. After CALEN is set to 1, software must wait CAL_DONE bit set by internal hardware. The detail timing is shown as below:

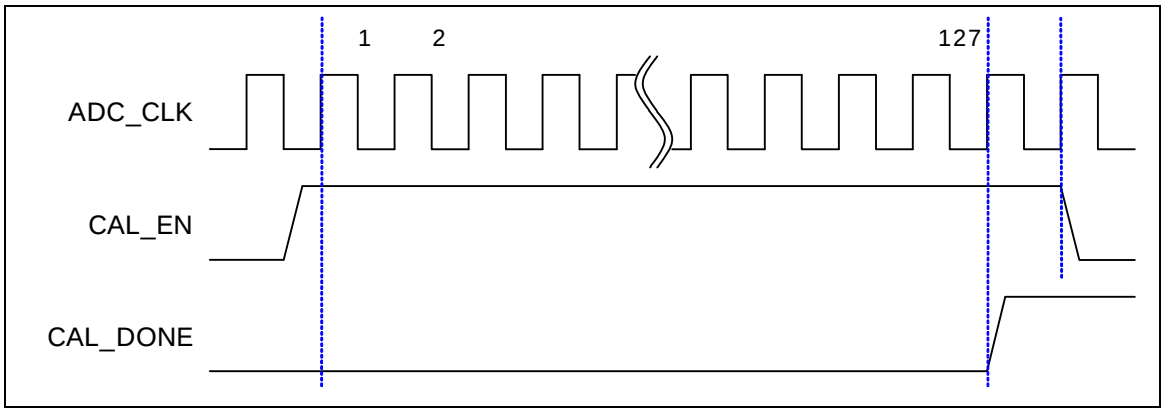


Figure 6.11-3 ADC Converter Self-Calibration Timing Diagram

6.11.4.2 ADC Clock Generator

The maximum sampling rate is up to 600K. The ADC engine has three clock sources selected by 2-bit ADC_S (CLKSEL[3:2]), the ADC clock frequency is divided by an 8-bit prescaler with the formula:

The ADC clock frequency = (ADC clock source frequency) / (ADC_N+1);
where the 8-bit ADC_N is located in register CLKDIV[23:16].

In generally, software can set ADC_S and ADC_N to get 16 MHz or slightly less.

Because the Bandgap voltage source lacks the driving capability, a conversion rate lower than 15 kHz (@5V) is recommended.

6.11.4.3 Single Mode

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

1. A/D conversion is started when the ADST bit in ADCR is set to 1 by software or external trigger input.

2. When A/D conversion is finished, the result is stored in the A/D data register corresponding to the channel.

3. On completion of conversion, the ADF bit in ADSR is set to 1 and ADC interrupt (ADINT) is requested If the ADIE bit is set to 1.

4. The ADST bit remains 1 during A/D conversion. When A/D conversion ends, the ADST bit is automatically cleared to 0 and the A/D converter enters idle state.

Note: If software enables more than one channel in single mode, the least channel is converted and other enabled channels will be ignored.

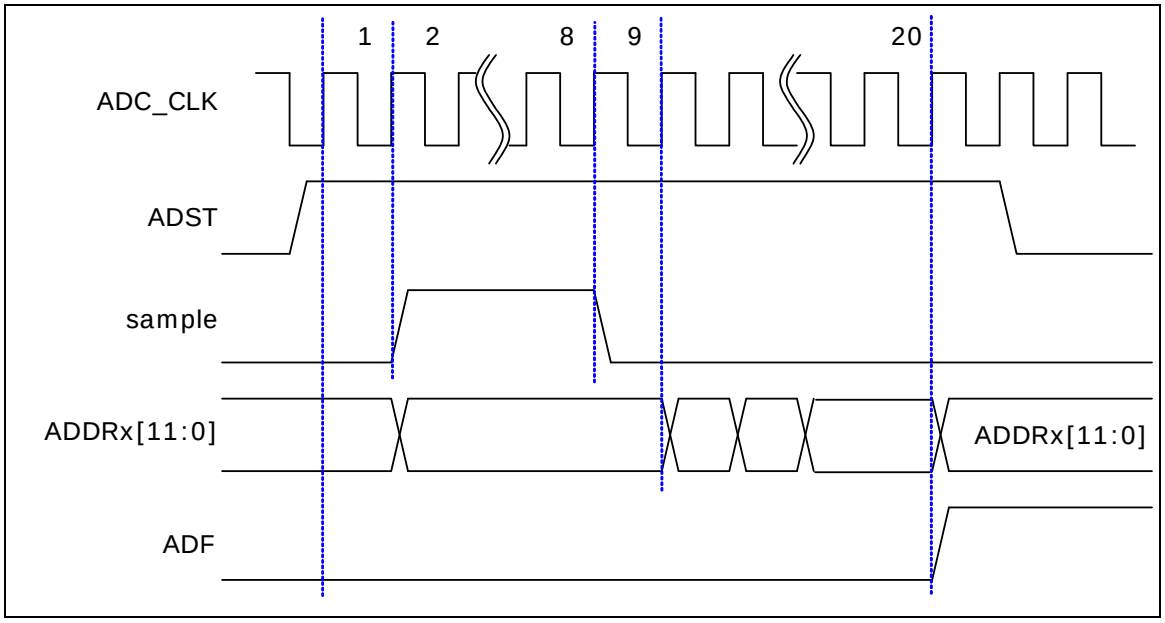


Figure 6.11-4 Single Mode Conversion Timing Diagram

6.11.4.4 Burst Mode

In burst mode, A/D conversion will sample and convert the specified single channel and sequentially store in FIFO (up to 8 samples). The operations are as follows:

1. When the ADST bit in ADCR is set to 1 by software or external trigger input, A/D conversion starts on the channel with the lowest number.

2. When A/D conversion for special enabled channel is completed, the result is sequentially transferred to FIFO and can be accessed from the A/D data register 0.

3. When more than 4 samples in FIFO, the ADF bit in ADSR is set to 1. If the ADIE bit is set to 1 at this time, an ADINT interrupt is requested after A/D conversion ends.

4. Steps 2 to 3 are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops and the A/D converter enters the idle state.

Note: If software enables more than one channel in burst mode, the channel with the lowest number is converted and other enabled channels will be ignored.

6.11.4.5 Single-Cycle Scan Mode

In single-cycle scan mode, A/D conversion will sample and convert the specified channels once in the sequence from the least to highest channel. Operations are as follows:

1. When the ADST bit in ADCR is set to 1 by a software or external trigger input, A/D conversion starts on the channel with the lowest number.

2. When A/D conversion for each enabled channel is completed, the result is sequentially transferred to the A/D data register corresponding to each channel.

3. When conversion of all the selected enabled channels is completed, the ADF bit in ADSR is set to 1. If the ADIE bit is set to 1 at this time, an ADINT interrupt is requested after A/D conversion ends.

4. After A/D conversion ends, the ADST bit is automatically cleared to 0 and the A/D converter enters idle state. If ADST is cleared to 0 before all enabled ADC channels conversion done, ADC controller will finish current conversion and the result of the lowest enabled ADC channel will become unpredictable.

An example timing diagram for single-cycle scan on enabled channels (0, 2, 3 and 7) is shown in the

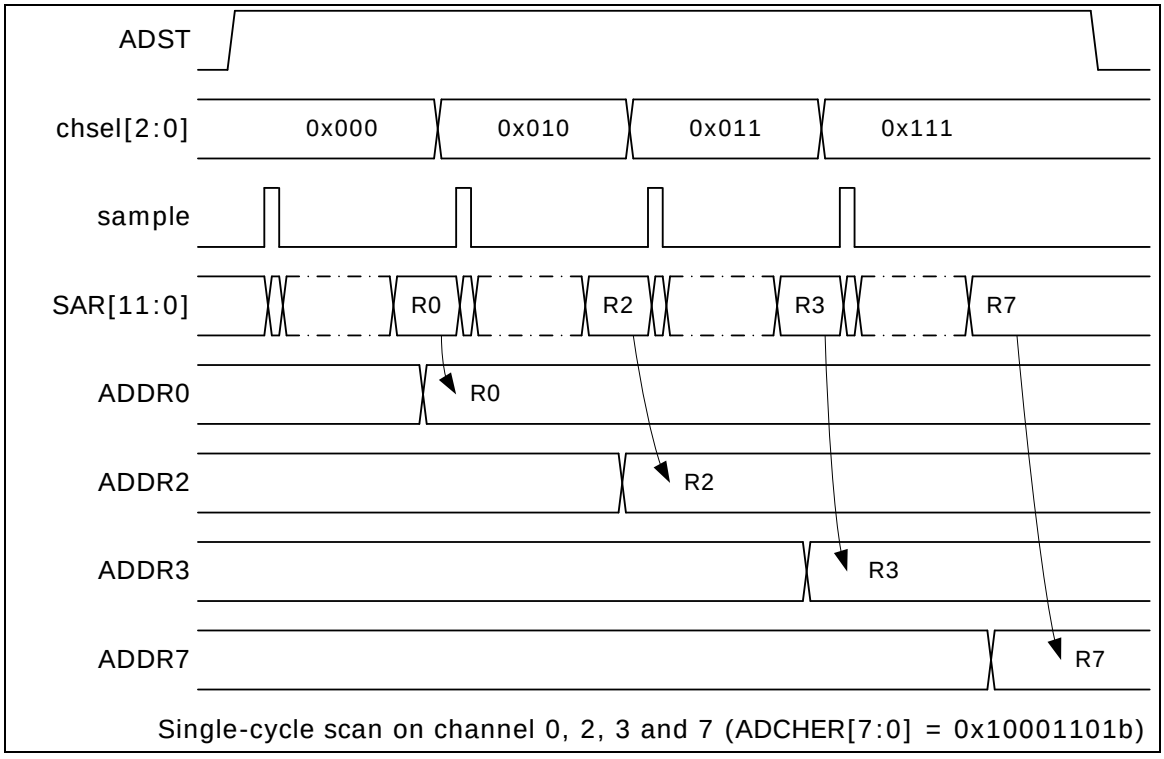


Figure 6.11-5 Single-Cycle Scan on Enabled Channels Timing Diagram

.

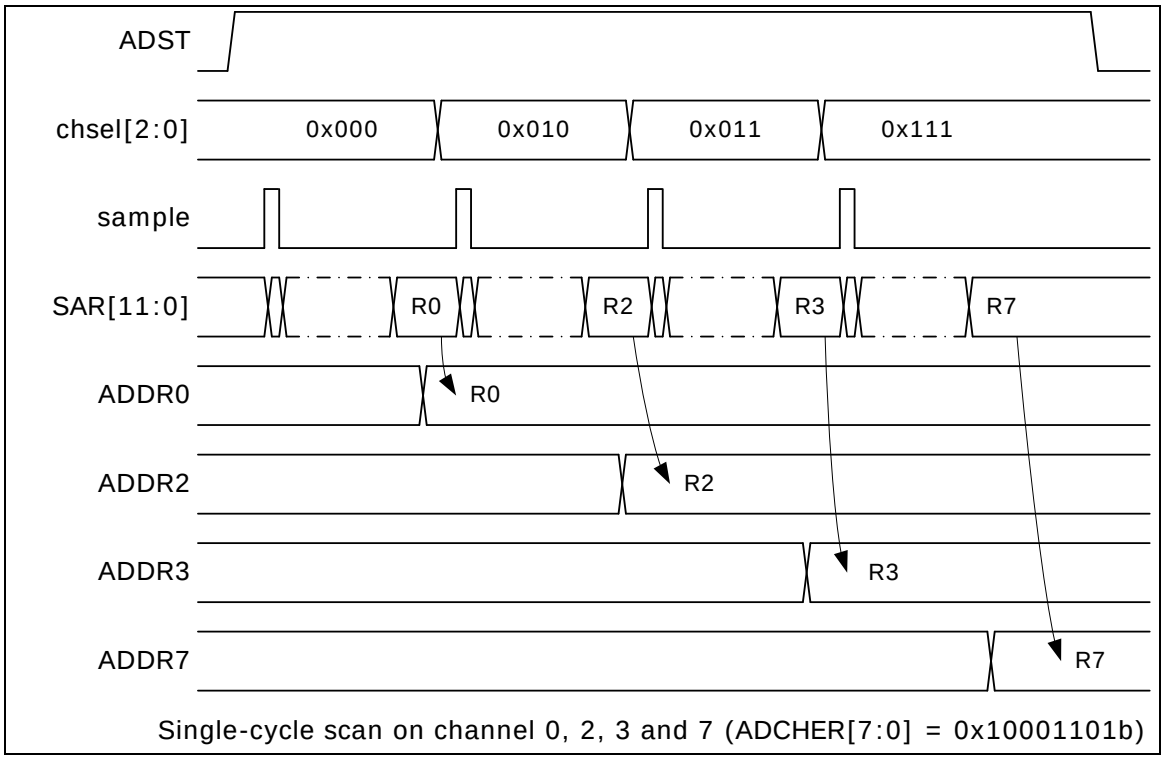


Figure 6.11-5 Single-Cycle Scan on Enabled Channels Timing Diagram

6.11.4.6 Continuous Scan Mode

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

1. When the ADST bit in ADCR is set to 1 by software or external trigger input, A/D conversion starts on the channel with the lowest number.

2. When A/D conversion for each enabled channel is completed, the result of each enabled channel is stored in the A/D data register corresponding to each enabled channel.

3. When all enabled channel sequentially completes A/D converting once, the ADF bit (ADSR[0]) will be set to 1. If the ADIE bit is set to 1 at this time, an ADINT interrupt is requested after A/D conversion ends. Conversion of the 1st enabled channel starts again.

4. Steps 2 to 3 are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops and the A/D converter enters the idle state. When ADST is cleared to 0, ADC controller will finish current conversion and the result of the lowest enabled ADC channel will become unpredictable.

An example timing diagram for continuous scan on enabled channels (0, 2, 3 and 7) is shown in the Figure 6.11-6 .

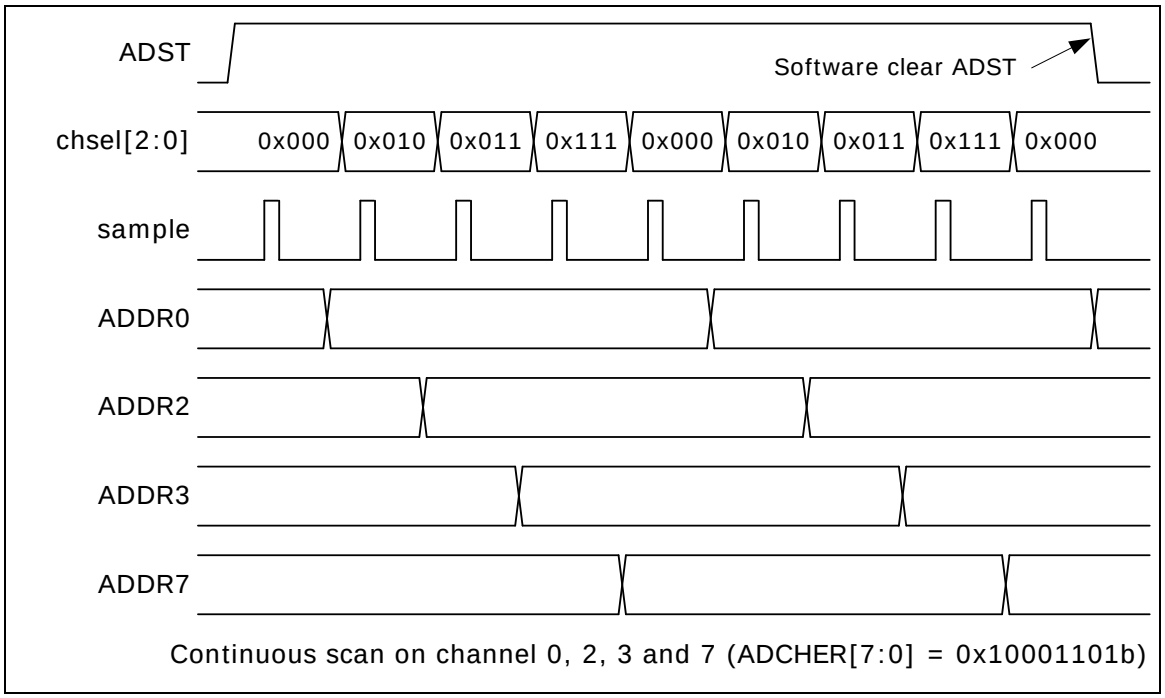


Figure 6.11-6 Continuous Scan on Enabled Channels Timing Diagram

6.11.4.7 Input Sampling and A/D Conversion Time

A/D conversion can be triggered by external pin request. When the ADCR.TRGEN is set to high to enable ADC external trigger function, setting the TRGS[1:0] bits to 00b is to select external trigger input from the STADC pin. Software can set TRGCOND[1:0] to select trigger condition is **falling/rising edge or low/high** level. An 8-bit sampling counter is used to deglitch. If level trigger condition is selected, the STADC pin must be kept at defined state at least 8 PCLKs. The ADST bit will be set to 1 at the 9th PCLK and start to conversion. Conversion is continuous if external trigger input is pull at low (or high state) in level trigger mode. It is stopped only when external condition trigger condition disappears. If edge trigger condition is selected, the high and low state must be kept at least 4 PLCKs. Pulse that is shorter than this specification will be ignored.

6.11.4.8 Conversion Result Monitor by Compare Mode

NuMicro M051™ series controller provide two sets of compare register ADCMPR0 and 1 to monitor maximum two specified channels conversion result from A/D conversion module, refer to Figure 6.11-7. Software can select which channel to be monitored by set CMPCH(ADCMPRx[5:0]) and CMPCOND bit is used to check conversion result is less than specify value or greater than (equal to) value specified in CMPD[11:0]. When the conversion of

the channel specified by CMPCH is completed, the comparing action will be triggered one time automatically. When the compare result meets the setting, compare match counter will increase 1, otherwise, the compare match counter will be clear to 0. When counter value reach the setting of (CMPMATCNT+1) then CMPF bit will be set to 1, if CMPIE bit is set then an ADINT interrupt request is generated. Software can use it to monitor the external analog input pin voltage transition in scan mode without imposing a load on software. Detail logics diagram is shown in the Figure 6.11-7.

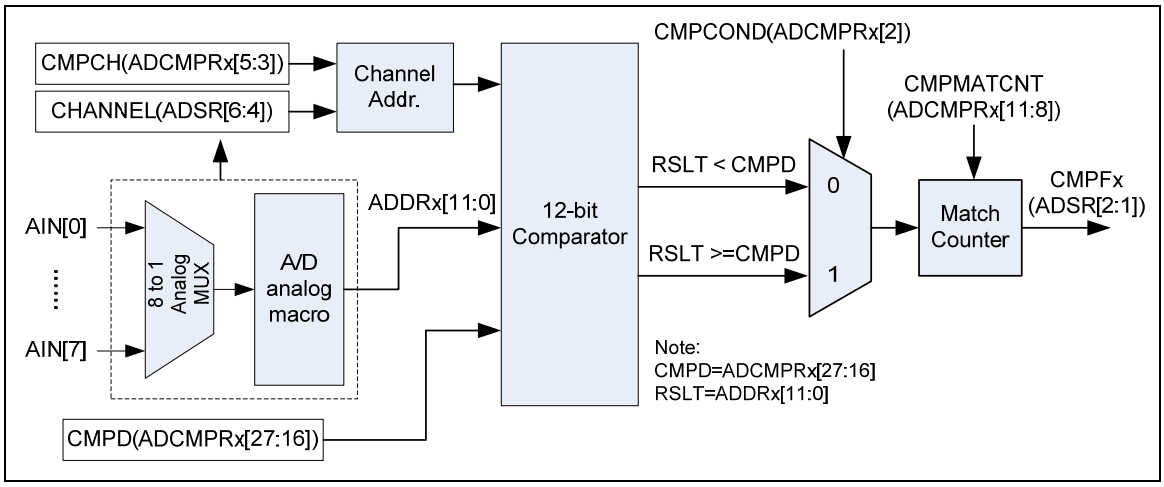


Figure 6.11-7 A/D Conversion Result Monitor Logics Diagram

6.11.4.9 Interrupt Sources

The A/D converter generates a conversion end ADF in ADSR register upon the end of A/D conversion. If ADIE bit in ADCR is set then conversion end interrupt request ADINT is generated. If CMPIE bit is enabled, when A/D conversion result meets setting in ADCMPR register, monitor interrupt is generated, ADINT will be set also. CPU can clear CMPF and ADF to stop interrupt request.

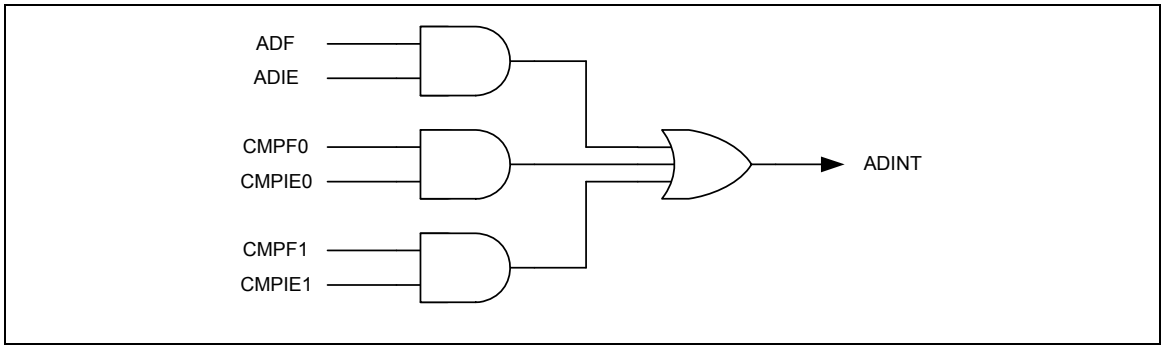


Figure 6.11-8 A/D Controller Interrupt

6.11.5 ADC Controller Registers Map

R: read only, **W**: write only, **R/W**: both read and write, **C**: Only value 0 can be written

Register	Offset	R/W	Description	Reset Value
ADC_BA = 0x400E_0000				
ADDR0	ADC_BA+0x00	R	A/D Data Register 0	0x0000_0000
ADDR1	ADC_BA+0x04	R	A/D Data Register 1	0x0000_0000
ADDR2	ADC_BA+0x08	R	A/D Data Register 2	0x0000_0000
ADDR3	ADC_BA+0x0C	R	A/D Data Register 3	0x0000_0000
ADDR4	ADC_BA+0x10	R	A/D Data Register 4	0x0000_0000
ADDR5	ADC_BA+0x14	R	A/D Data Register 5	0x0000_0000
ADDR6	ADC_BA+0x18	R	A/D Data Register 6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	A/D Data Register 7	0x0000_0000
ADCR	ADC_BA+0x20	R/W	A/D Control Register	0x0000_0000
ADCHER	ADC_BA+0x24	R/W	A/D Channel Enable Register	0x0000_0000
ADCMPR0	ADC_BA+0x28	R/W	A/D Compare Register 0	0x0000_0000
ADCMPR1	ADC_BA+0x2C	R/W	A/D Compare Register 1	0x0000_0000
ADSR	ADC_BA+0x30	R/W	A/D Status Register	0x0000_0000
ADCALR	ADC_BA+0x34	R/W	A/D Calibration Register	0x0000_0000

6.11.6 ADC Controller Registers Description

A/D Data Registers (ADDR0 ~ ADDR7)

Register	Offset	R/W	Description	Reset Value
ADDR0	ADC_BA+0x00	R	A/D Data Register 0	0x0000_0000
ADDR1	ADC_BA+0x04	R	A/D Data Register 1	0x0000_0000
ADDR2	ADC_BA+0x08	R	A/D Data Register 2	0x0000_0000
ADDR3	ADC_BA+0x0c	R	A/D Data Register 3	0x0000_0000
ADDR4	ADC_BA+0x10	R	A/D Data Register 4	0x0000_0000
ADDR5	ADC_BA+0x14	R	A/D Data Register 5	0x0000_0000
ADDR6	ADC_BA+0x18	R	A/D Data Register 6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	A/D Data Register 7	0x0000_0000

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

Bits	Descriptions	
[31:18]	**Reserved**	**-**
[17]	**VALID**	**Valid Flag** 1 = Data in RSLT[11:0] bits is valid. 0 = Data in RSLT[11:0] bits is not valid. This bit is set to 1 when corresponding channel analog input conversion is completed and cleared by hardware after ADDR register is read. This is a read only bit

[16]	**OVERRUN**	**Over Run Flag (Read Only)** 1 = Data in RSLT[11:0] is overwrite. 0 = Data in RSLT[11:0] is recent conversion result. If converted data in RSLT[11:0] has not been read before new conversion result is loaded to this register, OVERRUN is set to 1. It is cleared by hardware after ADDR register is read.
[15:12]	**Reserved**	-
[11:0]	**RSLT**	**A/D Conversion Result** This field contains conversion result of ADC.

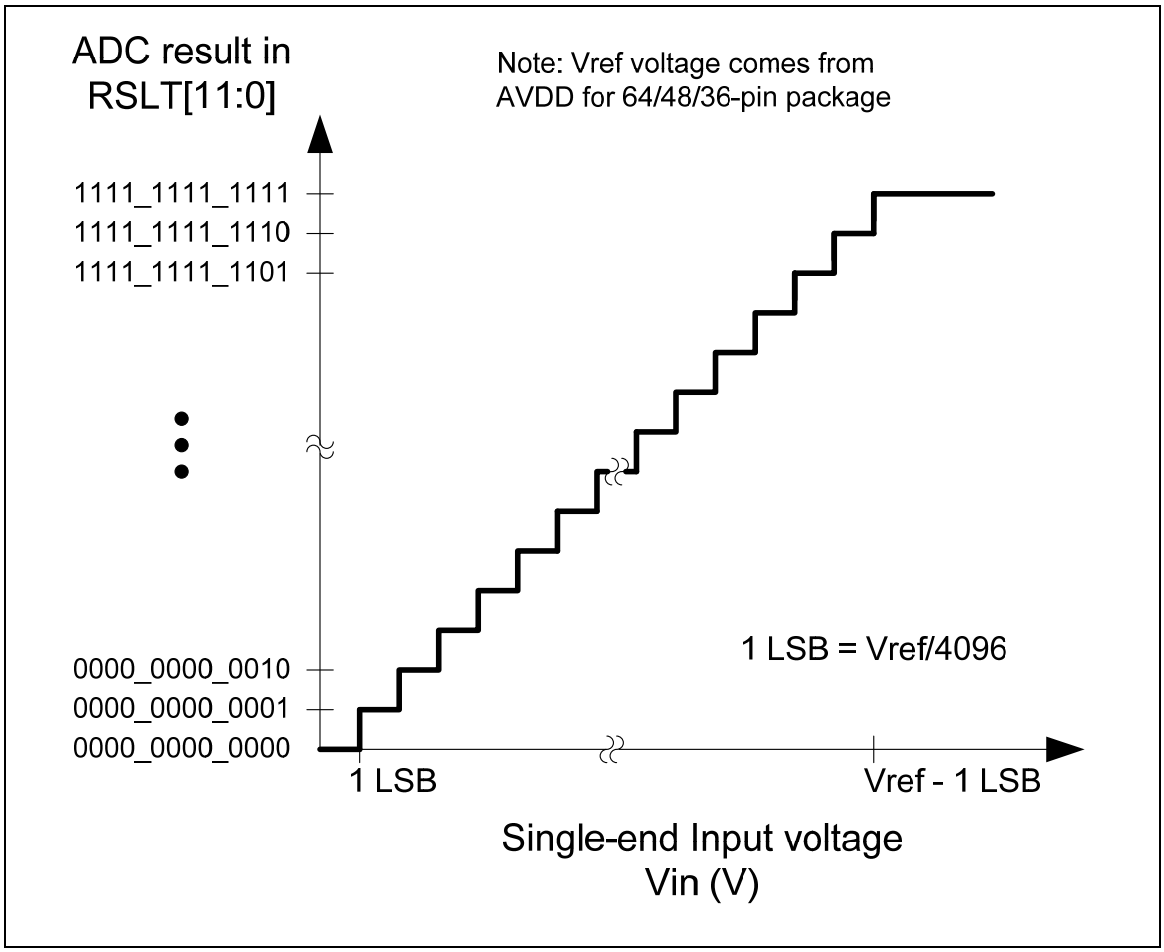


Figure 6.11-9 ADC single-end input conversion voltage and conversion result mapping diagram

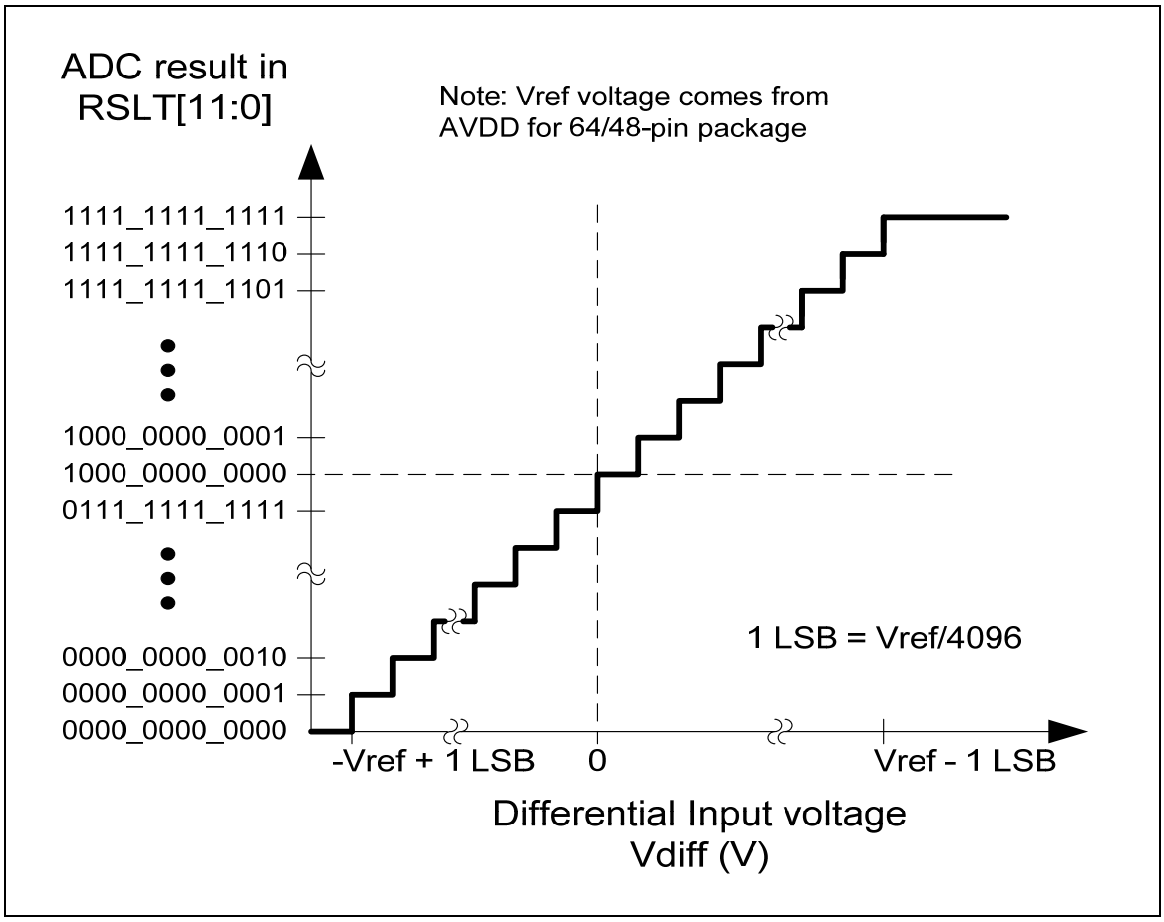


Figure 6.11-10 ADC differential input conversion voltage and conversion result mapping diagram

A/D Control Register (ADCR)

Register	Offset	R/W	Description	Reset Value
ADCR	ADC_BA+0x20	R/W	ADC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				ADST	DIFFEN	Reserved	TRGEN

7	6	5	4	3	2	1	0
TRGCOND		TRGS		ADMD		ADIE	ADEN

Bits	Descriptions	
[31:12]	**Reserved**	-
[11]	**ADST**	**A/D Conversion Start** 1 = Conversion start. 0 = Conversion stopped and A/D converter enter idle state. ADST bit can be set to 1 from two sources: software write and external pin STADC. ADST is cleared to 0 by hardware automatically at the ends of single mode and single-cycle scan mode on specified channels. In continuous scan mode, A/D conversion is continuously performed sequentially until software write 0 to this bit or chip reset.
[10]	**DIFFEN**	**A/D Differential Input Mode Enable** 1 = A/D is in differential analog input mode 0 = A/D is in single-end analog input mode Differential input paired channel / ADC analog input: 0 → AIN0, AIN1; 1 → AIN2, AIN3; 2 → AIN4, AIN5; 3 → AIN6, AIN7 **Differential input voltage (V_{diff}) = V_{plus} - V_{minus}** **Note:** In differential input mode, only one of the two corresponding channels needs to be enabled in ADCHER. The conversion result will be placed to the corresponding data register of the enabled channel. If both channels of a differential input paired channel are enabled, the ADC will convert it twice in scan mode. And then write the conversion result to the two corresponding data registers.
[9]	**Reserved**	-
[8]	**TRGEN**	**External Trigger Enable** Enable or disable triggering of A/D conversion by external STADC pin. 1= Enable 0= Disable

[7:6]	**TRGCOND**	**External Trigger Condition** These two bits decide external pin STADC trigger event is level or edge. The signal must be kept at stable state at least 8 PCLKs for level trigger and 4 PCLKs at high and low state for edge trigger. 00 = Low level 01 = High level 10 = Falling edge 11 = Rising edge
[5:4]	**TRGS**	**Hardware Trigger Source** 00 = A/D conversion is started by external STADC pin. Others = Reserved Software should disable TRGEN and ADST before change TRGS. In hardware trigger mode, the ADST bit is set by the external trigger from STADC.
[3:2]	**ADMD**	**A/D Converter Operation Mode** 00 = Single conversion 01 = Burst conversion 10 = Single-cycle scan 11 = Continuous scan When changing the operation mode, software should disable ADST bit firstly. **Note:** In Burst Mode, the A/D result data always at Data Register 0.
[1]	**ADIE**	**A/D Interrupt Enable** 1 = Enable A/D interrupt function 0 = Disable A/D interrupt function A/D conversion end interrupt request is generated if ADIE bit is set to 1.
[0]	**ADEN**	**A/D Converter Enable** 1 = Enable 0 = Disable Before starting A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit power consumption.

A/D Channel Enable Register (ADCHER)

Register	Offset	R/W	Description	Reset Value
ADCHER	ADC_BA+0x24	R/W	A/D Channel Enable	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						PRESEL[1:0]	
7	6	5	4	3	2	1	0
CHEN7	CHEN6	CHEN5	CHEN4	CHEN3	CHEN2	CHEN1	CHEN0

Bits	Descriptions	
[31:10]	**Reserved**	-
[9:8]	**PRESEL[1:0]**	**Analog Input Channel 7 select** 00= External Analog Input 01= Internal Bandgap voltage 10= Reserved 11= Reserved
[7]	**CHEN7**	**Analog Input Channel 7 Enable** 1 = Enable 0 = Disable
[6]	**CHEN6**	**Analog Input Channel 6 Enable** 1 = Enable 0 = Disable
[5]	**CHEN5**	**Analog Input Channel 5 Enable** 1 = Enable 0 = Disable
[4]	**CHEN4**	**Analog Input Channel 4 Enable** 1 = Enable 0 = Disable
[3]	**CHEN3**	**Analog Input Channel 3 Enable** 1 = Enable 0 = Disable
[2]	**CHEN2**	**Analog Input Channel 2 Enable**

		1 = Enable 0 = Disable
[1]	**CHEN1**	**Analog Input Channel 1 Enable** 1 = Enable 0 = Disable
[0]	**CHEN0**	**Analog Input Channel 0 Enable** 1 = Enable 0 = Disable This channel is enabled if CHEN1~7 are set as 0s. **If software enables more than one channel in single mode, the least channel is converted and other enabled channels will be ignored.**

A/D Compare Register 0/1 (ADCMPR0/1)

Register	Offset	R/W	Description	Reset Value
ADCMPR0	ADC_BA+0x28	R/W	A/D Compare Register 0	0x0000_0000
ADCMPR1	ADC_BA+0x2C	R/W	A/D Compare Register 1	0x0000_0000

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

Bits	Descriptions	
[31:28]	**Reserved**	-
[27:16]	**CMPD**	**Comparison Data** The 12 bits data is used to compare with conversion result of specified channel. Software can use it to monitor the external analog input pin voltage transition in scan mode without imposing a load on software.
[15:12]	**Reserved**	**-**
[11:8]	**CMPMATCNT**	**Compare Match Count** When the specified A/D channel analog conversion result matches the compare condition defined by CMPCOND[2], the internal match counter will increase 1. When the internal counter reaches the value to (CMPMATCNT +1), the CMPF bit will be set.
[5:3]	**CMPCH**	**Compare Channel Selection** 000 = Channel 0 conversion result is selected to be compared. 001 = Channel 1 conversion result is selected to be compared. 010 = Channel 2 conversion result is selected to be compared. 011 = Channel 3 conversion result is selected to be compared. 100 = Channel 4 conversion result is selected to be compared. 101 = Channel 5 conversion result is selected to be compared.

		110 = Channel 6 conversion result is selected to be compared. 111 = Channel 7 conversion result is selected to be compared.
[2]	**CMPCOND**	**Compare Condition** 1= Set the compare condition as that when a 12-bit A/D conversion result is greater or equal to the 12-bit CMPD (ADCMPRx[27:16]), the internal match counter will increase one. 0= Set the compare condition as that when a 12-bit A/D conversion result is less than the 12-bit CMPD (ADCMPRx[27:16]), the internal match counter will increase one. **Note:** When the internal counter reaches the value to (CMPMATCNT +1), the CMPF bit will be set.
[1]	**CMPIE**	**Compare Interrupt Enable** 1 = Enable compare function interrupt. 0 = Disable compare function interrupt. If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMATCNT, CMPF bit will be asserted, in the meanwhile, if CMPIE is set to 1, a compare interrupt request is generated.
[0]	**CMPEN**	**Compare Enable** 1 = Enable compare. 0 = Disable compare. Set this bit to 1 to enable ADC controller to compare CMPD[11:0] with specified channel conversion result when converted data is loaded into ADDR register.

A/D Status Register (ADSR)

Register	Offset	R/W	Description	Reset Value
ADSR	ADC_BA+0x30	R/W	ADC Channel Selection Enable Register	undefined

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
OVERRUN							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VALID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CHANNEL**			**BUSY**	**CMPF1**	**CMPF0**	**ADF**

Bits	Descriptions	
[31:24]	**Reserved**	-
[23:16]	**OVERRUN**	**Over Run flag (Read Only)** It is a mirror to OVERRUN bit in ADDRx When ADC in Burst Mode, and the FIFO is overrun, OVERRUN[7:0] will all set to 1.
[15:8]	**VALID**	**Data Valid flag (Read Only)** It is a mirror of VALID bit in ADDRx When ADC in Burst Mode, and the FIFO is valid, VALID[7:0] will all set to 1.
[7]	**Reserved**	-
[6:4]	**CHANNEL**	**Current Conversion Channel** This filed reflects current conversion channel when BUSY=1. When BUSY=0, it shows the next channel will be converted. It is read only.
[3]	**BUSY**	**BUSY/IDLE** 1 = A/D converter is busy at conversion. 0 = A/D converter is in idle state. This bit is mirror of as ADST bit in ADCR. It is read only.

[2]	**CMPF1**	**Compare Flag** When the selected channel A/D conversion result meets setting condition in ADCMPR1 then this bit is set to 1. And it is cleared by writing 1 to self. 1 = Conversion result in ADDR meets ADCMPR1 setting 0 = Conversion result in ADDR does not meet ADCMPR1 setting
[1]	**CMPF0**	**Compare Flag** When the selected channel A/D conversion result meets setting condition in ADCMPR0 then this bit is set to 1. And it is cleared by writing 1 to self. 1 = Conversion result in ADDR meets ADCMPR0setting 0 = Conversion result in ADDR does not meet ADCMPR0setting
[0]	**ADF**	**A/D Conversion End Flag** A status flag that indicates the end of A/D conversion. ADF is set to 1 at these three conditions: 1. When A/D conversion ends in single mode 2. When A/D conversion ends on all specified channels in scan mode. 3. When more than 4 samples in FIFO in Burst mode. This flag can be cleared by writing 1 to self.

A/D Calibration Register (ADCALR)

Register	Offset	R/W	Description	Reset Value
ADCALR	ADC_BA+0x34	R/W	A/D Calibration Register	0x0000_0000

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

Bits	Descriptions	
[31:2]	**Reserved**	-
[1]	**CALDONE**	**Calibration is Done (read only)** 1 = A/D converter self calibration is done 0 = A/D converter has not been calibrated or calibration is in progress if CALEN bit is set. When 0 is written to CALEN bit, CALDONE bit is cleared by hardware immediately. It is a read only bit.
[0]	**CALEN**	**Self Calibration Enable** 1 = Enable self calibration 0 = Disable self calibration Software can set this bit to 1 enables A/D converter to do self calibration function. It needs 127 ADC clocks to complete calibration. This bit must be kept at 1 after CALDONE asserted. Clearing this bit will disable self calibration function.

6.12 External Bus Interface (EBI)

6.12.1 Overview

NuMicro M051™ series equips an external bus interface (EBI) for external device used.

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

6.12.2 Features

External Bus Interface has the following functions:

1. External devices with max. 64K-byte size (8 bit data width)/128K-byte (16 bit data width) supported
2. Variable external bus base clock (MCLK) supported
3. 8 bit or 16 bit data width supported
4. Variable data access time (tACC), address latch enable time (tALE) and address hold time (tAHD) supported
5. Address bus and data bus multiplex mode supported to save the address pins
6. Configurable idle cycle supported for different access condition: Write command finish (W2X), Read-to-Read (R2R)

6.12.3 EBI Block Diagram

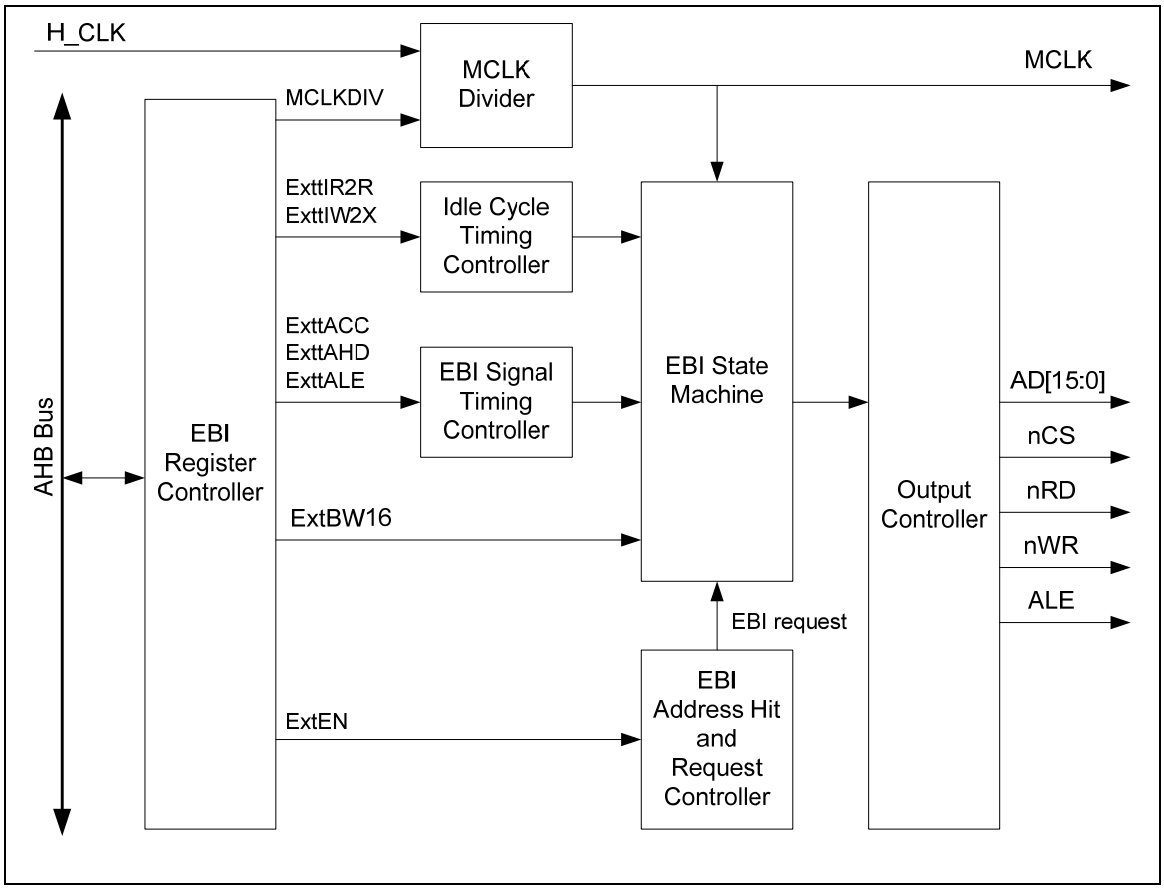


Figure 6.12-1 EBI Block Diagram

6.12.4 Operation Procedure

6.12.4.1 EBI Area and Address Hit

NuMicro M051™ series EBI mapping address is located at 0x6000_0000 ~ 0x6001_FFFF and the total memory space is 128Kbyte. When system request address hit EBI's memory space, the corresponding EBI chip select signal is assert and EBI state machine operates.

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

6.12.4.2 EBI Data Width Connection

NuMicro M051™ series EBI support device whose address bus and data bus are multiplexed. For the external device with separated address and data bus, the connection to device needs additional logic to latch the address. In this case, pin ALE is connected to the latch device such as 74HC373. Pin AD is the input of the latch device, and the output of the latch device is connected to the address of external device. For 16-bit device, the AD [15:0] shared by address and 16-bit data. For 8-bit device, only AD [7:0] shared by address and 8-bit data, AD [15:8] is dedicated for address and could be connected to 8-bit device directly.

For 8-bit data width, NuMicro M051™ system address bit [15:0] is used as the device's address [15:0]. For 16-bit data width, NuMicro M051™ system address bit [16:1] is used as the device's address [15:0] and NuMicro M051™ system address bit [0] is useless.

EBI bit width	System address (AHBADR)	EBI address (AD)
8 bit	AHBADR[15:0]	AD[15:0]
16 bit	AHBADR[16:1]	AD[15:0]

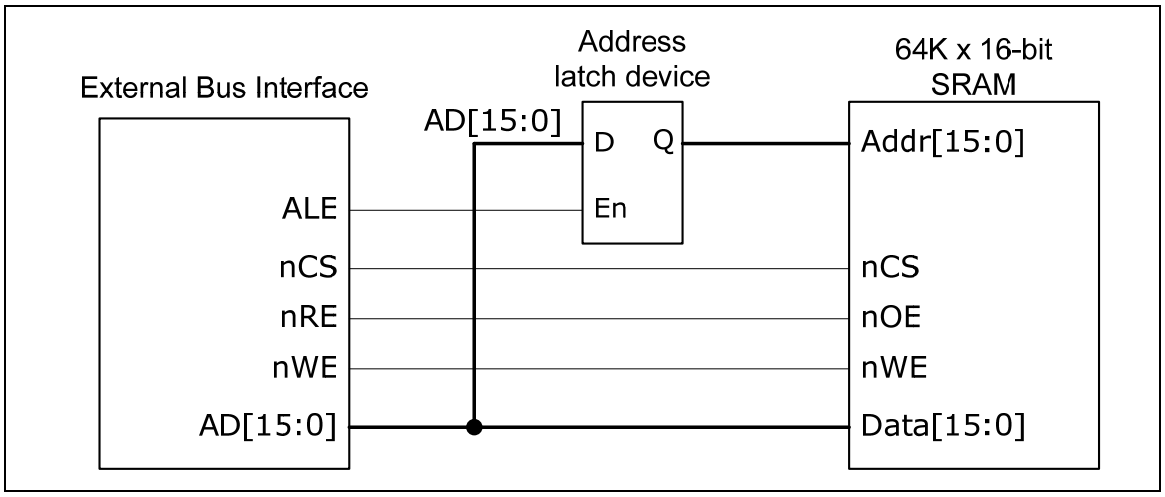


Figure 6.12-2 Connection of 16-bit EBI Data Width with 16-bit Device

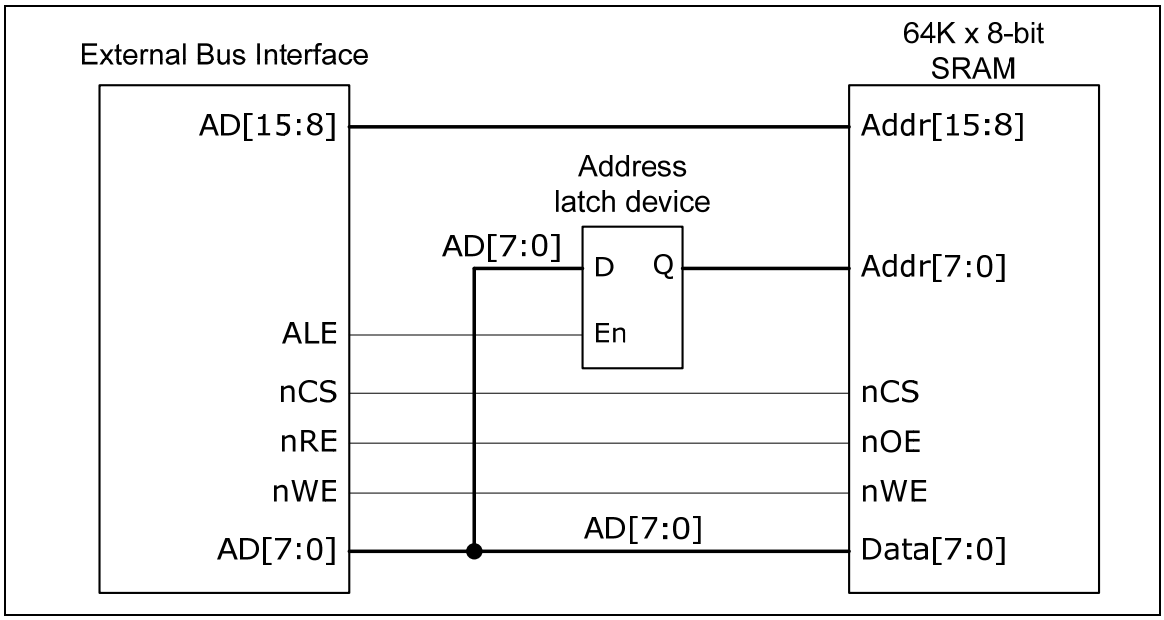


Figure 6.12-3 Connection of 8-bit EBI Data Width with 8-bit Device

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

6.12.4.3 EBI Operating Control

MCLK Control

In NuMicro M051™ series, all EBI signals will be synchronized by MCLK when EBI is operating. When NuMicro M051™ series connects to the external device with slower operating frequency, the MCLK can divide most to HCLK/32 by setting MCLKDIV of register EBICON. Therefore, NuMicro M051™ can suitable for a wide frequency range of EBI device. If MCLK is set to HCLK/1, EBI signals are synchronized by positive edge of MCLK, else by negative edge of MCLK.

Operation and Access Timing Control

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

for data access hold time (tAHD) and chip select asserts to high, address is released by current access control.

NuMicro M051™ series provide a flexible EBI timing control for different external device. In NuMicro M051™ EBI timing control, tASU, tLHD and tA2D are fixed to 1 MCLK cycle, tAHD can modulate to 1~8 MCLK cycles by setting ExttAHD of register EXTIME, tACC can modulate to 1~32 MCLK cycles by setting ExttACC of register EXTIME, and tALE can modulate to 1~8 MCLK cycles by setting tALE of register EBICON.

Parameter	Value	Unit	Description
tASU	1	MCLK	Address Latch Setup Time.
tALE	1 ~ 8	MCLK	ALE High Period. Controlled by ExttALE of EBICON.
tLHD	1	MCLK	Address Latch Hold Time.
tA2D	1	MCLK	Address To Data Delay (Bus Turn-Around Time).
tACC	1 ~ 32	MCLK	Data Access Time. Controlled by ExttACC of EXTIME.
tAHD	1 ~ 8	MCLK	Data Access Hold Time. Controlled by ExttAHB of EXTIME.
IDLE	0 ~ 15	MCLK	Idle Cycle. Controlled by ExtIR2R and ExtIW2X of EXTIME.

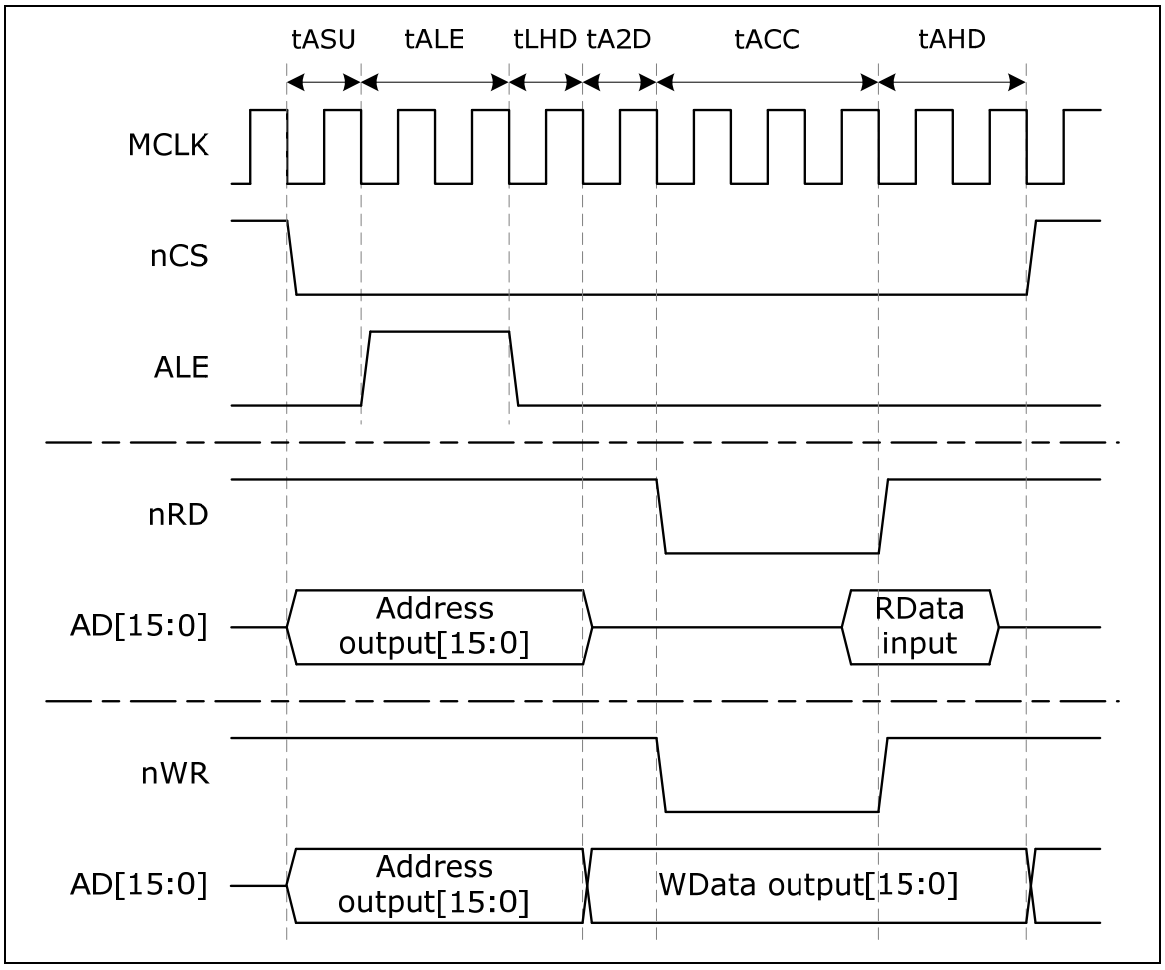


Figure 6.12-4 Timing Control Waveform for 16bit Data Width

Above timing waveform is an example of setting 16bit data width. In this example, AD bus is used for being address[15:0] and data[15:0]. When ALE assert to high, AD is address output. After address is latched, ALE asserts to low and the AD bus change to high impedance to wait device output data in read access operation, or it is used for being write data output.

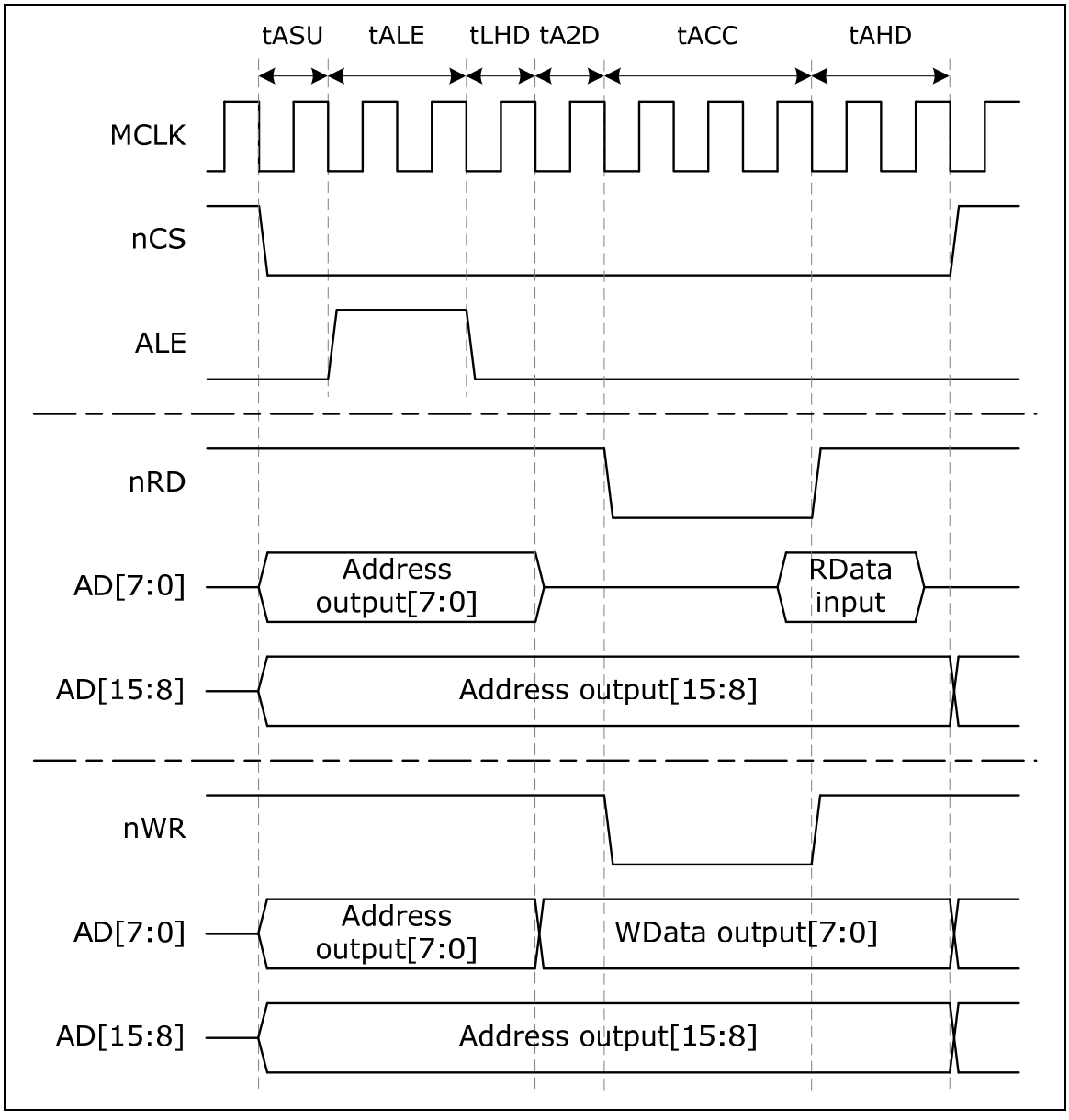


Figure 6.12-5 Timing Control Waveform for 8bit Data Width

Above timing waveform is an example of setting 8bit data width. The difference between 8bit and 16bit data width is AD[15:8]. In 8bit data width setting, AD[15:8] always be Address[15:8] output so that external latch need only 8 bit width.

Insert Idle Cycle

When EBI accessing continuously, there may occur bus conflict if the device access time is much slow with system operating. NuMicro M051™ supply additional idle cycle to solve this problem. During idle cycle, all control signals of EBI are inactive. The Figure 6.12-6 shows the idle cycle waveform.

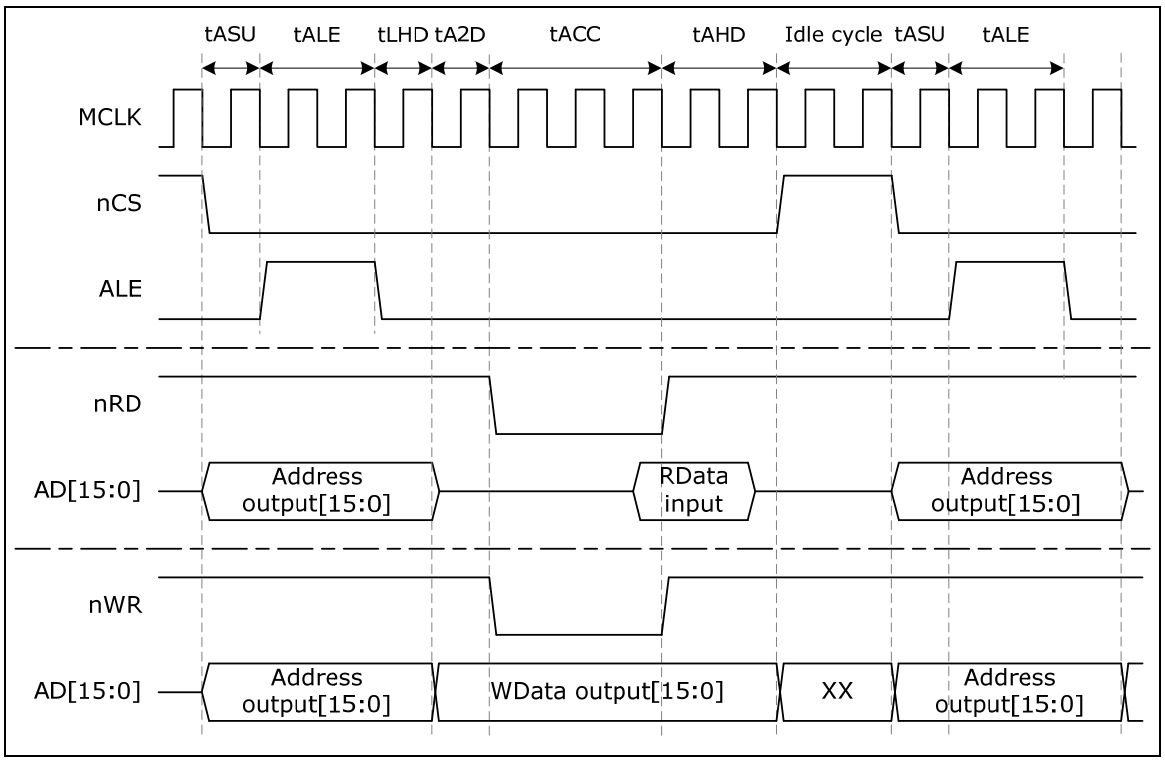


Figure 6.12-6 Timing Control Waveform for Insert Idle Cycle

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

1. After write access

2. After read access and before next read access

By setting ExtIW2X and ExtIR2R of register EXTIME, the time of idle cycle can be specified from 0~15 MCLK.

6.12.5 EBI Controller Registers Map

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

Register	Offset	R/W	Description	Reset Value
EBI_CTL_BA = 0x5001_0000				
EBICON	EBI_CTL_BA+0x00	R/W	External Bus Interface General Control Register	0x0000_0000
EXTIME	EBI_CTL_BA+0x04	R/W	External Bus Interface Timing Control Register	0x0000_0000

6.12.6 EBI Controller Registers Description

External Bus Interface CONTROL REGISTER (EBICON)

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

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

Bits	Descriptions	
[31:19]	**Reserved**	Reserved
[18:16]	**ExttALE**	**Expand Time of ALE** The ALE width (tALE) to latch the address can be controlled by ExttALE. tALE = (ExttALE+1)*MCLK

[15:11]	**Reserved**	Reserved
[10:8]	**MCLKDIV**	**External Output Clock Divider** The frequency of EBI output clock is controlled by MCLKDIV. MCLKDIV / Output clock (MCLK) 000 / HCLK/1 001 / HCLK/2 010 / HCLK/4 011 / HCLK/8 100 / HCLK/16 101 / HCLK/32 11X / default Notice: Default value of output clock is HCLK/1
[7:2]	**Reserved**	Reserved
[1]	**ExtBW16**	**EBI data width 16 bit** This bit defines if the data bus is 8-bit or 16-bit. 0 = EBI data width is 8 bit 1 = EBI data width is 16 bit
[0]	**ExtEN**	**EBI Enable** This bit is the functional enable bit for EBI. 0 = EBI function is disabled 1 = EBI function is enabled

External Bus Interface Timing CONTROL REGISTER (EXTIME)

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

31	30	29	28	27	26	25	24
Reserved				**ExtIR2R**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reversed							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**

ExtIW2X				Reversed	ExttAHD		
7	6	5	4	3	2	1	0
ExttACC					Reversed		

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

6.13 Flash Memory Controller (FMC)

6.13.1 Overview

NuMicro M051™ series equips with 64K/32K/16K/8K bytes on chip embedded Flash EEPROM for application program memory (APROM) that can be updated through ISP/IAP 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 M051™ series also provide additional 4K bytes DATA Flash for user to store some application depended data before chip power off in 64/32/16/8K bytes APROM model.

6.13.2 Features

- Run up to 50 MHz with zero wait state for continuous address read access
- 64/32/16/8KB 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)/In Application Program (IAP) to update on chip Flash EPROM
- In Circuit Program (ISP) via serial wire debug interface (SWD)

6.13.3 FMC 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 in the Figure 6.13-1.

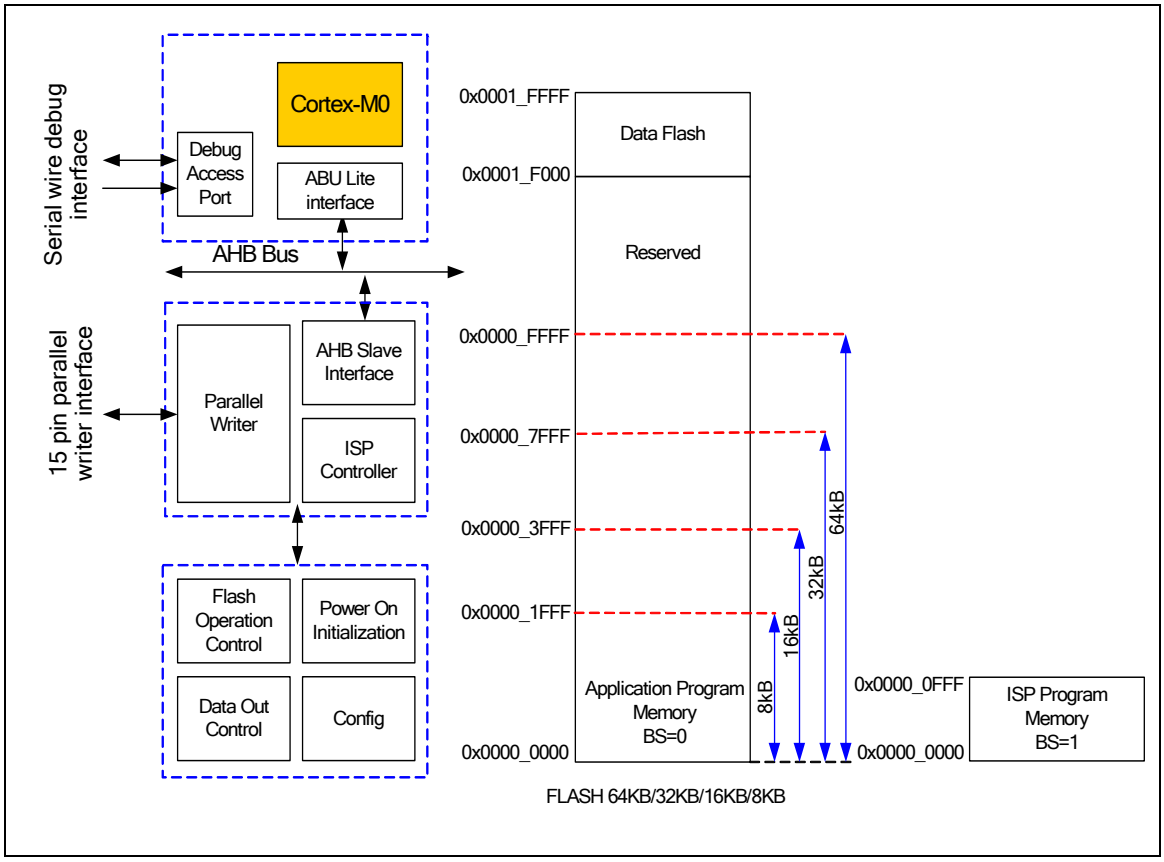


Figure 6.13-1 Flash Memory Control Block Diagram

6.13.4 FMC Organization

The NuMicro M051™ series flash memory consists of Program memory (64/32/16/8KB), data flash, ISP loader program memory, user configuration. User configuration block provides several bytes to control system logic, like flash security lock, boot select, brown out voltage level, data flash base address, ..., 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. The data flash start address and its size can defined by user depends on application. But for 64/32/16/8KB flash memory devices, its size is 4KB and start address is fixed at 0x0001_F000.

Block Name	Size	Start Address	End Address
AP-ROM	8/16/32/64KB	0x0000_0000	0x0000_1FFF (8KB) 0x0000_3FFF (16KB) 0x0000_7FFF (32KB) 0x0000_FFFF (64KB)
Data Flash	4KB	0x0001_F000	0x0001_FFFF
LD-ROM	4KB	0x0010_0000	0x0010_0FFF
User Configuration	1 Words	0x0030_0000	0x0030_0000

Table 6.13-1 Memory Address Map

The Flash memory organization is shown as below:

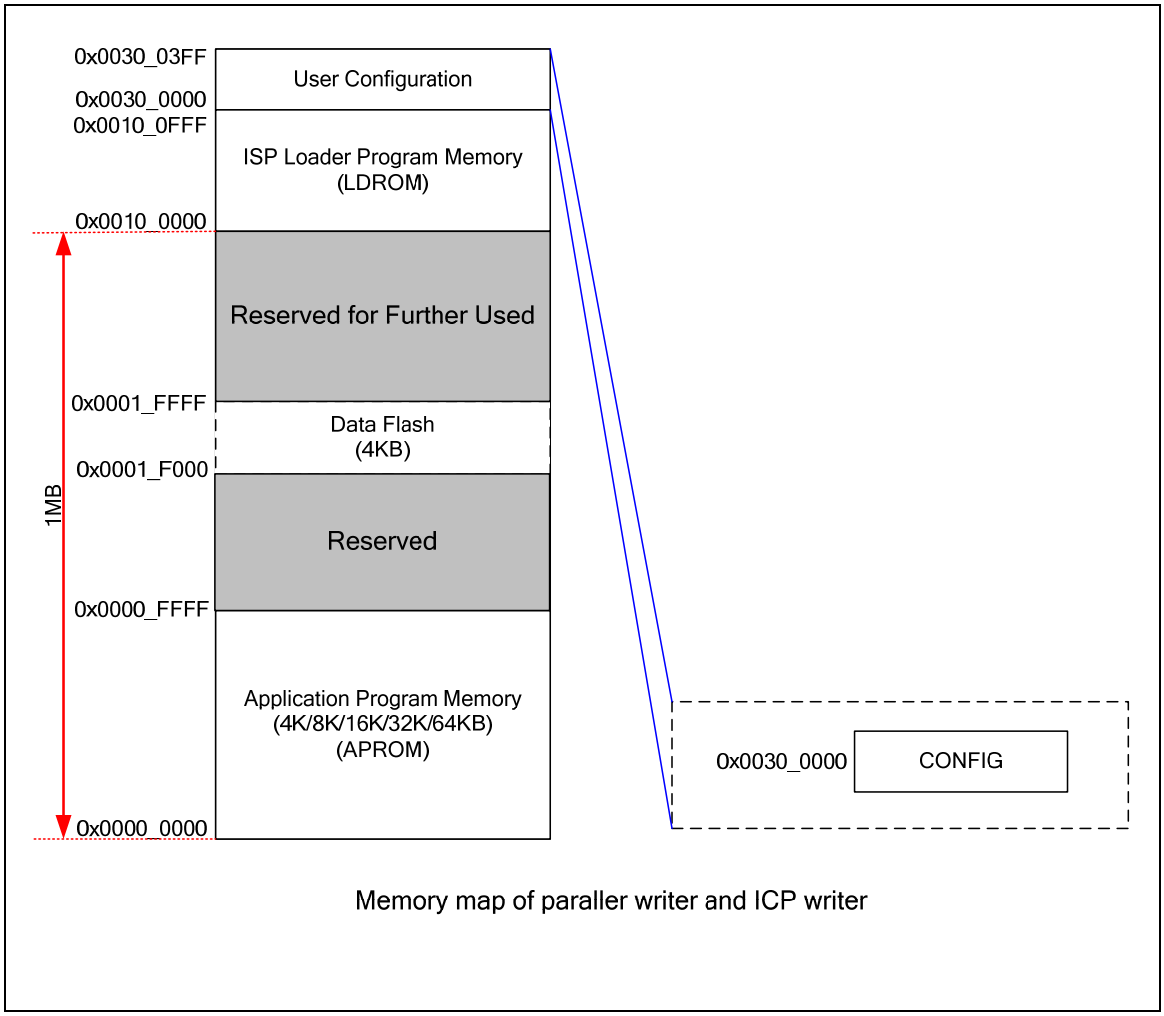


Figure 6.13-2 Flash Memory Organization

6.13.5 Boot Selection

NuMicro M051™ series provides in system programming (ISP) feature to enable user to update program memory when chip is mounted on PCB. A dedicated 4KB program memory is used to store ISP firmware. Users can select to start program fetch from APROM or LDROM by (CBS) in Config0.

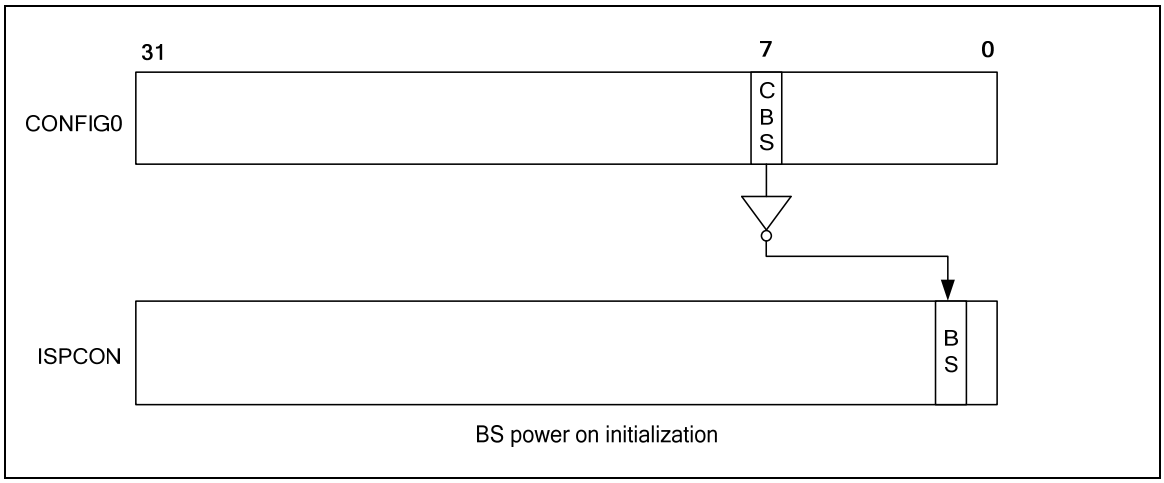
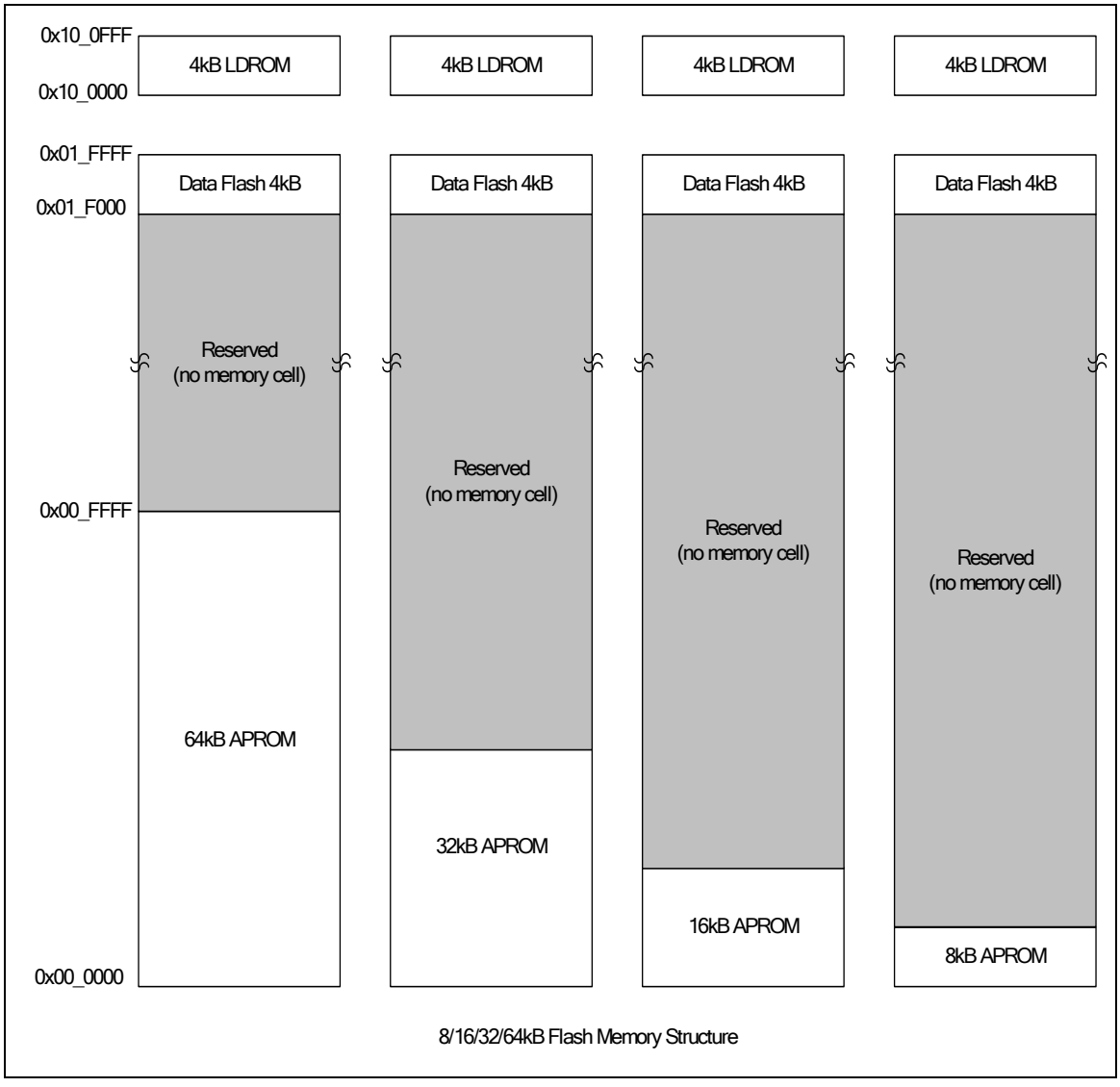


Figure 6.13-3 Boot Select (BS) for power-on action

6.13.6 Data Flash

NuMicro M051™ series provides data flash for user to store data. It is read/write thru ISP procedure. The 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 8/16/32/64KB flash memory device, data flash size is 4KB and start address is fixed at 0x0001_F000.



8/16/32/64kB Flash Memory Structure

Figure 6.13-4 Flash Memory Structure

6.13.7 In System Program (ISP)

Note: Before use this ISP function, please turn-on the ISP clock in ISP_EN(AHBCLK[2]).

Figure 6.13-5 shows the ISP Clock Source Block Diagram

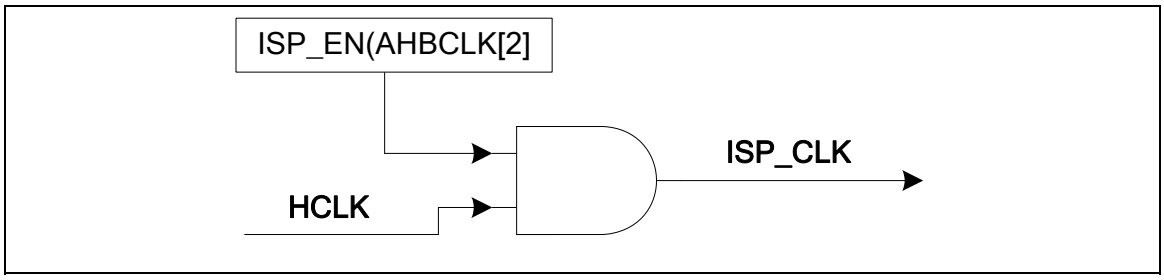


Figure 6.13-5 ISP Clock Source Control

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 M051™ 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 M051™ series. It makes users quite easy perform ISP through Nuvoton ISP tool.

ISP Procedure

NuMicro M051™ series supports boot 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 s/w 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 software 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 finish, during this period; peripheral still keeps working as usual. If any interrupt request occur, CPU will not service it till ISP operation finish.

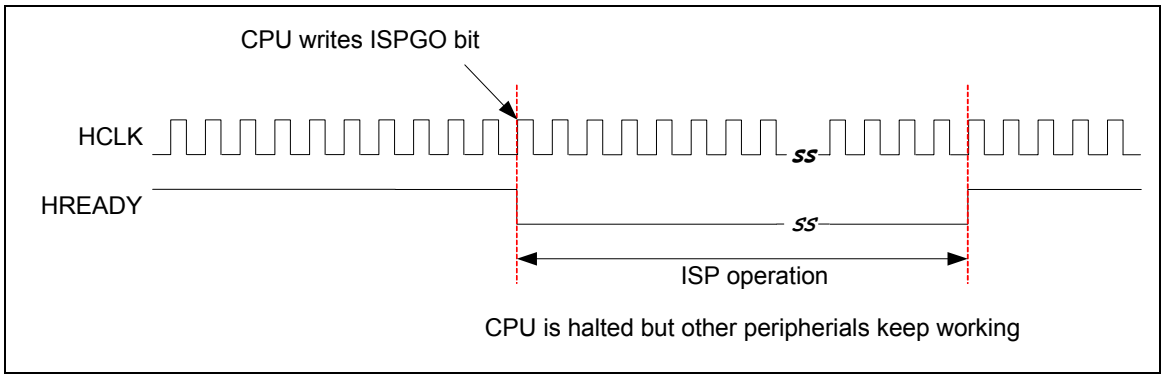


Figure 6.13-6 ISPGo Timing Diagram

Note that NuMicro M051™ series allows user to update CONFIG value by ISP, but for application program code security issue, software is required to erase APROM by page erase before erase CONFIG. Otherwise, erase CONFIG will not be allowed.

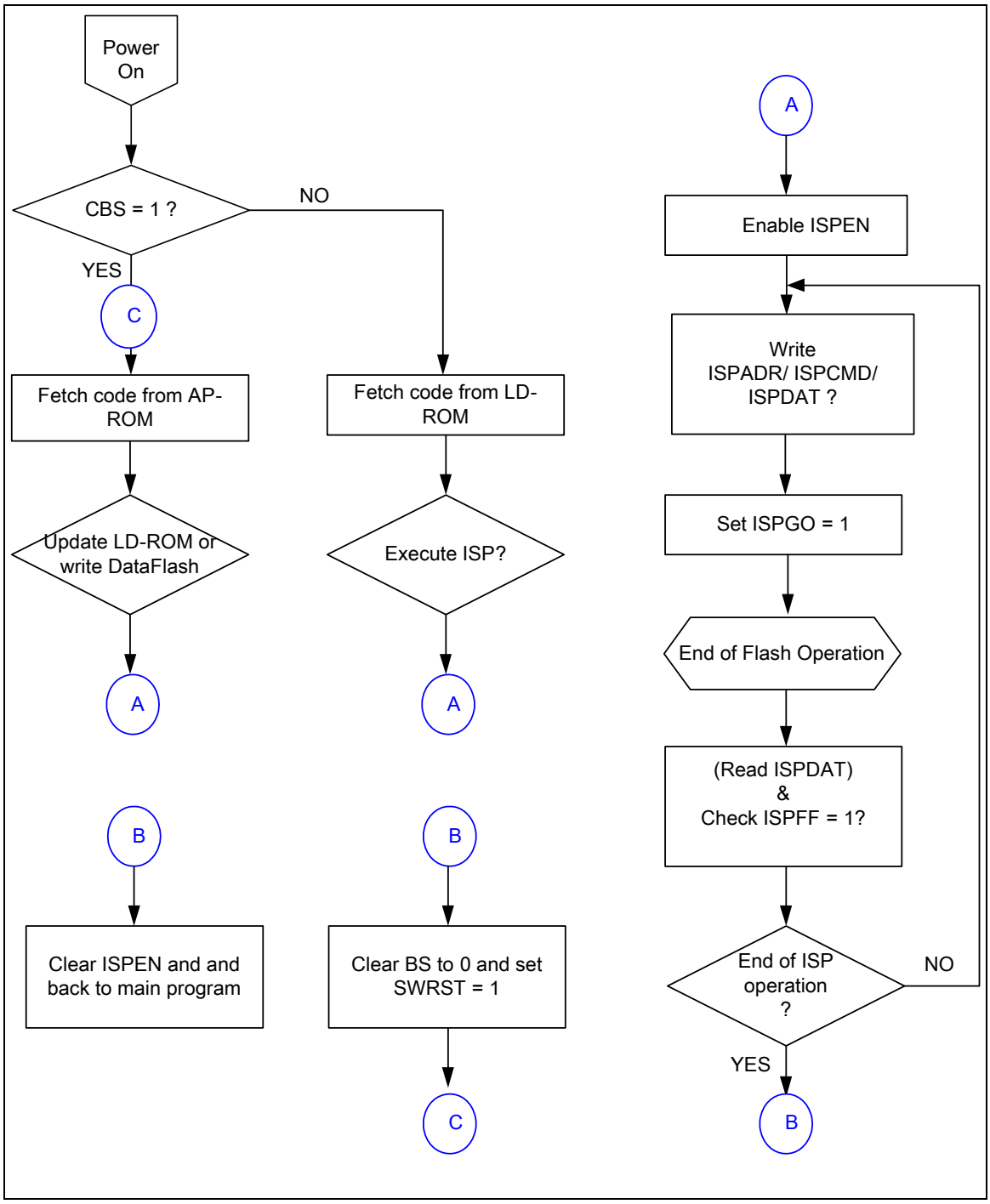


Figure 6.13-7 ISP Software Programming Flow

ISP Mode	ISPCMD			ISPADR			ISPDAT
	FOEN	FCEN	FCTRL[3:0]	A21	A20	A[19:0]	D[31:0]
Standby	1	1	X	x	x	x	x
Read Company ID	0	0	1011	x	x	x	Data out D[31:0] = 0x0000_00DA
FLASH Page Erase	1	0	0010	0	A20[#1]	Address in A[19:0]	x
FLASH Program	1	0	0001	0	A20[#1]	Address in A[19:0]	Data in D[31:0]
FLASH Read	0	0	0000	0	A20[#1]	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 6.13-2 ISP Mode

Note1: A20=0 for APROM and DATA, A20=1, for LDROM

6.13.8 FMC Controller Registers 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
DFBADR	FMC_BA+0x014	R	Data Flash Start Address	0x0000_0000 0x0001_F000
FATCON	FMC_BA+0x018	R/W	Flash Access Window Control Register	0x0000_0000

6.13.9 FMC Controller Registers 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	**ET2**	**ET1**	**ET0**	**Reserved**	**PT2**	**PT1**	**PT0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SWRST	**ISPFF**	**LDUEN**	**CFGUEN**	**Reserved**		**BS**	**ISPEN**

Bits	Descriptions	
[31:15]	**Reserved**	Reserved
[14:12]	**ET[2:0]**	**Flash Erase Time** 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

<table>
<tr><td>[8:10]</td><td>PT[2:0]</td><td>Flash Program Time
<table>
<tr><td>PT[2]</td><td>PT[1]</td><td>PT[0]</td><td>Program Time (us)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>40</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>45</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>50</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>55</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>20</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>25</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>30</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>35</td></tr>
</table></td></tr>
<tr><td>[7]</td><td>SWRST</td><td>Software Reset
Writing 1 to this bit to start software reset.
It is cleared by hardware after reset is finished.</td></tr>
<tr><td>[6]</td><td>ISPFF</td><td>ISP Fail Flag
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) Destination address is illegal, such as over an available range.
Note: Write 1 to clear this bit to zero.</td></tr>
<tr><td>[5]</td><td>LDUEN</td><td>LDROM Update Enable
LDROM update enable bit.
1 = LDROM can be updated when the MCU runs in APROM.
0 = LDROM cannot be updated</td></tr>
<tr><td>[4]</td><td>CFGUEN</td><td>Config Update Enable
Writing this bit to 1 enables s/w to update Config value by ISP procedure regardless of program code is running in APROM or LDROM.
1 = Config update enable
0 = Config update disable</td></tr>
<tr><td>[2]</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>[1]</td><td>BS</td><td>Boot Select
This bit is protected bit. Set/clear this bit to select next booting from LDROM/APROM, respectively. This bit also functions as MCU booting status flag, which can be used to check where MCU booted from. This bit is initiated with the inversed value of CBS</td></tr>
</table>

		in Config0 after power-on reset; It keeps the same value at other reset. 1 = boot from LDROM 0 = boot from APROM
[0]	**ISPEN**	**ISP Enable** **This bit is protected 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 M051™ series equips with a 32kx32 embedded flash, it supports word program only. ISPADR[1:0] must be kept 2'b00 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**	**FCTRL3**	**FCTRL2**	**FCTRL1**	**FCTRL0**

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

Operation Mode	FOEN	FCEN	FCTRL[3:0]			
Standby	1	1	0	0	0	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 finish. 1 = ISP is on going 0 = ISP done

Data Flash Base Address Register (DFBADR)

Register	Address	R/W/C	Description	Reset Value
DFBADR	FMC_BA+0x14	R	Data flash Base Address	0x0001_F000

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

Bits	Descriptions	
[31:0]	**DFBADR**	**Data Flash Base Address** This register indicates data flash start address. It is a read only register. For 8/16/32/64KB flash memory device, the data flash size is 4KB and it start address is fixed at 0x0001_F000 by hardware internally.

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			L_SPEED	FATS[2:0]			FPSEN

Bits	Descriptions	
[31:5]	**Reserved**	Reserved
[4]	**L_SPEED**	**Flash Low Speed Mode Enable** 1 = Flash access always no wait state (zero wait state) 0 = Insert wait state while Flash access discontinued address. Note: Set this bit only when HCLK ≤ 25 MHz. If HCLK > 25 MHz, CPU will fetch wrong code and cause fail result.
[3:1]	**FATS**	**Flash Access Time Window Select** These bits are used to decide flash sense amplifier active duration.

FATS	Access Time window (ns)
000	40 (default)
001	50
010	60
011	70
100	80
101	90

		110 100 111 Reserved
[0]	**FPSEN**	**Flash Power Save Enable** On chip flash memory access time is around 40ns, if CPU clock is slower than 50 MHz, then s/w can enable flash power saving function. 1 = Enable flash power saving 0 = Disable flash power saving

7 USER CONFIGURATION

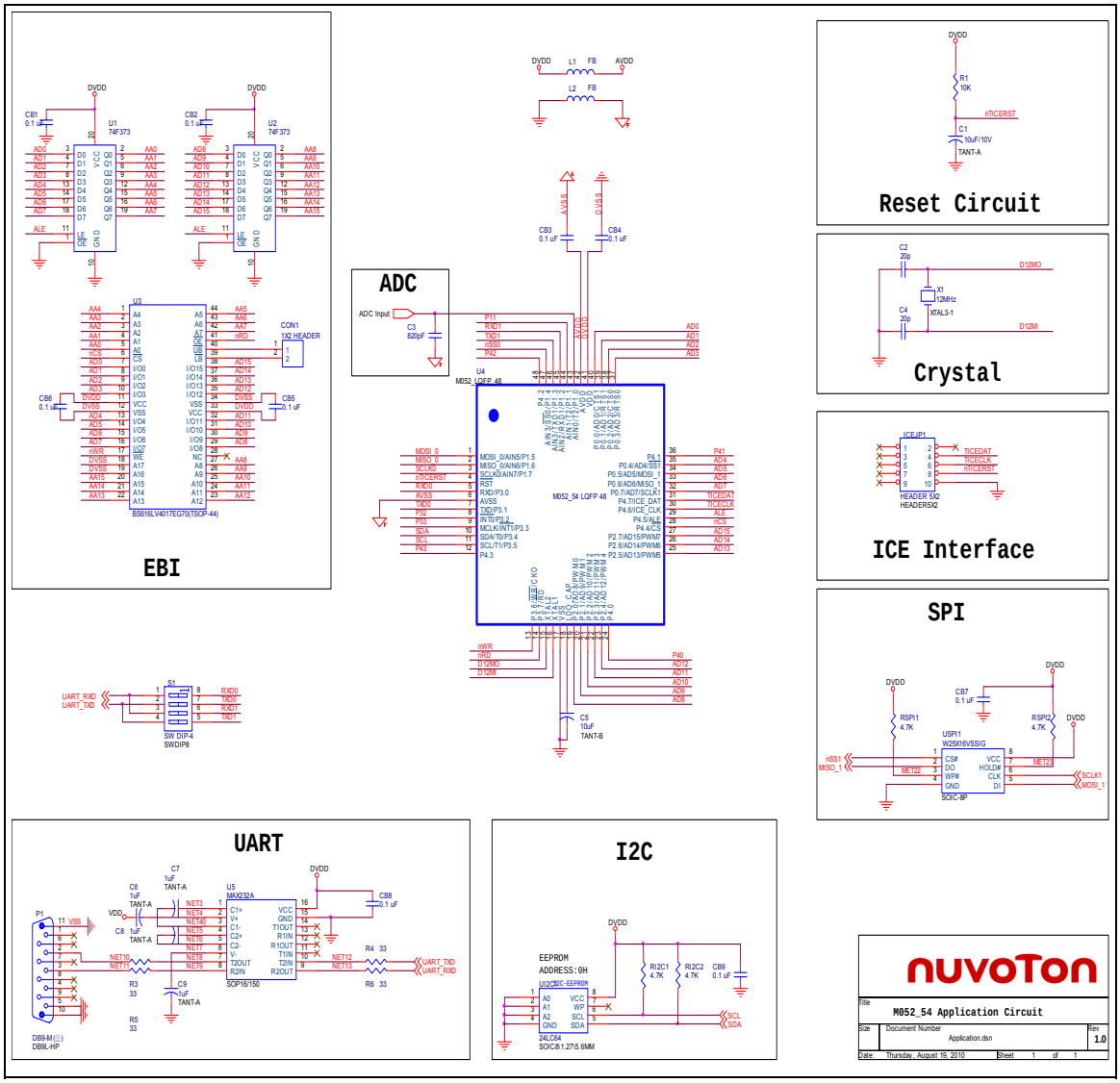
CONFIG (Address = 0x0030_0000)

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 **for further used**
[28]	**CKF**	**XT1 Clock Filter Enable** 0 = Disable XT1 clock filter 1 = Enable XT1 clock filter
[27]	**Reserved**	Reserved for further used
[26:24]	**CFOSC**	**CPU Clock Source Selection After Reset** FOSC[2:0] / Clock Source 000 / External crystal clock (4 ~ 24 MHz) 111 / Internal RC 22.1184 MHz 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**	**Brown Out Detector Enable** 0= Enable brown out detect after power on 1= Disable brown out detect after power on
[22:21]	**CBOV1-0**	**Brown Out Voltage Selection** **CBOV1** / **CBOV0** / **Brown out voltage**

		1	1	4.5V	
		1	0	3.8V	
		0	1	2.7V	
		0	0	2.2V	
[20]	**CBORST**	**Brown Out Reset Enable** 0 = Enable brown out reset after power on 1 = Disable brown out reset after power on			
[19:8]	**Reserved**	Reserved for further used			
[7]	**CBS**	**Config Boot Selection** 0 = Chip boot from LDROM 1 = Chip boot from APROM			
[6:2]	**Reserved**	Reserved for further used			
[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 thru serial debug interface. Others data is locked as 0xFFFFFFFF. ISP can read data anywhere regardless of LOCK bit value.			
[0]	**Reserved**	Reserved			

8 TYPICAL APPLICATION CIRCUIT



9 ELECTRICAL CHARACTERISTICS

9.1 Absolute Maximum Ratings

SYMBOL	PARAMETER	MIN	MAX	UNIT
DC Power Supply	VDD–VSS	-0.3	+7.0	V
Input Voltage	VIN	VSS-0.3	VDD+0.3	V
Oscillator Frequency	$1/t_{CLCL}$	0	40	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.

9.2 DC Electrical Characteristics

(VDD-VSS=2.5~5.5V, TA = 25°C, F_{OSC} = 50 MHz unless otherwise specified.)

PARAMETER	SYM.	SPECIFICATION				TEST CONDITIONS
		MIN.	TYP.	MAX.	UNIT	
Operation voltage	V_{DD}	2.5		5.5	V	V_{DD} =2.5V ~ 5.5V up to 50 MHz
Power Ground	V_{SS} AV_{SS}	-0.3			V	
LDO Output Voltage	V_{LDO}	-10%	2.45	+10%	V	V_{DD} > 2.7V
Band Gap Analog Input	V_{BG}	-5%	1.26	+5%	V	V_{DD} =2.5V ~ 5.5V
Analog Operating Voltage	AV_{DD}	0		V_{DD}	V	
Operating Current Normal Run Mode @ 50 MHz	I_{DD1}		32		mA	V_{DD}= 5.5V@50 MHz, enable all IP and PLL, XTAL=12 MHz
	I_{DD2}		24		mA	V_{DD}=5.5V@50 MHz, disable all IP and enable PLL, XTAL=12 MHz
	I_{DD3}		31		mA	V_{DD} = 3V@50 MHz, enable all IP and PLL, XTAL=12 MHz
	I_{DD4}		23		mA	V_{DD} = 3V@50 MHz, disable all IP and enable PLL, XTAL=12 MHz
Operating Current Normal Run Mode @ 12 MHz	I_{DD5}		17		mA	V_{DD} = 5.5V@ 12MHz, enable all IP and disable PLL, XTAL=12 MHz
	I_{DD6}		14		mA	V_{DD} = 5.5V@12 MHz, disable all IP and disable PLL, XTAL=12 MHz
	I_{DD7}		16		mA	V_{DD} = 3V@12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I_{DD8}		13		mA	V_{DD} = 3V@12 MHz, disable all IP and disable PLL, XTAL=12 MHz
Operating Current Normal Run Mode @ 4 MHz	I_{DD9}		12		mA	V_{DD} = 5.5V@4 MHz, enable all IP and disable PLL, XTAL=4MHz
	I_{DD10}		10		mA	V_{DD} = 5.5V@4 MHz, disable all IP and disable PLL, XTAL=4MHz
	I_{DD11}		10		mA	V_{DD} = 3V@4 MHz, enable all IP and disable PLL, XTAL=4MHz
	I_{DD12}		9		mA	V_{DD} = 3V@4 MHz, disable all IP and disable PLL, XTAL=4 MHz

Operating Current Idle Mode @ 50 MHz	I_{IDLE1}		19		mA	V_{DD}= 5.5V@50 MHz, enable all IP and PLL, XTAL=12 MHz
	I_{IDLE2}		11		mA	V_{DD}=5.5V@50 MHz, disable all IP and enable PLL, XTAL=12 MHz
	I_{IDLE3}		18		mA	V_{DD} = 3V@50 MHz, enable all IP and PLL, XTAL=12 MHz
	I_{IDLE4}		10		mA	V_{DD} = 3V@50 MHz, disable all IP and enable PLL, XTAL=12 MHz
Operating Current Idle Mode @ 12 MHz	I_{IDLE5}		10		mA	V_{DD} = 5.5V@12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I_{IDLE6}		7		mA	V_{DD} = 5.5V@12 MHz, disable all IP and disable PLL, XTAL=12 MHz
	I_{IDLE7}		9		mA	V_{DD} = 3V@12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I_{IDLE8}		6		mA	V_{DD} = 3V@12 MHz, disable all IP and disable PLL, XTAL=12 MHz
Operating Current Idle Mode @ 4 MHz	I_{IDLE9}		5		mA	V_{DD} = 5.5V@4 MHz, enable all IP and disable PLL, XTAL=4 MHz
	I_{IDLE10}		4		mA	V_{DD} = 5.5V@4 MHz, disable all IP and disable PLL, XTAL=4 MHz
	I_{IDLE11}		4		mA	V_{DD} = 3V@4 MHz, enable all IP and disable PLL, XTAL=4 MHz
	I_{IDLE12}		3		mA	V_{DD} = 3V@4 MHz, disable all IP and disable PLL, XTAL=4 MHz
Standby Current Power-down Mode (Deep Sleep Mode)	I_{PWD1}		15		µA	V_{DD} = 5.5V, No load @ Disable BOV function
	I_{PWD2}		11		µA	V_{DD} = 3.0V, No load @ Disable BOV function
Input Current P0/1/2/3/4 (Quasi-bidirectional mode)	I_{IN1}		-50	-60	µA	V_{DD} = 5.5V, V_{IN} = 0.4V
Input Leakage Current P0/1/2/3/4	I_{LK}	-2	-	+2	µA	V_{DD} = 5.5V, 0<V_{IN}<V_{DD}
Logic 1 to 0 Transition Current P0/1/2/3/4 (Quasi-bidiretional mode)	I_{TL} [3]	-650	-	-200	µA	V_{DD} = 5.5V, V_{IN}<2.0V
Input Low Voltage P0/1/2/3/4 (TTL input)	V_{IL1}	-0.3	-	0.8	V	V_{DD} = 4.5V
		-0.3	-	0.6		V_{DD} = 2.5V
Input High Voltage P0/1/2/3/4 (TTL input)	V_{IH1}	2.0	-	V_{DD} +0.2	V	V_{DD} = 5.5V
		1.5	-	V_{DD} +0.2		V_{DD} =3.0V
Input Low Voltage XT1[*2]	V_{IL3}	0	-	0.8	V	V_{DD} = 4.5V
		0	-	0.4		V_{DD} = 3.0V
Input High Voltage XT1[*2]	V_{IH3}	3.5	-	V_{DD} +0.2	V	V_{DD} = 5.5V
		2.4	-	V_{DD} +0.2		V_{DD} = 3.0V

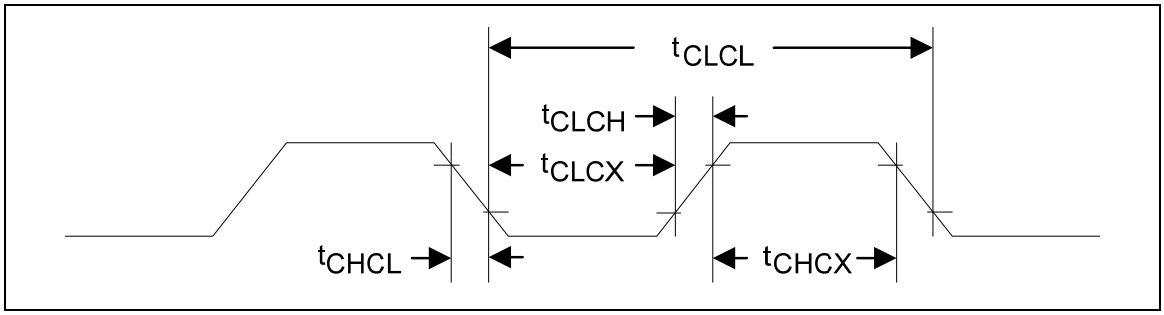
Negative going threshold (Schmitt input), /RST	V_{ILS}	-0.5	-	$0.3V_{DD}$	V	
Positive going threshold (Schmitt input), /RST	V_{IHS}	$0.7V_{DD}$	-	V_{DD}+0.5	V	
Internal /RST pin pull up resistor	R_{RST}	40		150	KΩ	
Negative going threshold (Schmitt input), P0/1/2/3/4	V_{ILS}	-0.5	-	$0.2V_{DD}$	V	
Positive going threshold (Schmitt input), P0/1/2/3/4	V_{IHS}	$0.4V_{DD}$	-	V_{DD} +0.5	V	
Source Current P0/1/2/3/4 (Quasi-bidirectional Mode)	I_{SR11}	-300	-370	-450	μA	V_{DD} = 4.5V, V_S = 2.4V
	I_{SR12}	-50	-70	-90	μA	V_{DD} = 2.7V, V_S = 2.2V
	I_{SR12}	-40	-60	-80	μA	V_{DD} = 2.5V, V_S = 2.0V
Source Current P0/1/2/3/4 (Push-pull Mode)	I_{SR21}	-20	-24	-28	mA	V_{DD} = 4.5V, V_S = 2.4V
	I_{SR22}	-4	-6	-8	mA	V_{DD} = 2.7V, V_S = 2.2V
	I_{SR22}	-3	-5	-7	mA	V_{DD} = 2.5V, V_S = 2.0V
Sink Current P0/1/2/3/4 (Quasi-bidirectional and Push-pull Mode)	I_{SK1}	10	16	20	mA	V_{DD} = 4.5V, V_S = 0.45V
	I_{SK1}	7	10	13	mA	V_{DD} = 2.7V, V_S = 0.45V
	I_{SK1}	6	9	12	mA	V_{DD} = 2.5V, V_S = 0.45V
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	$V_{BO3.8}$	3.7	3.8	3.9	V	
Brownout voltage with BOV_VL [1:0] =11b	$V_{BO4.5}$	4.4	4.5	4.6	V	
Hysteresis range of BOD voltage	V_{BH}	30	-	150	mV	V_{DD} = 2.5V~5.5V

Notes:

1. /RST pin is a Schmitt trigger input.
2. XTAL1 is a CMOS input.
3. Pins of P0, P1, P2, P3 and P4 can source a transition current when they are being externally driven from 1 to 0. In the condition of V_{DD}=5.5V, 5he transition current reaches its maximum value when Vin approximates to 2V .

9.3 AC Electrical Characteristics

9.3.1 External Crystal



Note: Duty cycle is 50%.

PARAMETER	SYMBOL	MIN.	TYP.	MAX.	UNITS	CONDITION
Clock High Time	t_{CHCX}	20	-	125	nS	
Clock Low Time	t_{CLCX}	20	-	125	nS	
Clock Rise Time	t_{CLCH}	-	-	10	nS	
Clock Fall Time	t_{CHCL}	-	-	10	nS	

9.3.2 External Oscillator

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Input clock frequency	External crystal	4	12	24	MHz
Temperature	-	-40	-	85	℃
V_{DD}	-	2.5	5	5.5	V
Operating current	12 MHz@ V_{DD} = 5V	-	5	-	mA

9.3.3 Typical Crystal Application Circuits

CRYSTAL	C1	C2
4 MHz ~ 24 MHz	Optional (Depend on crystal specification)	

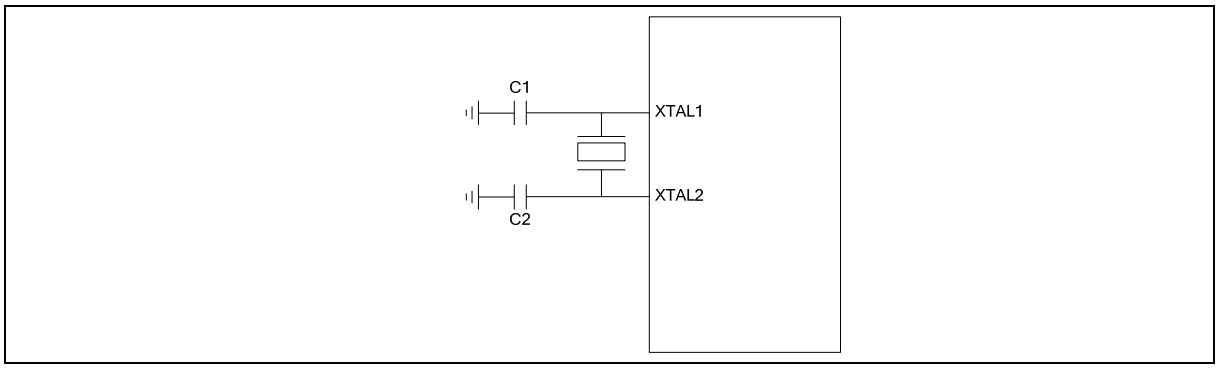


Figure 9.3-1 Typical Crystal Application Circuit

9.3.4 Internal 22.1184 MHz RC Oscillator

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Supply voltage[1]	-	2.5	-	5.5	V
Center Frequency	-	-	22.1184		MHz
Calibrated Internal Oscillator Frequency	+25 C; V_{DD} =5V	-1	-	+1	%
	-40 C~+85 C; V_{DD}=2.5V~5.5V	-3	-	+3	%
Accuracy of Un-calibrated Internal Oscillator Frequency	-40 C~+85 C; V_{DD}=2.5V~5.5V	-25	-	+25	%
Operating current	V_{DD} =5V	-	500	-	uA

9.3.5 Internal 10kHz RC Oscillator

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Supply voltage[1]	-	2.5	-	5.5	V
Center Frequency	-	-	10	-	kHz
Calibrated Internal Oscillator Frequency	+25 C; V_{DD} =5V	-30	-	+30	%
	-40 C~+85 C; V_{DD}=2.5V~5.5V	-50	-	+50	%
Operating current	V_{DD} =5V	-	5	-	uA

Notes:

1. Internal operation voltage comes form LDO.

9.4 Analog Characteristics

9.4.1 Specification of 600kHz sps 12-bit SARADC

PARAMETER	SYM.	MIN.	TYP.	MAX.	UNIT
Resolution	-	-	-	12	Bit
Differential nonlinearity error	DNL	-	±1.2	-	LSB
Integral nonlinearity error	INL	-	±1.5	-	LSB
Offset error	EO	-	+4	10	LSB
Gain error (Transfer gain)	EG	-	+7	1.005	-
Monotonic	-		Guaranteed		-
ADC clock frequency	FADC	-	-	20	MHz
Calibration time	TCAL	-	127	-	Clock
Sample time	TS	-	7	-	Clock
Conversion time	TADC	-	13	-	Clock
Sample rate	FS	-	-	600	k sps
Supply voltage	V_{LDO}	-	2.5	-	V
	VADD	3	-	5.5	V
Supply current (Avg.)	IDD	-	0.5	-	mA
	IDDA	-	1.5	-	mA
Input voltage range	VIN	0	-	AVDD	V
Capacitance	CIN	-	5	-	pF

9.4.2 Specification of LDO & Power management

PARAMETER	MIN	TYP	MAX	UNIT	NOTE
Input Voltage	2.7	5	5.5	V	V_{DD} input voltage
Output Voltage (bypass=0)	-10%	2.45	+10%	V	LDO output voltage
Output Voltage (bypass=1)	-10%	Input Voltage	+10%	V	Input Voltage < 2.7V
Quiescent Current (PD=0, bypass=0)	-	100	-	uA	
Quiescent Current (PD=1, bypass=0)	-	5	-	uA	
Quiescent Current (PD=1, bypass=1)	-	5	-	uA	
Iload (PD=0)	-	-	100	mA	
Iload (PD=1)	-	-	100	uA	
Cbp	-	1u	-	F	Resr=1ohm
Cload	-	250p	-	F	

Note:

1. It is recommended that a 10uF or higher capacitor and a 100nF bypass capacitor are connected between VDD and the closest VSS pin of the device.
2. For ensuring power stability, a 4.7uF or higher capacitor must be connected between LDO pin and the closest VSS pin of the device. Also a 100nF bypass capacitor between LDO and VSS help suppressing output noise.

9.4.3 Specification of Low Voltage Reset

PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
Operation voltage	-	1.7	-	5.5	V
Quiescent current	VDD5V=5.5V	-	-	5	uA
Threshold voltage	Temperature=25°	1.7	2.0	2.3	V
	Temperature=-40°	-	2.4	-	V
	Temperature=85°	-	1.6	-	V
Hysteresis	-	0	0	0	V

9.4.4 Specification of Brownout Detector

Parameter	Condition	Min.	Typ.	Max.	Unit
Operation voltage	-	2.5	-	5.5	V
Quiescent current	AVDD=5.5V	-	-	125	μA
Temperature	-	-40	25	85	℃
Brown-out voltage	BOV_VL[1:0]=11	4.4	4.5	4.6	V
	BOV_VL [1:0]=10	3.7	3.8	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	-	30m	-	150m	V

9.4.5 Specification of Power-On Reset (5V)

Parameter	Condition	Min.	Typ.	Max.	Unit
Reset voltage	V+	-	2	-	V
Quiescent current	Vin>reset voltage	-	1	-	nA

9.5 SPI Dynamic characteristics

Symbol	Parameter	Min	Typ	Max	Unit
SPI master mode (VDD = 4.5V ~ 5.5V, 30pF loading Capacitor)					
t_{DS}	Data setup time	35	-	-	ns
t_{DH}	Data hold time	0	-	-	ns
t_V	Data output valid time	-	-	10	ns
SPI master mode (VDD = 3.0V ~ 3.6V, 30pF loading Capacitor)					
t_{DS}	Data setup time	45	-	-	ns
t_{DH}	Data hold time	0	-	-	ns
t_V	Data output valid time	-	-	16	ns
SPI slave mode (VDD = 4.5V ~ 5.5V, 30pF 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+40	ns
SPI slave mode (VDD = 3.0V ~ 3.6V, 30pF loading Capacitor)					
t_{DS}	Data setup time	0	-	-	ns
t_{DH}	Data hold time	2*PCLK+5	-	-	ns
t_V	Data output valid time	-	-	2*PCLK+50	ns

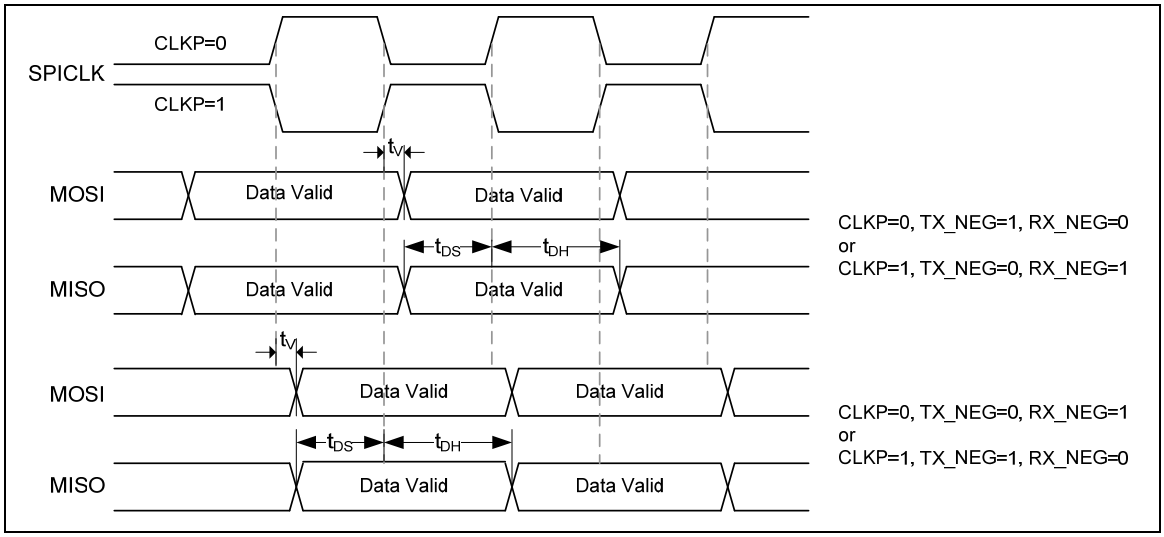


Figure 9.5-1 SPI Master timing

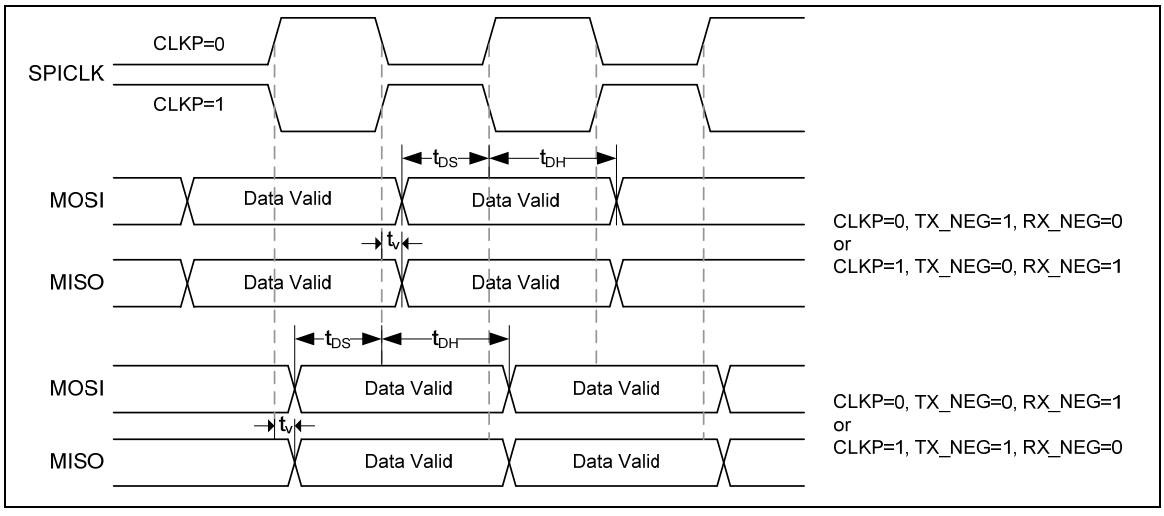
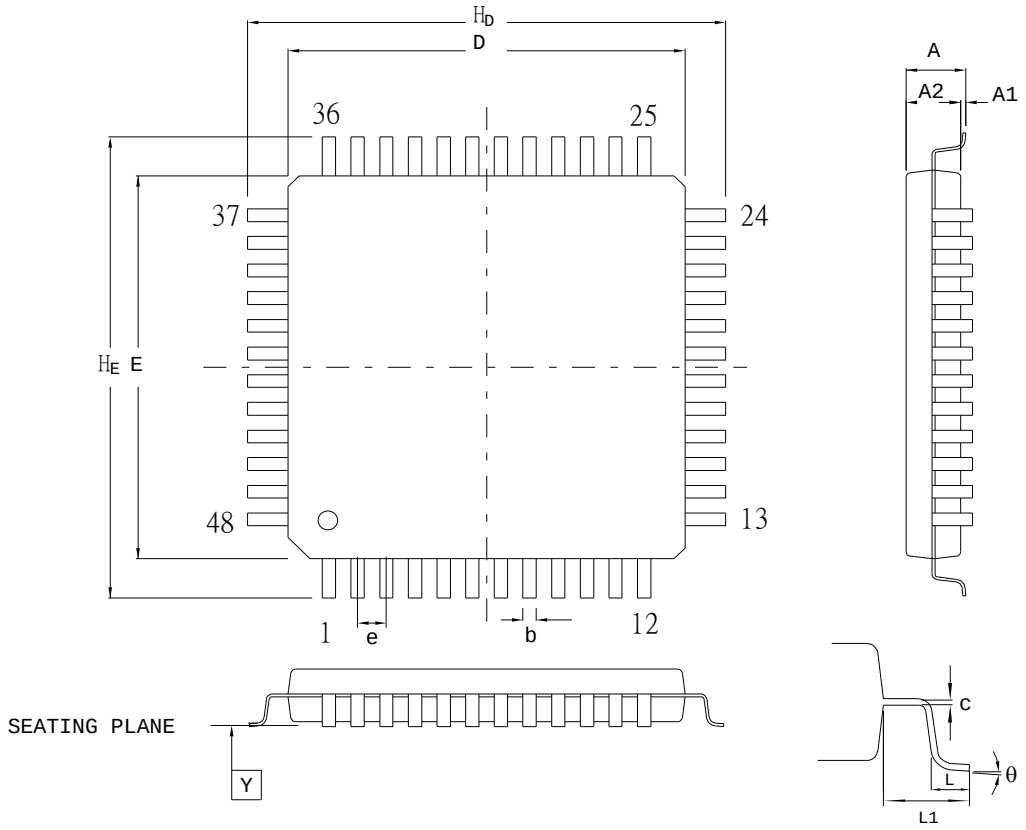


Figure 9.5-2 SPI Slave timing

10 PACKAGE DIMENSIONS

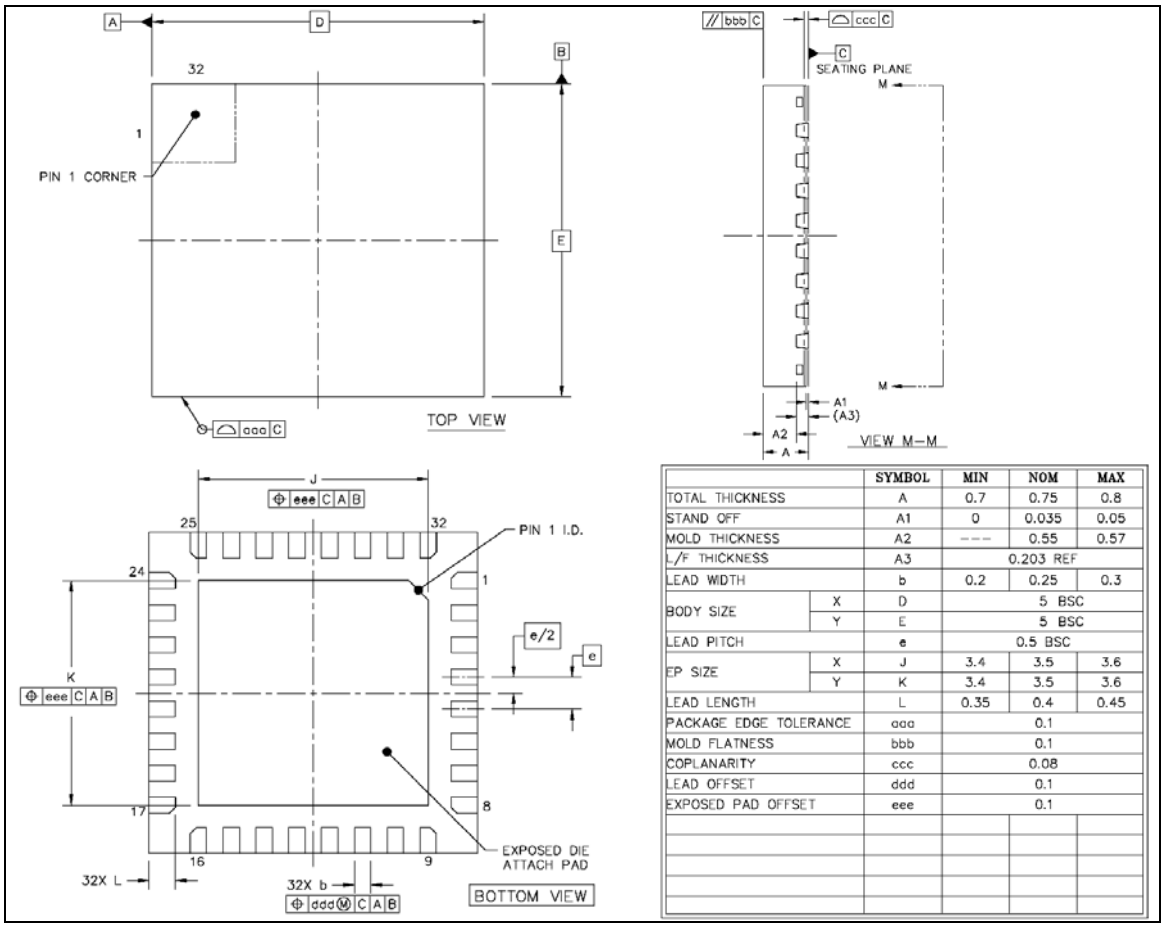
10.1 LQFP-48 (7x7x1.4mm² Footprint 2.0mm)



Controlling dimension : Millimeters

Symbol	Dimension in inch			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	—	—	—	—	—	—
A_1	0.002	0.004	0.006	0.05	0.10	0.15
A_2	0.053	0.055	0.057	1.35	1.40	1.45
b	0.006	0.008	0.010	0.15	0.20	0.25
c	0.004	0.006	0.008	0.10	0.15	0.20
D	0.272	0.276	0.280	6.90	7.00	7.10
E	0.272	0.276	0.280	6.90	7.00	7.10
e	0.014	0.020	0.026	0.35	0.50	0.65
H_D	0.350	0.354	0.358	8.90	9.00	9.10
H_E	0.350	0.354	0.358	8.90	9.00	9.10
L	0.018	0.024	0.030	0.45	0.60	0.75
L_1	—	0.039	—	—	1.00	—
Y	—	—	0.004	—	—	0.10
θ	0°	—	7 °	0°	—	7 °

10.2 QFN-33 (5X5 mm^2, Thickness 0.8mm, Pitch 0.5 mm)



		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	

11 REVISION HISTORY

VERSION	DATE	PAGE	DESCRIPTION
V1.0	Aug 23, 2010	-	Initial issued
V1.1	Sep. 8, 2010	6.10	1. Modify FIFO size of UART from 16bytes to 15bytes. 2. Reserve Bus Error Interrupt of UART.
V1.2	Sep. 14, 2010	6.10	1. Reverse the definition of CTS trigger level (UA_MSR.8).

Important Notice

Nuvoton products are not designed, intended, authorized or warranted for use as components in systems or equipment intended for surgical implantation, atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, or for other applications intended to support or sustain life. Further more, Nuvoton products are not intended for applications wherein failure of Nuvoton products could result or lead to a situation wherein personal injury, death or severe property or environmental damage could occur.

Nuvoton customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nuvoton for any damages resulting from such improper use or sales.

Please note that all data and specifications are subject to change without notice. All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.