ARM Cortex®-M0

32-BIT MICROCONTROLLER

NuMicro™ M058S Series
Technical Reference Manual

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro™ microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

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

www.nuvoton.com

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 GENERAL DESCRIPTION

The NuMicro™ M058S series 32-bit microcontroller is embedded with ARM® Cortex®-M0 core for industrial control and applications which need rich communication interfaces. The M058S series includes the following part numbers: M058SSAN, M058SLAN and M058SZAN.

The NuMicro™ M058S series can run up to 50 MHz and operate at 2.5V ~ 5.5V, -40℃ ~ 85℃, and thus can afford to support a variety of industrial control and applications which need high CPU performance. The M058S series offers 32 KB flash, 4 KB Data Flash, 4 KB flash for the ISP, and 4 KB SRAM.

Many system level peripheral functions, such as I/O Port, Timer, UART, SPI, I^{2}C, PWM, ADC, Watchdog Timer, and Window Watchdog. These useful functions make the M058S series powerful for a wide range of applications.

Additionally, the NuMicro™ M058S series is equipped with ISP (In-System Programming) and ICP (In-Circuit Programming) functions, and IAP (In-Application Programming), 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 running 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 two watchpoints/four breakpoints
- Wide Operating Voltage Range: 2.5V to 5.5V
- Memory
  - 32 KB Flash for program memory (APROM)
  - 4 KB Flash for data memory (Data Flash)
  - 4 KB Flash for loader (LDROM)
  - 4 KB SRAM for internal scratch-pad RAM (SRAM)
- Clock Control
  - Programmable system clock source
  - 22.1184 MHz internal oscillator
  - 4~24 MHz external crystal input
  - 10 kHz low-power oscillator for Watchdog Timer and wake-up in Sleep mode
  - PLL allows CPU operation up to the maximum 50 MHz
- I/O Port
  - Up to 55 general-purpose I/O (GPIO) pins for LQFP-64 package
  - Four I/O modes:
    - Quasi-bidirectional
    - Push-Pull output
    - Open-Drain output
    - Input only with high impendence
  - TTL/Schmitt trigger input selectable
  - I/O pin can be configured as interrupt source with edge/level setting
  - Configurable I/O mode after POR
- 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
  - Provides event counter function
  - Provides external capture/reset counter function
  - Additional functions:
    - Two more timer clock sources from external trigger and internal 10 kHz
    - TIMER wake-up function
    - External capture input source selected from TxEX
    - Toggle mode output pins selected from TxEX or TMx
    - Inter-Timer trigger mode
- WDT (Watchdog Timer)
  - Multiple clock sources

  - Supports wake-up from Power-down or Sleep mode
  - Interrupt or reset selectable on watchdog time-out
  - Time-out reset delay period time can be selected
- WWDT (Window Watchdog Timer)
  - 6-bit down counter with 11-bit pre-scale for wide range window selected
- PWM
  - Up to two built-in 16-bit PWM generators, providing four PWM outputs or two 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
  - Additional functions
    - Internal 10 kHz to PWM clock source
    - Polar inverse function
    - Center-aligned type function
    - Timer duty interrupt enable function
    - Two kinds of PWM interrupt period/duty type selection
    - Period/duty trigger ADC function
- UART
  - 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
  - Supports LIN function
- SPI
  - Supports Master/Slave mode
  - Full-duplex synchronous serial data transfer
  - Provides 3 wire function
  - Variable length of transfer data from 8 to 32 bits
  - MSB or LSB first data transfer
  - Supports Byte Suspend mode in 32-bit transmission
  - Additional functions
    - PLL clock source
    - 4-level depth FIFO buffer for better performance and flexibility in SPI Burst Transfer mode
- I^{2}C
  - Up to two sets of I^{2}C modules
  - 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 addresses with mask option)
- ADC
  - 12-bit SAR ADC
  - Up to 8-ch single-ended input or 4-ch differential input
  - Supports Single mode/Burst mode/Single-cycle Scan mode/Continuous Scan mode
  - Supports 2' complement/un-signed format in differential mode conversion results
  - Each channel with an individual result register
  - Supports conversion value monitoring (or comparison) for threshold voltage detection
  - Conversion started either by software trigger or external pin trigger
  - Additional functions
    - A/D conversion started by PWM center-aligned trigger or edge-aligned trigger
    - PWM trigger delay function
- ISP (In-System Programming) and ICP (In-Circuit Programming)
- IAP (In-Application Programming)
- One built-in temperature sensor with 1℃ resolution
- BOD (Brown-out Detector)
  - With 4 levels: 4.4V/3.7V/2.7V/2.2V
  - Supports Brown-out interrupt and reset option
- 96-bit unique ID
- LVR (Low Voltage Reset)
  - Threshold voltage level: 2.0V
- Operating Temperature:
  - -40℃~85℃
- Packages:
  - Green package (RoHS)
  - 64-pin LQFP, 48-pin LQFP, 33-pin QFN, 20-pin TSSOP

3 ABBREVIATIONS

3.1 List of Abbreviations

Acronym	Description
ADC	Analog-to-Digital Converter
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
BOD	Brown-out Detection
FIFO	First In, First Out
FMC	Flash Memory Controller
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
HIRC	22.1184 MHz Internal High Speed RC Oscillator
HXT	4~24 MHz External High Speed Crystal Oscillator
IAP	In Application Programming
ICP	In Circuit Programming
ISP	In System Programming
LDO	Low Dropout Regulator
LIRC	10 kHz internal low speed RC oscillator (LIRC)
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PLL	Phase-Locked Loop
PWM	Pulse Width Modulation
SPI	Serial Peripheral Interface
SPS	Samples per Second
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
UCID	Unique Customer ID
WDT	Watchdog Timer
WWDT	Window Watchdog Timer

Table 3.1-1 List of Abbreviations

4 PARTS INFORMATION AND PIN CONFIGURATION

4.1 NuMicro™ M058S Series Selection Guide

Part Number	APROM (KB)	RAM (KB)	Data Flash (KB)	ISP ROM (KB)	I/O	Timer (32-Bit)	Connectivity			PWM (16-bit)	ADC (12-bit)	WDT	WWDT	ISP/ICP/IAP	Package	Operating Temperature Range(℃)
							UART	SPI	I^2C							
M058SFAN	32	4	4	4	14	4	1	1	1	1	2	√	√	√	TSSOP20	-40 to +85
M058SZAN	32	4	4	4	26	4	1	1	1	2	5	√	√	√	QFN33	-40 to +85
M058SLAN	32	4	4	4	42	4	1	1	2	4	8	√	√	√	LQFP48	-40 to +85
M058SSAN	32	4	4	4	55	4	1	1	2	4	8	√	√	√	LQFP64	-40 to +85

Table 4.1-1 NuMicro™ M058S Series Selection Guide

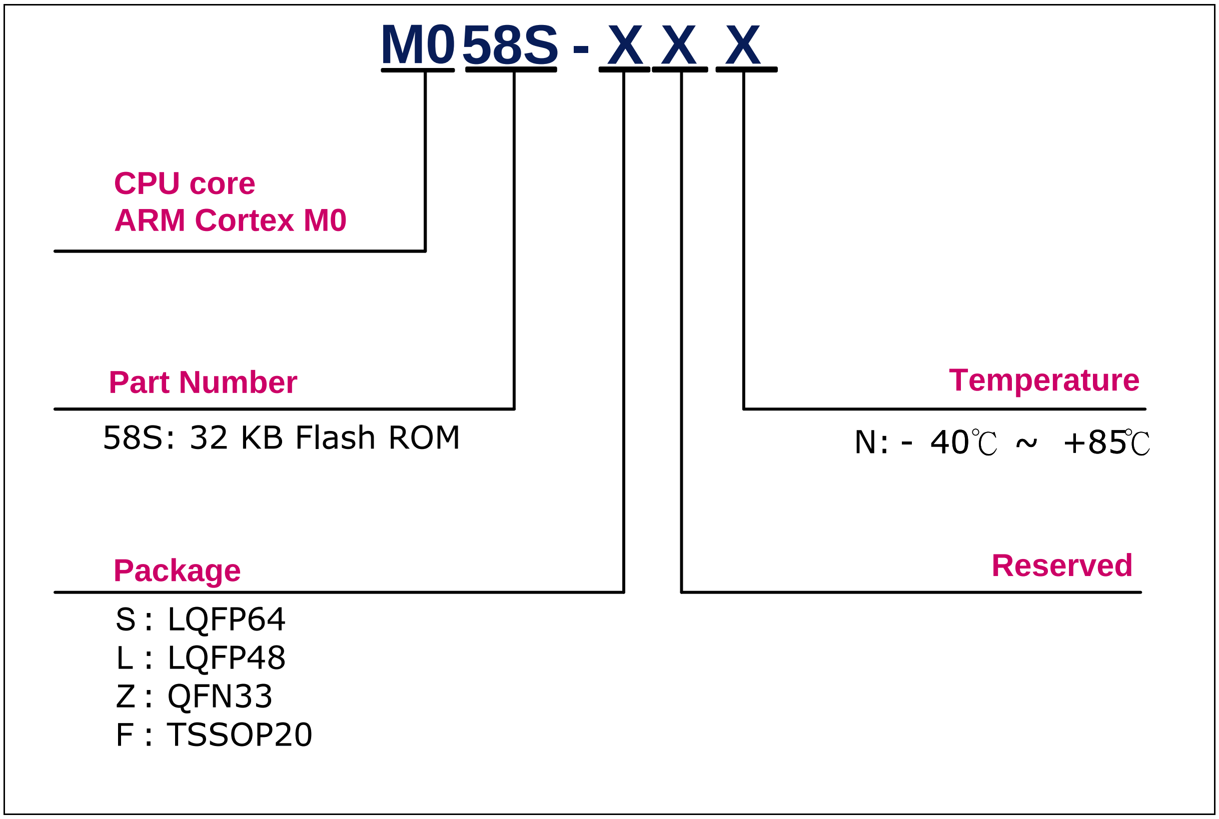


Figure 4.1-1 NuMicro™ M058S Series Selection Code

4.2 Pin Configuration

4.2.1 TSSOP20 pin

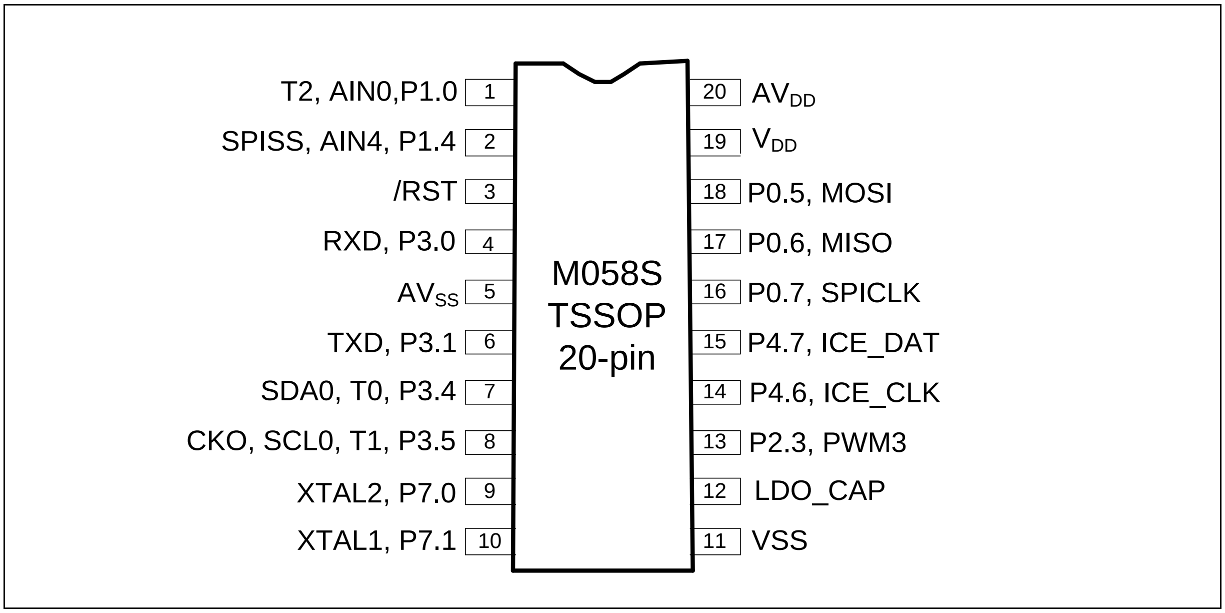


Figure 4.2-1 NuMicro™ M058S TSSOP20 Pin Diagram

4.2.2 QFN 33-pin

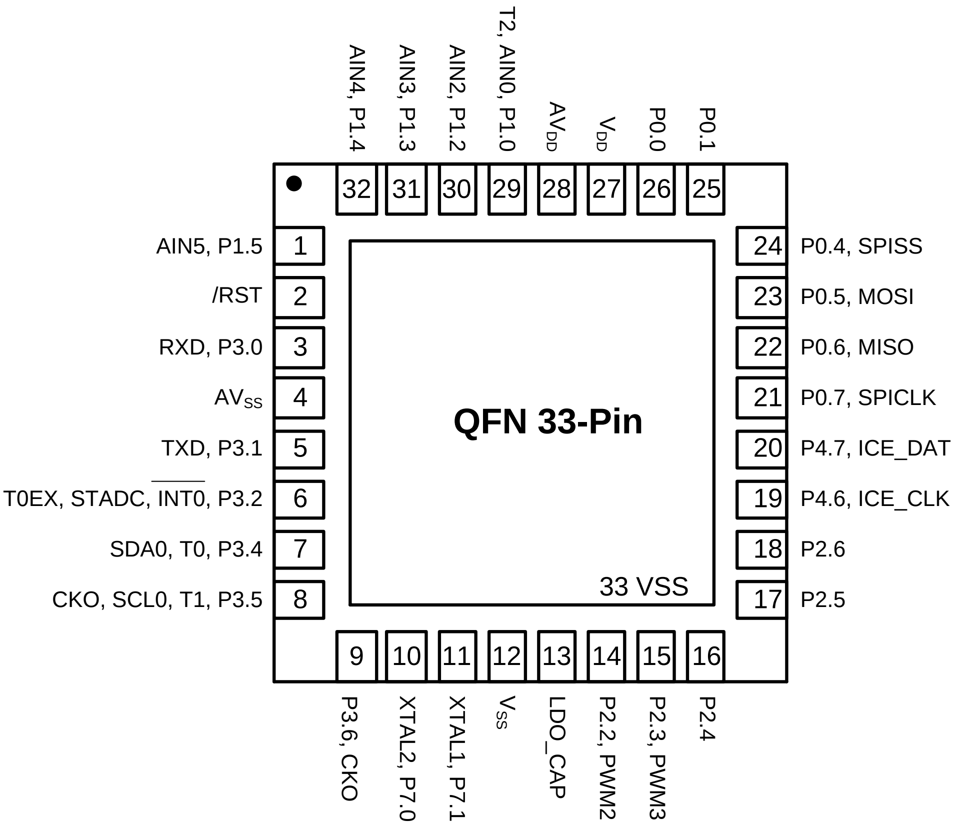


Figure 4.2-2 NuMicro™ M058S Series QFN-33 Pin Diagram

4.2.3 LQFP 48-pin

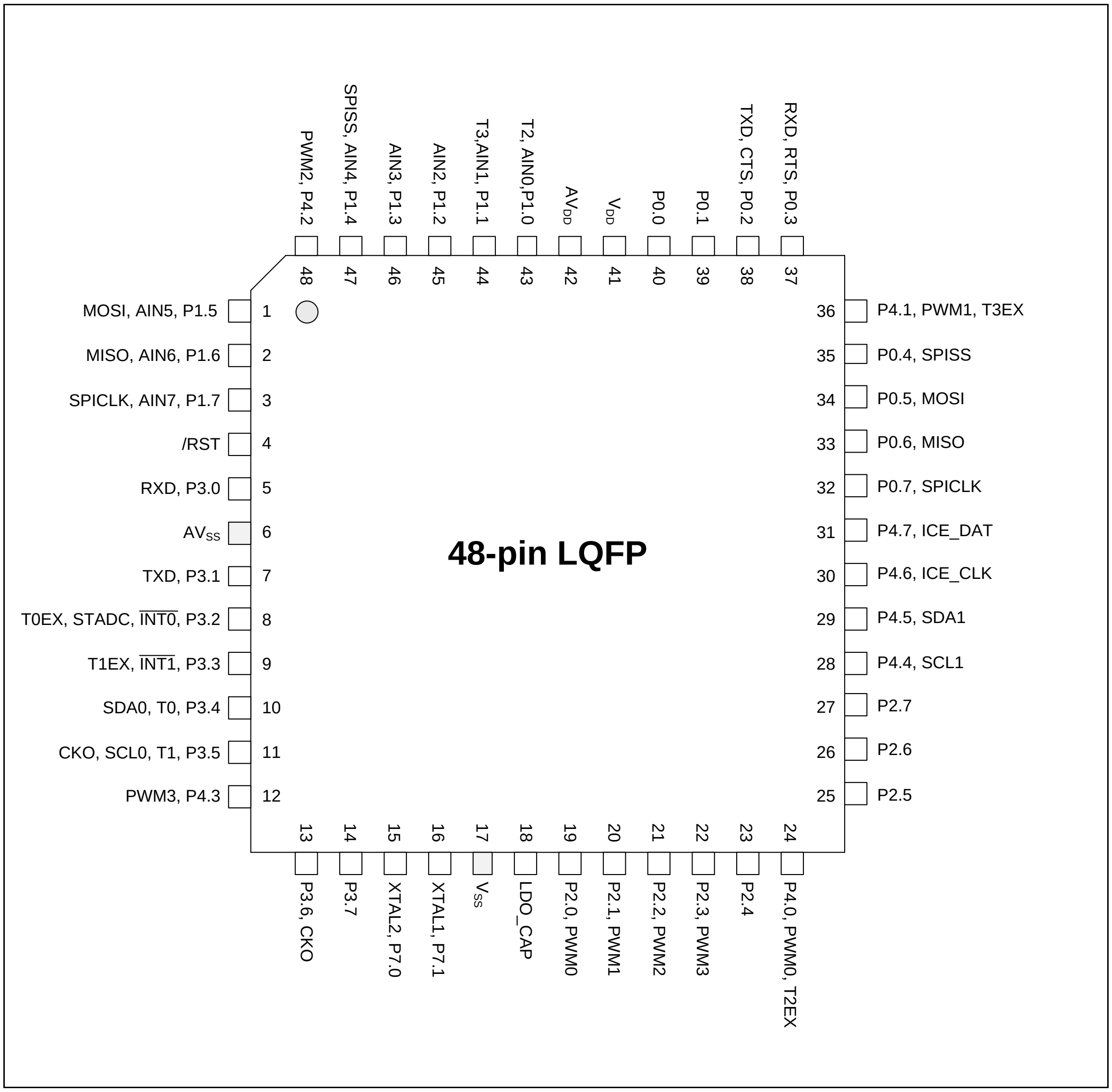


Figure 4.2-3 NuMicro™ M058S Series LQFP-48 Pin Diagram

4.2.4 LQFP 64-pin

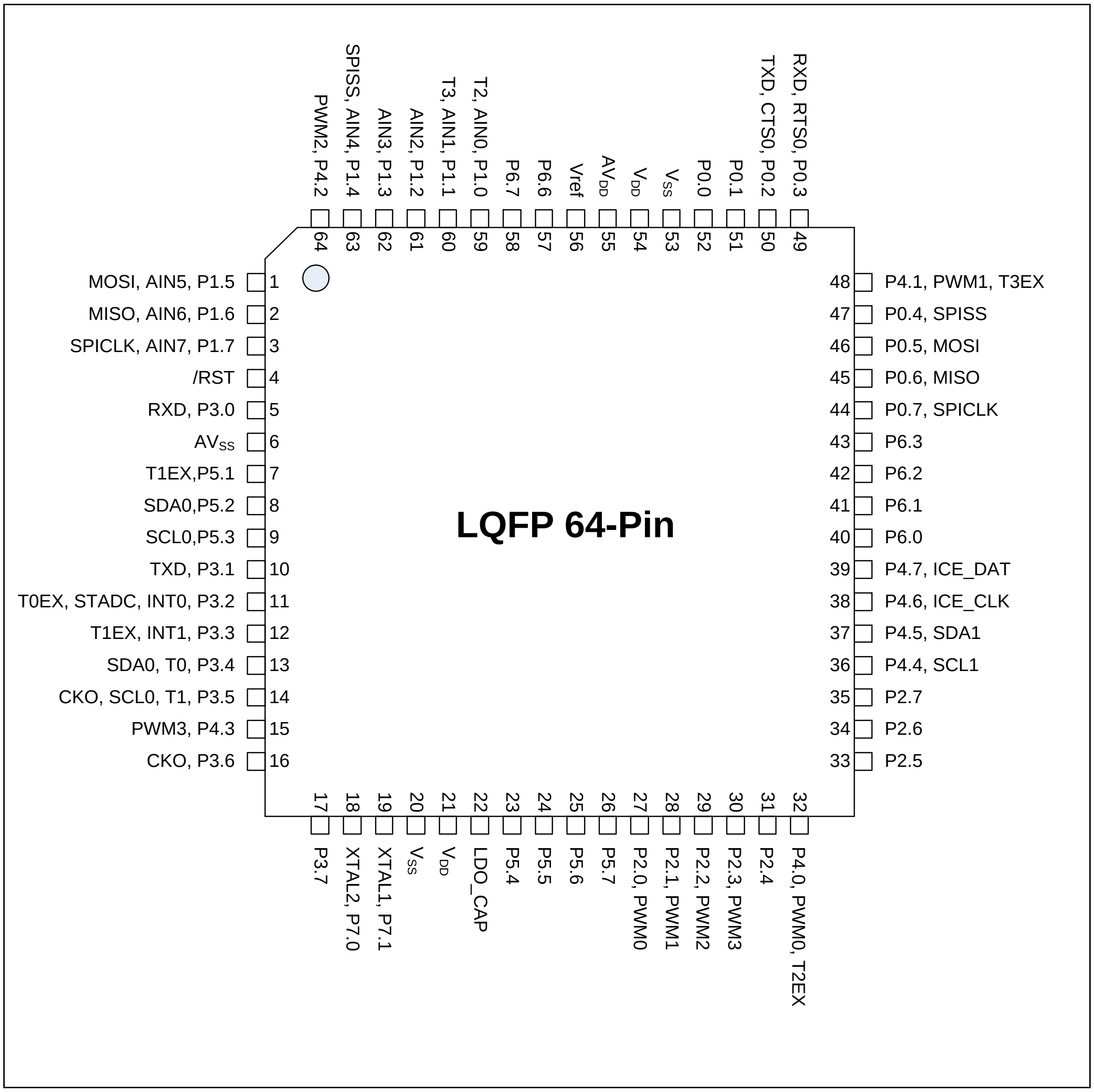


Figure 4.2-4 NuMicro™ M058S Series LQFP-64 Pin Diagram

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

4.3 Pin Description

Pin Number				Symbol	Alternate Function			Type[1]	Description
TSSOP 20	QFN 33	LQFP 48	LQFP 64		1	2	3		
19	27	41	21 54	V_{DD}				P	Power supply to I/O ports and LDO source for internal PLL and digital circuit.
11	12 33	17	20 53	V_{SS}				P	Ground pin for digital circuit.
20	28	42	55	AV_{DD}				P	Power supply to internal analog circuit.
5	4	6	6	AV_{SS}				P	Analog Ground pin for analog circuit.
	NC	NC	56	V_{ref}				P	Voltage reference input for ADC
12	13	18	22	LDO _CAP				P	LDO output pin **Note:** This pin needs to be connected with a 1uF capacitor.
3	2	4	4	/RST				I (ST)	/RST pin is a Schmitt trigger input pin for hardware device reset. A "**Low**" on this pin for 768 clock counter of Internal RC 22M 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. **Note:** It is recommended to use 10 kΩ pull-up resistor and 10 uF capacitor on /RST pin.
	26	40	52	P0.0				I/O	**PORT0:** General purpose I/O port, which can be configured by software in four modes. Its multifunction pins are for CTS1, RTS1, CTS0, RTS0, SPISS, MOSI, MISO, and SPICLK. The pins SPISS, MOSI, MISO, and SPICLK are for the SPI function use.
	25	39	51	P0.1				I/O	
	NC	38	50	P0.2	CTS		TXD[2]	I/O	
	NC	37	49	P0.3	RTS		RXD[2]	I/O	
	24	35	47	P0.4	SPISS[2]			I/O	

Pin Number				Symbol	Alternate Function			Type[1]	Description
TSSOP 20	QFN 33	LQFP 48	LQFP 64		1	2	3		
18	23	34	46	P0.5	MOSI[2]			I/O	CTS: Clear to Send input pin for UART RTS: Request to Send output pin for UART The RXD/TXD pins are for UART function use.
17	22	33	45	P0.6	MISO[2]			I/O	
16	21	32	44	P0.7	SPICLK[2]			I/O	
1	29	43	59	P1.0	T2	AIN0		I/O	**PORT1:** General purpose I/O port, which can be configured by software in four modes. Its multifunction pins are for T2, T3, SPISS0, MOSI, MISO, and SPICLK. The pins SPISS0, MOSI, MISO, and SCLK are for the SPI function use. The pins AIN0~AIN7 are for the 12 bits ADC function use. The T2/T3 pins are for Timer2/3 external event counter input.
	NC	44	60	P1.1	T3	AIN1		I/O	
	30	45	61	P1.2		AIN2		I/O	
	31	46	62	P1.3		AIN3		I/O	
2	32	47	63	P1.4	SPISS[2]	AIN4		I/O	
	1	1	1	P1.5	MOSI[2]	AIN5		I/O	
	NC	2	2	P1.6	MISO[2]	AIN6		I/O	
	NC	3	3	P1.7	SPICLK[2]	AIN7		I/O	
	NC	19	27	P2.0	PWM0[2]			I/O	**PORT2:** General purpose I/O port, which can be configured by software in four modes. It has an alternative function. The pins PWM0~PWM3 are for the PWM function use.
	NC	20	28	P2.1	PWM1[2]			I/O	
	14	21	29	P2.2	PWM2[2]			I/O	
13	15	22	30	P2.3	PWM3[2]			I/O	
	16	23	31	P2.4				I/O	
	17	25	33	P2.5				I/O	
	18	26	34	P2.6				I/O	
	NC	27	35	P2.7				I/O	
4	3	5	5	P3.0	RXD[2]			I/O	**PORT3:** General purpose I/O port, which can be configured by software in four modes. Its multifunction pins are for RXD, TXD, /INT0, /INT1, T0 and T1. The RXD/TXD pins are for UART function use.
6	5	7	10	P3.1	TXD[2]			I/O	
	6	8	11	P3.2	/INT0	STADC	T0EX	I/O	
	NC	9	12	P3.3	/INT1		T1EX	I/O	

Pin Number				Symbol	Alternate Function			Type[1]	Description
TSSOP 20	QFN 33	LQFP 48	LQFP 64		1	2	3		
7	7	10	13	P3.4	T0	SDA0		I/O	The SDA0/SCL0 pins are for I²C0 function use. CKO: HCLK clock output The STADC pin is for ADC external trigger input. The T0/T1 pins are for Timer0/1 external event counter input. The T0EX/T1EX pins are for external capture/reset trigger input of Timer0/1.
8	8	11	14	P3.5	T1	SCL0	CKO[2]	I/O	
	9	13	16	P3.6		CKO		I/O	
	NC	14	17	P3.7				I/O	
	NC	24	32	P4.0	PWM0[2]		T2EX	I/O	**PORT4:** General purpose I/O port, which can be configured by software in four modes. Its multifunction pins are for PWM0-3, SCL1, SDA1, ICE_CLK and ICE_DAT. The ICE_CLK/ICE_DAT pins are for JTAG-ICE function use. PWM0-3 can be used from P2.0-P2.3 or P4.0-P4.3. The T2EX/T3EX pins are for external capture/reset trigger input of Timer2/3. **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_CLK and ICE_DAT pins.
	NC	36	48	P4.1	PWM1[2]		T3EX	I/O	
	NC	48	64	P4.2	PWM2[2]			I/O	
	NC	12	15	P4.3	PWM3[2]			I/O	
	NC	28	36	P4.4		SCL1		I/O	
	NC	29	37	P4.5		SDA1		I/O	
14	19	30	38	P4.6	ICE_CLK			I/O	
15	20	31	39	P4.7	ICE_DAT			I/O	
	NC	NC	7	P5.1	T1EX			I/O	**PORT5:** General purpose I/O port, which can be configured by software in four modes. Its multifunction pins are for T0EX, T1EX, SDA0 and SCL0. The T0EX/T1EX pins are for external capture/reset trigger input of Timer0/1. The SDA0/SCL0 pins are for I²C0 function use.
	NC	NC	8	P5.2	SDA0			I/O	
	NC	NC	9	P5.3	SCL0			I/O	
	NC	NC	23	P5.4				I/O	
	NC	NC	24	P5.5				I/O	
	NC	NC	25	P5.6				I/O	
	NC	NC	26	P5.7				I/O	
	NC	NC	40	P6.0				I/O	**PORT6:** General purpose I/O port, which can be configured by software in four modes.
	NC	NC	41	P6.1				I/O	

Pin Number				Symbol	Alternate Function			Type[1]	Description
TSSOP 20	QFN 33	LQFP 48	LQFP 64		1	2	3		
	NC	NC	42	P6.2				I/O	
	NC	NC	43	P6.3				I/O	
	NC	NC	57	P6.6				I/O	
	NC	NC	58	P6.7				I/O	
9	10	15	18	P7.0	XTAL2			I/O, O	PORT7: General purpose I/O port, which can be configured by software in four modes. Its multifunction pins are for XTAL XTAL: External 4~24 MHz (high speed) crystal pin.
10	11	16	19	P7.1	XTAL1			I/O, I(ST)	

Note 1: I/O type description. I: Input, O: Output, I/O: Quasi-bidirectional, D: Open-drain, P: Power pins, ST: Schmitt trigger.

Note 2: PWM0 ~ PWM3, RXD, TXD, RXD1, TXD1, SCL1, SDA1 and CKO can be assigned to different pins. However, a pin function can only be assigned to a pin at the same time, i.e. software cannot assign RXD to P0.3 and P3.0 at the same time.

5 BLOCK DIAGRAM

5.1 NuMicro™ M058S Series Block Diagram

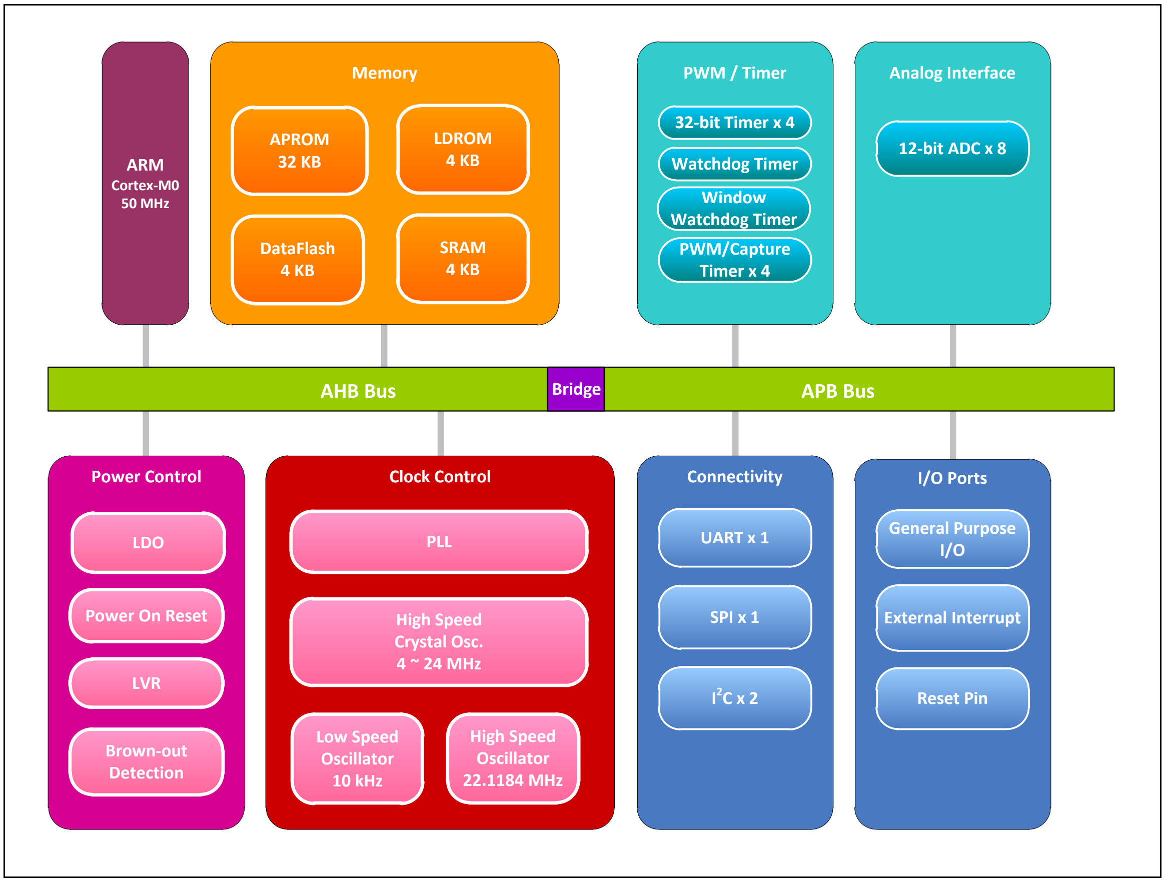


Figure 5.1-1 NuMicro™ M058S Series Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 ARM® Cortex®-M0 Core

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

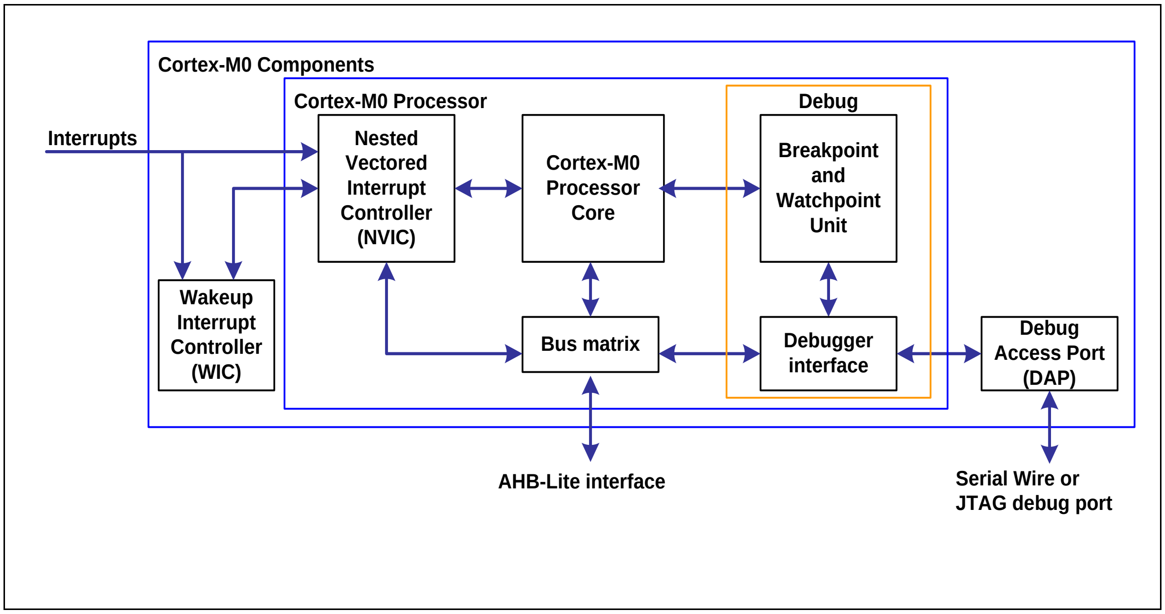


Figure 6.1-1 Functional Block Diagram

The implemented device provides:

- A low gate count processor:
  - ARMv6-M Thumb® instruction set
  - Thumb-2 technology
  - ARMv6-M compliant 24-bit SysTick timer
  - A 32-bit hardware multiplier
  - System interface supported with little-endian data accesses
  - Ability to have deterministic, fixed-latency, interrupt handling
  - Load/store-multiples and multicycle-multiplies that can be abandoned and restarted to facilitate rapid interrupt handling
  - C Application Binary Interface compliant exception model. This is the ARMv6-M, C Application Binary Interface (C-ABI) compliant exception model that enables the use of pure C functions as interrupt handlers
  - Low Power Sleep mode entry using the Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or return from interrupt sleep-on-exit feature

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

6.2 System Manager

6.2.1 Overview

System management includes the following sections:

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

6.2.2 System Reset

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

- Hardware Reset
  - Power-on Reset (POR)
  - Low level on the Reset Pin (nRST)
  - Watchdog Timer Time-out Reset (WDT)
  - Low Voltage Reset (LVR)
  - Brown-out Detector Reset (BOD)
- Software Reset
  - MCU Reset - SYSRESETREQ(AIRCR[2])
  - Cortex-M0 Core One-shot Reset - CPU_RST(IPRSTC1[1])
  - Chip One-shot Reset - CHIP_RST(IPRSTC1[0])

Note: ISPCON.BS keeps the original value after MCU Reset and CPU Reset.

6.2.3 System Power Architecture

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

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

The output of internal voltage regulator, LDO_CAP, requires an external capacitor which should be located close to the corresponding pin. Analog power (AV_{DD}) should be the same voltage level as the digital power (V_{DD}). The following figure shows the power distribution of the M058S series.

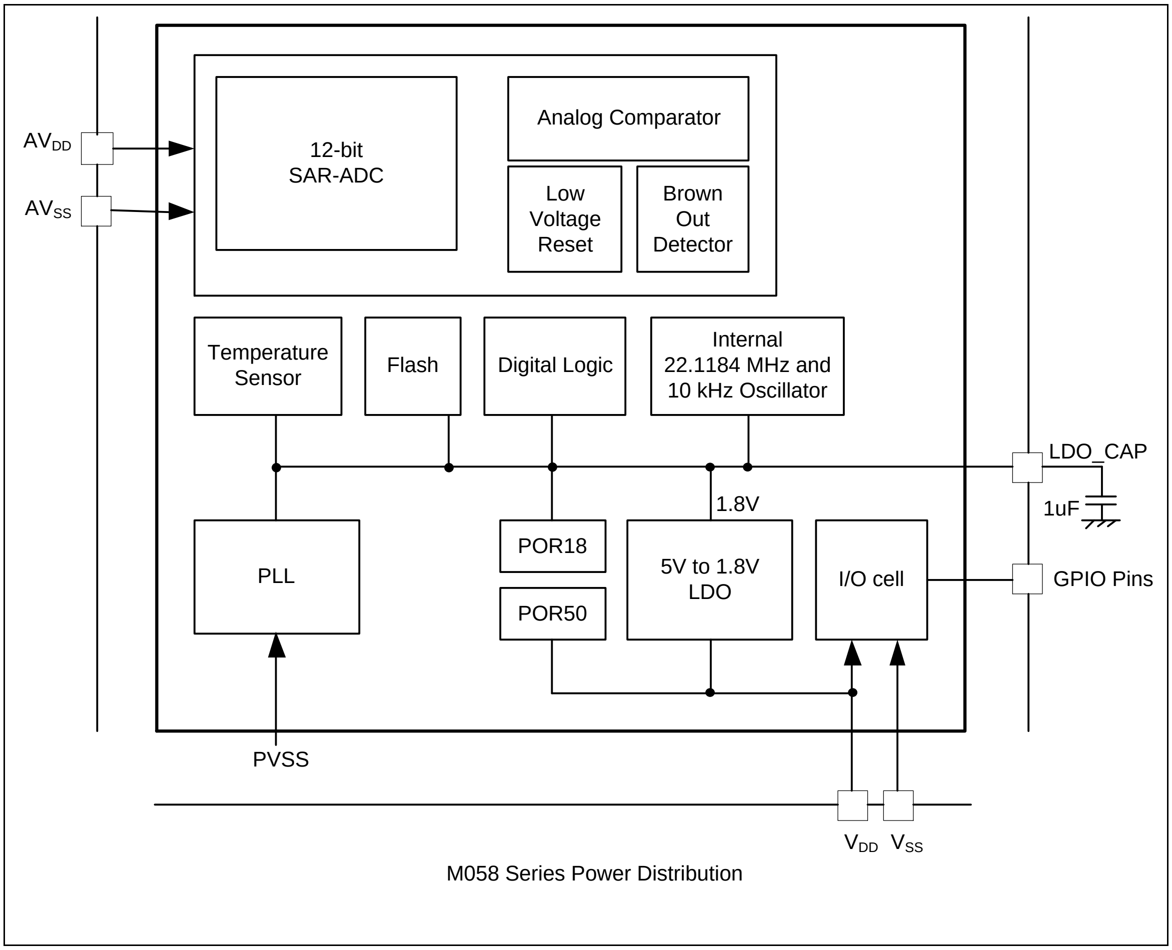


Figure 6.2-1 NuMicro M058S Series Power Architecture Diagram

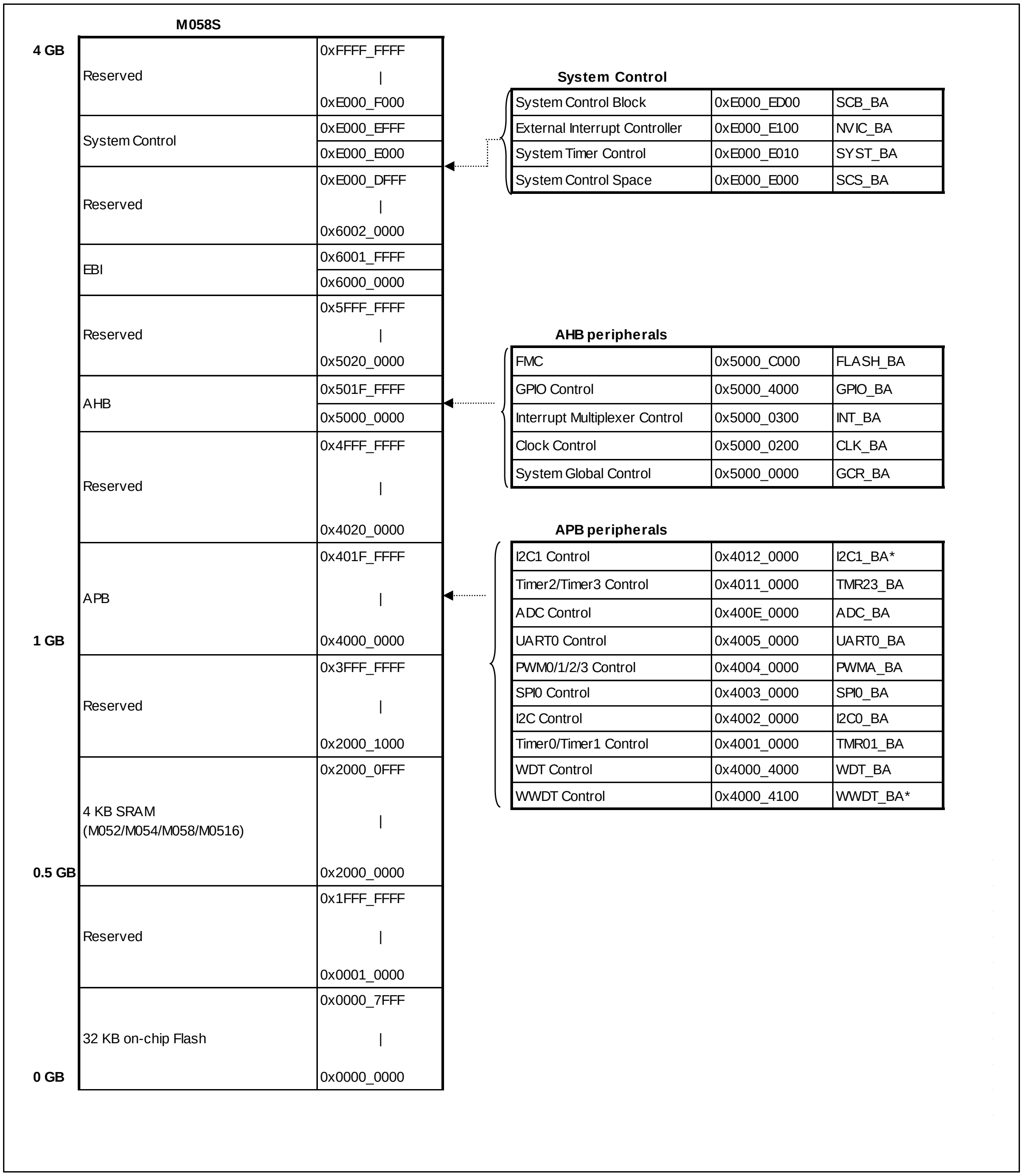
6.2.4 System Memory Map

The NuMicro™ M058S series provides 4 GB addressing space. The addressing space assigned to each on-chip controllers are shown in the following table. The detailed register definition, addressing space, and programming details will be described in the following sections for each on-chip peripheral. The NuMicro™ M058S series only supports little-endian data format.

Addressing Space	Token	Modules
Flash & SRAM Memory Space		
0x0000_0000 – 0x0000_7FFF	FLASH_BA	FLASH Memory Space (32 KB)
0x2000_0000 – 0x2000_0FFF	SRAM_BA	SRAM Memory Space (4 KB)
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~P7) Control Registers
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash Memory Control Registers
APB Modules Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4000_4000 – 0x4000_00FF	WDT_BA	Watchdog Timer Control Registers
0x4000_4100 – 0x4000_7FFF	WWDT_BA	Window Watchdog Timer Control Registers
0x4001_0000 – 0x4001_3FFF	TMR01_BA	Timer0/Timer1 Control Registers
0x4002_0000 – 0x4002_3FFF	I2C0_BA	I^{2}C0 Interface Control Registers
0x4003_0000 – 0x4003_3FFF	SPI0_BA	SPI0 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
0x4012_0000 – 0x4012_3FFF	I2C1_BA	I^{2}C1 Interface Control Registers
System Control Space (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SYST_BA	System Timer Control Registers
0xE000_E100 – 0xE000_ECFF	NVIC_BA	External Interrupt Controller Control Registers
0xE000_ED00 – 0xE000_ED8F	SCB_BA	System Control Block Registers

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

6.2.5 Whole System Memory Mapping



	M058S	
4 GB	Reserved	0xFFFF_FFFF | 0xE000_F000
	System Control	0xE000_EFFF 0xE000_E000
	Reserved	0xE000_DFFF | 0x6002_0000
	EBI	0x6001_FFFF 0x6000_0000
	Reserved	0x5FFF_FFFF | 0x5020_0000
	AHB	0x501F_FFFF 0x5000_0000
	Reserved	0x4FFF_FFFF | 0x4020_0000
1 GB	APB	0x401F_FFFF | 0x4000_0000
	Reserved	0x3FFF_FFFF | 0x2000_1000
0.5 GB	4 KB SRAM (M052/M054/M058/M0516)	0x2000_0FFF | 0x2000_0000
	Reserved	0x1FFF_FFFF | 0x0001_0000
0 GB	32 KB on-chip Flash	0x0000_7FFF | 0x0000_0000

System Control

System Control Block	0xE000_ED00	SCB_BA
External Interrupt Controller	0xE000_E100	NVIC_BA
System Timer Control	0xE000_E010	SYST_BA
System Control Space	0xE000_E000	SCS_BA

AHB peripherals

FMC	0x5000_C000	FLASH_BA
GPIO Control	0x5000_4000	GPIO_BA
Interrupt Multiplexer Control	0x5000_0300	INT_BA
Clock Control	0x5000_0200	CLK_BA
System Global Control	0x5000_0000	GCR_BA

APB peripherals

I2C1 Control	0x4012_0000	I2C1_BA*
Timer2/Timer3 Control	0x4011_0000	TMR23_BA
ADC Control	0x400E_0000	ADC_BA
UART0 Control	0x4005_0000	UART0_BA
PWM0/1/2/3 Control	0x4004_0000	PWMA_BA
SPI0 Control	0x4003_0000	SPI0_BA
I2C Control	0x4002_0000	I2C0_BA
Timer0/Timer1 Control	0x4001_0000	TMR01_BA
WDT Control	0x4000_4000	WDT_BA
WWDT Control	0x4000_4100	WWDT_BA*

6.2.6 System Manager Controller Register Map

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

Register	Offset	R/W	Description	Reset Value
GCR Base Address: **GCR_BA = 0x5000_0000**				
PDID	GCR_BA+0x00	R	Part Device Identification number Register	0x0000_5810
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
TEMPCR	GCR_BA+0x1C	R/W	Temperature Sensor Control Register	0x0000_0000
PORCR	GCR_BA+0x24	R/W	Power-On-Reset Controller Register	0x0000_00XX
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
P5_MFP	GCR_BA+0x44	R/W	P5 input type control register	0x0000_0000
P6_MFP	GCR_BA+0x48	R/W	P6 input type control register	0x0000_0000
P7_MFP	GCR_BA+0x4C	R/W	P7 input type control register	0x0000_0003
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_5810[1]

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

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

Bits	Description	
[31:0]	**PDID**	**Part Device Identification Number** This register reflects the device part number code. Software can read this register to identify which device is used. For example, M058SLAN PDID code is 0x0000_5810.

NuMicro™ M058S Series	Part Device Identification Number
M058SSAN	0x0000_5816
M058SFAN	0x0000_5818
M058SLAN	0x0000_5810
M058SZAN	0x0000_5813

System Reset Source Register (RSTSRC)

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

Register	Offset	R/W	Description	Reset Value
RSTSRC	GCR_BA+0x04	R/W	System Reset Source Register	0x0000_00XX

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**RSTS_CPU**	**CPU Reset Flag** The RSTS_CPU flag is set by hardware if software writes CPU_RST (IPRSTC1[1]) 1 to reset Cortex®-M0 core and Flash memory controller (FMC). 0 = No reset from CPU. 1 = Cortex®-M0 core and FMC are reset by software setting CPU_RST to 1. **Note:** Write 1 to clear this bit to 0.
[6]	**Reserved**	Reserved.
[5]	**RSTS_MCU**	**MCU Reset Flag** The RSTS_MCU flag is set by the “reset signal” from the Cortex®-M0 core to indicate the previous reset source. 0 = No reset from Cortex®-M0. 1 = The Cortex®-M0 had issued the reset signal to reset the system by writing 1 to bit SYSRESETREQ (AIRCR[2], Application Interrupt and Reset Control Register, address = 0xE000ED0C) in system control registers of Cortex®-M0 core. **Note:** Write 1 to clear this bit to 0.
[4]	**RSTS_BOD**	**Brown-out Detector Reset Flag** The RSTS_BOD flag is set by the “reset signal” from the Brown-out Detector to indicate the previous reset source. 0 = No reset from BOD. 1 = The BOD had issued the reset signal to reset the system

		Note: Write 1 to clear this bit to 0.
[3]	**RSTS_LVR**	**Low Voltage Reset Flag** The RSTS_LVR flag is set by the “reset signal” from the Low-Voltage-Reset controller to indicate the previous reset source. 0 = No reset from LVR. 1 = The LVR controller had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[2]	**RSTS_WDT**	**Watchdog Reset Flag** The RSTS_WDT flag is set by the “reset signal” from the watchdog timer to indicate the previous reset source. 0 = No reset from watchdog timer. 1 = The watchdog timer had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[1]	**RSTS_RESET**	**Reset Pin Reset Flag** The RSTS_RESET flag is set by the “reset signal” from the nRST pin to indicate the previous reset source. 0 = No reset from the nRST pin. 1 = The nRST pin had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[0]	**RSTS_POR**	**Power-on Reset Flag** The RSTS_POR flag is set by the “reset signal” from the Power-on Reset (POR) controller or bit CHIP_RST (IPRSTC1[0]) to indicate the previous reset source. 0 = No reset from POR or CHIP_RST. 1 = The Power-on Reset (POR) or CHIP_RST had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.

Peripheral Reset Control Register1 (IPRSTC1)

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

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CPU_RST**	**Cortex-M0 Core One-shot Reset (Write Protect)** Setting this bit will only reset the Cortex-M0 core and Flash Memory Controller (FMC), and this bit will automatically return 0 after the two clock cycles. 0 = Cortex-M0 core normal operation. 1 = Cortex-M0 core one-shot reset. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[0]	**CHIP_RST**	**Chip One-shot Reset (Write Protect)** Setting this bit will reset the whole chip, including Cortex-M0 core and all peripherals, and this bit will automatically return to 0 after the 2 clock cycles. The CHIP_RST is the same as the POR reset. All the chip controllers are reset and the chip setting from CONFIG0 are also reload. 0 = Chip normal operation. 1 = Chip one-shot reset. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Peripheral Reset Control Register2 (IPRSTC2)

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

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

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

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**ADC_RST**	**ADC Controller Reset** 0 = ADC controller normal operation. 1 = ADC controller reset.
[27:21]	**Reserved**	Reserved.
[20]	**PWM03_RST**	**PWM03 Controller Reset** 0 = PWM03 controller normal operation. 1 = PWM03 controller reset.
[19:17]	**Reserved**	Reserved.
[16]	**UART0_RST**	**UART0 Controller Reset** 0 = UART0 controller normal operation. 1 = UART0 controller reset.
[15:13]	**Reserved**	Reserved.
[12]	**SPI0_ RST**	**SPI0 Controller Reset** 0 = SPI0 controller normal operation. 1 = SPI0 controller reset.
[11:10]	**Reserved**	Reserved.
[9]	**I2C1_RST**	**I^{2}C1 Controller Reset**

		0= I^2C1 controller normal operation. 1= I^2C1 controller reset.
[8]	**I2C0_RST**	**I^2C Controller Reset** 0= I^2C0 controller normal operation. 1= I^2C0 controller reset.
[7:6]	**Reserved**	Reserved.
[5]	**TMR3_RST**	**Timer3 Controller Reset** 0 = Timer3 controller normal operation. 1 = Timer3 controller reset.
[4]	**TMR2_RST**	**Timer2 Controller Reset** 0 = Timer2 controller normal operation. 1 = Timer2 controller reset.
[3]	**TMR1_RST**	**Timer1 Controller Reset** 0 = Timer1 controller normal operation. 1 = Timer1 controller reset.
[2]	**TMR0_RST**	**Timer0 Controller Reset** 0 = Timer0 controller normal operation. 1 = Timer0 controller reset.
[1]	**GPIO_RST**	**GPIO (P0~P4) Controller Reset** 0 = GPIO controller normal operation. 1 = GPIO controller reset.
[0]	**Reserved**	Reserved.

Brown-out Detector Control Register (BODCR)

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

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**LVR_EN**	**Low Voltage Reset Enable Control (Write Protect)** The LVR function reset the chip when the input power voltage is lower than LVR circuit setting. LVR function is enabled by default. 0 = Low Voltage Reset function Disabled. 1 = Low Voltage Reset function Enabled – After enabling the bit, the LVR function will be active with 100us delay for LVR output stable (default). **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.
[6]	**BOD_OUT**	**Brown-out Detector Output Status** 0 = Brown-out Detector output status is 0, which means the detected voltage is higher than BOD_VL setting or BOD_EN is 0. 1 = Brown-out Detector output status is 1, which means the detected voltage is lower than BOD_VL setting. If the BOD_EN is 0, BOD function disabled, this bit always responds to 0.
[5]	**BOD_LPM**	**Brown-out Detector Low power Mode (Write Protect)** 0 = BOD operated in Normal mode (default). 1 = BOD Low Power mode Enabled. **Note1:** The BOD consumes about 100 uA in Normal mode, and the low power mode can

<table>
<tr><td></td><td></td><td>reduce the current to about 1/10 but slow the BOD response.
<b>Note2:</b> This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</td></tr>
<tr><td>[4]</td><td><b>BOD_INTF</b></td><td><b>Brown-out Detector Interrupt Flag</b>
0 = Brown-out Detector does not detect any voltage draft at V_{DD} down through or up through the voltage of BOD_VL setting.
1 = When Brown-out Detector detects the V_{DD} is dropped down through the voltage of BOD_VL setting or the V_{DD} is raised up through the voltage of BOD_VL setting, this bit is set to 1 and the Brown-out interrupt is requested if Brown-out interrupt is enabled.
<b>Note:</b> Write 1 to clear this bit to 0.</td></tr>
<tr><td>[3]</td><td><b>BOD_RSTEN</b></td><td><b>Brown-out Reset Enable Control (Write Protect)</b>
0 = Brown-out “INTERRUPT” function Enabled.
While the BOD function is enabled (BOD_EN high) and BOD interrupt function is enabled (BOD_RSTEN low), BOD will assert an interrupt if BOD_OUT is high. BOD interrupt will keep till to the BOD_EN set to 0. BOD interrupt can be blocked by disabling the NVIC BOD interrupt or disabling BOD function (set BOD_EN low).
1 = Brown-out “RESET” function Enabled.
<b>Note1:</b> While the Brown-out Detector function is enabled (BOD_EN high) and BOD reset function is enabled (BOD_RSTEN high), BOD will assert a signal to reset chip when the detected voltage is lower than the threshold (BOD_OUT high).
<b>Note2:</b> The default value is set by flash controller user configuration register config0 bit[20].
<b>Note3:</b> This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</td></tr>
<tr><td>[2:1]</td><td><b>BOD_VL</b></td><td><b>Brown-out Detector Threshold Voltage Select (Write Protect)</b>
The default value is set by flash controller user configuration register config0 bit[22:21]
<table>
<tr><th>BOV_VL[1]</th><th>BOV_VL[0]</th><th>Brown-out voltage</th></tr>
<tr><td>1</td><td>1</td><td>4.4V</td></tr>
<tr><td>1</td><td>0</td><td>3.7V</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>
<b>Note:</b> This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</td></tr>
<tr><td>[0]</td><td><b>BOD_EN</b></td><td><b>Brown-out Detector Enable Control (Write Protect)</b>
The default value is set by flash controller user configuration register config0 bit[23]
0 = Brown-out Detector function Disabled.
1 = Brown-out Detector function Enabled.
<b>Note:</b> This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</td></tr>
</table>

Temperature Sensor Control Register (TEMPCR)

Register	Offset	R/W	Description	Reset Value
TEMPCR	GCR_BA+0x1C	R/W	Temperature Sensor Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**VTEMP_EN**	**Temperature Sensor Enable Control** This bit is used to enable/disable temperature sensor function. 0 = Temperature sensor function Disabled (default). 1 = Temperature sensor function Enabled. **Note:** After this bit is set to 1, the value of temperature sensor output can be obtained from the ADC conversion result. Please refer to the ADC chapter for detailed ADC conversion functional description.

Power-on Reset Control Register (PORCR)

Register	Offset	R/W	Description	Reset Value
PORCR	GCR_BA+0x24	R/W	Power-on Reset Controller Register	0x0000_00XX

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**POR_DIS_CODE**	**Power-on Reset Enable Control (Write Protect)** When powered on, the POR circuit generates a reset signal to reset the whole chip function, but noise on the power may cause the POR active again. User can disable internal POR circuit to avoid unpredictable noise to cause chip reset by writing 0x5AA5 to this field. The POR function will be active again when this field is set to another value or chip is reset by other reset source, including: nRST, Watchdog reset, LVR reset, BOD reset, ICE reset command and the software-chip reset function. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Multiple Function Port0 Control Register (P0_MFP)

Register	Offset	R/W	Description	Reset Value
P0_MFP	GCR_BA+0x30	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]							

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:24]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[23:16]</td><td><b>P0_TYPEn</b></td><td><b>P0[7:0] Input Schmitt Trigger Function Enable Control</b>
0 = P0[7:0] I/O input Schmitt Trigger function Disabled.
1 = P0[7:0] I/O input Schmitt Trigger function Enabled.</td></tr>
<tr><td>[15]</td><td><b>P0_ ALT[7]</b></td><td><b>P0.7 Alternate Function Selection</b>
The pin function of P0.7 depends on P0_MFP[7] and P0_ALT[7].
<table>
<tr><th>P0_ALT[7]</th><th>P0_MFP[7]</th><th>P0.7 function</th></tr>
<tr><td>0</td><td>0</td><td>P0.7</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>SPICLK0(SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[14]</td><td><b>P0_ ALT[6]</b></td><td><b>P0.6 Alternate Function Selection</b>
The pin function of P0.6 depends on P0_MFP[6] and P0_ALT[6].
<table>
<tr><th>P0_ALT[6]</th><th>P0_MFP[6]</th><th>P0.6 function</th></tr>
<tr><td>0</td><td>0</td><td>P0.6</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td>
<table>
<tr><td>1</td><td>0</td><td>MISO0(SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[13]</td><td><b>P0_ ALT[5]</b></td><td>
<b>P0.5 Alternate Function Selection</b>
The pin function of P0.5 depends on P0_MFP[5] and P0_ALT[5].
<table>
<tr><th>P0_ALT[5]</th><th>P0_MFP[5]</th><th>P0.5 function</th></tr>
<tr><td>0</td><td>0</td><td>P0.5</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>MOSI0(SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[12]</td><td><b>P0_ ALT[4]</b></td><td>
<b>P0.4 Alternate Function Selection</b>
The pin function of P0.4 depends on P0_MFP[4] and P0_ALT[4].
<table>
<tr><th>P0_ALT[4]</th><th>P0_MFP[4]</th><th>P0.4 function</th></tr>
<tr><td>0</td><td>0</td><td>P0.4</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</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><b>P0_ ALT[3]</b></td><td>
<b>P0.3 Alternate Function Selection</b>
The pin function of P0.3 depends on P0_MFP[3] and P0_ALT[3].
<table>
<tr><th>P0_ALT[3]</th><th>P0_MFP[3]</th><th>P0.3 function</th></tr>
<tr><td>0</td><td>0</td><td>P0.3</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>RTS0(UART0)</td></tr>
<tr><td>1</td><td>1</td><td>RXD0(UART0)</td></tr>
</table>
</td></tr>
<tr><td>[10]</td><td><b>P0_ ALT[2]</b></td><td>
<b>P0.2 Alternate Function Selection</b>
The pin function of P0.2 depends on P0_MFP[2] and P0_ALT[2].
<table>
<tr><th>P0_ALT[2]</th><th>P0_MFP[2]</th><th>P0.2 function</th></tr>
<tr><td>0</td><td>0</td><td>P0.2</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>CTS0(UART0)</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td></td><td><table><tr><td>1</td><td>1</td><td>TXD0(UART0)</td></tr></table></td></tr>
<tr><td>[9]</td><td><b>P0_ ALT[1]</b></td><td><b>P0.1 Alternate Function Selection</b>
The pin function of P0.1 depends on P0_MFP[1], P0_ALT[1] and P0_ALT1[1].<table><tr><th>P0_ALT[1]</th><th>P0_MFP[1]</th><th>P0.1 function</th></tr><tr><td>0</td><td>0</td><td>P0.1</td></tr><tr><td>0</td><td>1</td><td>Reserved</td></tr><tr><td>1</td><td>0</td><td>Reserved</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[8]</td><td><b>P0_ ALT[0]</b></td><td><b>P0.0 Alternate Function Selection</b>
The pin function of P0.0 depends on P0_MFP[0], P0_ALT[0] and P0_ALT1[0].<table><tr><th>P0_ALT[0]</th><th>P0_MFP[0]</th><th>P0.0 function</th></tr><tr><td>0</td><td>0</td><td>P0.0</td></tr><tr><td>0</td><td>1</td><td>Reserved</td></tr><tr><td>1</td><td>0</td><td>Reserved</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[7:0]</td><td><b>P0_MFP[7:0]</b></td><td><b>P0 Multiple Function Selection</b>
The pin function of P0 depends on P0_MFP and P0_ALT.
Refer to P0_ALT for detailed description.</td></tr>
</table>

Multiple Function Port1 Control Register (P1_MFP)

Register	Offset	R/W	Description	Reset Value
P1_MFP	GCR_BA+0x34	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]							

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:24]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[23:16]</td><td><b>P1_TYPEn</b></td><td><b>P1[7:0] Input Schmitt Trigger function Enable Control</b>
0 = P1[7:0] I/O input Schmitt Trigger function Disabled.
1 = P1[7:0] I/O input Schmitt Trigger function Enabled.</td></tr>
<tr><td>[15]</td><td><b>P1_ ALT[7]</b></td><td><b>P1.7 Alternate Function Selection</b>
The pin function of P1.7 depends on P1_MFP[7] and P1_ALT[7].
<table>
<tr><th>P1_ALT[7]</th><th>P1_MFP[7]</th><th>P1.7 function</th></tr>
<tr><td>0</td><td>0</td><td>P1.7</td></tr>
<tr><td>0</td><td>1</td><td>AIN7(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>SPICLK0(SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[14]</td><td><b>P1_ ALT[6]</b></td><td><b>P1.6 Alternate Function Selection</b>
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>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td><table><tr><td>1</td><td>0</td><td>MISO0(SPI0)</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[13]</td><td><b>P1_ ALT[5]</b></td><td><b>P1.5 Alternate Function Selection</b>
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>MOSI0(SPI0)</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[12]</td><td><b>P1_ ALT[4]</b></td><td><b>P1.4 Alternate Function Selection</b>
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>P1.4 function</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><b>P1_ ALT[3]</b></td><td><b>P1.3 Alternate Function Selection</b>
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>P1.3 function</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><tr><td>1</td><td>0</td><td>Reserved</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[10]</td><td><b>P1_ ALT[2]</b></td><td><b>P1.2 Alternate Function Selection</b>
The pin function of P1.2 depends on P1_MFP[2] and P1_ALT[2].<table><tr><th>P1_ALT[2]</th><th>P1_MFP[2]</th><th>P1.2 function</th></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>Reserved</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
</table>

<table>
<tr>
<td>[9]</td>
<td><b>P1_ ALT[1]</b></td>
<td><b>P1.1 Alternate Function Selection</b>
The pin function of P1.1 depends on P1_MFP[1] and P1_ALT[1].
<table>
<tr><th>P1_ALT[1]</th><th>P1_MFP[1]</th><th>P1.1 function</th></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><b>P1_ ALT[0]</b></td>
<td><b>P1.0 Alternate Function Selection</b>
The pin function of P1.0 depends on P1_MFP[0] and P1_ALT[0].
<table>
<tr><th>P1_ALT[0]</th><th>P1_MFP[0]</th><th>P1.0 function</th></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><b>P1_MFP[7:0]</b></td>
<td><b>P1 Multiple Function Selection</b>
The pin function of P1 depends on P1_MFP and P1_ALT.
Refer to P1_ALT for detailed description.</td>
</tr>
</table>

Multiple Function Port2 Control Register (P2_MFP)

Register	Offset	R/W	Description	Reset Value
P2_MFP	GCR_BA+0x38	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]							

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:24]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[23:16]</td><td><b>P2_TYPEn</b></td><td><b>P2[7:0] Input Schmitt Trigger Function Enable Control</b>
0 = P2[7:0] I/O input Schmitt Trigger function Disabled.
1 = P2[7:0] I/O input Schmitt Trigger function Enabled.</td></tr>
<tr><td>[15]</td><td><b>P2_ ALT[7]</b></td><td><b>P2.7 Alternate Function Selection</b>
The pin function of P2.7 depends on P2_MFP[7] and P2_ALT[7].
<table>
<tr><th>P2_ALT[7]</th><th>P2_MFP[7]</th><th>P2.7 function</th></tr>
<tr><td>0</td><td>0</td><td>P2.7</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[14]</td><td><b>P2_ ALT[6]</b></td><td><b>P2.6 Alternate Function Selection</b>
The pin function of P2.6 depends on P2_MFP[6] and P2_ALT[6].
<table>
<tr><th>P2_ALT[6]</th><th>P2_MFP[6]</th><th>P2.6 function</th></tr>
<tr><td>0</td><td>0</td><td>P2.6</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td><table>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[13]</td><td><b>P2_ ALT[5]</b></td><td><b>P2.5 Alternate Function Selection</b>
The pin function of P2.5 depends on P2_MFP[5] and P2_ALT[5].<table>
<tr><th>P2_ALT[5]</th><th>P2_MFP[5]</th><th>P2.5 function</th></tr>
<tr><td>0</td><td>0</td><td>P2.5</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[12]</td><td><b>P2_ ALT[4]</b></td><td><b>P2.4 Alternate Function Selection</b>
The pin function of P2.4 depends on P2_MFP[4] and P2_ALT[4].<table>
<tr><th>P2_ALT[4]</th><th>P2_MFP[4]</th><th>P2.4 function</th></tr>
<tr><td>0</td><td>0</td><td>P2.4</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[11]</td><td><b>P2_ ALT[3]</b></td><td><b>P2.3 Alternate Function Selection</b>
The pin function of P2.3 depends on P2_MFP[3] and P2_ALT[3].<table>
<tr><th>P2_ALT[3]</th><th>P2_MFP[3]</th><th>P2.3 function</th></tr>
<tr><td>0</td><td>0</td><td>P2.3</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>PWM3(PWMA channel 3)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[10]</td><td><b>P2_ ALT[2]</b></td><td><b>P2.2 Alternate Function Selection</b>
The pin function of P2.2 depends on P2_MFP[2] and P2_ALT[2].<table>
<tr><th>P2_ALT[2]</th><th>P2_MFP[2]</th><th>P2.2 function</th></tr>
<tr><td>0</td><td>0</td><td>P2.2</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td>
<table>
<tr><td>1</td><td>0</td><td>PWM2(PWMA channel 2)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[9]</td><td><b>P2_ ALT[1]</b></td><td>
<b>P2.1 Alternate Function Selection</b>
The pin function of P2.1 depends on P2_MFP[1] and P2_ALT[1].
<table>
<tr><th>P2_ALT[1]</th><th>P2_MFP[1]</th><th>P2.1 function</th></tr>
<tr><td>0</td><td>0</td><td>P2.1</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>PWM1(PWMA channel 1)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[8]</td><td><b>P2_ ALT[0]</b></td><td>
<b>P2.0 Alternate Function Selection</b>
The pin function of P2.0 depends on P2_MFP[0] and P2_ALT[0].
<table>
<tr><th>P2_ALT[0]</th><th>P2_MFP[0]</th><th>P2.0 function</th></tr>
<tr><td>0</td><td>0</td><td>P2.0</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>PWM0(PWMA channel 0)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>[7:0]</td><td><b>P2_MFP[7:0]</b></td><td>
<b>P2 Multiple Function Selection</b>
The pin function of P2 depends on P2_MFP and P2_ALT.
Refer to P2_ALT for detailed description.
</td></tr>
</table>

Multiple Function Port3 Control Register (P3_MFP)

Register	Offset	R/W	Description	Reset Value
P3_MFP	GCR_BA+0x3C	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]							

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:24]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[23:16]</td><td><b>P3_TYPEn</b></td><td><b>P3[7:0] Input Schmitt Trigger function Enable</b>
0 = P3[7:0] I/O input Schmitt Trigger function Disabled.
1 = P3[7:0] I/O input Schmitt Trigger function Enabled.</td></tr>
<tr><td>[15]</td><td><b>P3_ ALT[7]</b></td><td><b>P3.7 Alternate Function Selection</b>
The pin function of P3.7 depends on P3_MFP[7] and P3_ALT[7].
<table>
<tr><th>P3_ALT[7]</th><th>P3_MFP[7]</th><th>P3.7 function</th></tr>
<tr><td>0</td><td>0</td><td>P3.7</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[14]</td><td><b>P3_ ALT[6]</b></td><td><b>P3.6 Alternate Function Selection</b>
The pin function of P3.6 depends on P3_MFP[6] and P3_ALT[6].
<table>
<tr><th>P3_ALT[6]</th><th>P3_MFP[6]</th><th>P3.6 function</th></tr>
<tr><td>0</td><td>0</td><td>P3.6</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>CKO(Clock Driver</td></tr>
</table></td></tr>
</table>

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

<table>
<tr><td></td><td></td><td><table><tr><td></td><td></td><td>output)</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[13]</td><td><b>P3_ ALT[5]</b></td><td><b>P3.5 Alternate Function Selection</b>
The pin function of P3.5 depends on P3_MFP[5] and P3_ALT[5].<table><tr><th>P3_ALT[5]</th><th>P3_MFP[5]</th><th>P3.5 function</th></tr><tr><td>0</td><td>0</td><td>P3.5</td></tr><tr><td>0</td><td>1</td><td>T1(Timer1)</td></tr><tr><td>1</td><td>0</td><td>SCL0(I^{2}C0)</td></tr><tr><td>1</td><td>1</td><td>CKO(Clock Driver output)</td></tr></table></td></tr>
<tr><td>[12]</td><td><b>P3_ ALT[4]</b></td><td><b>P3.4 Alternate Function Selection</b>
The pin function of P3.4 depends on P3_MFP[4] and P3_ALT[4].<table><tr><th>P3_ALT[4]</th><th>P3_MFP[4]</th><th>P3.4 function</th></tr><tr><td>0</td><td>0</td><td>P3.4</td></tr><tr><td>0</td><td>1</td><td>T0(Timer0)</td></tr><tr><td>1</td><td>0</td><td>SDA0(I^{2}C0)</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[11]</td><td><b>P3_ ALT[3]</b></td><td><b>P3.3 Alternate Function Selection</b>
The pin function of P3.3 depends on P3_MFP[3] and P3_ALT[3].<table><tr><th>P3_ALT[3]</th><th>P3_MFP[3]</th><th>P3.3 function</th></tr><tr><td>0</td><td>0</td><td>P3.3</td></tr><tr><td>0</td><td>1</td><td>/INT1</td></tr><tr><td>1</td><td>0</td><td>Reserved</td></tr><tr><td>1</td><td>1</td><td>T1EX</td></tr></table></td></tr>
<tr><td>[10]</td><td><b>P3_ ALT[2]</b></td><td><b>P3.2 Alternate Function Selection</b>
The pin function of P3.2 depends on P3_MFP[2] and P3_ALT[2].<table><tr><th>P3_ALT[2]</th><th>P3_MFP[2]</th><th>P3.2 function</th></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>0</td><td>T0EX</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[9]</td><td><b>P3_ ALT[1]</b></td><td><b>P3.1 Alternate Function Selection</b></td></tr>
</table>

<table>
<tr><td></td><td></td><td>The pin function of P3.1 depends on P3_MFP[1] and P3_ALT[1].
<table><tr><th>P3_ALT[1]</th><th>P3_MFP[1]</th><th>P3.1 function</th></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>0</td><td>Reserved</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[8]</td><td><b>P3_ ALT[0]</b></td><td><b>P3.0 Alternate Function Selection</b>
The pin function of P3.0 depends on P3_MFP[0] and P3_ALT[0].
<table><tr><th>P3_ALT[0]</th><th>P3_MFP[0]</th><th>P3.0 function</th></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>0</td><td>Reserved</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[7:0]</td><td><b>P3_MFP[7:0]</b></td><td><b>P3 Multiple Function Selection</b>
The pin function of P3 depends on P3_MFP and P3_ALT.
Refer to P3_ALT for detailed description.</td></tr>
</table>

Multiple Function Port4 Control Register (P4_MFP)

Register	Offset	R/W	Description	Reset Value
P4_MFP	GCR_BA+0x40	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]							

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:24]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[23:16]</td><td><b>P4_TYPEn</b></td><td><b>P4[7:0] Input Schmitt Trigger function Enable Control</b>
0= P4[7:0] I/O input Schmitt Trigger function Disabled.
1= P4[7:0] I/O input Schmitt Trigger function Enabled.</td></tr>
<tr><td>[15]</td><td><b>P4_ ALT[7]</b></td><td><b>P4.7 Alternate Function Selection</b>
The pin function of P4.7 depends on P4_MFP[7] and P4_ALT[7].
<table>
<tr><th>P4_ALT[7]</th><th>P4_MFP[7]</th><th>P4.7 function</th></tr>
<tr><td>0</td><td>0</td><td>P4.7</td></tr>
<tr><td>0</td><td>1</td><td>ICE_DAT(ICE)</td></tr>
<tr><td>1</td><td>x</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[14]</td><td><b>P4_ ALT[6]</b></td><td><b>P4.6 Alternate Function Selection</b>
The pin function of P4.6 depends on P4_MFP[6] and P4_ALT[6].
<table>
<tr><th>P4_ALT[6]</th><th>P4_MFP[6]</th><th>P4.6 function</th></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>
</table>

<table>
<tr><td>[13]</td><td><b>P4_ ALT[5]</b></td><td><b>P4.5 Alternate Function Selection</b>
The pin function of P4.5 depends on P4_MFP[5] and P4_ALT[5].<table>
<tr><th>P4_ALT[5]</th><th>P4_MFP[5]</th><th>P4.5 function</th></tr>
<tr><td>0</td><td>0</td><td>P4.5</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>SDA1(I^{2}C1)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[12]</td><td><b>P4_ ALT[4]</b></td><td><b>P4.4 Alternate Function Selection</b>
The pin function of P4.4 depends on P4_MFP[4] and P4_ALT[4].<table>
<tr><th>P4_ALT[4]</th><th>P4_MFP[4]</th><th>P4.4 function</th></tr>
<tr><td>0</td><td>0</td><td>P4.4</td></tr>
<tr><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>SCL1(I^{2}C1)</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[11]</td><td><b>P4_ ALT[3]</b></td><td><b>P4.3 Alternate Function Selection</b>
The pin function of P4.3 depends on P4_MFP[3] and P4_ALT[3].<table>
<tr><th>P4_ALT[3]</th><th>P4_MFP[3]</th><th>P4.3 function</th></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><b>P4_ ALT[2]</b></td><td><b>P4.2 Alternate Function Selection</b>
The pin function of P4.2 depends on P4_MFP[2] and P4_ALT[2].<table>
<tr><th>P4_ALT[2]</th><th>P4_MFP[2]</th><th>P4.2 function</th></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><b>P4_ ALT[1]</b></td><td><b>P4.1 Alternate Function Selection</b>
The pin function of P4.1 depends on P4_MFP[1] and P4_ALT[1].<table>
<tr><th>P4_ALT[1]</th><th>P4_MFP[1]</th><th>P4.1 function</th></tr>
<tr><td>0</td><td>0</td><td>P4.1</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td><table>
<tr><td>0</td><td>1</td><td>PWM1(PWM generator 0)</td></tr>
<tr><td>1</td><td>0</td><td>T3EX</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[8]</td><td><b>P4_ ALT[0]</b></td><td><b>P4.0 Alternate Function Selection</b>
The pin function of P4.0 depends on P4_MFP[0] and P4_ALT[0].<table>
<tr><td><b>P4_ALT[0]</b></td><td><b>P4_MFP[0]</b></td><td><b>P4.0 function</b></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>0</td><td>T2EX</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[7:0]</td><td><b>P4_MFP[7:0]</b></td><td><b>P4 Multiple Function Selection</b>
The pin function of P4 depends on P4_MFP and P4_ALT.
Refer to P4_ALT for detailed description.</td></tr>
</table>

Multiple Function Port5 Control Register (P5_MFP)

Register	Offset	R/W	Description	Reset Value
P5_MFP	GCR_BA+0x44	R/W	P5 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**
P5_TYPE[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**P5_ALT[3:0]**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**P5_MFP[3:0]**			

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:24]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[23:16]</td><td><b>P5_TYPEn</b></td><td><b>P5[7:0] Input Schmitt Trigger function Enable Control</b>
0= P5[7:0] I/O input Schmitt Trigger function Disabled.
1= P5[7:0] I/O input Schmitt Trigger function Enabled.</td></tr>
<tr><td>[15:12]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[11]</td><td><b>P5_ ALT[3]</b></td><td><b>P5.3 Alternate Function Selection</b>
The pin function of P4.3 depends on P5_MFP[3] and P5_ALT[3].
<table>
<tr><th>P5_ALT[3]</th><th>P5_MFP[3]</th><th>P5.3 function</th></tr>
<tr><td>0</td><td>0</td><td>P5.3</td></tr>
<tr><td>0</td><td>1</td><td>SCL0(I^{2}C0)</td></tr>
<tr><td>1</td><td>x</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>[10]</td><td><b>P5_ ALT[2]</b></td><td><b>P5.2 Alternate Function Selection</b>
The pin function of P5.2 depends on P5_MFP[2] and P5_ALT[2].
<table>
<tr><th>P4_ALT[2]</th><th>P4_MFP[2]</th><th>P4.2 function</th></tr>
<tr><td>0</td><td>0</td><td>P5.2</td></tr>
<tr><td>0</td><td>1</td><td>SDA0(I^{2}C0)</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td><table><tr><td>1</td><td>x</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[9]</td><td><b>P5_ ALT[1]</b></td><td><b>P5.1 Alternate Function Selection</b>
The pin function of P4.1 depends on P5_MFP[1] and P5_ALT[1].<table><tr><th>P5_ALT[1]</th><th>P5_MFP[1]</th><th>P5.1 function</th></tr><tr><td>0</td><td>0</td><td>P5.1</td></tr><tr><td>0</td><td>1</td><td>T1EX</td></tr><tr><td>1</td><td>0</td><td>Reserved</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[8]</td><td><b>P5_ ALT[0]</b></td><td><b>P5.0 Alternate Function Selection</b>
The pin function of P5.0 depends on P5_MFP[0] and P5_ALT[0].<table><tr><th>P5_ALT[0]</th><th>P5_MFP[0]</th><th>P5.0 function</th></tr><tr><td>0</td><td>0</td><td>P5.0</td></tr><tr><td>0</td><td>1</td><td>T0EX</td></tr><tr><td>1</td><td>0</td><td>Reserved</td></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr></table></td></tr>
<tr><td>[7:4]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[3:0]</td><td><b>P5_MFP[3:0]</b></td><td><b>P5 Multiple Function Selection</b>
The pin function of P5 depends on P5_MFP and P5_ALT.
Refer to P5_ALT for detailed description.</td></tr>
</table>

Multiple Function Port6 Control Register (P6_MFP)

Register	Offset	R/W	Description	Reset Value
P6_MFP	GCR_BA+0x48	R/W	P6 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**
P6_TYPE[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**P6_TYPEn**	**P6[7:0] Input Schmitt Trigger function Enable Control** 0= P6[7:0] I/O input Schmitt Trigger function Disabled. 1= P6[7:0] I/O input Schmitt Trigger function Enabled.
[15:0]	**Reserved**	Reserved.

Multiple Function Port7 Control Register (P7_MFP)

Register	Offset	R/W	Description	Reset Value
P7_MFP	GCR_BA+0x4C	R/W	P7 Multiple Function and Input Type Control Register	0x0000_0003

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**P7_TYPE[1:0]**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**P7_MFP[1:0]**	

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**P7_TYPEn**	**P7[1:0] Input Schmitt Trigger function Enable Control** 0= P7.0 and P7.1 input Schmitt Trigger function Disabled. 1= P7.0 and P7.1 input Schmitt Trigger function Enabled.
[15:2]	**Reserved**	Reserved.
[1:0]	**P7_MFP[1:0]**	**P7 Multiple Function Selection** The pin function of P7 depends on CONFIG0[27] 0= P7.0 and P7.1 pins are configured as GPIO pins. 1 = P7.0 and P7.1 pins are configured as external 4~24MHz (high speed) crystal pins.

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.

Write 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+0x100	R/W	Register Write-Protection Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[7:0]	**REGWRPROT**	**Register Write-Protection Code (Write Only)** Some registers have write-protection function. Writing these registers have to disable the protected function by writing the sequence value “59h”, “16h”, “88h” to this field. After this sequence is completed, the REGPROTDIS bit will be set to 1 and write-protection registers can be normal write.
[0]	**REGPROTDIS**	**Register Write-Protection Disable index (Read Only)** 0 = Write-protection Enabled for writing protected registers. Any write to the protected register is ignored. 1 = Write-protection Disabled for writing protected registers.

The Protected registers are:

Registers	Address	Note
IPRSTC1	0x5000_0008	
BODCR	0x5000_0018	
PORCR	0x5000_0024	
PWRCON	0x5000_0200	Bit[6] is not protected for power wake-up interrupt clear.
APBCLK bit[0]	0x5000_0208	Bit[0] is watchdog clock enable.
CLKSEL0	0x5000_0210	HCLK and CPU STCLK clock source select.
CLKSEL1 bit[1:0]	0x5000_0214	Watchdog clock source select.
NMI_SEL bit[8]	0x5000_0380	NMI interrupt enable.
ISPCON	0x5000_C000	Flash ISP Control.
ISPTRG	0x5000_C010	ISP Trigger Control.
WTCR	0x4000_4000	Watchdog Timer Control.
FATCON	0x5000_C018	Flash Access Time Control.

Note: The bits which are write-protected will be noted as” **(Write Protect)**” beside the description.

6.2.7 System Timer (SysTick)

The Cortex®-M0 includes an integrated system timer, SysTick, which provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used as a Real Time Operating System (RTOS) tick timer or as a simple counter.

When system timer is enabled, it will count down from the value in the SysTick Current Value Register (SYST_CVR) to 0, and reload (wrap) to the value in the SysTick Reload Value Register (SYST_RVR) on the next clock cycle, then decrement on subsequent clocks. When the counter transitions to 0, the COUNTFLAG status bit is set. The COUNTFLAG bit clears on reads.

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

If the SYST_RVR is 0, the timer will be maintained with a current value of 0 after it is reloaded with this value. This mechanism can be used to disable the feature independently from the timer enable bit.

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

6.2.8 System Timer Control Register Map

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

Register	Offset	R/W	Description	Reset Value
SYST Base Address: **SYST_BA = 0xE000_E010**				
SYST_CSR	SYST_BA+0x00	R/W	SysTick Control and Status Register	0x0000_0000
SYST_RVR	SYST_BA+0x04	R/W	SysTick Reload Value Register	0xXXXX_XXXX
SYST_CVR	SYST_BA+0x08	R/W	SysTick Current Value Register	0xXXXX_XXXX

SysTick Control and Status （SYST_CSR）

Register	Offset	R/W	Description	Reset Value
SYST_CSR	SYST_BA+0x00	R/W	SysTick Control and Status Register	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**COUNTFLAG**	**System Tick Counter Flag** Returns 1 if timer counted to 0 since last time this register was read. COUNTFLAG is set by a count transition from 1 to 0. COUNTFLAG is cleared on read or by a write to the Current Value register.
[15:3]	**Reserved**	Reserved.
[2]	**CLKSRC**	**System Tick Clock Source Selection** 0 = Clock source is optional, refer to STCLK_S. 1 = Core clock used for SysTick timer.
[1]	**TICKINT**	**System Tick Interrupt Enabled** 0 = Counting down to 0 does not cause the SysTick exception to be pended. Software can use COUNTFLAG to determine if a count to 0 has occurred. 1 = Counting down to 0 will cause the SysTick exception to be pended. Clearing the SysTick Current Value register by a register write in software will not cause SysTick to be pended.
[0]	**ENABLE**	**System Tick Counter Enabled** 0 = Counter Disabled. 1 = Counter Enabled and will operate in a multi-shot manner.

SysTick Reload Value Register（SYST_RVR）

Register	Offset	R/W	Description	Reset Value
SYST_RVR	SYST_BA+0x04	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	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**RELOAD**	**System Tick Reload Value** Value to load into the Current Value register when the counter reaches 0.

SysTick Current Value Register （SYST_CVR）

Register	Offset	R/W	Description	Reset Value
SYST_CVR	SYST_BA+0x08	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	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CURRENT**	**System Tick Current Value** Current counter value. This is the value of the counter at the time it is sampled. The counter does not provide read-modify-write protection. The register is write-clear. A software write of any value will clear the register to 0. Unsupported bits RAZ (see SysTick Reload Value register).

6.2.9 Nested Vectored Interrupt Controller (NVIC)

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

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

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

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

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

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

6.2.9.1 Exception Model and System Interrupt Map

The following table lists the exception model supported by NuMicro M058S™ series. Software can set four levels of priority on some of these exceptions as well as on all interrupts. The highest user-configurable priority is denoted as "0" and the lowest priority is denoted as "3". The default priority of all the user-configurable interrupts is "0". Note that priority "0" is treated as the fourth priority on the system, after three system exceptions "Reset", "NMI" and "Hard Fault".

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

PendSV	14	Configurable
SysTick	15	Configurable
Interrupt (IRQ0 ~ IRQ31)	16 ~ 47	Configurable

Table 6.2-2 Exception Model

Exception Number	Vector Address	Interrupt Number (Bit in Interrupt Registers)	Interrupt Name	Source Module	Interrupt description	Power-down Wakeup
1-15					System exceptions	
16	0x40	0	BOD_INT	Brown-out	Brown-out low voltage detected interrupt	Yes
17	0x44	1	WDT_INT	WDT	Watchdog 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	Reserved	-	-	-
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	Reserved	-	-	-
30	0x78	14	SPI0_INT	SPI0	SPI0 interrupt	No
31	0x7C	15	Reserved	-	-	-
32	0x80	16	GP5_INT	GPIO	External signal interrupt from P5[7:0]	Yes

33	0x84	17	GP67_INT	GPIO	External signal interrupt from P6[7:0] / P7[1:0]	Yes
34	0x88	18	I2C0_INT	I^{2}C0	I^{2}C0 interrupt	Yes
35	0x8C	19	I2C1_INT	I^{2}C1	I^{2}C1 interrupt	Yes
36	0x90	20	CAP0_INT	PWM	PWM0 capture in interrupt	No
37	0x94	21	CAP1_INT	PWM	PWM1 capture in interrupt	No
38	0x98	22	CAP2_INT	PWM	PWM2 capture in interrupt	No
39	0x9C	23	CAP3_INT	PWM	PWM3 capture in interrupt	No
40-43	0x90-0xAC	20-27	Reserved	-	-	-
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	Reserved	-	-	

Table 6.2-3 System Interrupt Map Vector Table

6.2.9.2 *Vector Table*

When an interrupt is accepted, the processor will automatically fetch the starting address of the interrupt service routine (ISR) from a vector table in memory. For ARMv6-M, the vector table base address is fixed at 0x00000000. The vector table contains the initialization value for the stack pointer on reset, and the entry point addresses for all exception handlers. The vector number on previous page defines the order of entries in the vector table associated with exception handler entry as illustrated in previous section.

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

Table 6.2-4 Vector Figure Format

6.2.9.3 *Operation Description*

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

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

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

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

6.2.9.4 NVIC Control Register Map

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

Register	Offset	R/W	Description	Reset Value
NVIC Base Address: **NVIC_BA = 0xE000_E100**				
NVIC_ISER	NVIC_BA+0x000	R/W	IRQ0 ~ IRQ31 Set-Enable Control Register	0x0000_0000
NVIC_ICER	NVIC_BA+0x080	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000
NVIC_ISPR	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000
NVIC_ICPR	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000
NVIC_IPR0	NVIC_BA+0x300	R/W	IRQ0 ~ IRQ3 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR1	NVIC_BA+0x304	R/W	IRQ4 ~ IRQ7 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR2	NVIC_BA+0x308	R/W	IRQ8 ~ IRQ11 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR3	NVIC_BA+0x30C	R/W	IRQ12 ~ IRQ15 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR4	NVIC_BA+0x310	R/W	IRQ16 ~ IRQ19 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR5	NVIC_BA+0x314	R/W	IRQ20 ~ IRQ23 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR6	NVIC_BA+0x318	R/W	IRQ24 ~ IRQ27 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR7	NVIC_BA+0x31C	R/W	IRQ28 ~ IRQ31 Interrupt Priority Control Register	0x0000_0000

IRQ0 ~ IRQ31 Set-Enable Control Register （NVIC_ISER）

Register	Offset	R/W	Description	Reset Value
NVIC_ISER	NVIC_BA+0x000	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	Description	
[31:0]	**SETENA**	**Interrupt Enable Register** Enable one or more interrupts. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Write: 0 = No effect. 1 = Write 1 to enable associated interrupt. Read: 0 = Associated interrupt status is Disabled. 1 = Associated interrupt status is Enabled. Read value indicates the current enable status.

IRQ0 ~ IRQ31 Clear-Enable Control Register (NVIC_ICER)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER	NVIC_BA+0x080	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	Description	
[31:0]	CLRENA	**Interrupt Disable Register** Disable one or more interrupts. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Write: 0 = No effect. 1 = Write 1 to disable associated interrupt. Read: 0 = Associated interrupt status is Disabled. 1 = Associated interrupt status is Enabled. Read value indicates the current enable status.

IRQ0 ~ IRQ31 Set-Pending Control Register（NVIC_ISPR）

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR	NVIC_BA+0x100	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	Description	
[31:0]	**SETPEND**	**Set Interrupt Pending register** Write: 0 = No effect. 1 = Write 1 to set pending state. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Read: 0 = Associated interrupt in not in pending status. 1 = Associated interrupt is in pending status. Read value indicates the current pending status.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

IRQ0 ~ IRQ31 Clear-Pending Control Register （NVIC_ICPR）

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR	NVIC_BA+0x180	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	Description	
[31:0]	**CLRPEND**	**Clear Interrupt Pending register** Write: 0 = No effect. 1 = Write 1 to clear pending state. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Read: 0 = Associated interrupt in not in pending status. 1 = Associated interrupt is in pending status. Read value indicates the current pending status.

IRQ0 ~ IRQ3 Interrupt Priority Register （NVIC_IPR0）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR0	NVIC_BA+0x300	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	Description	
[31:30]	**PRI_3**	**Priority of IRQ3** “0” denotes the highest priority and “3” denotes lowest priority.
[23:22]	**PRI_2**	**Priority of IRQ2** “0” denotes the highest priority and “3” denotes lowest priority.
[15:14]	**PRI_1**	**Priority of IRQ1** “0” denotes the highest priority and “3” denotes lowest priority.
[7:6]	**PRI_0**	**Priority of IRQ0** “0” denotes the highest priority and “3” denotes lowest priority.

IRQ4 ~ IRQ7 Interrupt Priority Register （NVIC_IPR1）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR1	NVIC_BA+0x304	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	Description	
[31:30]	**PRI_7**	**Priority of IRQ7** “0” denotes the highest priority and “3” denotes lowest priority.
[23:22]	**PRI_6**	**Priority of IRQ6** “0” denotes the highest priority and “3” denotes lowest priority.
[15:14]	**PRI_5**	**Priority of IRQ5** “0” denotes the highest priority and “3” denotes lowest priority.
[7:6]	**PRI_4**	**Priority of IRQ4** “0” denotes the highest priority and “3” denotes lowest priority.

IRQ8 ~ IRQ11 Interrupt Priority Register （NVIC_IPR2）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR2	NVIC_BA+0x308	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	Description	
[31:30]	**PRI_11**	**Priority of IRQ11** “0” denotes the highest priority and “3” denotes lowest priority.
[23:22]	**PRI_10**	**Priority of IRQ10** “0” denotes the highest priority and “3” denotes lowest priority.
[15:14]	**PRI_9**	**Priority of IRQ9** “0” denotes the highest priority and “3” denotes lowest priority.
[7:6]	**PRI_8**	**Priority of IRQ8** “0” denotes the highest priority and “3” denotes lowest priority.

IRQ12 ~ IRQ15 Interrupt Priority Register（NVIC_IPR3）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR3	NVIC_BA+0x30C	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	Description	
[31:30]	**PRI_15**	**Priority of IRQ15** "0" denotes the highest priority and "3" denotes lowest priority.
[23:22]	**PRI_14**	**Priority of IRQ14** "0" denotes the highest priority and "3" denotes lowest priority.
[15:14]	**PRI_13**	**Priority of IRQ13** "0" denotes the highest priority and "3" denotes lowest priority.
[7:6]	**PRI_12**	**Priority of IRQ12** "0" denotes the highest priority and "3" denotes lowest priority.

IRQ16 ~ IRQ19 Interrupt Priority Register （NVIC_IPR4）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR4	NVIC_BA+0x310	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	Description	
[31:30]	**PRI_19**	**Priority of IRQ19** “0” denotes the highest priority and “3” denotes lowest priority.
[23:22]	**PRI_18**	**Priority of IRQ18** “0” denotes the highest priority and “3” denotes lowest priority.
[15:14]	**PRI_17**	**Priority of IRQ17** “0” denotes the highest priority and “3” denotes lowest priority.
[7:6]	**PRI_16**	**Priority of IRQ16** “0” denotes the highest priority and “3” denotes lowest priority.

IRQ20 ~ IRQ23 Interrupt Priority Register （NVIC_IPR5）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR5	NVIC_BA+0x314	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	**Description**	
[31:30]	**PRI_23**	**Priority of IRQ23** “0” denotes the highest priority and “3” denotes lowest priority.
[23:22]	**PRI_22**	**Priority of IRQ22** “0” denotes the highest priority and “3” denotes lowest priority.
[15:14]	**PRI_21**	**Priority of IRQ21** “0” denotes the highest priority and “3” denotes lowest priority.
[7:6]	**PRI_20**	**Priority of IRQ20** “0” denotes the highest priority and “3” denotes lowest priority.

IRQ24 ~ IRQ27 Interrupt Priority Register （NVIC_IPR6）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR6	NVIC_BA+0x318	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	Description	
[31:30]	**PRI_27**	**Priority of IRQ27** “0” denotes the highest priority and “3” denotes lowest priority.
[23:22]	**PRI_26**	**Priority of IRQ26** “0” denotes the highest priority and “3” denotes lowest priority.
[15:14]	**PRI_25**	**Priority of IRQ25** “0” denotes the highest priority and “3” denotes lowest priority.
[7:6]	**PRI_24**	**Priority of IRQ24** “0” denotes the highest priority and “3” denotes lowest priority.

IRQ28 ~ IRQ31 Interrupt Priority Register （NVIC_IPR7）

Register	Offset	R/W	Description	Reset Value
NVIC_IPR7	NVIC_BA+0x31C	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	Description	
[31:30]	**PRI_31**	**Priority of IRQ31** “0” denotes the highest priority and “3” denotes lowest priority.
[23:22]	**PRI_30**	**Priority of IRQ30** “0” denotes the highest priority and “3” denotes lowest priority.
[15:14]	**PRI_29**	**Priority of IRQ29** “0” denotes the highest priority and “3” denotes lowest priority.
[7:6]	**PRI_28**	**Priority of IRQ28** “0” denotes the highest priority and “3” denotes lowest priority.

Interrupt Source Control Registers

Besides the interrupt control registers associated with the NVIC, the NuMicro M058S™ series also implements some specific control registers to facilitate the interrupt functions, including "interrupt source identification", "NMI source selection" and "interrupt test mode", which are described below.

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

Register	Offset	R/W	Description	Reset Value
INT Base Address: **INT_BA = 0x5000_0300**				
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) Interrupt Source Identity	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) Interrupt Source Identity	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) Interrupt Source Identity	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) Interrupt Source Identity	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (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	Reserved	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) Interrupt Source Identity	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) Interrupt Source Identity	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) Interrupt Source Identity	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) Interrupt Source Identity	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0) Interrupt Source Identity	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	Reserved	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) Interrupt Source Identity	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	Reserved	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	IRQ16 (P5) Interrupt Source Identity	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (P6/7) Interrupt Source Identity	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I^{2}C0) Interrupt Source Identity	0xXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I2C1) Interrupt Source Identity	0xXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	R	IRQ20 (CAP0) Interrupt Source Identity	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (CAP1) Interrupt Source Identity	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (CAP2) Interrupt Source Identity	0xXXXX_XXXX

IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (CAP3) Interrupt Source Identity	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	R	Reserved.	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	Reserved	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	Reserved	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	Reserved.	0xXXXX_XXXX
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) Interrupt Source Identity	0xXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) Interrupt Source Identity	0xXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	R	Reserved.	0xXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	R	Reserved.	0xXXXX_XXXX
NMI_SEL	INT_BA+0x80	R/W	NMI Source Interrupt Select Control Register	0x0000_0000

IRQ0 (BOD) Interrupt Source Identity Register (IRQ0_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ0 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ0 source is not from BOD interrupt (BOD_INT). 1 = IRQ0 source is from BOD interrupt (BOD_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ1 (WDT) Interrupt Source Identity Register (IRQ1_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ1 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ1 source is not from watchdog interrupt (WDT _INT). 1 = IRQ1 source is from watchdog interrupt (WDT_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ2 (EINT0) Interrupt Source Identity Register (IRQ2_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ2 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ2 source is not from external signal interrupt 0 – P3.2 (EINT0). 1 = IRQ2 source is from external signal interrupt 0 – P3.2 (EINT0). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ3 (EINT1) Interrupt Source Identity Register (IRQ3_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ3 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ3 source is not from external signal interrupt 1 – P3.3 (EINT1). 1 = IRQ3 source is from external signal interrupt 1 – P3.3 (EINT1). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ4 (P0/1) Interrupt Source Identity Register (IRQ4_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (P0/1) 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[2:0]		

Bits	Description	
[2:0]	INT_SRC	**IRQ4 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1]: 0 = IRQ4 source is not from P1 interrupt (P1_INT). 1 = IRQ4 source is from P1 interrupt (P1_INT). INT_SRC[0]: 0 = IRQ4 source is not from P0 interrupt (P0_INT). 1 = IRQ4 source is from P0 interrupt (P0_INT). **Note1:** IRQ4 source can be from multiple interrupt sources at the same time. **Note2:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ5 (P2/3/4) Interrupt Source Identity Register (IRQ5_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (P2/3/4) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ5 Source Identity** INT_SRC[2]: 0 = IRQ5 source is not from P4 interrupt (P4_INT). 1 = IRQ5 source is from P4 interrupt (P4_INT). INT_SRC[1]: 0 = IRQ5 source is not from P3 interrupt (P3_INT). 1 = IRQ5 source is from P3 interrupt (P3_INT). INT_SRC[0]: 0 = IRQ5 source is not from P2 interrupt (P2_INT). 1 = IRQ5 source is from P2 interrupt (P2_INT). **Note1:** IRQ5 source can be from multiple interrupt sources at the same time. **Note2:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ6 (PWMA) Interrupt Source Identity Register (IRQ6_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWMA) 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[3:0]**			

Bits	Description	
[3:0]	**INT_SRC**	**IRQ6 Source Identity** INT_SRC[3]: 0 = IRQ6 source is not from PWM3(PWMA channel 3) interrupt (PWM3_INT). 1 = IRQ6 source is from PWM3(PWMA channel 3) interrupt (PWM3_INT). INT_SRC[2]: 0 = IRQ6 source is not from PWM2(PWMA channel 2) interrupt (PWM2_INT). 1 = IRQ6 source is from PWM2(PWMA channel 2) interrupt (PWM2_INT). INT_SRC[1]: 0 = IRQ6 source is not from PWM1(PWMA channel 1) interrupt (PWM1_INT). 1 = IRQ6 source is from PWM1(PWMA channel 1) interrupt (PWM1_INT). INT_SRC[0]: 0 = IRQ6 source is not from PWM0(PWMA channel 0) interrupt (PWM0_INT). 1 = IRQ6 source is from PWM0(PWMA channel 0) interrupt (PWM0_INT). **Note1**: IRQ6 source can be from multiple interrupt sources at the same time. **Note2:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ8 (TMR0) Interrupt Source Identity Register (IRQ8_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ8 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ8 source is not from Timer0 interrupt (TMR0_INT). 1 = IRQ8 source is from Timer0 interrupt (TMR0_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ9 (TMR1) Interrupt Source Identity Register (IRQ9_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ9 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ9 source is not from Timer1 interrupt (TMR1_INT). 1 = IRQ9 source is from Timer1 interrupt (TMR1_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ10 (TMR2) Interrupt Source Identity Register (IRQ10_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ10 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ10 source is not from Timer2 interrupt (TMR2_INT). 1 = IRQ10 source is from Timer2 interrupt (TMR2_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ11 (TMR3) Interrupt Source Identity Register (IRQ11_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ11 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ11 source is not from Timer3 interrupt (TMR3_INT). 1 = IRQ11 source is from Timer3 interrupt (TMR3_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ12 (UART0) Interrupt Source Identity Register (IRQ12_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ12 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ12 source is not from UART0 interrupt (UART0_INT). 1 = IRQ12 source is from UART0 interrupt (UART0_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ14 (SPI0) Interrupt Source Identity Register (IRQ14_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ14 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ14 source is not from SPI0 interrupt (SPI0_INT). 1 = IRQ14 source is from SPI0 interrupt (SPI0_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ16 (P5) Interrupt Source Identity Register (IRQ16_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ16_SRC	INT_BA+0x10	R	IRQ16 (P5) 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[0]**

Bits	Description	
[2:0]	**INT_SRC**	**IRQ16 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ16 source is not from P5 interrupt (P5_INT). 1 = IRQ16 source is from P5 interrupt (P5_INT). **Note1:** IRQ16 source can be from multiple interrupt sources at the same time. **Note2:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ17 (P6/7) Interrupt Source Identity Register (IRQ17_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (P6/7) 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[1:0]**	

Bits	Description	
[2:0]	**INT_SRC**	**IRQ5 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1]: 0 = IRQ17 source is not from P3 interrupt (P7_INT). 1 = IRQ17 source is from P3 interrupt (P7_INT). INT_SRC[0]: 0 = IRQ17 source is not from P6 interrupt (P6_INT). 1 = IRQ17 source is from P6 interrupt (P6_INT). **Note1:** IRQ17 source can be from multiple interrupt sources at the same time. **Note2:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ18 (I^2C0) Interrupt Source Identity Register (IRQ18_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I^2C0) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ18 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ18 source is not from I^2C0 interrupt (I2C0_INT). 1 = IRQ18 source is from I^2C0 interrupt (I2C0_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ19 (I^{2}C1) Interrupt Source Identity Register (IRQ19_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I^{2}C1) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ19 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ19 source is not from I^{2}C1 interrupt (I2C1_INT). 1 = IRQ19 source is from I^{2}C1 interrupt (I2C1_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ20(CAP0) Interrupt Source Identity Register (IRQ20_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ20_SRC	INT_BA+0x50	R	IRQ20 (CAP0) 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[2:0]		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ20 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ20 source is not from PWM0 Cpature interrupt (CAP0_INT). 1 = IRQ20 source is from PWM0 Cpature interrupt (CAP0_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ21(CAP1) Interrupt Source Identity Register (IRQ21_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ21_SRC	INT_BA+0x54	R	IRQ20 (CAP1) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ20 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ21 source is not from PWM1 Cpature interrupt (CAP1_INT). 1 = IRQ21 source is from PWM1 Cpature interrupt (CAP1_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ22(CAP2) Interrupt Source Identity Register (IRQ22_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (CAP2) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ20 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ22 source is not from PWM2 Cpature interrupt (CAP2_INT). 1 = IRQ22 source is from PWM2 Cpature interrupt (CAP2_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ23(CAP3) Interrupt Source Identity Register (IRQ23_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (CAP0) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ23 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ23 source is not from PWM3 Cpature interrupt (CAP3_INT). 1 = IRQ23 source is from PWM3 Cpature interrupt (CAP3_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

IRQ28 (PWRWU) Interrupt Source Identity Register (IRQ28_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ28 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ28 source is not from Power-down mode Wake-up interrupt (PWRWU_INT). 1 = IRQ28 source is from Power-down mode Wake-up interrupt interrupt (PWRWU_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

IRQ29 (ADC) Interrupt Source Identity Register (IRQ29_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) 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[2:0]**		

Bits	Description	
[2:0]	**INT_SRC**	**IRQ29 Source Identity** INT_SRC[2] = Reserved. INT_SRC[1] = Reserved. INT_SRC[0]: 0 = IRQ29 source is not from ADC interrupt (ADC_INT). 1 = IRQ29 source is from ADC interrupt (ADC_INT). **Note:** When the interrupt flag is cleared, the corresponding bits will be cleared automatically.

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							**NMI_EN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**NMI_SEL[4:0]**				

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

6.2.10 System Control Block (SCB)

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

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

6.2.10.1 System Control Block Register Map

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

Register	Offset	R/W	Description	Reset Value
SCB Base Address: **SCB_BA = 0xE000_ED00**				
CPUID	SCB_BA+0x000	R	CPUID Register	0x410CC200
ICSR	SCB_BA+0x004	R/W	Interrupt Control State Register	0x0000_0000
AIRCR	SCB_BA+0x00C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000
SCR	SCB_BA+0x010	R/W	System Control Register	0x0000_0000
SHPR2	SCB_BA+0x01C	R/W	System Handler Priority Register 2	0x0000_0000
SHPR3	SCB_BA+0x020	R/W	System Handler Priority Register 3	0x0000_0000

CPUID Base Register (CPUID)

Register	Offset	R/W	Description	Reset Value
CPUID	SCB_BA+0x000	R	CPUID Register	0x410CC200

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8"><b>IMPLEMENTER[7:0]</b></td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="4"><b>Reserved</b></td><td colspan="4"><b>PART[3:0]</b></td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="8"><b>PARTNO[11:4]</b></td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="4"><b>PARTNO[3:0]</b></td><td colspan="4"><b>REVISION[3:0]</b></td></tr>
</table>

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

Interrupt Control State Register (ICSR)

Register	Offset	R/W	Description	Reset Value
ICSR	SCB_BA+0x004	R/W	Interrupt Control State Register	0x0000_0000

31	30	29	28	27	26	25	24
NMIPENDSET	**Reserved**		**PENDSVSET**	**PENDSVCLR**	**PENDSTSET**	**PENDSTCLR**	**Reserved**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISRPREEMPT	**ISRPENDING**	**Reserved**				**VECTPENDING[5:4]**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VECTPENDING[3:0]				**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**VECTACTIVE[5:0]**					

Bits	Description	
[31]	**NMIPENDSET**	**NMI Set-pending Bit** Write: 0 = No effect. 1 = Changes NMI exception state to pending. Read: 0 = NMI exception not pending. 1 = NMI exception pending. **Note:** Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it detects a write of 1 to this bit. Entering the handler then clears this bit to 0. This means a read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler.
[30:29]	**Reserved**	Reserved.
[28]	**PENDSVSET**	**PendSV Set-pending Bit** Write: 0 = No effect. 1 = Changes PendSV exception state to pending. Read: 0 = PendSV exception is not pending. 1 = PendSV exception is pending. **Note:** Writing 1 to this bit is the only way to set the PendSV exception state to pending

[27]	**PENDSVCLR**	**PendSV Clear-pending Bit** Write: 0 = No effect. 1 = Removes the pending state from the PendSV exception. This bit is write-only. To clear the PENDSV bit, you must “write 0 to PENDSVSET and write 1 to PENDSVCLR” at the same time.
[26]	**PENDSTSET**	**SysTick Exception Set-pending Bit** Write: 0 = No effect. 1 = Changes SysTick exception state to pending. Read: 0 = SysTick exception is not pending. 1 = SysTick exception is pending.
[25]	**PENDSTCLR**	**SysTick Exception Clear-pending Bit** Write: 0 = No effect. 1 = Removes the pending state from the SysTick exception. **Note:** This bit is write-only. When you want to clear PENDST bit, you must “write 0 to PENDSTSET and write 1 to PENDSTCLR” at the same time.
[24]	**Reserved**	Reserved.
[23]	**ISRPREEMPT**	**Interrupt Preempt Bit** If set, a pending exception will be serviced on exit from the debug halt state. This bit is read only.
[22]	**ISRPENDING**	**Interrupt Pending Flag, excluding NMI and Faults (Read Only)** 0 = Interrupt not pending. 1 = Interrupt pending.
[21:18]	**Reserved**	Reserved.
[17:12]	**VECTPENDING**	**Exception Number of the Highest Priority Pending Enabled Exception** 0 = No pending exceptions. Non-zero = Exception number of the highest priority pending enabled exception.
[11:6]	**Reserved**	Reserved.
[5:0]	**VECTACTIVE**	**Contains the Active Exception Number** 0 = Thread mode. Non-zero = Exception number of the currently active exception.

Application Interrupt and Reset Control Register (AIRCR)

Register	Offset	R/W	Description	Reset Value
AIRCR	SCB_BA+0x00C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000

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

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

System Control Register (SCR)

Register	Offset	R/W	Description	Reset Value
SCR	SCB_BA+0x010	R/W	System Control Register	0x0000_0000

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

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**SEVONPEND**	**Send Event on Pending Bit** 0 = Only enabled interrupts or events can wake-up the processor, disabled interrupts are excluded. 1 = Enabled events and all interrupts, including disabled interrupts, can wake-up the processor. When an event or interrupt enters pending state, the event signal wakes up the processor from WFE. If the processor is not waiting for an event, the event is registered and affects the next WFE. The processor also wakes up on execution of an SEV instruction or an external event.
[3]	**Reserved**	Reserved.
[2]	**SLEEPDEEP**	**Processor Deep Sleep and Sleep Mode Selection** Controls whether the processor uses sleep or deep sleep as its low power mode: 0 = Sleep mode. 1 = Deep Sleep mode.
[1]	**SLEEPONEXIT**	**Sleep-on-exit Enable Control** This bit indicates sleep-on-exit when returning from Handler mode to Thread mode. 0 = Do not sleep when returning to Thread mode. 1 = Enter Sleep or Deep Sleep when returning from ISR to Thread mode. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.
[0]	**Reserved**	Reserved.

System Handler Priority Register 2 (SHPR2)

Register	Offset	R/W	Description	Reset Value
SHPR2	SCB_BA+0x01C	R/W	System Handler Priority Register 2	0x0000_0000

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

Bits	Description	
[31:30]	**PRI_11**	**Priority of System Handler 11 – SVCall** “0” denotes the highest priority and “3” denotes the lowest priority
[29:0]	**Reserved**	Reserved.

System Handler Priority Register 3 (SHPR3)

Register	Offset	R/W	Description	Reset Value
SHPR3	SCB_BA+0x020	R/W	System Handler Priority Register 3	0x0000_0000

31	30	29	28	27	26	25	24
PRI_15		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_14		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:30]	**PRI_15**	**Priority of System Handler 15 – SysTick** "0" denotes the highest priority and "3" denotes the lowest priority
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_14**	**Priority of System Handler 14 – PendSV** "0" denotes the highest priority and "3" denotes the lowest priority
[21:0]	**Reserved**	Reserved.

6.3 Clock Controller

6.3.1 Overview

The clock controller generates clocks for the whole chip, including system clocks and all peripheral clocks. The clock controller also implements the power control function with the individually clock ON/OFF control, clock source selection and clock divider. The chip enters Power-down mode when Cortex®-M0 core executes the WFI instruction only if the PWR_DOWN_EN (PWRCON[7]) bit and PD_WAIT_CPU (PWRCON[8]) bit are both set to 1. After that, chip enters Power-down mode and waits for wake-up interrupt source triggered to exit Power-down mode. In Power-down mode, the clock controller turns off the 4~24 MHz external high speed crystal (HXT) and 22.1184 MHz internal high speed RC oscillator (HIRC) to reduce the overall system power consumption. The following figures show the clock generator and the overview of the clock source control.

The clock generator consists of 4 clock sources as listed below:

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

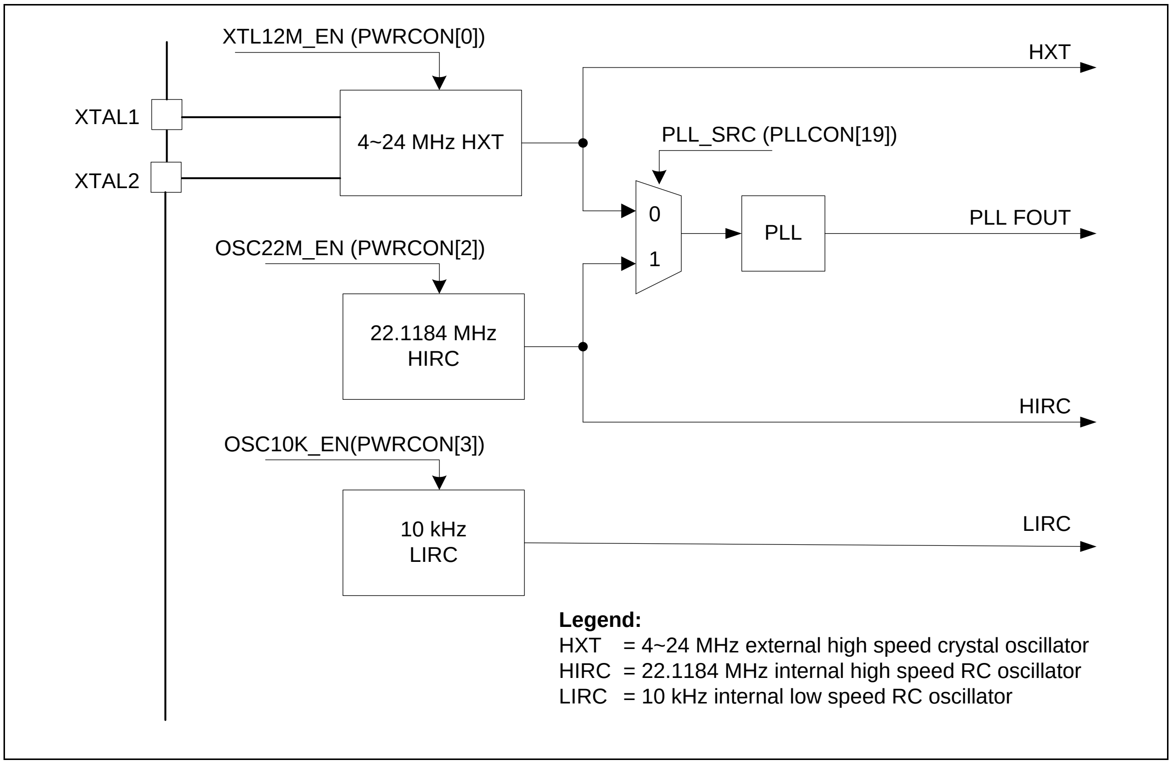


Figure 6.3-1 Clock Generator Block Diagram

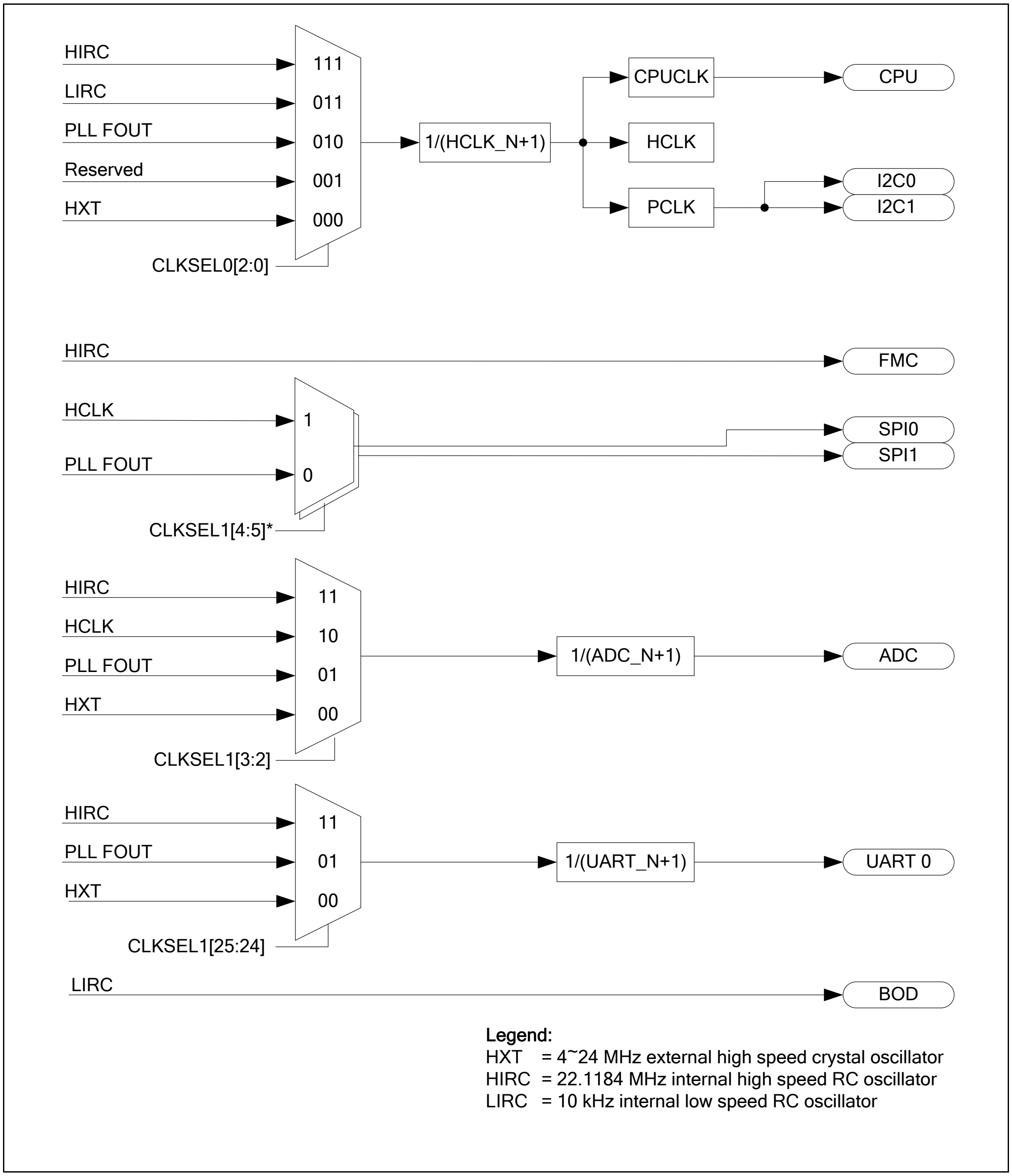
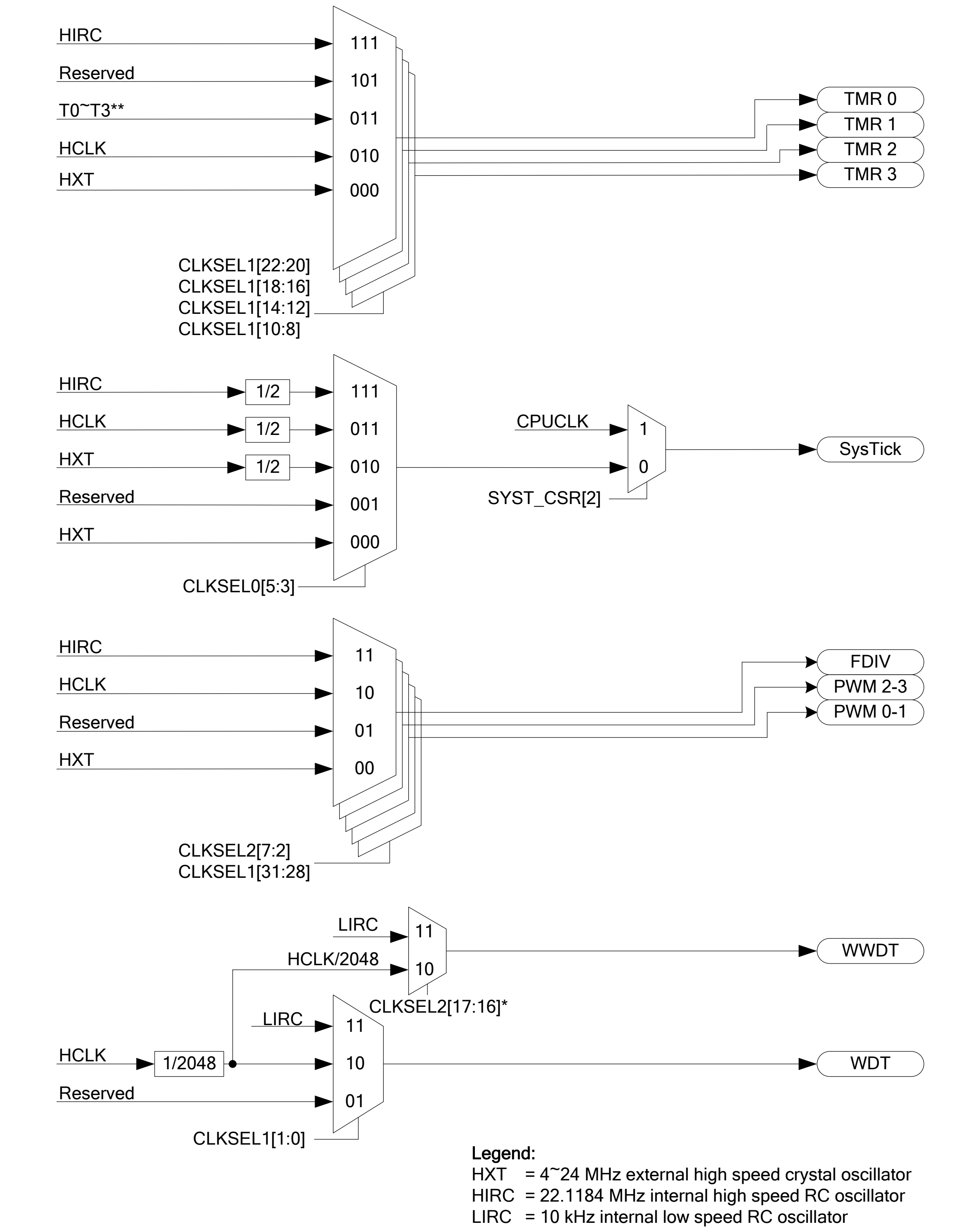


Figure 6.3-2 Clock Source Controller Overview (1/2)



Note1: T0 is for TMR0, T1 is for TMR1, T2 is for TMR2 and T3 is for TMR3.

Figure 6.3-3 Clock Source Controller Overview (2/2)

6.3.2 System Clock and SysTick Clock

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

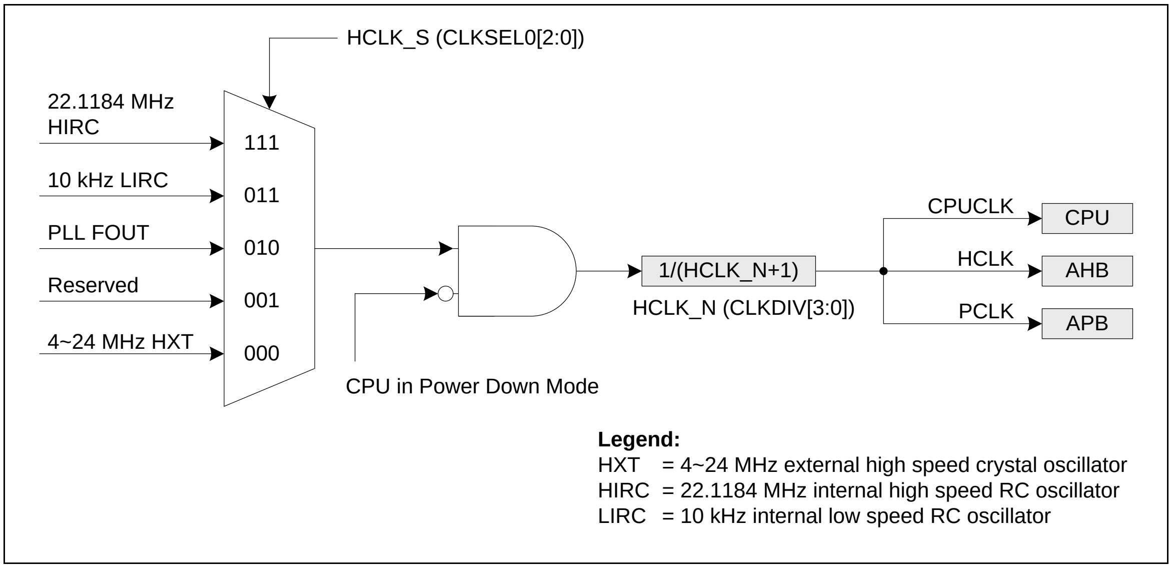


Figure 6.3-4 System Clock Block Diagram

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 below.

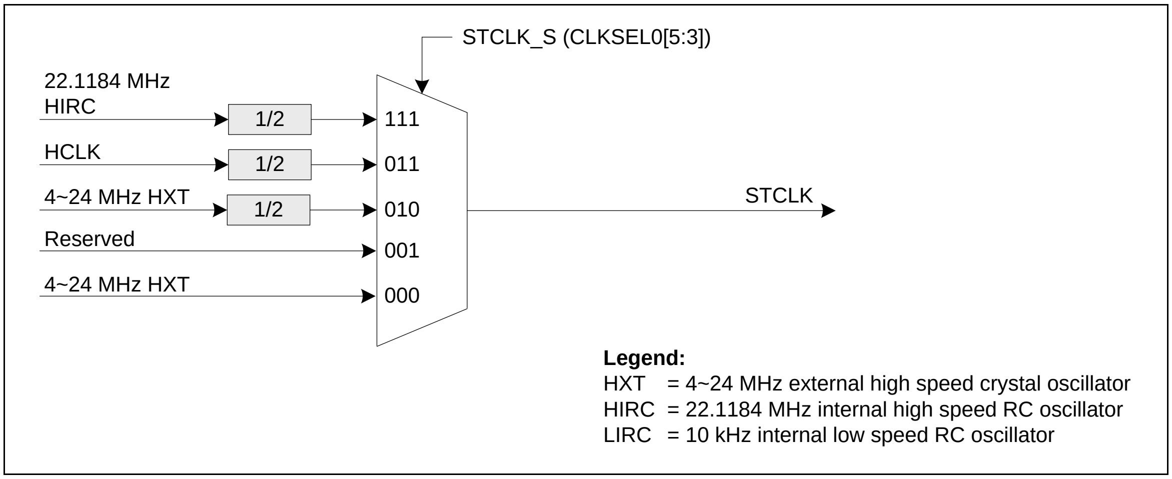

STCLK_S (CLKSEL0[5:3])
22.1184 MHz
HIRC
HCLK
4~24 MHz HXT
Reserved
4~24 MHz HXT
1/2
1/2
1/2
111
011
010
001
000
STCLK
Legend:
HXT = 4~24 MHz external high speed crystal oscillator
HIRC = 22.1184 MHz internal high speed RC oscillator
LIRC = 10 kHz internal low speed RC oscillator

Figure 6.3-5 SysTick clock Control Block Diagram

6.3.3 Power-down Mode Clock

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

The clocks still kept active are listed below:

- Clock Generator
- 10 kHz internal low speed oscillator clock
- Peripherals Clock (when 10 kHz low speed oscillator is adopted as clock source)

6.3.4 Frequency Divider Output

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

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

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

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

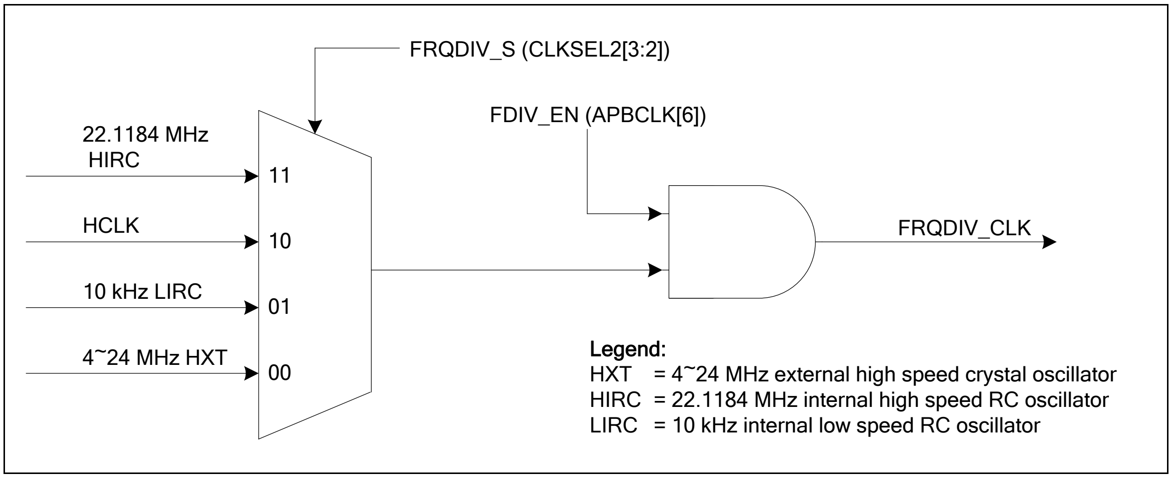

FRQDIV_S (CLKSEL2[3:2])
FDIV_EN (APBCLK[6])
22.1184 MHz HIRC
HCLK
10 kHz LIRC
4~24 MHz HXT
11
10
01
00
FRQDIV_CLK
Legend:
HXT = 4~24 MHz external high speed crystal oscillator
HIRC = 22.1184 MHz internal high speed RC oscillator
LIRC = 10 kHz internal low speed RC oscillator

Figure 6.3-6 Clock Source of Frequency Divider

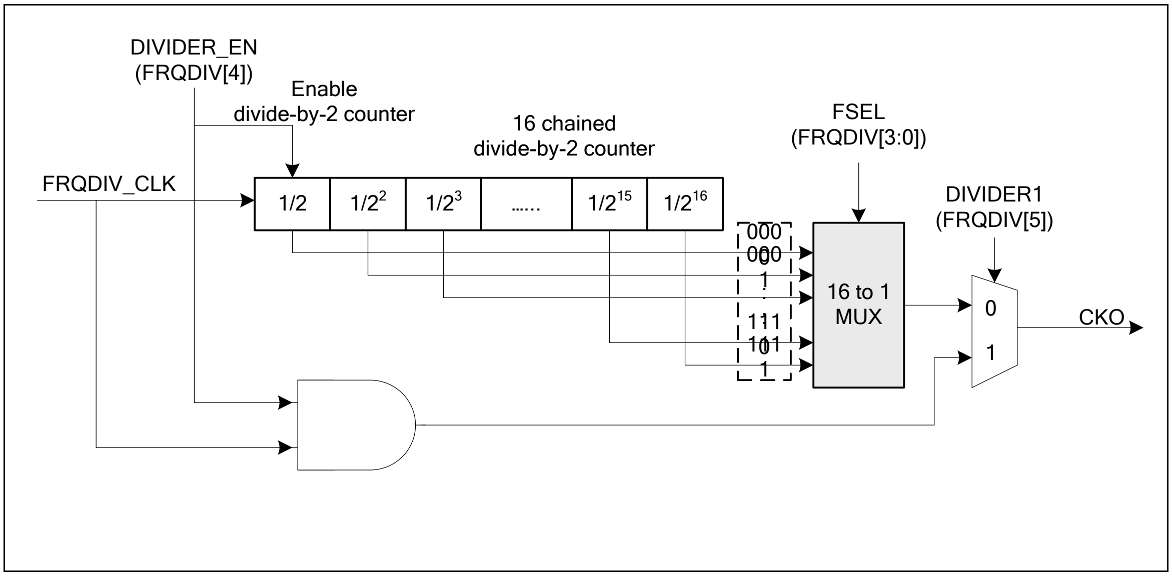


Figure 6.3-7 Block Diagram of Frequency Divider

6.3.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
CLK Base Address: **CLK_BA = 0x5000_0200**				
PWRCON	CLK_BA+0x00	R/W	System Power-down Control Register	0x0000_001X
AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_000D
APBCLK	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register	0x0000_000X
CLKSTATUS	CLK_BA+0x0C	R/W	Clock status monitor Register	0x0000_00XX
CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0000_003X
CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0xFFFF_FFFF
CLKDIV	CLK_BA+0x18	R/W	Clock Divider Number Register	0x0000_0000
CLKSEL2	CLK_BA+0x1C	R/W	Clock Source Select Control Register 2	0x0000_00FF
PLLCON	CLK_BA+0x20	R/W	PLL Control Register	0x0005_C22E
FRQDIV	CLK_BA+0x24	R/W	Frequency Divider Control Register	0x0000_0000

6.3.6 Register Description

Power-down Control Register (PWRCON)

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

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

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**PD_WAIT_CPU**	**Power-down Entry Condition Control (Write Protect)** 0 = Chip enters Power-down mode when the PWR_DOWN_EN bit is set to 1. 1 = Chip enters Power- down mode when the both PD_WAIT_CPU and PWR_DOWN_EN bits are set to 1 and CPU run WFI instruction.
[7]	**PWR_DOWN_EN**	**System Power-down Enable Bit (Write Protect)** When this bit is set to 1, Power-down mode is enabled and chip Power-down behavior will depend 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 CPU run WFI instruction. When chip wakes up from Power-down mode, this bit is cleared by hardware. User needs to set this bit again for next Power-down. In Power-down mode, 4~24 MHz external high speed crystal oscillator (HXT) and the 22.1184 MHz internal high speed oscillator (HIRC) will be disabled in this mode, and 10 kHz internal low speed RC oscillator (LIRC) are not controlled by Power-down mode. In Power- down mode, the PLL and system clock are disabled, and ignored the clock source selection. The clocks of peripheral are not controlled by Power-down mode, if the peripheral clock source is from 10 kHz internal low speed oscillator. 0 = Chip operating normally or chip in Idle mode because of WFI command.

		1 = Chip enters Power-down mode instantly or waits CPU sleep command WFI.
[6]	**PD_WU_STS**	**Power-down Mode Wake-up Interrupt Status** Set by “Power-down wake-up event”, which indicates that resume from Power-down mode” The flag is set if the GPIO, UART, WDT or BOD wake-up occurred. **Note:** This bit works only if PD_WU_INT_EN (PWRCON[5]) set to 1. Write 1 to clear the bit to 0.
[5]	**PD_WU_INT_EN**	**Power-down Mode Wake-up Interrupt Enable Control (Write Protect)** 0 = Disabled. 1 = Enabled. **Note:** The interrupt will occur when both PD_WU_STS and PD_WU_INT_EN are high.
[4]	**PD_WU_DLY**	**Wake-up Delay Counter Enable Control (Write Protect)** When the chip wakes up from Power-down mode, the clock control will delay certain clock cycles to wait system clock stable. The delayed clock cycle is 4096 clock cycles when chip works at 4~24 MHz external high speed crystal (HXT), and 256 clock cycles when chip works at 22.1184 MHz internal high speed RC oscillator (HIRC). 0 = Clock cycles delay Disabled. 1 = Clock cycles delay Enabled.
[3]	**OSC10K_EN**	**10 kHz Internal Low Speed RC Oscillator (LIRC) Enable Control (Write Protect)** 0 = 10 kHz internal low speed RC oscillator (LIRC) Disabled. 1 = 10 kHz internal low speed RC oscillator (LIRC) Enabled.
[2]	**OSC22M_EN**	**22.1184 MHz Internal High Speed RC Oscillator (HIRC) Enable Control (Write Protect)** 0 = 22.1184 MHz internal high speed RC oscillator (HIRC) Disabled. 1 = 22.1184 MHz internal high speed RC oscillator (HIRC) Enabled.
[1]	**Reserved**	Reserved.
[0]	**XTL12M_EN**	**4~24 MHz External High Speed Crystal (HXT) Enable Control (Write Protect)** The bit default value is set by flash controller user configuration register config0 [26:24]. When the default clock source is from 4~24 MHz external high speed crystal, this bit is set to 1 automatically. 0 = 4 ~ 24 MHz external high speed crystal oscillator (HXT) Disabled. 1 = 4 ~ 24 MHz external high speed crystal oscillator (HXT) Enabled.

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

Table 6.3-1 Power-down Mode Control

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

AHB Devices Clock Enable Control Register (AHBCLK)

The bits in this register are used to enable/disable clock for system clock.

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ISP_EN**	**Flash ISP Controller Clock Enable Control** 0 = Flash ISP engine clock Disabled. 1 = Flash ISP engine clock Enabled.
[1:0]	**Reserved**	Reserved.

APB Devices Clock Enable Control Register (APBCLK)

The bits in this register are used to enable/disable clock for peripheral controller clocks.

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

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="3">Reserved</td><td>ADC_EN</td><td colspan="4">Reserved</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="2">Reserved</td><td>PWM23_EN</td><td>PWM01_EN</td><td colspan="3">Reserved</td><td>UART0_EN</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="3">Reserved</td><td>SPI0_EN</td><td colspan="2">Reserved</td><td>I2C1_EN</td><td>I2C0_EN</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>Reserved</td><td>FDIV_EN</td><td>TMR3_EN</td><td>TMR2_EN</td><td>TMR1_EN</td><td>TMR0_EN</td><td>Reserved</td><td>WDT_EN</td></tr>
</table>

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**ADC_EN**	**Analog-Digital-Converter (ADC) Clock Enable Control** 0 = ADC peripheral clock Disabled. 1 = ADC peripheral clock Enabled.
[27:23]	**Reserved**	Reserved.
[21]	**PWM23_EN**	**PWM_23 Clock Enable Control** 0 = PWM23 clock Disabled. 1 = PWM23 clock Enabled.
[20]	**PWM01_EN**	**PWM_01 Clock Enable Control** 0 = PWM01 clock Disabled. 1 = PWM01 clock Enabled.
[19:17]	**Reserved**	Reserved.
[16]	**UART0_EN**	**UART0 Clock Enable Control** 0 = UART0 clock Disabled. 1 = UART0 clock Enabled.
[15:13]	**Reserved**	Reserved.
[12]	**SPI0_EN**	**SPI0 Peripheral Clock Enable Control** 0 = SPI0 peripheral clock Disabled. 1 = SPI0 peripheral clock Enabled.

[11:10]	**Reserved**	Reserved.
[9]	**I2C1_EN**	**I^2C1 Clock Enable Control** 0 = I^2C clock Disabled. 1 = I^2C clock Enabled.
[8]	**I2C0_EN**	**I^2C0 Clock Enable Control** 0 = I^2C clock Disabled. 1 = I^2C clock Enabled.
[7]	**Reserved**	Reserved.
[6]	**FDIV_EN**	**Frequency Divider Output Clock Enable Control** 0 = FDIV clock Disabled. 1 = FDIV clock Enabled.
[5]	**TMR3_EN**	**Timer3 Clock Enable Control** 0 = Timer3 clock Disabled. 1 = Timer3 clock Enabled.
[4]	**TMR2_EN**	**Timer2 Clock Enable Control** 0 = Timer2 clock Disabled. 1 = Timer2 clock Enabled.
[3]	**TMR1_EN**	**Timer1 Clock Enable Control** 0 = Timer1 clock Disabled. 1 = Timer1 clock Enabled.
[2]	**TMR0_EN**	**Timer0 Clock Enable Control** 0 = Timer0 clock Disabled. 1 = Timer0 clock Enabled.
[1]	**Reserved**	Reserved.
[0]	**WDT_EN**	**Watchdog Timer Clock Enable Control (Write Protect)** 0 = Watchdog Timer clock Disabled. 1 = Watchdog Timer clock Enabled. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Clock Status Register (CLKSTATUS)

The bits in 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+0x0C	R/W	Clock status monitor Register	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLK_SW_FAIL	**Reserved**		**OSC22M_STB**	**OSC10K_STB**	**PLL_STB**	**Reserved**	**XTL12M_STB**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**CLK_SW_FAIL**	**Clock Switching Fail Flag** 0 = Clock switching success. 1 = Clock switching failed. **Note1:** This bit is updated when software switches system clock source. If switch target clock is stable, this bit will be set to 0. If switch target clock is not stable, this bit will be set to 1. **Note2:** This bit is read only. After selected clock source is stable, hardware will switch system clock to selected clock automatically, and CLK_SE_FAIL will be cleared automatically by hardware.
[6:5]	**Reserved**	Reserved.
[4]	**OSC22M_STB**	**22.1184 MHz Internal High Speed RC Oscillator (HIRC) Stable Flag (Read Only)** 0 = 22.1184 MHz internal high speed RC oscillator (HIRC) clock is not stable or disabled. 1 = 22.1184 MHz internal high speed RC oscillator (HIRC) clock is stable.
[3]	**OSC10K_STB**	**10 kHz Internal Low Speed RC Oscillator (LIRC) Stable Flag (Read Only)** 0 = 10 kHz internal low speed RC oscillator (LIRC) clock is not stable or disabled. 1 = 10 kHz internal low speed RC oscillator (LIRC) clock is stable.
[2]	**PLL_STB**	**Internal PLL Clock Source Stable Flag (Read Only)** 0 = Internal PLL clock is not stable or disabled.

		1 = Internal PLL clock is stable.
[1]	**Reserved**	Reserved.
[0]	**XTL12M_STB**	**4~24 MHz External High Speed Crystal (HXT) Stable Flag (Read Only)** 0 = 4~24 MHz external high speed crystal (HXT) clock is not stable or disabled. 1 = 4~24 MHz external high speed crystal (HXT) clock is stable.

Clock Source Select Control Register 0 (CLKSEL0)

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

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:3]	**STCLK_S**	**Cortex®-M0 SysTick Clock Source Selection from Reference Clock (Write Protect)** If SYST_CSR[2] = 1, SysTick clock source is from HCLK. If SYST_CSR[2] = 0, SysTick clock source is defined by below settings. 000 = Clock source is from HXT. 001 = Reserved. 010 = Clock source is from HXT/2 011 = Clock source is from HCLK/2. 111 = Clock source is from HIRC/2. Others = Reserved. **Note1:** These bits are protected bits, and programming them needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100. **Note2:** if the SysTick clock source is not from HCLK (i.e. SYST_CSR[2] = 0), SysTick clock source must less than or equal to HCLK/2.
[2:0]	**HCLK_S**	**HCLK Clock Source Selection (Write Protect)** 000 = Clock source is from HXT. 001 = Reserved. 010 = Clock source is from PLL. 011 = Clock source is from LIRC. 111 = Clock source is from HIRC. Others = Reserved.

		Note1: Before clock switching, the related clock sources (both pre-select and new-select) must be turn-on and stable. **Note2:** 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. **Note3:** These bits are protected bit, and programming them needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Clock Source Select Control Register 1 (CLKSEL1)

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

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

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

Bits	Description	
[31:30]	**PWM23_S**	**PWM2 and PWM3 Clock Source Selection** PWM2 and PWM3 use the same Engine clock source; both of them use the same prescaler. 00 = Clock source is from HXT. 01 = Clock source is from LIRC. 10 = Clock source is from HCLK. 11 = Clock source is from HIRC.
[29:28]	**PWM01_S**	**PWM0 and PWM1 Clock Source Selection** PWM0 and PWM1 use the same Engine clock source; both of them use the same prescaler. 00 = Clock source is from HXT. 01 = Clock source is from LIRC. 10 = Clock source is from HCLK. 11 = Clock source is from HIRC.
[27:26]	**Reserved**	Reserved.
[25:24]	**UART_S**	**UART Clock Source Selection** 00 = Clock source is from HXT. 01 = Clock source is from PLL. 10 = Reserved.

		11 = Clock source is from HIRC.
[23]	**Reserved**	Reserved.
[22:20]	**TMR3_S**	**TIMER3 Clock Source Selection** 000 = Clock source is from HXT. 010 = Clock source is from HCLK. 011 = Clock source is from external trigger T3. 101 = Clock source is from LIRC. 111 = Clock source is from HIRC. Others = Reserved.
[19]	**Reserved**	Reserved.
[18:16]	**TMR2_S**	**TIMER2 Clock Source Selection** 000 = Clock source is from HXT. 010 = Clock source is from HCLK. 011 = Clock source is from external trigger T2. 101 = Clock source is from LIRC. 111 = Clock source is from HIRC. Others = Reserved.
[15]	**Reserved**	Reserved.
[14:12]	**TMR1_S**	**TIMER1 Clock Source Selection** 000 = Clock source is from HXT. 010 = Clock source is from HCLK. 011 = Clock source is from external trigger T1. 101 = Clock source is from LIRC. 111 = Clock source is from HIRC. Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**TMR0_S**	**TIMER0 Clock Source Selection** 000 = Clock source is from HXT. 010 = Clock source is from HCLK. 011 = Clock source is from external trigger T0. 101 = Clock source is from LIRC. 111 = Clock source is from HIRC. Others = Reserved.
[7:5]	**Reserved**	Reserved.
[4]	**SPI0_S**	**SPI0 clock Source Selection** 0 = Clock source is from PLL. 1 = Clock source is from HCLK.

[3:2]	**ADC_S**	**ADC Peripheral Clock Source Selection** 00 = Clock source is from HXT. 01 = Clock source is from PLL. 10 = Clock source is from HCLK. 11 = Clock source is from HIRC.
[1:0]	**WDT_S**	**Watchdog Timer Clock Source Selection (Write Protect)** 00 = Reserved. 01 = Reserved. 10 = Clock source is from HCLK/2048 clock. 11 = Clock source is from LIRC. **Note:** These bits are protected bits, and programming them needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.

Clock Source Select Control Register (CLKSEL2)

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

Register	Offset	R/W	Description	Reset Value
CLKSEL2	CLK_BA+0x1C	R/W	Clock Source Select Control Register 2	0x0000_00FF

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**WWDT_S**	**Window Watchdog Timer Clock Source Selection** 10 = Clock source is from HCLK/2048 clock. 11 = Clock source is from LIRC.
[15:4]	**Reserved**	Reserved.
[3:2]	**FRQDIV_S**	**Clock Divider Clock Source Selection** 00 = Clock source is from HXT. 01 = Reserved 10 = Clock source is from HCLK. 11 = Clock source is from HIRC.
[1:0]	**Reserved**	Reserved.

Clock Divider Register (CLKDIV)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**ADC_N**	**ADC peripheral Clock Divide Number from ADC peripheral Clock Source** ADC peripheral clock frequency = (ADC peripheral clock source frequency) / (ADC_N + 1).
[15:12]	**Reserved**	Reserved.
[11:8]	**UART_N**	**UART Clock Divide Number from UART Clock Source** UART clock frequency = (UART clock source frequency) / (UART_N + 1).
[7:4]	**Reserved**	Reserved.
[3:0]	**HCLK_N**	**HCLK Clock Divide Number from HCLK Clock Source** HCLK clock frequency = (HCLK clock source frequency) / (HCLK_N + 1).

PLL Control Register (PLLCON)

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

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

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

Bits	Description	
[19]	**PLL_SRC**	**PLL Source Clock Selection** 0 = PLL source clock from HXT. 1 = PLL source clock from HIRC.
[18]	**OE**	**PLL OE (FOUT Enable) Pin Control** 0 = PLL FOUT Enabled. 1 = PLL FOUT is fixed low.
[17]	**BP**	**PLL Bypass Control** 0 = PLL is in Normal mode (default). 1 = PLL clock output is same as PLL source clock input.
[16]	**PD**	**Power-down Mode** If the PWR_DOWN_EN bit is set to 1 in PWRCON register, the PLL will enter Power-down mode too. 0 = PLL is in Normal mode. 1 = PLL is in Power-down mode (default).
[15:14]	**OUT_DV**	**PLL Output Divider Control** Refer to the formulas below the table.
[13:9]	**IN_DV**	**PLL Input Divider Control**

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

PLL Output Clock Frequency Setting:

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

Constrain:

1. $4MHz < FIN < 24MHz$

2. $800KHz < \frac{FIN}{2 * NR} < 7.5MHz$

3. $100MHz < FCO = FIN \times \frac{NF}{NR} < 200MHz$

   $120MHz < FCO \text{ is preferred}$

Symbol	**Description**
FOUT	Output Clock Frequency
FIN	Input (Reference) Clock Frequency
NR	Input Divider (IN_DV + 2)
NF	Feedback Divider (FB_DV + 2)

NO	OUT_DV = “00” : NO = 1
	OUT_DV = “01” : NO = 2
	OUT_DV = “10” : NO = 2
	OUT_DV = “11” : NO = 4

Default Frequency Setting

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

Frequency Divider Control Register (FRQDIV)

Register	Offset	R/W	Description	Reset Value
FRQDIV	CLK_BA+0x24	R/W	Frequency Divider Control Register	0x0000_0000

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

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

6.4 Flash Memory Controller (FMC)

6.4.1 Overview

The M058S Series are equipped with 64/32/16/8 KB on chip embedded Flash memory for application program (APROM) that can be updated through ISP registers. In-System-Programming (ISP) and In-Application-Programming (IAP) enable 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, it also provides additional 4 KB DATA Flash for user to store some application depended data before chip power off in 64/32/16/8 KB APROM model.

It provides more settings in CONFIG0 to support more advanced functions, including power-on with tri-state I/O, default to enable WDT after booting, enable WDT in Power-down mode, and IAP functions. The following table shows the added functions of M058S Series.

	M058S Series
CONFIG[6]	To support IAP function and Multi-Boot function
CONFIG[10]	Select I/O state after booting
CONFIG[30]	To support WDT in Power-Down mode when WDT is default on after booting
CONFIG[31]	To support WDT default on after booting

Table 6.4-1 M058S Series Function Difference List (FMC)

6.4.2 Features

- Runs up to 50 MHz with zero wait state for continuous address read access
- 32 KB application program memory (APROM)
- 4 KB in system programming (ISP) loader program memory (LDROM)
- Fixed 4 KB Data Flash
- All embedded flash memory supports 512 bytes page erase
- In System Program (ISP)/In Application Program (IAP) to update on chip Flash EPROM

6.4.3 Block Diagram

The flash memory controller consist of AHB slave interface, ISP control logic, writer interface and flash macro interface timing control logic. The block diagram of flash memory controller is shown in the following figure.

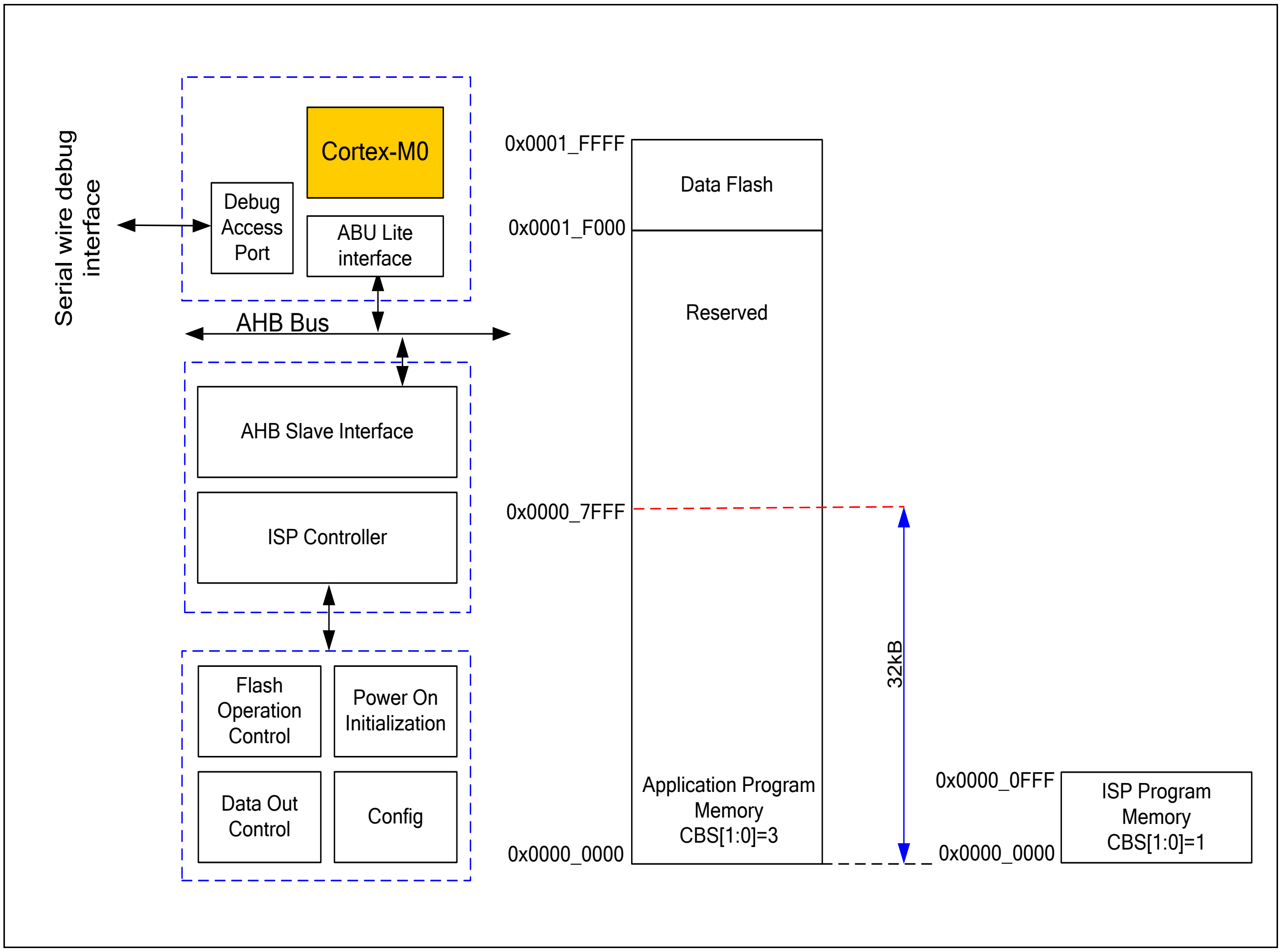


Figure 6.4-1 Flash Memory Control Block Diagram

6.4.4 FMC Memory Organization

The M058S Series flash memory consists of program memory (APROM), Data Flash, ISP loader program memory (LDROM), and user configuration.

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

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

Data Flash is used for user to store data. It can be read by ISP read or memory read and programmed through ISP register. The size of each erase unit is 512 bytes. Data Flash size is 4 KB and start address is fixed at 0x0001_F000.

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

In the NuMicro™ Family, the flash memory organization is different to system memory map. Flash memory organization is used when user using ISP command to read, program or erase flash memory. System memory map is used when CPU access flash memory to fetch code or data. For example, When system is set to boot from LDROM by CBS[1:0] = 1, CPU will be able to fetch code of LDROM from 0x0 ~ 0xFFF. However, if user want to read LDROM by ISP, they still need to read the address of LDROM as 0x0010_0000 ~ 0x0010_0FFF.

The following table and figure show the address mapping information of APROM, LDROM, Data Flash and user configuration.

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

Table 6-18 Flash Memory Address Map

The Flash memory organization is shown below:

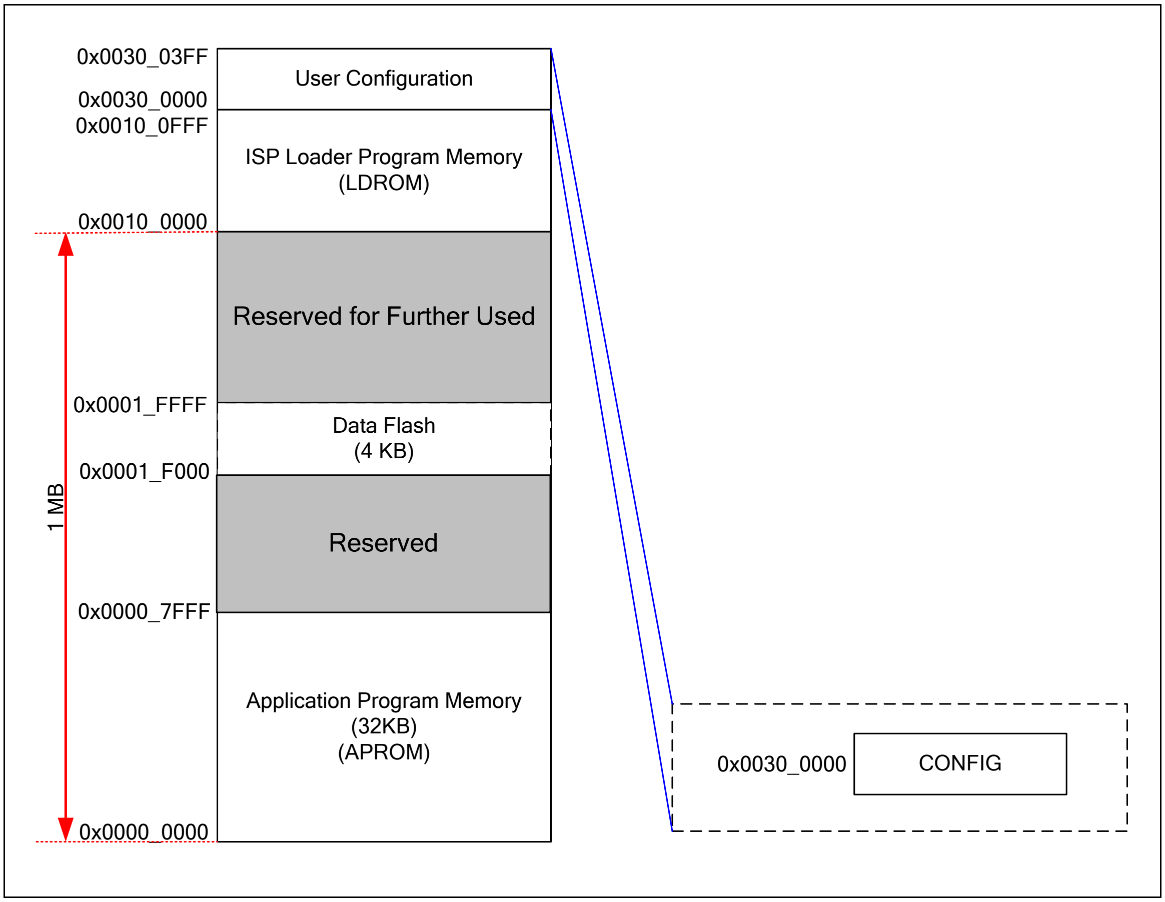


Figure 6.4-2 Flash Memory Organization

6.4.5 Data Flash

The M058S Series provides Data Flash for user to store data. It is read/write thru ISP registers. 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 8/16/32/64 KB flash memory device, Data Flash size is 4 KB and start address is fixed at 0x0001_F000.

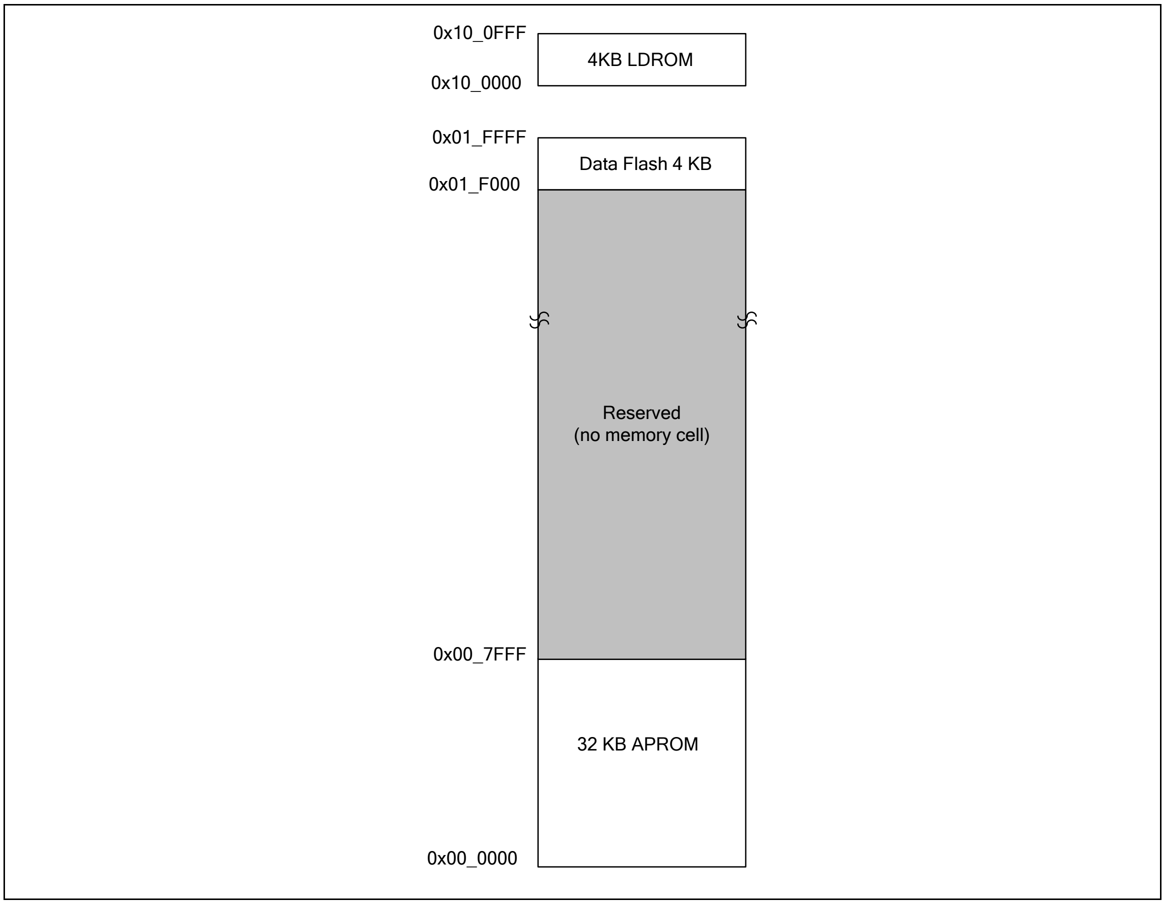


Figure 6.4-3 Flash Memory Structure

6.4.6 User Configuration

6.4.6.1 Boot options

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

CONFIG0 (Address = 0x0030_0000)

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

<table>
<tr><th>BIT</th><th colspan="2">Description</th></tr>
<tr><td>[31]</td><td>CWDTEN</td><td><b>Watchdog Enable Control</b>
0 = Watchdog Timer Enabled and force Watchdog Timer clock source as OSC10K after chip powered on.
1 = Watchdog Timer Disabled after chip powered on.</td></tr>
<tr><td>[30]</td><td>CWDTPDEN</td><td><b>Watchdog Clock Power-Down Enable Control</b>
0 = OSC10K Watchdog Timer clock source is forced to be always enabled.
1 = OSC10K Watchdog Timer clock source is controlled by OSC10K_EN (PWRCON[3]) when chip enters Power-Down mode.
<b>Note:</b> This bit only works at CWDTEN is set to 0</td></tr>
<tr><td>[31:28]</td><td>Reserved</td><td>Reserved<b>.</b></td></tr>
<tr><td>[27]</td><td>CFGXT1</td><td><b>P7_MFP [1:0] Multi-Function Select</b>
0 = P7_MFP[1:0] pins are configured as GPIO pins.
1 = P7_MFP[1:0] pins are configured as external 4~24MHz external high speed crystal oscillator (HXT) pins.</td></tr>
<tr><td>[26:24]</td><td>CFOSC</td><td><b>CPU Clock Source Selection After Reset</b>
<table>
<tr><th>CFOSC[2:0]</th><th>Clock Source</th></tr>
<tr><td>000</td><td>External crystal clock (4 ~ 24 MHz)</td></tr>
<tr><td>111</td><td>Internal RC 22.1184 MHz oscillator clock</td></tr>
<tr><td>Others</td><td>Reserved</td></tr>
</table>
The value of CFOSC will be load to CLKSEL0.HCLK_S[2:0] in system register after any reset occurs.</td></tr>
<tr><td>[23]</td><td>CBODEN</td><td><b>Brown-out Detector Enable Control</b>
0= Brown-out detect Enabled after powered on.</td></tr>
</table>

		1= Brown-out detect Disabled after powered on.
[22:21]	**CBOV**	**Brown-out Voltage Selection** **CBOV** / **Brown-out Voltage** 11 / 4.4V 10 / 3.7V 01 / 2.7V 00 / 2.2V
[20]	**CBORST**	**Brown-out Reset Enable Control** 0 = Brown-out reset Enabled after powered on. 1 = Brown-out reset Disabled after powered on.
[19:11]	**Reserved**	Reserved.
[10]	**CIOINI**	**I/O Initial State Selection** 0 = All GPIO default to be input tri-state mode after powered on. 1 = All GPIO default to be Quasi-bidirectional mode after chip is powered on.
[9:8]	**Reserved**	Reserved.
[7:6]	**CBS**	**Chip Boot Selection** 00 = LDROM with IAP function. 01 = LDROM without IAP function. 10 = APROM with IAP function. 11 = APROM without IAP function. User can set CBS[0] = 0 to support IAP function supported. When CBS[0] = 0, the LDROM is mapping to address 0x100000 and APROM is mapping to address 0x0. User could access them by their address without boot switching. In other words, if IAP function is supported, the code in LDROM and APROM can be called by each other. **Note1:** BS bit of ISPCON is only can be used to control boot switching when CBS[0] = 1. **Note2:** VECMAP is only can be used to remap to 0x0~0x1ff when CBS[0] = 0.
[5:2]	**Reserved**	Reserved.
[1]	**LOCK**	**Security Lock Control** 0 = Flash data is locked 1 = Flash data is not locked. When flash data is locked, only device ID, unique ID, user configuration can be read by writer and ICP through serial debug interface. Others data is locked as 0xFFFFFFFF. ISP can read data anywhere regardless of LOCK bit value. To unlock system, user can use ISP command to disable LOCK bit or erase whole chip (Chip Erase) by ICP tool.
[0]	**Reserved**	Reserved.

Note: The reserved bits of user configuration should be kept as ‘1’.

6.4.6.2 Brown-out Detection

The M058S Series includes brown-out detection function for monitoring the voltage on V_{DD} pin. If V_{DD} voltage falls below level setting of CBOV, the BOD event will be triggered when BOD enabled. User can decide to use BOD reset by enable CBORST or just enable BOD interrupt by NVIC when BOD detected. Because BOD reset is issued whenever V_{DD} voltage falls below the level setting of CBOV, user must make sure the CBOV setting to avoid BOD reset right after BOD reset enabled. For example, if the V_{DD} is 3.3V, CBOV could only be 00'b or 01'b. Otherwise, the system will be halted in BOD reset state when BOD reset is enabled and CBOV is 10'b or 11'b.

6.4.7 Boot Selection

The M058S Series provides in system programming (ISP) feature to support to update program memory when chip is mounted on PCB. A dedicated 4 KB program memory is used to store ISP firmware. Users can select to start program fetch from APROM or LDROM by (CBS) in CONFIG0.

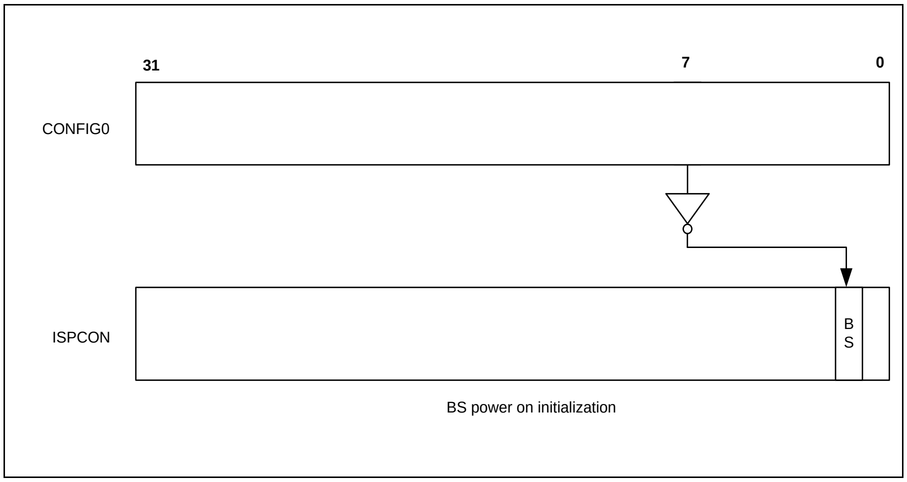


Figure 6.4-4 Boot Select (BS) for Power-on Action

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

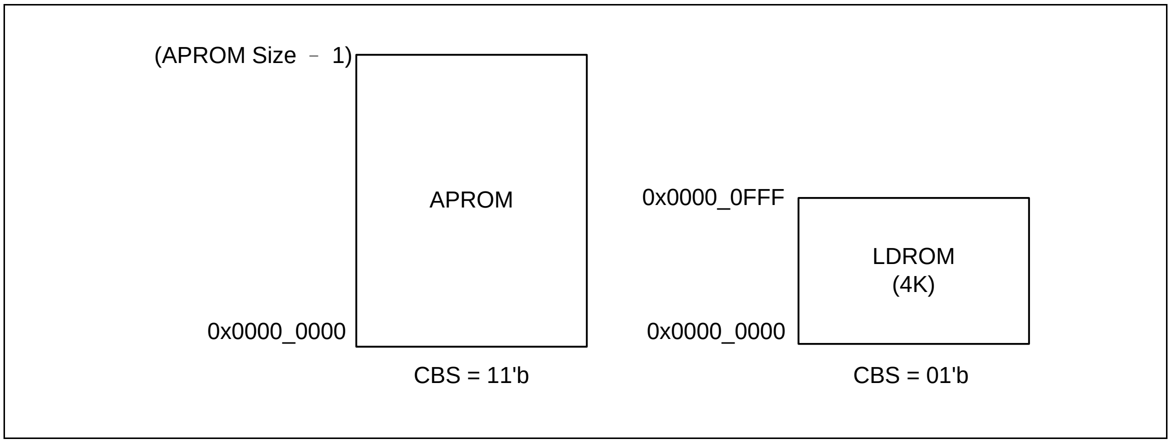


Figure 6.4-5 Program Executing Range for Booting from APROM and LDROM

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

CBS[1:0]	Boot Selection
00	**LDROM with IAP function** Chip booting from LDROM, program executing range including LDROM and most of APROM (all except first 512 bytes as the first 512 bytes is mapped from LDROM). LDROM address is mapping to 0x0010_0000 ~ 0x0010_0FFF, and also the first 512 bytes of LDROM is mapping to the address 0x0000_0000 ~ 0x0000_01FF. The address 0x0000_0000 ~ 0x0000_01FF can be re-mapping to any other page within executing range though ISP command.
01	**LDROM without IAP function** Chip booting from LDROM, program executing range only including LDROM; APROM cannot be access by program directly, except by through ISP. LDROM is write-protected in this mode.
10	**APROM with IAP function** Chip booting from APROM, program executing range including LDROM and APROM LDROM address is mapping to 0x0010_0000~0x0010_0FFF The address 0x0000_0000 ~ 0x0000_01FF can be re-mapping to any other page within executing range though ISP command.
11	**APROM without IAP function** Chip booting from APROM and program executing range only including APROM. LDROM cannot be access by program directly, except by through ISP. APROM is write-protected in this mode.

Table 6.4-2 Supported Boot Selection Options

6.4.8 In-Application-Programming (IAP)

The M058S Series provides In-application-programming (IAP) function for user to switch the code executing between APROM and LDROM without reset. User can enable the IAP function by re-booting chip and setting the chip boot selection bits in CONFIG0 (CBS[1:0]) as 2 or 0.

In the case that the chip boots from APROM with the IAP function enabled (CBS[1:0] = 2), the executable range of code includes all of APROM and LDROM. The address space of APROM is kept as the original size but the address space of the 4 KB LDROM is mapped to 0x0010_0000~ 0x0010_0FFF.

In the case that the chip boots from LDROM with the IAP function enabled (CBS[1:0] = 0), the executable range of code includes all of LDROM and almost all of APROM except for its first page. User cannot access the first page of APROM because the first page of executable code range becomes the mirror of the first page of LDROM as set by default. Meanwhile, the address space of 4 KB LDROM is mapped to 0x0010_0000~0x0010_0FFF.

Please refer to following figure for the address map while IAP is activating.

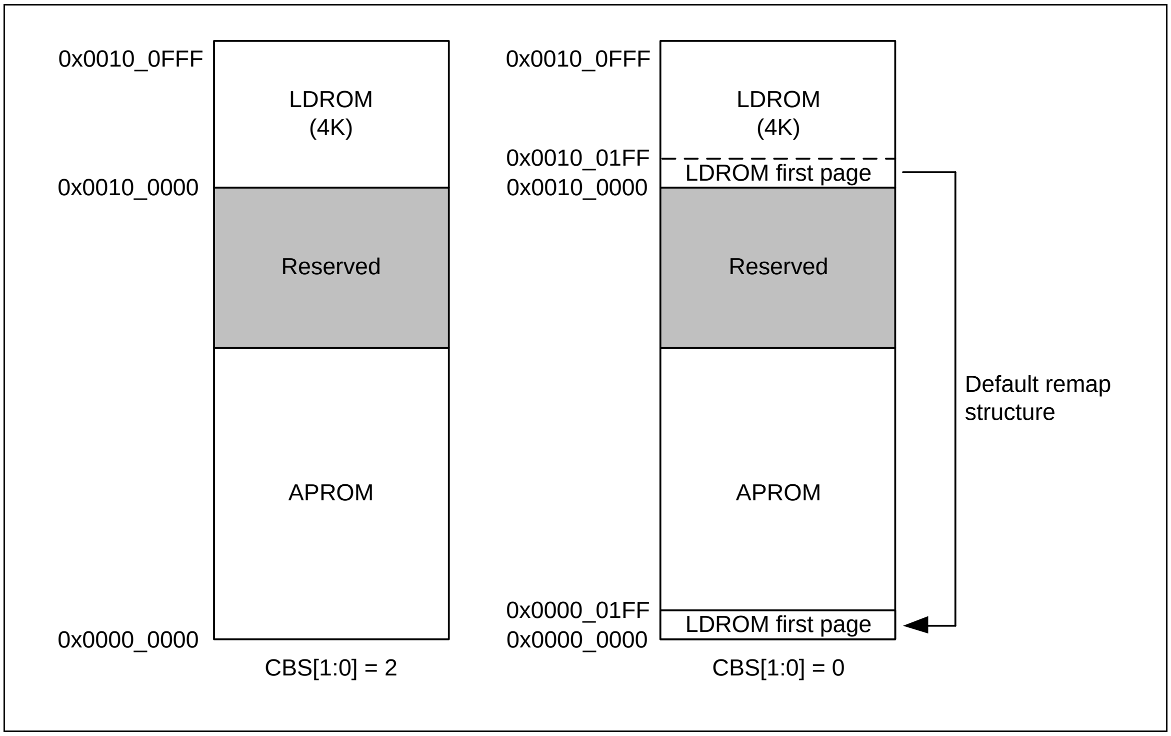


Figure 6.4-6 Executable Range of Code with IAP Function Enabled

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

6.4.9 In-System-Programming (ISP)

The M058S Series supports In-System-Programming which allows a device to be reprogrammed under software control and avoids system fail risk when download or programming fail. Furthermore, the capability to update the application firmware makes a wide range of applications possible.

To supports In-System-Programming, M058S Series includes LDROM and ISP controller. User can implement their ISP loader programming in LDROM and this loader can programming user application code (APROM) through ISP register. In other words, the loader could provide the ability to update system firmware on board. By ISP loader, various hardware peripheral interfaces make it be easier to receive new program code. The most common method to perform ISP is via UART along with the ISP loader in LDROM. General speaking, PC transfers the new APROM code through serial port. Then ISP loader receives it and re-programs into APROM through ISP commands.

6.4.10 ISP Registers Control Procedure

The M058S Series supports booting from APROM or LDROM initially defined by user configuration. The change of user configuration needs to reboot system to make it take effect. If user wants to switch between APROM or LDROM mode without changing user configuration with CBS[0] = 1, he needs to control BS bit of ISPCON control register, then reset CPU by IPRSTC1 control register. The boot switching flow by BS bit is shown in the following figure. Boot switching function by BS bit is only valid when CBS[0] = 1.

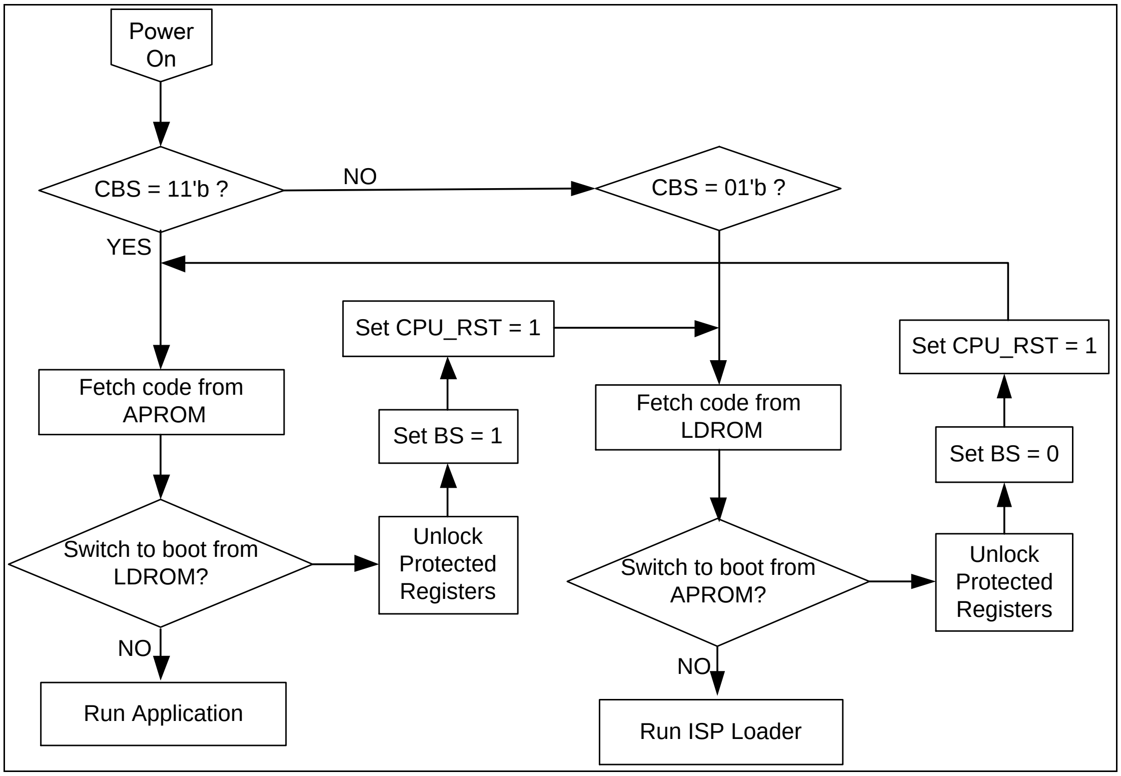


Figure 6.4-7 Example Flow of Boot Selection by BS Bit when CBS[0] = 1

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

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

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

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

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

Several error conditions are checked after ISP register function is completed. If an error condition occurs, ISP register operation is not started and the ISP fail flag will be set instead. ISPFF flag can only be cleared by software. The next ISP register control procedure can be started even ISPFF bit is kept as 1. Therefore, it is recommended to check the ISPFF bit and clear it after each ISP register operation if it is set to 1.

When the ISPGO bit is set, CPU will wait for ISP operation to finish during this period; the peripheral still keeps working as usual. If any interrupt request occurs, CPU will not service it till ISP operation is finished. When ISP operation is finished, the ISPGO bit will be cleared by hardware automatically. User can check whether ISP operation is finished or not by the ISPGO bit. User should add ISB instruction next to the instruction in which the ISPGO bit is set 1 to ensure correct execution of the instructions following ISP operation.

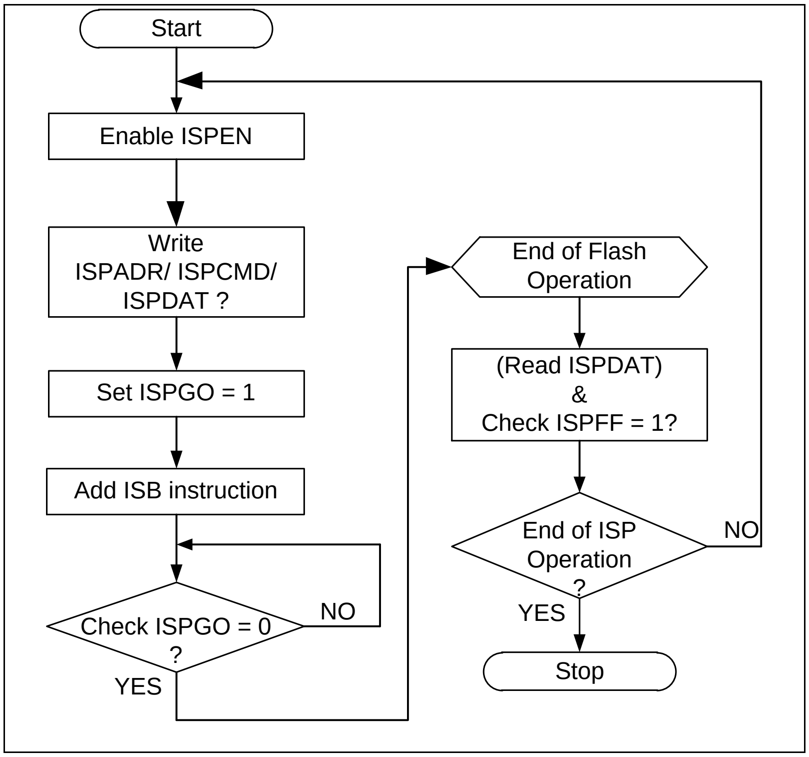


Figure 6.4-8 ISP Flow Example

ISP Command	ISPCMD	ISPADR	ISPDAT
FLASH Page Erase	0x22	Valid address of flash memory origination. It must be 512 bytes page alignment.	Don't care
FLASH Program	0x21	Valid address of flash memory origination	Programming Data
FLASH Read	0x00	Valid address of flash memory origination	Return Data
Read Unique ID	0x04	0x0000_0000	Unique ID Word 0
		0x0000_0004	Unique ID Word 1
		0x0000_0008	Unique ID Word 2
Read Company ID	0x0B	Don't care	Company ID (0xDA)
Vector Page Re-Map	0x2E	Page in APROM or LDROM It must be 512 bytes page alignment	Don't care

Table 6.4-3 ISP Command List

6.4.11 Multi-booting by Vector Remapping

The M058S Series can support to boot from different address by vector page remapping function. When CBS[0] = 0, All pages of LDROM and APROM can be remap to vector address 0x0. The remapping address can be got by VECMAP of ISPSTA register. When CBS[1:0] = 2, the remapping address is default to 0x0 when powered on. This means the vector page is mapping from first page of APROM to boot from APROM at power on. When CBS[1:0] = 0, the remapping address is 0x100000. This means the vector page is mapping from first page of LDROM to boot from LDROM at power on.

The remapping address can be changed by Vector Page Re-Map command. User may remap specified page to vector page by Vector Page Re-Map command, than using CPU_RST or SYSRESETREQ to reset system to reboot. The CPU will fetch the stack and reset handler pointer from new vector page, than boot to the specified application.

For example, if user has two independent applications in APROM called App0 and App1. App0 is located at 0x0, and App1 is located at 0x8000. The CBS[1:0] was set to 0 to boot from LDROM. When power on, the system will execute the code in LDROM. The code in LDROM will decide to boot to App0 or App1. For boot to App0, the code in LDROM will enable ISP and set vector page remapping to 0x0, than reset CPU by CPU_RST(Not reset I/O and peripherals) or SYSRESETREQ(Reset I/O and peripherals) to boot to App0. For boot to App1, the code in LDROM will enable ISP and set vector page remapping to 0x8000, than reset CPU by CPU_RST or SYSRESETREQ to boot to App1. The following figure shows how to use vector page remapping to boot to different applications.

To set vector page remapping, user needs to set new page address to ISPADR, set remap command code 0x2E to ISPCMD. Than trigger ISP command by set ISPGO to 1. User can confirm the new vector page mapping address by ISPSTA control register.

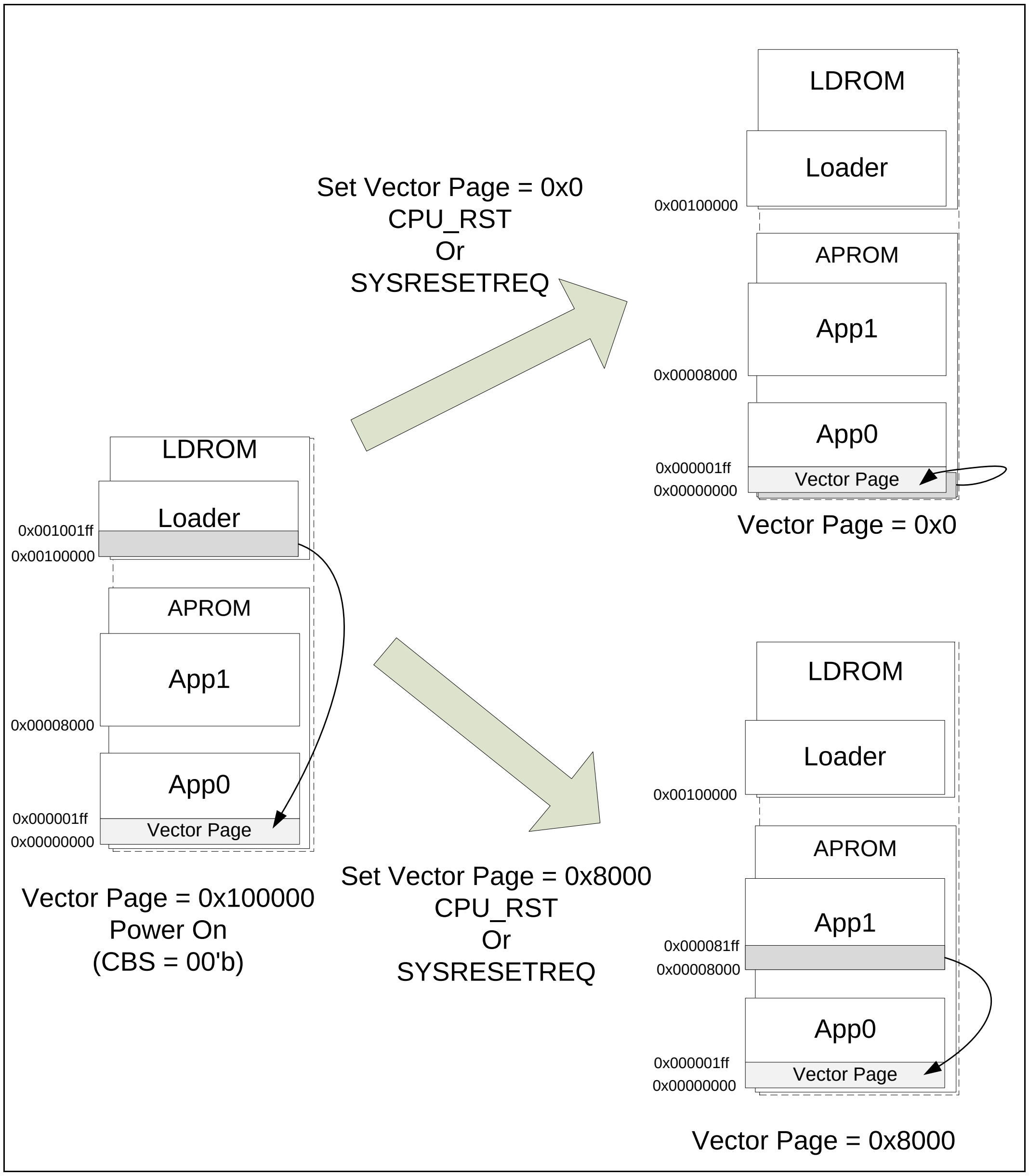


Figure 6.4-9 Multi-booting by Vector Page Remapping

6.4.12 Register Map

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

Register	Offset	R/W	Description	Reset Value
FMC Base Address: **FMC_BA = 0x5000_C000**				
ISPCON	FMC_BA+0x00	R/W	ISP Control Register	0x0000_0000
ISPADR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000
ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000
ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000
ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000
DFBADR	FMC_BA+0x14	R	Data Flash Base Address	0x0001_F000
FATCON	FMC_BA+0x18	R/W	Flash Access Time Control Register	0x0000_0000
ISPSTA	FMC_BA+0x40	R/W	ISP Status Register	0x0000_0000

6.4.13 Register Description

ISP Control Register (ISPCON)

Register	Offset	R/W	Description	Reset Value
ISPCON	FMC_BA+0x00	R/W	ISP Control Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is set by hardware when a triggered ISP meets any of the following conditions: (1) APROM writes to itself if APUEN is set to 0. (2) LDROM writes to itself. (3) Destination address is illegal, such as over an available range. **Note:** Write 1 to clear this bit to 0.
[5]	**LDUEN**	**LDROM Update Enable Control (Write Protect)** 0 = LDROM cannot be updated. 1 = LDROM can be updated when the MCU runs in APROM.
[4]	**CFGUEN**	**CONFIG Update Enable Control (Write Protect)** Writing this bit to 1 enables software to update CONFIG value by ISP register control procedure regardless of program code is running in APROM or LDROM. 0 = ISP update User Configuration Disabled. 1 = ISP update User Configuration Enabled.
[3]	**APUEN**	**APROM Update Enable Control (Write Protect)** 0 = APROM cannot be updated when chip runs in APROM. 1 = APROM can be updated when chip runs in APROM.
[2]	**Reserved**	Reserved.
[1]	**BS**	**Boot Select (Write Protect)** Set/clear this bit to select next booting from LDROM/APROM, respectively. This bit also functions as chip booting status flag, which can be used to check where chip booted

		from. This bit is initiated with the inversed value of CBS in CONFIG0 after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened. 0 = Boot from APROM. 1 = Boot from LDROM.
[0]	**ISPEN**	**ISP Enable Control (Write Protect)** Set this bit to enable ISP function. 0 = ISP function Disabled. 1 = ISP function Enabled.

ISP Address Register (ISPADR)

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

31	30	29	28	27	26	25	24
ISPADR[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	Description	
[31:0]	**ISPADR**	**ISP Address** The M058S Series has a maximum 16K x 32-bit (32 KB) of embedded Flash, which supports word program only. ISPADR[1:0] must be kept 00'b 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	Description	
[31:0]	**ISPDAT**	**ISP Data** Write data to this register before ISP program operation. Read data from this register after ISP read operation.

ISP Command (ISPCMD)

Register	Offset	R/W	Description	Reset Value
ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000

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

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

ISP Trigger Control Register (ISPTRG)

Register	Offset	R/W	Description	Reset Value
ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**ISPGO**	**ISP Start Trigger (Write Protect)** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. 0 = ISP operation is finished. 1 = ISP operation is progressed.

Data Flash Base Address Register (DFBADR)

Register	Offset	R/W	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	Description	
[31:0]	**DFBADR**	**Data Flash Base Address** This register indicates Data Flash start address. It is a read only register. For 32 KB flash memory device, the Data Flash size is 4 KB 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			**LFOM**	**Reserved**			

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**Reserved**	Always keep 0.
[6:5]	**Reserved**	Reserved.
[4]	**LFOM**	**Low Frequency Optimization Mode Control (Write Protect)** When chip operation frequency is lower than 25 MHz, chip can work more efficiently by setting this bit to 1 0 = Low frequency optimization mode Disabled. 1 = Low frequency optimization mode Enabled.
[0]	**Reserved**	Reserved.

ISP Status Register (ISPSTA)

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

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

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

6.5 General Purpose I/O (GPIO)

6.5.1 Overview

There are 58 General Purpose I/O pins shared with special feature functions in this MCU. The 58 pins are arranged in 8 ports named with P0, P1… to P7. Each port equips maximum 8 pins except P7[1:0]. Each one of the 58 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} which is from 5.0V to 2.5V.

6.5.2 Features

- Four I/O modes:
  - Input only with high impedance
  - Push-pull output
  - Open-drain output
- Quasi-bidirectional TTL/Schmitt trigger input mode selected by Px_MFP[23:16]
- I/O pin configured as interrupt source with edge/level setting
- I/O pin internal pull-up resistor enabled only in Quasi-bidirectional I/O mode
- Enabling the pin interrupt function will also enable the pin wake-up function
- Configurable default I/O mode of all pins after reset by CIOINI(CONFIG[10]) setting
  - CIOINI = 0, all GPIO pins in Input tri-state mode after chip reset
  - CIOINI = 1, all GPIO pins in Quasi-bidirectional mode after chip reset

6.5.3 Basic Configuration

The GPIO pin functions are configured in P0_MFP, P1_MFP, P2_MFP, P3_MFP, P4_MFP, P5_MFP and P7_MFP registers.

6.5.4 Functional Description

6.5.4.1 Input Mode

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

6.5.4.2 Push-pull Output Mode

Set Px_PMD (PMDn[1:0]) to 01 as the Px.n pin is in Push-pull output mode and the I/O pin supports digital output function with source/sink current capability. The bit value in the corresponding Px_DOUT[n] bit is driven on the pin.

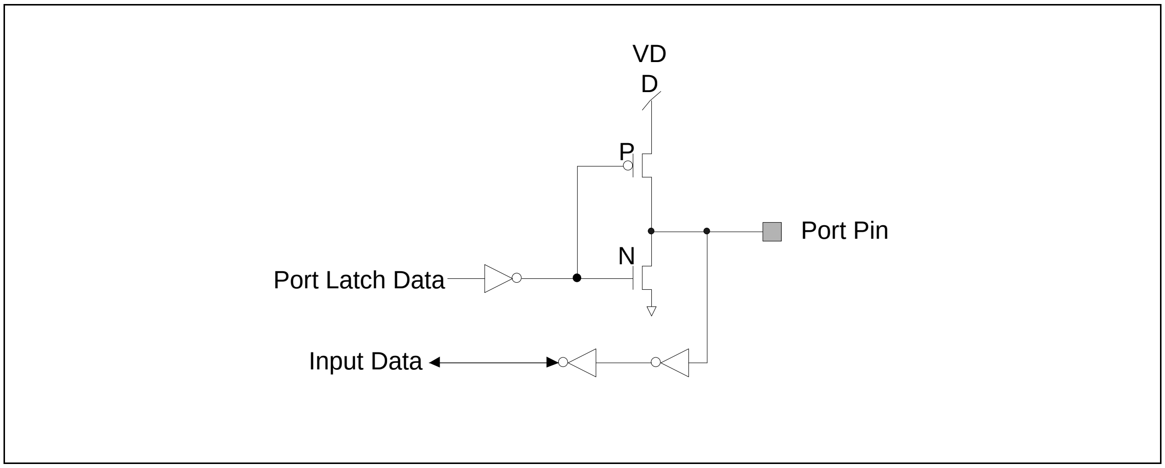


Figure 6.5-1 Push-Pull Output

6.5.4.3 Open-drain Output Mode

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

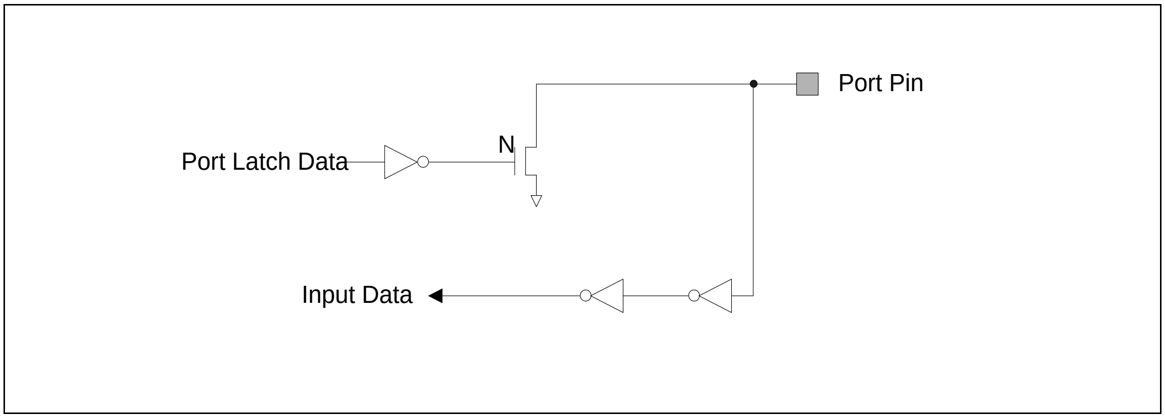


Figure 6.5-2 Open-Drain Output

6.5.4.4 Quasi-bidirectional Mode

Set Px_PMD (PMDn[1:0]) to 11 as the Px.n pin is in Quasi-bidirectional mode and the I/O pin supports digital output and input function at the same time but the source current is only up to hundreds of uA. Before the digital input function is performed the corresponding Px_DOUT[n] bit must be set to 1. The Quasi-bidirectional output is common on the 80C51 and most of its derivatives. If the bit value in the corresponding Px_DOUT[n] bit is 0, the pin drive a low output on the pin. If the bit value in the corresponding Px_DOUT[n] bit is 1, the pin will check the pin value. If pin value is high, no action takes. If pin state is low, then pin will drive strong high with 2 clock cycles on the pin and then disable the strong output drive. Meanwhile, the pin status is controlled by internal pull-up resistor.

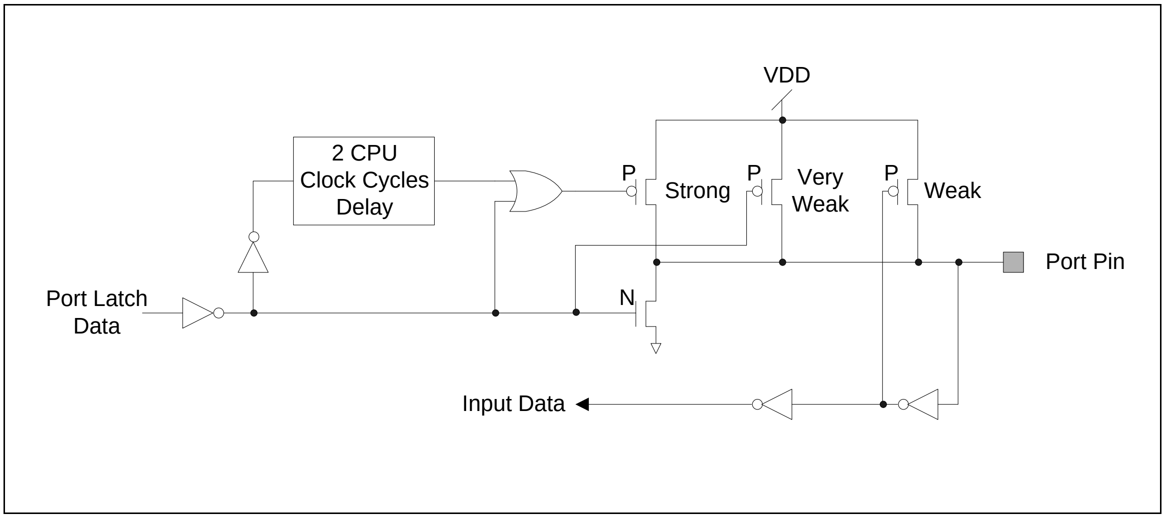


Figure 6.5-3 Quasi-Bidirectional I/O Mode

6.5.5 GPIO Interrupt and Wake-up Function

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

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

6.5.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
GP_BA = 0x5000_4000				
P0_PMD	**GP_BA+0x000**	**R/W**	**P0 Pin I/O Mode Control**	**0x0000_FFFF**
P0_OFFD	**GP_BA+0x004**	**R/W**	**P0 Digital Input Path Disable Control**	**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 Pin I/O Mode Control**	**0x0000_FFFF**
P1_OFFD	**GP_BA+0x044**	**R/W**	**P1 Digital Input Path Disable Control**	**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 Pin I/O Mode Control**	**0x0000_FFFF**
P2_OFFD	**GP_BA+0x084**	**R/W**	**P2 Digital Input Path Disable Control**	**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 Pin I/O Mode Control	0x0000_FFFF
P3_OFFD	GP_BA+0x0C4	R/W	P3 Digital Input Path Disable Control	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 Pin I/O Mode Control	0x0000_FFFF
P4_OFFD	GP_BA+0x104	R/W	P4 Digital Input Path Disable Control	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

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

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
P5_PMD	GP_BA+0x2C0	R/W	P5 Pin I/O Mode Control	0x0000_FFFF
P5_OFFD	GP_BA+0x2C4	R/W	P5 Digital Input Path Disable Control	0x0000_0000
P5_DOUT	GP_BA+0x2C8	R/W	P5 Data Output Value	0x0000_00FF
P5_DMASK	GP_BA+0x2CC	R/W	P5 Data Output Write Mask	0x0000_0000
P5_PIN	GP_BA+0x2D0	R	P5 Pin Value	0x0000_00XX
P5_DBEN	GP_BA+0x2D4	R/W	P5 De-bounce Enable	0x0000_0000
P5_IMD	GP_BA+0x2D8	R/W	P5 Interrupt Mode Control	0x0000_0000
P5_IEN	GP_BA+0x2DC	R/W	P5 Interrupt Enable	0x0000_0000
P5_ISRC	GP_BA+0x2E0	R/WC	P5 Interrupt Source Flag	0xXXXX_XXXX
P6_PMD	GP_BA+0x300	R/W	P6 Pin I/O Mode Control	0x0000_FFFF
P6_OFFD	GP_BA+0x304	R/W	P6 Digital Input Path Disable Control	0x0000_0000
P6_DOUT	GP_BA+0x308	R/W	P6 Data Output Value	0x0000_00FF
P6_DMASK	GP_BA+0x30C	R/W	P6 Data Output Write Mask	0x0000_0000
P6_PIN	GP_BA+0x310	R	P6 Pin Value	0x0000_00XX
P6_DBEN	GP_BA+0x314	R/W	P6 De-bounce Enable	0x0000_0000
P6_IMD	GP_BA+0x318	R/W	P6 Interrupt Mode Control	0x0000_0000
P6_IEN	GP_BA+0x31C	R/W	P6 Interrupt Enable	0x0000_0000
P6_ISRC	GP_BA+0x320	R/WC	P6 Interrupt Source Flag	0xXXXX_XXXX
P7_PMD	GP_BA+0x340	R/W	P7 Pin I/O Mode Control	0x0000_FFFF
P7_OFFD	GP_BA+0x344	R/W	P7 Digital Input Path Disable Control	0x0000_0000
P7_DOUT	GP_BA+0x348	R/W	P7 Data Output Value	0x0000_00FF
P7_DMASK	GP_BA+0x34C	R/W	P7 Data Output Write Mask	0x0000_0000
P7_PIN	GP_BA+0x350	R	P7 Pin Value	0x0000_00XX
P7_DBEN	GP_BA+0x354	R/W	P7 De-bounce Enable	0x0000_0000
P7_IMD	GP_BA+0x358	R/W	P7 Interrupt Mode Control	0x0000_0000
P7_IEN	GP_BA+0x35C	R/W	P7 Interrupt Enable	0x0000_0000
P7_ISRC	GP_BA+0x360	R/WC	P7 Interrupt Source Flag	0xXXXX_XXXX

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

P50_DOUT	**GP_BA+0x380**	**R/W**	**P5.0 Data Output Value**	**0x0000_0001**
P51_DOUT	**GP_BA+0x384**	**R/W**	**P5.1 Data Output Value**	**0x0000_0001**
P52_DOUT	**GP_BA+0x388**	**R/W**	**P5.2 Data Output Value**	**0x0000_0001**
P53_DOUT	**GP_BA+0x38C**	**R/W**	**P5.3 Data Output Value**	**0x0000_0001**
P54_DOUT	**GP_BA+0x390**	**R/W**	**P5.4 Data Output Value**	**0x0000_0001**
P55_DOUT	**GP_BA+0x394**	**R/W**	**P5.5 Data Output Value**	**0x0000_0001**
P56_DOUT	**GP_BA+0x398**	**R/W**	**P5.6 Data Output Value**	**0x0000_0001**
P57_DOUT	**GP_BA+0x39C**	**R/W**	**P5.7 Data Output Value**	**0x0000_0001**
P60_DOUT	**GP_BA+0x3A0**	**R/W**	**P6.0 Data Output Value**	**0x0000_0001**
P61_DOUT	**GP_BA+0x3A4**	**R/W**	**P6.1 Data Output Value**	**0x0000_0001**
P62_DOUT	**GP_BA+0x3A8**	**R/W**	**P6.2 Data Output Value**	**0x0000_0001**
P63_DOUT	**GP_BA+0x3AC**	**R/W**	**P6.3 Data Output Value**	**0x0000_0001**
P64_DOUT	**GP_BA+0x3B0**	**R/W**	**P6.4 Data Output Value**	**0x0000_0001**
P65_DOUT	**GP_BA+0x3B4**	**R/W**	**P6.5 Data Output Value**	**0x0000_0001**
P66_DOUT	**GP_BA+0x3B8**	**R/W**	**P6.6 Data Output Value**	**0x0000_0001**
P67_DOUT	**GP_BA+0x3BC**	**R/W**	**P6.7 Data Output Value**	**0x0000_0001**
P70_DOUT	**GP_BA+0x3C0**	**R/W**	**P7.0 Data Output Value**	**0x0000_0001**
P71_DOUT	**GP_BA+0x3C4**	**R/W**	**P7.1 Data Output Value**	**0x0000_0001**

6.5.7 Register Description

Port 0-7 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
P5_PMD	GP_BA+0x2C0	R/W	P5 Pin I/O Mode Control	0x0000_FFFF
P6_PMD	GP_BA+0x300	R/W	P6 Pin I/O Mode Control	0x0000_FFFF
P7_PMD	GP_BA+0x340	R/W	P7 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=0~7, n = 0~7

Port 0-7 Digital Input Path Disable Control (Px_OFFD)

Register	Offset	R/W	Description	Reset Value
P0_OFFD	GP_BA+0x004	R/W	P0 Digital Input Path Disable Control	0x0000_0000
P1_OFFD	GP_BA+0x044	R/W	P1 Digital Input Path Disable Control	0x0000_0000
P2_OFFD	GP_BA+0x084	R/W	P2 Digital Input Path Disable Control	0x0000_0000
P3_OFFD	GP_BA+0x0C4	R/W	P3 Digital Input Path Disable Control	0x0000_0000
P4_OFFD	GP_BA+0x104	R/W	P4 Digital Input Path Disable Control	0x0000_0000
P5_OFFD	GP_BA+0x2C4	R/W	P5 Digital Input Path Disable Control	0x0000_0000
P6_OFFD	GP_BA+0x304	R/W	P6 Digital Input Path Disable Control	0x0000_0000
P7_OFFD	GP_BA+0x344	R/W	P7 Digital Input Path Disable Control	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	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**OFFD**	**Port 0-7 Pin [n] Digital Input Path Disable Control** Each of these bits is used to control if the digital input path of corresponding Px.n pin is disabled. If input is analog signal, users can disable Px.n digital input path to avoid input current leakage. 0 = Px.n digital input path Enabled. 1 = Px.n digital input path Disabled (digital input tied to low). **Note:** x = 0~7, n = 0~7.
[15:0]	**Reserved**	Reserved.

Port 0-7 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
P5_DOUT	GP_BA+0x2C8	R/W	P5 Data Output Value	0x0000_00FF
P6_DOUT	GP_BA+0x308	R/W	P6 Data Output Value	0x0000_00FF
P7_DOUT	GP_BA+0x348	R/W	P7 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	Description	
[31:8]	**Reserved**	Reserved.
[15:0]	**DOUT[n]**	**Port 0-7 Pin [n] Output Value** Each of these bits controls the status of a Px.n pin when the Px.n is configured as Push-pull output, Open-drain output or Quasi-bidirectional mode. 0 = Px.n will drive Low if the Px.n pin is configured as Push-pull output, Open-drain output or Quasi-bidirectional mode. 1 = Px.n will drive High if the Px.n pin is configured as Push-pull output or Quasi-bidirectional mode. **Note:** x = 0~7, n = 0~7.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

Port0-7 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	0x0000_0000
P1_DMASK	GP_BA+0x04C	R/W	P1 Data Output Write Mask	0x0000_0000
P2_DMASK	GP_BA+0x08C	R/W	P2 Data Output Write Mask	0x0000_0000
P3_DMASK	GP_BA+0x0CC	R/W	P3 Data Output Write Mask	0x0000_0000
P4_DMASK	GP_BA+0x10C	R/W	P4 Data Output Write Mask	0x0000_0000
P5_DMASK	GP_BA+0x2CC	R/W	P5 Data Output Write Mask	0x0000_0000
P6_DMASK	GP_BA+0x30C	R/W	P6 Data Output Write Mask	0x0000_0000
P7_DMASK	GP_BA+0x34C	R/W	P7 Data Output Write Mask	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DMASK[n]**	**Port 0-7 Pin [n] Data Output Write Mask** These bits are used to protect the corresponding Px_DOUT[n] bit. When the DMASK[n] bit is set to 1, the corresponding Px_DOUT[n] bit is protected. If the write signal is masked, writing data to the protect bit is ignored. 0 = Corresponding Px_DOUT[n] bit can be updated. 1 = Corresponding Px_DOUT[n] bit protected. **Note1:** x = 0~7, n = 0~7. **Note2**: This function only protects the corresponding Px_DOUT[n] bit, and will not protect the corresponding Pxn_PDIO bit.

Port 0-7 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
P5_PIN	GP_BA+0x2D0	R	P5 Pin Value	0x0000_00XX
P6_PIN	GP_BA+0x310	R	P6 Pin Value	0x0000_00XX
P7_PIN	GP_BA+0x350	R	P7 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	Description	
[31:8]	**Reserved**	Reserved.
[15:0]	**PIN[n]**	**Port 0-7 Pin [n] Pin Value** Each bit of the register reflects the actual status of the respective Px.n pin. If the bit is 1, it indicates the corresponding pin status is high; else the pin status is low. **Note:** x = 0~7, n = 0~7.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

Port 0-7 De-bounce Enable (Px_DBEN)

Register	Offset	R/W	Description	Reset Value
P0_DBEN	GP_BA+0x014	R/W	P0 De-bounce Enable Control	0x0000_0000
P1_DBEN	GP_BA+0x054	R/W	P1 De-bounce Enable Control	0x0000_0000
P2_DBEN	GP_BA+0x094	R/W	P2 De-bounce Enable Control	0x0000_0000
P3_DBEN	GP_BA+0x0D4	R/W	P3 De-bounce Enable Control	0x0000_0000
P4_DBEN	GP_BA+0x114	R/W	P4 De-bounce Enable Control	0x0000_0000
P5_DBEN	GP_BA+0x2D4	R/W	P5 De-bounce Enable Control	0x0000_0000
P6_DBEN	GP_BA+0x314	R/W	P6 De-bounce Enable Control	0x0000_0000
P7_DBEN	GP_BA+0x354	R/W	P7 De-bounce Enable Control	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[15:0]	**DBEN[n]**	**Port 0-7 Pin [n] Input Signal De-bounce Enable Control** DBEN[n] bit is used to enable the de-bounce function for each corresponding bit. If the input signal pulse width cannot be sampled by continuous two de-bounce sample cycle, the input signal transition is seen as the signal bounce and will not trigger the interrupt. The de-bounce clock source is controlled by DBNCECON[4], one de-bounce sample cycle period is controlled by DBNCECON[3:0]. 0 = Px.n de-bounce function Disabled. 1 = Px.n de-bounce function Enabled. The de-bounce function is valid only for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored. **Note1:** x = 0~7, n = 0~7. **Note2**: If Px.n pin is chosen as Power-down wake-up source, user should be disable the de-bounce function before entering Power-down mode to avoid the second interrupt event occurred after system waken up which caused by Px.n de-bounce function.

Port 0-7 Interrupt Mode Control (Px_IMD)

Register	Offset	R/W	Description	Reset Value
P0_IMD	GP_BA+0x018	R/W	P0 Interrupt Mode Control	0x0000_0000
P1_IMD	GP_BA+0x058	R/W	P1 Interrupt Mode Control	0x0000_0000
P2_IMD	GP_BA+0x098	R/W	P2 Interrupt Mode Control	0x0000_0000
P3_IMD	GP_BA+0x0D8	R/W	P3 Interrupt Mode Control	0x0000_0000
P4_IMD	GP_BA+0x118	R/W	P4 Interrupt Mode Control	0x0000_0000
P5_IMD	GP_BA+0x2D8	R/W	P5 Interrupt Mode Control	0xXXXX_XX00
P6_IMD	GP_BA+0x318	R/W	P6 Interrupt Mode Control	0xXXXX_XX00
P7_IMD	GP_BA+0x358	R/W	P7 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	Description	
[31:8]	**Reserved**	Reserved.
[15:0]	**IMD[n]**	**Port 0-7 Pin [n] Edge or Level Detection Interrupt Mode Control** IMD[n] bit is used to control the triggered interrupt is by level trigger or by edge trigger. If the interrupt is by edge trigger, the trigger source can be controlled by de-bounce. If the interrupt is by level trigger, the input source is sampled by one HCLK clock and generates the interrupt. 0 = Edge trigger interrupt. 1 = Level trigger interrupt. If the pin is set as the level trigger interrupt, only one level can be set on the registers Px_IEN. If both levels to trigger interrupt are set, the setting is ignored and no interrupt will occur. The de-bounce function is valid only for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored. **Note:** x = 0~7, n = 0~7.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

Port 0-7 Interrupt Enable Control (Px_IEN)

Register	Offset	R/W	Description	Reset Value
P0_IEN	GP_BA+0x01C	R/W	P0 Interrupt Enable Control	0x0000_0000
P1_IEN	GP_BA+0x05C	R/W	P1 Interrupt Enable Control	0x0000_0000
P2_IEN	GP_BA+0x09C	R/W	P2 Interrupt Enable Control	0x0000_0000
P3_IEN	GP_BA+0x0DC	R/W	P3 Interrupt Enable Control	0x0000_0000
P4_IEN	GP_BA+0x11C	R/W	P4 Interrupt Enable Control	0x0000_0000
P5_IEN	GP_BA+0x2DC	R/W	P5 Interrupt Enable Control	0x0000_0000
P6_IEN	GP_BA+0x31C	R/W	P6 Interrupt Enable Control	0x0000_0000
P7_IEN	GP_BA+0x35C	R/W	P7 Interrupt Enable Control	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	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**IR_EN[n]**	**Port 0-7 Pin [n] Interrupt Enable by Input Rising Edge or Input Level High** IR_EN[n] bit is used to enable the interrupt for each of the corresponding input Px.n pin. Set bit to 1 also enable the pin wake-up function. When setting the IR_EN[n] bit to 1 : If the interrupt is level trigger (IMD[n] is 1), the input Px.n pin will generate the interrupt while this pin state is at high level. If the interrupt is edge trigger (IMD[n] is 0), the input Px.n pin will generate the interrupt while this pin state changed from low to high. 0 = Px.n level high or low to high interrupt Disabled. 1 = Px.n level high or low to high interrupt Enabled. **Note:** x = 0~7, n = 0~7.

[15:8]	**Reserved**	Reserved.
[7:0]	**IF_EN[n]**	**Port 0-7 Pin [n] Interrupt Enable by Input Falling Edge or Input Level Low** IF_EN[n] bit is used to enable the interrupt for each of the corresponding input Px.n pin. Set bit to 1 also enable the pin wake-up function. When setting the IF_EN[n] bit to 1 : If the interrupt is level trigger (IMD[n] is 1), the input Px.n pin will generate the interrupt while this pin state is at low level. If the interrupt is edge trigger (IMD[n] is 0), the input Px.n pin will generate the interrupt while this pin state changed from high to low. 0 = Px.n level low or high to low interrupt Disabled. 1 = Px.n level low or high to low interrupt Enabled. **Note:** x = 0~7, n = 0~7.

Port 0-7 Interrupt Source Flag (Px_ISRC)

Register	Offset	R/W	Description	Reset Value
P0_ISRC	GP_BA+0x020	R/W	P0 Interrupt Source Flag	0x0000_0000
P1_ISRC	GP_BA+0x060	R/W	P1 Interrupt Source Flag	0x0000_0000
P2_ISRC	GP_BA+0x0A0	R/W	P2 Interrupt Source Flag	0x0000_0000
P3_ISRC	GP_BA+0x0E0	R/W	P3 Interrupt Source Flag	0x0000_0000
P4_ISRC	GP_BA+0x120	R/W	P4 Interrupt Source Flag	0x0000_0000
P5_ISRC	GP_BA+0x2E0	R/WC	P5 Interrupt Source Flag	0x0000_0000
P6_ISRC	GP_BA+0x320	R/WC	P6 Interrupt Source Flag	0x0000_0000
P7_ISRC	GP_BA+0x360	R/WC	P7 Interrupt Source Flag	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**ISRC[n]**	**Port 0-7 Pin [n] Interrupt Source Flag** Write : 0 = No action. 1 = Clear the corresponding pending interrupt. Read : 0 = No interrupt at Px.n. 1 = Px.n generates an interrupt. **Note:** x = 0~7, n = 0~7.

Interrupt De-bounce Cycle Control (DBNCECON)

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

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**ICLK_ON**	**Interrupt Clock On Mode** 0 = Edge detection circuit is active only if I/O pin corresponding Px_IEN bit is set to 1. 1 = All I/O pins edge detection circuit is always active after reset. **Note:** It is recommended to turn off this bit to save system power if no special application concern.
[4]	**DBCLKSRC**	**De-bounce Counter Clock Source Selection** 0 = De-bounce counter clock source is the HCLK. 1 = De-bounce counter clock source is the 10 kHz internal low speed oscillator.
[3:0]	**DBCLKSEL**	**De-bounce Sampling Cycle Selection** 0000 = Sample interrupt input once per 1 clocks. 0001 = Sample interrupt input once per 2 clocks. 0010 = Sample interrupt input once per 4 clocks. 0011 = Sample interrupt input once per 8 clocks. 0100 = Sample interrupt input once per 16 clocks. 0101 = Sample interrupt input once per 32 clocks. 0110 = Sample interrupt input once per 64 clocks. 0111 = Sample interrupt input once per 128 clocks. 1000 = Sample interrupt input once per 256 clocks. 1001 = Sample interrupt input once per 2*256 clocks. 1010 = Sample interrupt input once per 4*256 clocks. 1011 = Sample interrupt input once per 8*256 clocks.

		1100 = Sample interrupt input once per 16*256 clocks. 1101 = Sample interrupt input once per 32*256 clocks. 1110 = Sample interrupt input once per 64*256 clocks. 1111 = Sample interrupt input once per 128*256 clocks.

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

Register	Offset	R/W	Description	Reset Value
P0n_PDIO n = 0,1…7	GP_BA+0x200 + 0x04 * n	R/W	GPIO P0.n Pin Data Input/Output	0x0000_000X
P1n_PDIO n = 0,1…7	GP_BA+0x220 + 0x04 * n	R/W	GPIO P1.n Pin Data Input/Output	0x0000_000X
P2n_PDIO n = 0,1…7	GP_BA+0x240 + 0x04 * n	R/W	GPIO P2.n Pin Data Input/Output	0x0000_000X
P3n_PDIO n = 0,1…7	GP_BA+0x260 + 0x04 * n	R/W	GPIO P3.n Pin Data Input/Output	0x0000_000X
P4n_PDIO n = 0,1…7	GP_BA+0x280 + 0x04 * n	R/W	GPIO P4.n Pin Data Input/Output	0x0000_000X
P5n_PDIO	GP_BA+0x380 + 0x04 * n	R/W	GPIO P5.n Pin Data Input/Output	0x0000_000X
P6n_PDIO	GP_BA+0x3A0 + 0x04 * n	R/W	GPIO P6.n Pin Data Input/Output	0x0000_000X
P7n_PDIO	GP_BA+0x3C0 + 0x04 * n	R/W	GPIO P7.n Pin Data Input/Output	0x0000_000X

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**Pxn_PDIO**	**GPIO Px.n Pin Data Input/Output** Writing this bit can control one GPIO pin output value. 0 = Corresponding GPIO pin set to low. 1 = Corresponding GPIO pin set to high.

		Read this register to get GPIO pin status. For example, writing P00_PDIO will reflect the written value to bit P0_DOUT[0], reading P00_PDIO will return the value of P0_PIN[0]. **Note1:** x = 0~7, n = 0~7. **Note2:** The writing operation will not be affected by register Px_DMASK[n].

6.6 Timer Controller (TMR)

6.6.1 Overview

The Timer Controller includes four 32-bit timers, TIMER0 ~ TIMER3, allowing user to easily implement a timer control for applications. The timer can perform functions, such as frequency measurement, delay timing, clock generation, and event counting by external input pins, and interval measurement by external capture pins.

6.6.2 Features

- Four sets of 32-bit timers with 24-bit up counter and one 8-bit prescale counter
- Independent clock source for each timer
- Provides four timer counting modes: one-shot, periodic, toggle and continuous counting
- Time-out period = (Period time of timer clock input) * (8-bit prescale counter + 1) * (24-bit TCMP)
- Maximum counting cycle time = (1 / T MHz) * (2^8) * (2^{24}), T is the period time of timer clock
- 24-bit up counter value is readable through TDR (Timer Data Register)
- Supports event counting function to count the input event from external counter pin (T0~T3)
- 24-bit capture value is readable through TCAP (Timer Capture Data Register)
- Supports external capture pin (T0EX~T3EX) for interval measurement
- Supports external capture pin (T0EX~T3EX) to reset 24-bit up counter
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated
- Supports Inter-Timer trigger mode

6.6.3 Block Diagram

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

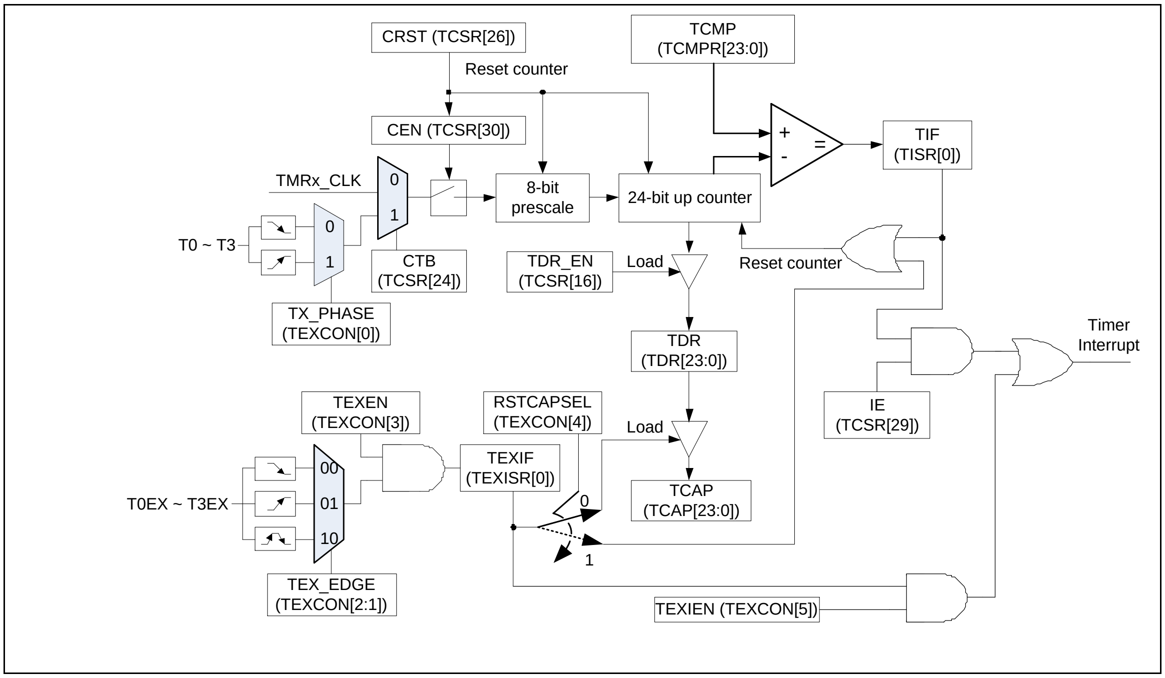
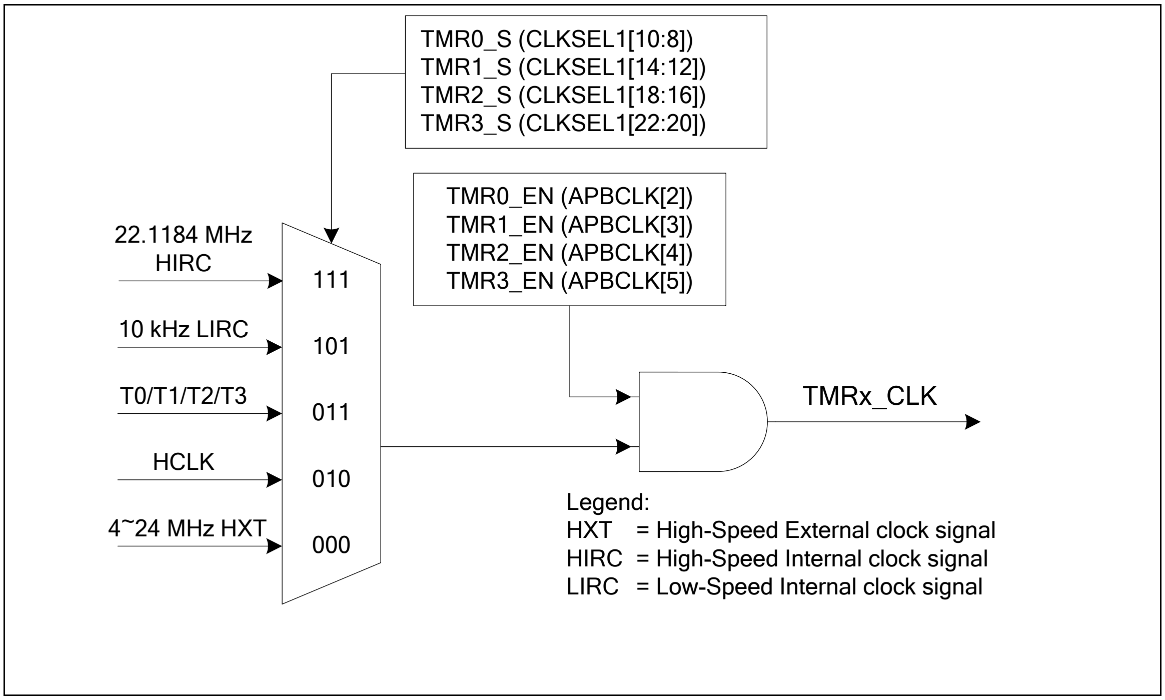


Figure 6.6-1 Timer Controller Block Diagram



Legend:
HXT = High-Speed External clock signal
HIRC = High-Speed Internal clock signal
LIRC = Low-Speed Internal clock signal

Figure 6.6-2 Clock Source of Timer Controller

6.6.4 Basic Configuration

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

6.6.5 Functional Description

6.6.5.1 Timer Interrupt Flag

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

6.6.5.2 Timer Counting Operation Mode

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

One-shot Mode

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

Periodic Mode

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

Toggle-output Mode

If timer controller is configured at toggle-out mode (TCSR[28:27] is 10) and CEN bit is set, the timer counter starts up counting. The counting operation of toggle-out mode is almost the same as periodic mode, except toggle-out mode has associated T0~T3 pin to output signal while specify TIF bit is set. Thus, the toggle-output signal on T0~T3 pin is changing back and forth with 50% duty cycle.

Continuous Counting Mode

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

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

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

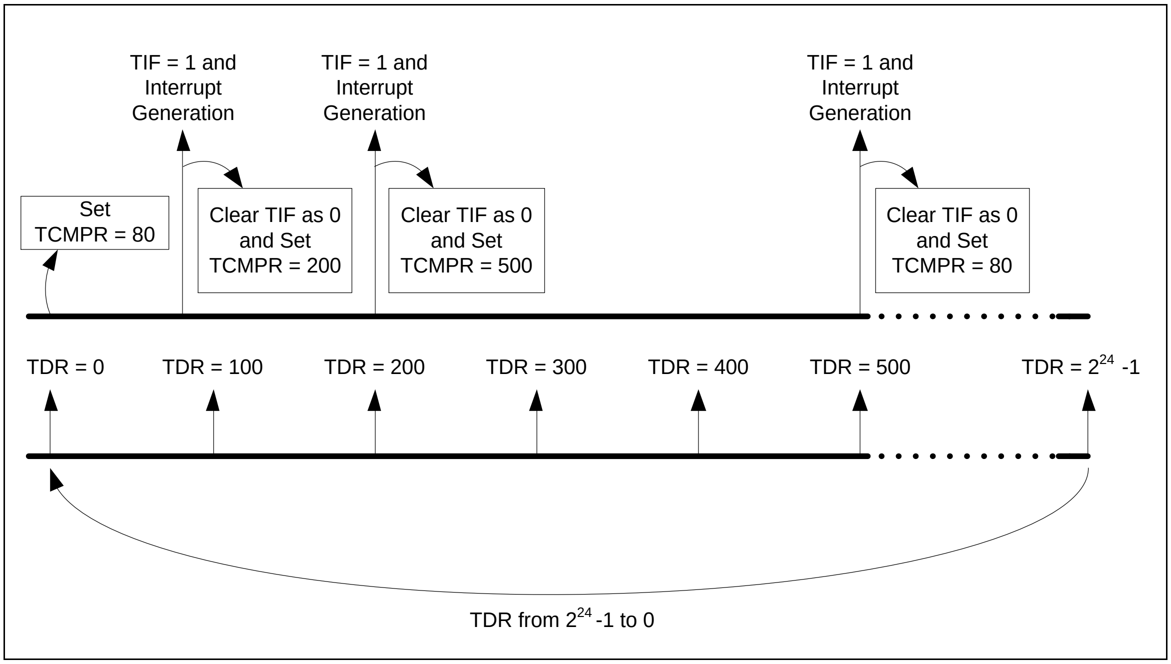

TIF = 1 and Interrupt Generation
TIF = 1 and Interrupt Generation
TIF = 1 and Interrupt Generation
Set TCMPR = 80
Clear TIF as 0 and Set TCMPR = 200
Clear TIF as 0 and Set TCMPR = 500
Clear TIF as 0 and Set TCMPR = 80
TDR = 0
TDR = 100
TDR = 200
TDR = 300
TDR = 400
TDR = 500
TDR = 2^24 -1
TDR from 2^24 -1 to 0

Figure 6.6-3 Continuous Counting Mode

6.6.5.3 Event Counting Mode

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

Software can enable or disable Tx pin de-bounce circuit by TCDB (TEXCON[7]) bit. The input event frequency should be less than 1/3 HCLK if Tx pin de-bounce disabled or less than 1/8 HCLK if Tx pin de-bounce enabled to assure the returned TDR value is incorrect, and software can also select edge detection phase of Tx pin by TX_PHASE (TEXCON[0]) bit.

In event counting mode, the timer counting operation mode can be selected as one-shot, periodic and continuous counting mode to counts the TDR value by input event from Tx pin.

6.6.5.4 Event Capture Mode

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

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

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

6.6.5.5 Event Reset Counter Mode

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

6.6.5.6 Inter-Timer Trigger Capture Mode

In this mode, the Timer0/2 will be forced in counter mode, counting with external event, and will generate an internal signal (INTR_TMR_TRG) to trigger Timer1/3 start or stop counting. Also, the Timer1/3 will be forced in capture mode and start/stop trigger-counting by Timer0/2 counter status.

Setting Timer0 Inter-timer Trigger Capture enabled, trigger-counting capture function is forced on Timer1. Setting Timer2 Inter-Timer Trigger enabled, trigger-counting capture function is forced on Timer3.

Start Trigger

While INTR_TRG_EN bit in Timer0/2 is set, the Timer0/2 will make a rising-edge transition of INTR_TMR_TRG while Timer0/2 24-bit counter value (TDR) is counting from 0x0 to 0x1 and Timer1/3 counter will start counting immediately and automatically.

Stop Trigger

When Timer0/2 TDR reaches the Timer0/2 TCMPR value, the Timer0/2 will make a falling-edge transition of INTR_TMR_TRG. Then Timer0/2 counter mode function will be disabled and INTR_TRG_EN bit will be cleared by hardware then Timer1/3 will stop counting also. At the same time, the Timer1/3 TDR value will be saved into Timer1/3 TCAP register.

User can use inter-timer trigger mode to measure the period of external event (Tx) more precisely. shows the sample flow of Inter-Timer Trigger Capture Mode for Timer0 as event counter mode and Timer1 as trigger-counting capture mode.

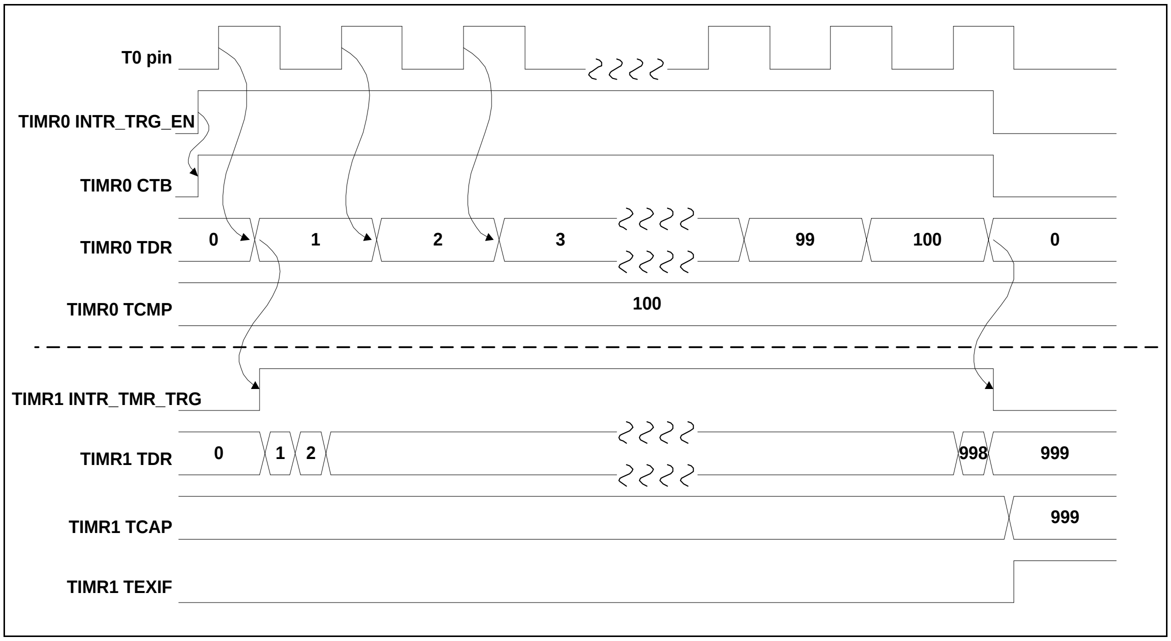


Figure 6.6-4 Inter-Timer Trigger Capture Timing

6.6.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
TIMER Base Address: **TMR01_BA = 0x4001_0000** **TMR23_BA = 0x4011_0000**				
TCSR0	TMR01_BA+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TCMPR0	TMR01_BA+0x04	R/W	Timer0 Compare Register	0x0000_0000
TISR0	TMR01_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TDR0	TMR01_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TCAP0	TMR01_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TEXCON0	TMR01_BA+0x14	R/W	Timer0 External Control Register	0x0000_0000
TEXISR0	TMR01_BA+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TCSR1	TMR01_BA+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TCMPR1	TMR01_BA+0x24	R/W	Timer1 Compare Register	0x0000_0000
TISR1	TMR01_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TDR1	TMR01_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TCAP1	TMR01_BA+0x30	R	Timer1 Capture Data Register	0x0000_0000
TEXCON1	TMR01_BA+0x34	R/W	Timer1 External Control Register	0x0000_0000
TEXISR1	TMR01_BA+0x38	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TCSR2	TMR23_BA+0x00	R/W	Timer2 Control and Status Register	0x0000_0005
TCMPR2	TMR23_BA+0x04	R/W	Timer2 Compare Register	0x0000_0000
TISR2	TMR23_BA+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TDR2	TMR23_BA+0x0C	R	Timer2 Data Register	0x0000_0000
TCAP2	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TEXCON2	TMR23_BA+0x14	R/W	Timer2 External Control Register	0x0000_0000
TEXISR2	TMR23_BA+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TCSR3	TMR23_BA+0x20	R/W	Timer3 Control and Status Register	0x0000_0005
TCMPR3	TMR23_BA+0x24	R/W	Timer3 Compare Register	0x0000_0000
TISR3	TMR23_BA+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000
TDR3	TMR23_BA+0x2C	R	Timer3 Data Register	0x0000_0000
TCAP3	TMR23_BA+0x30	R	Timer3 Capture Data Register	0x0000_0000

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

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

6.6.7 Register Description

Timer Control and Status Register (TCSR)

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

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

Bits	Description	
[31]	**DBGACK_TMR**	**ICE Debug Mode Acknowledge Disable (Write Protect)** 0 = ICE debug mode acknowledgement effects TIMER counting. Timer counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. Timer counter will keep going no matter CPU is held by ICE or not.
[30]	**CEN**	**Timer Enable Control** 0 = Stops/Suspends counting. 1 = Starts counting. **Note1**: In stop status, and then set CEN to 1 will enable the 24-bit up counter to keep counting from the last stop counting value. **Note2**: This bit is auto-cleared by hardware in one-shot mode (TCSR[28:27] = 00) when the timer interrupt flag (TIF) is generated.
[29]	**IE**	**Interrupt Enable Control** 0 = Timer Interrupt function Disabled. 1 = Timer Interrupt function Enabled. If this bit is enabled, when the timer interrupt flag (TIF) is set to 1, the timer interrupt signal is generated and inform to CPU.
[28:27]	**MODE**	**Timer Operating Mode**

		00 = The Timer controller is operated in One-shot mode. 01 = The Timer controller is operated in Periodic mode. 10 = The Timer controller is operated in Toggle-output mode. 11 = The Timer controller is operated in Continuous Counting mode.
[26]	**CRST**	**Timer Reset** 0 = No effect. 1 = Reset 8-bit prescale counter, 24-bit up counter value and CEN bit if CACT is 1.
[25]	**CACT**	**Timer Active Status (Read Only)** This bit indicates the 24-bit up counter status. 0 = 24-bit up counter is not active. 1 = 24-bit up counter is active.
[24]	**CTB**	**Counter Mode Enable Control** This bit is for external counting pin function enabled. When timer is used as an event counter, this bit should be set to 1 and select HCLK as timer clock source. Please refer to section 6.6.5.3 for detail description. 0 = External event counter mode Disabled. 1 = External event counter mode Enabled.
[23]	**WAKE_EN**	**Wake-up Function Enable Control** 0 = Wake-up trigger event Disabled. 1 = Wake-up trigger event Enabled.
[22]	**Reserved**	Reserved.
[21]	**TOUT_SEL**	**Toggle-output Pin Selection** 0 = Toggle-output pin is from Tx pin. 1 = Toggle-output pin is from TxEx pin.
[20]	**PERIODIC_SEL**	**Periodic Mode Behavior Selection Enable** 0 = The behavior selection in periodic mode is Disabled. When user updates TCMP while timer is running in periodic mode, TDR will be reset to default value. 1 = The behavior selection in periodic mode is Enabled. When user update TCMP while timer is running in periodic mode, the limitations as bellows list, If updated TCMP value > TDR, TCMP will be updated and TDR keep running continually. If updated TCMP value = TDR, timer time-out interrupt will be asserted immediately. If updated TCMP value < TDR, TDR will be reset to default value.
[19]	**INTR_TRG_EN**	**Inter-Timer Trigger Mode Enable Control** Setting this bit will enable the inter-timer trigger capture function. The Timer0/2 will be in counter mode and counting with external clock source or event. Also, Timer1/3 will be in trigger-counting mode of capture function. 0 = Inter-Timer Trigger Capture mode Disabled. 1 = Inter-Timer Trigger Capture mode Enabled.

		Note: For Timer1/3, this bit is ignored and the read back value is always 0.
[18:17]	**Reserved**	Reserved.
[16]	**TDR_EN**	**Data Load Enable Control** When TDR_EN is set, TDR (Timer Data Register) will be updated continuously with the 24-bit up-timer value as the timer is counting. 0 = Timer Data Register update Disabled. 1 = Timer Data Register update Enabled while Timer counter is active.
[15:8]	**Reserved**	Reserved.
[7:0]	**PRESCALE**	**Prescale Counter** Timer input clock source is divided by (PRESCALE+1) before it is fed to the Timer up counter. If this field is 0 (PRESCALE = 0), then there is no scaling.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

Timer Compare Register (TCMPR)

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TCMP [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	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**TCMP**	**Timer Compared Value** TCMP is a 24-bit compared value register. When the internal 24-bit up counter value is equal to TCMP value, the TIF flag will set to 1. Time-out period = (Period of Timer clock source) * (8-bit PRESCALE + 1) * (24-bit TCMP) **Note1**: Never write 0x0 or 0x1 in TCMP field, or the core will run into unknown state. **Note2**: When Timer is operating at Continuous Counting mode, the 24-bit up counter will keep counting continuously even if software writes a new value into TCMP field. But if Timer is operating at other modes except Periodic mode on M058S, the 24-bit up counter will restart counting and using newest TCMP value to be the timer compared value if software writes a new value into TCMP field.

Timer Interrupt Status Register (TISR)

Register	Offset	R/W	Description	Reset Value
TISR0	TMR01_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TISR1	TMR01_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TISR2	TMR23_BA+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TISR3	TMR23_BA+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TWF**	**Timer Wake-up Flag** This bit indicates the interrupt wake-up flag status of Time. 0 = Timer does not cause chip wake-up. 1 = Chip wake-up from Idle or Power-down mode if Timer time-out interrupt signal generated. **Note:** This bit is cleared by writing 1 to it.
[0]	**TIF**	**Timer Interrupt Flag** This bit indicates the interrupt flag status of Timer while TDR value reaches to TCMP value. 0 = No effect. 1 = TDR value matches the TCMP value. **Note:** This bit is cleared by writing 1 to it.

Timer Data Register (TDR)

Register	Offset	R/W	Description	Reset Value
TDR0	TMR01_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TDR1	TMR01_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TDR2	TMR23_BA+0x0C	R	Timer2 Data Register	0x0000_0000
TDR3	TMR23_BA+0x2C	R	Timer3 Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TDR[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	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**TDR**	**Timer Data Register** If TDR_EN is set to 1, TDR register value will be updated continuously to monitor 24-bit up counter value.

Timer Capture Data Register (TCAP)

Register	Offset	R/W	Description	Reset Value
TCAP0	TMR01_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TCAP1	TMR01_BA+0x30	R	Timer1 Capture Data Register	0x0000_0000
TCAP2	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TCAP3	TMR23_BA+0x30	R	Timer3 Capture Data Register	0x0000_0000

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

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

Timer External Control Register (TEXCON)

Register	Offset	R/W	Description	Reset Value
TEXCON0	TMR01_BA+0x14	R/W	Timer0 External Control Register	0x0000_0000
TEXCON1	TMR01_BA+0x34	R/W	Timer1 External Control Register	0x0000_0000
TEXCON2	TMR23_BA+0x14	R/W	Timer2 External Control Register	0x0000_0000
TEXCON3	TMR23_BA+0x34	R/W	Timer3 External Control Register	0x0000_0000

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

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

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

Timer External Interrupt Status Register (TEXISR)

Register	Offset	R/W	Description	Reset Value
TEXISR0	TMR01_BA+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TEXISR1	TMR01_BA+0x38	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TEXISR2	TMR23_BA+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TEXISR3	TMR23_BA+0x38	R/W	Timer3 External Interrupt Status Register	0x0000_0000

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

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

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

6.7 PWM Generator and Capture Timer (PWM)

6.7.1 Overview

The NuMicro M058S has one set of PWM groups supporting a total of two sets of PWM generators, which can be configured as four independent PWM outputs, PWM0~PWM3, or as two complementary PWM pairs, (PWM0, PWM1) and (PWM2, PWM3) with 2 programmable Dead-zone generators.

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

When DZEN01 (PCR[4]) is set, PWM0 and PWM1 perform complementary PWM paired function; the paired PWM period, duty and Dead-zone are determined by PWM0 timer and Dead-zone generator 0. Similarly, the complementary PWM pairs of (PWM2, PWM3) are controlled by PWM2 timers and Dead-zone generator 2. Refer to figures below 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 loaded into the 16-bit down counter/ comparator at the time down counter reaching 0. The double buffering feature avoids glitch at PWM outputs.

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

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

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

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

6.7.2 Features

PWM function:

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.

- PWMA (PWM group A) is a group of PWM which support 4 PWM channels or 2 complementary PWM paired channels
- PWM group has two PWM generators with each PWM generator supporting one 8-bit prescaler, two clock dividers, two PWM-Timers, one Dead-zone generator and two PWM outputs.
- Up to 16-bit resolution
- One-shot or Auto-reload mode
- Edge-aligned type or Center-aligned type option
- PWM trigger ADC start-to-conversion

Capture function:

- Timing control logic shared with PWM generators
- Supports 4 Capture input channels shared with 4 PWM output channels
- Each channel supports one rising latch register (CRLRx), one falling latch register (CFLRx) and Capture interrupt flag (CAPIFx)

6.7.3 Block Diagram

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

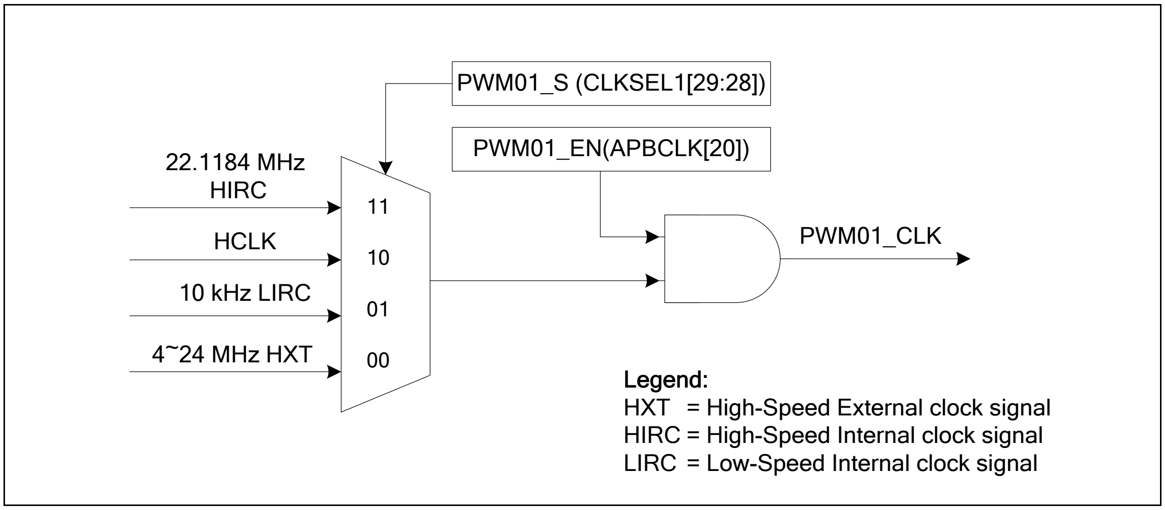

PWM01_S (CLKSEL1[29:28])
PWM01_EN(APBCLK[20])
22.1184 MHz
HIRC
HCLK
10 kHz LIRC
4~24 MHz HXT
11
10
01
00
PWM01_CLK
Legend:
HXT = High-Speed External clock signal
HIRC = High-Speed Internal clock signal
LIRC = Low-Speed Internal clock signal

Figure 6.7-1 PWM Generator 0 Clock Source Control

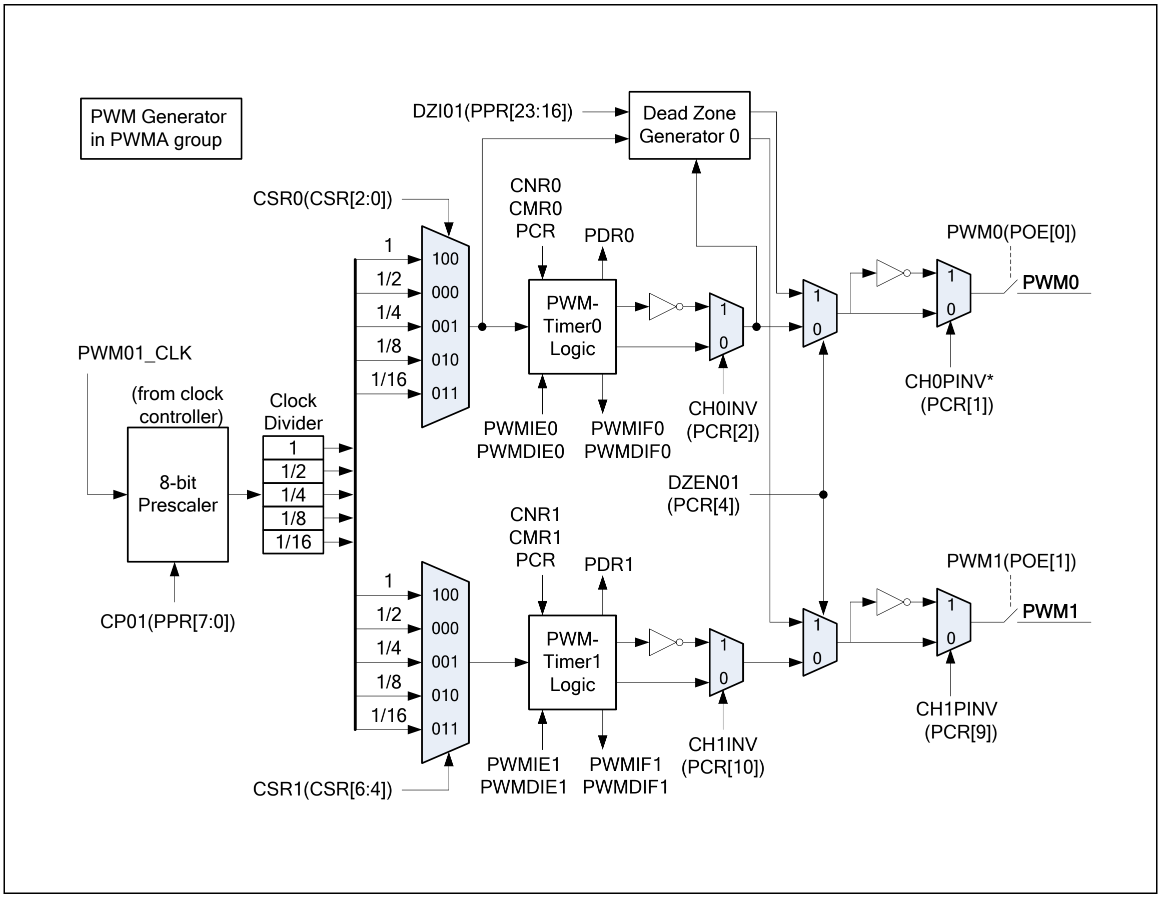


Figure 6.7-2 PWM Generator 0 Architecture Diagram

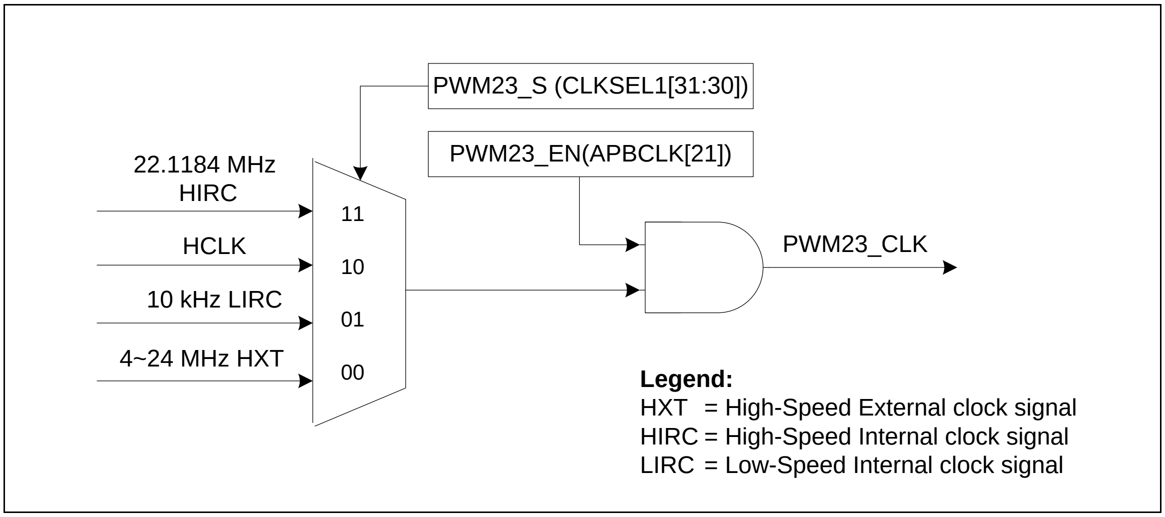

PWM23_S (CLKSEL1[31:30])
PWM23_EN(APBCLK[21])
22.1184 MHz
HIRC
HCLK
10 kHz LIRC
4~24 MHz HXT
11
10
01
00
PWM23_CLK
Legend:
HXT = High-Speed External clock signal
HIRC = High-Speed Internal clock signal
LIRC = Low-Speed Internal clock signal

Figure 6.7-3 PWM Generator 2 Clock Source Control

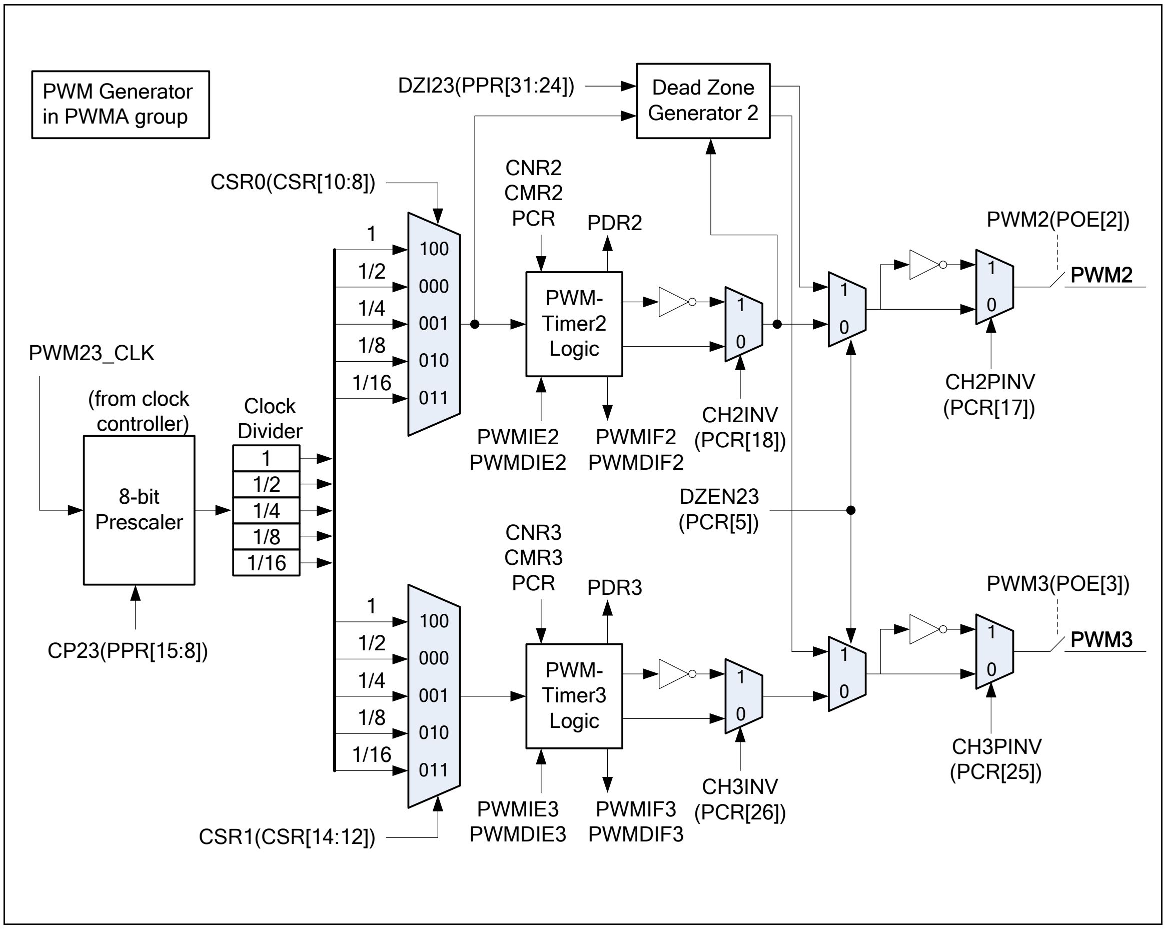


Figure 6.7-4 PWM Generator 2 Architecture Diagram

6.7.4 Basic Configuration

The PWM pin functions are configured in P2_MFP and P4_MFP registers.

The PWM clock can be enabled in APBCLK[21:20]. The PWM clock source is selected by CLKSEL1[31:28].

6.7.5 Functional Description

6.7.5.1 PWM-Timer Operation

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

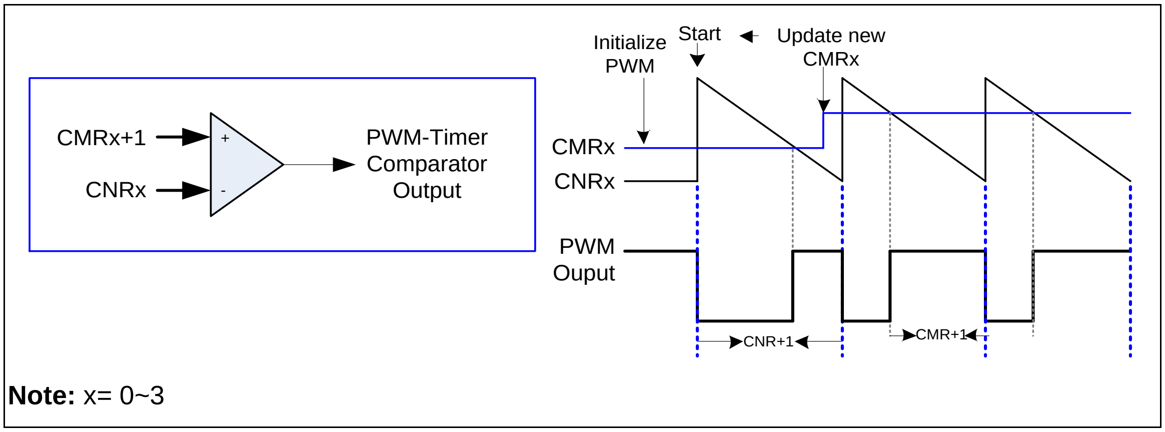
Edge-aligned PWM (Down-counter)

In Edge-aligned PWM Output mode, the 16 bits PWM counter will starts down-counting from CNRx to match with the value of the duty cycle CMRx (old), when this happened it will toggle the PWMx generator output to low. The counter will continue down-counting to zero, at this moment, it toggles the PWMx generator output to high and CMRx(new) and CNRx(new) are updated with CHxMODE=1 and request the PWM interrupt if PWM interrupt is enabled(PIERx=1).

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

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

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



Note: x= 0~3

Figure 6.7-5 Legend of Internal Comparator Output of PWM-Timer

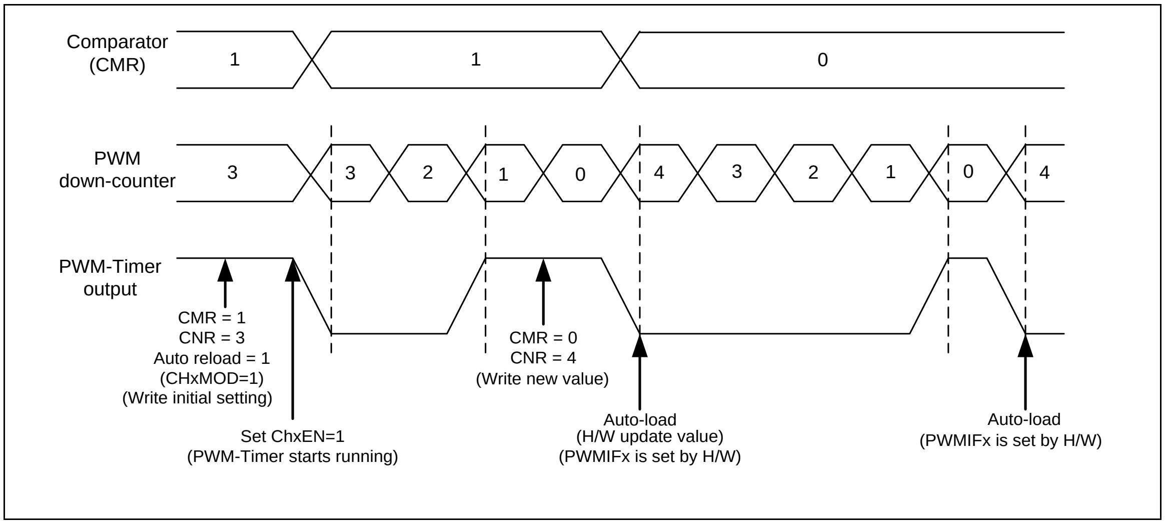


Figure 6.7-6 PWM Timer Operation Timing

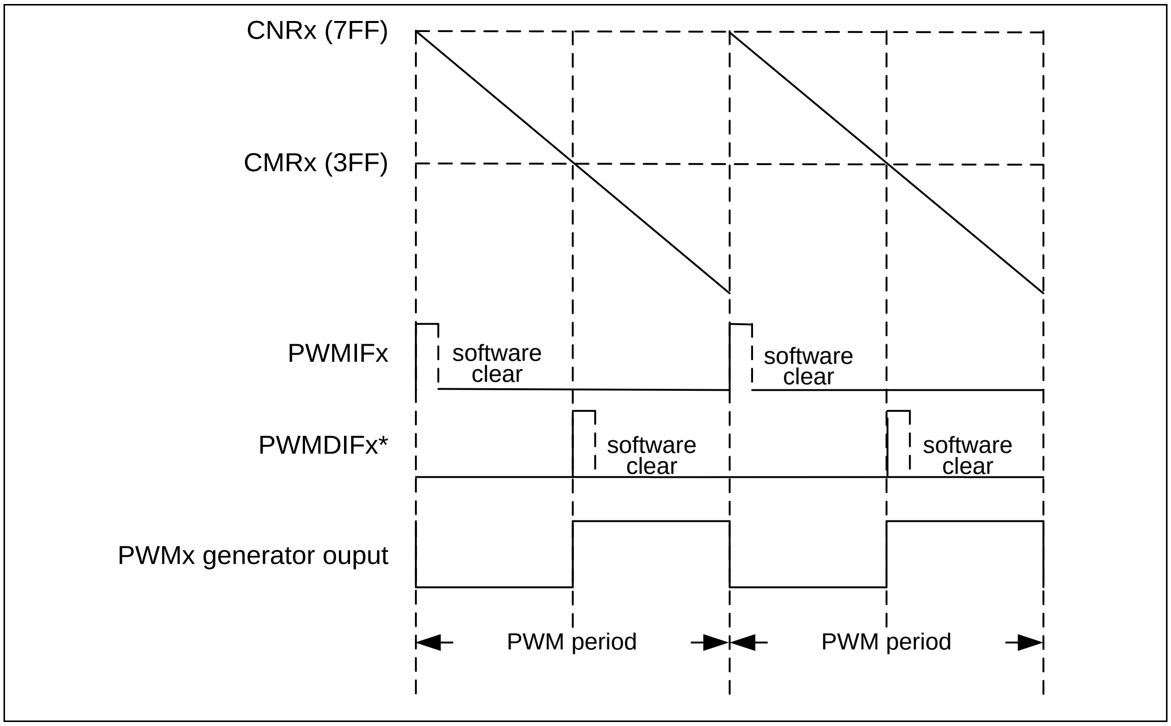


Figure 6.7-7 PWM Period Interrupt Generate Timing Waveform

Center-aligned PWM (Up/Down-counter)

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

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

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

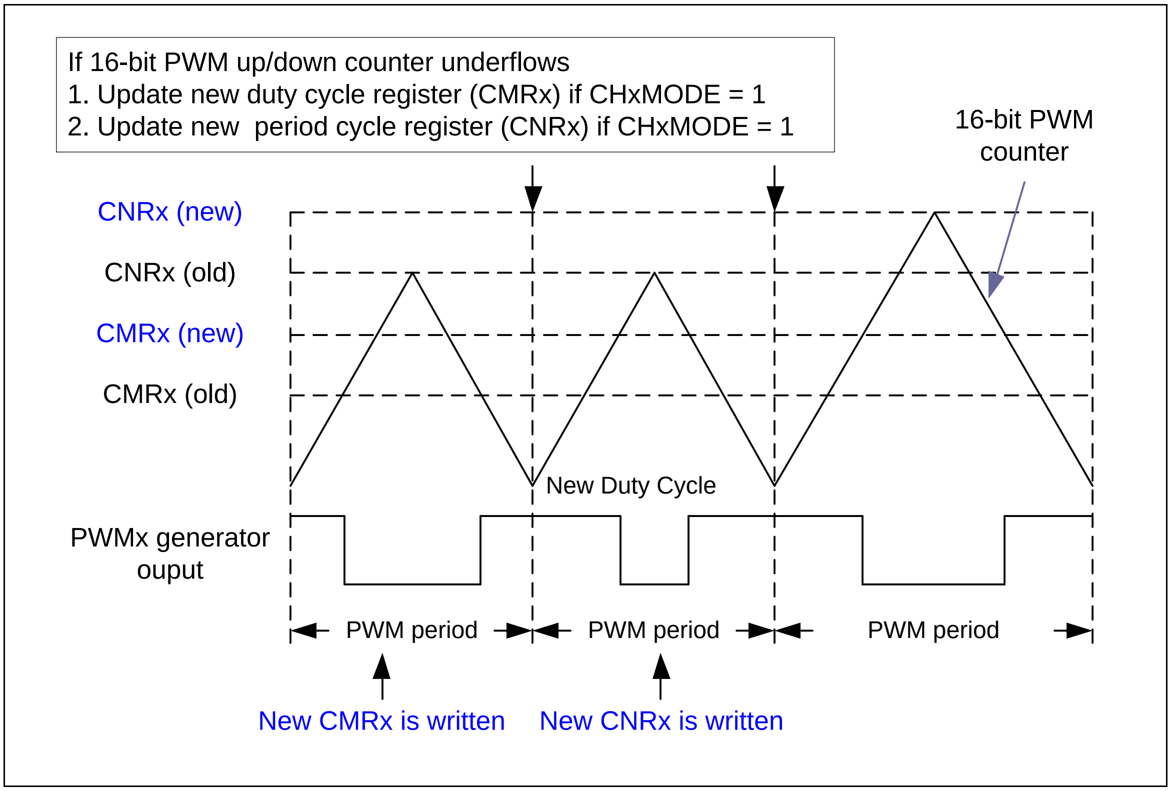


Figure 6.7-8 Center-aligned type Output Waveform

In Center-aligned type, system can generate two kinds of interrupt, period interrupt and duty interrupt, at four specified timings. PWM period interrupt is generated at counter equals zero on down-count if INTxxTYPE (PIER[17:16]) = 0 or at counter equals CNRx on up-count if INTxxTYPE (PIER[17:16]) = 1, i.e. at center point of PWM cycle. PWM duty interrupt is requested at counter equals CMR on down-count if INTxxDTYPE (PIER[25:24] = 0) or at counter equals CMR on up-count if INTxxDTYPE (PIER[25:24] = 1)

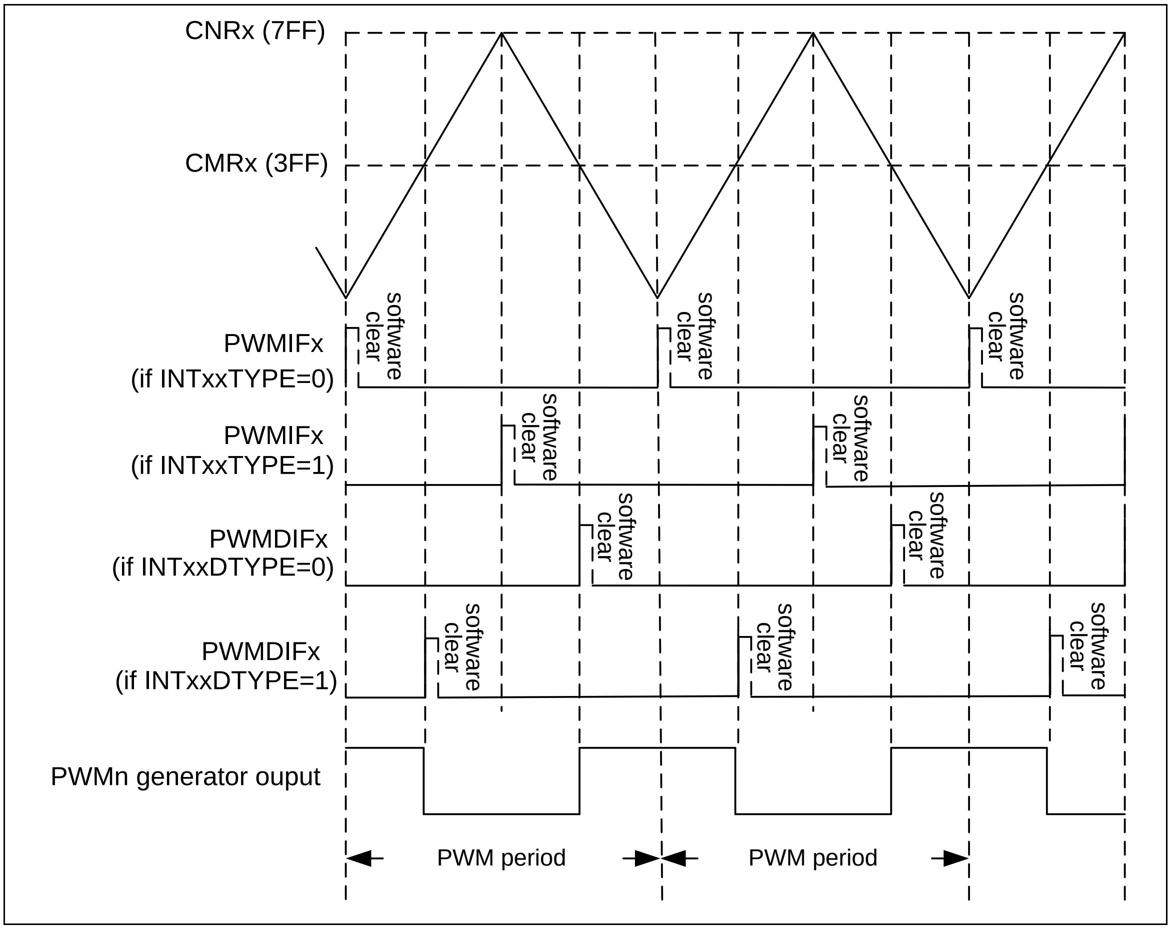


Figure 6.7-9 PWM Center-aligned Interrupt Generate Timing Waveform

6.7.5.2 *PWM Double Buffering, Auto-reload and One-shot Operation*

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

PWM0 will operate in One-shot mode if CH0MOD bit is set to 0, and operate in Auto-reload mode if CH0MOD bit is set to 1. It is recommend that switch PWM0 operating mode before set CH0EN bit to 1 to enable PWM0 counter start running because the content of CNR0 and CMR0 will be cleared to 0 to reset the PWM0 period and duty setting when PWM0 operating mode is changed. As PWM0 operate in One-shot mode, CMR0 and CNR0 should be written first and then set CH0EN bit to 1 to enable PWM0 counter start running. After PWM0 counter down count from CNR0 value to 0, CNR0 and CMR0 will be cleared to 0 by hardware and PWM counter will be held. Software need to write new CMR0 and CNR0 value to set next one-shot period and duty.

When re-start next one-shot operation, the CMR0 should be written first because PWM0 counter will auto re-start counting when CNR0 is written a non-zero value. As PWM0 operates at auto-reload mode, CMR0 and CNR0 should be written first and then set CH0EN bit to 1 to enable PWM0 counter start running. The value of CNR0 will reload to PWM0 counter when it down count reaches 0. If CNR0 is set to 0, PWM0 counter will be held. PWM1~PWM3 performs the same function as PWM0.

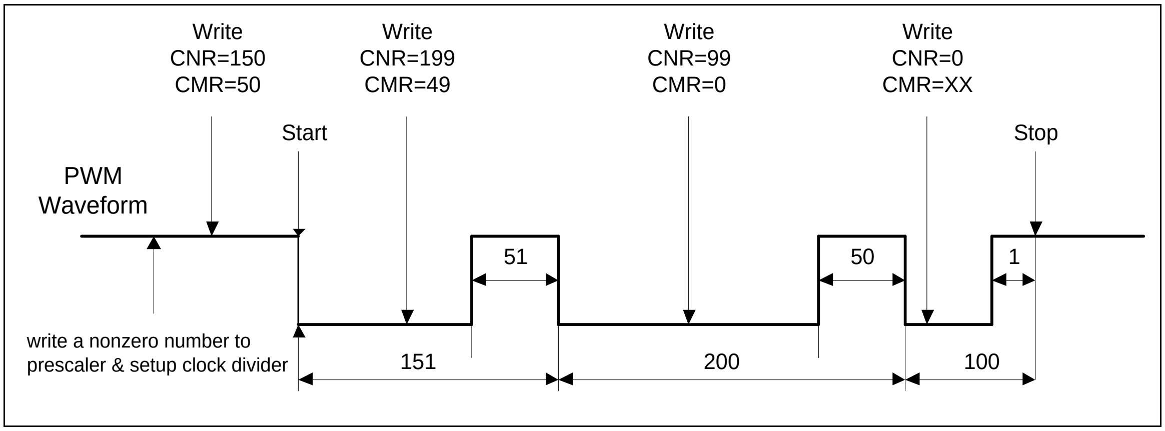


Figure 6.7-10 PWM Double Buffering Illustration

6.7.5.3 *Modulate Duty Ratio*

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

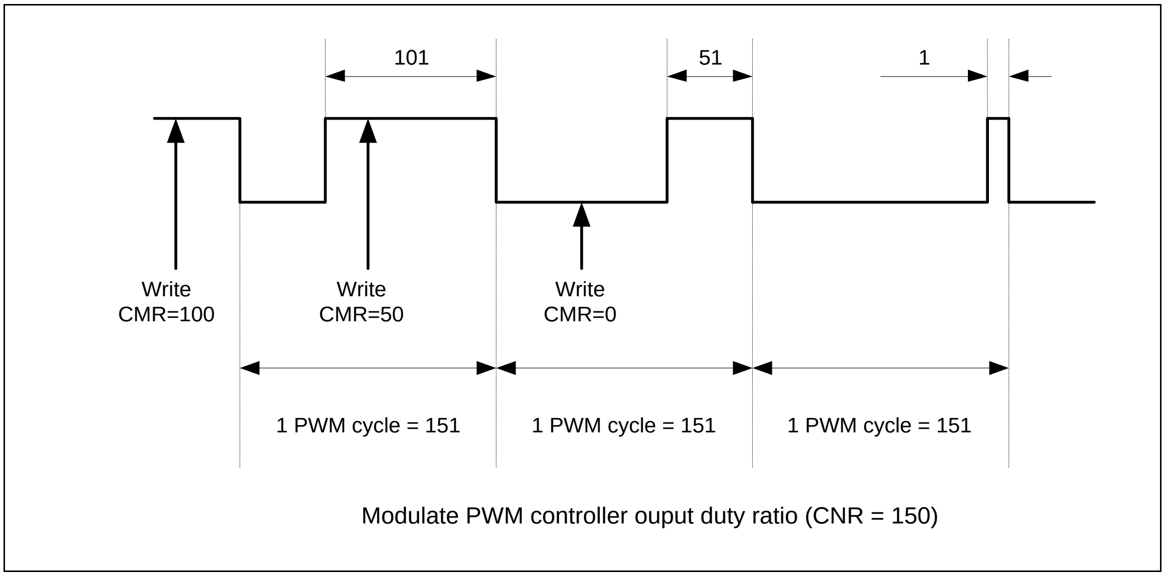


Figure 6.7-11 PWM Controller Output Duty Ratio

6.7.5.4 Dead-Zone Generator

The PWM controller is implemented with Dead-zone generator. They are built for power device protection. This function generates a programmable time gap to delay PWM rising output. User can program PPRx.DZI to determine the Dead-zone interval.

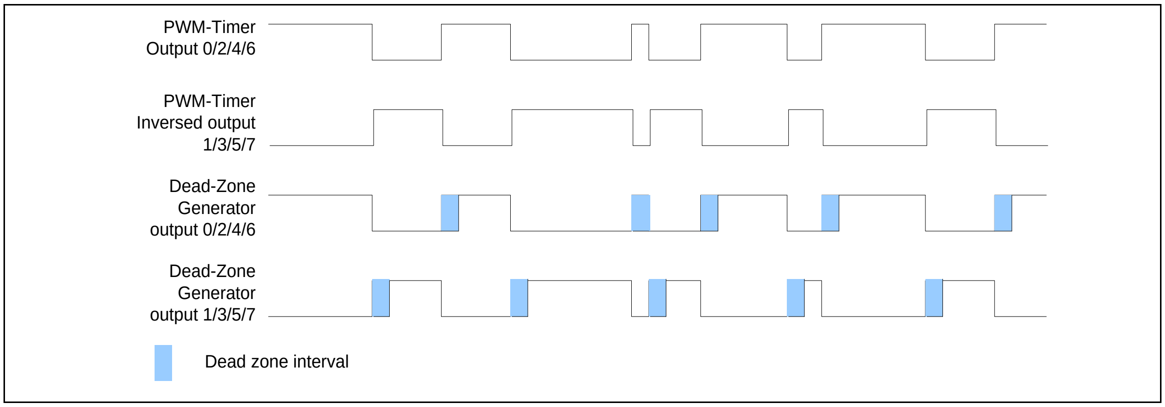

PWM-Timer Output 0/2/4/6
PWM-Timer Inversed output 1/3/5/7
Dead-Zone Generator output 0/2/4/6
Dead-Zone Generator output 1/3/5/7
Dead zone interval

Figure 6.7-12 Paired-PWM Output with Dead-zone Generation Operation

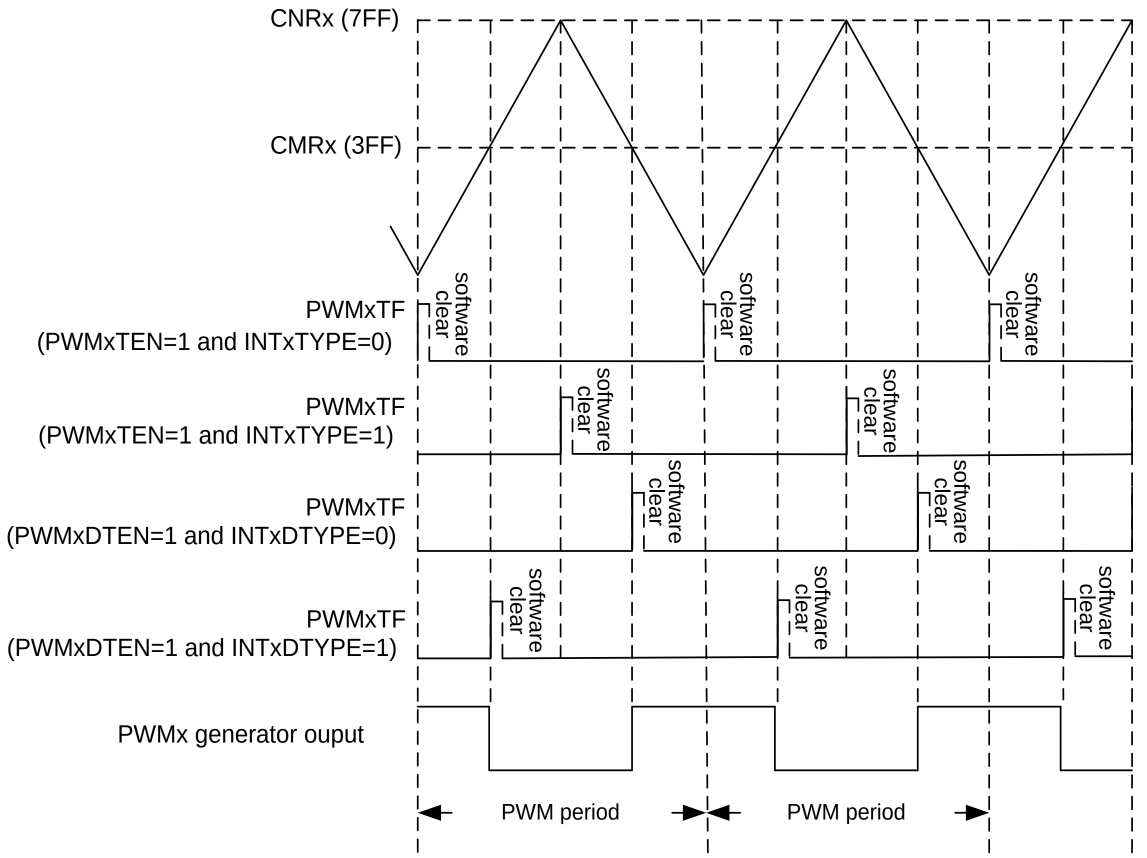
6.7.5.5 PWM-Timer Trigger ADC to Start Conversion

PWM can be one of the ADC conversion trigger source. When PWM operating at center aligned type, four PWM trigger ADC-to-Conversion timings can be selected. When PWM operating at edge-aligned type, two PWM trigger ADC-to-Conversion timings can be selected.

For Center-aligned type, PWM can trigger ADC to start conversion in different timings by setting INTxxDTYPE/ INTxxTYPE and PWMxDTEN/ PWMxTEN. Trigger ADC timings and settings as shown below.

INTxTEN	INTxxTYPE	Description
1	0	PWM counter equals to 0 at down-counting
1	1	PWM counter equal to CNRx + 1 at up-counting

INTxDTEN	INTxxDTYPE	Description
1	0	PWM counter equals CMRx at down-counting
1	1	PWM counter equals CMRx at up-counting


CNRx (7FF)
CMRx (3FF)
PWMxTF
(PWMxTEN=1 and INTxTYPE=0)
PWMxTF
(PWMxTEN=1 and INTxTYPE=1)
PWMxTF
(PWMxDTEN=1 and INTxDTYPE=0)
PWMxTF
(PWMxDTEN=1 and INTxDTYPE=1)
software clear
PWMx generator ouput
PWM period
PWM period

Note1: The ADST bit can be set to 1 when PWMxTF is set to 1.

Figure 6.7-13 PWM Trigger ADC Flag (PWMxTF) in Center-aligned Type Timing Waveform

For Edge-aligned type, PWM can trigger ADC to start conversion in different timings by setting PWMxDTEN/ PWMxTEN. Trigger ADC timings and settings as shown below.

INTxDTEN	Description
1	PWM counter equals CMRx at down-counting
INTxTEN	**Description**
1	PWM counter equals to 0 at down-counting

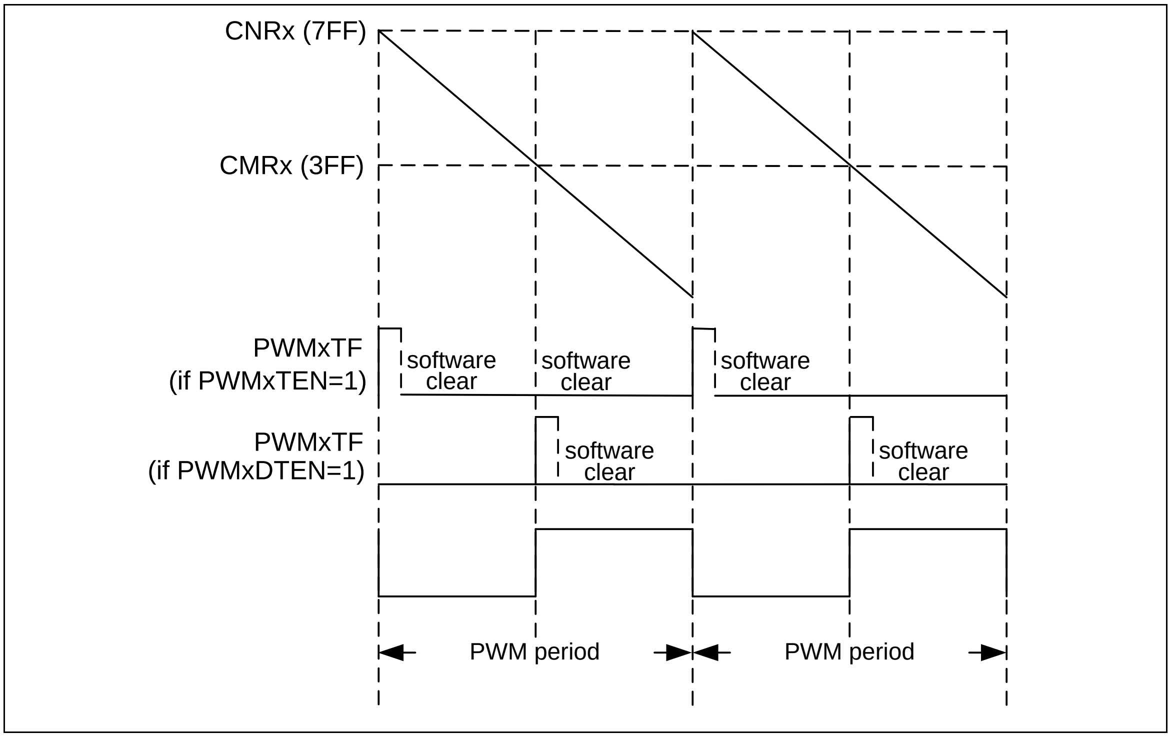


Figure 6.7-14 PWM Trigger ADC Flag (PWMxTF) in Edge-aligned Type Timing Waveform

6.7.5.6 *PWM trigger ADC start-of-conversion procedure*

1. Configure prescaler register (PPRx) for setting clock prescaler (CPxx).
2. Configure clock select register (CSRx) for setting clock source divider selection (CSRx).
3. Configure PWM control register (PCRx) for setting auto-reload mode (CHxMOD = 1), PWM aligned type (PWMxxTYPE) and DISABLE PWM-Timer (CHxEN = 0).
4. Configure PWM control register (PCRx) for setting inverter on/off (CHxINV), polar inverse on/off (CHxPINV) and Dead-zone generator on/off (DZENxx). (optional)
5. Configure PWM interrupt enable register (PIER) for setting PWM period interrupt type (INTxxTYPE) and PWM duty interrupt type (INTxxDTYPE).
6. Configure PWM trigger control register (TCON) to enable/disable PWM period trigger ADC (PWMxTEN) and PWM duty trigger ADC (PWMxDTEN).
7. Configure comparator register (CMRx) for setting PWM duty (CMRx).
8. Configure PWM counter register (CNRx) for setting PWM-Timer loaded value (CNRx).
9. Configure PWM output enable register (POE) to enable PWM output channel (PWMx = 1)
10. Configure ADC trigger delay control register (ADTDCR) in ADC controller registers for setting PWM trigger delay time (PTDT) (optional)
11. Configure ADC channel enable register (ADCHER) in ADC controller registers to enable analog input channel (CHENx =1)
12. Configure ADC control register (ADCR) in ADC controller registers for setting hardware trigger source is from PWM trigger (TRGS = 3), external trigger enable (TRGEN = 1), A/D converter operation mode (ADMD = 0,2,3) and A/D converter enable (ADEN = 1)
13. Configure PWM control register (PCR) to enable PWM timer start running (CHxEN = 1)

14. ADST (ADCR[11]) bit will be set to 1 by hardware when PWM trigger flag (PWMxTF) is set to 1 by hardware.
15. Software can poll A/D conversion end flag – ADF (ADSR[0]) to check if the conversion is finished or not.
16. ADST (ADCR[11]) bit will be cleared to 0 automatically in single mode and single cycle scan mode. In continuous scan mode, the ADST bit will keep 1 until software clears it.

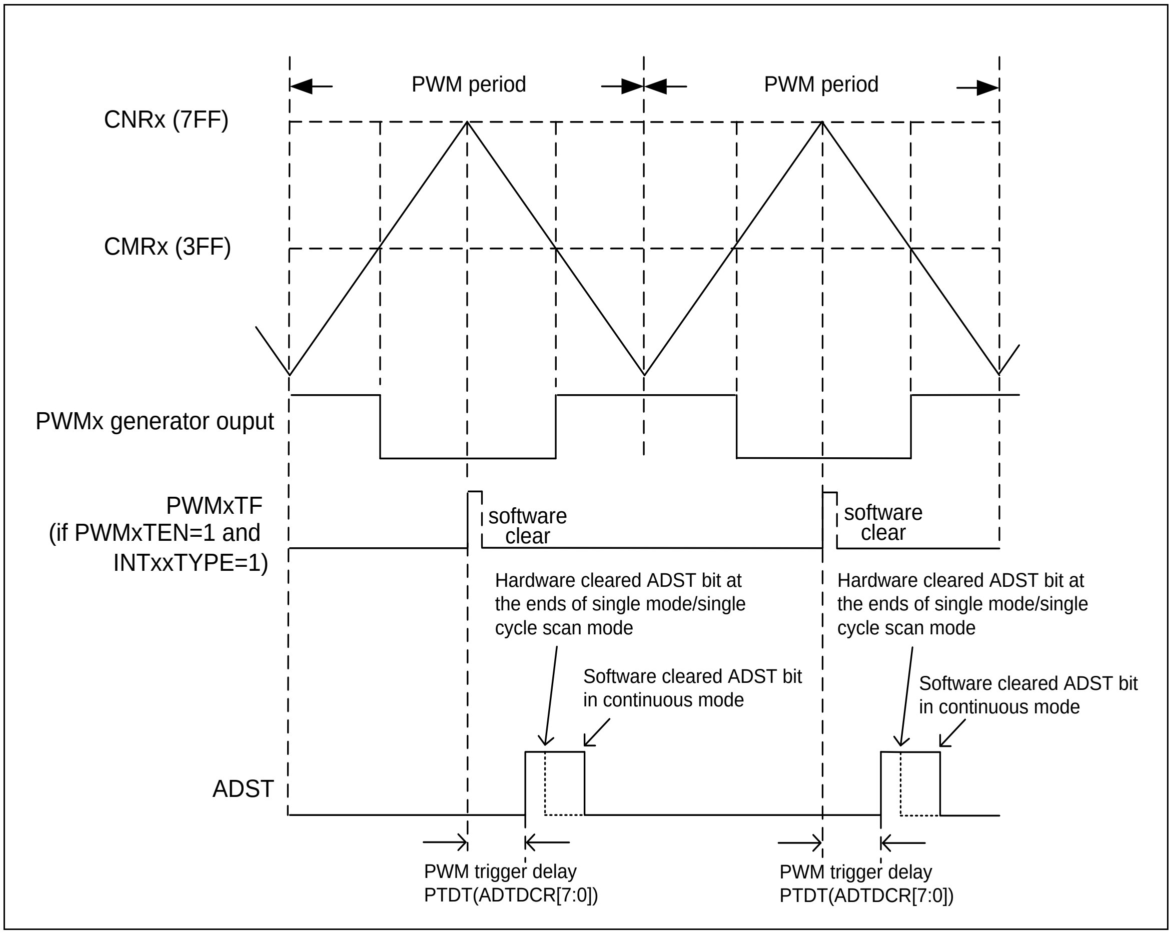
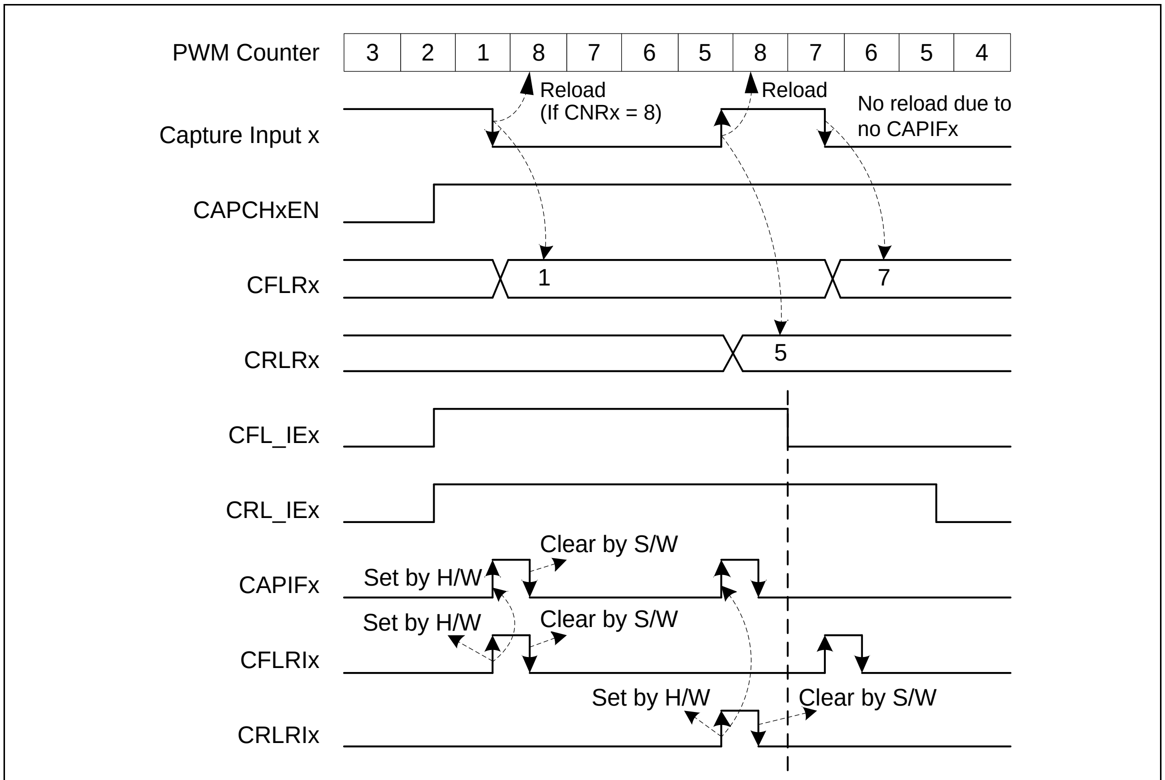


Figure 6.7-15 PWM Trigger ADC Start-of-Conversion in Center-aligned Type

6.7.5.7 Capture Operation

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



Note: x=0~3

Figure 6.7-16 Capture Operation Timing

In this case, the CNR is 8:

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

6.7.5.8 PWM-Timer Interrupt Architecture

There are four PWM interrupts, PWM0_INT~PWM3_INT, which are PWMA_INT for Advanced

Interrupt Controller (AIC). PWM 0 and Capture 0 share one interrupt, PWM1 and Capture 1 share the same interrupt and so on. Therefore, PWM function and Capture function in the same channel cannot be used at the same time. The following figure demonstrates the architecture of PWM-Timer interrupts.

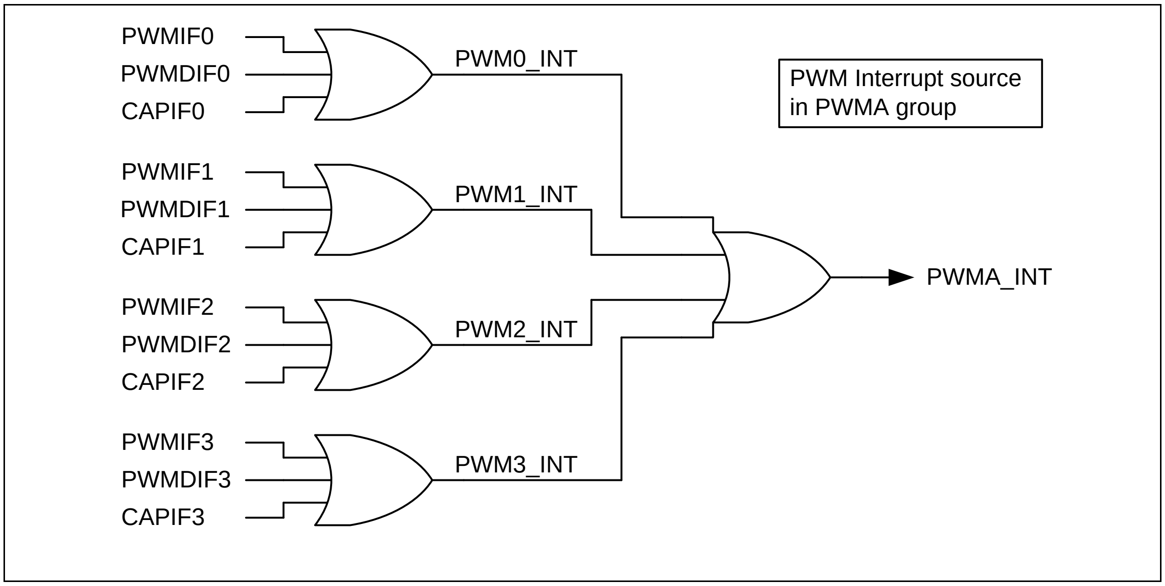


Figure 6.7-17 PWM Group A PWM-Timer Interrupt Architecture Diagram

6.7.5.9 PWM-Timer Start Procedure

The following procedure is recommended for starting a PWM drive.

1. Configure prescaler register (PPR) for setting clock prescaler (CPxx).
2. Configure clock select register (CSR) for setting clock source select (CSRx).
3. Configure PWM control register (PCR) for setting auto-reload mode (CHxMOD = 1) and DISABLE PWM-Timer (CHxEN = 0).
4. Configure PWM control register (PCR) for setting inverter on/off (CHxINV) and Dead-zone generator on/off (DZENxx). (optional)
5. Configure comparator register (CMRx) for setting PWM duty (CMRx).
6. Configure PWM counter register (CNRx) for setting PWM-Timer loaded value (CNRx).
7. Configure PWM interrupt enable register (PIER) for setting PWM period interrupt enable bit (PWMIEx). (optional)
8. Configure PWM output enable register (POE) to enable PWM output channel (PWMx = 1)
9. Configure PWM control register (PCR) to enable PWM timer start running (CHxEN = 1)

6.7.5.10 PWM-Timer Re-Start Procedure in One-shot mode

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

1. Configure comparator register (CMRx) for setting PWM duty (CMRx).
2. Configure PWM counter register (CNRx) for setting PWM period (CNRx). After setup CNRx, PWM wave will be generated.

6.7.5.11 PWM-Timer Stop Procedure

Method 1:

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

Method 2:

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

Method 3:

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

The reason why method 3 is not recommended is that disabling 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.

6.7.5.12 Capture Start Procedure

1. Configure prescaler register (PPR) for setting clock prescaler (CPxx).
2. Configure clock select register (CSR) for setting clock source select (CSRx)
3. Configure PWM control register (PCR) for setting auto-reload mode (CHxMOD = 1) and DISABLE PWM-Timer (CHxEN = 0).
4. Configure PWM capture control register (CCRx) for setting rising interrupt enable (CRL_IEx), falling interrupt enable (CFL_IEx) and input signal inverter on/off (INVx)
5. Configure PWM counter register (CNRx) for setting PWM-Timer loaded value (CNRx).
6. Configure PWM control register (PCR) to enable PWM timer start running (CHxEN = 1)
7. Configure the capture input channel enable register (CAPENR) to enable the corresponding GPIO pins as capture function.

6.7.6 Register Map

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

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

CFLR2	PWMA_BA+0x6C	R	PWM Capture Falling Latch Register (Channel 2)	0x0000_0000
CRLR3	PWMA_BA+0x70	R	PWM Capture Rising Latch Register (Channel 3)	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM Capture Falling Latch Register (Channel 3)	0x0000_0000
CAPENR	PWMA_BA+0x78	R/W	PWM Capture Input 0~3 Enable Register	0x0000_0000
POE	PWMA_BA+0x7C	R/W	PWM Output Enable Register for Channel 0~3	0x0000_0000
TCON	PWMA_BA+0x80	R/W	PWM Trigger Control Register for Channel 0~3	0x0000_0000
TSTATUS	PWMA_BA+0x84	R/W	PWM Trigger Status Register	0x0000_0000

6.7.7 Register Description

PWM Prescaler Register (PPR)

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

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

Bits	Description	
[31:24]	**DZI23**	**Dead-zone Interval for Pair of Channel 2 and Channel 3** These 8-bit determine the Dead-zone length. The unit time of Dead-zone length = [(prescale+1)*(clock source divider)]/ PWMxy_CLK (where xy could be 23 or 67, depends on selected PWM channel.)
[23:16]	**DZI01**	**Dead-zone Interval for Pair of Channel 0 and Channel 1** These 8-bit determine the Dead-zone length. The unit time of Dead-zone length = [(prescale+1)*(clock source divider)]/ PWMxy_CLK (where xy could be 01 or 45, depends on selected PWM channel.)
[15:8]	**CP23**	**Clock Prescaler 2** Clock input is divided by (CP23 + 1) before it is fed to the corresponding PWM-Timer.

		If CP23=0, then the clock prescaler 2 output clock will be stopped. So corresponding PWM-Timer will also be stopped.
[7:0]	**CP01**	**Clock Prescaler 0** Clock input is divided by (CP01 + 1) before it is fed to the corresponding PWM-Timer If CP01=0, then the clock prescaler 0 output clock will be stopped. So corresponding PWM-Timer will also be stopped.

PWM Clock Selector Register (CSR)

Register	Offset	R/W	Description	Reset Value
CSR	PWMA_BA+0x04	R/W	PWM Clock Source Divider Select Register	0x0000_0000

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

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:15]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[14:12]</td><td><b>CSR3</b></td><td><b>PWM Timer 3 Clock Source Divider</b>
Select clock source divider for PWM timer 3.
<table>
<tr><th>CSR3 [14:12]</th><th>Input Clock Divided by</th></tr>
<tr><td>000</td><td>2</td></tr>
<tr><td>001</td><td>4</td></tr>
<tr><td>010</td><td>8</td></tr>
<tr><td>011</td><td>16</td></tr>
<tr><td>100</td><td>1</td></tr>
</table></td></tr>
<tr><td>[11]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[10:8]</td><td><b>CSR2</b></td><td><b>PWM Timer 2 Clock Source Divider Selection</b>
Select clock source divider for PWM timer 2.
(Table is the same as CSR3)</td></tr>
<tr><td>[7]</td><td><b>Reserved</b></td><td>Reserved</td></tr>
<tr><td>[6:4]</td><td><b>CSR1</b></td><td><b>PWM Timer 1 Clock Source Divider</b>
Select clock source divider for PWM timer 1.
(Table is the same as CSR3)</td></tr>
<tr><td>[3]</td><td><b>Reserved</b></td><td>Reserved</td></tr>
<tr><td>[2:0]</td><td><b>CSR0</b></td><td><b>PWM Timer 0 Clock Source Divider</b>
Select clock source divider for PWM timer 0.</td></tr>
</table>

		(Table is the same as CSR3)

PWM Control Register (PCR)

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

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

Bits	Description	
[31]	**PWM23TYPE**	**PWM23 Aligned Type Selection Bit** 0 = Edge-aligned type. 1 = Center-aligned type.
[30]	**PWM01TYPE**	**PWM01 Aligned Type Selection Bit** 0 = Edge-aligned type. 1 = Center-aligned type.
[29:28]	**Reserved**	Reserved.
[27]	**CH3MOD**	**PWM-Timer 3 Auto-reload/One-Shot Mode Control** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause CNR3 and CMR3 be cleared.
[26]	**CH3INV**	**PWM-Timer 3 Output Inverter Enable Control** 0 = Inverter Disabled. 1 = Inverter Enabled.
[25]	**CH3PINV**	**PWM-Timer 3 Output Polar Inverse Enable Control** 0 = Polar Inverter Disabled. 1 = Polar Inverter Enabled.
[24]	**CH3EN**	**PWM-Timer 3 Enable Control** 0 = Corresponding PWM-Timer Stopped. 1 = Corresponding PWM-Timer Start Running.
[23:20]	**Reserved**	Reserved.

[19]	**CH2MOD**	**PWM-Timer 2 Auto-reload/One-Shot Mode Control** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause CNR2 and CMR2 be cleared.
[18]	**CH2INV**	**PWM-Timer 2 Output Inverter Enable Control** 0 = Inverter Disabled. 1 = Inverter Enabled.
[17]	**CH2PINV**	**PWM-Timer 2 Output Polar Inverse Enable** 0 = Polar Inverter Disabled. 1 = Polar Inverter Enabled.
[16]	**CH2EN**	**PWM-Timer 2 Enable** 0 = Corresponding PWM-Timer Stopped. 1 = Corresponding PWM-Timer Start Running.
[15:12]	**Reserved**	Reserved.
[11]	**CH1MOD**	**PWM-Timer 1 Auto-reload/One-Shot Mode** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause CNR1 and CMR1 be cleared.
[10]	**CH1INV**	**PWM-Timer 1 Output Inverter Enable** 0 = Inverter Disabled. 1 = Inverter Enabled.
[9]	**CH1PINV**	**PWM-Timer 1 Output Polar Inverse Enable** 0 = Polar Inverter Disabled. 1 = Polar Inverter Enabled.
[8]	**CH1EN**	**PWM-Timer 1 Enable** 0 = Corresponding PWM-Timer Stopped. 1 = Corresponding PWM-Timer Start Running.
[7:6]	**Reserved**	Reserved.
[5]	**DZEN23**	**Dead-zone 2 Generator Enable** 0 = Disabled. 1 = Enabled. **Note:** When Dead-zone generator is enabled, the pair of PWM2 and PWM3 becomes a complementary pair for PWM group A.
[4]	**DZEN01**	**Dead-zone 0 Generator Enable Control** 0 = Disabled. 1 = Enabled. **Note:** When Dead-zone generator is enabled, the pair of PWM0 and PWM1 becomes a complementary pair for PWM group A.
[3]	**CH0MOD**	**PWM-Timer 0 Auto-reload/One-Shot Mode Control** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause CNR0 and CMR0 be cleared.
[2]	**CH0INV**	**PWM-Timer 0 Output Inverter Enable Control**

		0 = Inverter Disabled. 1 = Inverter Enabled.
[1]	**CH0PINV**	**PWM-Timer 0 Output Polar Inverse Enable Control** 0 = Polar Inverter Disabled. 1 = Polar Inverter Enabled.
[0]	**CH0EN**	**PWM-Timer 0 Enable Control** 0 = The corresponding PWM-Timer stops running. 1 = The corresponding PWM-Timer starts running.

PWM Counter Register 3-0 (CNR3-0)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved
[15:0]	**CNRx**	**PWM Timer Loaded Value** CNR determines the PWM period. PWM frequency = PWMxy_CLK/[(prescale+1)*(clock divider)*(CNR+1)]; where xy, could be 01or 23, depends on selected PWM channel. **For Edge-aligned Type:** ● Duty ratio = (CMR+1)/(CNR+1). ● CMR >= CNR: PWM output is always high. ● CMR < CNR: PWM low width = (CNR-CMR) unit; PWM high width = (CMR+1) unit. ● CMR = 0: PWM low width = (CNR) unit; PWM high width = 1 unit. **For Center-aligned Type:** ● Duty ratio = [(2 x CMR) + 1]/[2 x (CNR+1)]. ● CMR > CNR: PWM output is always high. ● CMR <= CNR: PWM low width = 2 x (CNR-CMR) + 1 unit; PWM high width = (2

		x CMR) + 1 unit. ● CMR = 0: PWM low width = 2 x CNR + 1 unit; PWM high width = 1 unit (Unit = one PWM clock cycle) **Note1:** Any write operation to CNR will take effect in next PWM cycle. **Note2:** When CNR value is set to 0, PWM output is always high. **Note3:** When PWM operating at center-aligned type, CNR value should be set between 0x0001 to 0xFFFE. If CNR equal to 0x0000 or 0xFFFF, the PWM will work unpredictable.

PWM Comparator Register 3-0 (CMR3-0)

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

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

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

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

		● CMR <= CNR: PWM low width = 2 x (CNR-CMR) + 1 unit; PWM high width = (2 x CMR) + 1 unit. ● CMR = 0: PWM low width = 2 x CNR + 1 unit; PWM high width = 1 unit (Unit = one PWM clock cycle) **Note:** Any write operation to CMR will take effect in next PWM cycle.

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

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

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

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

PWM Interrupt Enable Register (PIER)

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

31	30	29	28	27	26	25	24
Reserved						**INT23DTYPE**	**INT01DTYPE**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**INT23TYPE**	**INT01TYPE**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**PWMDIE3**	**PWMDIE2**	**PWMDIE1**	**PWMDIE0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PWMIE3**	**PWMIE2**	**PWMIE1**	**PWMIE0**

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**INT23DTYPE**	**PWM23 Duty Interrupt Type Selection** 0 = PWMDIFx will be set if PWM counter down count and matches CMRx register. PWM will trigger ADC to conversion when PWM counter down count and matches CMRx register if correlate PWM trigger enable bit (PWMxDTEN) is set to 1. 1 = PWMDIFx will be set when PWM counter up count and matches CMRx register. PWM will trigger ADC to conversion when PWM counter up count and matches CMRx register if correlate PWM trigger enable bit (PWMxDTEN) is set to 1. **Note:** Set INT23DTYPE to 1 only work when PWM operating in center aligned type.
[24]	**INT01DTYPE**	**PWM01 Duty Interrupt Type Selection** 0 = PWMDIFx will be set if PWM counter down count and matches CMRx register. PWM will trigger ADC to conversion when PWM counter down count and matches CMRx register if correlate PWM trigger enable bit (PWMxDTEN) is set to 1. 1 = PWMDIFx will be set when PWM counter up count and matches CMRx register. PWM will trigger ADC to conversion when PWM counter up count and matches CMRx register if correlate PWM trigger enable bit (PWMxDTEN) is set to 1. **Note:** Set INT01DTYPE to 1 only work when PWM operating in center aligned type.
[23:18]	**Reserved**	Reserved.
[17]	**INT23TYPE**	**PWM23 Interrupt Period Type Selection** 0 = PWMIFx will be set if PWM counter underflow. PWM will trigger ADC to conversion when PWM counter underflow if correlate PWM trigger enable bit (PWMxTEN) is set to 1. 1 = PWMIFx will be set if PWM counter matches CNRx register. PWM will trigger ADC to conversion when PWM counter matches CNRx register if correlate PWM trigger enable bit (PWMxTEN) is set to 1. **Note:** Setting INT23TYPE to 1 only works when PWM operating is in center-aligned

		type.
[16]	**INT01TYPE**	**PWM01 Interrupt Period Type Selection** 0 = PWMIFx will be set if PWM counter underflow. PWM will trigger ADC to conversion when PWM counter underflow if correlate PWM trigger enable bit (PWMxTEN) is set to 1. 1 = PWMIFx will be set if PWM counter matches CNRx register. PWM will trigger ADC to conversion when PWM counter matches CNRx register if correlate PWM trigger enable bit (PWMxTEN) is set to 1. **Note:** Setting INT01TYPE to 1 only works when PWM operating is in center-aligned type.
[15:12]	**Reserved**	Reserved.
[11]	**PWMDIE3**	**PWM channel 3 Duty Interrupt Enable Control** 0 = PWM channel 3 duty interrupt Disabled. 1 = PWM channel 3 duty interrupt Enabled.
[10]	**PWMDIE2**	**PWM channel 2 Duty Interrupt Enable Control** 0 = PWM channel 2 duty interrupt Disabled. 1 = PWM channel 2 duty interrupt Enabled.
[9]	**PWMDIE1**	**PWM channel 1 Duty Interrupt Enable Control** 0 = PWM channel 1 duty interrupt Disabled. 1 = PWM channel 1 duty interrupt Enabled.
[8]	**PWMDIE0**	**PWM channel 0 Duty Interrupt Enable Control** 0 = PWM channel 0 duty interrupt Disabled. 1 = PWM channel 0 duty interrupt Enabled.
[3]	**PWMIE3**	**PWM channel 3 Period Interrupt Enable Control** 0 = PWM channel 3 period interrupt Disabled. 1 = PWM channel 3 period interrupt Enabled.
[2]	**PWMIE2**	**PWM channel 2 Period Interrupt Enable Control** 0 = PWM channel 2 period interrupt Disabled. 1 = PWM channel 2 period interrupt Enabled.
[1]	**PWMIE1**	**PWM channel 1 Period Interrupt Enable Control** 0 = PWM channel 1 period interrupt Disabled. 1 = PWM channel 1 period interrupt Enabled.
[0]	**PWMIE0**	**PWM channel 0 Period Interrupt Enable Control** 0 = PWM channel 0 period interrupt Disabled. 1 = PWM channel 0 period interrupt Enabled.

PWM Interrupt Indication Register (PIIR)

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

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**PWMDIF3**	**PWM Channel 3 Duty Interrupt Status** When INTDTYPE23 = 0, this bit is set by hardware when channel 3 PWM counter equals CMR3 on down-count. When INTDTYPE23 = 1, this bit is set by hardware when channel 3 PWM counter equals CMR3 on up-count. **Note:** Write 1 to clear this bit to 0.
[10]	**PWMDIF2**	**PWM channel 2 Duty Interrupt Status** When INTDTYPE23 = 0, this bit is set by hardware when channel 2 PWM counter equals CMR2 on down-count. When INTDTYPE23 = 1, this bit is set by hardware when channel 2 PWM counter equals CMR2 on up-count. **Note:** Write 1 to clear this bit to 0.
[9]	**PWMDIF1**	**PWM Channel 1 Duty Interrupt Status** When INTDTYPE01 = 0, this bit is set by hardware when channel 1 PWM counter equals CMR1 on down-count. When INTDTYPE01 = 1, this bit is set by hardware when channel 1 PWM counter equals CMR1 on up-count. **Note:** Write 1 to clear this bit to 0.
[8]	**PWMDIF0**	**PWM Channel 0 Duty Interrupt Status** When INTDTYPE01 = 0, this bit is set by hardware when channel 0 PWM counter equals CMR0 on down-count. When INTDTYPE01 = 1, this bit is set by hardware when channel 0 PWM counter equals CMR0 on up-count. **Note:** Write 1 to clear this bit to 0.

[7:4]	**Reserved**	Reserved.
[3]	**PWMIF3**	**PWM Channel 3 Period Interrupt Status** When INTTYPE23 = 0, this bit is set by hardware when channel 3 PWM counter equals 0. When INTTYPE23 = 1, this bit is set by hardware when channel 3 PWM counter equals CNR3. **Note:** Write 1 to clear this bit to 0.
[2]	**PWMIF2**	**PWM Channel 2 Period Interrupt Status** When INTTYPE23 = 0, this bit is set by hardware when channel 2 PWM counter equals 0. When INTTYPE23 = 1, this bit is set by hardware when channel 2 PWM counter equals CNR2. **Note:** Write 1 to clear this bit to 0.
[1]	**PWMIF1**	**PWM channel 1 Period Interrupt Status** When INTTYPE01 = 0, this bit is set by hardware when channel 1 PWM counter equals 0. When INTTYPE01 = 1, this bit is set by hardware when channel 1 PWM counter equals CNR1. **Note:** Write 1 to clear this bit to 0.
[0]	**PWMIF0**	**PWM Channel 0 Period Interrupt Status** When INTTYPE01 = 0, this bit is set by hardware when channel 0 PWM counter equals 0. When INTTYPE01 = 1, this bit is set by hardware when channel 0 PWM counter equals CNR0. **Note:** Write 1 to clear this bit to 0.

Capture Control Register (CCR0)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**CFLRI1**	**CFLR1 Latched Indicator** When PWM group input channel 1 has a falling transition, CFLR1 was latched with the value of PWM down-counter and this bit is set by hardware. **Note:** Write 1 to clear this bit to 0.
[22]	**CRLRI1**	**CRLR1 Latched Indicator** When PWM group input channel 1 has a rising transition, CRLR1 was latched with the value of PWM down-counter and this bit is set by hardware. **Note:** Write 1 to clear this bit to 0.
[21]	**Reserved**	Reserved.
[20]	**CAPIF1**	**Channel 1 Capture Interrupt Indication Flag** If PWM group channel 1 rising latch interrupt is enabled (CRL_IE1 = 1), a rising transition occurs at PWM group channel 1 will result in CAPIF1 to high; Similarly, a falling transition will cause CAPIF1 to be set high if PWM group channel 1 falling latch interrupt is enabled (CFL_IE1 = 1). **Note:** Write 1 to clear this bit to 0.
[19]	**CAPCH1EN**	**Channel 1 Capture Function Enable Control** 0 = Capture function on PWM group channel 1 Disabled. 1 = Capture function on PWM group channel 1 Enabled. **Note1:** When Enabled, Capture latched the PWM-counter and saved to CRLR (Rising latch) and CFLR (Falling latch). **Note2:** When Disabled, Capture does not update CRLR and CFLR, and disable PWM group channel 1 Interrupt.
[18]	**CFL_IE1**	**Channel 1 Falling Latch Interrupt Enable Control**

		0 = Falling latch interrupt Disabled. 1 = Falling latch interrupt Enabled. **Note:** When Enabled, if Capture detects PWM group channel 1 has falling transition, Capture will issue an Interrupt.
[17]	**CRL_IE1**	**Channel 1 Rising Latch Interrupt Enable Control** 0 = Rising latch interrupt Disabled. 1 = Rising latch interrupt Enabled. **Note:** When Enabled, if Capture detects PWM group channel 1 has rising transition, Capture will issue an Interrupt.
[16]	**INV1**	**Capture Channel 1 Inverter Enable Control** 0 = Inverter Disabled. 1 = Inverter Enabled. Reverse the input signal from GPIO before fed to Capture timer.
[15:8]	**Reserved**	Reserved.
[7]	**CFLRI0**	**CFLR0 Latched Indicator Control** When PWM group input channel 0 has a falling transition, CFLR0 was latched with the value of PWM down-counter and this bit is set by hardware. **Note:** Write 1 to clear this bit to 0.
[6]	**CRLRI0**	**CRLR0 Latched Indicator** When PWM group input channel 0 has a rising transition, CRLR0 was latched with the value of PWM down-counter and this bit is set by hardware. **Note:** Write 1 to clear this bit to 0.
[5]	**Reserved**	Reserved.
[4]	**CAPIF0**	**Channel 0 Capture Interrupt Indication Flag** If PWM group channel 0 rising latch interrupt is enabled (CRL_IE0 = 1), a rising transition occurs at PWM group channel 0 will result in CAPIF0 to high; Similarly, a falling transition will cause CAPIF0 to be set high if PWM group channel 0 falling latch interrupt is enabled (CFL_IE0 = 1). **Note:** Write 1 to clear this bit to 0.
[3]	**CAPCH0EN**	**Channel 0 Capture Function Enable Control** 0 = Capture function on PWM group channel 0 Disabled. 1 = Capture function on PWM group channel 0 Enabled. **Note1:** When Enabled, Capture latches the PWM-counter value and saved to CRLR (Rising latch) and CFLR (Falling latch). **Note2:** When Disabled, Capture does not update CRLR and CFLR, and disable PWM group channel 0 Interrupt.
[2]	**CFL_IE0**	**Channel 0 Falling Latch Interrupt Enable Control** 0 = Falling latch interrupt Disabled. 1 = Falling latch interrupt Enabled. **Note:** When Enabled, if Capture detects PWM group channel 0 has falling transition, and Capture will issue an Interrupt.
[1]	**CRL_IE0**	**Channel 0 Rising Latch Interrupt Enable Control** 0 = Rising latch interrupt Disabled. 1 = Rising latch interrupt Enabled.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

		Note: When Enabled, if Capture detects PWM group channel 0 has rising transition, Capture will issue an Interrupt.
[0]	**INV0**	**Capture Channel 0 Inverter Enable Control** 0 = Inverter Disabled. 1 = Inverter Enabled. Reverse the input signal from GPIO before fed to Capture timer.

Capture Control Register (CCR2)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**CFLRI3**	**CFLR3 Latched Indicator** When PWM group input channel 3 has a falling transition, CFLR3 was latched with the value of PWM down-counter and this bit is set by hardware. **Note:** Write 1 to clear this bit to 0.
[22]	**CRLRI3**	**CRLR3 Latched Indicator** When PWM group input channel 3 has a rising transition, CRLR3 was latched with the value of PWM down-counter and this bit is set by hardware. **Note:** Write 1 to clear this bit to 0.
[21]	**Reserved**	Reserved.
[20]	**CAPIF3**	**Channel 3 Capture Interrupt Indication Flag** If PWM group channel 3 rising latch interrupt is enabled (CRL_IE3=1), a rising transition occurs at PWM group channel 3 will result in CAPIF3 to high; Similarly, a falling transition will cause CAPIF3 to be set high if PWM group channel 3 falling latch interrupt is enabled (CFL_IE3=1). **Note:** Write 1 to clear this bit to 0.
[19]	**CAPCH3EN**	**Channel 3 Capture Function Enable Control** 0 = Capture function on PWM group channel 3 Disabled. 1 = Capture function on PWM group channel 3 Enabled. **Note1:** When Enabled, Capture latched the PWM-counter and saved to CRLR (Rising latch) and CFLR (Falling latch). **Note2:** When Disabled, Capture does not update CRLR and CFLR, and disable PWM group channel 3 Interrupt.

[18]	**CFL_IE3**	**Channel 3 Falling Latch Interrupt Enable Control** 0 = Falling latch interrupt Disabled. 1 = Falling latch interrupt Enabled. **Note:** When Enabled, if Capture detects PWM group channel 3 has falling transition, Capture will issue an Interrupt.
[17]	**CRL_IE3**	**Channel 3 Rising Latch Interrupt Enable Control** 0 = Rising latch interrupt Disabled. 1 = Rising latch interrupt Enabled. **Note:** When Enabled, if Capture detects PWM group channel 3 has rising transition, Capture will issue an Interrupt.
[16]	**INV3**	**Capture Channel 3 Inverter Enable Control** 0 = Inverter Disabled. 1 = Inverter Enabled. Reverse the input signal from GPIO before fed to Capture timer.
[15:8]	**Reserved**	Reserved.
[7]	**CFLRI2**	**CFLR2 Latched Indicator** When PWM group input channel 2 has a falling transition, CFLR2 was latched with the value of PWM down-counter and this bit is set by hardware. **Note:** Write 1 to clear this bit to 0.
[6]	**CRLRI2**	**CRLR2 Latched Indicator** When PWM group input channel 2 has a rising transition, CRLR2 was latched with the value of PWM down-counter and this bit is set by hardware. **Note:** Write 1 to clear this bit to 0.
[5]	**Reserved**	Reserved.
[4]	**CAPIF2**	**Channel 2 Capture Interrupt Indication Flag** If PWM group channel 2 rising latch interrupt is enabled (CRL_IE2=1), a rising transition occurs at PWM group channel 2 will result in CAPIF2 to high; Similarly, a falling transition will cause CAPIF2 to be set high if PWM group channel 2 falling latch interrupt is enabled (CFL_IE2=1). **Note:** Write 1 to clear this bit to 0.
[3]	**CAPCH2EN**	**Channel 2 Capture Function Enable Control** 1 = Capture function on PWM group channel 2 Enabled. 0 = Capture function on PWM group channel 2 Disabled. **Note1:** When Enabled, Capture latched the PWM-counter value and saved to CRLR (Rising latch) and CFLR (Falling latch). **Note2:** When Disabled, Capture does not update CRLR and CFLR, and disable PWM group channel 2 Interrupt.
[2]	**CFL_IE2**	**Channel 2 Falling Latch Interrupt Enable Control** 0 = Falling latch interrupt Disabled. 1 = Falling latch interrupt Enabled. **Note:** When Enabled, if Capture detects PWM group channel 2 has falling transition, Capture will issue an Interrupt.
[1]	**CRL_IE2**	**Channel 2 Rising Latch Interrupt Enable Control** 0 = Rising latch interrupt Disabled.

		1 = Rising latch interrupt Enabled. **Note:** When Enabled, if Capture detects PWM group channel 2 has rising transition, Capture will issue an Interrupt.
[0]	**INV2**	**Capture Channel 2 Inverter Enable Control** 0 = Inverter Disabled. 1 = Inverter Enabled. Reverse the input signal from GPIO before fed to Capture timer.

Capture Rising Latch Register3-0 (CRLR3-0)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CRLRx**	**Capture Rising Latch Register** Latch the PWM counter value 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 Capture Falling Latch Register (Channel 0)	0x0000_0000
CFLR1	PWMA_BA+0x64	R	PWM Capture Falling Latch Register (Channel 1)	0x0000_0000
CFLR2	PWMA_BA+0x6C	R	PWM Capture Falling Latch Register (Channel 2)	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM Capture Falling Latch Register (Channel 3)	0x0000_0000

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

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

Capture Input Enable Register (CAPENR)

Register	Offset	R/W	Description	Reset Value
CAPENR	PWMA_BA+0x78	R/W	PWM 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	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**CAPENR**	**Capture Input Enable Register** 0 = Capture input Disabled. (PWMx multi-function pin input does not affect input capture function.) 1 = Capture input Enabled. (PWMx multi-function pin input will affect its input capture function.) CAPENR **Bit 3210 for PWM group A** Bit xxx1 → Capture channel 0 enabled. Capture input channel can be from P2.0 or P4.0. User can only select one of pins by setting multi-function pin register. Bit xx1x → Capture channel 1 enabled. Capture input channel can be from P2.1 or P4.1. User can only select one of pins by setting multi-function pin register. Bit x1xx → Capture channel 2 enabled. Capture input channel can be from P2.2 or P4.2. User can only select one of pins by setting multi-function pin register. Bit 1xxx → Capture channel 3 enabled. Capture input channel can be from P2.3 or P4.3. User can only select one of pins by setting multi-function pin register.

PWM Output Enable Register (POE)

Register	Offset	R/W	Description	Reset Value
POE	PWMA_BA+0x7C	R/W	PWM 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	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PWM3**	**Channel 3 Output Enable Register** 0 = PWM channel 3 output to pin Disabled. 1 = PWM channel 3 output to pin Enabled. **Note:** The corresponding GPIO pin must also be switched to PWM function.
[2]	**PWM2**	**Channel 2 Output Enable Register** 0 = PWM channel 2 output to pin Disabled. 1 = PWM channel 2 output to pin Enabled. **Note:** The corresponding GPIO pin must also be switched to PWM function.
[1]	**PWM1**	**Channel 1 Output Enable Register** 0 = PWM channel 1 output to pin Disabled. 1 = PWM channel 1 output to pin Enabled. **Note:** The corresponding GPIO pin must also be switched to PWM function.
[0]	**PWM0**	**Channel 0 Output Enable Register** 0 = PWM channel 0 output to pin Disabled. 1 = PWM channel 0 output to pin Enabled. **Note:** The corresponding GPIO pin must also be switched to PWM function.

PWM Trigger Control Register (TCON)

Register	Offset	R/W	Description	Reset Value
TCON	PWMA_BA+0x80	R/W	PWM Trigger Control 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				PWM3DTEN	PWM2DTEN	PWM1DTEN	PWM0DTEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				PWM3TEN	PWM2TEN	PWM1TEN	PWM0TEN

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**PWM3DTEN**	**Channel 3 PWM Duty Trigger ADC Enable Control** 0 = PWM channel 3 duty trigger ADC function Disabled. 1 = PWM channel 3 duty trigger ADC function Enabled. As PWM operating at edge-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 3 counter down count to match CMR3. As PWM operating at center-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 3 counter up count or down count to match CMR3 based on INT23DTYPE setting.
[10]	**PWM2DTEN**	**Channel 2 PWM Duty Trigger ADC Enable Control** 0 = PWM channel 2 duty trigger ADC function Disabled. 1 = PWM channel 2 duty trigger ADC function Enabled. As PWM operating at edge-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 2 counter down count to match CMR2. As PWM operating at center-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 2 counter up count or down count to match CMR2 based on INT23DTYPE setting.
[9]	**PWM1DTEN**	**Channel 1 PWM Duty Trigger ADC Enable Control** 0 = PWM channel 1 duty trigger ADC function Disabled. 1 = PWM channel 1 duty trigger ADC function Enabled. As PWM operating at edge-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 1 counter down count to match CMR1. As PWM operating at center-aligned type, enable this bit can make PWM trigger

		ADC to start conversion when PWM channel 1 counter up count or down count to match CMR1 based on INT01DTYPE setting.
[8]	**PWM0DTEN**	**Channel 0 PWM Duty Trigger ADC Enable Control** 0 = PWM channel 0 duty trigger ADC function Disabled. 1 = PWM channel 0 duty trigger ADC function Enabled. As PWM operating at edge-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 0 counter down count to match CMR0. As PWM operating at center-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 0 counter up count or down count to match CMR0 based on INT01DTYPE setting.
[7:4]	**Reserved**	Reserved.
[3]	**PWM3TEN**	**Channel 3 PWM Period Trigger ADC Enable Control** 0 = PWM channel 3 period trigger ADC function Disabled. 1 = PWM channel 3 period trigger ADC function Enabled. As PWM operating at edge-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 3 counter down count to underflow. As PWM operating at center-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 3 counter up count to (CNR3 + 1) or down count to underflow based on INT23PTYPE setting.
[2]	**PWM2TEN**	**Channel 2 PWM Period Trigger ADC Enable Control** 0 = PWM channel 2 period trigger ADC function Disabled. 1 = PWM channel 2 period trigger ADC function Enabled. As PWM operating at edge-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 2 counter down count to underflow. As PWM operating at center-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 2 counter up count to (CNR2 + 1) or down count to underflow based on INT23PTYPE setting.
[1]	**PWM1TEN**	**Channel 1 PWM Period Trigger ADC Enable Control** 0 = PWM channel 1 period trigger ADC function Disabled. 1 = PWM channel 1 period trigger ADC function Enabled. As PWM operating at edge-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 1 counter down count to underflow. As PWM operating at center-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 1 counter up count to (CNR1 + 1) or down count to underflow based on INT01PTYPE setting.
[0]	**PWM0TEN**	**Channel 0 PWM Period Trigger ADC Enable Control** 0 = PWM channel 0 period trigger ADC function Disabled. 1 = PWM channel 0 period trigger ADC function Enabled. As PWM operating at edge-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 0 counter down count to underflow. As PWM operating at center-aligned type, enable this bit can make PWM trigger ADC to start conversion when PWM channel 0 counter up count to (CNR0 + 1) or down count to underflow based on INT01PTYPE setting.

PWM Trigger Status Register (TSTATUS)

Register	Offset	R/W	Description	Reset Value
TSTATUS	PWMA_BA+0x84	R/W	PWM Trigger Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PWM3TF**	**PWM2TF**	**PWM1TF**	**PWM0TF**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PWM3TF**	**Channel 3 PWM Trigger ADC Flag** This bit is set to 1 by hardware when PWM channel 3 trigger ADC condition matched. ADC will start conversion if ADC triggered source is selected by PWM while this bit is set to 1. **Note:** Write 1 to clear this bit to 0.
[2]	**PWM2TF**	**Channel 2 PWM Trigger ADC Flag** This bit is set to 1 by hardware when PWM channel 2 trigger ADC condition matched. ADC will start conversion if ADC triggered source is selected by PWM while this bit is set to 1. **Note:** Write 1 to clear this bit to 0.
[1]	**PWM1TF**	**Channel 1 PWM Trigger ADC Flag** This bit is set to 1 by hardware when PWM channel 1 trigger ADC condition matched. ADC will start conversion if ADC triggered source is selected by PWM while this bit is set to 1. **Note:** Write 1 to clear this bit to 0.
[0]	**PWM0TF**	**Channel 0 PWM Trigger ADC Flag** This bit is set to 1 by hardware when PWM channel 0 trigger ADC condition matched. ADC will start conversion if ADC triggered source is selected by PWM while this bit is set to 1. **Note:** Write 1 to clear this bit to 0.

6.8 Watchdog Timer (WDT)

6.8.1 Overview

The purpose of Watchdog Timer is to perform a system reset when system runs into an unknown state. This prevents system from hanging for an infinite period of time. Besides, this Watchdog Timer supports the function to wake-up system from Idle/Power-down mode.

6.8.2 Features

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

6.8.3 Block Diagram

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

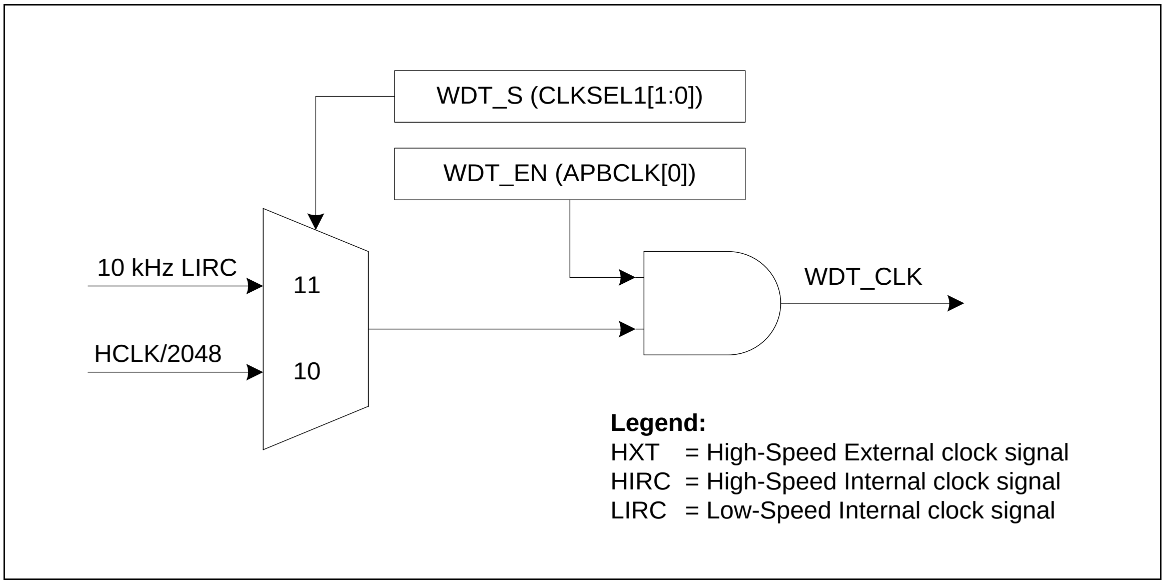
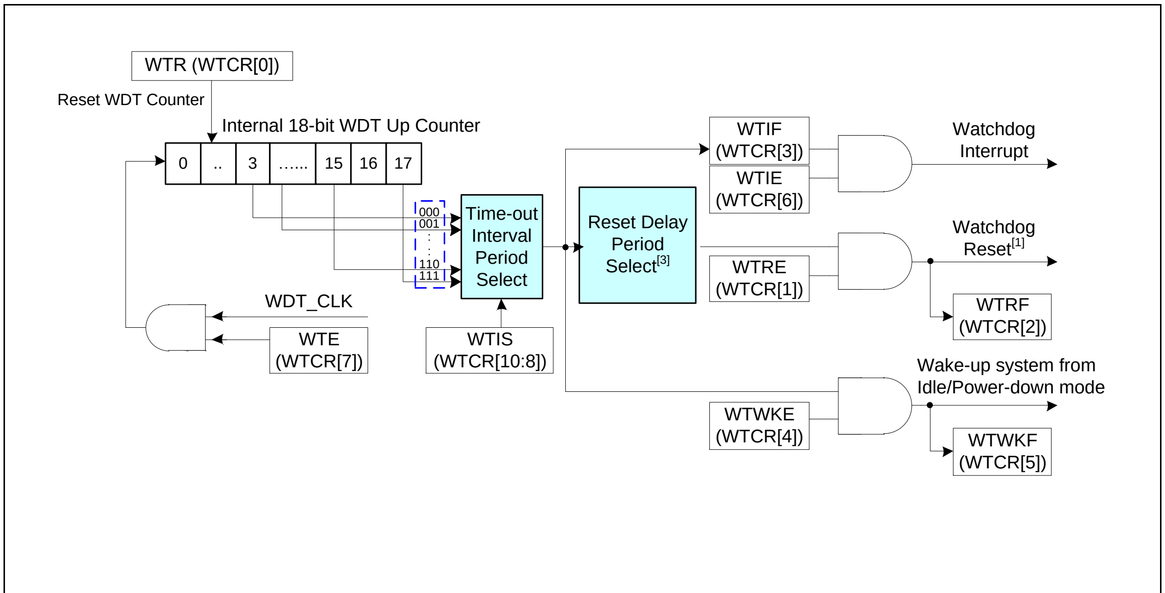

WDT_S (CLKSEL1[1:0])
WDT_EN (APBCLK[0])
10 kHz LIRC
HCLK/2048
11
10
WDT_CLK
Legend:
HXT = High-Speed External clock signal
HIRC = High-Speed Internal clock signal
LIRC = Low-Speed Internal clock signal

Figure 6.8-1 Watchdog Timer Clock Control



Note1: WDT resets CPU and lasts 63 WDT_CLK.

Note2: If user intends to use WDT to wake-up Idle/Power-down mode, it is recommended that CPU clock source is set the same as WDT clock source before CPU enters Power-Down mode.

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

Figure 6.8-2 Watchdog Timer Block Diagram

6.8.4 Basic Configuration

The WDT peripheral clock is enabled in APBCLK[0] and clock source can be selected in CLKSEL1[1:0].

Or user can setting CONFIG[31] 0 to force Watchdog Timer enabled and active in 10 kHz after chip powered on or reset.

6.8.5 Functional Description

The Watchdog Timer (WDT) includes an 18-bit free running up counter with programmable time-out intervals. Table 6-21 shows the WDT time-out interval period selection and Figure 6.8-3 shows the WDT time-out interval and reset period timing.

WDT Time-out Interrupt

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

WDT Reset Delay Period and Reset System

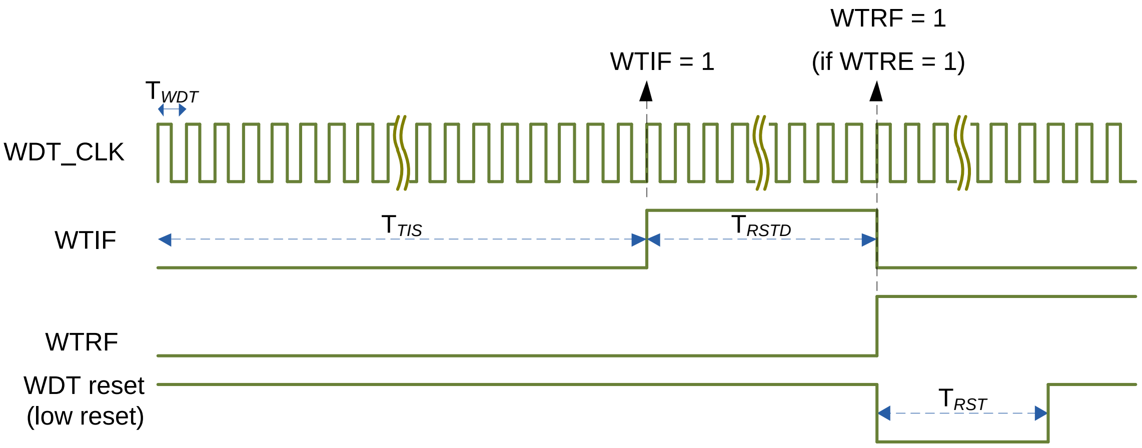
There is a specified T_{RSTD} delay period follows the WTIF flag is setting to 1. User should set WTR bit to reset the 18-bit WDT up counter value to avoid generate WDT time-out reset signal before the T_{RSTD} delay period expires. If the WDT up counter value has not been cleared after the specific T_{RSTD} delay period expires, the WDT control will set WTRF flag to 1 if WTRE bit is enabled, then chip enters to reset state immediately. Refer to Figure 6.8-3, the T_{RST} reset period will keep last 63 WDT clocks then chip restart executing program from reset vector (0x0000_0000). The WTRF flag will keep 1 after WDT time-out reset the chip, user can check WTRF flag by software to recognize the system has been reset by WDT time-out reset or not.

WDT Wake-up

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

WTIS	Time-out Interval Period T_{TIS}	Reset Delay Period T_{RSTD}
000	$2^4 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
001	$2^6 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
010	$2^8 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
011	$2^{10} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
100	$2^{12} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
101	$2^{14} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
110	$2^{16} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
111	$2^{18} * T_{WDT}$	(3/18/130/1026) * T_{WDT}

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



- *T_{WDT} : Watchdog Clock Time Period*
- *T_{TIS} : Watchdog Time-out Interval Period ((2^4 ~ 2^{18}) * T_{WDT})*
- *T_{RSTD} : Watchdog Reset Delay Period*
  - *Fixed 1024 * T_{WDT} delay period on M05xxBN*
  - *Selectable 3/18/130/1026 * T_{WDT} delay period controlled by WTRDSEL on M05xxDE*
- *T_{RST} : Watchdog Reset Period (63 * T_{WDT})*

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

6.8.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
WDT Base Address: **WDT_BA = 0x4000_4000**				
WTCR	WDT_BA+0x00	R/W	Watchdog Timer Control Register	0x0000_0700
WTCRALT	WDT_BA+0x04	R/W	Watchdog Timer Alternative Control Register	0x0000_0000

6.8.7 Register Description

Watchdog Timer Control Register (WTCR)

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

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

Bits	Description	
[31]	**DBGACK_WDT**	**ICE Debug Mode Acknowledge Disable Control (Write Protect)** 0 = ICE debug mode acknowledgement effects WDT counting. WDT up counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. WDT up counter will keep going no matter CPU is held by ICE or not.
[30:11]	**Reserved**	Reserved.
[10:8]	**WTIS**	**Watchdog Timer Time-out Interval Selection (Write Protect)** These three bits select the time-out interval period for the WDT. 000 = 2^{4} *T_{WDT}. 001 = 2^{6} *T_{WDT}. 010 = 2^{8} *T_{WDT}. 011 = 2^{10} *T_{WDT}. 100 = 2^{12} *T_{WDT}. 101 = 2^{14} *T_{WDT}. 110 = 2^{16} *T_{WDT}. 111 = 2^{18} *T_{WDT}.
[7]	**WTE**	**Watchdog Timer Enable Control (Write Protect)** 0 = WDT Disabled. (This action will reset the internal up counter value.) 1 = WDT Enabled.
[6]	**WTIE**	**Watchdog Timer Time-out Interrupt Enable Control (Write Protect)** If this bit is enabled, the WDT time-out interrupt signal is generated and inform to CPU. 0 = WDT time-out interrupt Disabled.

		1 = WDT time-out interrupt Enabled.
[5]	**WTWKF**	**Watchdog Timer Time-out Wake-up Flag** This bit indicates the interrupt wake-up flag status of WDT. 0 = WDT does not cause chip wake-up. 1 = Chip wake-up from Idle or Power-down mode if WDT time-out interrupt signal generated. **Note:** This bit is cleared by writing 1 to it.
[4]	**WTWKE**	**Watchdog Timer Time-out Wake-up Function Control (Write Protect)** If this bit is set to 1, while WTIF is generated to 1 and WTIE enabled, the WDT time-out interrupt signal will generate a wake-up trigger event to chip. 0 = Wake-up trigger event Disabled if WDT time-out interrupt signal generated. 1 = Wake-up trigger event Enabled if WDT time-out interrupt signal generated. **Note:** Chip can be woken-up by WDT time-out interrupt signal generated only if WDT clock source is selected to 10 kHz oscillator.
[3]	**WTIF**	**Watchdog Timer Time-out Interrupt Flag** This bit will set to 1 while WDT up counter value reaches the selected WDT time-out interval. 0 = WDT time-out interrupt did not occur. 1 = WDT time-out interrupt occurred. **Note:** This bit is cleared by writing 1 to it.
[2]	**WTRF**	**Watchdog Timer Time-out Reset Flag** This bit indicates the system has been reset by WDT time-out reset or not. 0 = WDT time-out reset did not occur. 1 = WDT time-out reset occurred. **Note:** This bit is cleared by writing 1 to it.
[1]	**WTRE**	**Watchdog Timer Time-out Reset Enable Control (Write Protect)** Setting this bit will enable the WDT time-out reset function If the WDT up counter value has not been cleared after the specific WDT reset delay period expires. 0 = WDT time-out reset function Disabled. 1 = WDT time-out reset function Enabled. **Note:** Selectable 3/18/130/1026 * T_{WDT} delay period controlled by WTRDSEL
[0]	**WTR**	**Reset Watchdog Timer Up Counter (Write Protect)** 0 = No effect. 1 = Reset the internal 18-bit WDT up counter value. **Note:** This bit will be automatically cleared by hardware.

Watchdog Timer Alternative Control Register (WTCRALT)

Register	Offset	R/W	Description	Reset Value
WTCRALT	WDT_BA+0x04	R/W	Watchdog Timer Alternative Control Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1:0]	**WTRDSEL**	**Watchdog Timer Reset Delay Selection (Write Protect)** When WDT time-out happened, software has a time named WDT Reset Delay Period to clear WDT counter to prevent WDT time-out reset happened. Software can select a suitable value of WDT Reset Delay Period for different WDT time-out period. 00 = Watchdog Timer Reset Delay Period is 1026 * WDT_CLK. 01 = Watchdog Timer Reset Delay Period is 130* WDT_CLK. 10 = Watchdog Timer Reset Delay Period is 18 * WDT_CLK. 11 = Watchdog Timer Reset Delay Period is 3 * WDT_CLK. **Note:** This register will be reset to 0 if WDT time-out reset happened.

6.9 Window Watchdog Timer (WWDT)

6.9.1 Overview

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

6.9.2 Features

- 6-bit down counter value (WWDTCVAL) and 6-bit compare window value (WINCMP) to make the WWDT time-out window period flexible
- Supports 4-bit value to programmable maximum 11-bit prescale counter period of WWDT counter

6.9.3 Block Diagram

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

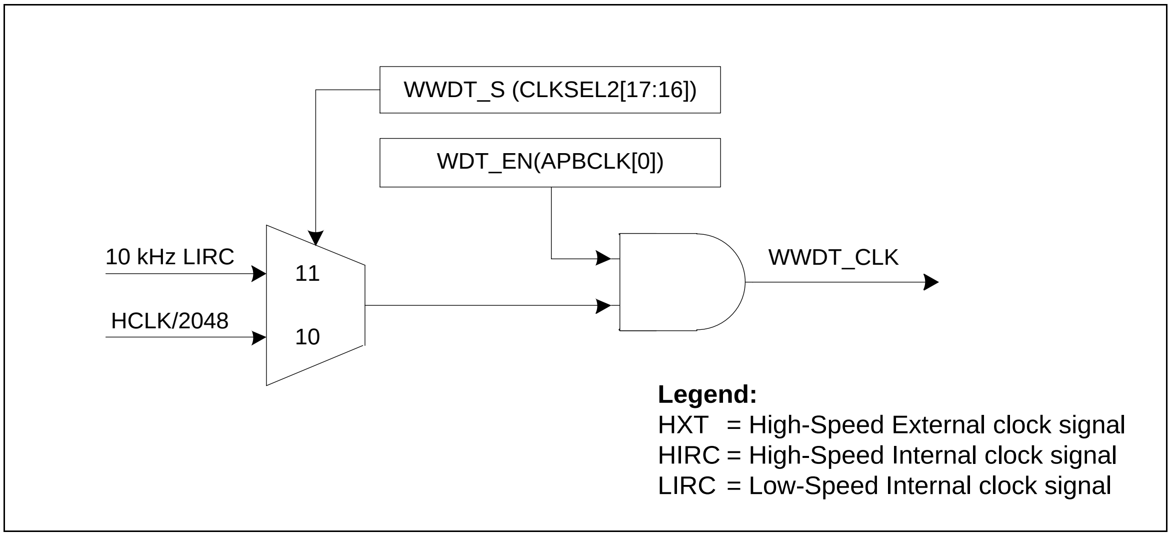

WWDT_S (CLKSEL2[17:16])
WDT_EN(APBCLK[0])
10 kHz LIRC
HCLK/2048
11
10
WWDT_CLK
Legend:
HXT = High-Speed External clock signal
HIRC = High-Speed Internal clock signal
LIRC = Low-Speed Internal clock signal

Figure 6.9-1 Window Watchdog Timer Clock Control

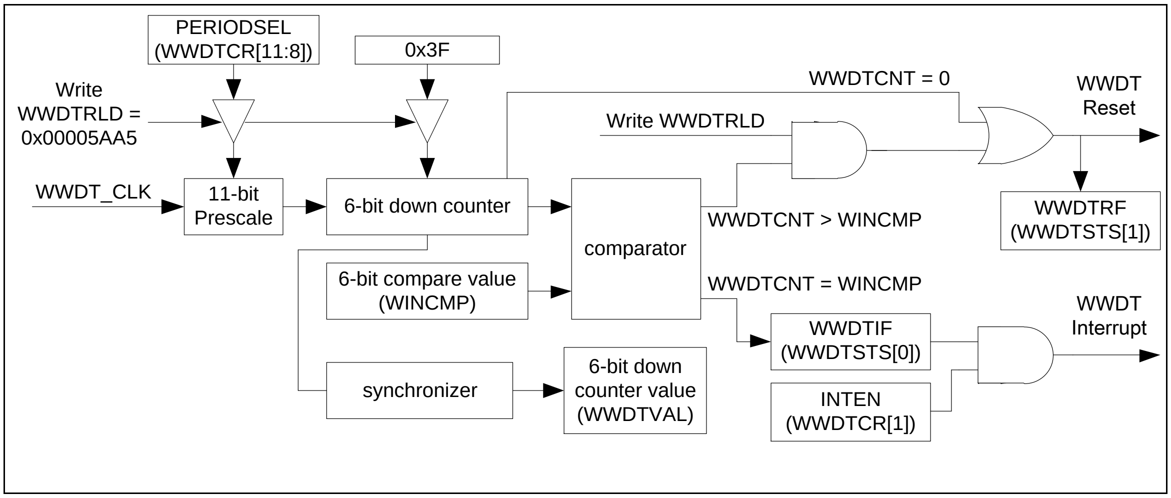


Figure 6.9-2 Window Watchdog Timer Block Diagram

6.9.4 Basic Configuration

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

6.9.5 Functional Description

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

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

Table 6.9-1 Window Watchdog Timer Prescale Value Selection

WWDT Counting

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

WWDT Compare Match Interrupt

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

WWDT Reset System

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

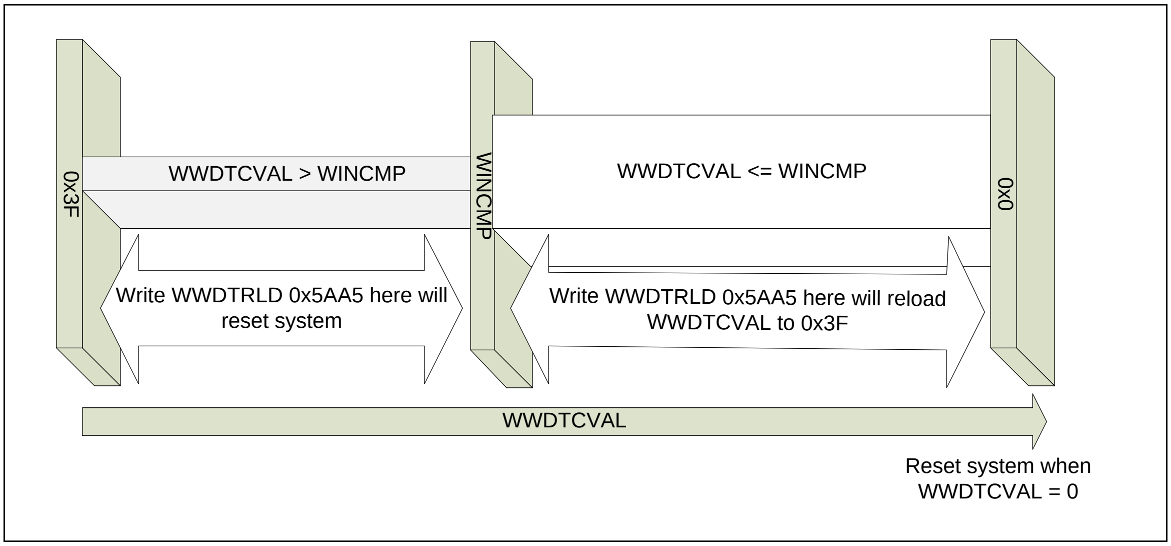

0x3F
WWDTCVAL > WINCMP
WINCMP
WWDTCVAL <= WINCMP
0x0
Write WWDTRLD 0x5AA5 here will reset system
Write WWDTRLD 0x5AA5 here will reload WWDTCVAL to 0x3F
WWDTCVAL
Reset system when WWDTCVAL = 0

Figure 6.9-3 Window Watchdog Timer Reset and Reload Behavior

WWDT Window Setting Limitation

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

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

Table 6-9 WINCMP Setting Limitation

6.9.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
WWDT Base Address: **WWDT_BA = 0x4000_4100**				
WWDTRLD	WWDT_BA+0x00	W	Window Watchdog Timer Reload Counter Register	0x0000_0000
WWDTCR	WWDT_BA+0x04	R/W	Window Watchdog Timer Control Register	0x003F_0800
WWDTSR	WWDT_BA+0x08	R/W	Window Watchdog Timer Status Register	0x0000_0000
WWDTCVR	WWDT_BA+0x0C	R	Window Watchdog Timer Counter Value Register	0x0000_003F

6.9.7 Register Description

Window Watchdog Timer Reload Counter Register (WWDTRLD)

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

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

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

Window Watchdog Timer Control Register (WWDTCR)

Register	Offset	R/W	Description	Reset Value
WWDTCR	WWDT_BA+0x04	R/W	Window Watchdog Timer Control Register	0x003F_0800

Note: This register can be written only one time after chip is powered on or reset.

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

Bits	Description	
[31]	**DBGACK_WWDT**	**ICE Debug Mode Acknowledge Disable Control** 0 = ICE debug mode acknowledgement effects WWDT counting. WWDT down counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. WWDT down counter will keep going no matter CPU is held by ICE or not.
[30:22]	**Reserved**	Reserved.
[21:16]	**WINCMP**	**WWDT Window Compare Register** Set this register to adjust the valid reload window. **Note:** Software can only write WWDTRLD to reload WWDT counter value when current WWDT counter value between 0 and WINCMP. If Software writes WWDTRLD when current WWDT counter value larger than WINCMP, WWDT reset signal will generate immediately.
[15:12]	**Reserved**	Reserved.
[11:8]	**PERIODSEL**	**WWDT Counter Prescale Period Selection** 0000 = Pre-scale is 1; Max time-out period is 1 * 64 * T_{WWDT}. 0001 = Pre-scale is 2; Max time-out period is 2 * 64 * T_{WWDT}. 0010 = Pre-scale is 4; Max time-out period is 4 * 64 * T_{WWDT}. 0011 = Pre-scale is 8; Max time-out period is 8 * 64 * T_{WWDT}. 0100 = Pre-scale is 16; Max time-out period is 16 * 64 * T_{WWDT}. 0101 = Pre-scale is 32; Max time-out period is 32 * 64 * T_{WWDT}. 0110 = Pre-scale is 64; Max time-out period is 64 * 64 * T_{WWDT}. 0111 = Pre-scale is 128; Max time-out period is 128 * 64 * T_{WWDT}. 1000 = Pre-scale is 192; Max time-out period is 192 * 64 * T_{WWDT}.

		1001 = Pre-scale is 256; Max time-out period is 256 * 64 * T_{WWDT}. 1010 = Pre-scale is 384; Max time-out period is 384 * 64 * T_{WWDT}. 1011 = Pre-scale is 512; Max time-out period is 512 * 64 * T_{WWDT}. 1100 = Pre-scale is 768; Max time-out period is 768 * 64 * T_{WWDT}. 1101 = Pre-scale is 1024; Max time-out period is 1024 * 64 * T_{WWDT}. 1110 = Pre-scale is 1536; Max time-out period is 1536 * 64 * T_{WWDT}. 1111 = Pre-scale is 2048; Max time-out period is 2048 * 64 * T_{WWDT}.
[7:2]	**Reserved**	Reserved.
[1]	**WWDTIE**	**WWDT Interrupt Enable Control** If this bit is enabled, the WWDT counter compare match interrupt signal is generated and inform to CPU. 0 = WWDT counter compare match interrupt Disabled. 1 = WWDT counter compare match interrupt Enabled.
[0]	**WWDTEN**	**WWDT Enable Control** 0 = WWDT counter is stopped. 1 = WWDT counter is starting counting.

Window Watchdog Timer Status Register (WWDTSR)

Register	Offset	R/W	Description	Reset Value
WWDTSR	WWDT_BA+0x08	R/W	Window Watchdog Timer Status Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WWDTRF**	**WWDT Timer-out Reset Flag** This bit indicates the system has been reset by WWDT time-out reset or not. 0 = WWDT time-out reset did not occur. 1 = WWDT time-out reset occurred. **Note:** This bit is cleared by writing 1 to it.
[0]	**WWDTIF**	**WWDT Compare Match Interrupt Flag** This bit indicates the interrupt flag status of WWDT while WWDT counter value matches WINCMP value. 0 = No effect. 1 = WWDT counter value matches WINCMP value. **Note:** This bit is cleared by writing 1 to it.

Window Watchdog Timer Counter Value Register (WWDTCVR)

Register	Offset	R/W	Description	Reset Value
WWDTCVR	WWDT_BA+0x0C	R	Window Watchdog Timer Counter Value Register	0x0000_003F

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

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

6.10 UART Controller (UART)

6.10.1 Overview

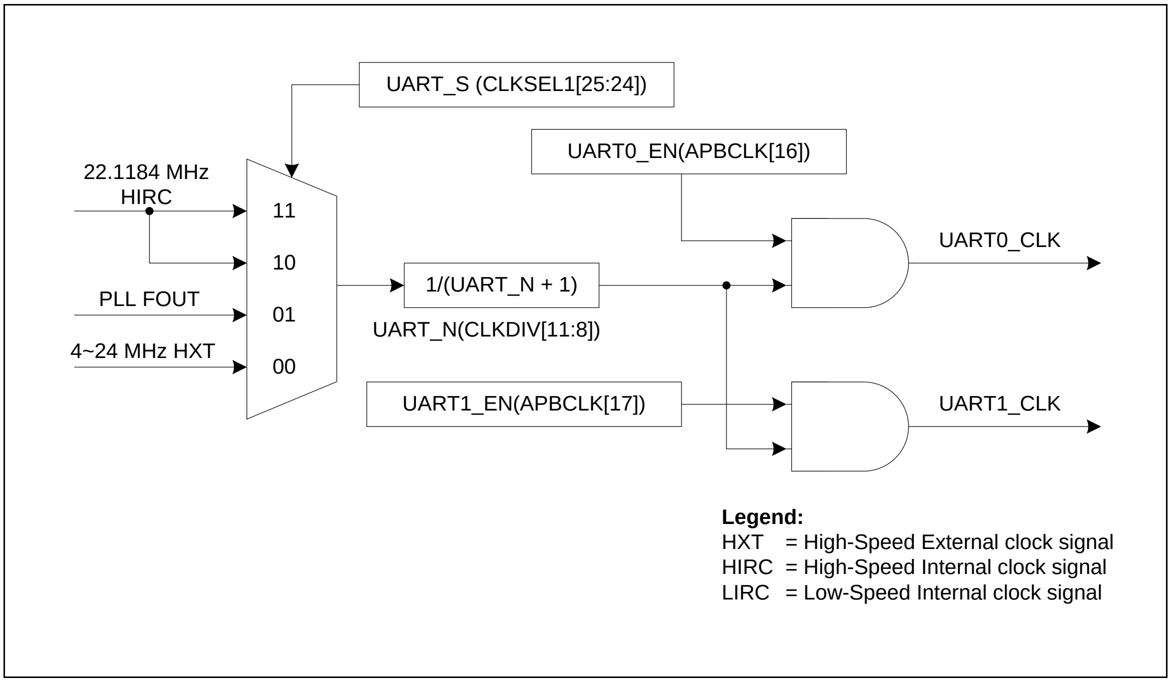
The NuMicro M058S provides two channels of Universal Asynchronous Receiver/Transmitters (UART). UART Controller performs Normal Speed UART, and support flow control function. The UART Controller performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the CPU. The UART controller also supports IrDA SIR Function, LIN master/slave function and RS-485 function mode. Each UART Controller channel supports seven types of interrupts.

6.10.2 Features

- Full-duplex, asynchronous communications
- Separate receive / transmit 16/16 bytes entry FIFO for data payloads
- Supports hardware auto-flow control and flow control function (CTS, RTS) and programmable RTS flow control trigger level
- Programmable receiver buffer trigger level
- Supports programmable baud-rate generator for each channel individually
- Supports CTS wake-up function
- Supports 8-bit receiver buffer time-out detection function
- Programmable transmitting data delay time between the last stop and the next start bit by setting DLY (UA_TOR [15:8]) register
- Supports break error, frame error, parity error and receive / transmit buffer overflow detect function
- Fully programmable serial-interface characteristics
  - Programmable number of data bit, 5-, 6-, 7-, 8- bit character
  - Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
  - Programmable stop bit, 1, 1.5, or 2 stop bit generation
- Supports IrDA SIR function mode
  - Supports for 3/16 bit duration for normal mode
- Supports LIN function mode
  - Supports LIN master/slave mode
  - Supports programmable break generation function for transmitter
  - Supports break detect function for receiver
- Supports RS-485 function mode
  - Supports RS-485 9-bit mode
  - Supports hardware or software enable to program RTS pin to control RS-485 transmission direction directly

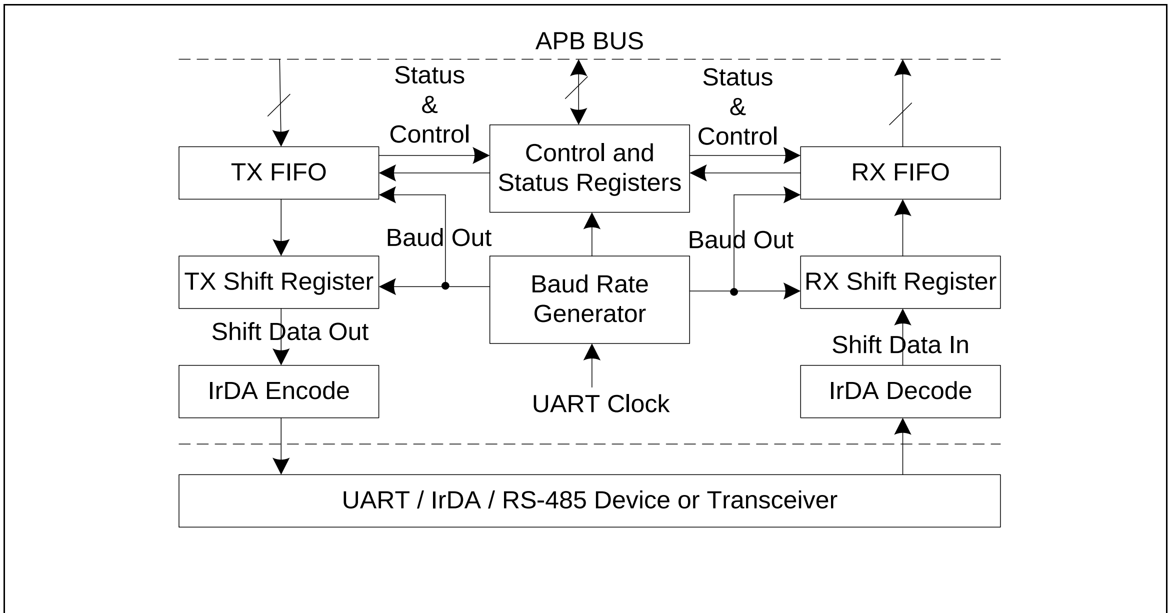
6.10.3 Block Diagram

The UART clock control and block diagram are shown below.



Legend:
HXT = High-Speed External clock signal
HIRC = High-Speed Internal clock signal
LIRC = Low-Speed Internal clock signal

Figure 6.10-1 UART Controller Clock Control



Note: UART0 is equipped with 16 bytes FIFO.

Figure 6.10-2 UART Controller Block Diagram

Each block is described in detail as follows:

TX_FIFO
The transmitter is buffered with a 16 bytes FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO
The receiver is buffered with a 16 bytes FIFO (plus three error bits 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(THRE_INT), receiver threshold level reaching interrupt (RDA_INT), line status interrupt (parity error or framing error or break interrupt) (RLS_INT), time-out interrupt (TOUT_INT), MODEM/Wake-up status interrupt (MODEM_INT), Buffer error interrupt (BUF_ERR_INT) and LIN receiver break field detected interrupt (LIN_RX_BREAK_INT).

In addition, the block diagram of auto-flow control is demonstrated in the following diagram.

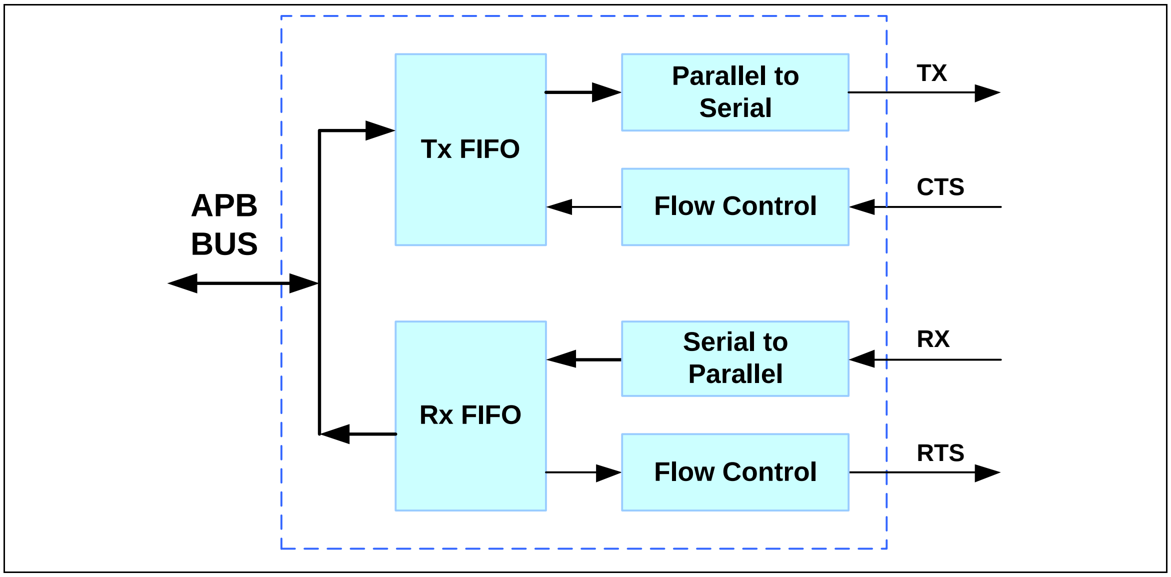


Figure 6.10-3 Auto Flow Control Block Diagram

6.10.4 Basic Configuration

The UART Controller function pins are configured in P0_MFP registers for UART.

The UART Controller clock are enabled in UART0_EN(APBCLK[16]) for UART0.

The UART Controller clock source is selected by UART_S(CLKSEL1[25:24]).

The UART Controller clock prescaler is determined by UART_N(CLKDIV[11:8]).

6.10.5 Functional Description

The UART Controller supports four function modes including UART, IrDA, LIN and RS-485 mode. User can select a function by setting the UA_FUN_SEL register. The four function modes will be described in detail in sections 6.10.5.1 to section 6.10.5.4. Also refer to sections 6.10.5.5 to section 1.1.5.8 for more information.

6.10.5.1 UART Controller Baud Rate Generator

The UART Controller includes a programmable baud rate generator capable of dividing clock input by divisors to produce the serial clock that transmitter and receiver need. The baud rate equation is Baud Rate = UART_CLK / M * [BRD + 2], where M and BRD are defined in Baud Rate Divider Register (UA_BAUD). The following tables list the UART baud rate equations in the various conditions and UART baud rate parameter settings. There is no error for the baud rate results calculated through the baud rate parameter and register setting below. In IrDA function mode, the baud rate generator must be set in Mode 0.

Mode	DIV_X_EN	DIV_X_ONE	Divider X	BRD	M	Baud Rate Equation
Mode 0	0	0	B	A	16	UART_CLK / [16 * (A+2)]
Mode 1	1	0	B	A	B+1	UART_CLK / [(B+1) * (A+2)], B must >= 8
Mode 2	1	1	Don't care	A	1	UART_CLK / (A+2), A must >=8

Table 6.10-1 UART Controller Baud Rate Equation Table

UART Peripheral Clock = 22.1184 MHz			
Baud Rate	**Mode 0**	**Mode 1**	**Mode 2**
921600	Not support	A=0, B=11	A=22

460800	A=1	A=1, B=15
		A=2, B=11
			A=46

230400	A=4	A=4, B=15
		A=6, B=11
			A=94

115200	A=10	A=10, B=15
		A=14, B=11
			A=190

57600	A=22	A=22, B=15
		A=30, B=11
			A=382

38400	A=34	A=62, B=8
		A=46, B=11
		A=34, B=15
			A=574

19200	A=70	A=126, B=8
		A=94, B=11
		A=70, B=15
			A=1150

9600	A=142	A=254, B=8
		A=190, B=11
		A=142, B=15
			A=2302

4800	A=286	A=510, B=8
		A=382, B=11
		A=286, B=15
			A=4606

Table 6.10-2 UART Controller Baud Rate Parameter Setting Table

UART Peripheral Clock = 22.1184 MHz			
Baud Rate	**Mode 0**	**Mode 1**	**Mode 2**
921600	Not support	0x2B00_0000	0x3000_0016
460800	0x0000_0001	0x2F00_0001 0x2B00_0002	0x3000_002E
230400	0x0000_0004	0x2F00_0004 0x2B00_0006	0x3000_005E
115200	0x0000_000A	0x2F00_000A 0x2B00_000E	0x3000_00BE
57600	0x0000_0016	0x2F00_0016 0x2B00_001E	0x3000_017E
38400	0x0000_0022	0x2800_003E 0x2B00_002E 0x2F00_0022	0x3000_023E
19200	0x0000_0046	0x2800_007E 0x2B00_005E 0x2F00_0046	0x3000_047E
9600	0x0000_008E	0x2800_00FE 0x2B00_00BE 0x2F00_008E	0x3000_08FE
4800	0x0000_011E	0x2800_01FE 0x2B00_017E 0x2F00_011E	0x3000_11FE

Table 6.10-3 UART Controller Baud Rate Register Setting Table

6.10.5.2 UART Controller FIFO Control and Status

The UART Controller is built-in with a 16 bytes transmitter FIFO (TX_FIFO) and a 16 bytes receiver FIFO (RX_FIFO) that reduces the number of interrupts presented to the CPU. The CPU can read the status of the UART at any time during operation. The reported status information includes the type and condition of the transfer operations being performed by the UART, as well as 3 error conditions (parity error, framing error, break interrupt) probably occur while receiving data. This FIFO control and status also support all of UART, IrDA, LIN and RS-485 function mode.

6.10.5.3 UART Controller Wake-up Function

When the chip is in Power-down mode, an external CTS change will wake up chip from Power-down mode. This wake-up function is available in every function mode. User must enable the MODEN_INT interrupt to use the wake-up function.

6.10.5.4 UART Controller Interrupt and Status

Each UART Controller supports seven types of interrupts including:

- Receiver threshold level reached interrupt (RDA_INT)
- Transmitter FIFO empty interrupt (THRE_INT)
- Line status interrupt (parity error, frame error or break interrupt) (RLS_INT)
- MODEM/Wake-up status interrupt (MODEM_INT)
- Receiver buffer time-out interrupt (TOUT_INT)
- Buffer error interrupt (BUF_ERR_INT)
- LIN receiver break field detected interrupt (LIN_RX_BREAK_INT)

The following tables describe the interrupt sources and flags. The interrupt is generated when the interrupt flag is generated and the interrupt enable bit is set. User must clear the interrupt flag after the interrupt is generated.

Interrupt Source	Interrupt Indicator	Interrupt Enable Bit	Interrupt Flag	Flag Cleared by
Receive Data Available Interrupt	RDA_INT	RDA_IEN	**RDA_IF**	Read UA_RBR
Transmit Holding Register Empty Interrupt	THRE_INT	THRE_IEN	**THRE_IF**	Write UA_THR
Receive Line Status Interrupt	RLS_INT	RLS_IEN	**RLS_IF** = (BIF or FEF or PEF)	**Writing '1' to** BIF/FEF/ PEF
			RLS_IF = RS485_ADD_DETF	**Writing '1' to** RS485_ADD_DETF
Modem Status Interrupt	MODEM_INT	MODEM_IEN	**MODEM_IF** = DCTSF	**Write '1' to** DCTSF
RX Time-out Interrupt	TOUT_INT	RTO_IEN	**TOUT_IF**	Read UA_RBR
Buffer Error Interrupt	BUF_ERR_INT	BUF_ERR_IEN	**BUF_ERR_IF** = (TX_OVER_IF or RX_OVER_IF)	**Writing '1' to** TX_OVER_IF / RX_OVER_IF
LIN RX Break Field Detected interrupt	LIN_RX_BREAK_INT	LIN_RX_BRK_IEN	**LIN_RX_BREAK_IF**	**Write '1' to** LIN_RX_BREAK_IF

Table 6.10-4 UART Controller Interrupt Source and Flag List

6.10.5.5 UART Function Mode

The UART Controller provides UART function (user must set UA_FUN_SEL [1:0] to ’00’ to enable UART function mode). The UART baud rate is up to 1 Mbps.

The UART provides full-duplex and asynchronous communications. The transmitter and receiver contain 16 bytes FIFO for payloads. User can program receiver buffer trigger level and receiver buffer time-out detection for receiver. The transmitting data delay time between the last stop and the next start bit can be programed by setting DLY (UA_TOR [15:8]) register. The UART supports hardware auto-flow control and flow control function (CTS, RTS), programmable RTS flow control trigger level and fully programmable serial-interface characteristics.

UART Line Control Function

The UART Controller supports fully programmable serial-interface characteristics by setting the UA_LCR register. Software can use the UA_LCR register to program the word length, stop bit and parity bit. The following tables list the UART word and stop bit length settings and the UART parity bit settings.

NSB (UA_LCR[2])	WLS (UA_LCR[1:0])	Word Length (bit)	Stop Length (bit)
0	00	5	1
0	01	6	1
0	10	7	1
0	11	8	1
1	00	5	1.5
1	01	6	2
1	10	7	2
1	11	8	2

Table 6.10-5 UART Line Control of Word and Stop Length Setting

Parity Type	SPE (UA_LCR[5])	EPE (UA_LCR[4])	PBE (UA_LCR[3])	Description
No Parity	x	x	0	No parity bit output.
Odd Parity	0	0	1	Odd Parity is calculated by adding all the “1’s” in a data stream and adding a parity bit to the total bits, to make the total count an odd number.
Even Parity	0	1	1	Even Parity is calculated by adding all the “1’s” in a data stream and adding a parity bit to the total bits, to make the count an even number.
Forced Mask Parity	1	0	1	Parity bit always logic 1. Parity bit on the serial byte is set to “1” regardless of total number of “1’s” (even or odd counts).
Forced Space Parity	1	1	1	Parity bit always logic 0. Parity bit on the serial byte is set to “0” regardless of total number of “1’s” (even or odd counts).

Table 6.10-6 UART Line Control of Parity Bit Setting

UART Auto-Flow Control Function

The UART supports auto-flow control function that uses two signals, CTS (clear-to-send) and RTS (request-to-send), to control the flow of data transfer between the UART and external devices (e.g. Modem). When auto flow is enabled, the UART is not allowed to receive data until the UART asserts RTS to external device. When the number of bytes in the RX FIFO equals the value of RTS_TRI_LEV (UA_FCR [19:16]), the RTS is de-asserted. The UART sends data out when UART detects CTS is asserted from external device. If the valid asserted CTS is not detected, the UART will not send data out.

The following diagram demonstrates the CTS auto flow control of UART function mode. User must set AUTO_CTS_EN (UA_IER [13]) to enable CTS auto flow control function. The LEV_CTS (UA_MCR [8]) can set CTS pin input active state. The DCTSF (UA_MSR [0]) is set when any state change of CTS pin input has occurred, and then TX data will be automatically transmitted from TX FIFO.

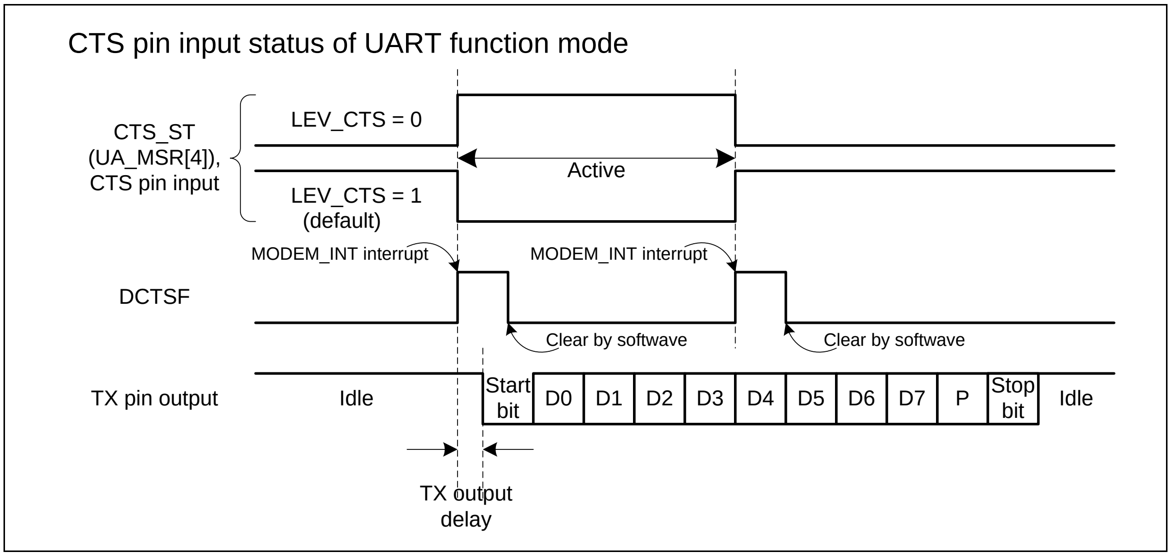


Figure 6.10-4 UART CTS Auto Flow Control Enabled

As shown in the following figure, in UART RTS Auto Flow control mode (AUTO_RTS_EN(UA_IER[12])=1), the nRTS internal signal is controlled by UART FIFO controller with RTS_RTI_LEV(UA_FCR[19:16]) trigger level.

Setting LEV_RTS(UA_MCR[9]) can control the RTS pin output is inverse or non-inverse from nRTS signal. User can read the RTS_ST(UA_MCR[13]) bit to get real RTS pin output voltage logic status.

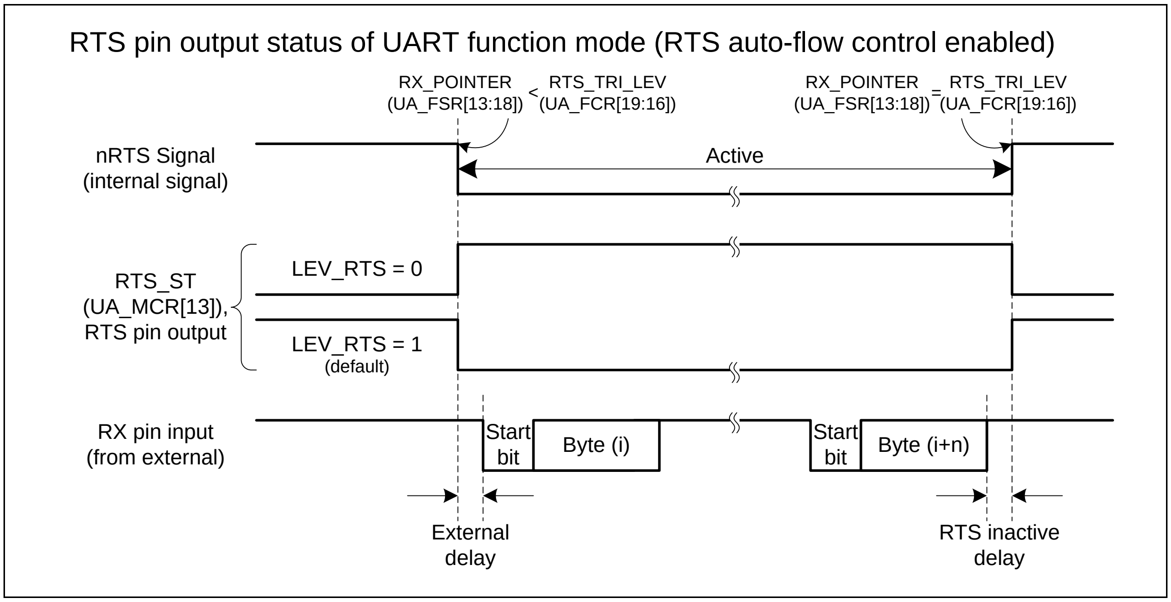

RTS pin output status of UART function mode (RTS auto-flow control enabled)
RX_POINTER (UA_FSR[13:18]) < RTS_TRI_LEV (UA_FCR[19:16])
RX_POINTER (UA_FSR[13:18]) = RTS_TRI_LEV (UA_FCR[19:16])
nRTS Signal (internal signal)
Active
RTS_ST (UA_MCR[13]), RTS pin output
LEV_RTS = 0
LEV_RTS = 1 (default)
RX pin input (from external)
Start bit
Byte (i)
Start bit
Byte (i+n)
External delay
RTS inactive delay

Figure 6.10-5 UART RTS Auto Flow Control Enabled

As shown in the following figure, in software mode (AUTO_RTS_EN(UA_IER[12])=0) the RTS flow is directly controlled by software programming of RTS(UA_MCR[1]) control bit.

Setting LEV_RTS(UA_MCR[9]) can control the RTS pin output is inverse or non-inverse from RTS(UA_MCR[1]) control bit. User can read the RTS_ST(UA_MCR[13]) bit to get real RTS pin output voltage logic status.

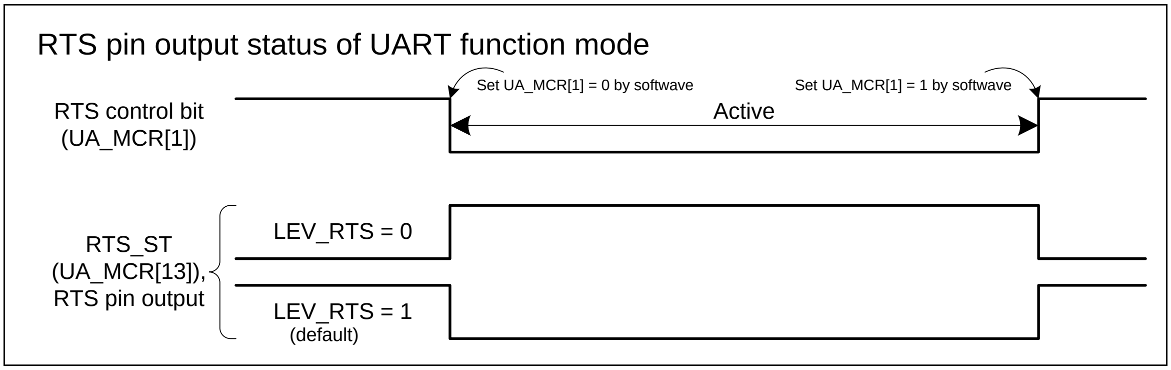

RTS pin output status of UART function mode
Set UA_MCR[1] = 0 by softwave
Set UA_MCR[1] = 1 by softwave
RTS control bit (UA_MCR[1])
Active
RTS_ST (UA_MCR[13]), RTS pin output
LEV_RTS = 0
LEV_RTS = 1 (default)

Figure 6.10-6 UART RTS Flow with Software Control

6.10.5.6 IrDA Function Mode

The UART Controller also provides Serial IrDA (SIR, Serial Infrared) function (user must set UA_FUN_SEL [1:0] to '10' to enable the IrDA function). The SIR specification defines a short-range infrared asynchronous serial transmission mode with one start bit, 8 data bits, and 1 stop bit. The maximum data rate is 115.2 kbps. The IrDA SIR block contains an IrDA SIR protocol encoder/decoder. The IrDA SIR protocol is half-duplex only. So it cannot transmit and receive data at the same time. The IrDA SIR physical layer specifies a minimum 10 ms transfer delay between transmission and reception, and this delay feature must be implemented by software.

In IrDA mode, the DIV_X_EN (UA_BAUD [29]) register must be disabled.

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

The following diagram demonstrates the IrDA control block diagram.

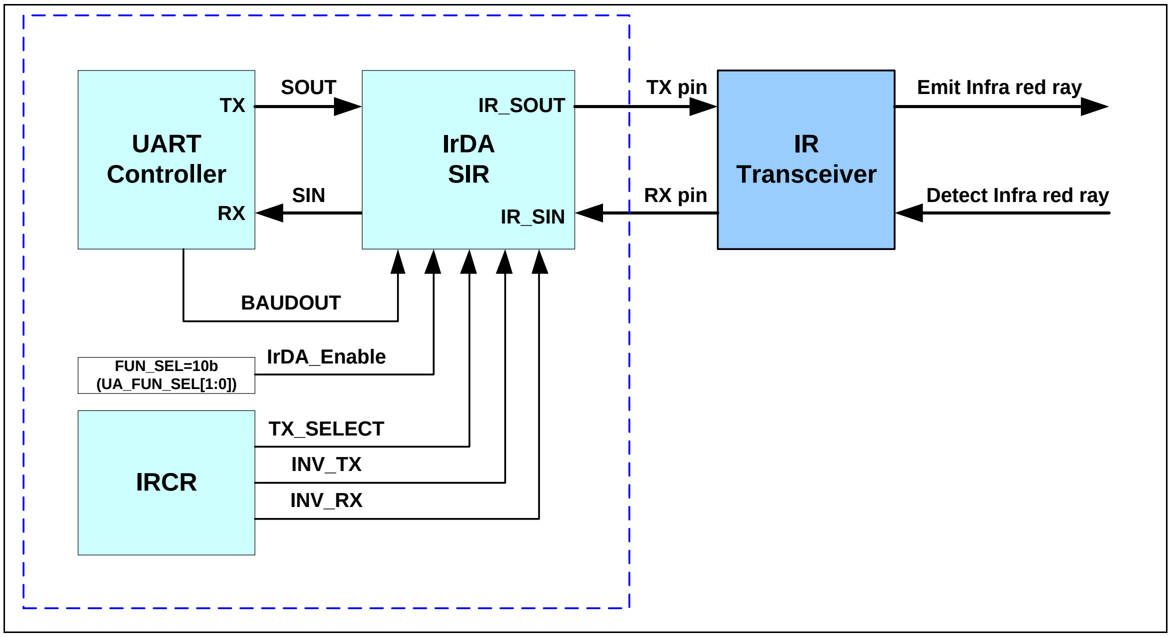


Figure 6.10-7 IrDA Control Block Diagram

IrDA SIR Transmit Encoder

The IrDA SIR Transmit Encoder modulates Non-Return-to Zero (NRZ) transmit bit stream output from UART. The IrDA SIR physical layer specifies the use of Return-to-Zero, Inverted (RZI) modulation scheme which represents logic 0 as an infra light pulse. The modulated output pulse stream is transmitted to an external output driver and infrared light emitting diode.

In Normal mode, the transmitted pulse width is specified as 3/16 period of baud rate.

IrDA SIR Receive Decoder

The IrDA SIR Receive Decoder demodulates the Return-to-Zero bit stream from the input detector and outputs the NRZ serial bits stream to the UART received data input. The decoder input is normally high in idle state. (Because of this, IRCR (INV_RX [6]) should be set as 1 by default).

A start bit is detected when the decoder input is LOW.

IrDA SIR Operation

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

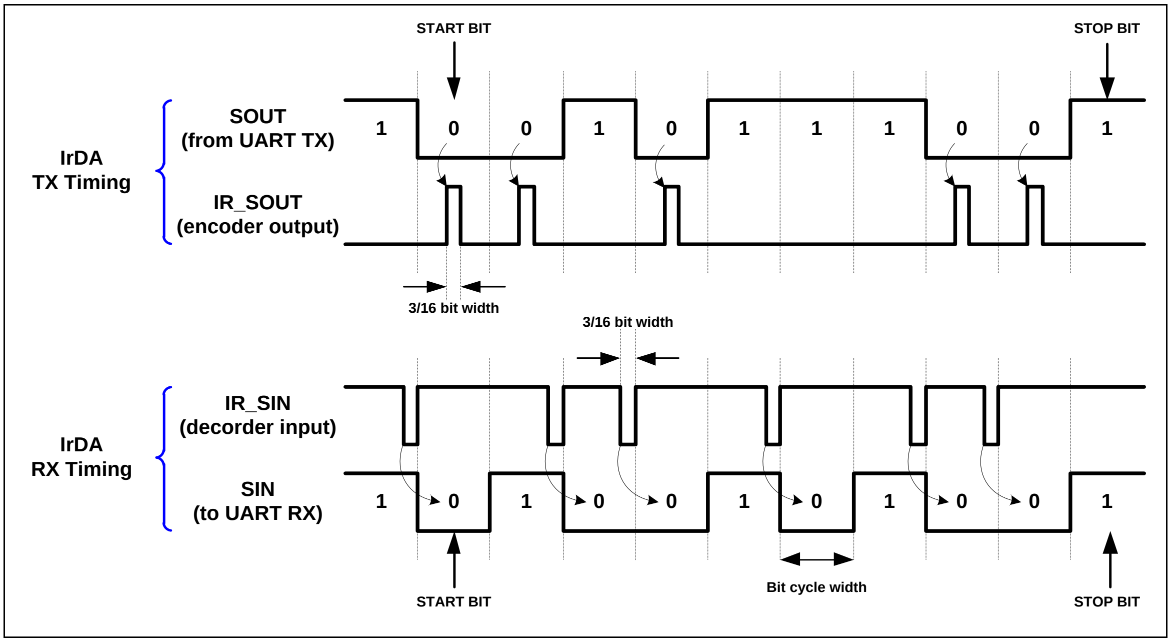


Figure 6.10-8 IrDA TX/RX Timing Diagram

6.10.5.7 LIN (Local Interconnection Network) Function Mode

The UART Controller supports LIN function (user must set UA_FUN_SEL [1:0] to ’01’ to enable LIN function). In LIN function mode, each byte field is initialed by a start bit with value 0 (dominant), followed by 8 data bits (LSB is first) and ended by 1 stop bit with value one (recessive) in accordance with the LIN standard. The following diagram is the structure of LIN function mode:

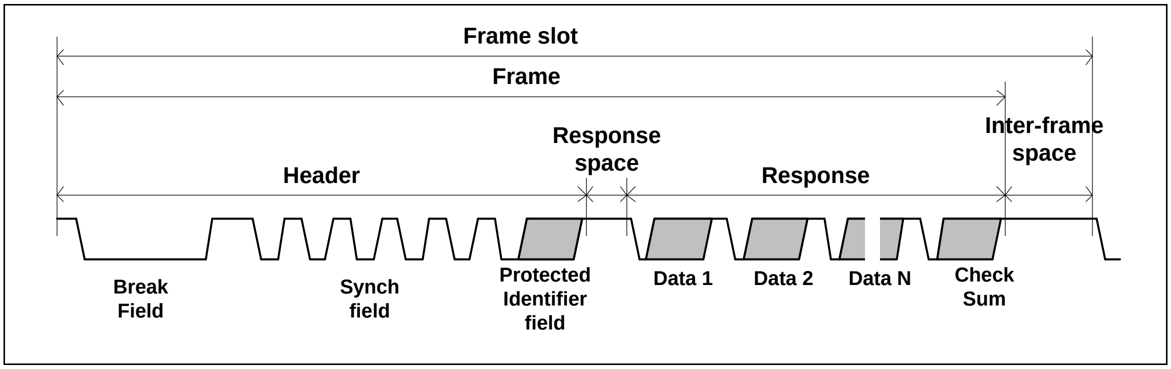


Figure 6.10-9 Structure of LIN Frame

The program flow of LIN bus transmit transfer (TX) is shown as follows:

1. Setting the UA_FUN_SEL[1:0] to ’01’ to enable LIN bus mode.
2. Fill UA_LIN_BKFL in UA_LIN_BCNT to choose break field length. (The break field length is UA_LIN_BKFL + 2).
3. Fill 0x55 to UA_THR to request synch field transmission.
4. Request identifier field transmission by writing the protected identifier value in the UA_THR
5. Setting the LIN_TX_EN bit in UA_LIN_BCNT register to start transmission. (When break filed operation is finished, LIN_TX_EN will be cleared automatically).
6. When the STOP bit of the last byte THR has been sent to bus, hardware will set flag TE_FLAG in UA_FSR to 1.
7. Fill N bytes data and checksum to UA_THR then repeat step 5 and 6 to transmit the data.

The program flow of LIN bus receiver transfer (RX) is show as follows:

1. Setting the UA_FUN_SEL[1:0] to ’01’ to enable LIN bus mode.
2. Setting the LIN_RX_EN bit in UA_LIN_BCNT register to enable LIN RX mode.
3. Waiting for the flag LIN_RX_BREAK_IF in UA_ISR to check RX received break field or not.
4. Waiting for the flag RDA_IF in UA_ISR and read back the UR_RBR register.

6.10.5.8 RS-485 Function Mode

Another alternate function of UART Controller is RS-485 function (user must set UA_FUN_SEL [1:0] to ’11’ to enable RS-485 function), and direction control provided by RTS pin or can program GPIO (P0.3 for RTS0 and P0.1 for RTS1) to implement the function by software. The RS-485 transceiver control is implemented by using the RTS control signal from an asynchronous serial port to enable the RS-485 driver. Many characteristics of the RX and TX are same as UART 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 supports three operation modes: RS-485 Normal Multidrop Operation Mode (NMM), RS-485 Auto Address Detection Operation Mode (AAD) and RS-485 Auto Direction Control Operation Mode (AUD). Software can choose any operation mode by programming the UA_ALT_CSR register, and drive the transfer delay time between the last stop bit leaving the TX FIFO and the de-assertion of by setting DLY (UA_TOR [15:8]) register.

RS-485 Normal Multidrop Operation Mode (NMM)

In RS-485 Normal Multidrop Operation Mode, in first, software must decide the data which before the address byte be detected will be stored in RX FIFO or not. If software wants to ignore any data before address byte detected, the flow is set RX_DIS (UA_FCR [8]) then enable RS485_NMM (UA_ALT_CSR [8]) and the receiver will ignore any data until an address byte is detected (bit 9 = 1) and the address byte data will be stored in the RX FIFO. If software wants to receive any data before address byte detected, the flow is disables RX_DIS (UA_FCR [8]) then enable RS485_NMM (UA_ALT_CSR [8]) and the receiver will received any data.

If an address byte is detected (bit 9 = 1), it will generate an interrupt to CPU and RX_DIS (UA_FCR [8]) can decide whether accepting the following data bytes are stored in the RX FIFO. If software disables receiver by setting RX_DIS (UA_FCR [8]) register, when a next address byte is detected, the controller will clear the RX_DIS (UA_FCR [8]) bit and the address byte data will be stored in the RX FIFO.

RS-485 Auto Address Detection Operation Mode (AAD)

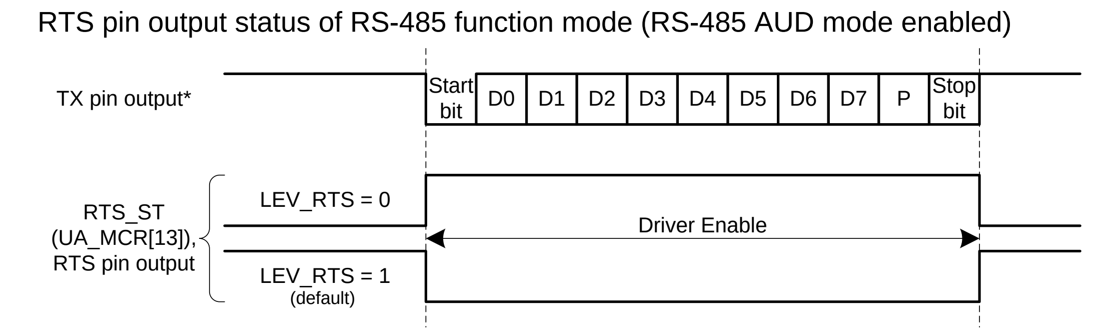
In RS-485 Auto Address Detection Operation Mode, the receiver will ignore any data until an address byte is detected (bit 9 = 1) and the address byte data matches the ADDR_MATCH (UA_ALT_CSR[31:24]) value. The address byte data will be stored in the RX FIFO. The all received byte data will be accepted and stored in the RX FIFO until and address byte data not match the ADDR_MATCH (UA_ALT_CSR[31:24]) value.

RS-485 Auto Direction Mode (AUD)

Another option function of RS-485 controllers is **RS-485 auto direction control function**. User must set RS485_AUD(UA_ALT_CSR[10]) to 0 to enabled RS-485 auto direction mode. The RS-485 transceiver control is implemented using the RTS control signal from an asynchronous serial port. The RTS line is connected to the RS-485 transceiver enable pin such that setting the RTS line to high (logic 1) enables the RS-485 transceiver. Setting the RTS line to low (logic 0) puts the transceiver into the tri-state condition to disabled. User can set LEV_RTS in UA_MCR register to change the RTS driving level.

The following diagram demonstrates the RS-485 RTS driving level in AUD mode. The RTS pin will be automatically driven during TX data transmission.

Setting LEV_RTS(UA_MCR[9]) can control RTS pin output driving level. User can read the RTS_ST(UA_MCR[13]) bit to get real RTS pin output voltage logic status.

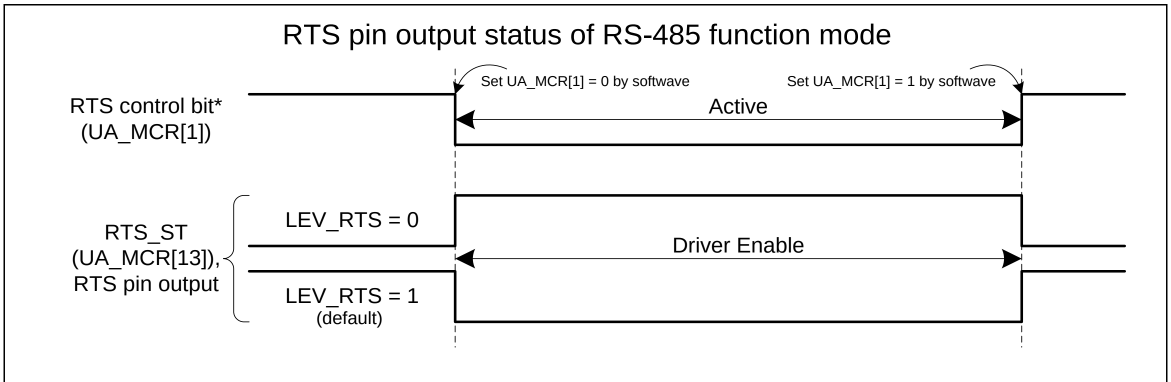

RTS pin output status of RS-485 function mode (RS-485 AUD mode enabled)
TX pin output*
Start bit
D0
D1
D2
D3
D4
D5
D6
D7
P
Stop bit
RTS_ST (UA_MCR[13]), RTS pin output
LEV_RTS = 0
Driver Enable
LEV_RTS = 1 (default)

Note: RS485_AUD(UA_ALT_CSR[10]) = 1, the RTS pin output by hardware control only.

Figure 6.10-10 RS-485 RTS Driving Level in Auto Direction Mode

The following diagram demonstrates the RS-485 RTS driving level in software control (RS485_AUD(UA_ALT_CSR[10])=0). The RTS driving level is controlled by programing the RTS(UA_MCR[1]) control bit.

Setting LEV_RTS(UA_MCR[9]) can control the RTS pin output is inverse or non-inverse from RTS(UA_MCR[1]) control bit. User can read the RTS_ST(UA_MCR[13]) bit to get real RTS pin output voltage logic status.

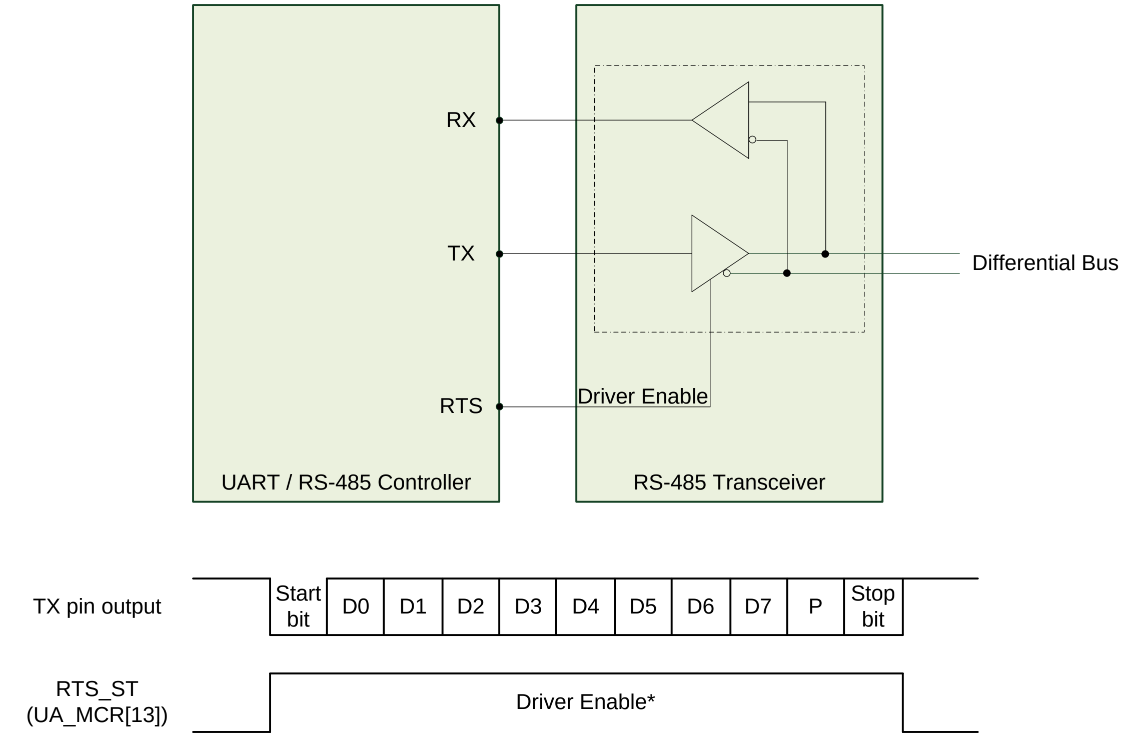


Note: RS485_AUD(UA_ALT_CSR[10]) = 0, support RTS control bit by software control only.

Figure 6.10-11 RS-485 RTS Driving Level with Software Control

Program Sequence Example:

1. Program FUN_SEL in UA_FUN_SEL to select RS-485 function.
2. Program the RX_DIS bit in UA_FCR register to determine enable or disable the receiver RS-485 receiver
3. Program the RS485_NMM or RS485_AAD mode.
4. If the RS485_AAD mode is selected, the ADDR_MATCH is programmed for auto address match value.
5. Determine auto direction control by programming RS485_AUD.



Note: RS485_AUD(UA_ALT_CSR[10]) must be set to 1, and LEV_RTS(UA_MCR[9]) must be Set to 0.

Figure 6.10-12 Structure of RS-485 Frame

6.10.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UART Base Address: **UART0_BA = 0x4005_0000**				
UA_RBR **x = 0, 1**	UARTx_BA+0x00	R	UART Receive Buffer Register	Undefined
UA_THR **x = 0, 1**	UARTx_BA+0x00	W	UART Transmit Holding Register	Undefined
UA_IER **x = 0, 1**	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000
UA_FCR **x = 0, 1**	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101
UA_LCR **x = 0, 1**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000
UA_MCR **x = 0, 1**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200
UA_MSR **x = 0, 1**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110
UA_FSR **x = 0, 1**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0x1040_4000
UA_ISR **x = 0, 1**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0000_0002
UA_TOR **x = 0, 1**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000
UA_BAUD **x = 0, 1**	UARTx_BA+0x24	R/W	UART Baud Rate Divisor Register	0x0F00_0000
UA_IRCR **x = 0, 1**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040
UA_ALT_CSR **x = 0, 1**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_0000
UA_FUN_SEL **x = 0, 1**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000

6.10.7 Register Description

UART Receive Buffer Register (UA_RBR)

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**RBR**	**Receive Buffer Register (Read Only)** By reading this register, the UART Controller will return an 8-bit data received from UART_RX pin (LSB first).

UART Transmit Holding Register (UA_THR)

Register	Offset	R/W	Description	Reset Value
UA_THR x=0,1	UARTx_BA+0x00	W	UART Transmit Holding Register	Undefined

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**THR**	**Transmit Holding Register** By writing to this register, the UART Controller will send out an 8-bit data through the UART_ TX pin (LSB first).

UART Interrupt Enable Register (UA_IER)

Register	Offset	R/W	Description	Reset Value
UA_IER x=0,1	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		AUTO_CTS_EN	AUTO_RTS_EN	TIME_OUT_EN	Reserved		LIN_RX_BRK_IEN
7	6	5	4	3	2	1	0
Reserved	WAKE_EN	BUF_ERR_IEN	RTO_IEN	MODEM_IEN	RLS_IEN	THRE_IEN	RDA_IEN

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**AUTO_CTS_EN**	**CTS Auto-Flow Control Enable Control** 0 = CTS auto-flow control Disabled. 1 = CTS auto-flow control Enabled. **Note:** 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 Control** 0 = RTS auto-flow control Disabled. 1 = RTS auto-flow control Enabled. **Note:** When RTS auto-flow is enabled, if the number of bytes in the RX FIFO equals the RTS_TRI_LEV (UA_FCR [19:16]), the UART will reassert RTS signal.
[11]	**TIME_OUT_EN**	**Time-out counter Enable Control** 0 = Time-out counter Disabled. 1 = Time-out counter Enabled.
[10:9]	**Reserved**	Reserved.
[8]	**LIN_RX_BRK_IEN**	**LIN RX Break Field Detected Interrupt Enable Control** 0 = LIN bus RX break filed interrupt Disabled. 1 = LIN bus RX break filed interrupt Enabled. **Note:** This bit is used for LIN function mode.
[7]	**Reserved**	Reserved.
[6]	**WAKE_EN**	**UART Wake-up Function Enable Control** 0 = UART wake-up function Disabled. 1 = UART wake-up function Enabled. **Note:** when the chip is in Power-down mode, an external CTS change will wake-up

		chip from Power-down mode.
[5]	**BUF_ERR_IEN**	**Buffer Error Interrupt Enable Control** 0 = BUF_ERR_INT Masked off. 1 = BUF_ERR_INT Enabled.
[4]	**RTO_IEN**	**RX Time-out Interrupt Enable Control** 0 = TOUT_INT Masked off. 1 = TOUT_INT Enabled.
[3]	**MODEM_IEN**	**Modem Status Interrupt Enable Control** 0 = MODEM_INT Masked off. 1 = MODEM_INT Enabled.
[2]	**RLS_IEN**	**Receive Line Status Interrupt Enable Control** 0 = RLS_INT Masked off. 1 = RLS_INT Enabled.
[1]	**THRE_IEN**	**Transmit Holding Register Empty Interrupt Enable Control** 0 = THRE_INT Masked off. 1 = THRE_INT Enabled.
[0]	**RDA_IEN**	**Receive Data Available Interrupt Enable Control** 0 = RDA_INT Masked off. 1 = RDA_INT Enabled.

UART FIFO Control Register (UA_FCR)

Register	Offset	R/W	Description	Reset Value
UA_FCR x=0,1	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**RTS_TRI_LEV**	**RTS Trigger Level (for Auto-flow Control Use)** 0000 = RTS Trigger Level is 1 byte. 0001 = RTS Trigger Level is 4 bytes. 0010 = RTS Trigger Level is 8 bytes. 0011 = RTS Trigger Level is 14 bytes. Other = Reserved. **Note:** This field is used for RTS auto-flow control.
[15:9]	**Reserved**	Reserved.
[8]	**RX_DIS**	**Receiver Disable Register** The receiver is disabled or not (set 1 to disable receiver) 1 = Receiver Disabled. 0 = Receiver Enabled. **Note:** This field is used for RS-485 Normal Multi-drop mode. It should be programmed before RS-485_NMM (UA_ALT_CSR [8]) is programmed.
[7:4]	**RFITL**	**RX FIFO Interrupt (RDA_INT) Trigger Level** When the number of bytes in the receive FIFO equals the RFITL then the RDA_IF will be set (if RDA_IEN in UA_IER register is enable, an interrupt will generated). 0000 = RX FIFO Interrupt Trigger Level is 1 byte. 0001 = RX FIFO Interrupt Trigger Level is 4 bytes. 0010 = RX FIFO Interrupt Trigger Level is 8 bytes. 0011 = RX FIFO Interrupt Trigger Level is 14 bytes. Other = Reserved.
[3]	**Reserved**	Reserved.

[2]	**TFR**	**TX Field Software Reset** When TX_RST is set, all the byte in the transmit FIFO and TX internal state machine are cleared. 0 = No effect. 1 = Reset the TX internal state machine and pointers. **Note:** This bit will automatically clear at least 3 UART Controller peripheral clock cycles.
[1]	**RFR**	**RX Field Software Reset** When RX_RST is set, all the byte in the receiver FIFO and RX internal state machine are cleared. 0 = No effect. 1 = Reset the RX internal state machine and pointers. **Note:** This bit will automatically clear at least 3 UART Controller peripheral clock cycles.
[0]	**Reserved**	Reserved.

UART Line Control Register (UA_LCR)

Register	Offset	R/W	Description	Reset Value
UA_LCR x=0,1	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**BCB**	**Break Control Bit** When this bit is set to logic 1, the serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX and has no effect on the transmitter logic. 0 = Break control Disabled. 1 = Break control Enabled.
[5]	**SPE**	**Stick Parity Enable Control** 0 = Stick parity Disabled. 1 = If PBE (UA_LCR[3]) and EBE (UA_LCR[4]) are logic 1, the parity bit is transmitted and checked as logic 0. If PBE (UA_LCR[3]) is 1 and EBE (UA_LCR[4]) is 0 then the parity bit is transmitted and checked as 1.
[4]	**EPE**	**Even Parity Enable Control** 0 = Odd number of logic 1's is transmitted and checked in each word. 1 = Even number of logic 1's is transmitted and checked in each word. This bit has effect only when PBE (UA_LCR[3]) is set.
[3]	**PBE**	**Parity Bit Enable Control** 0 = No parity bit. 1 = Parity bit is generated on each outgoing character and is checked on each incoming data.
[2]	**NSB**	**Number of "STOP bit"** 0 = One "STOP bit" is generated in the transmitted data. 1 = When select 5-bit word length, 1.5 "STOP bit" is generated in the transmitted data. When select 6-, 7- and 8-bti word length, 2 "STOP bit" is generated in the transmitted data.
[1:0]	**WLS**	**Word Length Selection**

		00 = Word length is 5-bit. 01 = Word length is 6-bit. 10 = Word length is 7-bit 11 = Word length is 8-bit

UART Modem Control Register (UA_MCR)

Register	Offset	R/W	Description	Reset Value
UA_MCR x=0,1	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**RTS_ST**	**RTS Pin Status (Read Only)** This bit mirror from RTS pin output of voltage logic status. 0 = RTS pin output is low level voltage logic state. 1 = RTS pin output is high level voltage logic state.
[12:10]	**Reserved**	Reserved.
[9]	**LEV_RTS**	**RTS Pin Active Level** This bit defines the active level state of RTS pin output. 0 = RTS pin output is high level active. 1 = RTS pin output is low level active. **Note1:** Refer to Figure 6.10-5 and Figure 6.10-6 for UART function mode. **Note2:** Refer to Figure 6.10-10 and Figure 6.10-11 for RS-485 function mode.
[8:2]	**Reserved**	Reserved.
[1]	**RTS**	**RTS (Request-To-Send) Signal Control** This bit is direct control internal RTS signal active or not, and then drive the RTS pin output with LEV_RTS bit configuration. 0 = RTS signal is active. 1 = RTS signal is inactive. **Note1:** This RTS signal control bit is not effective when RTS auto-flow control is enabled in UART function mode. **Note2:** This RTS signal control bit is not effective when RS-485 auto direction mode (AUD) is enabled in RS-485 function mode.
[0]	**Reserved**	Reserved.

UART Modem Status Register (UA_MSR)

Register	Offset	R/W	Description	Reset Value
UA_MSR x=0,1	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**LEV_CTS**	**CTS Pin Active Level** This bit defines the active level state of CTS pin input. 0 = CTS pin input is high level active. 1 = CTS pin input is low level active. **Note:** Refer to Figure 6.10-4 for more information.
[7:5]	**Reserved**	Reserved.
[4]	**CTS_ST**	**CTS Pin Status (Read Only)** This bit mirror from CTS pin input of voltage logic status. 0 = CTS pin input is low level voltage logic state. 1 = CTS pin input is high level voltage logic state. **Note:** This bit echoes when UART Controller peripheral clock is enabled, and CTS multi-function port is selected.
[3:1]	**Reserved**	Reserved.
[0]	**DCTSF**	**Detect CTS State Change Flag** This bit is set whenever CTS input has change state, and it will generate Modem interrupt to CPU when MODEM_IEN (UA_IER [3]) is set to 1. 0 = CTS input has not change state. 1 = CTS input has change state. **Note:** This bit is cleared by writing 1 to it.

UART FIFO Status Register (UA_FSR)

Register	Offset	R/W	Description	Reset Value
UA_FSR x=0,1	UARTx_BA+0x18	R/W	UART FIFO Status Register	0x1040_4000

31	30	29	28	27	26	25	24
Reserved			TE_FLAG	Reserved			TX_OVER_IF
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TX_FULL	TX_EMPTY	TX_POINTER					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RX_FULL	RX_EMPTY	RX_POINTER					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	BIF	FEF	PEF	RS485_ADD_DETF	Reserved		RX_OVER_IF

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**TE_FLAG**	**Transmitter Empty Flag (Read Only)** This bit is set by hardware when TX FIFO (UA_THR) is empty and the STOP bit of the last byte has been transmitted. 0 = TX FIFO is not empty. 1 = TX FIFO is empty. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[27:25]	**Reserved**	Reserved.
[24]	**TX_OVER_IF**	**TX Overflow Error Interrupt Flag** If TX FIFO (UA_THR) is full, an additional write to UA_THR will cause this bit to logic 1. 0 = TX FIFO is not overflow. 1 = TX FIFO is overflow. **Note:** This bit is cleared by writing 1 to it.
[23]	**TX_FULL**	**Transmitter FIFO Full (Read Only)** This bit indicates TX FIFO full or not. 0 = TX FIFO is not full. 1 = TX FIFO is full. **Note:** This bit is set when the number of usage in TX FIFO Buffer is equal to 16, otherwise is cleared by hardware.
[22]	**TX_EMPTY**	**Transmitter FIFO Empty (Read Only)** This bit indicates TX FIFO empty or not. 0 = TX FIFO is not empty. 1 = TX FIFO is empty.

		Note: When the last byte of TX FIFO has been transferred to Transmitter Shift Register, hardware sets this bit high. It will be cleared when writing data into THR (TX FIFO not empty).
[21:16]	**TX_POINTER**	**TX FIFO Pointer (Read Only)** This field indicates the TX FIFO Buffer Pointer. When CPU writes one byte into UA_THR, TX_POINTER increases one. When one byte of TX FIFO is transferred to Transmitter Shift Register, TX_POINTER decreases one. The Maximum value shown in TX_POINTER is 15. When the using level of TX FIFO Buffer equal to 16, the TX_FULL bit is set to 1 and TX_POINTER will show 0. As one byte of TX FIFO is transferred to Transmitter Shift Register, the TX_FULL bit is cleared to 0 and TX_POINTER will show 15.
[15]	**RX_FULL**	**Receiver FIFO Full (Read Only)** This bit initiates RX FIFO full or not. 0 = RX FIFO is not full. 1 = RX FIFO is full. **Note:** This bit is set when the number of usage in RX FIFO Buffer is equal to 16, otherwise is cleared by hardware.
[14]	**RX_EMPTY**	**Receiver FIFO Empty (Read Only)** This bit initiate RX FIFO empty or not. 0 = RX FIFO is not empty. 1 = RX FIFO is empty. **Note:** When the last byte of RX FIFO has been read by CPU, hardware sets this bit high. It will be cleared when UART receives any new data.
[13:8]	**RX_POINTER**	**RX FIFO Pointer (Read Only)** This field indicates the RX FIFO Buffer Pointer. When UART receives one byte from external device, RX_POINTER increases one. When one byte of RX FIFO is read by CPU, RX_POINTER decreases one. The Maximum value shown in RX_POINTER is 15. When the using level of RX FIFO Buffer equal to 16, the RX_FULL bit is set to 1 and RX_POINTER will show 0. As one byte of RX FIFO is read by CPU, the RX_FULL bit is cleared to 0 and RX_POINTER will show 15.
[7]	**Reserved**	Reserved.
[6]	**BIF**	**Break Interrupt Flag (Read Only)** This bit is set to logic 1 whenever the received data input (RX) is held in the “spacing state” (logic 0) for longer than a full word transmission time (that is, the total time of “start bit” + data bits + parity + stop bits). 0 = No Break interrupt is generated. 1 = Break interrupt is generated. **Note1**: This bit is read only, but can be cleared by writing ‘1’ to it **Note2**: This bit is read only, but can be cleared by writing ‘1’ to RFR (UA_FCR [1])
[5]	**FEF**	**Framing Error Flag (Read Only)** This bit is set to logic 1 whenever the received character does not have a valid “stop bit” (that is, the stop bit following the last data bit or parity bit is detected as logic 0). 0 = No framing error is generated. 1 = Framing error is generated. **Note1**: This bit is read only, but can be cleared by writing ‘1’ to it **Note2**: This bit is read only, but can be cleared by writing ‘1’ to RFR (UA_FCR [1])
[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". 0 = No parity error is generated. 1 = Parity error is generated. **Note1**: This bit is read only, but can be cleared by writing '1' to it **Note2**: This bit is read only, but can be cleared by writing '1' to RFR (UA_FCR [1])
[3]	**RS485_ADD_DETF**	**RS-485 Address Byte Detection Flag** This bit is set to 1 while RS485_ADD_EN (UA_ALT_CSR[15]) is set to 1 to enable Address detection mode and receive detect a data with an address bit (bit 9 = 1). **Note1:** This field is used for RS-485 function mode. **Note2:** This bit is cleared by writing 1 to it.
[2:1]	**Reserved**	Reserved.
[0]	**RX_OVER_IF**	**RX Overflow Error Interrupt Flag** This bit is set when RX FIFO overflow. If the number of bytes of received data is greater than RX_FIFO (UA_RBR) size, 16 bytes this bit will be set. 0 = RX FIFO is not overflow. 1 = RX FIFO is overflow. **Note:** This bit is cleared by writing 1 to it.

UART Interrupt Status Control Register (UA_ISR)

Register	Offset	R/W	Description	Reset Value
UA_ISR x=0,1	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0000_0002

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[15]	**LIN_RX_BREAK_INT**	**LIN Bus RX Break Field Detected Interrupt Indicator (Read Only)** This bit is set if LIN_RX_BRK_IEN and LIN_RX_BREAK_IF are both set to 1. 0 = No LIN RX Break interrupt is generated. 1 = LIN RX Break interrupt is generated.
[14]	**Reserved**	Reserved.
[13]	**BUF_ERR_INT**	**Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUF_ERR_IEN and BUF_ERR_IF are both set to 1. 0 = No buffer error interrupt is generated. 1 = buffer error interrupt is generated.
[12]	**TOUT_INT**	**Time-out Interrupt Indicator (Read Only)** This bit is set if RTO_IEN and TOUT_IF are both set to 1. 0 = No Time-out interrupt is generated. 1 = Time-out interrupt is generated.
[11]	**MODEM_INT**	**MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEM_IEN and MODEM_IF are both set to 1. 0 = No Modem interrupt is generated. 1 = Modem interrupt is generated.
[10]	**RLS_INT**	**Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLS_IEN and RLS_IF are both set to 1. 0 = No RLS interrupt is generated.

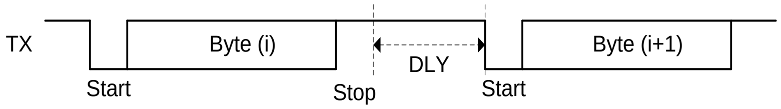
		1 = RLS interrupt is generated.
[9]	**THRE_INT**	**Transmit Holding Register Empty Interrupt Indicator (Read Only)** This bit is set if THRE_IEN and THRE_IF are both set to 1. 0 = No THRE interrupt is generated. 1 = THRE interrupt is generated.
[8]	**RDA_INT**	**Receive Data Available Interrupt Indicator (Read Only)** This bit is set if RDA_IEN and RDA_IF are both set to 1. 0 = No RDA interrupt is generated. 1 = RDA interrupt is generated.
[7]	**LIN_RX_BREAK_IF**	**LIN Bus RX Break Field Detected Flag** This bit is set when RX received LIN Break Field. If LIN_RX_BRK_IEN (UA_IER [8]) is enabled the LIN RX Break interrupt will be generated. 0 = No LIN RX Break received 1 = LIN RX Break received **Note:** This bit is cleared by writing 1 to it.
[6]	**Reserved**	Reserved.
[5]	**BUF_ERR_IF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX/RX FIFO overflow flag (TX_OVER_IF or RX_OVER_IF) is set. When BUF_ERR_IF is set, the transfer is not correct. If BUF_ERR_IEN (UA_IER [5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated. 1 = Buffer error interrupt flag is generated. **Note1:** this bit is read only and reset to 0 when all bits of TX_OVER_IF and RX_OVER_IF are cleared.
[4]	**TOUT_IF**	**Time-out Interrupt Flag (Read Only)** This bit is set when the RX FIFO is not empty and no activities occurred in the RX FIFO and the time-out counter equal to TOIC. If RTO_IEN (UA_IER [4]) is enabled, the Tout interrupt will be generated. 0 = No Time-out interrupt flag is generated. 1 = Time-out interrupt flag is generated. **Note:** This bit is read only and user can read UA_RBR (RX is in active) to clear it.
[3]	**MODEM_IF**	**MODEM Interrupt Flag (Read Only)** 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. 0 = No Modem interrupt flag is generated. 1 = Modem interrupt flag is generated. **Note:** This bit is read only and reset to 0 when bit DCTSF is cleared by a write 1 on DCTSF.
[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 RLS_IEN (UA_IER [2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated. 1 = RLS interrupt flag is generated. **Note1:** In RS-485 function mode, this field is set including “receiver detect and

		received address byte character (bit 9 = 1) bit". At the same time, the bit of RS485_ADD_DETF (UA_FSR[3]) is also set. **Note2:** This bit is read only and reset to 0 when all bits of BIF, FEF, PEF and RS485_ADD_DETF are cleared.
[1]	**THRE_IF**	**Transmit Holding Register Empty Interrupt Flag (Read Only)** This bit is set when the last data of TX FIFO is transferred to Transmitter Shift Register. If THRE_IEN (UA_IER [1]) is enabled, the THRE interrupt will be generated. 0 = No THRE interrupt flag is generated. 1 = THRE interrupt flag is generated. **Note:** This bit is read only and it will be cleared when writing data into THR (TX FIFO not empty).
[0]	**RDA_IF**	**Receive Data Available Interrupt Flag (Read Only)** When the number of bytes in the RX FIFO equals the RFITL then the RDA_IF will be set. If RDA_IEN (UA_IER [0]) is enabled, the RDA interrupt will be generated. 0 = No RDA interrupt flag is generated. 1 = RDA interrupt flag is generated. **Note:** This bit is read only and it will be cleared when the number of unread bytes of RX FIFO drops below the threshold level (RFITL).

UART Time-out Register (UA_TOR)

Register	Offset	R/W	Description	Reset Value
UA_TOR x=0,1	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**DLY**	**TX Delay Time Value** This field is use to programming the transfer delay time between the last stop bit and next start bit.  TX Byte (i) Start Stop DLY Start Byte (i+1)
[7:0]	**TOIC**	**Time-out Interrupt Comparator** The time-out counter resets and starts counting (the counting clock = baud rate) whenever the RX FIFO receives a new data word. Once the content of time-out counter (TOUT_CNT) is equal to that of time-out interrupt comparator (TOIC), a receiver time-out interrupt (TOUT_INT) is generated if RTO_IEN (UA_IER [4]). A new incoming data word or RX FIFO empty clears TOUT_INT. In order to avoid receiver time-out interrupt generation immediately during one character is being received, TOIC value should be set between 40 and 255. So, for example, if TOIC is set with 40, the time-out interrupt is generated after four characters are not received when 1 stop bit and no parity check is set for UART transfer.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

UART Baud Rate Divider Register (UA_BAUD)

Register	Offset	R/W	Description	Reset Value
UA_BAUD x=0,1	UARTx_BA+0x24	R/W	UART Baud Rate Divisor Register	0x0F00_0000

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

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**DIV_X_EN**	**Divider X Enable Control** The BRD = Baud Rate Divider, and the baud rate equation is Baud Rate = Clock / [M * (BRD + 2)]; The default value of M is 16. 0 = Divider X Disabled (the equation of M = 16). 1 = Divider X Enabled (the equation of M = X+1, but DIVIDER_X [27:24] must >= 8). **Note1:** Refer to the section 6.10.5.1 for more information. **Note2:** In IrDA function 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 >= 8). **Note1:** Refer to the section 6.10.5.1 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.

UART IrDA Control Register (UART_IRCR)

Register	Offset	R/W	Description	Reset Value
UA_IRCR x=0,1	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**INV_RX**	**Inverse RX Input Control** 0 = No inversion. 1 = Inverse RX input signal.
[5]	**INV_TX**	**Inverse RX Output Control** 0 = No inversion. 1 = Inverse TX output signal.
[4:2]	**Reserved**	Reserved.
[1]	**TX_SELECT**	**IrDA Receiver Enable Control** 0 = IrDA receiver Enabled. 1 = IrDA transmitter Enabled.
[0]	**Reserved**	Reserved.

UART Alternate Control/Status Register (UA_ALT_CSR)

Register	Offset	R/W	Description	Reset Value
UA_ALT_CSR x=0,1	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_0000

31	30	29	28	27	26	25	24
ADDR_MATCH							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RS485_ADD_EN	Reserved				RS485_AUD	RS485_AAD	RS485_NMM
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LIN_TX_EN	LIN_RX_EN	Reserved		UA_LIN_BKFL			

Bits	Description	
[31:24]	**ADDR_MATCH**	**Address Match Value Register** This field contains the RS-485 address match values. **Note:** This field is used for RS-485 auto address detection mode.
[23:16]	**Reserved**	Reserved.
[15]	**RS485_ADD_EN**	**RS-485 Address Detection Enable Control** This bit is use to enable RS-485 Address Detection mode. 0 = RS-485 address detection mode Disabled. 1 = RS-485 address detection mode Enabled. **Note:** This field is used for RS-485 any operation mode.
[14:11]	**Reserved**	Reserved.
[10]	**RS485_AUD**	**RS-485 Auto Direction Mode (AUD) Control** 0 = RS-485 Auto Direction Operation Mode (AUD) Disabled. 1 = RS-485 Auto Direction Operation Mode (AUD) Enabled. **Note:** It can be active with RS485_AAD or RS485_NMM operation mode.
[9]	**RS485_AAD**	**RS-485 Auto Address Detection Operation Mode (AAD) Control** 0 = RS-485 Auto Address Detection Operation Mode (AAD) Disabled. 1 = RS-485 Auto Address Detection Operation Mode (AAD) Enabled. **Note:** It cannot be active with RS485_NMM operation mode.
[8]	**RS485_NMM**	**RS-485 Normal Multi-drop Operation Mode (NMM) Control** 0 = RS-485 Normal Multi-drop Operation Mode (NMM) Disabled. 1 = RS-485 Normal Multi-drop Operation Mode (NMM) Enabled. **Note:** It cannot be active with RS485_AAD operation mode.

[7]	**LIN_TX_EN**	**LIN TX Break Mode Enable Control** 0 = LIN TX Break Mode Disabled. 1 = LIN TX Break Mode Enabled. **Note:** When TX break field transfer operation finished, this bit will be cleared automatically.
[6]	**LIN_RX_EN**	**LIN RX Enable Control** 0 = LIN RX mode Disabled. 1 = LIN RX mode Enabled.
[5:4]	**Reserved**	Reserved.
[3:0]	**UA_LIN_BKFL**	**UART LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note:** This break field length is UA_LIN_BKFL + 2.

UART Function Select Register (UA_FUN_SEL)

Register	Offset	R/W	Description	Reset Value
UA_FUN_SEL x=0,1	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1:0]	**FUN_SEL**	**Function Select** UART Controller function mode selection. 00 = UART function mode. 01 = LIN function mode. 10 = IrDA function mode. 11 = RS-485 function mode.

6.11 I^{2}C Serial Interface Controller (I^{2}C)

6.11.1 Overview

I^{2}C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. The I^{2}C 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. There are two sets of I^{2}C which supports Power-down wake up function.

6.11.2 Features

The I^{2}C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main features of the I^{2}C bus include:

- Supports up to two I^{2}C ports
- Master/Slave mode
- Bidirectional data transfer between master and slave
- Multi-master bus (no central master)
- Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
- Serial clock synchronization allowing devices with different bit rates to communicate via one serial bus
- Serial clock synchronization used as a handshake mechanism to suspend and resume serial transfer
- Built-in a 14-bit time-out counter requesting the I^{2}C interrupt if the I^{2}C bus hangs up and timer-out counter overflows.
- Programmable clocks allowing for versatile rate control
- Supports 7-bit addressing mode
- Supports multiple address recognition (four slave address with mask option)
- Supports Power-down Wake-up function

6.11.3 Basic Configuration

The basic configurations of I^{2}C0 are as follows:

- I^{2}C0 pins are configured on P3_MFP [13:12] register.
- Enable I^{2}C0 clock (I2C0_EN) on APBCLK [8] register.
- Reset I^{2}C0 controller (I2C0_RST) on IPRSTC2 [8] register.

The basic configurations of I^{2}C1 are as follows:

- I^{2}C1 pins are configured on P2_MFP [5:4] or P4_MFP[5:4] registers.
- Enable I^{2}C1 clock (I2C1_EN) on APBCLK [9] register.
- Reset I^{2}C1 controller (I2C1_RST) on IPRSTC2 [9] register.

6.11.4 Functional Description

On I^{2}C bus, data is transferred between a Master and a Slave. Data bits transfer on the SCL and SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit long. There is one SCL clock pulse for each data bit with the MSB being transmitted first, and an acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (START or STOP). Please refer to the following figure for more detailed I^{2}C BUS Timing.

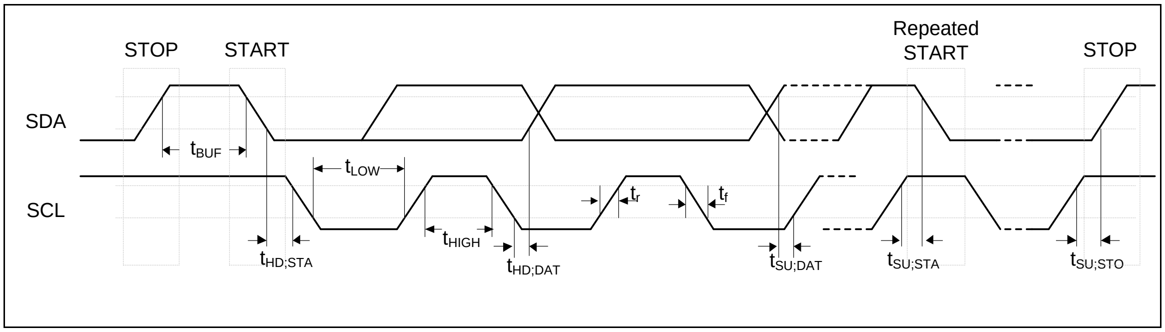


Figure 6.11-1 I^{2}C Bus Timing

The device's on-chip I^{2}C provides the serial interface that meets the I^{2}C bus standard mode specification. The I^{2}C port handles byte transfers autonomously. To enable this port, the bit ENS1 in I2CON should be set to '1'. The I^{2}C hardware interfaces to the I^{2}C bus via two pins: SDA and SCL. When I/O pins are used as I^{2}C ports, user must set the pins function to I^{2}C in advance.

Note: Pull-up resistor is needed for I^{2}C operation as the SDA and SCL are open-drain pins.

6.11.5 I²C Protocol

The following figure shows the typical 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

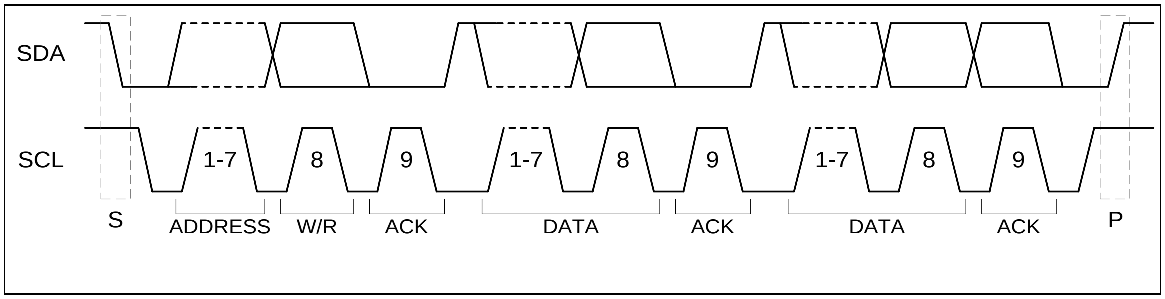


Figure 6.11-2 I²C Protocol

6.11.5.1 START or Repeated START signal

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

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

6.11.5.2 STOP signal

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

The following figure shows the waveform of START, Repeat START and STOP.

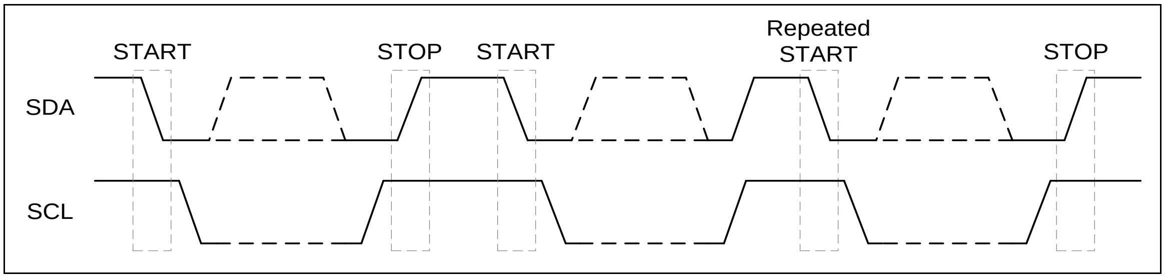


Figure 6.11-3 START and STOP Conditions

6.11.5.3 Slave Address Transfer

The first byte of data transferred by the master immediately after the START signal is the Slave address (SLA). This is a 7-bit calling address followed by a Read/Write (R/W) bit. The R/W bit signals of the slave indicate the data transfer direction. No two slaves in the system can have the same address. Only the slave with an address that matches the one transmitted by the master will respond by returning an acknowledge bit by pulling the SDA low at the 9th SCL clock cycle.

6.11.5.4 Data Transfer

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

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

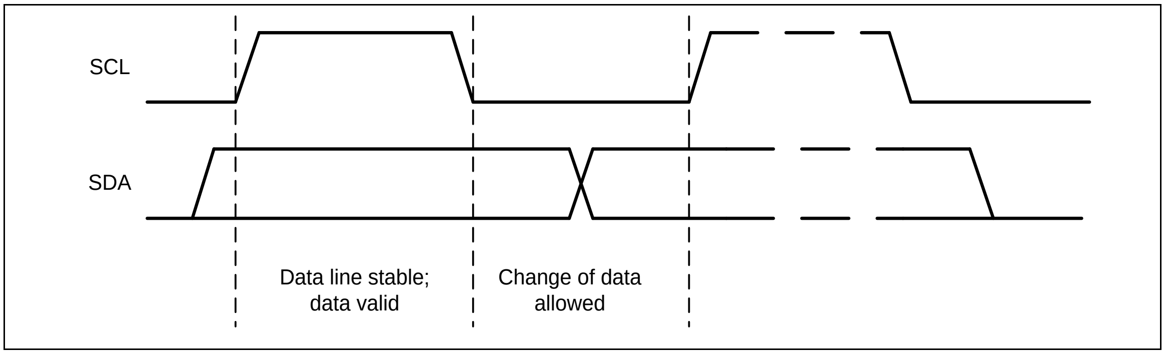


Figure 6.11-4 Bit Transfer on the I^{2}C Bus

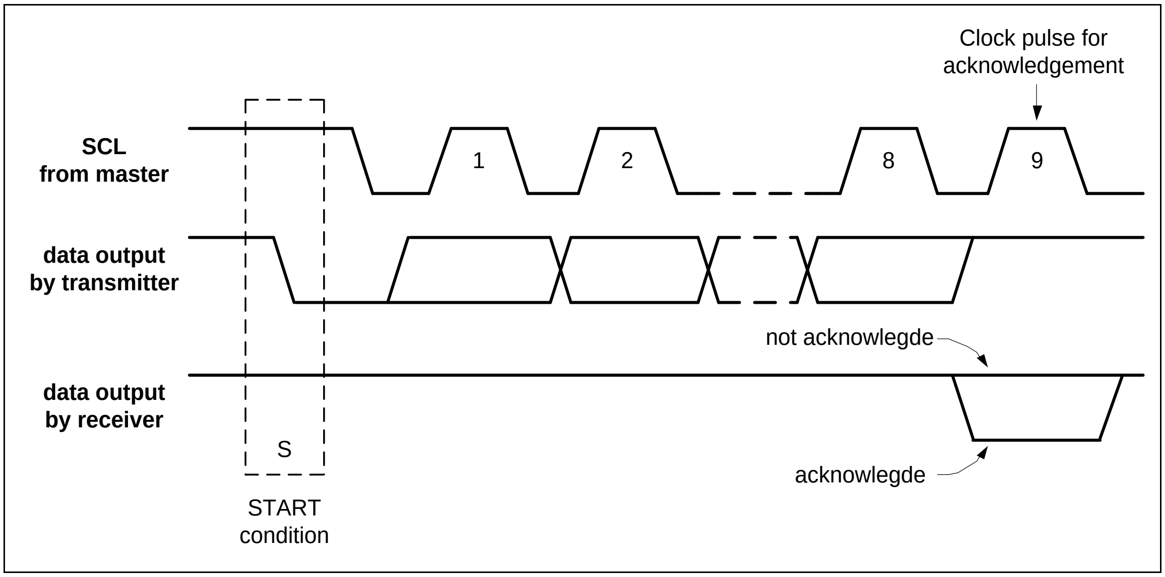


Figure 6.11-5 Acknowledge on the I^{2}C Bus

6.11.5.5 Data transfer on the I^2C bus

The following figure shows a master transmits data to slave. A master addresses a slave with a 7-bit address and 1-bit write index to denote that the master wants to transmit data to the slave. The master keeps transmitting data after the slave returns acknowledge to the master.

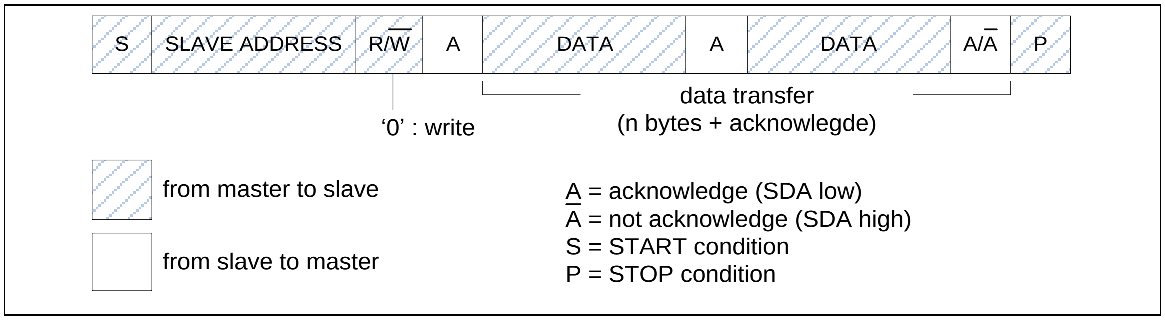


Figure 6.11-6 Master Transmits Data to Slave

The following figure shows a master read data from slave. A master addresses a slave with a 7-bit address and 1-bit read index to denote that the master wants to read data from the slave. The slave will start transmitting data after the slave returns acknowledge to the master.

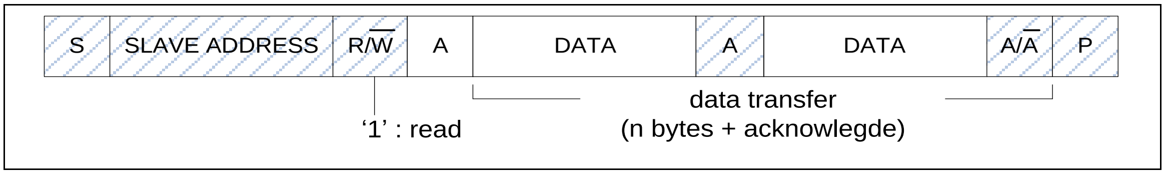


Figure 6.11-7 Master Reads Data from Slave

6.11.6 I^2C Protocol Registers

To control I^2C port through the following fifteen special function registers: I2CON (control register), I2CSTATUS (status register), I2CDAT (data register), I2CADDRn (address registers, n=0~3), I2CADMn (address mask registers, n=0~3), I2CLK (clock rate register), I2CTOC (Time-out counter register), I2CWKUPCON (wake up control register) and I2CWKUPSTS (wake up status register).

Address Registers (I2CADDR)

The I^2C port is equipped with four slave address registers, I2CADDRn (n=0~3). The contents of the register are irrelevant when I^2C is in Master mode. In Slave mode, the bit field I2CADDRn[7:1] must be loaded with the chip's own slave address. The I^2C hardware will react if the contents of I2CADDRn are matched with the received slave address.

The I^2C ports support the "General Call" function. If the GC bit (I2CADDRn [0]) is set the I^2C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.

When the GC bit is set and the I^2C is in Slave mode, it can receive the general call address by 00H after Master send general call address to I^2C bus, then it will follow status of GC mode.

Slave Address Mask Registers (I2CADM)

The I^{2}C bus controller supports multiple address recognition with four address mask registers I2CADMn (n=0~3). When the bit in the address mask register is set to 1, it means the received corresponding address bit is "Don't care". If the bit is set to 0, it means the received corresponding register bit should be exactly the same as address register.

Data Register (I2CDAT)

This register contains a byte of serial data to be transmitted or a byte which just has been received. The CPU can be read from or written to the 8-bit (I2CDAT [7:0]) directly while it is not in the process of shifting a byte. When I^{2}C is in a defined state and the serial interrupt flag (SI) is set, data in I2CDAT [7:0] remains stable. While data is being shifted out, data on the bus is simultaneously being shifted in; I2CDAT [7:0] always contains the last data byte presented on the bus.

The acknowledge bit is controlled by the I^{2}C hardware and cannot be accessed by the CPU. Serial data is shifted into I2CDAT [7:0] on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into I2CDAT [7:0], the serial data is available in I2CDAT [7:0], and the acknowledge bit (ACK or NACK) is returned by the control logic during the ninth clock pulse. In order to monitor bus status while sending data, the bus date will be shifted to I2CDATA[7:0] when sending I2CDATA[7:0] to bus. In the case of sending data, serial data bits are shifted out from I2CDAT [7:0] on the falling edge of SCL clocks, and is shifted to I2CDAT [7:0] on the rising edge of SCL clocks.

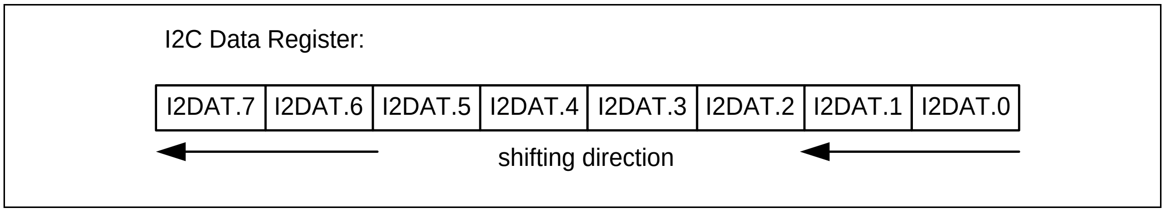


Figure 6.11-8 I^{2}C Data Shifting Direction

Control Register (I2CON)

The CPU can read and written I2CON [7:0] directly. When the I^{2}C port is enabled by setting ENS1 (I2CON [6]) to high, the internal states will be controlled by I2CON and I^{2}C logic hardware.

There are two bits are affected by hardware: the SI bit is set when the I^{2}C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when ENS1 = 0.

Once a new status code is generated and stored in I2CSTATUS, the I^{2}C Interrupt Flag bit SI (I2CON [3]) will be set automatically. If the Enable Interrupt bit EI (I2CON [7]) is set at this time, the I^{2}C interrupt will be generated. The bit field I2CSTATUS[7:0] stores the internal state code, the content keeps stable until SI is cleared by software.

Status Register (I2CSTATUS)

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

In addition, the state 00H stands for a Bus Error, which occurs when a START or STOP condition is present at an incorrect position in the I^{2}C format frame. A Bus Error may occur during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I^{2}C from bus error, STO should be set and SI should be cleared to enter Not Addressed Slave mode. Then STO is cleared to release bus and to wait for a new communication. The I^{2}C bus cannot recognize stop condition during this action when a bus error occurs.

Master Mode		**Slave Mode**	
STATUS	**Description**	**STATUS**	**Description**
0x08	Start	0xA0	Slave Transmit Repeat Start or Stop
0x10	Master Repeat Start	0xA8	Slave Transmit Address ACK
0x18	Master Transmit Address ACK	0xB0	Slave Transmit Arbitration Lost
0x20	Master Transmit Address NACK	0xB8	Slave Transmit Data ACK
0x28	Master Transmit Data ACK	0xC0	Slave Transmit Data NACK
0x30	Master Transmit Data NACK	0xC8	Slave Transmit Last Data ACK
0x38	Master Arbitration Lost	0x60	Slave Receive Address ACK
0x40	Master Receive Address ACK	0x68	Slave Receive Arbitration Lost
0x48	Master Receive Address NACK	0x80	Slave Receive Data ACK
0x50	Master Receive Data ACK	0x88	Slave Receive Data NACK
0x58	Master Receive Data NACK	0x70	GC mode Address ACK
0x00	Bus error	0x78	GC mode Arbitration Lost
		0x90	GC mode Data ACK
		0x98	GC mode Data NACK
0xF8	Bus Released **Note:** Status “0xF8” exists in both master/slave modes, and it won’t raise interrupt.		

Table 6.11-1 I^{2}C Status Code Description

Clock Baud Rate Bits (I2CLK)

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

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

Time-out Counter Register (I2CTOC)

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

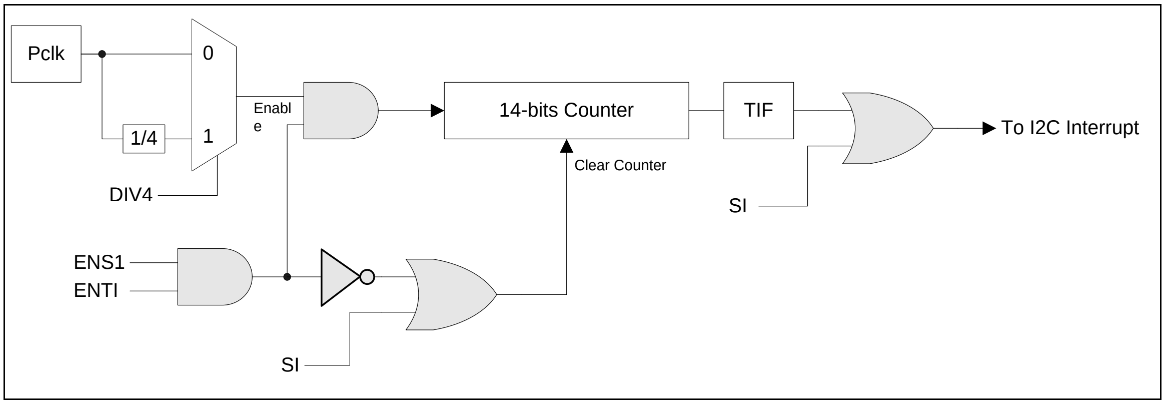


Figure 6.11-9 I^{2}C Time-out Count Block Diagram

Wake-up Control Register (I2CWKUPCON)

When chip enters Power-down mode, other I^{2}C master can wake up our chip by addressing our I^{2}C device, user must configure the related setting before entering Sleep mode. When the chip is woken-up by address match with one of the four address register, the following data will be abandoned at this time.

Wake-up Status Register (I2CWKUPSTS)

When system is woken up by other I^{2}C master device, WKUPIF is set to indicate this event. User needs write “1” to clear this bit.

6.11.7 Operation Modes

The on-chip I^{2}C ports support three operation modes, Master, Slave, and General Call Mode.

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

To control the I^{2}C bus transfer in each mode, user needs to set I2CON, I2CDAT registers according to current status code of I2CSTATUS register. In other words, for each I^{2}C bus action, user needs to check current status by I2CSTATUS register, and then set I2CON, I2CDAT registers to take bus action. Finally, check the response status by I2CSTATUS.

The bits, STA, STO and AA in I2CON register are used to control the next state of the I^{2}C hardware after SI flag of I2CON [3] register is cleared. Upon completion of the new action, a new status code will be updated in I2CSTATUS register and the SI flag of I2CON register will be set. If the I^{2}C interrupt control bit EI (I2CON [7]) is set, appropriate action or software branch of the new status code can be performed in the Interrupt service routine.

The following figure shows the current I^{2}C status code is 0x08, and then set I2CDATA=SLA+W and (STA,STO,SI,AA) = (0,0,1,x) to send the address to I^{2}C bus. If a slave on the bus matches the address and response ACK, the I2CSTATUS will be updated by status code 0x18.

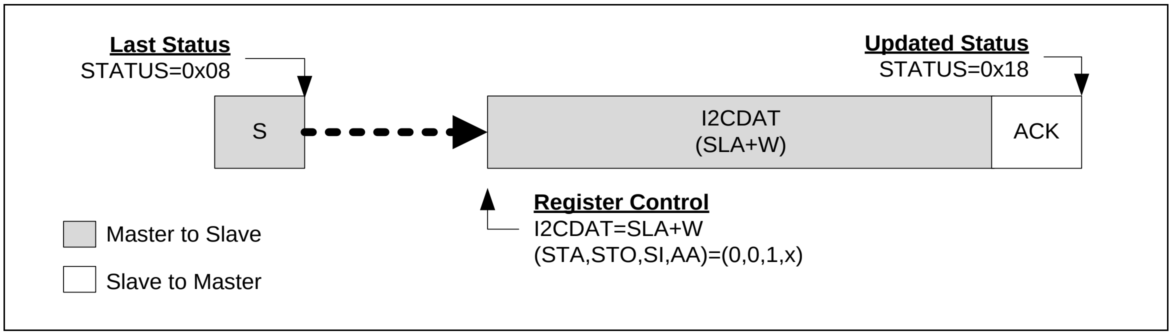


Figure 6.11-10 Control I^{2}C Bus according to Current I^{2}C Status

6.11.7.1 Master Mode

In below figures, all possible protocols for I^{2}C master are shown. User needs to follow proper path of the flow to implement required I^{2}C protocol.

In other words, user can send a START signal to bus and I^{2}C will be in Master Transmitter mode (Figure 6.11-11) or Master receiver mode (Figure 6.11-12) after START signal has been sent successfully and new status code would be 0x08. Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^{2}C protocol.

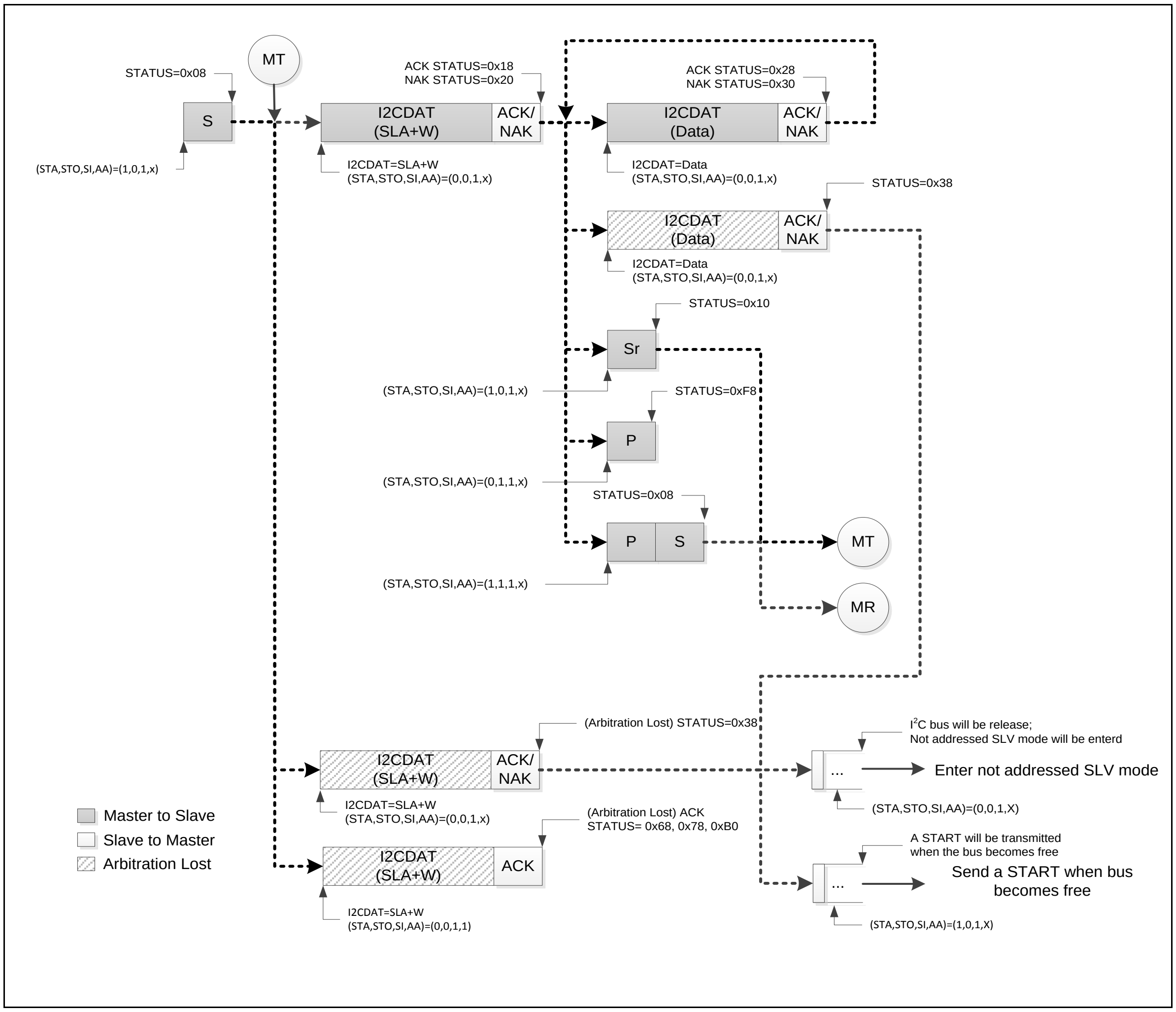

STATUS=0x08
MT
S
(STA,STO,SI,AA)=(1,0,1,x)
ACK STATUS=0x18
NAK STATUS=0x20
I2CDAT (SLA+W)
ACK/ NAK
I2CDAT=SLA+W
(STA,STO,SI,AA)=(0,0,1,x)
ACK STATUS=0x28
NAK STATUS=0x30
I2CDAT (Data)
ACK/ NAK
I2CDAT=Data
(STA,STO,SI,AA)=(0,0,1,x)
STATUS=0x38
I2CDAT (Data)
ACK/ NAK
I2CDAT=Data
(STA,STO,SI,AA)=(0,0,1,x)
STATUS=0x10
Sr
(STA,STO,SI,AA)=(1,0,1,x)
STATUS=0xF8
P
(STA,STO,SI,AA)=(0,1,1,x)
STATUS=0x08
P S
(STA,STO,SI,AA)=(1,1,1,x)
MT
MR
(Arbitration Lost) STATUS=0x38
I2CDAT (SLA+W)
ACK/ NAK
I2CDAT=SLA+W
(STA,STO,SI,AA)=(0,0,1,x)
(Arbitration Lost) ACK
STATUS= 0x68, 0x78, 0xB0
I2CDAT (SLA+W)
ACK
I2CDAT=SLA+W
(STA,STO,SI,AA)=(0,0,1,1)
I²C bus will be release;
Not addressed SLV mode will be enterd
...
Enter not addressed SLV mode
(STA,STO,SI,AA)=(0,0,1,X)
A START will be transmitted
when the bus becomes free
...
Send a START when bus becomes free
(STA,STO,SI,AA)=(1,0,1,X)
Master to Slave
Slave to Master
Arbitration Lost

Figure 6.11-11 Master Transmitter Mode Control Flow

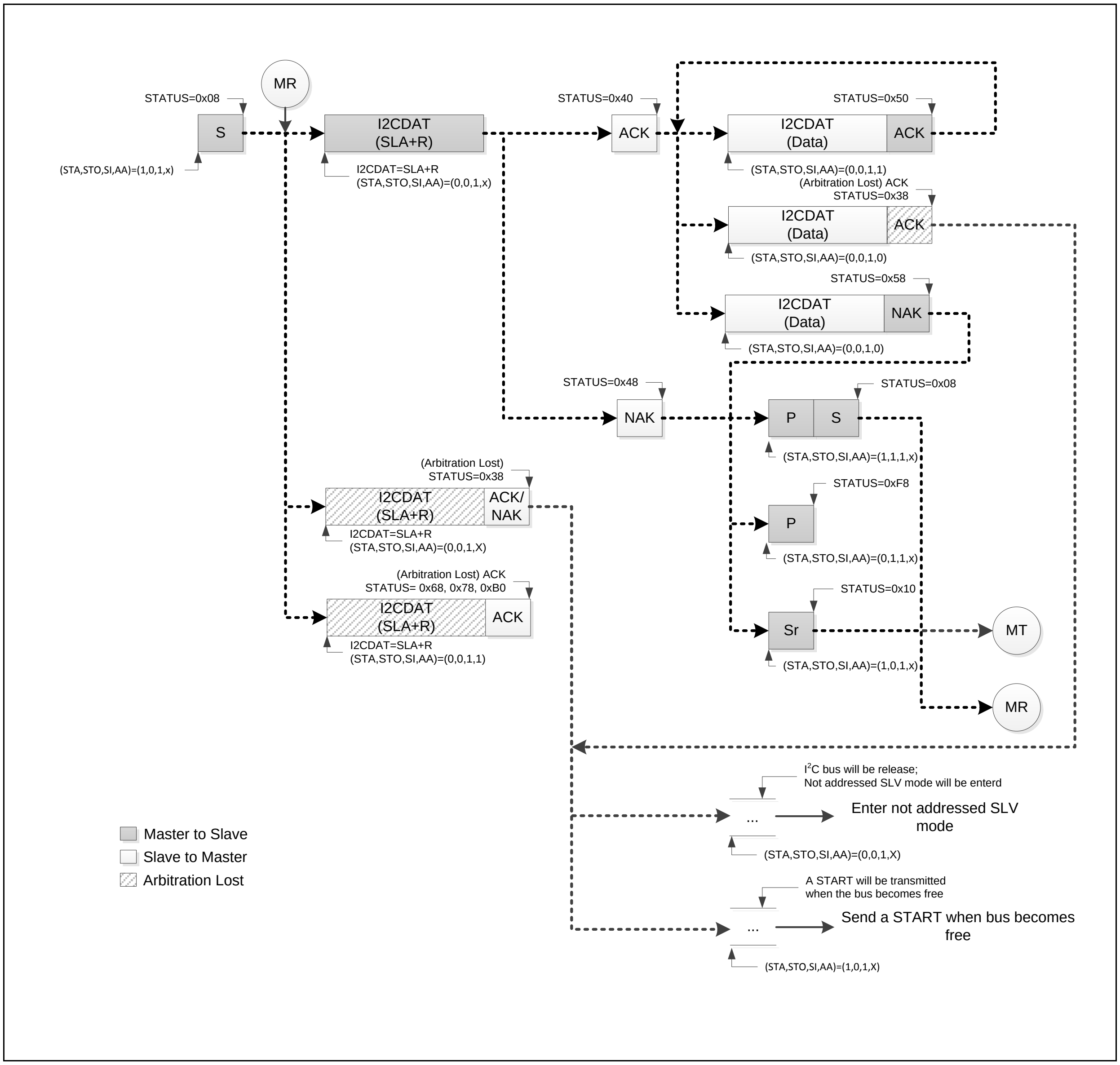

MR
STATUS=0x08
S
(STA,STO,SI,AA)=(1,0,1,x)
I2CDAT
(SLA+R)
I2CDAT=SLA+R
(STA,STO,SI,AA)=(0,0,1,x)
STATUS=0x40
ACK
STATUS=0x50
I2CDAT
(Data)
ACK
(STA,STO,SI,AA)=(0,0,1,1)
(Arbitration Lost) ACK
STATUS=0x38
I2CDAT
(Data)
ACK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0x58
I2CDAT
(Data)
NAK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0x48
NAK
STATUS=0x08
P
S
(STA,STO,SI,AA)=(1,1,1,x)
(Arbitration Lost)
STATUS=0x38
I2CDAT
(SLA+R)
ACK/
NAK
I2CDAT=SLA+R
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xF8
P
(STA,STO,SI,AA)=(0,1,1,x)
(Arbitration Lost) ACK
STATUS= 0x68, 0x78, 0xB0
I2CDAT
(SLA+R)
ACK
I2CDAT=SLA+R
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0x10
Sr
MT
(STA,STO,SI,AA)=(1,0,1,x)
MR
I^{2}C bus will be release;
Not addressed SLV mode will be enterd
...
Enter not addressed SLV mode
(STA,STO,SI,AA)=(0,0,1,X)
A START will be transmitted
when the bus becomes free
...
Send a START when bus becomes free
(STA,STO,SI,AA)=(1,0,1,X)
▭ Master to Slave
□ Slave to Master
▨ Arbitration Lost

Figure 6-23 Master Receiver Mode Control Flow

If the I^{2}C is in Master mode and gets arbitration lost, the status code will be 0x38. In status 0x38, user may set (STA, STO, SI, AA) = (1, 0, 1, X) to send START to re-start Master operation when bus become free. Otherwise, user may set (STA, STO, SI, AA) = (0, 0, 1, X) to release I^{2}C bus and enter not addressed Slave mode.

6.11.7.2 Slave Mode

When reset default, I²C is not addressed and will not recognize the address on I²C bus. User can set slave address by I2CADDRx and set (STA, STO, SI, AA) = (0, 0, 1, 1) to let I²C recognize the address sent by master. Figure 6.11-12 shows all the possible flow for I²C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.11-12 to implement their own I²C protocol.

If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) after arbitration lost, the status code is 0x68. If the detected address is SLA+R (Master want to read data from Slave) after arbitration lost, the status code is 0xB0.

Note: During I²C communication, the SCL clock will be released when writing '1' to clear SI flag in Slave mode.

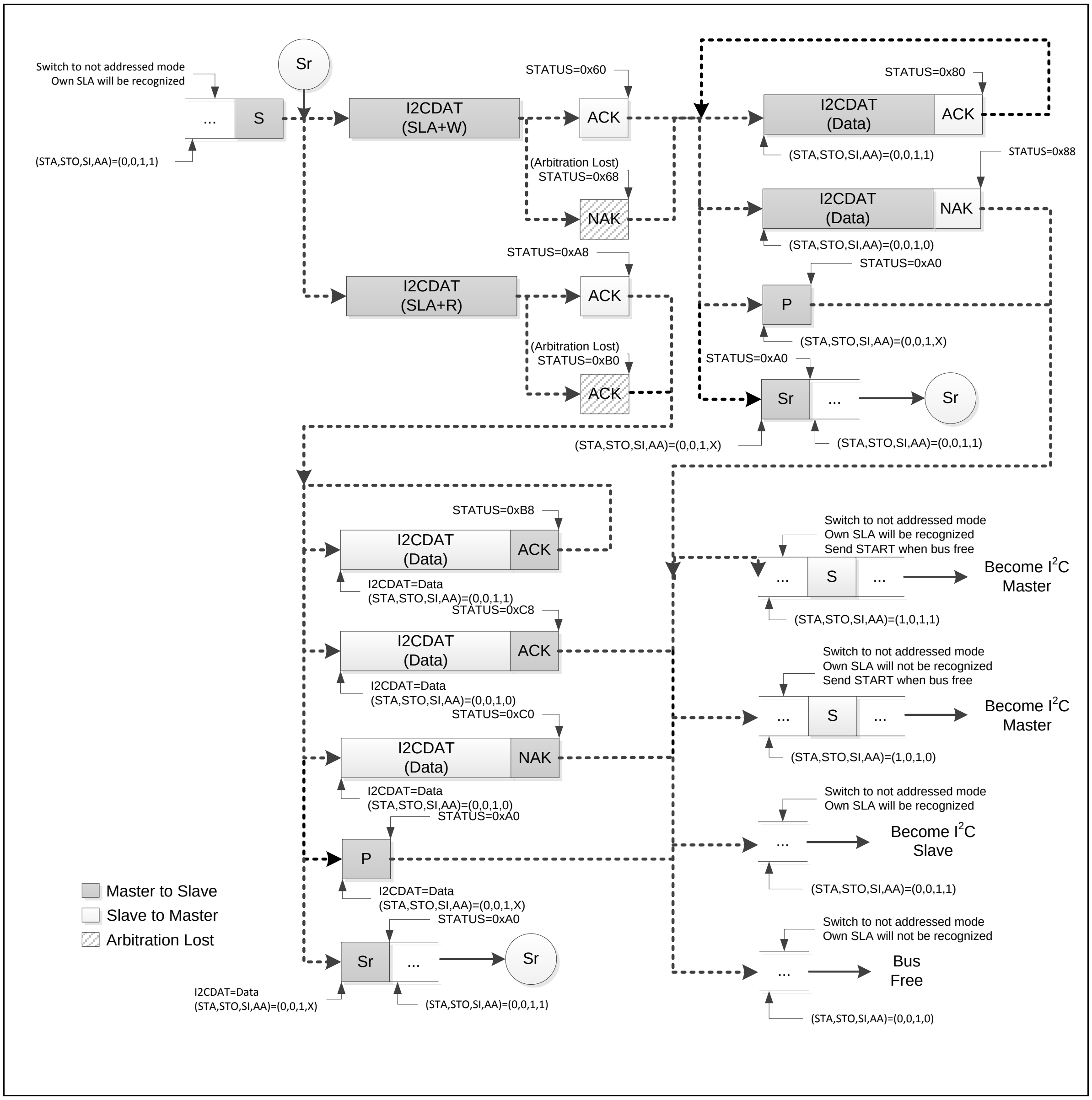

Switch to not addressed mode
Own SLA will be recognized
...
S
Sr
(STA,STO,SI,AA)=(0,0,1,1)
I2CDAT
(SLA+W)
STATUS=0x60
ACK
(Arbitration Lost)
STATUS=0x68
NAK
STATUS=0xA8
I2CDAT
(SLA+R)
ACK
(Arbitration Lost)
STATUS=0xB0
ACK
STATUS=0x80
I2CDAT
(Data)
ACK
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0x88
I2CDAT
(Data)
NAK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xA0
P
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xA0
Sr
...
Sr
(STA,STO,SI,AA)=(0,0,1,X)
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0xB8
I2CDAT
(Data)
ACK
I2CDAT=Data
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0xC8
I2CDAT
(Data)
ACK
I2CDAT=Data
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xC0
I2CDAT
(Data)
NAK
I2CDAT=Data
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xA0
P
I2CDAT=Data
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xA0
Sr
...
Sr
I2CDAT=Data
(STA,STO,SI,AA)=(0,0,1,X)
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will be recognized
Send START when bus free
...
S
...
Become I²C
Master
(STA,STO,SI,AA)=(1,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
Send START when bus free
...
S
...
Become I²C
Master
(STA,STO,SI,AA)=(1,0,1,0)
Switch to not addressed mode
Own SLA will be recognized
...
Become I²C
Slave
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
...
Bus
Free
(STA,STO,SI,AA)=(0,0,1,0)
Master to Slave
Slave to Master
Arbitration Lost

Figure 6.11-12 Save Mode Control Flow

If I^{2}C is still receiving data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x88 as shown in the above figure when getting 0xA0 status.

If I^{2}C is still transmitting data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0xC8 as shown in the above figure when getting 0xA0 status.

Note: After slave gets status of 0x88, 0xC8, 0xC0 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I^{2}C signal or address from master. At this status, I^{2}C should be reset to leave this status.

6.11.7.3 General Call (GC) Mode

If the GC bit (I2CADDRn [0]) is set, the I^{2}C port hardware will respond to General Call address (00H). User can clear GC bit to disable general call function. When the GC bit is set and the I^{2}C is in Slave mode, it can receive the general call address by 0x00 after master send general call address to I^{2}C bus, then it will follow status of GC mode.

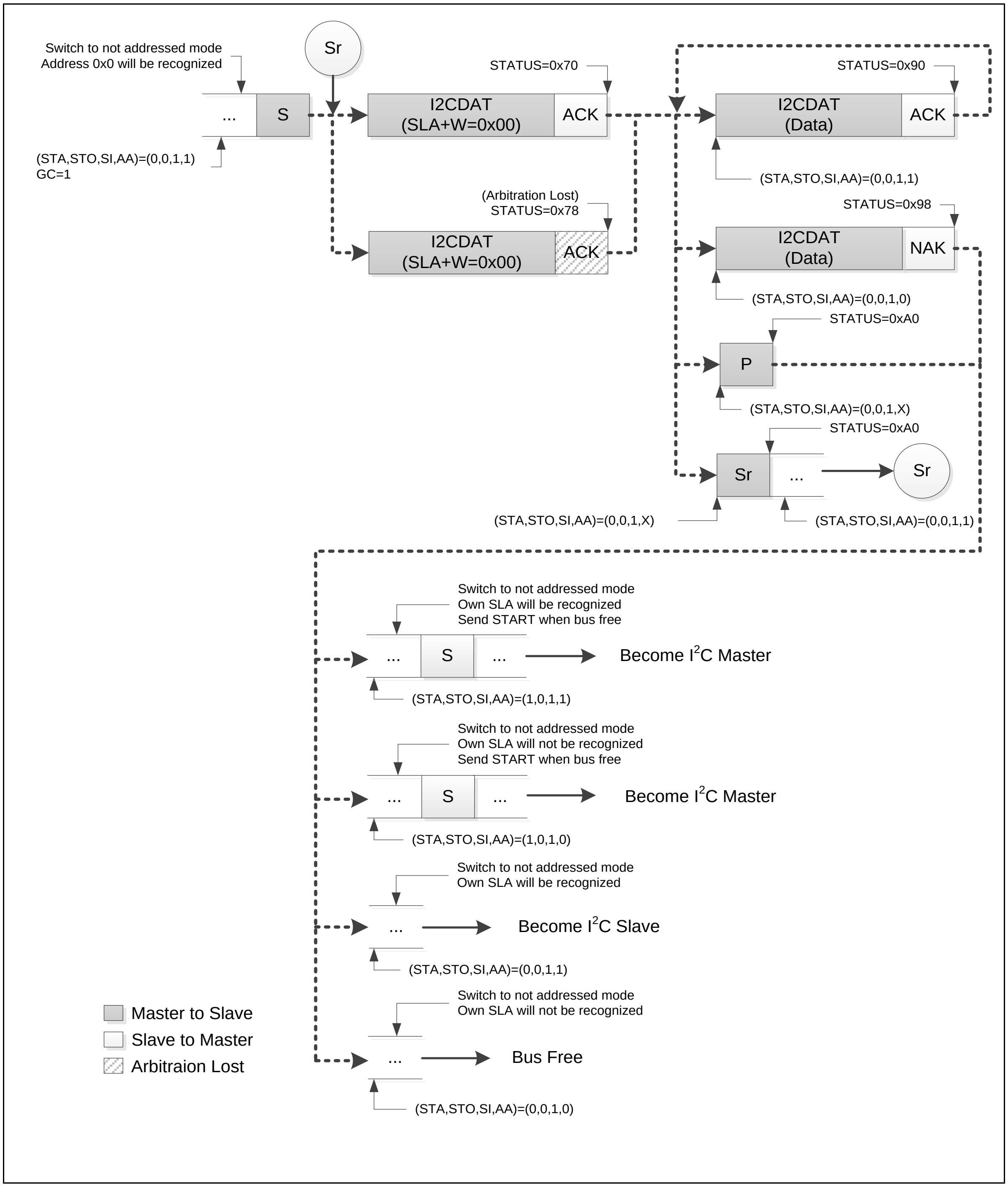

Switch to not addressed mode
Address 0x0 will be recognized
Sr
...
S
(STA,STO,SI,AA)=(0,0,1,1)
GC=1
STATUS=0x70
I2CDAT
(SLA+W=0x00)
ACK
(Arbitration Lost)
STATUS=0x78
I2CDAT
(SLA+W=0x00)
ACK
STATUS=0x90
I2CDAT
(Data)
ACK
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0x98
I2CDAT
(Data)
NAK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xA0
P
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xA0
Sr
...
Sr
(STA,STO,SI,AA)=(0,0,1,X)
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will be recognized
Send START when bus free
... S ...
Become I²C Master
(STA,STO,SI,AA)=(1,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
Send START when bus free
... S ...
Become I²C Master
(STA,STO,SI,AA)=(1,0,1,0)
Switch to not addressed mode
Own SLA will be recognized
...
Become I²C Slave
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
...
Bus Free
(STA,STO,SI,AA)=(0,0,1,0)
Master to Slave
Slave to Master
Arbitraion Lost

Figure 6.11-13 GC Mode

If I^{2}C is still receiving data in GC mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x98 in above figure when getting 0xA0 status.

Note: After slave gets status of 0x98 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I^{2}C signal or address from master. At this time, I^{2}C controller should be reset to leave this status.

6.11.7.4 Multi-Master

In some applications, there are two or more masters on the same I^{2}C bus to access slaves, and the masters may transmit data simultaneously. The I^{2}C in M058S Series supports multi-master by including collision detection and arbitration to prevent data corruption.

- When I2CSTATUS = 0x38, an "Arbitration Lost" is received. Arbitration lost event maybe occur during the send START bit, data bits or STOP bit. User could set (STA, STO, SI, AA) = (1, 0, 1, X) to send START again when bus free, or set (STA, STO, SI, AA) = (0, 0, 1, X) to send STOP to back to not addressed Slave mode.
- When I2CSTATUS = 0x00, a "Bus Error" is received. To recover I^{2}C bus from a bus error, STO should be set and SI should be cleared, and then STO is cleared to release bus.
  - Set (STA, STO, SI, AA) = (0, 1, 1, X) to stop current transfer
  - Set (STA, STO, SI, AA) = (0, 0, 1, X) to release bus

6.11.7.5 Example for Random Read on EEPROM

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

1. Set the multi-function pin in the "GP3_MFP" and "ALT_MFP" registers as SCL and SDA pins.
2. Enable I^{2}C APB clock, I2C_EN=1 in the "APBCLK" register.
3. Set I2C_RST=1 to reset I^{2}C controller then set I^{2}C controller to normal operation, I2C_RST=0 in the "IPRSTC2" register.
4. Set ENS1=1 to enable I^{2}C controller in the "I2CON" register.
5. Give I^{2}C clock a divided register value for I^{2}C clock rate in the "I2CLK".
6. Set SETENA=0x00040000 in the "NVIC_ISER" register to set I^{2}C IRQ.
7. Set EI=1 to enable I^{2}C Interrupt in the "I2CON" register.
8. Set I^{2}C address registers which are "I2CADDR0~I2CADDR3".

Random read operation is one of the methods of access EEPROM. The method allows the master to access any address of EEPROM space. The following figure shows the EEPROM random read operation.

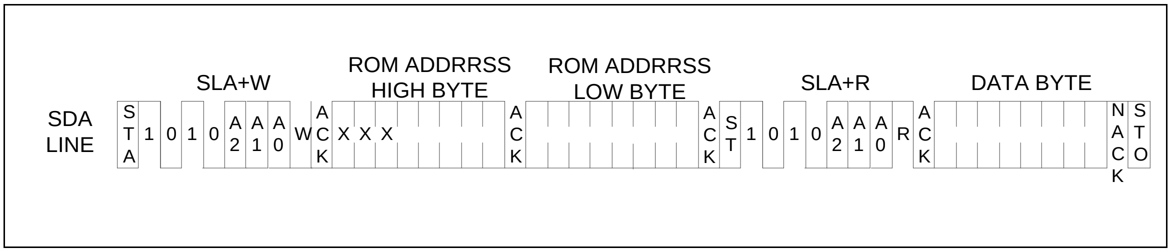


Figure 6.11-14 EEPROM Random Read

The following figure shows how to use I²C controller to implement the protocol of EEPROM random read.

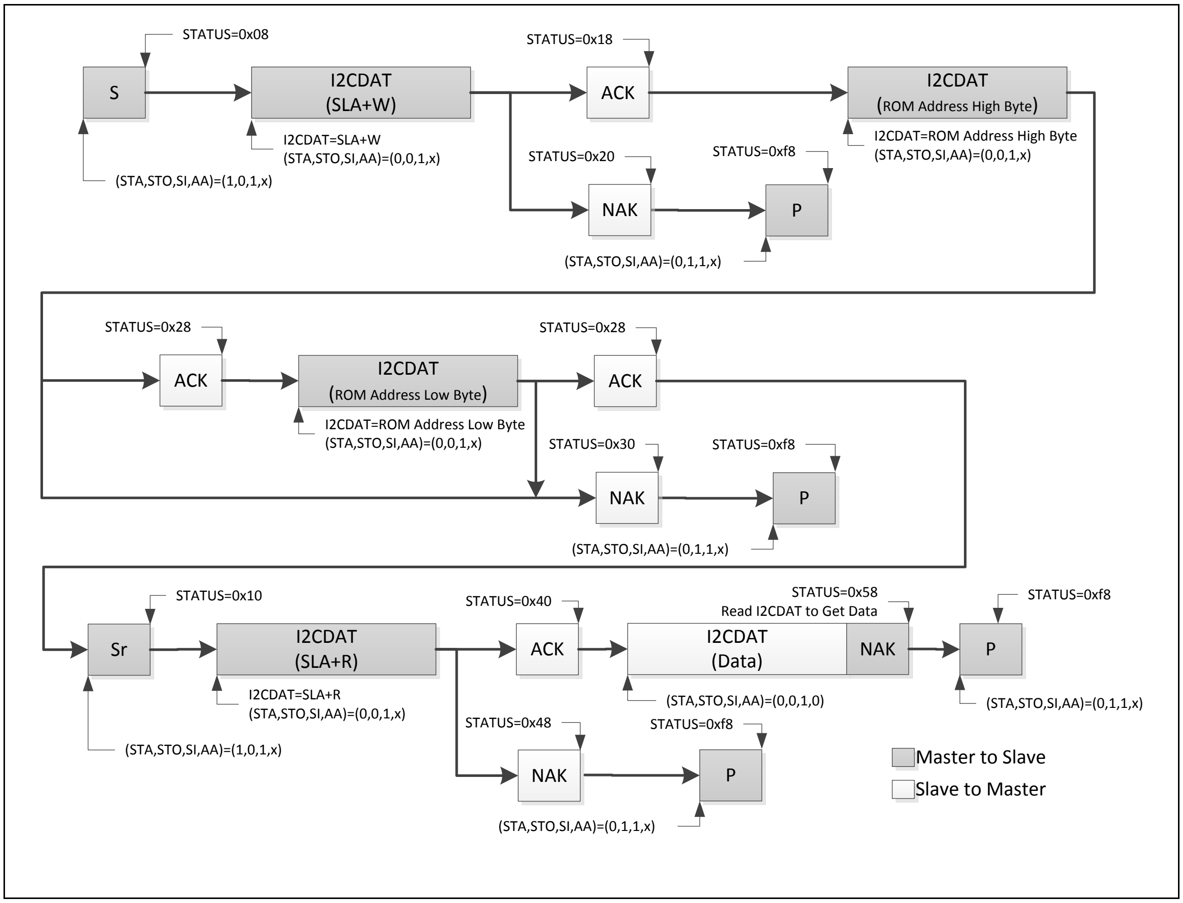


Figure 6.11-15 Protocol of EEPROM Random Read

The I²C controller sends START to bus to be a master. Then it sends a SLA+W (Slave address + Write bit) to EERPOM followed by two bytes data address to set the EEPROM address to read. Finally, a Repeat START followed by SLA+R is sent to read the data from EEPROM.

6.11.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
I^{2}C Base Address: **I2C0_BA = 0x4002_0000** **I2C1_BA = 0x4012_0000**				
I2CON x=0,1	I2Cx_BA+0x00	R/W	I^{2}C Control Register	0x0000_0000
I2CADDR0 x=0,1	I2Cx_BA+0x04	R/W	I^{2}C Slave Address Register0	0x0000_0000
I2CDAT x=0,1	I2Cx_BA+0x08	R/W	I^{2}C Data Register	0x0000_0000
I2CSTATUS x=0,1	I2Cx_BA+0x0C	R	I^{2}C Status Register	0x0000_00F8
I2CLK x=0,1	I2Cx_BA+0x10	R/W	I^{2}C Clock Divided Register	0x0000_0000
I2CTOC x=0,1	I2Cx_BA+0x14	R/W	I^{2}C Time-out Counter Register	0x0000_0000
I2CADDR1 x=0,1	I2Cx_BA+0x18	R/W	I^{2}C Slave Address Register1	0x0000_0000
I2CADDR2 x=0,1	I2Cx_BA+0x1C	R/W	I^{2}C Slave Address Register2	0x0000_0000
I2CADDR3 x=0,1	I2Cx_BA+0x20	R/W	I^{2}C Slave Address Register3	0x0000_0000
I2CADM0 x=0,1	I2Cx_BA+0x24	R/W	I^{2}C Slave Address Mask Register0	0x0000_0000
I2CADM1 x=0,1	I2Cx_BA+0x28	R/W	I^{2}C Slave Address Mask Register1	0x0000_0000
I2CADM2 x=0,1	I2Cx_BA+0x2C	R/W	I^{2}C Slave Address Mask Register2	0x0000_0000
I2CADM3 x=0,1	I2Cx_BA+0x30	R/W	I^{2}C Slave Address Mask Register3	0x0000_0000
I2CWKUPCON x=0,1	I2Cx_BA+0x3C	R/W	I^{2}C Wake-up Control Register	0x0000_0000

I2CWKUPSTS x=0,1	I2Cx_BA+0x40	R/W	I^{2}C Wake-up Status Register	0x0000_0000

6.11.9 Register Description

I^{2}C Control Register (I2CON)

Register	Offset	R/W	Description	Reset Value
I2CON	I2Cx_BA+0x00	R/W	I^{2}C Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EI	**ENS1**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**EI**	**I^{2}C Interrupt Enable Control** 0 = I^{2}C interrupt Disabled. 1 = I^{2}C interrupt Enabled.
[6]	**ENS1**	**I^{2}C Controller Enable Control** 0 = I^{2}C Controller Disabled. 1 = I^{2}C Controller Enabled. Set to enable I^{2}C serial function controller. When ENS1=1 the I^{2}C serial function enables. The function of multi-function pins must be set to I^{2}C first.
[5]	**STA**	**I^{2}C START Control** Setting STA to logic 1 to enter Master mode, the I^{2}C hardware sends a START or repeat START condition to bus when the bus is free.
[4]	**STO**	**I^{2}C STOP Control** In master mode, setting STO to transmit a STOP condition to bus then I^{2}C hardware will check the bus condition if a STOP condition is detected this bit will be cleared by hardware automatically. In Slave mode, setting STO resets I^{2}C 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**	**I^{2}C Interrupt Flag** When a new I^{2}C state is present in the I2CSTATUS register, the SI flag is set by hardware, and if bit EI (I2CON [7]) is set, the I^{2}C interrupt is requested. SI must be cleared by software. Clear SI is by writing 1 to it.
[2]	**AA**	**Assert Acknowledge Control** When AA=1 prior to address or data received, an acknowledged (low level to SDA)

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

I^{2}C Data Register (I2CDAT)

Register	Offset	R/W	Description	Reset Value
I2CDAT	I2Cx_BA+0x08	R/W	I^{2}C 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	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**I2CDAT**	**I^{2}C Data Register** Bit [7:0] is located with the 8-bit transferred data of I^{2}C serial port.

I^{2}C Status Register (I2CSTATUS)

Register	Offset	R/W	Description	Reset Value
I2CSTATUS	I2Cx_BA+0x0C	R	I^{2}C Status Register	0x0000_00F8

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**I2CSTATUS**	**I^{2}C Status** The three least significant bits are always 0. The five most significant bits contain the status code. There are 26 possible status codes. When I2CSTATUS contains F8H, no serial interrupt is requested. All other I2CSTATUS values correspond to defined I^{2}C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2CSTATUS one cycle after SI is set by hardware and is still present one cycle after SI has been reset by software. In addition, states 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an incorrect position in the formation frame. A bus error may occur during the serial transfer of an address byte, a data byte or an acknowledge bit.

I^{2}C Clock Divided Register (I2CLK)

Register	Offset	R/W	Description	Reset Value
I2CLK	I2Cx_BA+0x10	R/W	I^{2}C 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	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**I2CLK**	**I^{2}C Clock Divided** The I^{2}C clock rate bits: Data Baud Rate of I^{2}C = PCLK / (4x (I2CLK+1)). **Note:** The minimum value of I2CLK is 4.

I^{2}C Time-out Control Register (I2CTOC)

Register	Offset	R/W	Description	Reset Value
I2CTOC	I2Cx_BA+0x14	R/W	I^{2}C Time-out Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ENTI**	**DIV4**	**TIF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ENTI**	**Time-out Counter Enable Control** 0 = Time-out counter Disabled 1 = Time-out counter Enabled **Note:** When the 14 bit Time-out counter is enabled, it will start counting when SI is cleared. Writing 1 to the Si flag will reset the counter and re-start up counting after SI is cleared.
[1]	**DIV4**	**Time-out Counter Input Clock Divided by 4** 0 = Time-out counter input clock divided by 4 Disabled 1 = Time-out counter input clock divided by 4 Enabled **Note:** When Enabled, the time-out period is extended 4 times.
[0]	**TIF**	**Time-out Flag** This bit is set by hardware when I^{2}C time-out happened and it can interrupt CPU if I^{2}C interrupt enable bit (EI) is set to 1. **Note:** Software can write 1 to clear this bit.

I^{2}C Slave Address Register (I2CADDRx)

Register	Offset	R/W	Description	Reset Value
I2CADDR0	I2Cx_BA+0x04	R/W	I^{2}C Slave Address Register0	0x0000_0000
I2CADDR1	I2Cx_BA+0x18	R/W	I^{2}C Slave Address Register1	0x0000_0000
I2CADDR2	I2Cx_BA+0x1C	R/W	I^{2}C Slave Address Register2	0x0000_0000
I2CADDR3	I2Cx_BA+0x20	R/W	I^{2}C 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	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**I2CADDR**	**I^{2}C Address Register** The content of this register is irrelevant when I^{2}C is in Master mode. In Slave mode, the seven most significant bits must be loaded with the chip's own address. The I^{2}C hardware will react if one of the addresses is matched.
[0]	**GC**	**General Call Function Enable Control** 0 = General Call function Disabled. 1 = General Call Function Enabled.

I^{2}C Slave Address Mask Register (I2CADMx)

Register	Offset	R/W	Description	Reset Value
I2CADM0	I2Cx_BA+0x24	R/W	I^{2}C Slave Address Mask Register0	0x0000_0000
I2CADM1	I2Cx_BA+0x28	R/W	I^{2}C Slave Address Mask Register1	0x0000_0000
I2CADM2	I2Cx_BA+0x2C	R/W	I^{2}C Slave Address Mask Register2	0x0000_0000
I2CADM3	I2Cx_BA+0x30	R/W	I^{2}C Slave Address Mask Register3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CADM[7:1]							**Reserved**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**I2CADM**	**I^{2}C Address Mask** 0 = I^{2}C address mask Disabled (the received corresponding register bit should be exactly the same as address register). 1 = I^{2}C address mask Enabled (the received corresponding address bit is “Don’t care”).
[0]	**Reserved**	Reserved.

I^{2}C Wake-up Control Register (I2CWKUPCON)

Register	Offset	R/W	Description	Reset Value
I2CWKUPCON	I2Cx_BA+0x3C	R/W	I^{2}C Wake-up Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKUPEN**	**I^{2}C Wakeup Function Enable Control** 1 = I^{2}C wake up function Enabled. 0 = I^{2}C wake up function Disabled.

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

Register	Offset	R/W	Description	Reset Value
I2CWKUPSTS	I2Cx_BA+0x40	R/W	I^{2}C Wake-up Status Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKUPIF**	**I^{2}C Wake-up Interrupt Flag** When chip is woken up from Power-Down mode by I^{2}C, this bit is set to 1. Software can write 1 to clear this bit.

6.12 Serial Peripheral Interface (SPI)

6.12.1 Overview

The Serial Peripheral Interface (SPI) applies to synchronous serial data communication and allows full-duplex transfer. Devices communicate in Master/Slave mode with 4-wire bi-direction interface. The NuMicro M058S contains one set of SPI controllers performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. SPI controller can be configured as a master or a slave device.

6.12.2 Features

- One set of SPI controllers
- Supports Master or Slave mode operation
- Configurable transfer bit length
- Provides transmit/receive can be transferred up to two times word transaction in one transfer
- Provides FIFO buffers
- Supports MSB or LSB first transfer
- Supports byte reorder function
- Supports byte or word suspend mode
- Supports Slave 3-wire mode
- SPI bus clock rate can be configured to equal the system clock rate

6.12.3 Block Diagram

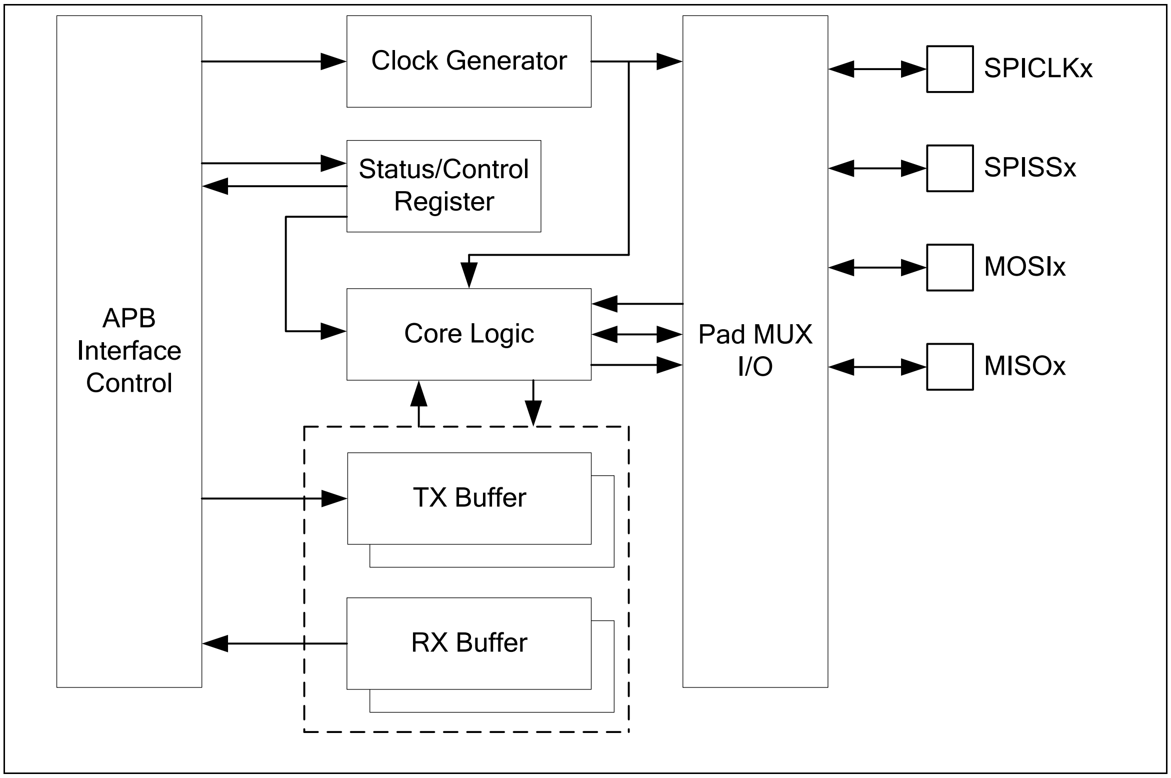


Figure 6.12-1 SPI Block Diagram

6.12.4 Basic Configuration

The SPI pin functions are configured in P1_MFP registers. The SPI0 peripheral clock can be enabled in APBCLK[12]. The SPI peripheral clock source is selected in CLKSEL1[4] and CLKSEL1[5].

6.12.5 Functional Description

6.12.5.1 Terminology

SPI Peripheral Clock and SPI Bus Clock

The SPI controller needs the SPI peripheral clock to drive the SPI logic unit to perform the data transfer. The SPI bus clock is the clock presented on SPICLKx pin.

In M058S SPI Master mode and Slave mode, the SPI peripheral clock frequency is determined by the settings of clock source, BCn (SPI_CNTRL2[31]) option and clock divisor (SPI_DIVIDER[7:0]). The SPIx_S bit of CLKSEL1 register determines the clock source of the SPI peripheral clock. The clock source can be HCLK or PLL output clock. Set the BCn bit to 0 for the compatible SPI clock frequency calculation of previous products. The DIVIDER setting of SPI_DIVIDER register determines the divisor of the clock frequency calculation.

In M058S SPI Master mode, the SPI peripheral clock is equal to the SPI bus clock.

In M058S SPI Slave mode, the SPI bus clock is provided by an off-chip master device. The SPI peripheral clock frequency of slave device must be faster than the bus clock frequency of the master device connected together. The frequency of SPI peripheral clock cannot be faster than the APB clock frequency regardless of Master mode or Slave mode.

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 or Slave mode are shown below.

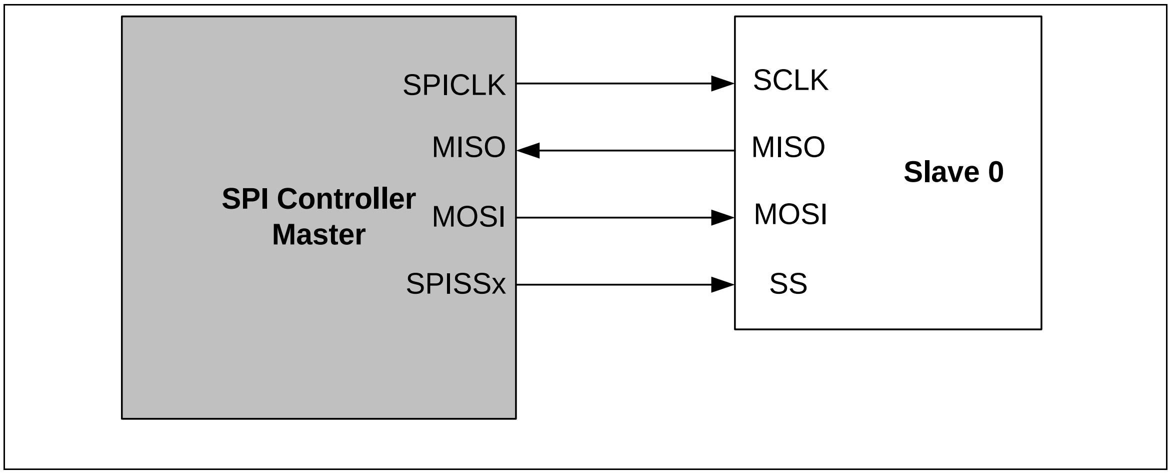


Figure 6.12-2 SPI Master Mode Application Block Diagram

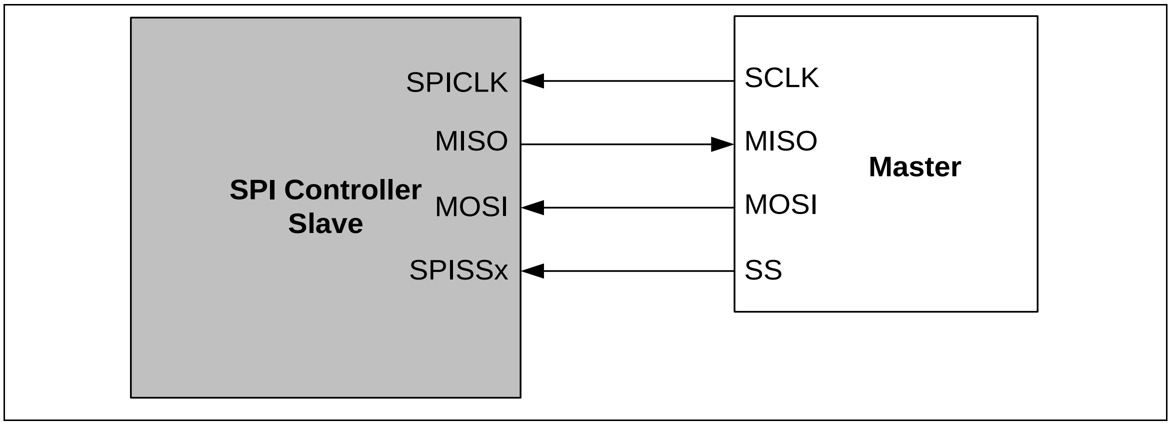


Figure 6.12-3 SPI Slave Mode Application Block Diagram

Clock Polarity

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

Transmit/Receive Bit Length

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

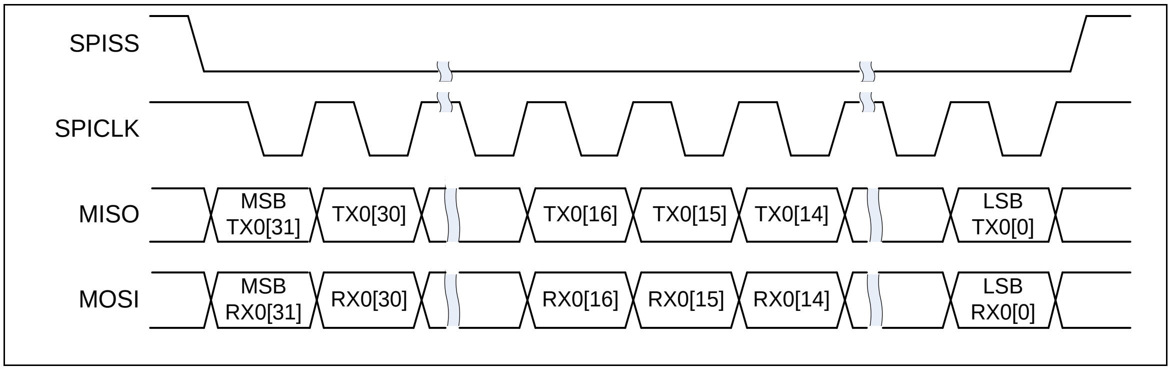


Figure 6.12-4 32-bit in One Transaction

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 on negative edge or on positive edge of bus clock SPICLK.

Receive Edge

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

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

Word Suspend

The four bits field of SP_CYCLE (SPI_CNTRL[15:12]) provide a configurable suspend interval between two successive transaction words in Master mode. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word.

The default value of SP_CYCLE is 0x3 (3.5 bus clock cycles), and the SP_CYCLE setting will not take effect to the word suspend interval if software disables the FIFO mode.

Slave Selection

In Master mode, each SPI controller can drive one off-chip slave device through the slave select output pin, SPISSx. In Slave mode, the off-chip master device drives the slave select signal from the SPISSx input pin to this SPI controller. In Master/Slave mode, the active state of slave select signal can be programmed to low active or high active in SS_LVL bit (SPI_SSR[2]), and the SS_LTRIG bit (SPI_SSR[4]) defines the slave select signal SPISSx is level trigger or edge trigger. The selection of trigger condition depends on what type of peripheral slave/master device is connected.

In Slave mode, if the SS_LTRIG bit is configured as level trigger, the LTRIG_FLAG bit (SPI_SSR[5]) is used to indicate if both the number of received data and the number of received bits meet the requirement which defined in TX_NUM and TX_BIT_LEN among one transaction done (the transaction done means the unit transfer interrupt flag is set to 1 when the slave select signal is inactivated or the SPI controller finishes one data transfer.)

Level-trigger / Edge-trigger

In Slave mode, the slave select signal can be configured as level-trigger or edge-trigger. In edge-trigger, the data transfer starts from an active edge of the slave select signal and ends on an inactive edge of the slave select signal. The unit-transfer interrupt flag (SPI_CNTRL[16]) will be set to 1 as an inactive edge is detected. If master does not send an inactive edge to slave, the transfer procedure will not be completed and the unit-transfer interrupt flag of slave will not be set. In level-trigger, the unit-transfer interrupt flag of slave will be set when one of the following two conditions occurs. The first condition is that if the number of transferred bits matches the settings TX_BIT_LEN, the unit-transfer interrupt flag of slave will be set. The second condition, if master set the slave select pin to inactive level during the transfer in progress, it will force slave device to terminate the current transfer no matter how many bits have been transferred and the unit-transfer interrupt flag will be set. User can read the status of LTRIG_FLAG bit to check if the data has been completely transferred.

6.12.5.2 Automatic Slave Selection

In Master mode, if the AUTOSS bit (SPI_SSR[3]) is set, the slave select signal will be generated automatically and output to SPISSx pin according to SSR bit (SPI_SSR[0]) whether be enabled or not. This means that the slave select signal will be asserted by the SPI controller when data transfer is started by setting the GO_BUSY bit (SPI_CNTRL[0]) and will be de-asserted after the data transfer is finished. If the AUTOSS bit is cleared, the slave select output signal will be asserted/de-asserted by setting/clearing the SSR bit. The active state of the slave select output signal is specified in SS_LVL bit (SPI_SSR[2]).

6.12.5.3 Byte Reorder Function

When the transfer is set as MSB first (LSB = 0) and the byte reorder function is enabled, the data stored in the TX buffer and RX buffer will be rearranged in the order as [Byte0, Byte1, Byte2, Byte3] when the bit length is configured as 32-bit (TX_BIT_LEN=0). The sequence of transmitted/received data will be Byte0, Byte1, Byte2, and then Byte3. If the TX_BIT_LEN bit field is set to 24, the data in TX buffer and RX buffer will be rearranged as [unknown byte, Byte0, Byte1, Byte2]. The SPI controller will transmit/receive data with the sequence of Byte0, Byte1 and then Byte2. Each byte will be transmitted/received with MSB first. The rule of 16-bit mode is the same as above. Byte reorder function is only available when TX_BIT_LEN is configured as 16, 24, or 32 bits.

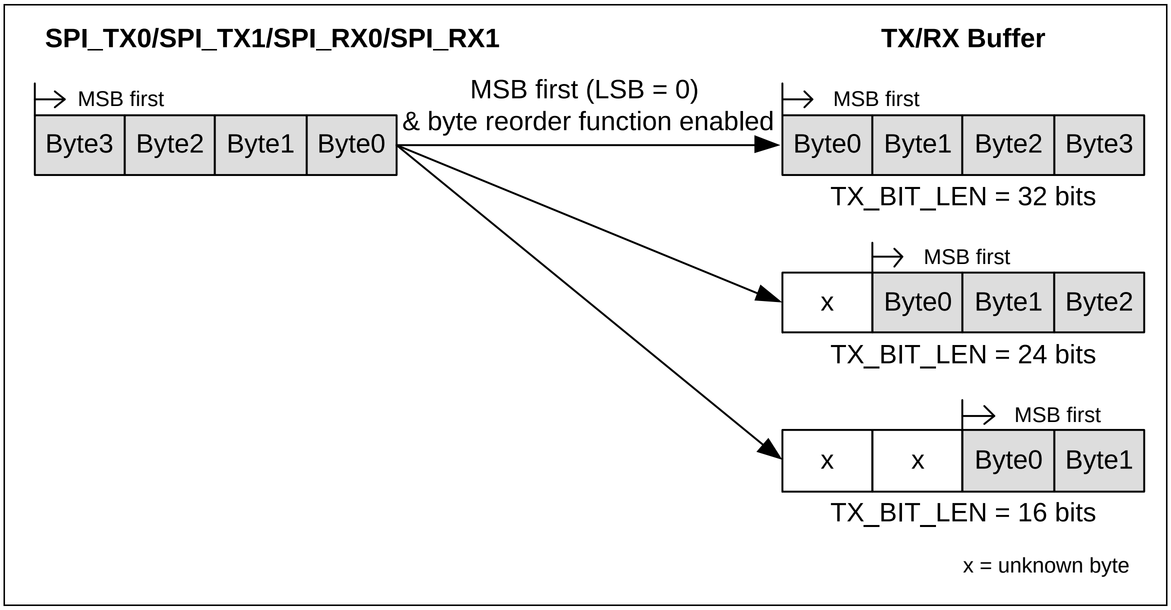


Figure 6.12-5 Byte Reorder

6.12.5.4 Byte Suspend Function

Both settings of byte suspend interval and word suspend interval are configured in SP_CYCLE.

In Master mode, if the byte reorder function is enabled by setting SPI_CNTRL[19] to 1, the hardware will insert a suspend interval of 0.5 ~ 15.5 bus clock periods between two successive bytes in a transaction word. The setting of TX_BIT_LEN can be configured as 16, 24 or 32 bits.

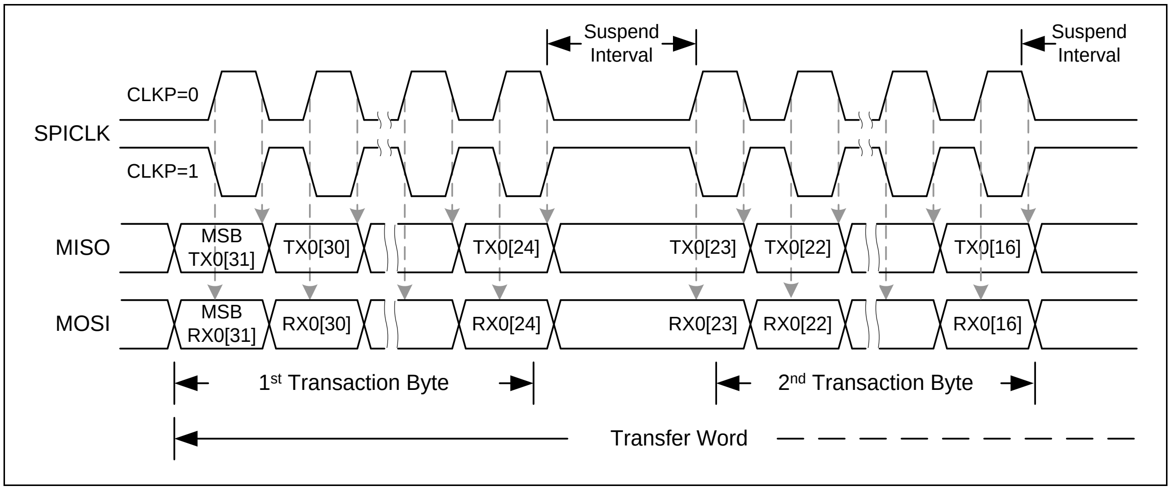


Figure 6.12-6 Timing Waveform for Byte Suspend

6.12.5.5 Slave 3-wire Mode

When the NOSLVSEL bit (SPI_CNTRL2[8]) is set by software to enable the Slave 3-wire mode, the SPI controller can work with no slave select signal in Slave mode. The NOSLVSEL bit only takes effect in Slave mode. Only three pins, SPICLK, MISO, and MOSI, are required to communicate with a SPI master. The SPISS pin can be configured as a GPIO. When the NOSLVSEL bit is set to 1, the SPI slave will be ready to transmit/receive data after the GO_BUSY bit is set to 1. As the number of received bits meets the requirement which defined in TX_BIT_LEN and TX_NUM, the unit-transfer interrupt flag, IF (SPI_CNTRL[16]), will be set to 1.

Note: In Slave 3-wire mode, the SS_LTRIG bit (SPI_SSR[4]) should be set as 1.

6.12.5.6 FIFO Mode

The SPI controllers support FIFO mode when the FIFO bit, SPI_CNTRL[21], is set as 1. The SPI controllers equip with four 32-bit wide transmit and receive FIFO buffers.

The transmit FIFO buffer is a 4-layer depth, 32-bit wide, first-in, first-out register buffer. The software can write data to the transmit FIFO buffer by writing the SPI_TX0 register. The data stored in the transmit FIFO buffer will be read and sent out by the transmission control logic. If the 4-layer transmit FIFO buffer is full, the TX_FULL bit will be set to 1. When the SPI transmission logic unit draws out the last datum of the transmit FIFO buffer, so that the 4-layer transmit FIFO buffer is empty, the TX_EMPTY bit will be set to 1. Note that the TX_EMPTY flag is set to 1 while the last transaction is still in progress.

The received FIFO buffer is also a 4-layer depth, 32-bit wide, first-in, first-out register buffer. The receive control logic will store the received data to this buffer. The software can read the FIFO buffer data from SPI_RX0 register. There are FIFO related status bits, like RX_EMPTY and RX_FULL, to indicate the current status of FIFO buffer.

In FIFO mode, the software can set the transmitting and receiving threshold by setting the TX_THRESHOLD and RX_THRESHOLD settings. When the count of valid data stored in transmit FIFO buffer is less than or equal to TX_THRESHOLD setting, the TX_INTSTS bit will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RX_THRESHOLD setting, the RX_INTSTS bit will be set to 1.

In FIFO mode, the software can write 4 data to the SPI transmit FIFO buffer in advance. When the SPI controller operates with FIFO mode, the GO_BUSY bit of SPI_CNTRL register will be controlled by hardware, software should not modify the content of SPI_CNTRL register unless clearing the FIFO bit to disable the FIFO mode.

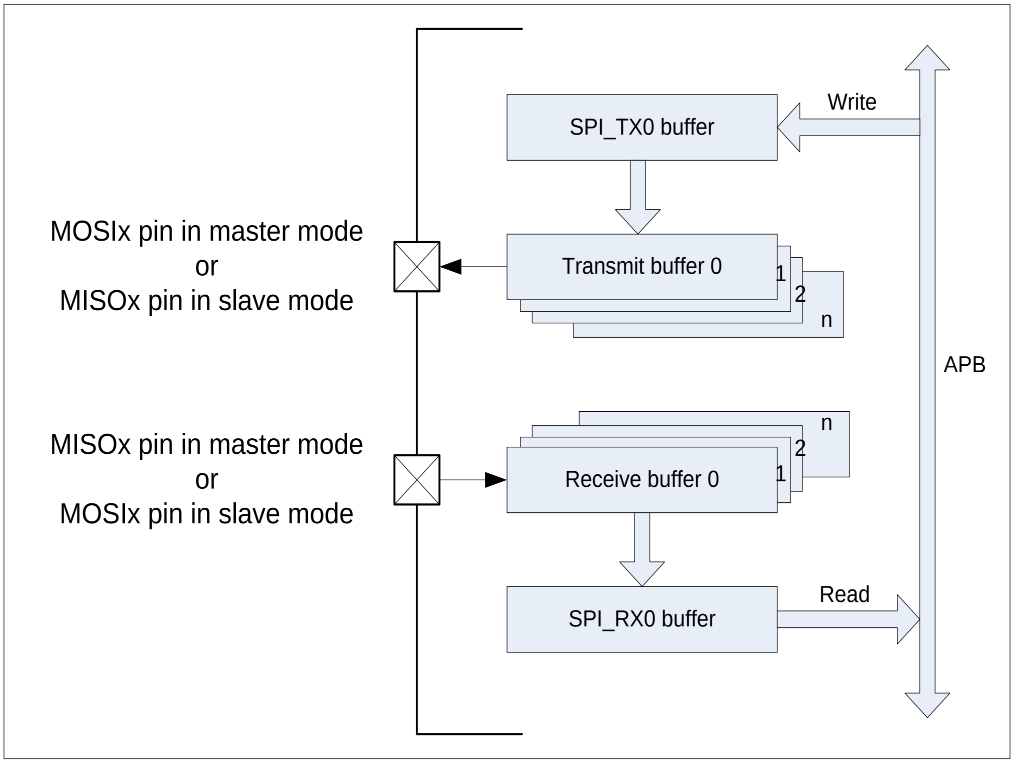


Figure 6.12-7 FIFO Mode Block Diagram

In Master mode transmission operation, the TX_EMPTY flag will be cleared to 0 when the FIFO bit is set to 1 and the software write the first datum to the SPI_TX0 register. The transmission starts immediately as long as the transmit FIFO buffer is not empty. User can write the next data into SPI_TX0 register immediately. The SPI controller will insert a suspend interval between two successive transactions in FIFO mode and the period of suspend interval is decided by the setting of SP_CYCLE (SPI_CNTRL [15:12]). User can write data into SPI_TX0 register as long as the TX_FULL flag is 0.

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the SPI_TX0 register is not updated after all data transfer are done, the transfer will stop.

In Master mode reception operation, the serial data are received from MISOx pin and stored to receive FIFO buffer. The RX_EMPTY flag will be cleared to 0 while the receive FIFO buffer contain unread data. The software can read the received data from SPI_RX0 register as long as the RX_EMPTY flag is 0. If the receive FIFO buffer contain 4 unread data, the RX_FULL flag will be set to 1. The SPI controller will stop receiving data until the software read the SPI_RX0 register.

In Slave mode, when the FIFO bit is set as 1, the GO_BUSY bit will be set as 1 by hardware automatically. If user wants to stop the Slave mode SPI data transfer, both the FIFO bit and

GO_BUSY bit must be cleared to 0 by software.

In Slave mode transmission operation, when the software writes data to SPI_TX0 register, the data will be loaded into transmit FIFO buffer and the TX_EMPTY flag will be set to 0. The transmission will start when the slave device receives clock signal from master. The software can write data to SPI_TX0 register as long as TX_FULL flag is 0. After all data have been drawn out by the SPI transmission logic unit and the software does not update the SPI_TX0 register, the TX_EMPTY flag will be set to 1.

In Slave mode reception operation, the serial data is received from MOSIx pin and stored to SPI_RX0 register. The reception mechanism is similar to Master mode reception operation.

6.12.5.7 Interrupt

SPI unit-transfer interrupt

As the SPI controller finishes a unit transfer, the unit-transfer interrupt flag IF (SPI_CNTRL[16]) will be set to 1. The unit-transfer interrupt event will generate an interrupt to CPU if the unit-transfer interrupt enable bit IE (SPI_CNTRL[17]) is set. The unit-transfer interrupt flag can be cleared only by writing 1 to it.

SPI Slave 3-wire mode start interrupt

In Slave 3-wire mode, the Slave 3-wire mode start interrupt flag, SLV_START_INTSTS (SPI_CNTRL2[11]), will be set to 1 when the slave senses the SPI bus clock signal. The SPI controller will issue an interrupt if the SSTA_INTEN bit (SPI_CNTRL2[10]) is set to 1. If the count of the received bits is less than the bit length setting of TX_BIT_LEN and there is no more SPI bus clock input over the expected time period which is defined by user, user can set the SLV_ABORT bit (SPI_CNTRL2[9]) to abort the current transfer. The unit-transfer interrupt flag, IF, will be set to 1 if the SLV_ABORT bit is set to 1 by software.

Receive FIFO time-out interrupt

In FIFO mode, there is time-out function to inform user. If there is a received data in the FIFO and it does not be read by software over 64 SPI engine clock periods in Master mode or over 576 SPI engine clock periods in Slave mode, it will send a time-out interrupt to the system if the time-out interrupt enable bit, FIFO_CTL[21], is set to 1.

Transmit FIFO interrupt

In FIFO mode, if the valid data count of the transmit FIFO buffer is less than or equal to the setting value of TX_THRESHOLD, the transmit FIFO interrupt flag will be set to 1. The SPI controller will generate a transmit FIFO interrupt to the system if the transmit FIFO interrupt enable bit, SPI_FIFO_CTL[3], is set to 1.

Receive FIFO interrupt

In FIFO mode, if the valid data count of the receive FIFO buffer is larger than the setting value of RX_THRESHOLD, the receive FIFO interrupt flag will be set to 1. The SPI controller will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, SPI_FIFO_CTL[2], is set to 1.

6.12.6 Timing Diagram

The active state of slave select signal can be defined by the settings of SS_LVL bit (SPI_SSR[2]) and SS_LTRIG bit (SPI_SSR[4]). The bus clock (SPICLK) idle state can be configured as high state or low state by setting the CLKP bit (SPI_CNTRL[11]). It also provides the bit length of a transaction word in TX_BIT_LEN bit field (SPI_CNTRL[7:3]), the transaction number in TX_NUM bit field (SPI_CNTRL[9:8]), and transmit/receive data from MSB or LSB first in LSB bit (SPI_CNTRL[10]). User also can select which edge of bus clock to transmit/receive data in TX_NEG/RX_NEG (SPI_CNTRL[2]/SPI_CNTRL[1]) bit. Four SPI timing diagrams for master/slave operations and the related settings are shown below.

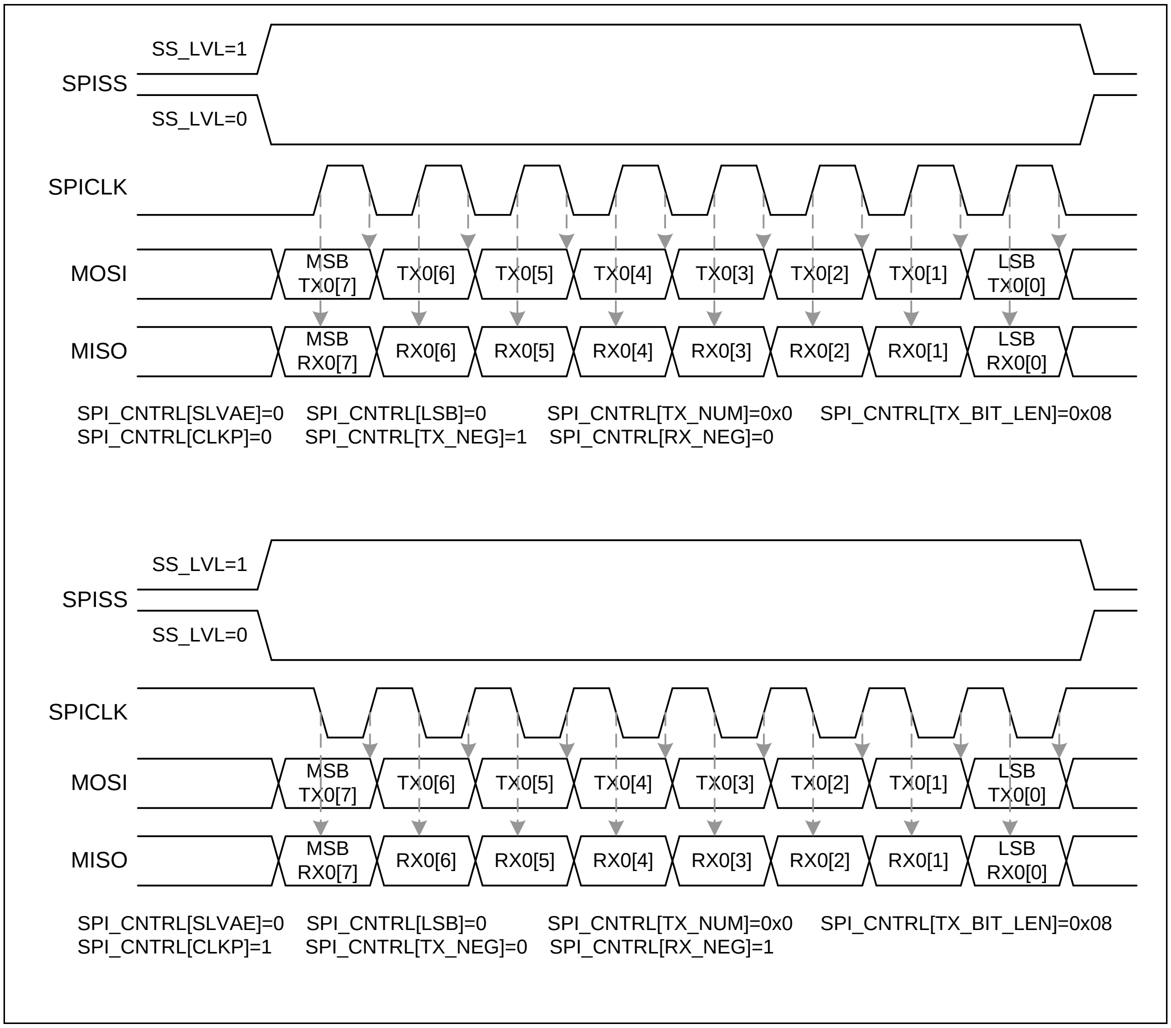


Figure 6.12-8 SPI Timing in Master Mode

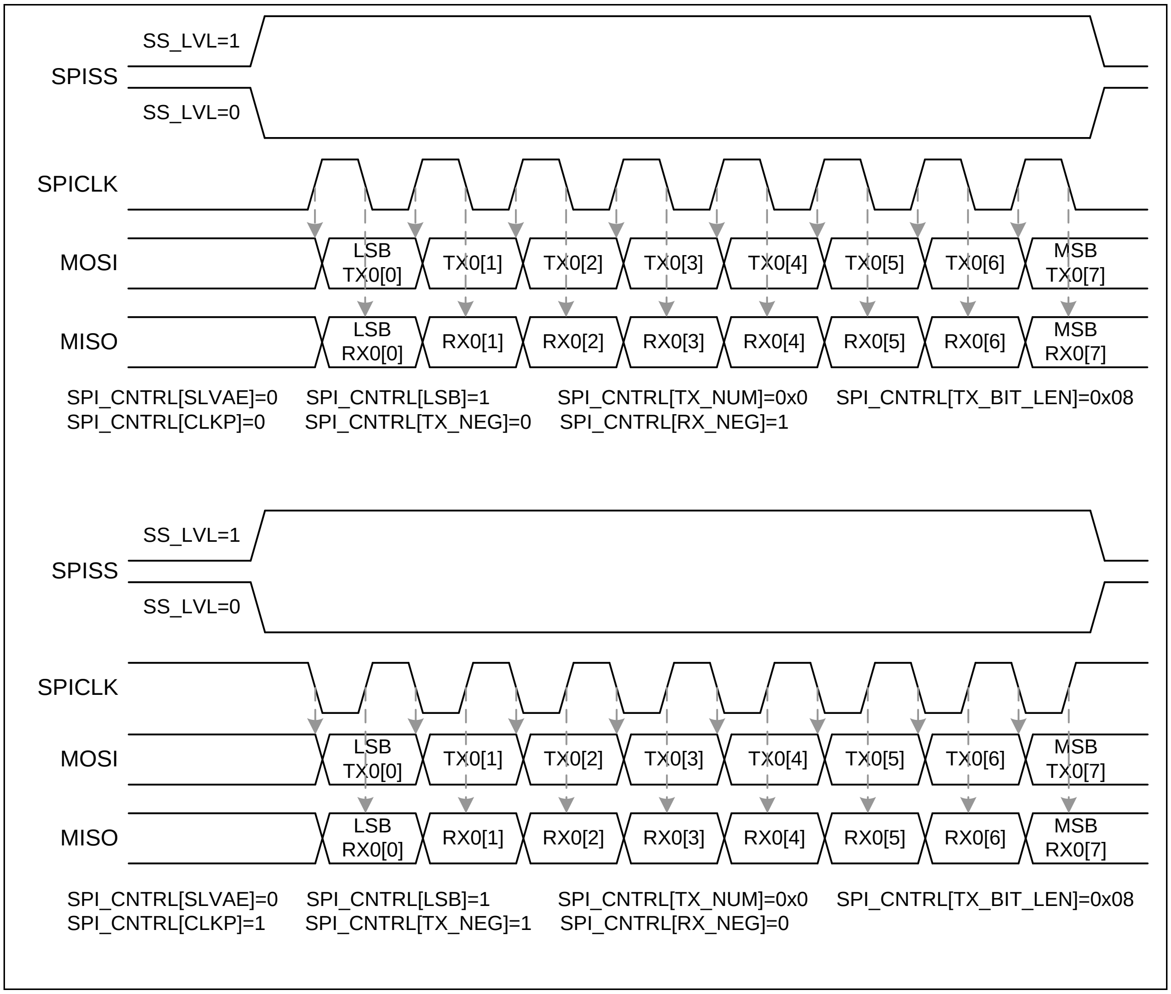


Figure 6.12-9 SPI Timing in Master Mode (Alternate Phase of SPICLK)

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

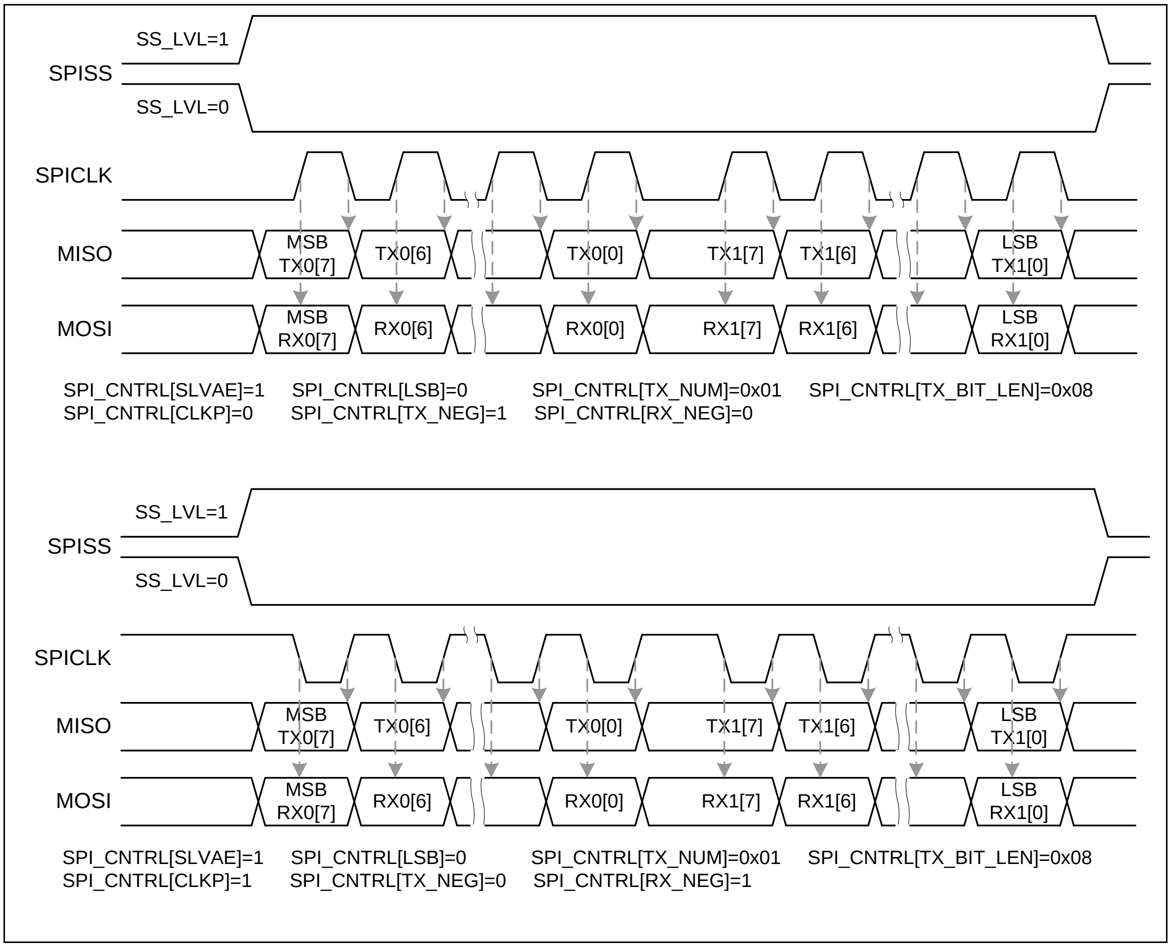


Figure 6.12-10 SPI Timing in Slave Mode

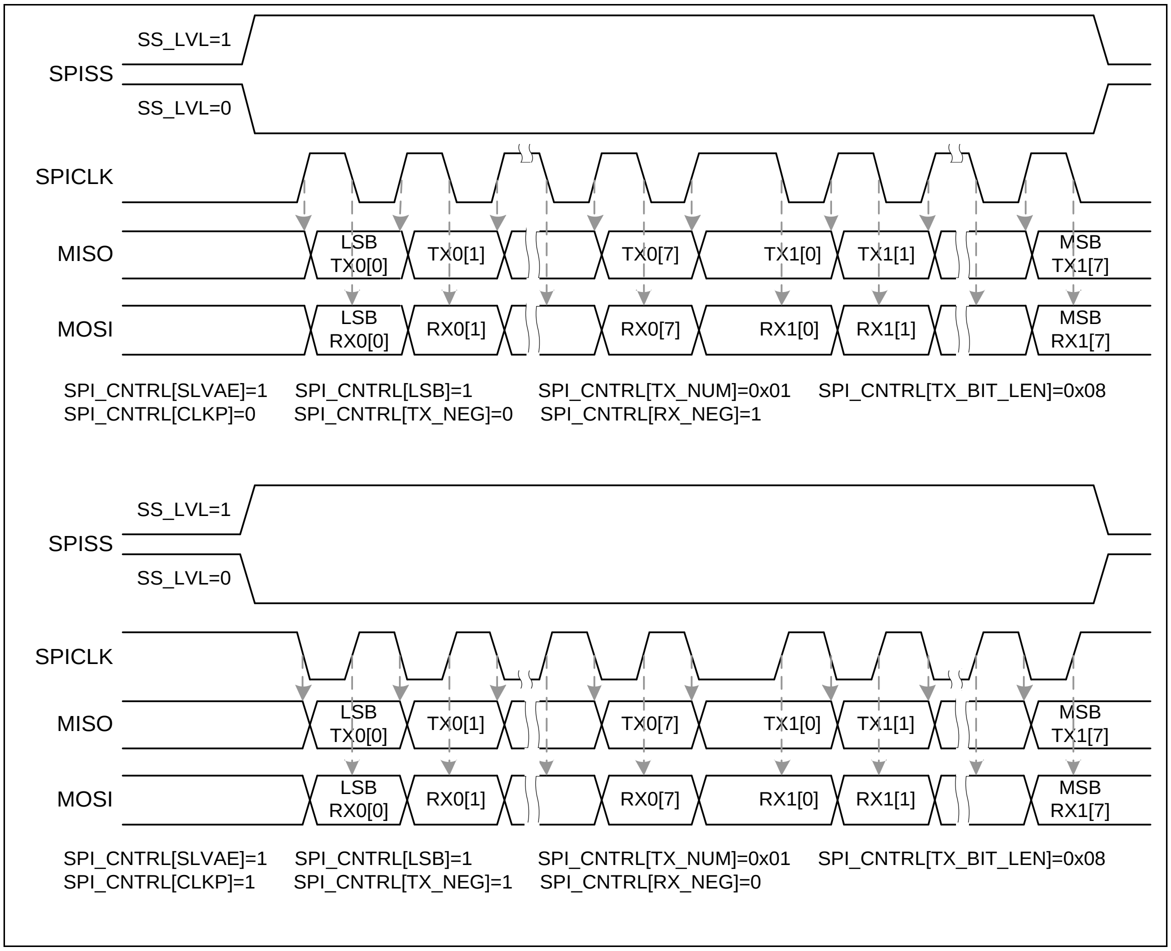


Figure 6.12-11 SPI Timing in Slave Mode (Alternate Phase of SPICLK)

6.12.7 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 bus clock
- Data bit is driven on negative edge of bus clock
- Data is transferred from MSB first
- Data width is 8 bits
- SPICLK is idle at low state
- Only one byte of data to be transmitted/received in a transaction
- Connect with an off-chip slave device. Slave select signal is active low

The operation flow is described below.

1) Set the DIVIDER (SPI_DIVIDER[7:0]) to determine the output frequency of bus clock.
2) Write the related settings into the SPI_CNTRL register to control the SPI master actions.

1. Set this SPI controller as master device in SLAVE bit (SPI_CNTRL[18] = 0).
2. Force the bus clock idle state at low in CLKP bit (SPI_CNTRL[11] = 0).
3. Select data transmitted on negative edge of bus clock in TX_NEG bit (SPI_CNTRL[2] = 1).
4. Select data latched on positive edge of bus clock in RX_NEG bit (SPI_CNTRL[1] = 0).
5. Set the bit length of word transfer as 8-bit in TX_BIT_LEN bit field (SPI_CNTRL[7:3] = 0x08).
6. Set MSB transfer first in MSB bit (SPI_CNTRL[10] = 0), and need not to care the SP_CYCLE bit field (SPI_CNTRL[15:12]) due to it is not in FIFO mode in this case.

3) Write the SPI_SSR register a proper value for the related settings of Master mode.

   1. Clear the Automatic Slave Select bit AUTOSS(SPI_SSR[3] = 0).

      Select low level trigger output of slave select signal in the Slave Select Active Level control bit, SS_LVL (SPI_SSR[2] = 0).

   2. Set the slave select signal to be active by setting the Slave Select control bit SSR (SPI_SSR[0]) to active the off-chip slave device.

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

5) If this SPI master just only attempts to receive (read) one byte data from the off-chip slave device and does not care what data will be transmitted, the SPI_TX0 register does not need to be updated by software.

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

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

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

9) Go to 4) to continue another data transfer or set SSR[0] bit to 0 to inactivate the off-chip slave device.

The following is a sample based on M058S series BSP for the SPI data transfer described above.

```
/* Set module clock divider. SPI clock rate = HCLK / ((11+1)*2) = 2MHz */
SPI0->DIVIDER = SPI0->DIVIDER & (~SPI_DIVIDER_DIVIDER_Msk) | SPI_DIVIDER_DIV(11);
/* Configure SPI0 as a master, MSB first, clock idle low, falling clock edge Tx, rising edge Rx and 8-bit transaction */
SPI0->CNTRL   =   SPI_CNTRL_MASTER_MODE   |   SPI_CNTRL_MSB_FIRST   |   SPI_CNTRL_CLK_IDLE_LOW   |
                  SPI_CNTRL_TX_FALLING | SPI_CNTRL_RX_RISING | SPI_CNTRL_TX_BIT_LEN(8);
/* Disable the automatic hardware slave select function. Select the SS pin and configure as low-active. */
SPI0->SSR = SPI_SSR_SW_SS_PIN_LOW;
/* write the first data of source buffer to Tx register of SPI0. */
_SPI_WRITE_TX_BUFFER0(SPI0, SourceData[0]);
/* Start the data transfer */
_SPI_SET_GO(SPI0);
/* Check the GO_BUSY bit */
While(SPI0->CNTRL & SPI_CNTRL_GO_BUSY);
/* Read the received data */
DestData[0] = _SPI_GET_RX0_DATA(SPI0)&0xFF;
…
```

Example 2: The SPI controller is set as a slave device and connects with an off-chip master device. The off-chip master device communicates with the on-chip SPI slave controller through the SPI interface with the following specifications:

- Data bit is latched on positive edge of bus clock
- Data bit is driven on negative edge of bus clock
- Data is transferred from LSB first
- Data width is 8 bits
- SPICLK is idle at high state
- Only one byte of data to be transmitted/received in a transaction
- Slave select signal is high level trigger

The operation flow is described below.

1) Set the DIVIDER (SPI_DIVIDER[7:0]) to determine the slave peripheral clock frequency. The slave peripheral clock frequency must be larger than the SPI bus clock frequency.

2) 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 control bit SS_LVL (SPI_SSR[2] = 1) and the Slave Select Level Trigger bit SS_LTRIG (SPI_SSR[4] = 1).

3) 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 bus clock idle state at high in CLKP bit (SPI_CNTRL[11] = 1).
    3. Select data transmitted on negative edge of bus clock in TX_NEG bit (SPI_CNTRL[2] = 1).
    4. Select data latched on positive edge of bus clock in RX_NEG bit (SPI_CNTRL[1] = 0).
    5. Set the bit length of word transfer as 8-bit in TX_BIT_LEN bit field (SPI_CNTRL[7:3] = 0x08).
    6. Set only one time of word transfer in TX_NUM (SPI_CNTRL[9:8] = 0x0).
    7. Set LSB transfer first in LSB bit (SPI_CNTRL[10] = 1), and need not to care the SP_CYCLE bit field (SPI_CNTRL[15:12]) due to it is not burst mode in this case.

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

5) If this SPI slave just only attempts to receive (be written) one byte data from the off-chip master device and does not care what data will be transmitted, the SPI_TX0 register does not need to be updated by software.

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

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

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

9) Go to 4) to continue another data transfer or clear the GO_BUSY bit to stop data transfer.

The following is a sample based on M058S series BSP for the SPI data transfer described above.

```
/* Configure SPI0 as a high level active device. */
SPI0->SSR = SPI_SSR_SLAVE_HIGH_LEVEL_ACTIVE;
/* Configure SPI0 as a slave, LSB first, clock idle high, falling clock edge Tx, rising edge Rx and 8-bit transaction */
SPI0->CNTRL = SPI_CNTRL_SLAVE_MODE | SPI_CNTRL_LSB_FIRST | SPI_CNTRL_CLK_IDLE_HIGH |
              SPI_CNTRL_TX_FALLING | SPI_CNTRL_RX_RISING | SPI_CNTRL_TX_BIT_LEN(8);
/* write the first data of source buffer to Tx register of SPI0. */
_SPI_WRITE_TX_BUFFER0(SPI0, SourceData[0]);
/* set the GO_BUSY bit of SPI0 */
_SPI_SET_GO(SPI0);
/* Check the GO_BUSY bit */
While(SPI0->CNTRL & SPI_CNTRL_GO_BUSY);
/* Read the received data */
DestData[0] = _SPI_GET_RX0_DATA(SPI0)&0xFF;
…
```

6.12.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI Base Address: **SPI0_BA = 0x4003_0000**				
SPI_CNTRL	SPI0_BA+0x00	R/W	Control and Status Register	0x0500_0004
SPI_DIVIDER	SPI0_BA+0x04	R/W	Clock Divider Register	0x0000_0000
SPI_SSR	SPI0_BA+0x08	R/W	Slave Select Register	0x0000_0000
SPI_RX0	SPI0_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPI_TX0	SPI0_BA+0x20	W	Data Transmit Register 0	0x0000_0000
SPI_CNTRL2	SPI0_BA+0x3C	R/W	Control and Status Register 2	0x0000_0000
SPI_FIFO_CTL	SPI0_BA+0x40	R/W	SPI FIFO Control Register	0x2200_0000
SPI_STATUS	SPI0_BA+0x44	R/W	SPI Status Register	0x0500_0000

6.12.9 Register Description

SPI Control and Status Register (SPI_CNTRL)

Register	Offset	R/W	Description	Reset Value
SPI_CNTRL	SPIx_BA+0x00	R/W	Control and Status Register	0x0500_0004

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**TX_FULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** It is a mutual mirror bit of SPI_STATUS[27]. 0 = Indicates that the transmit FIFO buffer is not full. 1 = Indicates that the transmit FIFO buffer is full.
[26]	**TX_EMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** It is a mutual mirror bit of SPI_STATUS[26]. 0 = Indicates that the transmit FIFO buffer is not empty. 1 = Indicates that the transmit FIFO buffer is empty.
[25]	**RX_FULL**	**Receive FIFO Buffer Full Indicator (Read Only)** It is a mutual mirror bit of SPI_STATUS[25]. 0 = Indicates that the receive FIOF buffer is not full. 1 = Indicates that the receive FIFO buffer is full.
[24]	**RX_EMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** It is a mutual mirror bit of SPI_CNTRL[24]. 0 = Indicates that the receive FIFO buffer is not empty. 1 = Indicates that the receive FIFO buffer is empty.
[23:22]	**Reserved**	Reserved.
[21]	**FIFO**	**FIFO Mode Enable Control** In Master mode, if the FIFO mode is enabled, the GO_BUSY bit will be set to 1 automatically after writing data into the 8-depth transmit FIFO. When all data stored at transmit FIFO buffer are transferred, the GO_BUSY bit will back to 0. 0 = FIFO Mode Disabled. 1 = FIFO Mode Enabled.

		Note 1: Before enabling FIFO mode, the other related settings should be set in advance.
[20:19]	**REORDER**	**Byte Reorder Function and Byte Suspend Function Selection** 00 = Byte reorder function and byte suspend function are Disabled. 01 = Byte reorder function Enabled. Byte suspend interval is determined by the setting of SP_CYCLE. Set SP_CYCLE to 0 to disabled byte suspend function. 10 = Reserved. 11 = Reserved. **Note:** Byte reorder function is only available if TX_BIT_LEN is defined as 16, 24, and 32 bits.
[18]	**SLAVE**	**Slave Mode Control** 0 = Master mode. 1 = Slave mode.
[17]	**IE**	**Unit-Transfer Interrupt Enable Control** 0 = SPI unit-transfer interrupt Disabled. 1 = SPI unit-transfer interrupt Enabled.
[16]	**IF**	**Unit-Transfer Interrupt Flag** 0 = No transaction has been finished since this bit was cleared to 0. 1 = SPI controller has finished one unit transfer. **Note:** This bit will be cleared by writing 1 to itself.
[15:12]	**SP_CYCLE**	**Suspend Interval (Master Only)** These four bits provide configurable suspend interval between two successive transactions in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation: (SP_CYCLE[3:0] + 0.5) * period of SPICLK clock cycle Example: SP_CYCLE = 0x0 … 0.5 SPICLK clock cycle SP_CYCLE = 0x1 … 1.5 SPICLK clock cycle …… SP_CYCLE = 0xE … 14.5 SPICLK clock cycle SP_CYCLE = 0xF … 15.5 SPICLK clock cycle
[11]	**CLKP**	**Clock Polarity** 0 = SPICLK idle low. 1 = SPICLK idle high.
[10]	**LSB**	**LSB First** 0 = The MSB is transmitted/received first. 1 = The LSB is transmitted/received first.
[9:8]	**Reserved**	Reserved.
[7:3]	**TX_BIT_LEN**	**Transfer Bit Length** This field specifies how many bits can be transmitted / received in one transaction. The minimum bit length is 8 bits and can up to 32 bits.

		TX_BIT_LEN = 0x08 … 8 bit
		TX_BIT_LEN = 0x09 … 9 bits
		……
		TX_BIT_LEN = 0x1F … 31 bits
		TX_BIT_LEN = 0x00 … 32 bits

[2]	**TX_NEG**	**Transmit on Negative Edge** 0 = The transmitted data output signal is driven on the rising edge of SPICLK. 1 = The transmitted data output signal is driven on the falling edge of SPICLK.
[1]	**RX_NEG**	**Receive on Negative Edge** 0 = The received data input signal is latched on the rising edge of SPICLK. 1 = The received data input signal is latched on the falling edge of SPICLK.
[0]	**GO_BUSY**	**SPI Transfer Trigger and Busy Status** In FIFO mode, this bit will be controlled by hardware. Software cannot modify this bit. If FIFO mode is disabled, during the data transfer, this bit keeps the value of 1. As the transfer is finished, this bit will be cleared automatically. 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. **Note 1:** When FIFO mode is disabled, all configurations should be ready before writing 1 to the GO_BUSY bit. **Note 2:** In SPI Slave mode, if FIFO mode is disabled and the SPI bus clock is kept at idle state during a data transfer, the GO_BUSY bit will not be cleared to 0 when slave select signal goes to inactive state.

SPI Divider Register (SPI_DIVIDER)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DIVIDER**	**Clock Divider** Only DIVIDER[7:0] is available. The value in this field is the frequency divider to determine the SPI peripheral clock frequency, f_{spi}, and the SPI master's bus clock frequency on the SPICLK output pin. The frequency is obtained according to the following equation. If the bit of BCn, SPI_CNTRL2[31], is set to 0, $$f_{spi} = \frac{f_{SPI\_clock\_src}}{(DIVIDER+1)*2}$$ else if BCn is set to 1, $$f_{spi} = \frac{f_{SPI\_clock\_src}}{(DIVIDER+1)}$$ where $f_{SPI\_clock\_src}$ is the SPI peripheral clock source which is defined in the CLKSEL1 register.

SPI Slave Select Register (SPI_SSR)

Register	Offset	R/W	Description	Reset Value
SPI_SSR	SPIx_BA+0x08	R/W	Slave Select Register	0x0000_0000

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

Bits	Description	
[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. 0 = The transaction number or the transferred bit length of one transaction does not meet the specified requirements. 1 = The transaction number and the transferred bit length met the specified requirements which defined in TX_NUM and TX_BIT_LEN. The TX_NUM setting is not available on M058S, only the setting of TX_BIT_LEN will be considered. **Note:** This bit is READ only and only available in Slave mode.
[4]	**SS_LTRIG**	**Slave Select Level Trigger Enable (Slave Only)** 0 = The input slave select signal is edge-trigger. This is the default value. It depends on SS_LVL to decide the signal is active at falling-edge or rising-edge. 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 Function Enable (Master Only)** 0 = If this bit is cleared, slave select signal will be asserted/de-asserted by setting /clearing SSR bit. 1 = If this bit is set, SPISSx signal will be generated automatically, which means that slave select signal will be asserted by the SPI controller when transmit/receive is started by setting GO_BUSY, and will be de-asserted after each transmit/receive is finished.
[2]	**SS_LVL**	**Slave Select Active Level** This bit defines the active status of slave select signal (SPISSx). 0 = The slave select signal SPISSx is active at low-level/falling-edge. 1 = The slave select signal SPISSx is active at high-level/rising-edge.
[1]	**Reserved**	Reserved.
[0]	**SSR**	**Slave Select Control (Master Only)**

		If AUTOSS bit is cleared to 0, 0 = Set the SPISSx line to inactive state. 1 = Set the proper SPISSx line to active state. If AUTOSS bit is set to 1, 0 = Keep the SPISSx line at inactive state. 1 = Select the SPISSx line to be automatically driven to active state for the duration of transmission/reception, and will be driven to inactive state for the rest of the time. The active state of SPISSx is specified in SS_LVL bit.

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

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	Description	
[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 , bit RX0[7:0] holds the received data. The values of the other bits are unknown. 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

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	Description	
[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 , the bit TX0[7:0] will be transmitted in next transfer.

SPI Control and Status Register 2 (SPI_CNTRL2)

Register	Offset	R/W	Description	Reset Value
SPI_CNTRL2	SPIx_BA+0x3C	R/W	Control and Status Register 2	0x0000_0000

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

Bits	Description	
[31]	**BCn**	**Clock Configuration Backward Compatible Option** 0 = The clock configuration is backward compatible to M05xxBN. 1 = The clock configuration is not backward compatible to M05xxBN. Refer to the description of SPI_DIVIDER register for details.
[30:17]	**Reserved**	Reserved.
[16]	**SS_INT_OPT**	**Slave Select Inactive Interrupt Option** This setting is only available if the SPI controller is configured as level trigger slave device. 0 = As the slave select signal goes to inactive level, the IF bit will NOT be set to 1. 1 = As the slave select signal goes to inactive level, the IF bit will be set to 1.
[15:12]	**Reserved**	Reserved.
[11]	**SLV_START_INTSTS**	**Slave 3-wire Mode Start Interrupt Status** This bit dedicates if a transaction has started in Slave 3-wire mode. It is a mutual mirror bit of SPI_STATUS[11]. 0 = Slave does not detect any SPI bus clock transition since the SSTA_INTEN bit was set to 1. 1 = A transaction has started in Slave 3-wire mode. It will be cleared automatically when a transaction is done or by writing 1 to this bit.
[10]	**SSTA_INTEN**	**Slave 3-wire Mode Start Interrupt Enable** It is used to enable interrupt when the transfer has started in Slave 3-wire mode. If there is no transfer done interrupt over the time period which is defined by user after the transfer start, user can set the SLV_ABORT bit to force the transfer done. 0 = Transaction start interrupt Disabled. 1 = Transaction start interrupt Enabled. It will be cleared to 0 as the current transfer is done or the SLV_START_INTSTS bit is cleared to 0.
[9]	**SLV_ABORT**	**Slave 3-wire Mode Abort Control** In normal operation, there is an interrupt event when the number of received bits meets the requirement which defined in TX_BIT_LEN and TX_NUM. The TX_NUM setting is not available on M058S, and only the setting of TX_BIT_LEN will be considered. If the number of received bits is less than the requirement and there is no more

		bus clock input over one transfer time in Slave 3-wire mode, user can set this bit to force the current transfer done and then user can get a unit transfer interrupt event. **Note:** This bit will be cleared to 0 automatically by hardware after it is set to 1 by software.
[8]	**NOSLVSEL**	**Slave 3-wire Mode Enable** This is used to ignore the slave select signal in Slave mode. The SPI controller can work with 3-wire interface including SPICLK, SPI_MISO, and SPI_MOSI when this bit is set to 1. 0 = The controller is 4-wire bi-direction interface. 1 = The controller is 3-wire bi-direction interface in Slave mode. When this bit is set to 1, the controller will be ready to transmit/receive data after the GO_BUSY bit is set to 1. **Note:** In Slave 3-wire mode, the SS_LTRIG bit (SPI_SSR[4]) shall be set as 1.
[7:0]	**Reserved**	Reserved.

SPI FIFO Control Register (SPI_FIFO_CTL)

Register	Offset	R/W	Description	Reset Value
SPI_FIFO_CTL	SPIx_BA+0x40	R/W	SPI FIFO Control Register	0x2200_0000

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

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:28]	**TX_THRESHOLD**	**Transmit FIFO Threshold** If the valid data count of the transmit FIFO buffer is less than or equal to the TX_THRESHOLD setting, the TX_INTSTS bit will be set to 1, else the TX_INTSTS bit will be cleared to 0.
[27:26]	**Reserved**	Reserved.
[25:24]	**RX_THRESHOLD**	**Received FIFO Threshold** If the valid data count of the receive FIFO buffer is larger than the RX_THRESHOLD setting, the RX_INTSTS bit will be set to 1, else the RX_INTSTS bit will be cleared to 0.
[23:22]	**Reserved**	Reserved.
[21]	**TIMEOUT_INTEN**	**Receive FIFO Time-out Interrupt Enable Control** 0 = Time-out interrupt Disabled. 1 = Time-out interrupt Enabled.
[20:7]	**Reserved**	Reserved.
[6]	**RXOV_INTEN**	**Receive FIFO Overrun Interrupt Enable Control** 0 = Receive FIFO overrun interrupt Disabled. 1 = Receive FIFO overrun interrupt Enabled.
[5:4]	**Reserved**	Reserved.
[3]	**TX_INTEN**	**Transmit Threshold Interrupt Enable Control** 0 = Transmit threshold interrupt Disabled. 1 = Transmit threshold interrupt Enabled.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

[2]	**RX_INTEN**	**Receive Threshold Interrupt Enable Control** 0 = Receive threshold interrupt Disabled. 1 = Receive threshold interrupt Enabled.
[1]	**TX_CLR**	**Clear Transmit FIFO Buffer** 0 = No effect. 1 = Clear transmit FIFO buffer. The TX_FULL flag will be cleared to 0 and the TX_EMPTY flag will be set to 1. This bit will be cleared to 0 by hardware after software sets it to 1 and the transmit FIFO is cleared.
[0]	**RX_CLR**	**Clear Receive FIFO Buffer** 0 = No effect. 1 = Clear receive FIFO buffer. The RX_FULL flag will be cleared to 0 and the RX_EMPTY flag will be set to 1. This bit will be cleared to 0 by hardware after software sets it to 1 and the receive FIFO is cleared.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

SPI Status Register (SPI_STATUS)

Register	Offset	R/W	Description	Reset Value
SPI_STATUS	SPIx_BA+0x44	R/W	SPI Status Register	0x0500_0000

31	30	29	28	27	26	25	24
TX_FIFO_COUNT				TX_FULL	TX_EMPTY	RX_FULL	RX_EMPTY
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			TIMEOUT	Reserved			IF
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RX_FIFO_COUNT				SLV_START_INTSTS	Reserved		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			TX_INTSTS	Reserved	RX_OVERRUN	Reserved	RX_INTSTS

Bits	Description	
[31:28]	**TX_FIFO_COUNT**	**Transmit FIFO Data Count (Read Only)** Indicates the valid data count of transmit FIFO buffer.
[27]	**TX_FULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** It is a mutual mirror bit of SPI_CNTRL[27]. 0 = The transmit FIFO buffer is not full. 1 = The transmit FIFO buffer is full.
[26]	**TX_EMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** It is a mutual mirror bit of SPI_CNTRL[26]. 0 = The transmit FIFO buffer is not empty. 1 = The transmit FIFO buffer is empty.
[25]	**RX_FULL**	**Receive FIFO Buffer Full Indicator (Read Only)** It is a mutual mirror bit of SPI_CNTRL[25]. 0 = The receive FIFO buffer is not full. 1 = The receive FIFO buffer is full.
[24]	**RX_EMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** It is a mutual mirror bit of SPI_CNTRL[24]. 0 = The receive FIFO buffer is not empty. 1 = The receive FIFO buffer is empty.
[23:21]	**Reserved**	Reserved.
[20]	**TIMEOUT**	**Time-out Interrupt Flag** 0 = No receive FIFO time-out event. 1 = The receive FIFO buffer is not empty and it does not be read over 64 SPI

		clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. **Note:** This bit will be cleared by writing 1 to itself.
[19:17]	**Reserved**	Reserved.
[16]	**IF**	**SPI Unit-Transfer Interrupt Flag** It is a mutual mirror bit of SPI_CNTRL[16]. 0 = The transfer does not finish yet. 1 = The SPI controller has finished one unit transfer. **Note:** This bit will be cleared by writing 1 to itself.
[15:12]	**RX_FIFO_COUNT**	**Receive FIFO Data Count (Read Only)** Indicates the valid data count of receive FIFO buffer.
[11]	**SLV_START_INTSTS**	**Slave Start Interrupt Status** It is used to dedicate that the transfer has started in Slave 3-wire mode. It's a mutual mirror bit of SPI_CNTRL2[11]. 0 = The transfer is not started. 1 = The transfer has started in Slave 3-wire mode. It will be cleared as transfer done or by writing one to this bit.
[10:5]	**Reserved**	Reserved.
[4]	**TX_INTSTS**	**Transmit FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TX_THRESHOLD. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TX_THRESHOLD. **Note:** If TX_INTEN = 1 and TX_INTSTS = 1, the SPI controller will generate a SPI interrupt request.
[3]	**Reserved**	Reserved.
[2]	**RX_OVERRUN**	**Receive FIFO Overrun Status** When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1. **Note:** This bit will be cleared by writing 1 to itself.
[1]	**Reserved**	Reserved.
[0]	**RX_INTSTS**	**Receive FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the Rx FIFO buffer is smaller than or equal to the setting value of RX_THRESHOLD. 1 = The valid data count within the receive FIFO buffer is larger than the setting value of RX_THRESHOLD. **Note:** If RX_INTEN = 1 and RX_INTSTS = 1, the SPI controller will generate a SPI interrupt request.

6.13 Analog-to-Digital Converter (ADC)

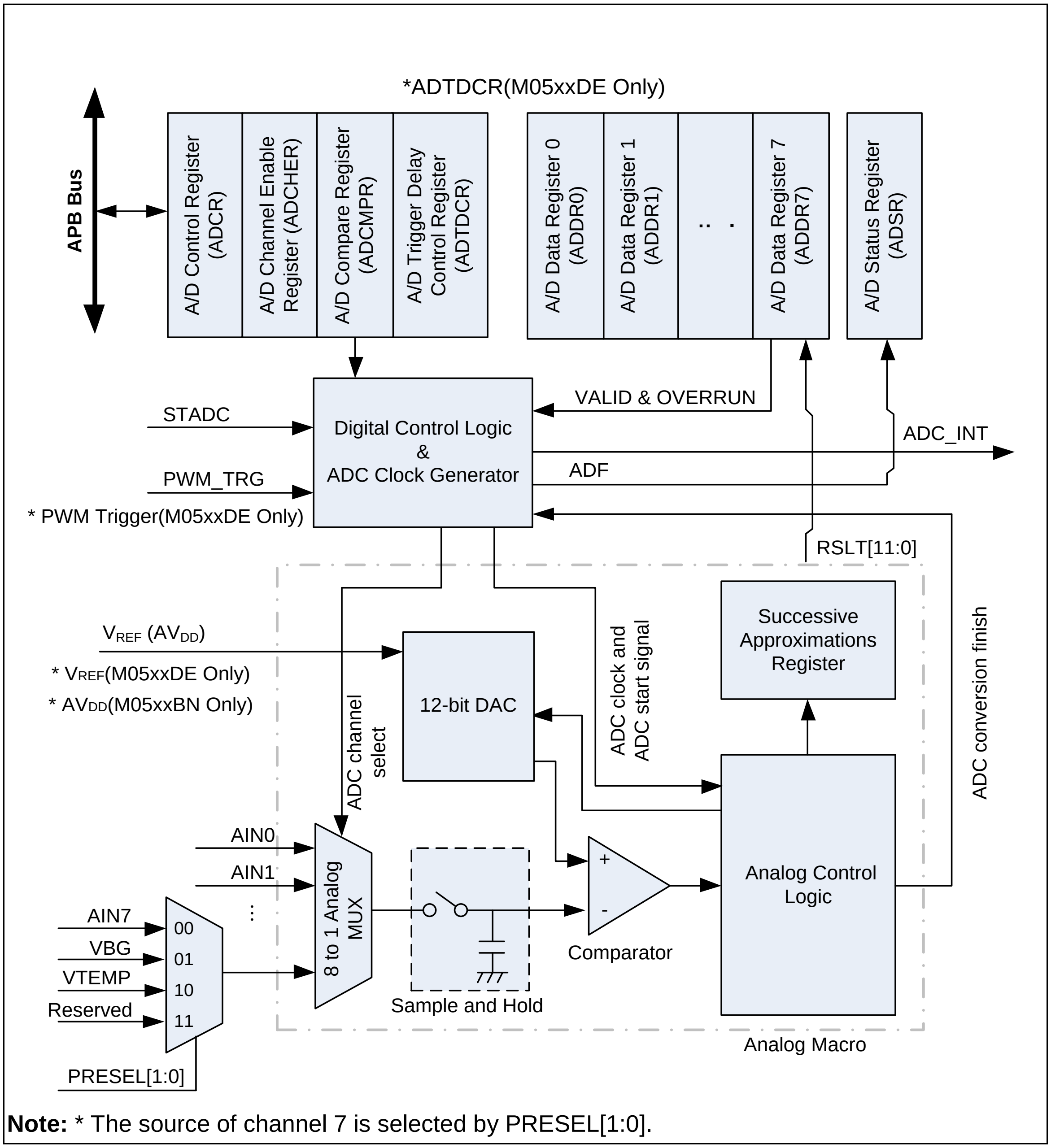
6.13.1 Overview

The NuMicro™ M058S series contains one 12-bit successive approximation analog-to-digital converter (SAR A/D converter) with eight input channels. The A/D converter supports four operation modes: Single, Burst, Single-cycle Scan and Continuous Scan mode. The A/D converter can be started by software, external pin (STADC/P3.2) or PWM trigger.

6.13.2 Features

- Analog input voltage range: 0 ~ AV_{DD}.
- 12-bit resolution and 10-bit accuracy is guaranteed
- Up to eight single-end analog input channels or 4 differential analog input channels
- Maximum ADC peripheral clock frequency is 16 MHz
- Up to 760 kSPS sample rate
- Four operation modes:
  - Single mode: A/D conversion is performed one time on a specified channel.
  - Burst mode: A/D converter samples and converts the specified single channel and sequentially stores the result in FIFO.
  - Single-cycle Scan mode: A/D conversion is performed only one cycle on all specified channels with the sequence from the smallest numbered channel to the largest numbered channel.
  - Continuous Scan mode: A/D converter continuously performs Single-cycle Scan mode until software stops A/D conversion.
- An A/D conversion can be started by:
  - Software Write 1 to ADST bit
  - External pin (STADC)
  - PWM trigger with optional start delay period
- Each conversion result is held in data register of each channel with valid and overrun indicators.
- Conversion result can be compared with specified value and user can select whether to generate an interrupt when conversion result matches the compare register setting.
- Channel 7 supports 3 input sources: external analog voltage, internal band-gap voltage and
- Internal temperature sensor output.

6.13.3 Block Diagram


*ADTDCR(M05xxDE Only)
APB Bus
A/D Control Register (ADCR)
A/D Channel Enable Register (ADCHER)
A/D Compare Register (ADCMPR)
A/D Trigger Delay Control Register (ADTDCR)
A/D Data Register 0 (ADDR0)
A/D Data Register 1 (ADDR1)
. . .
A/D Data Register 7 (ADDR7)
A/D Status Register (ADSR)
STADC
PWM_TRG
* PWM Trigger(M05xxDE Only)
Digital Control Logic & ADC Clock Generator
VALID & OVERRUN
ADC_INT
ADF
RSLT[11:0]
V_{REF} (AV_{DD})
* V_{REF}(M05xxDE Only)
* AV_{DD}(M05xxBN Only)
12-bit DAC
ADC channel select
ADC clock and ADC start signal
Successive Approximations Register
ADC conversion finish
AIN0
AIN1
⋮
AIN7
VBG
VTEMP
Reserved
00
01
10
11
PRESEL[1:0]
8 to 1 Analog MUX
Sample and Hold
+
-
Comparator
Analog Control Logic
Analog Macro

Note: * The source of channel 7 is selected by PRESEL[1:0].

Figure 6.13-1 AD Controller Block Diagram

6.13.4 Basic Configuration

The ADC pin functions are configured in P1_MFP register. It is recommended to disable the digital input path of the analog input pins to avoid the leakage current. User can disable the digital input path by configuring P1_OFFD register.

The ADC peripheral clock can be enabled in APBCLK[28]. The ADC peripheral clock source is selected by CLKSEL1[3:2]. The clock prescalar is determined by CLKDIV[23:16].

6.13.5 Functional Description

The A/D converter operates by successive approximation with 12-bit resolution. The ADC has four operation modes: Single, Burst, Single-cycle Scan mode and Continuous Scan mode. When user wants to change the operation mode or analog input channel, in order to prevent incorrect operation, software must clear ADST bit to 0 in advance.

6.13.5.1 ADC peripheral Clock Generator

The maximum sampling rate is up to 760 kSPS. The ADC has four clock sources selected by ADC_S (CLKSEL1[3:2]), the ADC peripheral clock frequency is divided by an 8-bit pre-scalar with the following formula:

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

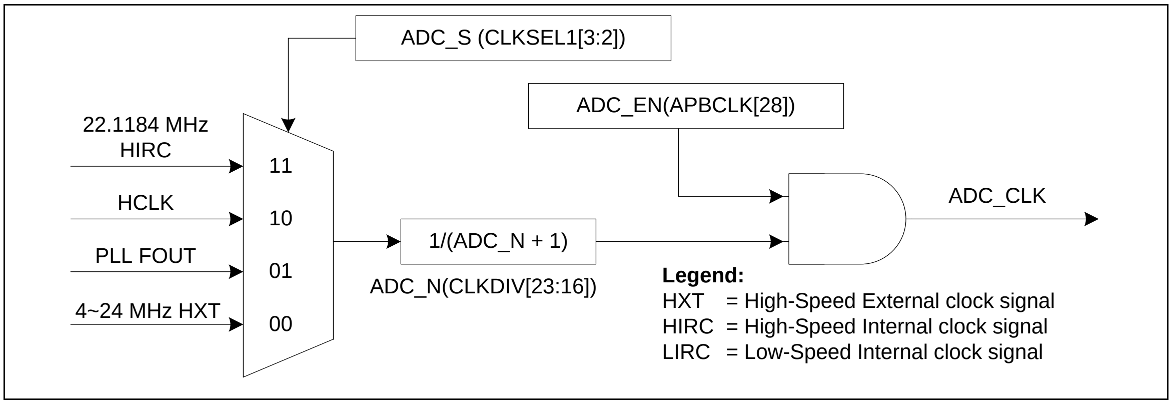


Figure 6.13-2 ADC Peripheral Clock Control

6.13.5.2 Single Mode

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

1. A/D conversion will be started when the ADST bit of 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. The ADF bit of ADSR register will be set to 1. If the ADIE bit of ADCR register is set to 1, the ADC interrupt will be asserted.

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

Note 1: If software enables more than one channel in Single mode, only the channel with the smallest number will be selected and the other enabled channels will be ignored.

Note 2: If ADST bit is cleared to 0 when ADC is in converting, the BUSY bit will be cleared to 0 immediately; But ADC cannot finish the current conversion and A/D converter enters idle state directly.

Through the ADC Driver, user can call the flow below to operate Single mode.

```
/* Set the ADC operation mode as Single, input mode as single-end and enable the ADC converter */
            ADC->ADCR = (ADC_ADCR_ADMD_SINGLE
                          | ADC_ADCR_DIFFEN_SINGLE_END
                          | ADC_ADCR_ADEN_CONVERTER_ENABLE);

/* Enable analog input channel 2 */
            _ADC_SET_CHANNEL(1<<2);

/* Clear the A/D interrupt flag for safe */
       ADC->ADSR = ADC_ADSR_ADF_Msk;

/* Enable the ADC interrupt */
       _ADC_ENABLE_ADC_INT();
       NVIC_EnableIRQ(ADC_IRQn);

/* Start A/D conversion */
       _ADC_START_CONVERT();
```

An example timing diagram for Single mode is shown below.

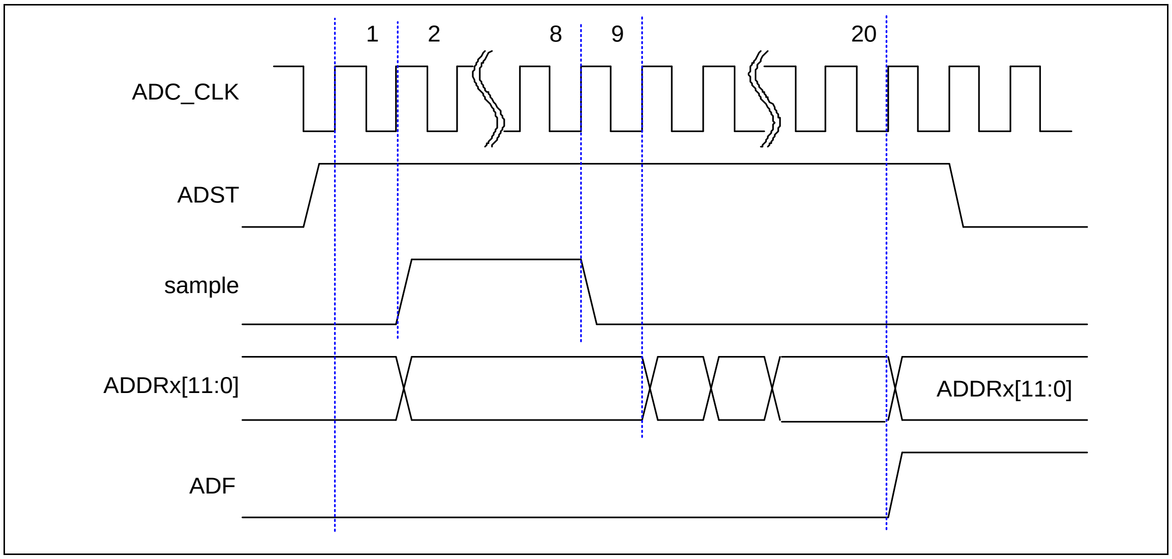


Figure 6.13-3 Single Mode Conversion Timing Diagram

6.13.5.3 Burst Mode

In Burst mode, A/D converter samples and converts the specified single channel and sequentially stores the result into FIFO (up to 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 is started on the channel with the smallest number.

2. When A/D conversion for the specified channel is completed, the result is sequentially transferred to FIFO and can be accessed only from the 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 ADC interrupt is requested after finishing the A/D conversion.

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, ADC cannot finish the current conversion and A/D converter enters idle state directly.

Note 1: If software enables more than one channel in Burst mode, only the channel with the smallest number is converted and other enabled channels will be ignored.

By Nuvoton ADC Driver, user can call the flow as shown below to operate Burst mode.

```
/* Set the ADC operation mode as Burst, input mode as single-end and enable the ADC converter */
        ADC->ADCR = (ADC_ADCR_ADMD_BURST
                                        | ADC_ADCR_DIFFEN_SINGLE_END
                                        | ADC_ADCR_ADEN_CONVERTER_ENABLE);
/* Enable analog input channel 2 */
        _ADC_SET_CHANNEL(1<<2);


/* Clear the A/D interrupt flag for safe */
```

```
        ADC->ADSR = ADC_ADSR_ADF_Msk;


/* Enable the ADC interrupt */
        _ADC_ENABLE_ADC_INT();
        NVIC_EnableIRQ(ADC_IRQn);


/* start A/D conversion */
        _ADC_START_CONVERT();
```

6.13.5.4 Single-Cycle Scan Mode

In Single-cycle Scan mode, A/D converter samples and converts all of the specified channels once in the sequence from the smallest number enabled channel to the largest number enabled channel. Operations are as follows:

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

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

3. When the conversions of all the enabled channels are completed, the ADF bit in ADSR is set to 1. If the ADC interrupt function is enabled, the ADC interrupt occurs.

4. After ADC finishes one cycle conversion, 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.

Through the ADC Driver, user can call the flow as shown below to operate Single-cycle Scan mode.

```
/* Set the ADC operation mode as Single-cycle, input mode as single-end and enable the ADC converter  */
        ADC->ADCR = (ADC_ADCR_ADMD_SINGLE_CYCLE
                              | ADC_ADCR_DIFFEN_SINGLE_END
                              | ADC_ADCR_ADEN_CONVERTER_ENABLE);
/* Enable analog input channel 0, 1, 2 and 3 */
        _ADC_SET_CHANNEL(0xF);


/* clear the A/D interrupt flag for safe */
        ADC->ADSR = ADC_ADSR_ADF_Msk;


/* start A/D conversion */
        _ADC_START_CONVERT();


/* Wait conversion done */
        _ADC_WAIT_CONVERSION_DONE();
```

An example timing diagram for Single-cycle Scan mode on enabled channels (0, 2, 3 and 7) is shown below.

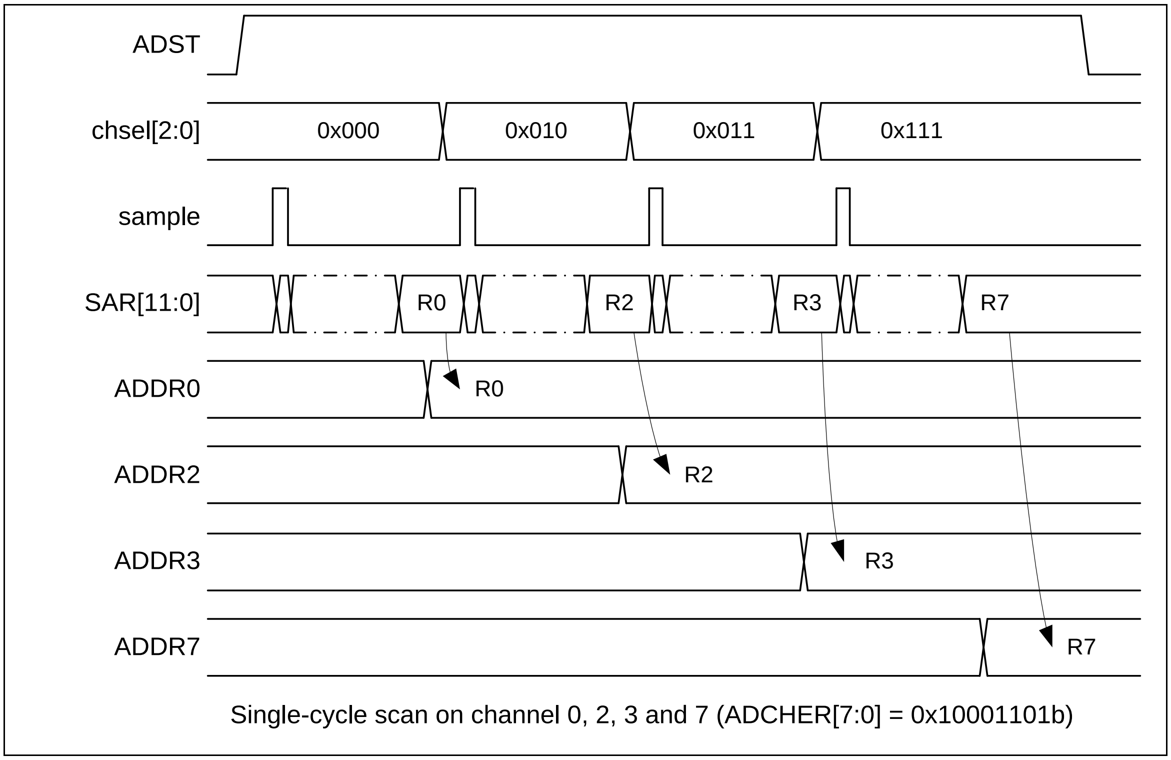


Figure 6.13-4 Single-Cycle Scan Mode on Enabled Channels Timing Diagram

6.13.5.5 Continuous Scan Mode

In Continuous Scan mode, A/D conversion is performed sequentially on the specified channels that enabled by CHEN bits in 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 is started on the channel with the smallest number.

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

3. When A/D converter completes the conversions of all enabled channels sequentially, the ADF bit (ADSR[0]) will be set to 1. If the ADC interrupt function is enabled, the ADC interrupt occurs. The conversion of the enabled channel with the smallest number will start again if software does not clear the ADST bit.

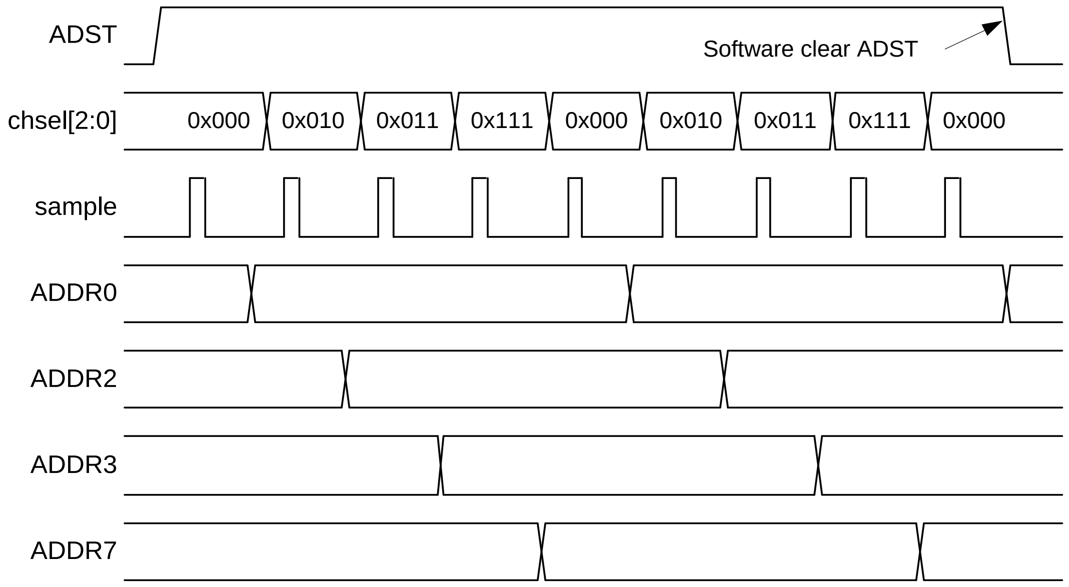
4. As long as the ADST bit remains at 1, the step 2 ~ 3 will be repeated. When ADST is cleared to 0, ADC cannot finish the current conversion and A/D converter enters idle state directly.

Through the ADC Driver, user can call the flow as shown below to operate Continuous Scan mode.

```
/* Set the ADC operation mode as Continuous Scan, input mode as single-end and enable the ADC converter */
        ADC->ADCR = (ADC_ADCR_ADMD_CONTINUOUS
                                     | ADC_ADCR_DIFFEN_SINGLE_END
```

```
                                   | ADC_ADCR_ADEN_CONVERTER_ENABLE);

/* Enable analog input channel 0, 1, 2 and 3 */
        _ADC_SET_CHANNEL(0xF);

/* Clear the A/D interrupt flag for safe */
        ADC->ADSR = ADC_ADSR_ADF_Msk;

/* Start A/D conversion */
        _ADC_START_CONVERT();

/* Wait conversion done */
        _ADC_WAIT_CONVERSION_DONE();

/* Stop A/D conversion */
        _ADC_STOP_CONVERT();
```

An example timing diagram for Continuous Scan mode on enabled channels (0, 2, 3 and 7) is shown below.



Continuous scan on channel 0, 2, 3 and 7 (ADCHER[7:0] = 0x10001101b)

Figure 6.13-5 Continuous Scan Mode on Enabled Channels Timing Diagram

6.13.5.6 External Trigger Input Sampling and A/D Conversion Time

In Single-cycle Scan mode, A/D conversion can be triggered by external pin request. When the TRGEN bit of ADCR register is set to 1 to enable ADC external trigger function, setting the

TRGS[1:0] bits to 00b is to select external trigger input from the STADC pin. Software can set TRGCOND[1:0] to select trigger condition is falling/rising edge or low/high level. If level trigger condition is selected, the STADC pin must be kept at specified state at least 8 PCLKs. The ADST bit will be set to 1 at the 9th PCLK and start to convert. Conversion will keep going if external trigger input is kept at active state in level trigger mode. It is stopped only when external condition trigger condition disappears. If edge trigger condition is selected, the high and low state must be kept at least 4 PLCKs. Pulse that is shorter than this specification will be ignored.

Note: User must enable the external trigger function or enable ADC at least 4 PCLKs after enabling ADC peripheral clock.

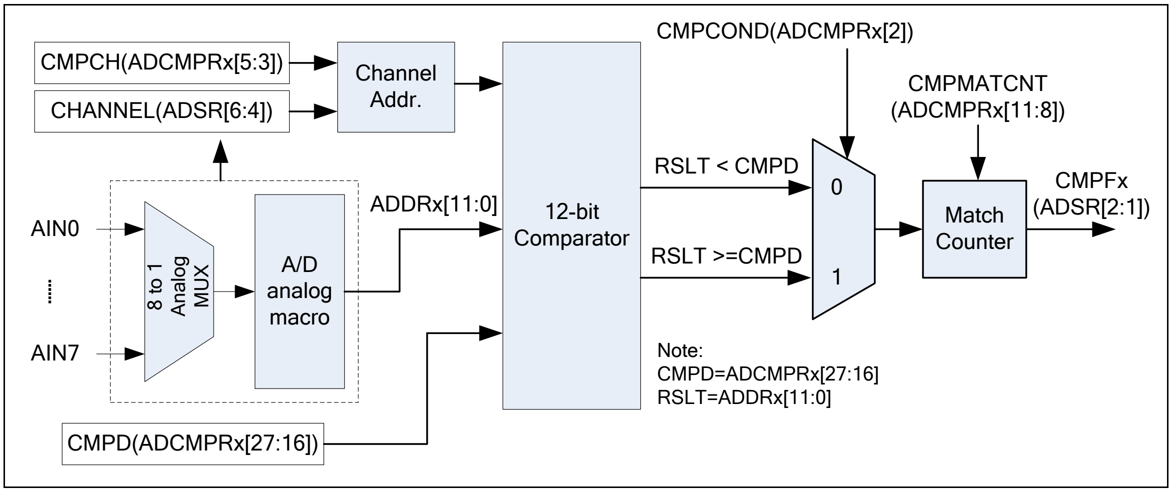
6.13.5.7 PWM trigger

In Single-cycle Scan mode, A/D conversion can be triggered by PWM request. When the TRGEN is set to high to enable ADC external hardware trigger function, setting the TRGS[1:0] bits to 11b is to select external hardware trigger input source from PWM trigger. When PWM trigger is enabled, setting PTDT[7:0] bits can insert a delay time between PWM trigger condition and ADC start conversion.

6.13.5.8 Conversion Result Monitor by Compare Mode Function

The NuMicro™ M058S series ADC controller provides two compare registers, ADCMPR0 and ADCMPR1, to monitor maximum two specified channels. Software can select which channel to be monitored by setting CMPCH (ADCMPRx[5:0]). CMPCOND bit is used to determine the compare condition. If CMPCOND bit is cleared to 0, the internal match counter will increase one when the conversion result is less than the value specified in CMPD[11:0]; if CMPCOND bit is set to 1, the internal match counter will increase one when the conversion result is greater than or equal to the value specified in CMPD[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 the match counter reaches the setting of (CMPMATCNT+1) then CMPF bit will be set to 1, if CMPIE bit is set then an ADC_INT interrupt request is generated. Software can use it to monitor the external analog input pin voltage transition in scan mode without imposing a load on software. The detailed logic diagram is shown below.

Note: If user enables differential input mode, the conversion result can be expressed with binary straight format (unsigned format) or 2's complement format (signed format). Software can set DMOF[31] bit of ADSR register to select output format.



Note:
CMPD=ADCMPRx[27:16]
RSLT=ADDRx[11:0]

Figure 6.13-6 A/D Conversion Result Monitor Logic Diagram

6.13.5.9 Interrupt Sources

There are three interrupt sources of ADC interrupt. When an ADC operation mode finishes its conversion, the A/D conversion end flag, ADF, will be set to 1. The CMPF0 and CMPF1 are the compare flags of compare function. When the conversion result meets the settings of ADCMPR0/1, the corresponding flag will be set to 1. When one of the flags, ADF, CMPF0 and CMPF1, is set to 1 and the corresponding interrupt enable bit, ADIE of ADCR and CMPIE of ADCMPR0/1, is set to 1, the ADC interrupt will be asserted. Software can clear these flags to revoke the interrupt request.

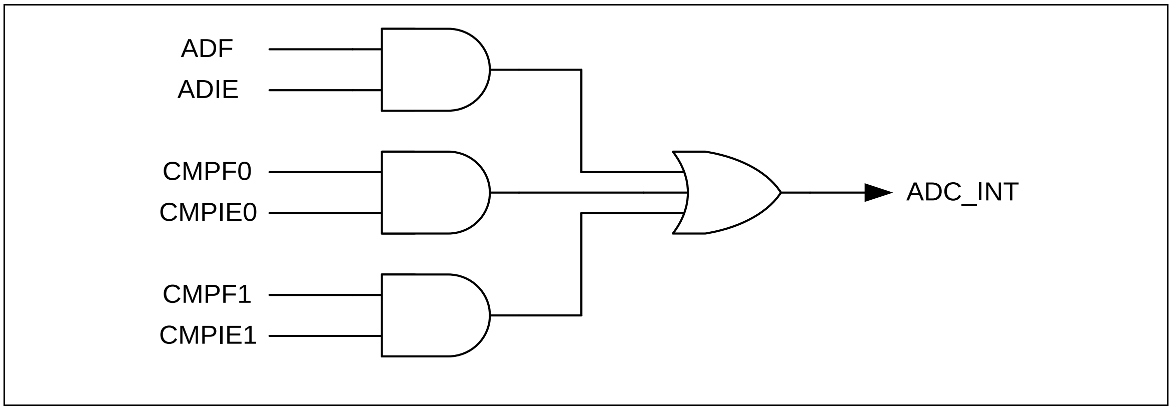


Figure 6.13-7 A/D Controller Interrupt

6.13.6 Register Map

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

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

ADTDCR	ADC_BA+0x44	R/W	ADC Trigger Delay Control Register	0x0000_0000

6.13.7 Register Description

ADC Data Registers (ADDR0 ~ ADDR7)

Register	Offset	R/W	Description	Reset Value
ADDR0	ADC_BA+0x00	R	ADC Data Register 0	0x0000_0000
ADDR1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000
ADDR2	ADC_BA+0x08	R	ADC Data Register 2	0x0000_0000
ADDR3	ADC_BA+0x0C	R	ADC Data Register 3	0x0000_0000
ADDR4	ADC_BA+0x10	R	ADC Data Register 4	0x0000_0000
ADDR5	ADC_BA+0x14	R	ADC Data Register 5	0x0000_0000
ADDR6	ADC_BA+0x18	R	ADC Data Register 6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	ADC Data Register 7	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**Valid Flag** 0 = Data in RSLT bits is not valid. 1 = Data in RSLT bits is valid. This bit will be set to 1 when the conversion of the corresponding channel is completed. This bit will be cleared to 0 by hardware after ADDR register is read. This is a read only bit. **Note:** When ADC is converting, if user wants to monitor the VALID flag of a specified channel, user should poll the VALID bit of ADSR register instead of polling this bit.
[16]	**OVERRUN**	**Overrun Flag (Read Only)** 0 = Data in RSLT is not overwrote. 1 = Data in RSLT is overwrote. If converted data in RSLT 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.

[15:0]	**RSLT**	**A/D Conversion Result** This field contains conversion result of ADC.

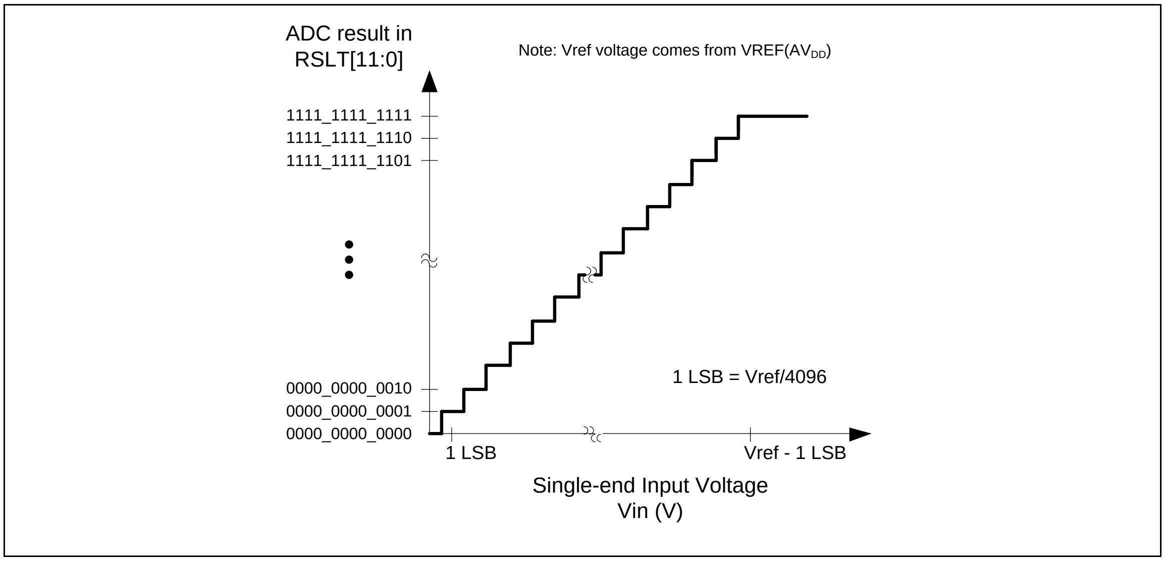


Figure 6.13-8 Conversion Result Mapping Diagram of ADC Single-end Input

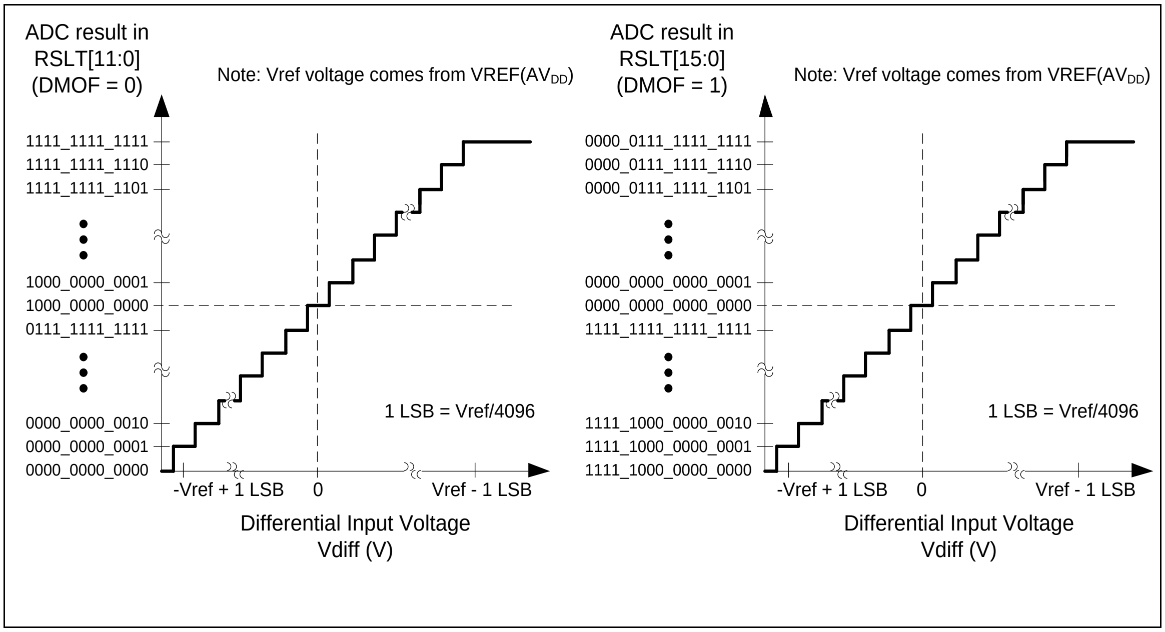


Figure 6.13-9 Conversion Result Mapping Diagram of ADC Differential Input

ADC Control Register (ADCR)

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

31	30	29	28	27	26	25	24
DMOF	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ADST**	**DIFFEN**	**Reserved**	**TRGEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRGCOND		**TRGS**		**ADMD**		**ADIE**	**ADEN**

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31]</td><td><b>DMOF</b></td><td><b>Differential Input Mode Output Format</b>
0 = A/D Conversion result will be filled in RSLT at ADDRx registers with unsigned format (straight binary format).
1 = A/D Conversion result will be filled in RSLT at ADDRx registers with 2's complement format.</td></tr>
<tr><td>[30:12]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[11]</td><td><b>ADST</b></td><td><b>A/D Conversion Start</b>
ADST bit can be set to 1 from two sources: software or external pin STADC. ADST will be cleared to 0 by hardware automatically at the ends of Single mode and Single-cycle Scan mode. In Continuous Scan mode and Burst mode, A/D conversion is continuously performed until software writes 0 to this bit or chip is reset.
0 = Conversion stops and A/D converter enters idle state.
1 = Conversion starts.</td></tr>
<tr><td>[10]</td><td><b>DIFFEN</b></td><td><b>Differential Input Mode Control</b>
0 = Single-end analog input mode.
1 = Differential analog input mode.
<table>
<tr><th>Differential Input Paired Channel</th><th colspan="2">ADC Analog Input</th></tr>
<tr><td></td><td>V_{plus}</td><td>V_{minus}</td></tr>
<tr><td>0</td><td>AIN0</td><td>AIN1</td></tr>
<tr><td>1</td><td>AIN2</td><td>AIN3</td></tr>
<tr><td>2</td><td>AIN4</td><td>AIN5</td></tr>
<tr><td>3</td><td>AIN6</td><td>AIN7</td></tr>
</table>
<b>Differential input voltage (V_{diff}) = V_{plus} - V_{minus}</b>, where V_{plus} is the analog input;</td></tr>
</table>

		V_{minus} is the inverted analog input. **Note:** In Differential Input mode, only the even number of the two corresponding channels needs to be enabled in ADCHER register. The conversion result will be placed to the corresponding data register of the enabled channel.
[9]	**Reserved**	Reserved.
[8]	**TRGEN**	**External Trigger Enable Control** Enable or disable triggering of A/D conversion by external STADC pin. If external trigger is enabled, the ADST bit can be set to 1 by the selected hardware trigger source. 0= External trigger Disabled. 1= External trigger Enabled. **Note:** The ADC external trigger function is only supported in Single-cycle Scan mode.
[7:6]	**TRGCOND**	**External Trigger Condition** These two bits decide external pin STADC trigger event is level or edge. The signal must be kept at stable state at least 8 PCLKs for level trigger and at least 4 PCLKs for edge trigger. 00 = Low level. 01 = High level. 10 = Falling edge. 11 = Rising edge.
[5:4]	**TRGS**	**Hardware Trigger Source** 00 = A/D conversion is started by external STADC pin. 11 = A/D conversion is started by PWM trigger. Software should clear TRGEN bit and ADST bit to 0 before changing TRGS.
[3:2]	**ADMD**	**A/D Converter Operation Mode Control** 00 = Single conversion. 01 = Burst conversion. 10 = Single-cycle Scan. 11 = Continuous Scan. **Note1:** When changing the operation mode, software should clear ADST bit firstly. **Note2:** In Burst mode, the A/D result data always at Data Register 0.
[1]	**ADIE**	**A/D Interrupt Enable Control** A/D conversion end interrupt request is generated if ADIE bit is set to 1. 0 = A/D interrupt function Disabled. 1 = A/D interrupt function Enabled.
[0]	**ADEN**	**A/D Converter Enable** 0 = A/D converter Disabled. 1 = A/D converter Enabled. **Note:** Before starting A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit to save power consumption.

ADC Channel Enable Register (ADCHER)

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

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**PRESEL[1:0]**	**Analog Input Channel 7 Source Selection** 00 = External analog input. 01 = Internal band-gap voltage. 10 = Internal temperature sensor. 11 = Reserved. **Note:** When the band-gap voltage is selected as the analog input source of ADC channel 7, ADC peripheral clock rate needs to be limited to lower than 300 kHz.
[7:0]	**CHEN**	**Analog Input Channel Enable Control** Set CHEN[7:0] to enable the corresponding analog input channel 7 ~ 0. If DIFFEN bit is set to 1, only the even number channel needs to be enabled. 0 = Channel Disabled. 1 = Channel Enabled.

NUMICRO™ M058S SERIES TECHNICAL REFERENCE MANUAL

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

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

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**CMPD**	**Comparison Data** The 12-bit data is used to compare with conversion result of specified channel. **Note:** CMPD should be filled in unsigned format (straight binary format).
[15:12]	**Reserved**	Reserved.
[11:8]	**CMPMATCNT**	**Compare Match Count** When the specified A/D channel analog conversion result matches the compare condition defined by CMPCOND[2], the internal match counter will increase 1. When the internal counter reaches the value to (CMPMATCNT +1), the CMPFx bit will be set.
[7:6]	**Reserved**	Reserved.
[5:3]	**CMPCH**	**Compare Channel Selection** 000 = Channel 0 conversion result is selected to be compared. 001 = Channel 1 conversion result is selected to be compared. 010 = Channel 2 conversion result is selected to be compared. 011 = Channel 3 conversion result is selected to be compared. 100 = Channel 4 conversion result is selected to be compared. 101 = Channel 5 conversion result is selected to be compared. 110 = Channel 6 conversion result is selected to be compared. 111 = Channel 7 conversion result is selected to be compared.
[2]	**CMPCOND**	**Compare Condition** 0= Set the compare condition as that when a 12-bit A/D conversion result is less than the 12-bit CMPD (ADCMPRx[27:16]), the internal match counter will increase one. 1= Set the compare condition as that when a 12-bit A/D conversion result is

		greater than or equal to the 12-bit CMPD (ADCMPRx[27:16]), the internal match counter will increase one. **Note:** When the internal counter reaches to (CMPMATCNT +1), the CMPFx bit will be set.
[1]	**CMPIE**	**Compare Interrupt Enable Control** If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMATCNT, CMPFx bit will be asserted, in the meanwhile, if CMPIE is set to 1, a compare interrupt request is generated. 0 = Compare function interrupt Disabled. 1 = Compare function interrupt Enabled.
[0]	**CMPEN**	**Compare Enable Control** Set this bit to 1 to enable ADC controller to compare CMPD[11:0] with specified channel conversion result when converted data is loaded into ADDR register. 0 = Compare function Disabled. 1 = Compare function Enabled.

A/D Status Register (ADSR)

Register	Offset	R/W	Description	Reset Value
ADSR	ADC_BA+0x30	R/W	ADC Status Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**OVERRUN**	**Overrun Flag (Read Only)** It is a mirror of OVERRUN bit in ADDRx register. When ADC is in Burst mode and the FIFO is overrun, all bits of OVERRUN[7:0] will be set to 1.
[15:8]	**VALID**	**Data Valid Flag (Read Only)** It is a mirror of VALID bit in ADDRx register. When ADC is in Burst mode and any conversion result is valid, all bits of VALID[7:0] will be set to 1.
[7]	**Reserved**	Reserved.
[6:4]	**CHANNEL**	**Current Conversion Channel** When BUSY=1, this filed reflects current conversion channel. When BUSY=0, it shows the number of the next converted channel. It is read only.
[3]	**BUSY**	**BUSY/IDLE** This bit is a mirror of ADST bit in ADCR register. It is read only. 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion.
[2]	**CMPF1**	**Compare Flag 1** When the A/D conversion result of the selected channel meets setting condition in ADCMPR1 then this bit is set to 1; it is cleared by writing 1 to self. 0 = Conversion result in ADDR does not meet ADCMPR1 setting. 1 = Conversion result in ADDR meets ADCMPR1 setting.
[1]	**CMPF0**	**Compare Flag 0** When the A/D conversion result of the selected channel meets setting condition

		in ADCMPR0 then this bit is set to 1. This bit is cleared by writing 1 to it. 0 = Conversion result in ADDR does not meet ADCMPR0 setting. 1 = Conversion result in ADDR meets ADCMPR0 setting.
[0]	**ADF**	**A/D Conversion End Flag** A status flag that indicates the end of A/D conversion. Software can write 1 to clear this bit. ADF is set to 1 at the following three conditions: 1. When A/D conversion ends in Single mode. 2. When A/D conversion ends on all specified channels in Single-cycle Scan mode and Continuous Scan mode. 3. When more than 4 samples in FIFO in Burst mode.

ADC Trigger Delay Control Register (ADTDCR)

Register	Offset	R/W	Description	Reset Value
ADTDCR	ADC_BA+0x44	R/W	ADC Trigger Delay Control Register	0x0000_0000

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

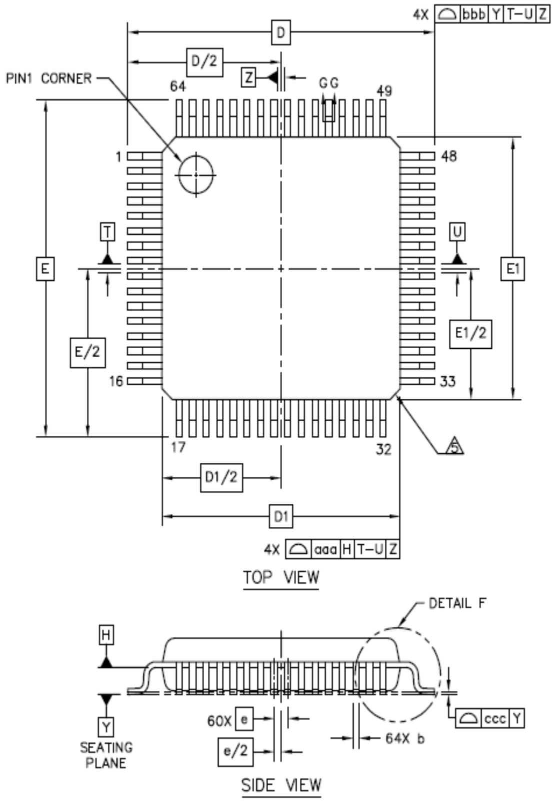
Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**PTDT**	**PWM Trigger Delay Time** Set this field will delay ADC start conversion time after PWM trigger. PWM trigger delay time is (4 * PTDT) * system clock

7 ELECTRICAL CHARACTERISTICS

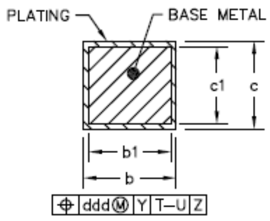
Please refer to the relative Datasheet for detailed information about the M058S Series electrical characteristics.

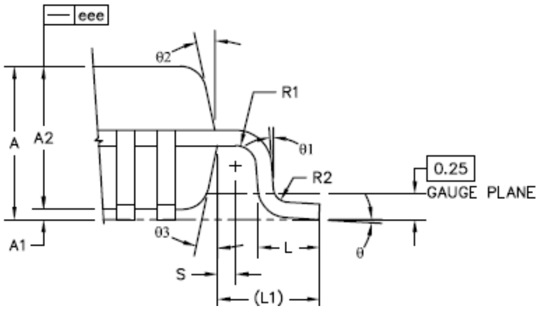
8 PACKAGE DIMENSIONS

8.1 LQFP-64 (7x7mm, Thickness 1.4 mm, Footprint 2.0mm)

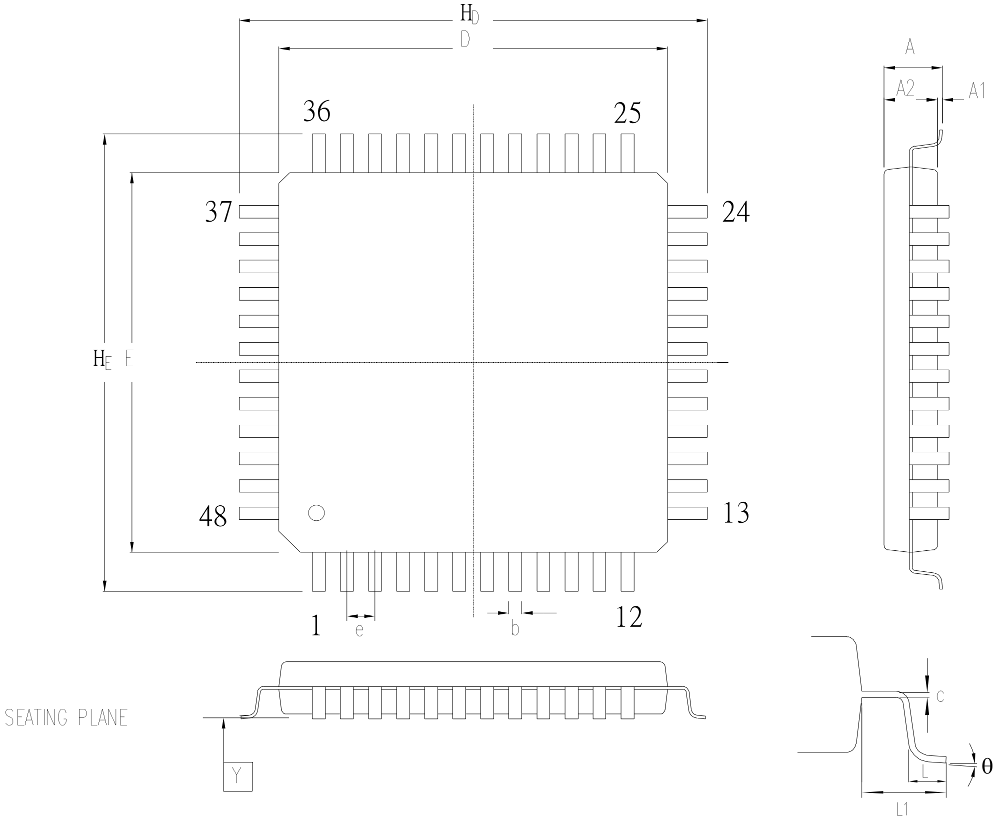


		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	---	---	1.6
STAND OFF		A1	0.05	---	0.15
MOLD THICKNESS		A2	1.35	1.4	1.45
LEAD WIDTH(PLATING)		b	0.13	0.18	0.23
LEAD WIDTH		b1	0.13	0.16	0.19
L/F THICKNESS(PLATING)		c	0.09	---	0.2
L/F THICKNESS		c1	0.09	---	0.16
	X	D	9 BSC		
	Y	E	9 BSC		
BODY SIZE	X	D1	7 BSC		
	Y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





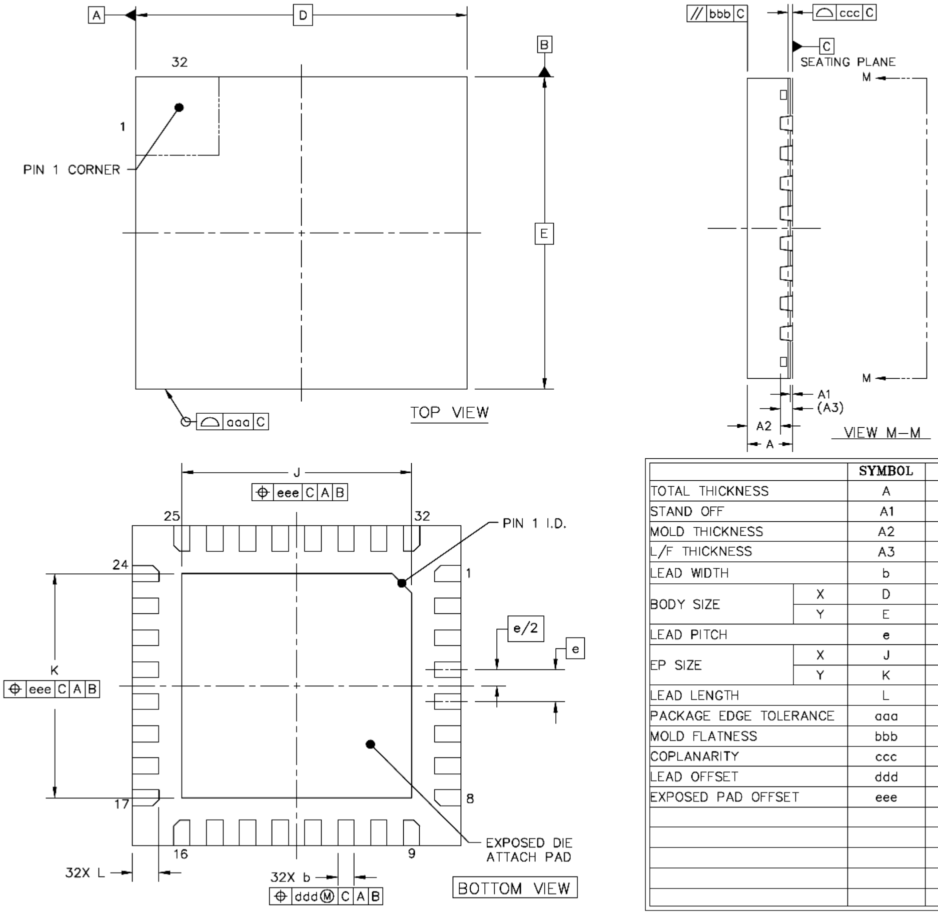
8.2 LQFP-48 (7x7x1.4mm^2 Footprint 2.0mm)



Controlling dimension : Millimeters

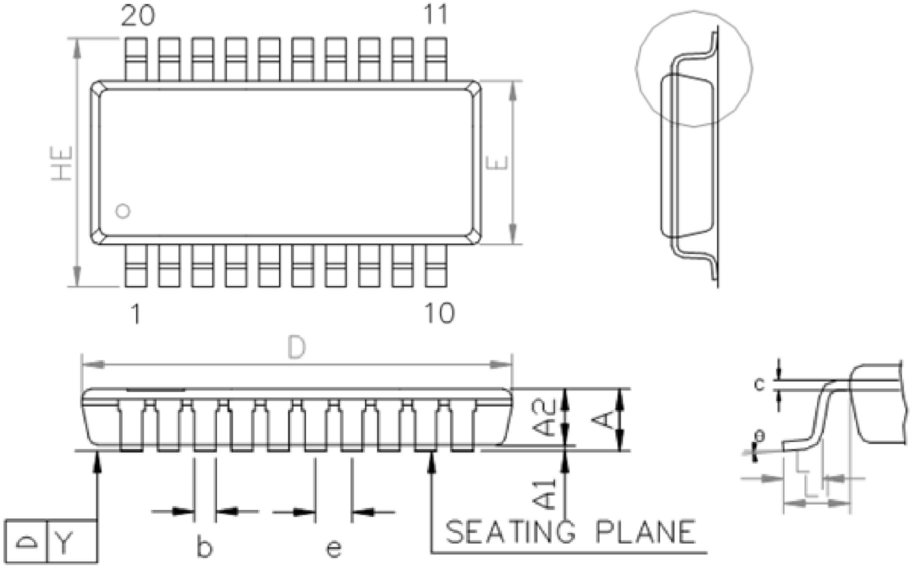
Symbol	Dimension in inch			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	—	—	—	—	—	—
A_1	0.002	0.004	0.006	0.05	0.10	0.15
A_2	0.053	0.055	0.057	1.35	1.40	1.45
b	0.006	0.008	0.010	0.15	0.20	0.25
c	0.004	0.006	0.008	0.10	0.15	0.20
D	0.272	0.276	0.280	6.90	7.00	7.10
E	0.272	0.276	0.280	6.90	7.00	7.10
e	0.014	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

8.3 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		

8.4 TSSOP-20 (4.4x6.5 mm)



SYMBOL	DIMENSION (MM)			DIMENSION (INCH)		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	–	–	1.20	–	–	0.047
A1	0.05	–	0.15	0.002	–	0.006
A2	0.80	0.90	1.05	0.031	0.035	0.041
E	4.30	4.40	4.50	0.169	0.173	0.177
HE	6.40 BSC			0.252 BSC		
D	6.40	6.50	6.60	0.252	0.256	0.260
L	0.50	0.60	0.75	0.020	0.024	0.030
L1	1.00 REF			0.039 REF		
b	0.19	–	0.30	0.007	–	0.012
e	0.65 BSC			0.026 BSC		
c	0.09	–	0.20	0.004	–	0.008
θ	0°	–	8°	0°	–	8°
Y	0.10 BASIC			0.004 BASIC		

9 REVISION HISTORY

Revision	Date	Description
1.00	Apr. 24, 2014	First version
1.01	Sep. 12, 2014	1. Adjusted the format of Table 4.1-1 NuMicro™ M058S Series Selection Guide. 2. Updated Figure 4.1-1 NuMicro™ M058S Series Selection Code 3. Changed the order of Chapter 5 BLOCK DIAGRAMand Chapter 6 FUNCTIONAL DESCRIPTION. 4. Fixed typos and obscure descriptions.
1.02	Nov. 27, 2014	1. Fixed typos of Table 4.1-1 NuMicro™ M058S Series Selection Guide.
1.03	Apr. 30, 2020	1. Add CONFIG0[27] “CFGXT1” and description at Section 6.4.6. 2. Added notes about the hardware reference design for ICE_DAT, ICE_CLK and /RST pins in section 4.3.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.