ARM® Cortex®-M0

32-bit Microcontroller

NuMicro® Family
NUC123 Series
Technical Reference Manual

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

www.nuvoton.com

TABLE OF CONTENTS

List of Figures

List of Tables

1 GENERAL DESCRIPTION

The NuMicro® NUC123 series is a new 32-bit Cortex®-M0 microcontroller with USB 2.0 Full-speed devices and a 10-bit ADC. The NUC123 series provides the high 72 MHz operating speed, large 20 Kbytes SRAM, 8 USB endpoints and three sets of SPI controllers, which make it powerful in USB communication and data processing. The NUC123 series is ideal for industrial control, consumer electronics, and communication system applications such as printers, touch panel, gaming keyboard, gaming joystick, USB audio, PC peripherals, and alarm systems.

The NUC123 series runs up to 72 MHz and supports 32-bit multiplier, structure NVIC (Nested Vector Interrupt Control), dual-channel APB and PDMA (Peripheral Direct Memory Access) with CRC function. Besides, the NUC123 series is equipped with 36/68 Kbytes Flash memory, 12/20 Kbytes SRAM, and 4 Kbytes loader ROM for the ISP. It operates at a wide voltage range of 2.5V ~ 5.5V and temperature range of -40˚C ~ +105˚C and -40˚C ~ +85˚C. It is also equipped with plenty of peripheral devices, such as 8-channel 10-bit ADC, UART, SPI, I^2C, I^2S, USB 2.0 FS devices, and offers low-voltage reset and Brown-out detection, PWM (Pulse-width Modulation), capture and compare features, four sets of 32-bit timers, Watchdog Timer, and internal RC oscillator. All these peripherals have been incorporated into the NUC123 series to reduce component count, board space and system cost.

Additionally, the NUC123 series is equipped with ISP (In-System Programming), IAP (In-Application-Programming) and ICP (In-Circuit Programming) functions, which allows the user to update the program under software control through the on-chip connectivity interface, such as SWD, UART and USB.

Product Line	UART	SPI	I^2C	USB	PS/2	I^2S	PWM	ADC
NUC123	2	3	2	1	1	1	4	8

Table 1-1 Key Features Support Table

2 FEATURES

2.1 NuMicro® NUC123 Series Features

- Core
  - ARM® Cortex®-M0 core runs up to 72 MHz
  - One 24-bit system timer
  - Supports low power sleep mode
  - Single-cycle 32-bit hardware multiplier
  - NVIC for the 32 interrupt inputs, each with 4-levels of priority
  - Supports Serial Wire Debug with 2 watchpoints/4 breakpoints
- Built-in LDO for wide operating voltage ranges from 2.5 V to 5.5 V
- Flash Memory
  - 36/68 KB Flash for program code
  - 4 KB flash for ISP loader
  - Supports In-System Program (ISP) application code update
  - 512 byte page erase for flash
  - Configurable Data Flash address and size for both 36KB and 68KB system
  - Supports 2-wire ICP update through SWD/ICE interface
  - Supports fast parallel programming mode by external programmer
- SRAM Memory
  - 12/20 KB embedded SRAM
  - Supports PDMA mode
- PDMA (Peripheral DMA)
  - Supports 6 channels PDMA for automatic data transfer between SRAM and peripherals such as SPI, UART, I^2S, USB 2.0 FS device, PWM and ADC
  - Supports CRC calculation with four common polynomials, CRC-CCITT, CRC-8, CRC-16 and CRC-32
- Clock Control
  - Flexible selection for different applications
  - Built-in 22.1184 MHz high speed oscillator (Trimmed to 1%) for system operation, and low power 10 kHz low speed oscillator for watchdog and wake-up operation
  - Supports one PLL, up to 144 MHz, for high performance system operation
  - External 4~24 MHz high speed crystal input for precise timing operation
- GPIO
  - Four I/O modes:
    - Quasi bi-direction
    - Push-Pull output
    - Open-Drain output
    - Input only with high impendence
  - TTL/Schmitt trigger input selectable
  - I/O pin configured as interrupt source with edge/level setting
  - Supports High Driver and High Sink I/O mode
- Timer
  - Supports 4 sets of 32-bit timers with 24-bit up-timer and one 8-bit pre-scale counter
  - Independent clock source for each timer
  - Provides one-shot, periodic, toggle and continuous counting operation modes
  - Supports event counting function
- Watchdog/Windowed-Watchdog Timer

- Multiple clock sources
- 8 selectable time-out period from 1.6ms ~ 26.0sec (depending on clock source)
- Wake-up from Power-down or Idle mode
- Interrupt or reset selectable on watchdog timer time-out
- Interrupt on windowed-watchdog timer time-out
- Reset on windowed-watchdog timer time-out or reload in an unexpected time window

- PWM/Capture
  - Up to two built-in 16-bit PWM generators provided with four PWM outputs or two complementary paired PWM outputs
  - Each PWM generator equipped with one clock source selector, one clock divider, one 8-bit prescaler and one Dead-zone generator for complementary paired PWM
  - Up to four 16-bit digital Capture timers (shared with PWM timers) provided with four rising/falling capture inputs
  - Supports Capture interrupt
- UART
  - Up to two UART controllers
  - UART ports with flow control (TXD, RXD, CTS and RTS)
  - UART0/1 with 16-byte FIFO for standard device
  - Support IrDA (SIR) function
  - Supports RS-485 9-bit mode and direction control.
  - Programmable baud-rate generator up to 1/16 system clock
  - Supports PDMA mode
- SPI
  - Up to three sets of SPI controllers
  - Supports SPI master/Slave mode
  - Full duplex synchronous serial data transfer
  - Variable length of transfer data from 8 to 32 bits
  - MSB or LSB first data transfer
  - Up to two slave/device select lines in Master mode
  - Supports Byte Suspend mode in 16/24/32-bit transmission
  - Supports PDMA transfer
- I^2C
  - Up to two sets of I^2C devices
  - Master/Slave mode
  - Bidirectional data transfer between masters and slaves
  - Multi-master bus (no central master)
  - Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
  - Serial clock synchronization allows devices with different bit rates to communicate via one serial bus
  - Serial clock synchronization used as a handshake mechanism to suspend and resume serial transfer
  - Programmable clocks allowing versatile rate control
  - Supports multiple address recognition (four slave address with mask option)
  - Supports wake-up by address recognition (for 1st slave address only)
- I^2S
  - Interface with external audio CODEC
  - Operated as either master or Slave mode
  - Capable of handling 8-, 16-, 24- and 32-bit word sizes
  - Supports Mono and stereo audio data
  - Supports I^2S and MSB justified data format
  - Two 8 word FIFO data buffers are provided, one for transmitting and the other for

receiving
  - Generates interrupt requests when buffer levels cross a programmable boundary
  - Supports two DMA requests, one for transmitting and the other for receiving
- PS/2 Device Controller
  - Host communication inhibit and request to send detection
  - Reception frame error detection
  - Programmable 1 to 16 bytes transmit buffer to reduce CPU intervention
  - Double buffer for data reception
  - S/W override bus
- USB 2.0 Full-Speed Device
  - One set of USB 2.0 FS Device 12 Mbps
  - On-chip USB transceiver
  - Provides 1 interrupt source with 4 interrupt events
  - Supports Control, Bulk In/Out, Interrupt and Isochronous transfers
  - Auto suspend function when no bus signaling for 3 ms
  - Provides 8 programmable endpoints
  - Includes 512 bytes internal SRAM as USB buffer
  - Provides remote wake-up capability
- ADC
  - 10-bit SAR ADC with 150K SPS (for NUC123xxxANx)
  - 10-bit SAR ADC with 200K SPS (for NUC123xxxAEx)
  - Up to 8-ch single-end input
  - Single scan/single cycle scan/continuous scan
  - Each channel with individual result register
  - Scan on enabled channels
  - Threshold voltage detection
  - Conversion start by software programming or external input
  - Supports PDMA mode
- Brown-out detector
  - With 4 levels: 4.4 V/3.7 V/2.7 V/2.2 V
  - Supports Brown-out Interrupt and Reset option
- Low Voltage Reset
  - Threshold voltage levels: 2.0 V
- One built-in LDO
- Operating Temperature: -40°C~85°C (for NUC123xxxANx)
- Operating Temperature: -40°C ~105°C (for NUC123xxxAEx)
- Packages:
  - All Green package (RoHS)
  - LQFP 64-pin
  - LQFP 48-pin
  - QFN 33-pin

3 ABBREVIATIONS

Acronym	Description
ACMP	Analog Comparator Controller
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
BOD	Brown-out Detection
CAN	Controller Area Network
DAP	Debug Access Port
DES	Data Encryption Standard
EBI	External Bus Interface
EPWM	Enhanced Pulse Width Modulation
FIFO	First In, First Out
FMC	Flash Memory Controller
FPU	Floating-point Unit
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
HIRC	22.1184 MHz Internal High Speed RC Oscillator
HXT	4~20 MHz External High Speed Crystal Oscillator
IAP	In Application Programming
ICP	In Circuit Programming
ISP	In System Programming
LDO	Low Dropout Regulator
LIN	Local Interconnect Network
LIRC	10 kHz internal low speed RC oscillator (LIRC)
MPU	Memory Protection Unit
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PDMA	Peripheral Direct Memory Access
PLL	Phase-Locked Loop
PWM	Pulse Width Modulation
QEI	Quadrature Encoder Interface
SD	Secure Digital
SPI	Serial Peripheral Interface

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

Table 3-1 List of Abbreviations

4 PARTS INFORMATION LIST AND PIN CONFIGURATION

4.1 NuMicro® NUC123 Series Naming Rule

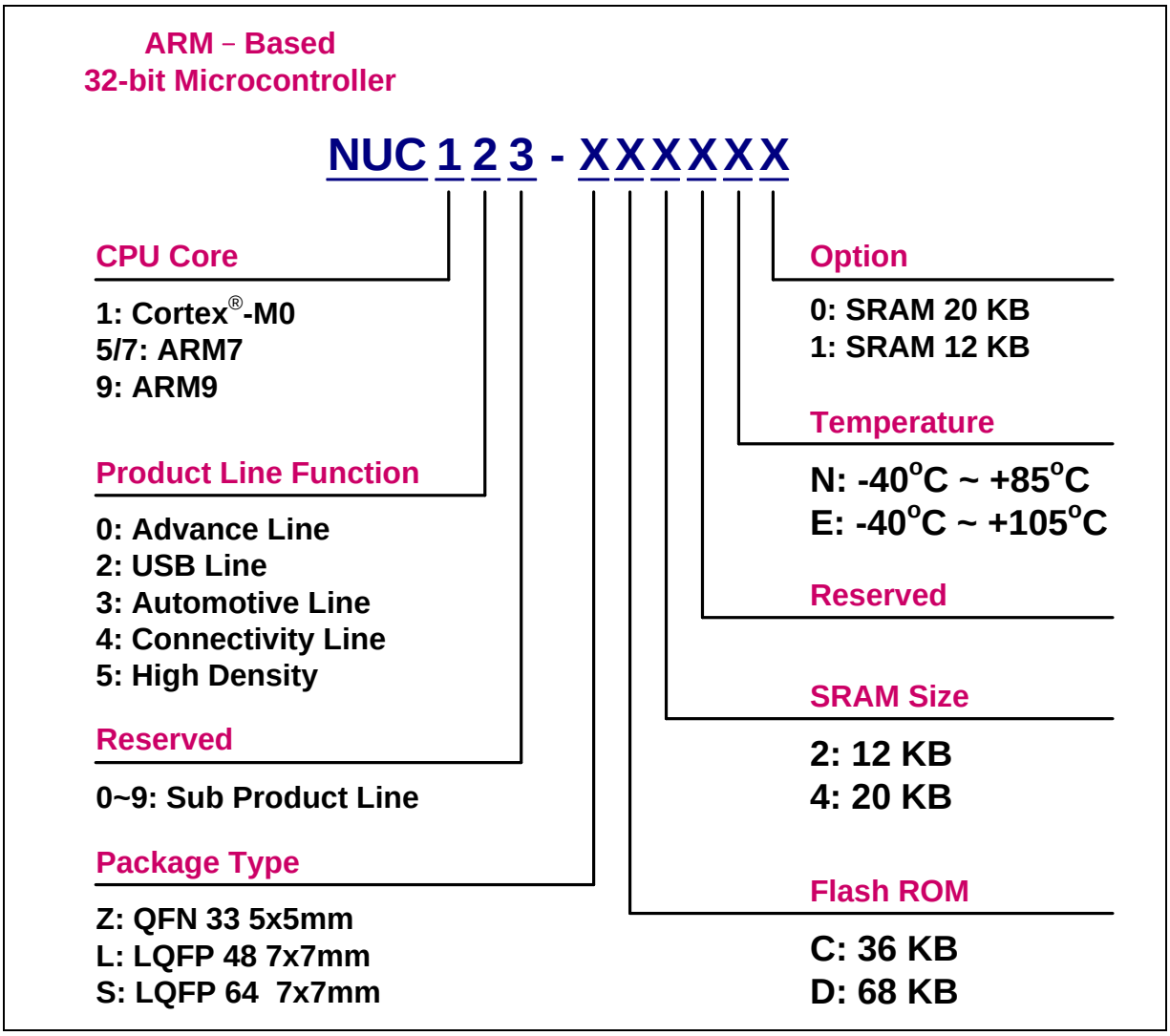


Figure 4-1 NuMicro® NUC123 Series Selection Code

4.2 NuMicro® NUC123 Series Selection Guide

4.2.1 NuMicro® NUC123xxxANx Selection Guide

Part Number	Flash (KB)	SRAM (KB)	ISP ROM (KB)	I/O	Timer	Connectivity						I²S	Comp.	PWM	ADC	RTC	EBI	ISP\ICP\IAP	1.8V Power Pin	Package
						UART	SPI	I²C	USB	LIN	PS/2									
NUC123ZD4AN0	68	20	4	Up to 20	4x32-bit	1	3	1	1	-	-	1	-	2	3x10-bit	-	-	v	-	QFN33
NUC123ZC2AN1	36	12	4	up to 20	4x32-bit	1	3	1	1	-	-	1	-	2	3x10-bit	-	-	v	-	QFN33
NUC123LD4AN0	68	20	4	up to 36	4x32-bit	2	3	2	1	-	1	1	-	4	8x10-bit	-	-	v	-	LQFP48
NUC123LC2AN1	36	12	4	up to 36	4x32-bit	2	3	2	1	-	1	1	-	4	8x10-bit	-	-	v	-	LQFP48
NUC123SD4AN0	68	20	4	up to 47	4x32-bit	2	3	2	1	-	1	1	-	4	8x10-bit	-	-	v	-	LQFP64
NUC123SC2AN1	36	12	4	up to 47	4x32-bit	2	3	2	1	-	1	1	-	4	8x10-bit	-	-	v	-	LQFP64

4.2.2 NuMicro® NUC123xxxAEx Selection Guide

Part Number	Flash (KB)	SRAM (KB)	ISP ROM (KB)	I/O	Timer	Connectivity						I²S	Comp.	PWM	ADC	RTC	EBI	ISP\ICP\IAP	1.8V Power Pin	Package
						UART	SPI	I²C	USB	LIN	PS/2									
NUC123ZD4AE0	68	20	4	Up to 20	4x32-bit	1	3	1	1	-	-	1	-	3	3x10-bit	-	-	v	-	QFN33
NUC123ZC2AE1	36	12	4	up to 20	4x32-bit	1	3	1	1	-	-	1	-	3	3x10-bit	-	-	v	-	QFN33
NUC123LD4AE0	68	20	4	up to 36	4x32-bit	2	3	2	1	-	1	1	-	4	8x10-bit	-	-	v	-	LQFP48
NUC123LC2AE1	36	12	4	up to 36	4x32-bit	2	3	2	1	-	1	1	-	4	8x10-bit	-	-	v	-	LQFP48
NUC123SD4AE0	68	20	4	up to 47	4x32-bit	2	3	2	1	-	1	1	-	4	8x10-bit	-	-	v	-	LQFP64
NUC123SC2AE1	36	12	4	up to 47	4x32-bit	2	3	2	1	-	1	1	-	4	8x10-bit	-	-	v	-	LQFP64

4.3 NuMicro® NUC123 Series Pin Configuration

4.3.1 NuMicro® NUC123xxxANx Pin Diagram

4.3.1.1 *NuMicro® NUC123SxxANx LQFP 64 pin*

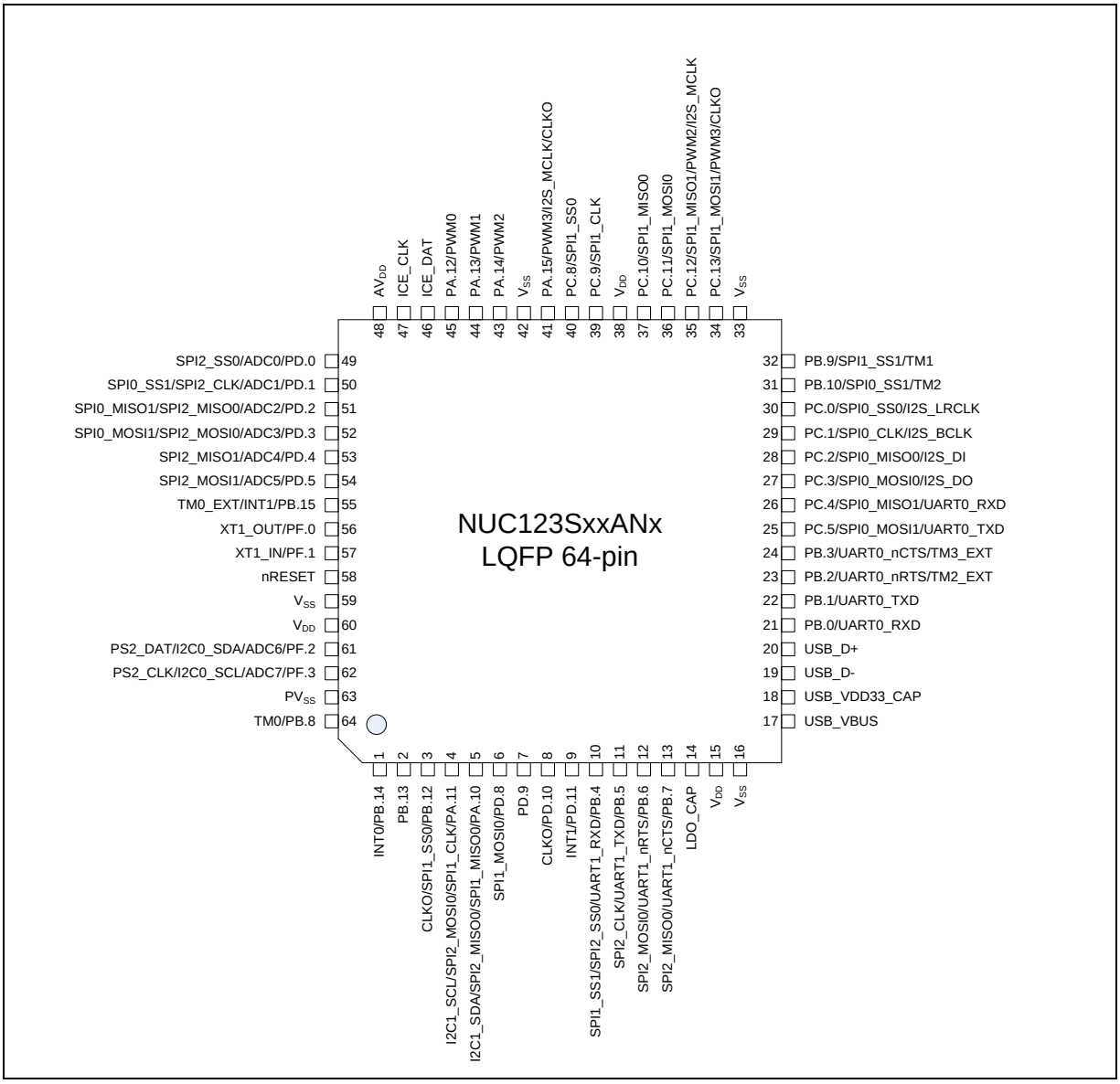


Figure 4-2 NuMicro® NUC123SxxANx LQFP 64-pin Diagram

4.3.1.2 *NuMicro® NUC123LxxANx LQFP 48 pin*

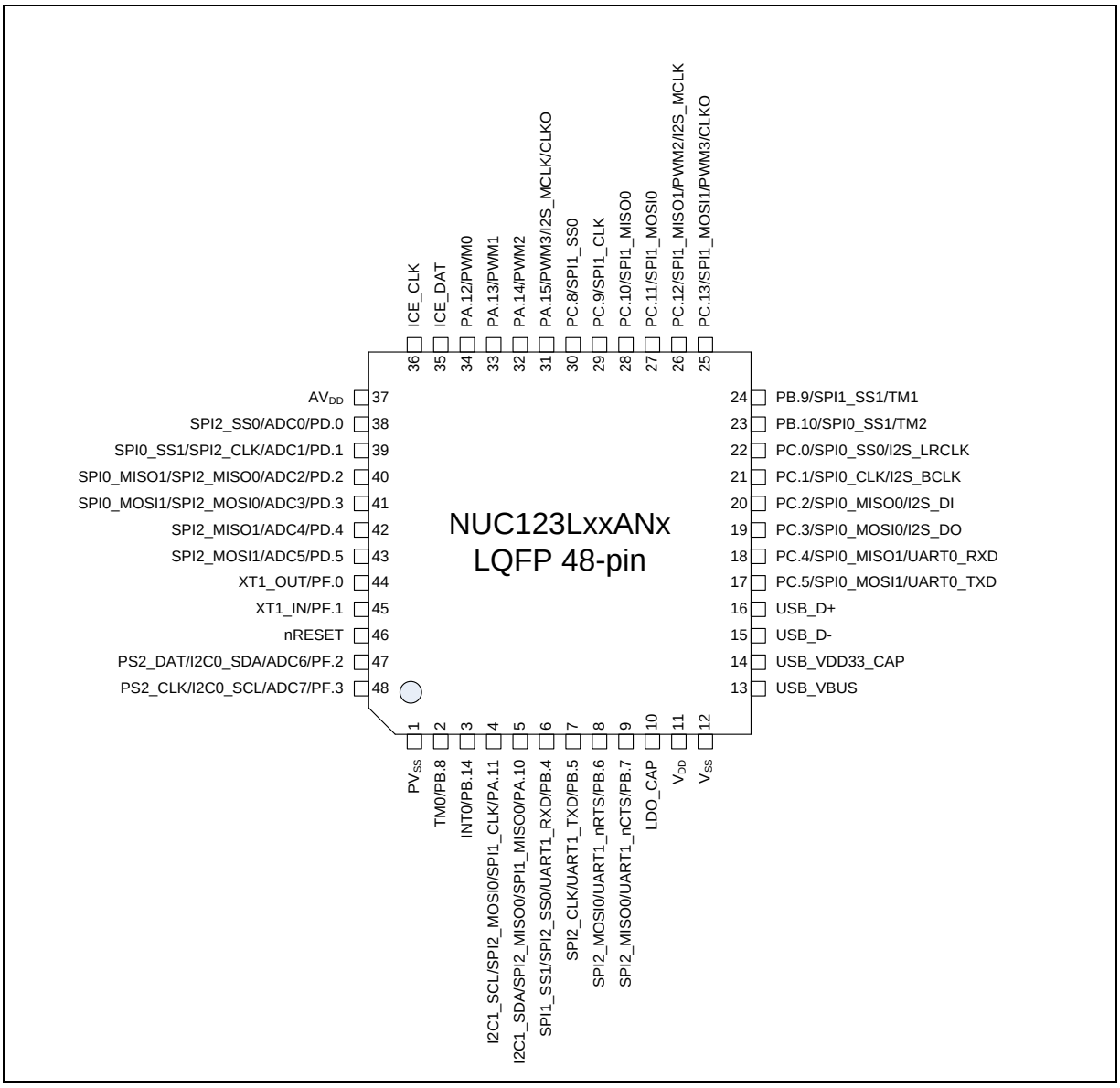


Figure 4-3 NuMicro® NUC123LxxANx LQFP 48-pin Diagram

4.3.1.3 *NuMicro® NUC123ZxxANx QFN 33 pin*

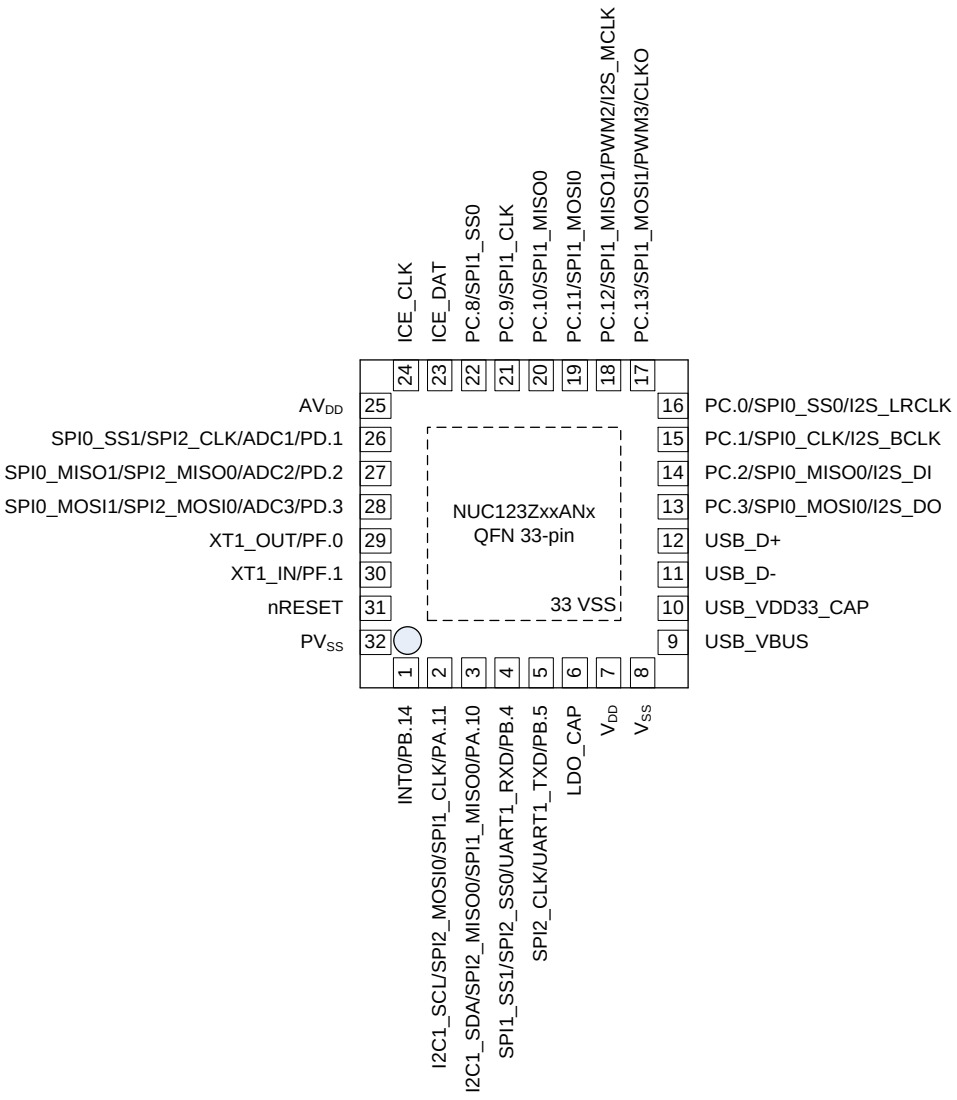


Figure 4-4 NuMicro® NUC123ZxxANx QFN 33-pin Diagram

4.3.2 NuMicro® NUC123xxxAEx Pin Diagram

4.3.2.1 *NuMicro® NUC123SxxAEx LQFP 64 pin*

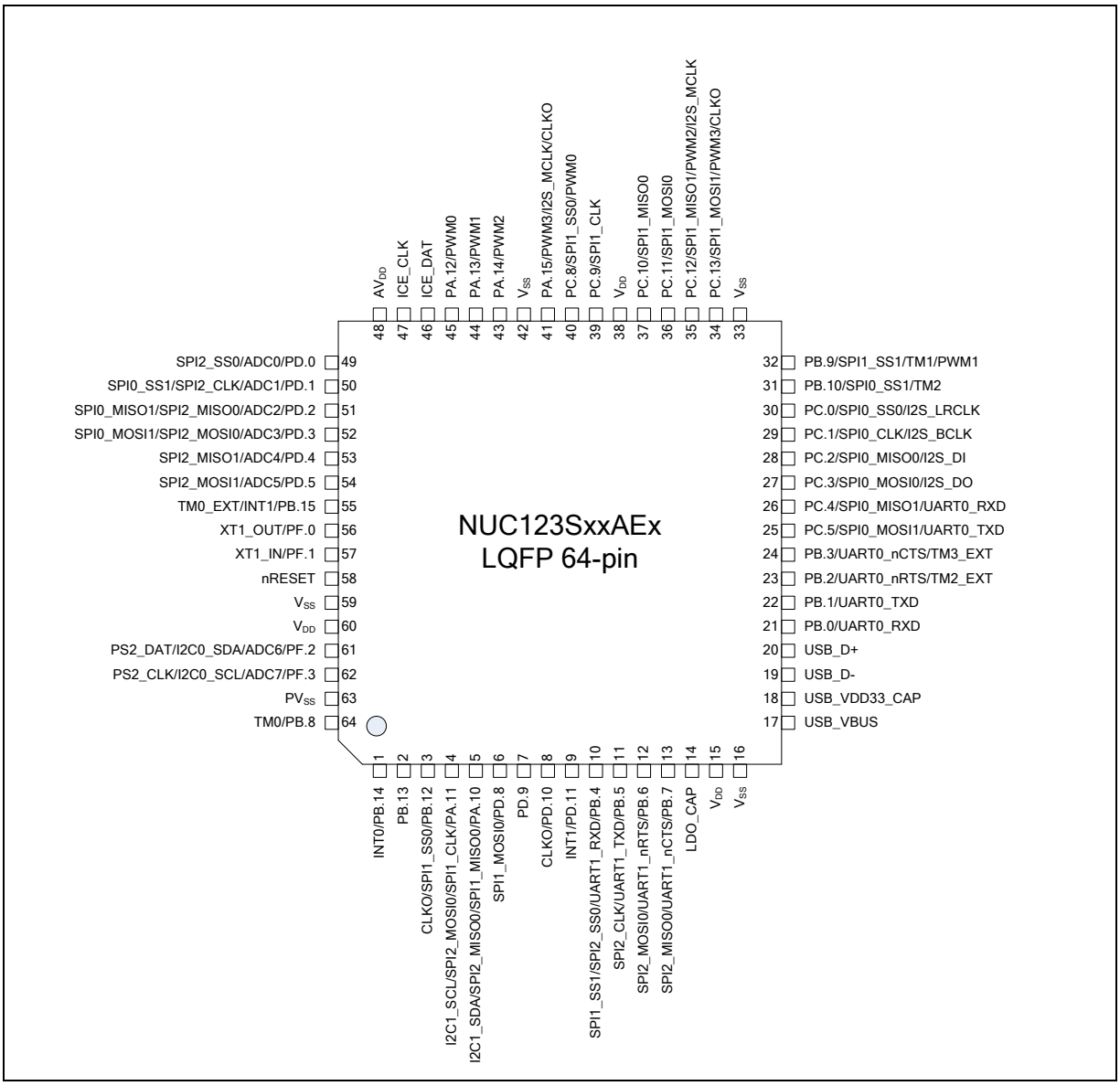


Figure 4-5 NuMicro® NUC123SxxAEx LQFP 64-pin Diagram

4.3.2.2 *NuMicro® NUC123LxxAEx LQFP 48 pin*

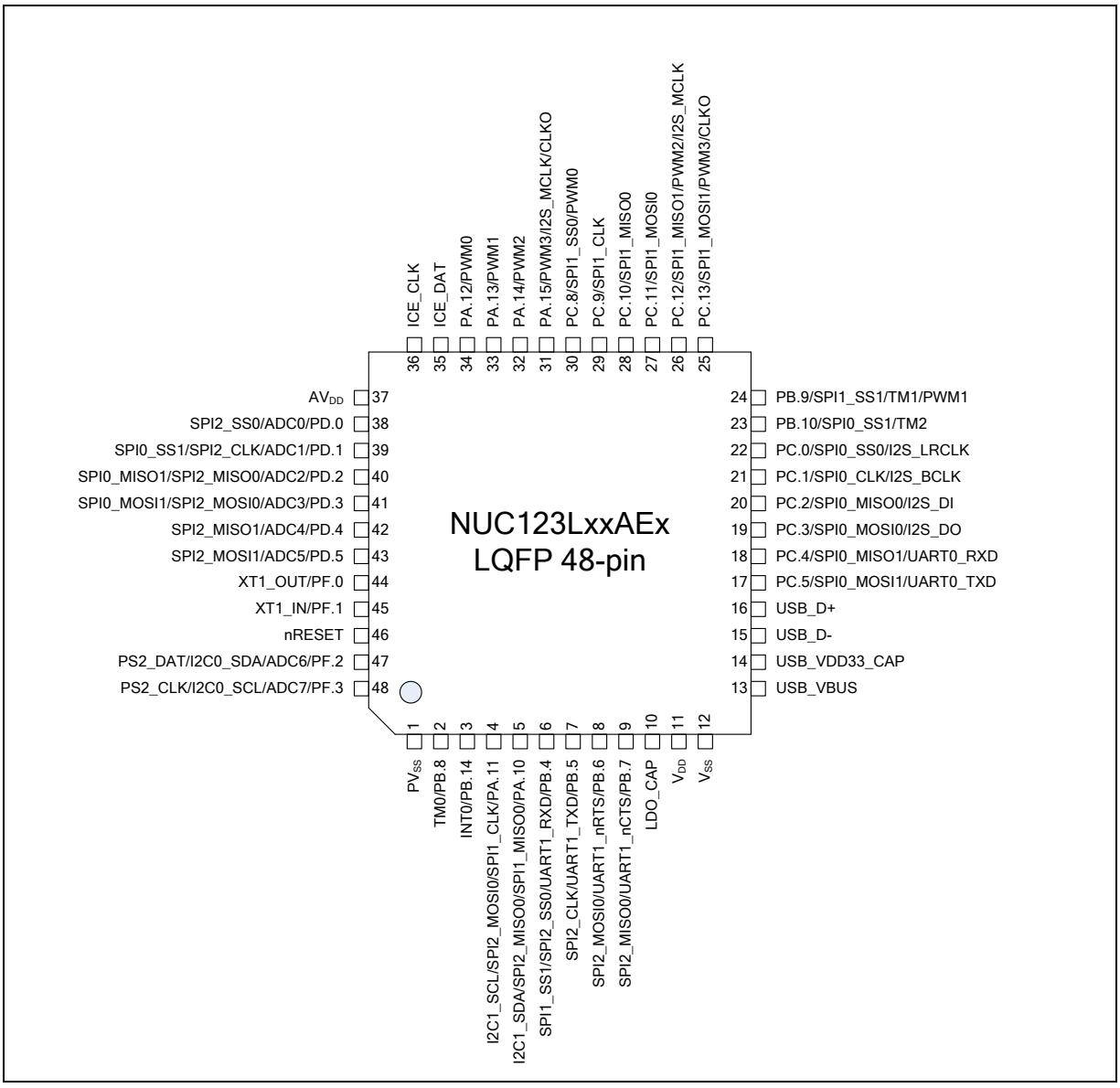


Figure 4-6 NuMicro® NUC123LxxAEx LQFP 48-pin Diagram

4.3.2.3 *NuMicro® NUC123ZxxAEx QFN 33 pin*

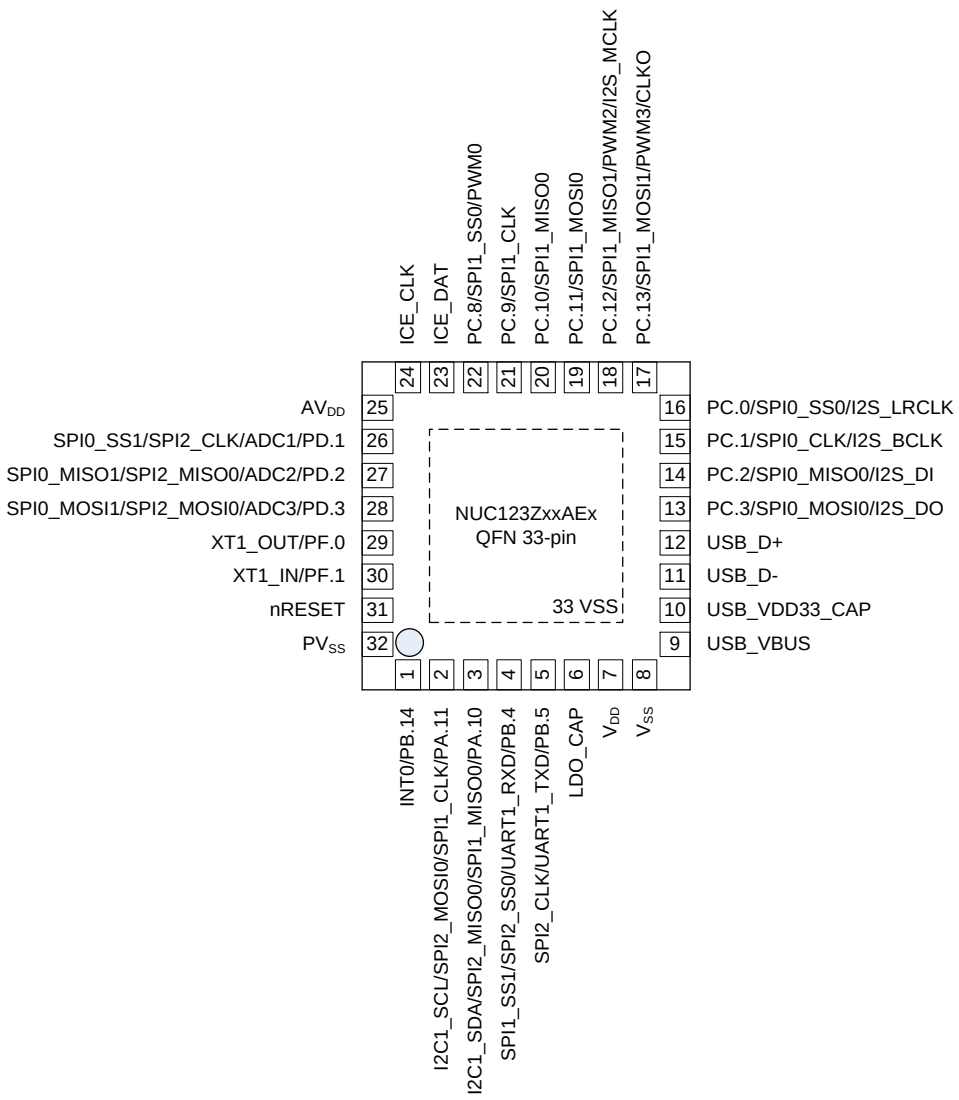


Figure 4-7 NuMicro® NUC123ZxxAEx QFN 33-pin Diagram

4.4 Pin Description

4.4.1 NuMicro® NUC123 Pin Description

Pin No			Pin Name	Type	Description
LQFP 64-pin	LQFP 48-pin	QFN 33-pin			
1	3	1	PB.14	**I/O**	Digital GPIO pin
			INT0	**I**	External interrupt 0 input pin
2			PB.13	**I/O**	Digital GPIO pin
3			PB.12	**I/O**	Digital GPIO pin
			SPI1_SS0	**I/O**	SPI1 1st slave select pin
			CLKO	**O**	Frequency Divider output pin
4	4	2	PA.11	**I/O**	Digital GPIO pin
			SPI1_CLK	**I/O**	SPI1 serial clock pin
			SPI2_MOSI0	**I/O**	SPI2 1st MOSI (Master Out, Slave In) pin
			I2C1_SCL	**I/O**	I^2C1 clock pin
5*	5*	3*	PA.10	**I/O**	Digital GPIO pin
			SPI1_MISO0	**I/O**	SPI1 1st MISO (Master In, Slave Out) pin
			SPI2_MISO0	**I/O**	SPI2 1st MISO (Master In, Slave Out) pin
			I2C1_SDA	**I/O**	I^2C1 data input/output pin
6			PD.8	**I/O**	Digital GPIO pin
			SPI1_MOSI0	**I/O**	SPI1 1st MOSI (Master Out, Slave In) pin
7			PD.9	**I/O**	Digital GPIO pin
8			PD.10	**I/O**	Digital GPIO pin
			CLKO	**O**	Frequency Divider output pin
9			PD.11	**I/O**	Digital GPIO pin
			INT1	**I**	External interrupt 1 input pin
10	6	4	PB.4	**I/O**	Digital GPIO pin
			UART1_RXD	**I**	UART1 data receiver input pin
			SPI2_SS0	**I/O**	SPI2 1st slave select pin
			SPI1_SS1	**I/O**	SPI1 2nd slave select pin
11	7	5	PB.5	**I/O**	Digital GPIO pin
			UART1_TXD	**O**	UART1 data transmitter output pin
			SPI2_CLK	**I/O**	SPI2 serial clock pin
12	8		PB.6	**I/O**	Digital GPIO pin
			UART1_nRTS	**O**	UART1 request to send output pin
			SPI2_MOSI0	**I/O**	SPI2 1st MOSI (Master Out, Slave In) pin

13	9		PB.7	**I/O**	Digital GPIO pin
			UART1_nCTS	**I**	UART1 clear to send input pin
			SPI2_MISO0	**I/O**	SPI2 1st MISO (Master In, Slave Out) pin
14	10	6	LDO_CAP	**P**	LDO output pin
15	11	7	V_{DD}	**P**	Power supply for I/O ports and LDO source for internal PLL and digital function. Voltage range is 2.5V ~ 5V.
16	12	8	V_{SS}	**P**	Ground
17	13	9	USB_VBUS	**USB**	Power supply from USB host or hub
18	14	10	USB_VDD33_CAP	**USB**	Internal power regulator output 3.3V decoupling pin
19	15	11	USB_D-	**USB**	USB differential signal D-
20	16	12	USB_D+	**USB**	USB differential signal D+
21			PB.0	**I/O**	Digital GPIO pin
			UART0_RXD	**I**	UART0 data receiver input pin
22			PB.1	**I/O**	Digital GPIO pin
			UART0_TXD	**O**	UART0 data transmitter output pin
23			PB.2	**I/O**	Digital GPIO pin
			UART0_nRTS	**O**	UART0 request to send output pin
			TM2_EXT	**I**	Timer2 external capture input pin
24			PB.3	**I/O**	Digital GPIO pin
			UART0_nCTS	**I**	UART0 clear to send input pin
			TM3_EXT	**I**	Timer3 external capture input pin
25	17		PC.5	**I/O**	Digital GPIO pin
			SPI0_MOSI1	**I/O**	SPI0 2nd MOSI (Master Out, Slave In) pin
			UART0_TXD	**O**	UART0 data transmitter output pin
26	18		PC.4	**I/O**	Digital GPIO pin
			SPI0_MISO1	**I/O**	SPI0 2nd MISO (Master In, Slave Out) pin
			UART0_RXD	**I**	UART0 data receiver input pin
27	19	13	PC.3	**I/O**	Digital GPIO pin
			SPI0_MOSI0	**I/O**	SPI0 1st MOSI (Master Out, Slave In) pin
			I2S_DO	**O**	I^2S data output pin
28	20	14	PC.2	**I/O**	Digital GPIO pin
			SPI0_MISO0	**I/O**	SPI0 1st MISO (Master In, Slave Out) pin
			I2S_DI	**I**	I^2S data input pin
29	21	15	PC.1	**I/O**	Digital GPIO pin
			SPI0_CLK	**I/O**	SPI0 serial clock pin
			I2S_BCLK	**I/O**	I^2S bit clock pin

30	22	16	PC.0	**I/O**	Digital GPIO pin
			SPI0_SS0	**I/O**	SPI0 1st slave select pin
			I2S_LRCLK	**I/O**	I^2S left/right channel clock pin
31	23		PB.10	**I/O**	Digital GPIO pin
			SPI0_SS1	**I/O**	SPI0 2nd slave select pin
			TM2	**I/O**	Timer2 event counter input / toggle output pin
32	24		PB.9	**I/O**	Digital GPIO pin
			SPI1_SS1	**I/O**	SPI1 2nd slave select pin
			TM1	**I/O**	Timer1 event counter input / toggle output pin
			PWM1	**I/O**	PWM1 PWM output / capture input pin (NUC123xxxAEx Only)
33			V_{SS}	**P**	Ground
34	25	17	PC.13	**I/O**	Digital GPIO pin
			SPI1_MOSI1	**I/O**	SPI1 2nd MOSI (Master Out, Slave In) pin
			PWM3	**I/O**	PWM3 PWM output / capture input pin
			CLKO	**O**	Frequency Divider output pin
35	26	18	PC.12	**I/O**	Digital GPIO pin
			SPI1_MISO1	**I/O**	SPI1 2nd MISO (Master In, Slave Out) pin
			PWM2	**I/O**	PWM2 PWM output / capture input pin
			I2S_MCLK	**O**	I^2S master clock output pin
36	27	19	PC.11	**I/O**	Digital GPIO pin
			SPI1_MOSI0	**I/O**	SPI1 1st MOSI (Master Out, Slave In) pin
37	28	20	PC.10	**I/O**	Digital GPIO pin
			SPI1_MISO0	**I/O**	SPI1 1st MISO (Master In, Slave Out) pin
38			V_{DD}	**P**	Power supply for I/O ports and LDO source for internal PLL and digital function. Voltage range is 2.5V ~ 5V.
39	29	21	PC.9	**I/O**	Digital GPIO pin
			SPI1_CLK	**I/O**	SPI1 serial clock pin
40	30	22	PC.8	**I/O**	Digital GPIO pin
			SPI1_SS0	**I/O**	SPI1 1st slave select pin
			PWM0	**I/O**	PWM0 PWM output / capture input pin (NUC123xxxAEx Only)
41	31		PA.15	**I/O**	Digital GPIO pin
			PWM3	**I/O**	PWM3 PWM output / capture input pin
			I2S_MCLK	**O**	I^2S master clock output pin
			CLKO	**O**	Frequency Divider output pin
42			V_{SS}	**P**	Ground

43	32		PA.14	**I/O**	Digital GPIO pin
			PWM2	**I/O**	PWM2 PWM output / capture input pin
44	33		PA.13	**I/O**	Digital GPIO pin
			PWM1	**I/O**	PWM1 PWM output / capture input pin
45	34		PA.12	**I/O**	Digital GPIO pin
			PWM0	**I/O**	PWM0 PWM output / capture input pin
46	35	23	ICE_DAT	**I/O**	Serial wired debugger data pin **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_DAT pin.
47	36	24	ICE_CLK	**I**	Serial wired debugger clock input pin **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_CLK pin.
48	37	25	AV_{DD}	**AP**	Power supply for internal analog circuit
49	38		PD.0	**I/O**	Digital GPIO pin
			ADC0	**AI**	ADC channel 0 analog input pin
			SPI2_SS0	**I/O**	SPI2 1st slave select pin
50	39	26	PD.1	**I/O**	Digital GPIO pin
			SPI2_CLK	**I/O**	SPI2 serial clock pin
			SPI0_SS1	**I/O**	SPI0 2nd slave select pin
			ADC1	**AI**	ADC channel 1 analog input pin
51	40	27	PD.2	**I/O**	Digital GPIO pin
			SPI2_MISO0	**I/O**	SPI2 1st MISO (Master In, Slave Out) pin
			SPI0_MISO1	**I/O**	SPI0 2nd MISO (Master In, Slave Out) pin
			ADC2	**AI**	ADC channel 2 analog input pin
52	41	28	PD.3	**I/O**	Digital GPIO pin
			SPI2_MOSI0	**I/O**	SPI2 1st MOSI (Master Out, Slave In) pin
			SPI0_MOSI1	**I/O**	SPI0 2nd MOSI (Master Out, Slave In) pin
			ADC3	**AI**	ADC channel 3 analog input pin
53	42		PD.4	**I/O**	Digital GPIO pin
			ADC4	**AI**	ADC channel 4 analog input pin
			SPI2_MISO1	**I/O**	SPI2 2nd MISO (Master In, Slave Out) pin
54	43		PD.5	**I/O**	Digital GPIO pin
			ADC5	**AI**	ADC channel 5 analog input pin
			SPI2_MOSI1	**I/O**	SPI2 2nd MOSI (Master Out, Slave In) pin
55			PB.15	**I/O**	Digital GPIO pin
			INT1	**I**	External interrupt 1 input pin
			TM0_EXT	**I**	Timer0 external capture input pin

56	44	29	PF.0	**I/O**	Digital GPIO pin
			XT1_OUT	**O**	External 4~24 MHz high speed crystal output pin
57	45	30	PF.1	**I/O**	Digital GPIO pin
			XT1_IN	**I**	External 4~24 MHz high speed crystal input pin
58	46	31	nRESET	**I**	External reset input: Low active, set this pin low reset chip to initial state. With internal pull-up. **Note:** It is recommended to use 10 kΩ pull-up resistor and 10 μF capacitor on nRESET pin.
59			V_{SS}	**P**	Ground
60			V_{DD}	**P**	Power supply for I/O ports and LDO source for internal PLL and digital circuit. Voltage range is 2.5 V ~ 5V.
61	47		PF.2	**I/O**	Digital GPIO pin
			ADC6	**AI**	ADC channel 6 analog input pin
			I2C0_SDA	**I/O**	I^2C0 data input/output pin
			PS2_DAT	**I/O**	PS/2 data pin
62	48		PF.3	**I/O**	Digital GPIO pin
			ADC7	**AI**	ADC channel 7 analog input pin
			I2C0_SCL	**I/O**	I^2C0 clock pin
			PS2_CLK	**I/O**	PS/2 clock pin
63	1	32	PV_{SS}	**P**	PLL ground
64	2		PB.8	**I/O**	Digital GPIO pin
			TM0	**I/O**	Timer0 event counter input / toggle output pin

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

5 BLOCK DIAGRAM

5.1 NuMicro® NUC123 Block Diagram

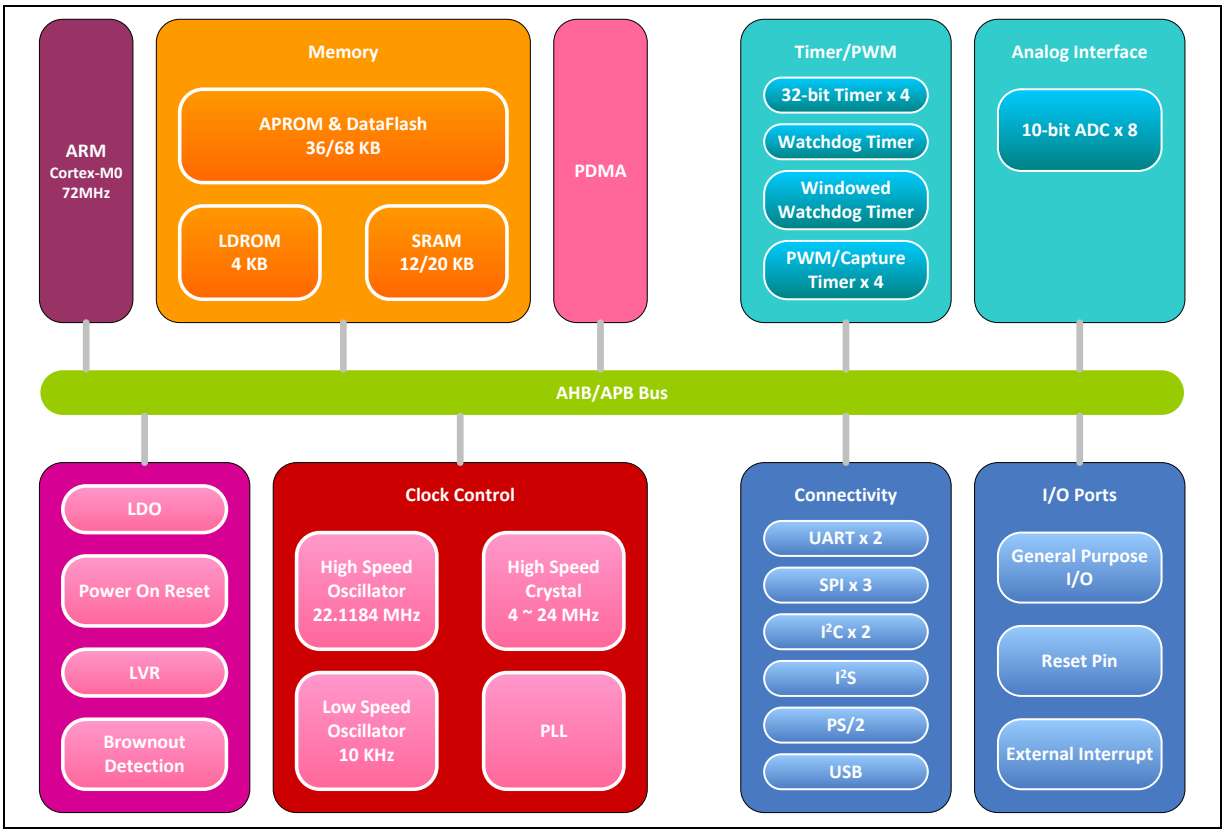


Figure 5-1 NuMicro® NUC123 Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 ARM® Cortex®-M0 Core

The Cortex®-M0 processor, a configurable, multistage, 32-bit RISC processor, has an AMBA AHB-Lite interface and includes an NVIC component. The processor has optional hardware debug functionality, can execute Thumb code, and is compatible with other Cortex®-M profile processors. The profile supports two modes -Thread mode and Handler mode. Handler mode is entered as a result of an exception. An exception return can only be issued in Handler mode. Thread mode is entered on Reset, and can be entered as a result of an exception return. Figure 6-1 shows the functional controller of processor.

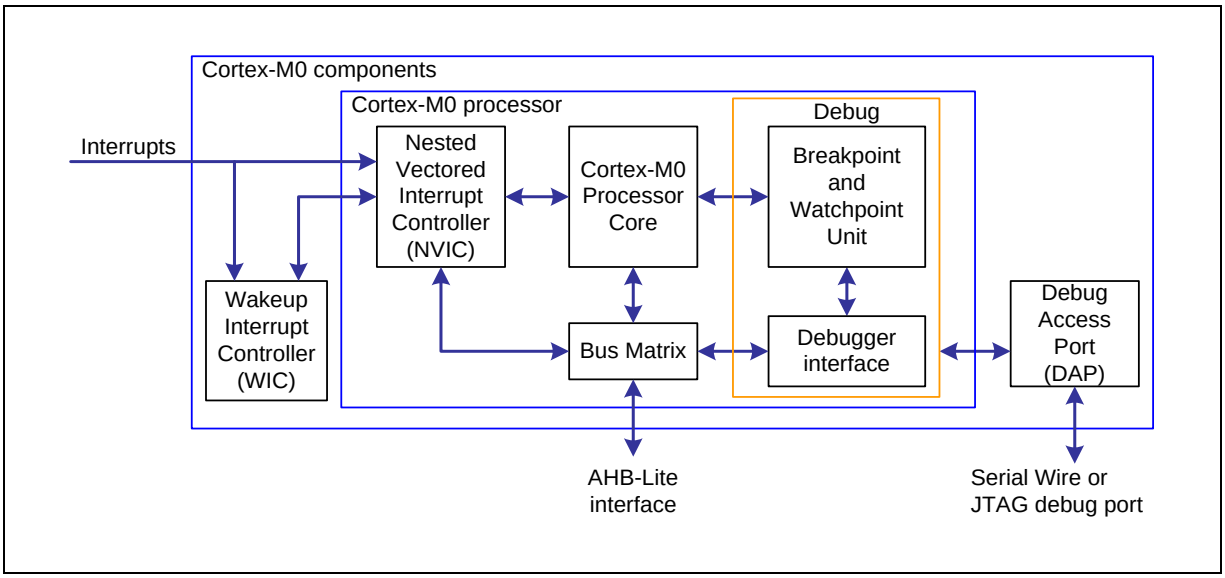


Figure 6-1 Functional Controller Diagram

The implemented device provides:

- 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 supporting little-endian data accesses
  - Ability to have deterministic, fixed-latency, interrupt handling
  - Load/store-multiples and multicycle-multiplies abandoned and restarted to facilitate rapid interrupt handling
  - C Application Binary Interface compliant exception model, which is the ARMv6-M, C Application Binary Interface (C-ABI) compliant exception model that enables the use of pure C functions as interrupt handlers
  - Low power sleep mode entry using Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or the return from interrupt sleep-on-exit feature
- NVIC :

- 32 external interrupt inputs, each with four levels of priority
- Dedicated Non-Maskable Interrupt (NMI) input
- Supports both level-sensitive and pulse-sensitive interrupt lines
- Supports Wake-up Interrupt Controller (WIC) with 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 providing simple integration to all system peripherals and memory
  - Single 32-bit slave port supporting the DAP (Debug Access Port)

6.2 System Manager

6.2.1 Overview

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

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

6.2.2 System Reset

The system reset can be issued by one of the events listed below. These reset event flags can be read from RSTSRC register to determine the reset source. Hardware reset can reset chip through peripheral reset signals. Software reset can trigger reset through control registers.

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

Power-on Reset or CHIP_RST (IPRST1[0]) resets the whole chip including all peripherals, external crystal circuit and BS (ISPCON[1]) bit.

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

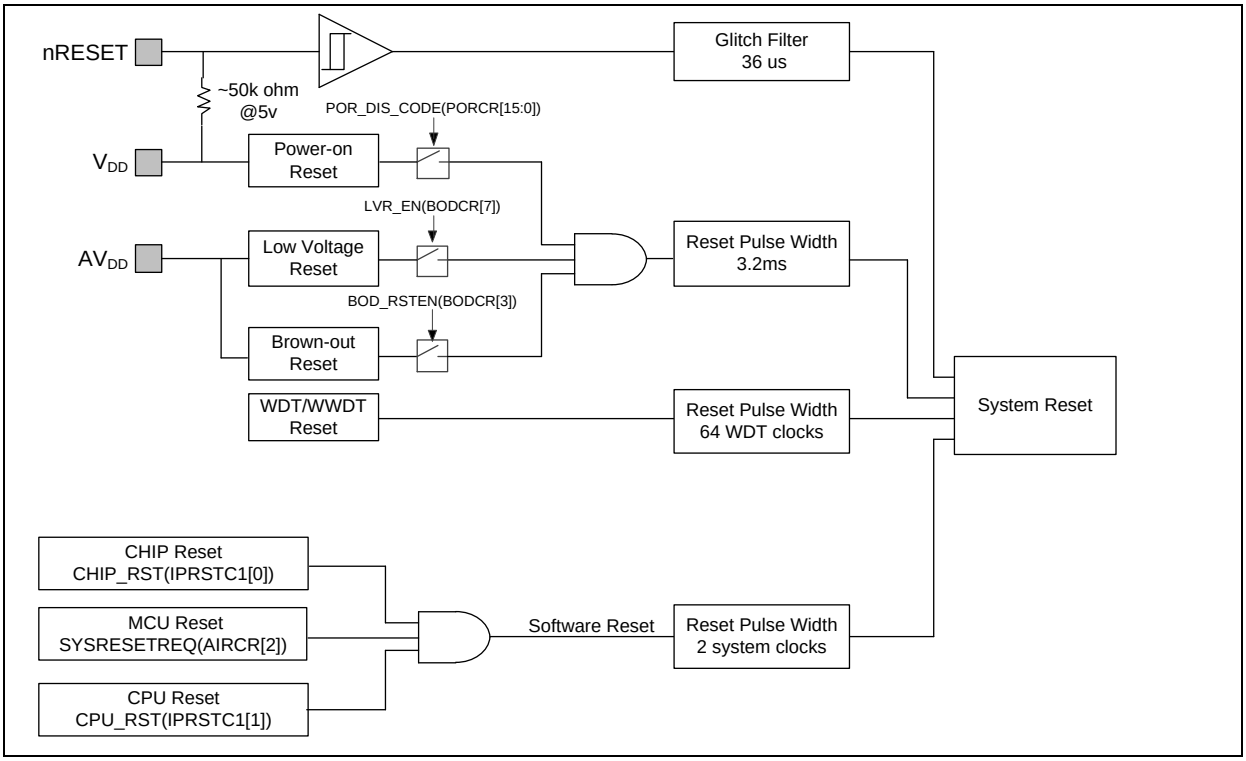


Figure 6-2 System Reset Resources

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

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

WDT_S (CLKSEL1[1:0])	0x3	0x3	-	-	-	-	-	-
XTL12M_STB (CLKSTATUS[0])	0x0	-	-	-	-	-	-	-
PLL_STB (CLKSTATUS[2])	0x0	-	-	-	-	-	-	-
OSC10K_STB (CLKSTATUS[3])	0x0	-	-	-	-	-	-	-
OSC22M_STB (CLKSTATUS[4])	0x0	-	-	-	-	-	-	-
CLK_SW_FAIL (CLKSTATUS[7])	0x0	0x0	0x0	0x0	0x0	0x0	0x0	-
WTE (WTCR[7])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0		
WTCR	0x0700	0x0700	0x0700	0x0700	0x0700	0x0700	-	-
WTCRALT	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	-	-
WWDTRLD	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	-	-
WWDTCR	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	-	-
WWDTSR	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	-	-
WWDTCVR	0x3F	0x3F	0x3F	0x3F	0x3F	0x3F	-	-
BS (ISPCON[1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	-
DFBADR	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	-	-
CBS (ISPSTA[2:1))	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	-
VECMAP (ISPSTA[20:9]) (NUC123xxxAEx Only)	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	-	-
Other Peripheral Registers	Reset Value							-
FMC Registers	Reset Value							
Note: '-' means that the value of register keeps original setting.								

Table 6-1 Reset Value of Registers

6.2.2.1 *nRESET Reset*

The nRESET reset means to generate a reset signal by pulling low nRESET pin, which is an asynchronous reset input pin and can be used to reset system at any time. When the nRESET voltage is lower than 0.2 V_{DD} and the state keeps longer than 36 us (glitch filter), chip will be reset. The

nRESET reset will control the chip in reset state until the nRESET voltage rises above 0.7 V_{DD} and the state keeps longer than 36 us (glitch filter). The RSTS_RESET (RSTSRC[1]) will be set to 1 if the previous reset source is nRESET reset.

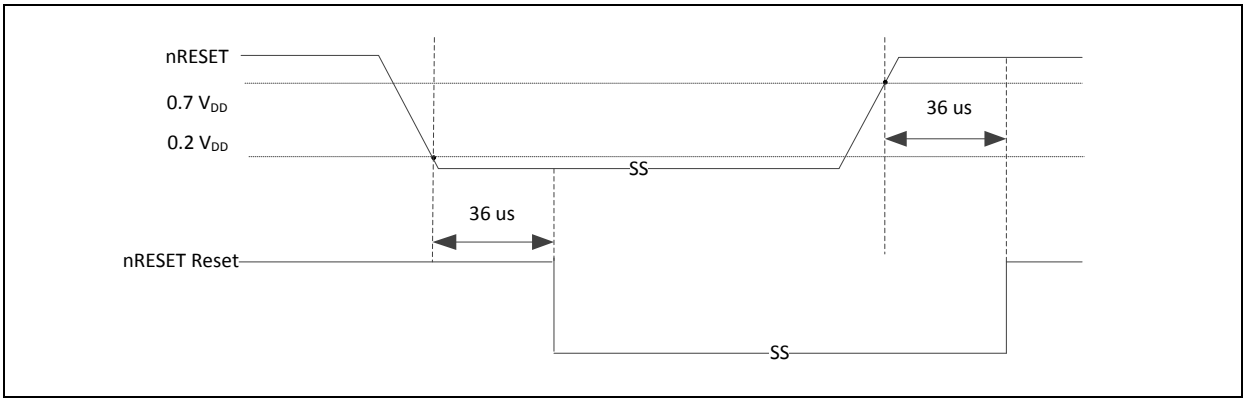


Figure 6-3 shows the nRESET reset waveform.

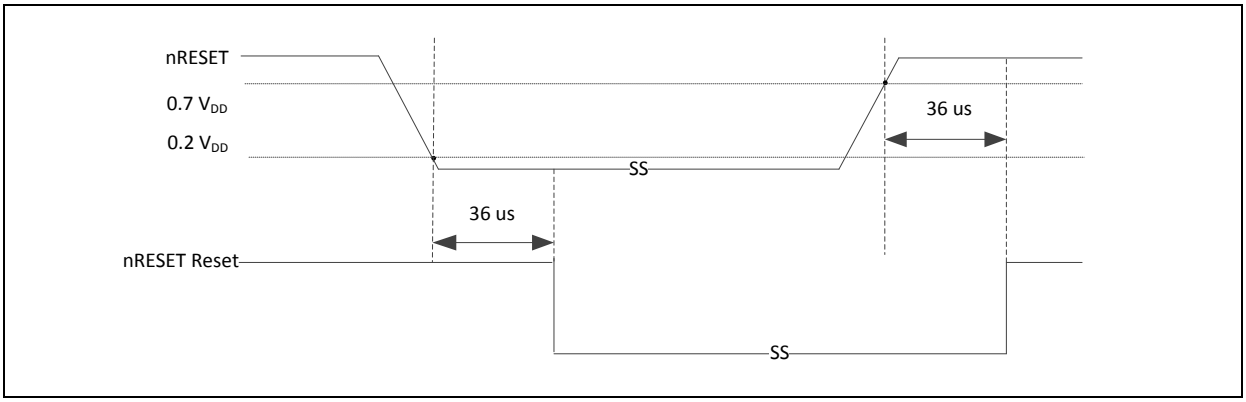


Figure 6-3 nRESET Reset Waveform

6.2.2.2 *Power-On Reset (POR)*

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

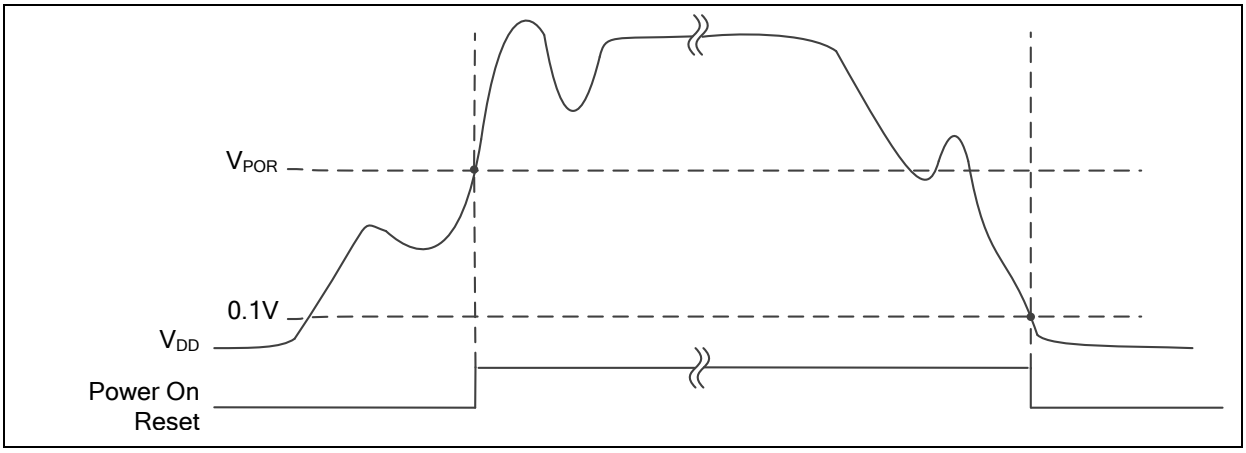


Figure 6-4 Power-on Reset (POR) Waveform

6.2.2.3 *Low Voltage Reset (LVR)*

If the Low Voltage Reset function is enabled by setting the Low Voltage Reset Enable Bit LVR_EN (BODCR[7]) to 1, after 100us delay, LVR detection circuit will be stable and the LVR function will be active. Then LVR function will detect AV_{DD} during system operation. When the AV_{DD} voltage is lower than V_{LVR} and the state keeps longer than De-glitch time (16*HCLK cycles), chip will be reset. The LVR reset will control the chip in reset state until the AV_{DD} voltage rises above V_{LVR} and the state keeps longer than De-glitch time. The RSTS_RESET (RSTSRC[1]) will be set to 1 if the previous reset source is nRESET reset. Figure 6-5 shows the Low Voltage Reset waveform.

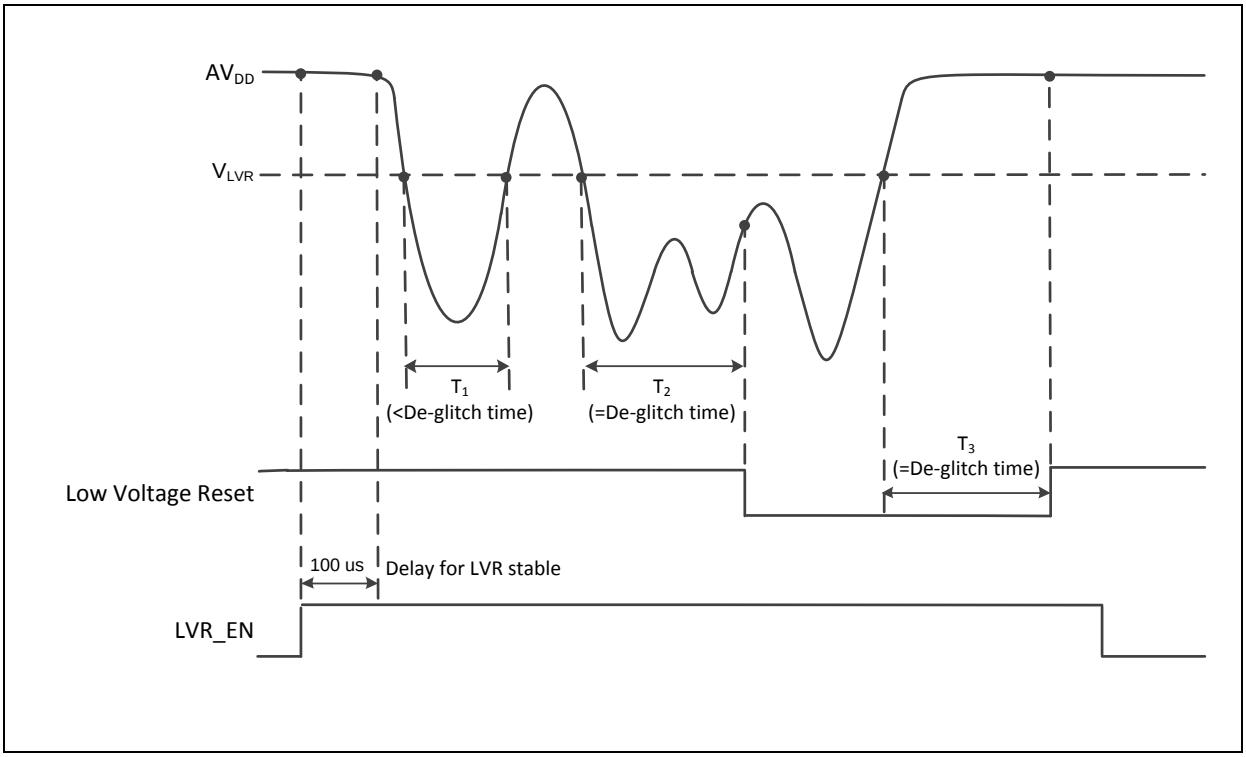


Figure 6-5 Low Voltage Reset Waveform

6.2.2.4 *Brown-out Detector Reset (BOD Reset)*

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

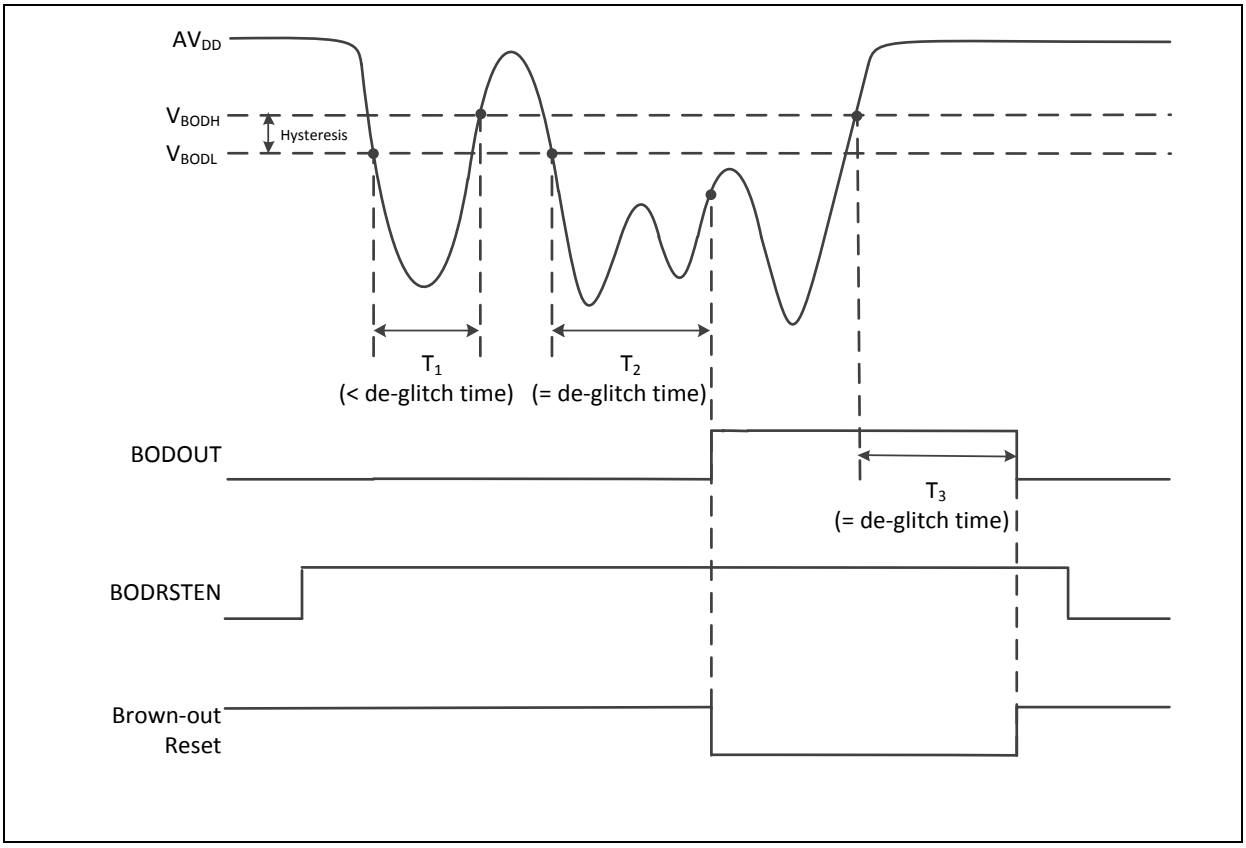


Figure 6-6 Brown-Out Detector Waveform

6.2.2.5 *Watch Dog Timer Reset*

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

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

6.2.2.6 *CPU Reset, CHIP Reset and MCU Reset*

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

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

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

6.2.3 Power modes and Wake-up sources

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

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

Table 6-2 Power Mode Difference Table

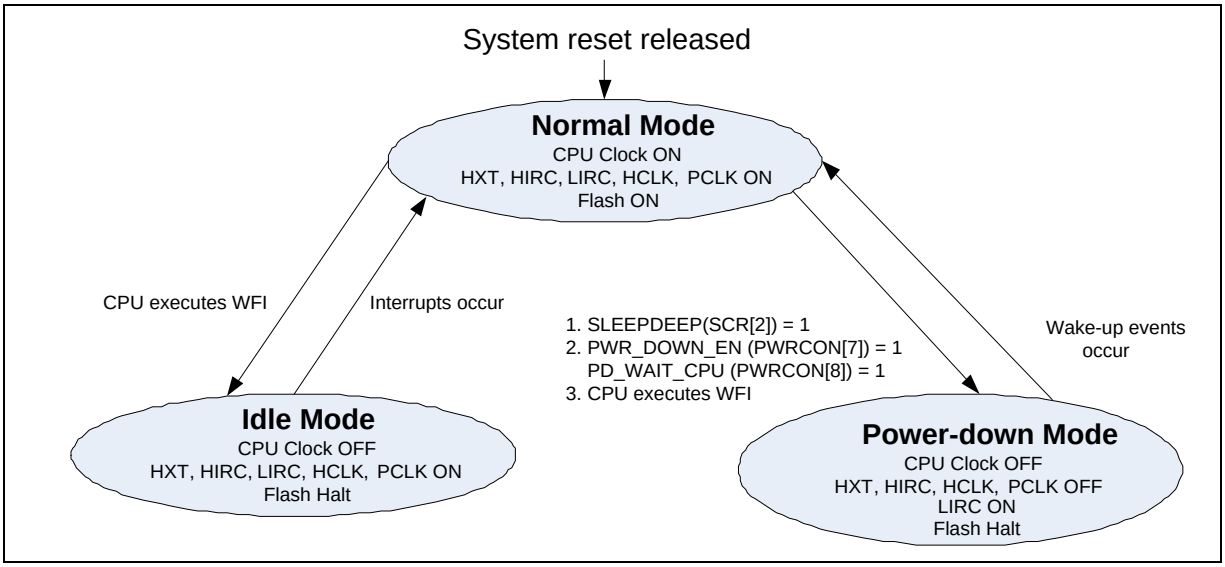


Figure 6-7 Power Mode State Machine

1. LIRC (10 kHz OSC) ON or OFF depends on Software setting in run mode.
2. If TIMER clock source is selected as LIRC and LIRC is on.
3. If WDT clock source is selected as LIRC and LIRC is on.

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

Table 6-3 Clocks in Power Modes

Wake-up sources in Power-down mode:

WDT, I²C, Timer, UART, BOD, GPIO and USB

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

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

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

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

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

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

6.2.4 System Power Distribution

In this chip, power distribution is divided into three segments:

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

The outputs of internal voltage regulators, LDO and USB_VDD33_CAP, require an external capacitor which should be located close to the corresponding pin. Analog power (AV_{DD}) should be the same voltage level of the digital power (V_{DD}). Figure 6-8 shows the power distribution of the NuMicro® NUC123 series.

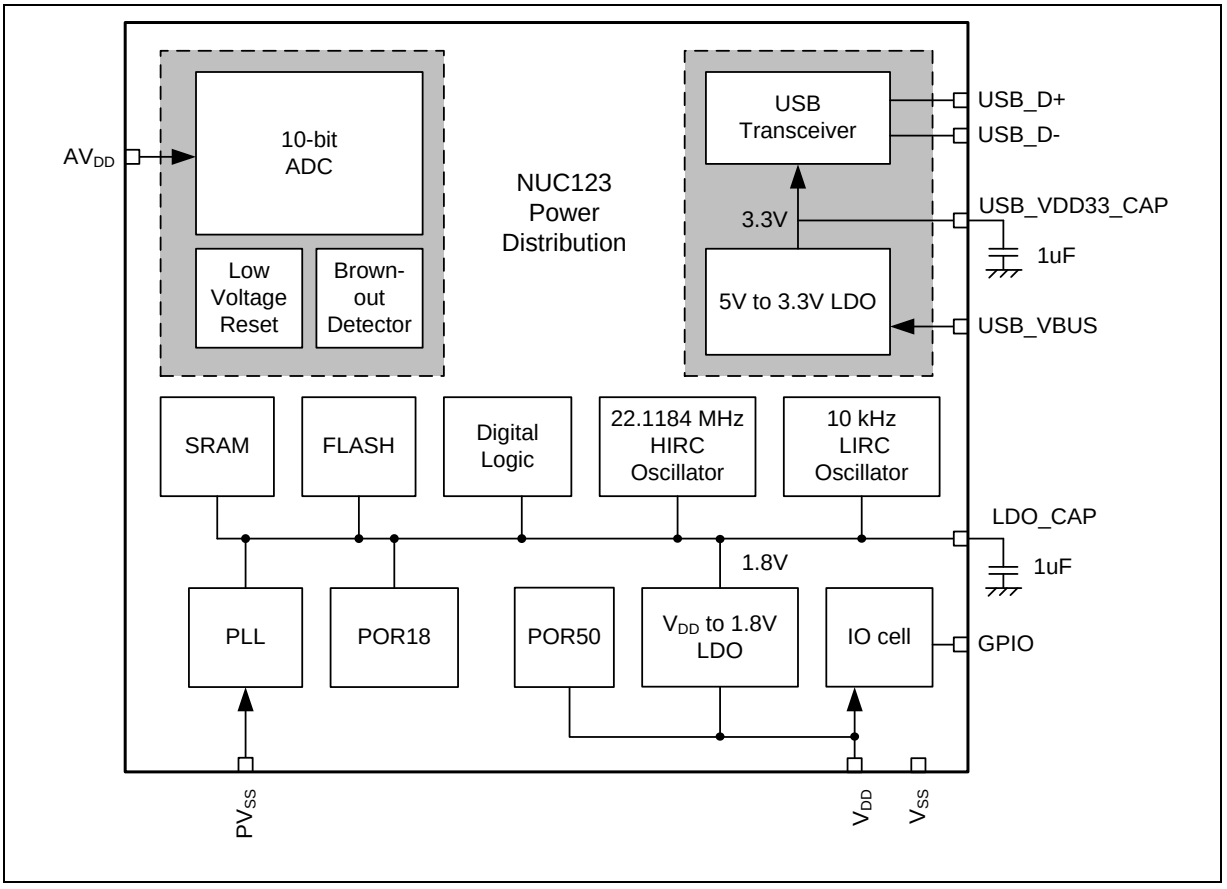


Figure 6-8 NuMicro® NUC123 Power Distribution Diagram

6.2.5 System Memory Map

The NuMicro® NUC123 series provides 4G-byte addressing space. The memory locations assigned to each on-chip controllers are shown in the Table 6-5. The detailed register definition, memory space, and programming detailed will be described in the following sections for each on-chip peripherals. The NuMicro® NUC123 Series only supports little-endian data format.

Address Space	Token	Controllers
Flash and SRAM Memory Space		
0x0000_0000 – 0x0000_FFFF	FLASH_BA	FLASH Memory Space (64KB)
0x2000_0000 – 0x2000_4FFF	SRAM_BA	SRAM Memory Space (20KB)
AHB Controllers Space (0x5000_0000 – 0x501F_FFFF)		
0x5000_0000 – 0x5000_01FF	GCR_BA	System Global Control Registers
0x5000_0200 – 0x5000_02FF	CLK_BA	Clock Control Registers
0x5000_0300 – 0x5000_03FF	INT_BA	Interrupt Multiplexer Control Registers
0x5000_4000 – 0x5000_7FFF	GPIO_BA	GPIO Control Registers
0x5000_8000 – 0x5000_BFFF	PDMA_BA	Peripheral DMA Control Registers
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash Memory Control Registers
APB1 Controllers Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4000_4000 – 0x4000_7FFF	WDT_BA	Watchdog/Window Watchdog Timer Control Registers
0x4001_0000 – 0x4001_3FFF	TMR01_BA	Timer0/Timer1 Control Registers
0x4002_0000 – 0x4002_3FFF	I2C0_BA	I^2C0 Interface Control Registers
0x4003_0000 – 0x4003_3FFF	SPI0_BA	SPI0 with master/slave function Control Registers
0x4003_4000 – 0x4003_7FFF	SPI1_BA	SPI1 with master/slave function Control Registers
0x4004_0000 – 0x4004_3FFF	PWMA_BA	PWM0/1/2/3 Control Registers
0x4005_0000 – 0x4005_3FFF	UART0_BA	UART0 Control Registers
0x4006_0000 – 0x4006_3FFF	USBD_BA	USB 2.0 FS device Controller Registers
0x400E_0000 – 0x400E_FFFF	ADC_BA	Analog-Digital-Converter (ADC) Control Registers
APB2 Controllers Space (0x4010_0000 ~ 0x401F_FFFF)		
0x4010_0000 – 0x4010_3FFF	PS2_BA	PS/2 Interface Control Registers
0x4011_0000 – 0x4011_3FFF	TMR23_BA	Timer2/Timer3 Control Registers
0x4012_0000 – 0x4012_3FFF	I2C1_BA	I^2C1 Interface Control Registers
0x4013_0000 – 0x4013_3FFF	SPI2_BA	SPI2 with master/slave function Control Registers
0x4015_0000 – 0x4015_3FFF	UART1_BA	UART1 Control Registers
0x401A_0000 – 0x401A_3FFF	I2S_BA	I^2S Interface Control Registers
System Controllers Space (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SCS_BA	System Timer Control Registers
0xE000_E100 – 0xE000_ECFF	SCS_BA	External Interrupt Controller Control Registers

0xE000_ED00 – 0xE000_ED8F	SCS_BA	System Control Registers

Table 6-5 Address Space Assignments for On-Chip Controllers

6.2.6 System Manager Control Registers

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	0x1001_23XX[1]
RSTSRC	GCR_BA+0x04	R/W	System Reset Source Register	0x0000_00XX
IPRSTC1	GCR_BA+0x08	R/W	Peripheral Reset Control Register1	0x0000_0000
IPRSTC2	GCR_BA+0x0C	R/W	Peripheral Reset Control Register2	0x0000_0000
BODCR	GCR_BA+0x18	R/W	Brown-out Detector Control Register	0x0000_008X
PORCR	GCR_BA+0x24	R/W	Power-on-Reset Control Register	0x0000_XXXX
GPA_MFP	GCR_BA+0x30	R/W	GPIOA Multiple Function and Input Type Control Register	0x0000_0000
GPB_MFP	GCR_BA+0x34	R/W	GPIOB Multiple Function and Input Type Control Register	0x0000_0000
GPC_MFP	GCR_BA+0x38	R/W	GPIOC Multiple Function and Input Type Control Register	0x0000_0000
GPD_MFP	GCR_BA+0x3C	R/W	GPIOD Multiple Function and Input Type Control Register	0x0000_0000
GPF_MFP	GCR_BA+0x44	R/W	GPIOF Multiple Function and Input Type Control Register	0x0000_000X
ALT_MFP	GCR_BA+0x50	R/W	Alternative Multiple Function Pin Control Register	0x0000_0000
ALT_MFP1	GCR_BA+0x54	R/W	Alternative Multiple Function Pin Control Register 1	0x0000_0000
GPA_IOCR	GCR_BA+0xC0	R/W	GPIOA I/O Control Register	0x0000_0000
GPB_IOCR	GCR_BA+0xC4	R/W	GPIOB I/O Control Register	0x0000_0000
GPD_IOCR	GCR_BA+0xCC	R/W	GPIOD I/O Control Register	0x0000_0000
REGWRPROT	GCR_BA+0x100	R/W	Register Write-Protection Control Register	0x0000_0000
GPA_MFPH	GCR_BA+0x134	R/W	GPIOA Multiple Function High Byte Control Register (NUC123xxxAEx Only)	0x0000_0000
GPB_MFPL	GCR_BA+0x138	R/W	GPIOB Multiple Function Low Byte Control Register (NUC123xxxAEx Only)	0x0000_0000
GPB_MFPH	GCR_BA+0x13C	R/W	GPIOB Multiple Function High Byte Control Register (NUC123xxxAEx Only)	0x0000_0000
GPC_MFPL	GCR_BA+0x140	R/W	GPIOC Multiple Function Low Byte Control Register (NUC123xxxAEx Only)	0x0000_0000
GPC_MFPH	GCR_BA+0x144	R/W	GPIOC Multiple Function High Byte Control Register (NUC123xxxAEx Only)	0x0000_0000
GPD_MFPL	GCR_BA+0x148	R/W	GPIOD Multiple Function Low Byte Control Register (NUC123xxxAEx Only)	0x0000_0000
GPD_MFPH	GCR_BA+0x14C	R/W	GPIOD Multiple Function High Byte Control Register (NUC123xxxAEx Only)	0x0000_0000

GPF_MFPL	GCR_BA+0x158	R/W	GPIOF Multiple Function Low Byte Control Register (NUC123xxxAEx Only)	0x0000_11XX

Note: [1] Depending on the part number.

Part Device Identification Number Register (PDID)

Register	Offset	R/W	Description	Reset Value
PDID	GCR_BA+0x00	R	Part Device Identification Number Register	0x1001_23xx[1]

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

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

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

Part No.	Package	FLASH	RAM	PDID
NUC123ZD4AN0	QFN-33	68	20	0x0001_2355
NUC123ZC2AN1	QFN-33	36	12	0x0001_2345
NUC123LD4AN0	LQFP-48	68	20	0x0001_2335
NUC123LC2AN1	LQFP-48	36	12	0x0001_2325
NUC123SD4AN0	LQFP-64	68	20	0x0001_2315
NUC123SC2AN1	LQFP-64	36	12	0x0001_2305

Table 6-6 NUC123xxxANx PDID Define Table

Part No.	Package	FLASH	RAM	PDID
NUC123ZD4AE0	QFN-33	68	20	0x1001_2355
NUC123ZC2AE1	QFN-33	36	12	0x1001_2345
NUC123LD4AE0	LQFP-48	68	20	0x1001_2335
NUC123LC2AE1	LQFP-48	36	12	0x1001_2325
NUC123SD4AE0	LQFP-64	68	20	0x1001_2315
NUC123SC2AE1	LQFP-64	36	12	0x1001_2305

Table 6-7 NUC123xxxAEx PDID Define Table

System Reset Source Register (RSTSRC)

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

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**RSTS_CPU**	**CPU Reset Flag** The RSTS_CPU flag Is set by hardware if software writes CPU_RST (IPRSTC1[1]) 1 to reset Cortex®-M0 kernel and flash memory controller (FMC). 0 = No reset from CPU. 1 = Cortex®-M0 CPU kernel and FMC are reset by software setting CPU_RST (IPRSTC1[1]) to 1. **Note:** This bit can be cleared by software writing '1'.
[6]	**Reserved**	Reserved.
[5]	**RSTS_SYS**	**SYS Reset Flag** The RSTS_SYS flag Is set by the "Reset Signal" from the Cortex®-M0 kernel to indicate the previous reset source. 0 = No reset from Cortex®-M0. 1 = The Cortex®-M0 had issued the reset signal to reset the system by writing 1 to bit SYSRESETREQ (AIRCR[2], Application Interrupt and Reset Control Register, address = 0xE000ED0C) in system control registers of Cortex®-M0 kernel. **Note:** This bit can be cleared by software writing '1'.
[4]	**RSTS_BOD**	**Brown-out Detector Reset Flag** The RSTS_BOD flag is set by the "Reset Signal" from the Brown-out Detector to indicate the previous reset source. 0 = No reset from BOD. 1 = The BOD had issued the reset signal to reset the system. **Note:** This bit can be cleared by software writing '1'.

[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:** This bit can be cleared by software writing '1'.
[2]	**RSTS_WDT**	**Watchdog Timer Reset Flag** The RSTS_WDT flag is set by the "Reset Signal" from the watchdog timer or window watchdog timer to indicate the previous reset source. 0 = No reset from watchdog timer or window watchdog timer. 1 = The watchdog timer or window watchdog timer had issued the reset signal to reset the system. **Note1:** This bit can be cleared by software writing '1'. **Note2:** Watchdog Timer register WTRF (WTCR[2]) bit is set if the system has been reset by WDT time-out reset. Window Watchdog Timer register WWDTRF (WWDTSR[1]) bit is set if the system has been reset by WWDT time-out reset.
[1]	**RSTS_RESET**	**Reset Pin Reset Flag** The RSTS_RESET flag is set by the "Reset Signal" from the nRESET Pin to indicate the previous reset source. 0 = No reset from nRESET pin. 1 = The Pin nRESET had issued the reset signal to reset the system. **Note:** This bit can be cleared by software writing '1'.
[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 (IPRSTC1[0]). 1 = Power-on Reset (POR) or CHIP_RST (IPRSTC1[0]) had issued the reset signal to reset the system. **Note:** This bit can be cleared by software writing '1'.

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					**PDMA_RST**	**CPU_RST**	**CHIP_RST**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMA_RST**	**PDMA Controller Reset (Write Protect)** Setting this bit to 1 will generate a reset signal to the PDMA. User need to set this bit to 0 to release from reset state. 0 = PDMA controller normal operation. 1 = PDMA controller reset. **Note:** This bit is write protected. Refer to the REGWRPROT register.
[1]	**CPU_RST**	**CPU Kernel One-shot Reset (Write Protect)** Setting this bit will only reset the CPU kernel and Flash Memory Controller(FMC), and this bit will automatically return to 0 after the 2 clock cycles. 0 = CPU normal operation. 1 = CPU one-shot reset. **Note:** This bit is write protected. Refer to the REGWRPROT register.
[0]	**CHIP_RST**	**CHIP One-shot Reset (Write Protect)** Setting this bit will reset the whole chip, including CPU kernel and all peripherals, and this bit will automatically return to 0 after the 2 clock cycles. The CHIP_RST is same as the POR reset, all the chip controllers is reset and the chip setting from flash are also reload. For the difference between CHIP_RST and SYSRESETREQ (AIRCR[2]), please refer to section 6.2.2. 0 = CHIP normal operation. 1 = CHIP one-shot reset. **Note:** This bit is write protected. Refer to the REGWRPROT register.

Peripheral Reset Control Register2 (IPRSTC2)

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

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

31	30	29	28	27	26	25	24
Reserved		I2S_RST	ADC_RST	USBD_RST	Reserved		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PS2_RST	Reserved		PWM03_RST	Reserved		UART1_RST	UART0_RST
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	SPI2_RST	SPI1_RST	SPI0_RST	Reserved		I2C1_RST	I2C0_RST
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		TMR3_RST	TMR2_RST	TMR1_RST	TMR0_RST	GPIO_RST	Reserved

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**I2S_RST**	**I^2S Controller Reset** 0 = I^2S controller normal operation. 1 = I^2S controller reset.
[28]	**ADC_RST**	**ADC Controller Reset** 0 = ADC controller normal operation. 1 = ADC controller reset.
[27]	**USBD_RST**	**USB Device Controller Reset** 0 = USB device controller normal operation. 1 = USB device controller reset.
[26:24]	**Reserved**	Reserved.
[23]	**PS2_RST**	**PS/2 Controller Reset** 0 = PS/2 controller normal operation. 1 = PS/2 controller reset.
[22:21]	**Reserved**	Reserved.
[20]	**PWM03_RST**	**PWM03 Controller Reset** 0 = PWM03 controller normal operation. 1 = PWM03 controller reset.
[19:18]	**Reserved**	Reserved.
[17]	**UART1_RST**	**UART1 Controller Reset** 0 = UART1 controller normal operation. 1 = UART1 controller reset.

[16]	**UART0_RST**	**UART0 Controller Reset** 0 = UART0 controller normal operation. 1 = UART0 controller reset.
[15]	**Reserved**	Reserved.
[14]	**SPI2_RST**	**SPI2 Controller Reset** 0 = SPI2 controller normal operation. 1 = SPI2 controller reset.
[13]	**SPI1_RST**	**SPI1 Controller Reset** 0 = SPI1 controller normal operation. 1 = SPI1 controller reset.
[12]	**SPI0_RST**	**SPI0 Controller Reset** 0 = SPI0 controller normal operation. 1 = SPI0 controller reset.
[11:10]	**Reserved**	Reserved.
[9]	**I2C1_RST**	**I^2C1 Controller Reset** 0 = I^2C1 controller normal operation. 1 = I^2C1 controller reset.
[8]	**I2C0_RST**	**I^2C0 Controller Reset** 0 = I^2C0 controller normal operation. 1 = I^2C0 controller reset.
[7:6]	**Reserved**	Reserved.
[5]	**TMR3_RST**	**Timer3 Controller Reset** 0 = Timer3 controller normal operation. 1 = Timer3 controller reset.
[4]	**TMR2_RST**	**Timer2 Controller Reset** 0 = Timer2 controller normal operation. 1 = Timer2 controller reset.
[3]	**TMR1_RST**	**Timer1 Controller Reset** 0 = Timer1 controller normal operation. 1 = Timer1 controller reset.
[2]	**TMR0_RST**	**Timer0 Controller Reset** 0 = Timer0 controller normal operation. 1 = Timer0 controller reset.
[1]	**GPIO_RST**	**GPIO Controller Reset** 0 = GPIO controller normal operation. 1 = GPIO controller reset.
[0]	**Reserved**	Reserved.

Brown-out Detector Control Register (BODCR)

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

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**LVR_EN**	**Low Voltage Reset Enable Bit (Write Protect)** The LVR function reset the chip when the input power voltage is lower than LVR circuit setting. LVR function is enabled by default. 0 = Low Voltage Reset function Disabled. 1 = Low Voltage Reset function Enabled– After enabling the bit, the LVR function will be active with 100us delay for LVR output stable (Default). **Note:** This bit is write protected. Refer to the REGWRPROT register.
[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. The BOD consumes about 100 uA in Normal mode, the low power mode can reduce the current to about 1/10 but slow the BOD response. **Note:** This bit is write protected. Refer to the REGWRPROT register.

[4]	**BOD_INTF**	**Brown-out Detector Interrupt Flag** 0 = Brown-out Detector does not detect any voltage draft at V_{DD} down through or up through the voltage of BOD_VL setting. 1 = When Brown-out Detector detects the V_{DD} is dropped down through the voltage of BOD_VL setting or the V_{DD} is raised up through the voltage of BOD_VL setting, this bit is set to 1 and the Brown-out interrupt is requested if Brown-out interrupt is enabled. This bit can be cleared to 0 by software writing '1'.
[3]	**BOD_RSTEN**	**Brown-out Reset Enable Bit (Write Protect)** 0 = Brown-out "INTERRUPT" function Enabled. While the BOD function is enabled (BOD_EN high) and BOD interrupt function is enabled (BOD_RSTEN low), BOD will assert an interrupt if BOD_OUT is high. BOD interrupt will keep till to the BOD_EN set to 0. BOD interrupt can be blocked by disabling the NVIC BOD interrupt or disabling BOD function (set BOD_EN low). 1 = Brown-out "RESET" function Enabled. While the Brown-out Detector function is enabled (BOD_EN high) and BOD reset function is enabled (BOD_RSTEN high), BOD will assert a signal to reset chip when the detected voltage is lower than the threshold (BOD_OUT high). The default value is set by flash controller user configuration register CBORST (Config0[20]) bit. **Note:** This bit is write protected. Refer to the REGWRPROT register.
[2:1]	**BOD_VL**	**Brown-out Detector Threshold Voltage Selection (Write Protect)** The default value is set by flash memory controller user configuration register CBOV (Config0[22:21]) bits. For NUC123xxxAN 00 = Brown-out voltage is 2.2V. 01 = Brown-out voltage is 2.7V. 10 = Brown-out voltage is 3.8V. 11 = Brown-out voltage is 4.5V. For NUC123xxxAE 00 = Brown-out voltage is 2.2V. 01 = Brown-out voltage is 2.7V. 10 = Brown-out voltage is 3.7V. 11 = Brown-out voltage is 4.4V.**Note:** This bit is write protected. Refer to the REGWRPROT register.
[0]	**BOD_EN**	**Brown-out Detector Enable Bit (Write Protect)** The default value is set by flash controller user configuration register CBODEN (Config0[23]) bit. 0 = Brown-out Detector function Disabled. 1 = Brown-out Detector function Enabled. **Note:** This bit is write protected. Refer to the REGWRPROT register.

Power-on-Reset Control Register (PORCR)

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

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

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

GPIOA Multiple Function and Input Type Control Register (GPA_MFP)

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

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

Bits	Description	
[31:16]	**GPA_TYPEn**	**Schmitt Trigger Function Selection** 0 = GPIOA[15:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOA[15:0] I/O input Schmitt Trigger function Enabled. GPA[9:0] are reserved in this chip.
[15]	**GPA_MFP15**	**PA.15 Pin Function Selection** Bits PA15_MFP1 (ALT_MFP[9]) and GPA_MFP[15] determine the PA.15 function. The PA15_MFP1 (ALT_MFP[9]), GPA_MFP[15]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = PWM3 function is selected. (1, 0) = CLKO (Clock Driver output) function is selected. (1, 1) = I2S_MCLK function is selected.
[14]	**GPA_MFP14**	**PA.14 Pin Function Selection** Bit GPA_MFP[14] determines the PA.14 function. 0 = GPIO function is selected. 1 = PWM2 function is selected.
[13]	**GPA_MFP13**	**PA.13 Pin Function Selection** Bit GPA_MFP[13] determines the PA.13 function. 0 = GPIO function is selected. 1 = PWM1 function is selected.
[12]	**GPA_MFP12**	**PA.12 Pin Function Selection** Bit GPA_MFP[12] determines the PA.12 function. 0 = GPIO function is selected. 1 = PWM0 function is selected.

[11]	**GPA_MFP11**	**PA.11 Pin Function Selection** Bits PA11_MFP1 (ALT_MFP[11]) and GPA_MFP[11] determine the PA.11 function. The PA11_MFP1 (ALT_MFP[11]), GPA_MFP[11]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = I2C1_SCL function is selected. (1, 0) = SPI1_CLK function is selected. (1, 1) = SPI2_MOSI0 function is selected.
[10]	**GPA_MFP10**	**PA.10 Pin Function Selection** Bits PA10_MFP1 (ALT_MFP[12]) and GPA_MFP[10] determine the PA.10 function. The PA10_MFP1 (ALT_MFP[12]), GPA_MFP[10]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = I2C1_SDA function is selected. (1, 0) = SPI1_MISO0 function is selected. (1, 1) = SPI2_MISO0 function is selected.
[9:0]	**Reserved**	Reserved.

GPIOB Multiple Function and Input Type Control Register (GPB_MFP)

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

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

Bits	Description	
[31:16]	**GPB_TYPEn**	**Schmitt Trigger Function Selection** 0 = GPIOB[15:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOB[15:0] I/O input Schmitt Trigger function Enabled.
[15]	**GPB_MFP15**	**PB.15 Pin Function Selection** Bits PB15_MFP1 (ALT_MFP[24]) and GPB_MFP[15] determine the PB.15 function. The PB15_MFP1 (ALT_MFP[24]), GPB_MFP[15]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = INT1 function is selected. (1, 0) = Reserved. (1, 1) = TM0_EXT function is selected.
[14]	**GPB_MFP14**	**PB.14 Pin Function Selection** Bit GPB_MFP[14] determines PB.14 function. 0 = GPIO function is selected. 1 = INT0 function is selected.
[13]	**GPB_MFP13**	**PB.13 Pin Function Selection** Bit GPB_MFP[13] determines the PB.13 function. 0 = GPIO function is selected.
[12]	**GPB_MFP12**	**PB.12 Pin Function Selection** Bits PB12_MFP1 (ALT_MFP[10]) and GPB_MFP[12] determine the PB.12 function. The PB12_MFP1 (ALT_MFP[10]), GPB_MFP[12]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI1_SS0 function is selected. (1, 0) = Reserved. (1, 1) = CLKO(Clock Driver output) function is selected.
[11]	**Reserved**	Reserved.

[10]	**GPB_MFP10**	**PB.10 Pin Function Selection** Bits PB10_MFP1 (ALT_MFP[0]) and GPB_MFP[10] determine the PB.10 function. The PB10_MFP1 (ALT_MFP[0]), GPB_MFP[10]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = TM2 function is selected. (1, 0) = Reserved. (1, 1) = SPI0_SS1 function is selected.
[9]	**GPB_MFP9**	**PB.9 Pin Function Selection** Bits PB9_MFP1 (ALT_MFP[1]) and GPB_MFP[9] determine the PB.9 function. The PB9_MFP1 (ALT_MFP[1]), GPB_MFP[9]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = TM1 function is selected. (1, 0) = PWM1 function is selected. (NUC123xxxAEx Only) (1, 1) = SPI1_SS1 function is selected.
[8]	**GPB_MFP8**	**PB.8 Pin Function Selection** Bit GPB_MFP[8] determines the PB.8 function. 0 = GPIO function is selected. 1 = TM0 function is selected.
[7]	**GPB_MFP7**	**PB.7 Pin Function Selection** Bits PB7_MFP1 (ALT_MFP[16]) and GPB_MFP[7] determine the PB.7 function. The PB7_MFP1 (ALT_MFP[16]), GPB_MFP[7]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART1_nCTS function is selected. (1, 0) = Reserved. (1, 1) = SPI2_MISO0 function is selected.
[6]	**GPB_MFP6**	**PB.6 Pin Function Selection** Bits PB6_MFP1 (ALT_MFP[17]) and GPB_MFP[6] determine the PB.6 function. The PB6_MFP1 (ALT_MFP[17]), GPB_MFP[6]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART1_nRTS function is selected. (1, 0) = Reserved. (1, 1) = SPI2_MOSI0 function is selected.
[5]	**GPB_MFP5**	**PB 5 Pin Function Selection** Bits PB5_MFP1(ALT_MFP[18]) and GPB_MFP[5] determine the PB.5 function. The PB5_MFP1 (ALT_MFP[18]), GPB_MFP[5]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART1_TXD function is selected. (1, 0) = Reserved. (1, 1) = SPI2_CLK function is selected.

[4]	**GPB_MFP4**	**PB.4 Pin Function Selection** Bits PB4_MFP1(ALT_MFP[15]) and GPB_MFP[4] determine the PB.4 function. The PB4_MFP1 (ALT_MFP[15]), GPB_MFP[4]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART1_RXD function is selected. (1, 0) = SPI2_SS0 function is selected. (1, 1) = SPI1_SS1 function is selected.
[3]	**GPB_MFP3**	**PB.3 Pin Function Selection** Bits PB3_MFP1(ALT_MFP[27]) and GPB_MFP[3] determine the PB.3 function. The PB3_MFP1 (ALT_MFP[27]), GPB_MFP[3]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART0_nCTS function is selected. (1, 0) = Reserved. (1, 1) = TM3_EXT function is selected.
[2]	**GPB_MFP2**	**PB.2 Pin Function Selection** Bits PB2_MFP1(ALT_MFP[26]) and GPB_MFP[2] determine the PB.2 function. The PB2_MFP1 (ALT_MFP[26]), GPB_MFP[2]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART0_nRTS function is selected. (1, 0) = Reserved. (1, 1) = TM2_EXT function is selected.
[1]	**GPB_MFP1**	**PB.1 Pin Function Selection** Bit GPB_MFP[1] determines the PB.1 function. 0 = GPIO function is selected. 1 = UART0_TXD function is selected.
[0]	**GPB_MFP0**	**PB.0 Pin Function Selection** Bit GPB_MFP[0] determines the PB.0 function. 0 = GPIO function is selected. 1 = UART0_RXD function is selected.

GPIOC Multiple Function and Input Type Control Register (GPC_MFP)

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

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

Bits	Description	
[31:16]	**GPC_TYPEn**	**Schmitt Trigger Function Selection** 0 = GPIOC[15:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOC[15:0] I/O input Schmitt Trigger function Enabled.
[15:14]	**Reserved**	Reserved.
[13]	**GPC_MFP13**	**PC.13 Pin Function Selection** Bits PC13_MFP1 (ALT_MFP[21]) and GPC_MFP[13] determine the PC.13 function. The PC13_MFP1 (ALT_MFP[21]), GPC_MFP[13]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI1_MOSI1 function is selected. (1, 0) = CLKO (Clock Driver output) function is selected. (1, 1) = PWM3 function is selected.
[12]	**GPC_MFP12**	**PC.12 Pin Function Selection** Bits PC12_MFP1 (ALT_MFP[20]) and GPC_MFP[12] determine the PC.12 function. The PC12_MFP1 (ALT_MFP[20]), GPC_MFP[12]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI1_MISO1 function is selected. (1, 0) = I2S_MCLK function is selected. (1, 1) = PWM2 function is selected.
[11]	**GPC_MFP11**	**PC.11 Pin Function Selection** Bit GPC_MFP[11] determines the PC.11 function. 0 = GPIO function is selected. 1 = SPI1_MOSI0 function is selected.

[10]	**GPC_MFP10**	**PC.10 Pin Function Selection** Bit GPC_MFP[10] determines the PC.10 function. 0 = GPIO function is selected. 1 = SPI1_MISO0 function is selected.
[9]	**GPC_MFP9**	**PC.9 Pin Function Selection** Bit GPC_MFP[9] determines the PC.9 function. 0 = GPIO function is selected. 1 = SPI1_CLK function is selected.
[8]	**GPC_MFP8**	**PC.8 Pin Function Selection** Bits PC8_MFP1 (ALT_MFP1[23]) and GPC_MFP[8] determine the PC.8 function. The PC8_MFP1 (ALT_MFP1[23]), GPC_MFP[8]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI1_SS0 function is selected. (1, 1) = PWM0 function is selected. (NUC123xxxAEx Only)
[7:6]	**Reserved**	Reserved.
[5]	**GPC_MFP5**	**PC.5 Pin Function Selection** Bits PC5_MFP1 (ALT_MFP[30]) and GPC_MFP[5] determine the PC.5 function. The PC5_MFP1 (ALT_MFP[30]), GPC_MFP[5]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MOSI1 function is selected. (1, 0) = Reserved. (1, 1) = UART0_TXD function is selected.
[4]	**GPC_MFP4**	**PC.4 Pin Function Selection** Bits PC4_MFP1 (ALT_MFP[29]) and GPC_MFP[4] determine the PC.4 function. The PC4_MFP1 (ALT_MFP[29]), GPC_MFP[4]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MISO1 function is selected. (1, 0) = Reserved. (1, 1) = UART0_RXD function is selected.
[3]	**GPC_MFP3**	**PC.3 Pin Function Selection** Bits PC3_MFP1 (ALT_MFP[8]) and GPC_MFP[3] determine the PC.3 function. The PC3_MFP1 (ALT_MFP[8]), GPC_MFP[3]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MOSI0 function is selected. (1, 0) = Reserved. (1, 1) = I2S_DO function is selected.
[2]	**GPC_MFP2**	**PC.2 Pin Function Selection** Bits PC2_MFP1 (ALT_MFP[7]) and GPC_MFP[2] determine the PC.2 function. The PC2_MFP1 (ALT_MFP[7]), GPC_MFP[2]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MISO0 function is selected. (1, 0) = Reserved. (1, 1) = I2S_DI function is selected.

[1]	**GPC_MFP1**	**PC.1 Pin Function Selection** Bits PC1_MFP1 (ALT_MFP[6]) and GPC_MFP[1] determine the PC.1 function. The PC1_MFP1 (ALT_MFP[6]), GPC_MFP[1]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_CLK function is selected. (1, 0) = Reserved. (1, 1) = I2S_BCLK function is selected.
[0]	**GPC_MFP0**	**PC.0 Pin Function Selection** Bits PC0_MFP1 (ALT_MFP[5]) and GPC_MFP[0] determine the PC.0 function. The PC0_MFP1 (ALT_MFP[5]), GPC_MFP[0]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_SS0 function is selected. (1, 0) = Reserved. (1, 1) = I2S_LRCLK function is selected.

GPIOD Multiple Function and Input Type Control Register (GPD_MFP)

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

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

Bits	Description	
[31:16]	**GPD_TYPEn**	**Schmitt Trigger Function Selection** 0 = GPIOD[15:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOD[15:0] I/O input Schmitt Trigger function Enabled.
[15:12]	**Reserved**	Reserved.
[11]	**GPD_MFP11**	**PD.11 Pin Function Selection** Bit GPD_MFP[11] determines the PD.11 function. 0 = GPIO function is selected. 1 = INT1 function is selected.
[10]	**GPD_MFP10**	**PD.10 Pin Function Selection** Bit GPD_MFP[10] determines the PD.10 function. 0 = GPIO function is selected. 1 = CLKO function is selected.
[9]	**GPD_MFP9**	**PD.9 Pin Function Selection** Bit GPD_MFP[9] determines the PD.9 function. 0 = GPIO function is selected.1 = Reserved.
[8]	**GPD_MFP8**	**PD.8 Pin Function Selection** Bit GPD_MFP[8] determines the PD.8 function. 0 = GPIO function is selected. 1 = SPI1_MOSI0 function is selected.
[7:6]	**Reserved**	Reserved.

[5]	**GPD_MFP5**	**PD.5 Pin Function Selection** Bits PD5_MFP1 (ALT_MFP1[21]) and GPD_MFP[5] determine the PD.5 function. The PD5_MFP1 (ALT_MFP1[21]), GPD_MFP[5]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = Reserved. (1, 0) = SPI2_MOSI1 function is selected. (1, 1) = ADC5 function is selected.
[4]	**GPD_MFP4**	**PD.4 Pin Function Selection** Bits PD4_MFP1 (ALT_MFP1[20]) and GPD_MFP[4] determine the PD.4 function. The PD4_MFP1 (ALT_MFP1[20]), GPD_MFP[4]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = Reserved. (1, 0) = SPI2_MISO1 function is selected. (1, 1) = ADC4 function is selected.
[3]	**GPD_MFP3**	**PD.3 Pin Function Selection** Bits PD3_MFP1 (ALT_MFP1[19]) and GPD_MFP[3] determine the PD.3 function. The PD3_MFP1 (ALT_MFP1[19]), GPD_MFP[3]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MOSI1 function is selected. (1, 0) = SPI2_MOSI0 function is selected. (1, 1) = ADC3 function is selected.
[2]	**GPD_MFP2**	**PD.2 Pin Function Selection** Bits PD2_MFP1 (ALT_MFP1[18]) and GPD_MFP[2] determine the PD.2 function. The PD2_MFP1 (ALT_MFP1[18]), GPD_MFP[2]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MISO1 function is selected. (1, 0) = SPI2_MISO0 function is selected. (1, 1) = ADC2 function is selected.
[1]	**GPD_MFP1**	**PD.1 Pin Function Selection** Bits PD1_MFP1 (ALT_MFP1[17]) and GPD_MFP[1] determine the PD.1 function. The PD1_MFP1 (ALT_MFP1[17]), GPD_MFP[1]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_SS1 function is selected. (1, 0) = SPI2_CLK function is selected. (1, 1) = ADC1 function is selected.
[0]	**GPD_MFP0**	**PD.0 Pin Function Selection** Bits PD0_MFP1 (ALT_MFP1[16]) and GPD_MFP[0] determine the PD.0 function. The PD0_MFP1 (ALT_MFP1[16]), GPD_MFP[0]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = Reserved. (1, 0) = SPI2_SS0 function is selected. (1, 1) = ADC0 function is selected.

GPIOF Multiple Function and Input Type Control Register (GPF_MFP)

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**GPF_TYPE**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**GPF_MFP3**	**GPF_MFP2**	**GPF_MFP1**	**GPF_MFP0**

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**GPF_TYPE**	**Schmitt Trigger Function Selection** 0 = GPIOF[3:0] I/O input Schmitt Trigger function Disabled. 1 = GPIOF[3:0] I/O input Schmitt Trigger function Enabled.
[15:4]	**Reserved**	Reserved.
[3]	**GPF_MFP3**	**PF.3 Pin Function Selection** Bits PF3_MFP1 (ALT_MFP1[27:26]) and GPF_MFP[3] determine the PF.3 function. The PF3_MFP1 (ALT_MFP1[27:26]), GPD_MFP[3]) value and function mapping are as following list. (00, 0) = GPIO function is selected. (00, 1) = PS2_CLK function is selected. (10, 1) = I2C0_SCL function is selected. (11, 1) = ADC7 function is selected. The reset value of this bit is 1.
[2]	**GPF_MFP2**	**PF.2 Pin Function Selection** Bits PF2_MFP1 (ALT_MFP1[25:24]) and GPF_MFP[2] determine the PF.2 function. The PF2_MFP1 (ALT_MFP1[25:24]), GPF_MFP[2]) value and function mapping are as following list. (00, 0) = GPIO function is selected. (00, 1) = PS2_DAT function is selected. (10, 1) = I2C0_SDA function is selected. (11, 1) = ADC6 function is selected. The reset value of this bit is 1.

[1]	**GPF_MFP1**	**PF.1 Pin Function Selection (Read Only)** Bit GPF_MFP[1] determines the PF.1 function. 0 = GPIO function is selected. 1 = XT1_IN function is selected. **Note1:** For NUC123xxxANx, the value of this bit is controlled by CGPFMFP (Config0[27]) when power-up, user can write this bit after chip power-up. **Note2:** For NUC123xxxAEx, the value of this bit is controlled by CGPFMFP (Config0[27]).
[0]	**GPF_MFP0**	**PF.0 Pin Function Selection (Read Only)** Bit GPF_MFP[0] determines the PF.0 function. 0 = GPIO function is selected. 1 = XT1_OUT function is selected. **Note1:** For NUC123xxxANx, the value of this bit is controlled by CGPFMFP (Config0[27]) when power-up, user can write this bit after chip power-up. **Note2:** For NUC123xxxAEx, the value of this bit is controlled by CGPFMFP (Config0[27]).

Alternative Multiple Function Pin Control Register (ALT_MFP)

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

31	30	29	28	27	26	25	24
Reserved	**PC5_MFP1**	**PC4_MFP1**	**Reserved**	**PB3_MFP1**	**PB2_MFP1**	**Reserved**	**PB15_MFP1**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**PC13_MFP1**	**PC12_MFP1**	**Reserved**	**PB5_MFP1**	**PB6_MFP1**	**PB7_MFP1**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PB4_MFP1	**Reserved**		**PA10_MFP1**	**PA11_MFP1**	**PB12_MFP1**	**PA15_MFP1**	**PC3_MFP1**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PC2_MFP1	**PC1_MFP1**	**PC0_MFP1**	**Reserved**			**PB9_MFP1**	**PB10_MFP1**

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**PC5_MFP1**	**PC.5 Pin Alternate Function Selection** Bits PC5_MFP1 (ALT_MFP[30]) and GPC_MFP[5] determine the PC.5 function. The PC5_MFP1 (ALT_MFP[30]), GPC_MFP[5]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MOSI1 function is selected. (1, 0) = Reserved. (1, 1) = UART0_TXD function is selected.
[29]	**PC4_MFP1**	**PC.4 Pin Alternate Function Selection** Bits PC4_MFP1 (ALT_MFP[29]) and GPC_MFP[4] determine the PC.4 function. The PC4_MFP1 (ALT_MFP1[29]), GPC_MFP[4]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MISO1 function is selected. (1, 0) = Reserved. (1, 1) = UART0_RXD function is selected.
[28]	**Reserved**	Reserved.
[27]	**PB3_MFP1**	**PB.3 Pin Alternate Function Selection** Bits PB3_MFP1 (ALT_MFP[27]) and GPB_MFP[3] determine the PB.3 function. The PB3_MFP1 (ALT_MFP[27]), GPB_MFP[3]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART0_nCTS function is selected. (1, 0) = Reserved. (1, 1) = TM3_EXT function is selected.

[26]	**PB2_MFP1**	**PB.2 Pin Alternate Function Selection** Bits PB2_MFP1 (ALT_MFP[26]) and GPB_MFP[2] determine the PB.2 function. The PB2_MFP1 (ALT_MFP[26]), GPB_MFP[2]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART0_nRTS function is selected. (1, 0) = Reserved. (1, 1) = TM2_EXT function is selected.
[25]	**Reserved**	Reserved.
[24]	**PB15_MFP1**	**PB.15 Pin Alternate Function Selection** Bits PB15_MFP1 (ALT_MFP[24]) and GPB_MFP[15] determine the PB.15 function. The PB15_MFP1 (ALT_MFP[24]), GPB_MFP[15]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = INT1 function is selected. (1, 0) = Reserved. (1, 1) = TM0_EXT function is selected.
[23:22]	**Reserved**	Reserved.
[21]	**PC13_MFP1**	**PC.13 Pin Alternate Function Selection** Bits PC13_MFP1 (ALT_MFP[21]) and GPC_MFP[13] determine the PC.13 function. The PC13_MFP1 (ALT_MFP[21]), GPC_MFP[13]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI1_MOSI1 function is selected. (1, 0) = CLKO(Clock Driver output) function is selected. (1, 1) = PWM3 function is selected.
[20]	**PC12_MFP1**	**PC.12 Pin Alternate Function Selection** Bits PC12_MFP1 (ALT_MFP[20]) and GPC_MFP[12] determine the PC.12 function. The PC12_MFP1 (ALT_MFP[20]), GPC_MFP[12]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI1_MISO1 function is selected. (1, 0) = I2S_MCLK function is selected. (1, 1) = PWM2 function is selected.
[19]	**Reserved**	Reserved.
[18]	**PB5_MFP1**	**PB 5 Pin Alternate Function Selection** Bits PB5_MFP1 (ALT_MFP[18]) and GPB_MFP[5] determine the PB.5 function. The PB5_MFP1 (ALT_MFP[18]), GPB_MFP[5]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART1_TXD function is selected. (1, 0) = Reserved. (1, 1) = SPI2_CLK function is selected.

[17]	**PB6_MFP1**	**PB.6 Pin Alternate Function Selection** Bits PB6_MFP1 (ALT_MFP[17]) and GPB_MFP[6] determine the PB.6 function. The PB6_MFP1 (ALT_MFP[17]), GPB_MFP[6]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART1_nRTS function is selected. (1, 0) = Reserved. (1, 1) = SPI2_MOSI0 function is selected.
[16]	**PB7_MFP1**	**PB.7 Pin Alternate Function Selection** Bits PB7_MFP1 (ALT_MFP[16]) and GPB_MFP[7] determine the PB.7 function. The PB7_MFP1 (ALT_MFP[16]), GPB_MFP[7]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART_nCTS function is selected. (1, 0) = Reserved. (1, 1) = SPI2_MISO0 function is selected.
[15]	**PB4_MFP1**	**PB.4 Pin Alternate Function Selection** Bits PB4_MFP1 (ALT_MFP[15]) and GPB_MFP[4] determine the PB.4 function. The PB4_MFP1 (ALT_MFP[15]), GPB_MFP[4]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = UART1_RXD function is selected. (1, 0) = SPI2_SS0 function is selected. (1, 1) = SPI1_SS1 function is selected.
[14:13]	**Reserved**	Reserved.
[12]	**PA10_MFP1**	**PA.10 Pin Alternate Function Selection** Bits PA10_MFP1 (ALT_MFP[12]) and GPA_MFP[10] determine the PA.10 function. The PA10_MFP1 (ALT_MFP[12]), GPA_MFP[10]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = I2C1_SDA function is selected. (1, 0) = SPI1_MISO0 function is selected. (1, 1) = SPI2_MISO0 function is selected.
[11]	**PA11_MFP1**	**PA.11 Pin Alternate Function Selection** Bits PA11_MFP1 (ALT_MFP[11]) and GPA_MFP[11] determine the PA.11 function. The PA11_MFP1 (ALT_MFP[11]), GPA_MFP[11]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = I2C1_SCL function is selected. (1, 0) = SPI1_CLK function is selected. (1, 1) = SPI2_MOSI0 function is selected.

[10]	**PB12_MFP1**	**PB.12 Pin Alternate Function Selection** Bits PB12_MFP1 (ALT_MFP[10]) and GPB_MFP[12] determine the PB.12 function. The PB12_MFP1 (ALT_MFP[10]), GPB_MFP[12]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI1_SS0 function is selected. (1, 0) = Reserved. (1, 1) = CLKO (Clock Driver output) function is selected.
[9]	**PA15_MFP1**	**PA.15 Pin Alternate Function Selection** Bits PA15_MFP1 (ALT_MFP[9]) and GPA_MFP[15] determine the PA.15 function. The PA15_MFP1 (ALT_MFP[9]), GPA_MFP[15])value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = PWM3 function is selected. (1, 0) = CLKO (Clock Driver output) function is selected. (1, 1) = I2S_MCLK function is selected.
[8]	**PC3_MFP1**	**PC.3 Pin Alternate Function Selection** Bits PC3_MFP1 (ALT_MFP[8]) and GPC_MFP[3] determine the PC.3 function. The PC3_MFP1 (ALT_MFP[8]), GPC_MFP[3]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MOSI0 function is selected. (1, 0) = Reserved. (1, 1) = I2S_DO function is selected.
[7]	**PC2_MFP1**	**PC.2 Pin Alternate Function Selection** Bits PC2_MFP1 (ALT_MFP[7]) and GPC_MFP[2] and determine the PC.2 function. The PC2_MFP1 (ALT_MFP[7]), GPC_MFP[2]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MISO0 function is selected. (1, 0) = Reserved. (1, 1) = I2S_DI function is selected.
[6]	**PC1_MFP1**	**PC.1 Pin Alternate Function Selection** Bits PC1_MFP1 (ALT_MFP[6]) and GPC_MFP[1] determine the PC.1 function. The PC1_MFP1 (ALT_MFP[6]), GPC_MFP[1]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_CLK function is selected. (1, 0) = Reserved. (1, 1) = I2S_BCLK function is selected.
[5]	**PC0_MFP1**	**PC.0 Pin Alternate Function Selection** Bits PC0_MFP1 (ALT_MFP[5]) and GPC_MFP[0] determine the PC.0 function. The PC0_MFP1 (ALT_MFP[5]), GPC_MFP[0]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_SS0 function is selected. (1, 0) = Reserved. (1, 1) = I2S_LRCLK function is selected.

[4:2]	**Reserved**	Reserved.
[1]	**PB9_MFP1**	**PB.9 Pin Alternate Function Selection** Bits PB9_MFP1 (ALT_MFP[1]) and GPB_MFP[9] determine the PB.9 function. The PB9_MFP1 (ALT_MFP[1]), GPB_MFP[9]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = TM1 function is selected. (1, 0) = PWM1 function is selected. (NUC123xxxAEx Only) (1, 1) = SPI1_SS1 function is selected.
[0]	**PB10_MFP1**	**PB.10 Pin Alternate Function Selection** Bits PB10_MFP1 (ALT_MFP[0]) and GPB_MFP[10] determine the PB.10 function. The PB10_MFP1 (ALT_MFP[0]), GPB_MFP[10]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = TM2 function is selected. (1, 0) = Reserved. (1, 1) = SPI0_SS1 function is selected.

Alternative Multiple Function Pin Control Register 1 (ALT_MFP1)

Register	Offset	R/W	Description	Reset Value
ALT_MFP1	GCR_BA+0x54	R/W	Alternative Multiple Function Pin Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**PF3_MFP1**		**PF2_MFP1**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PC8_MFP1	**Reserved**	**PD5_MFP1**	**PD4_MFP1**	**PD3_MFP1**	**PD2_MFP1**	**PD1_MFP1**	**PD0_MFP1**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:26]	**PF3_MFP1**	**PF.3 Pin Alternate Function Selection** Bits PF3_MFP1(ALT_MFP1[27:26]) and GPF_MFP[3] determine the PF.3 function. The PF3_MFP1 (ALT_MFP1[27:26]), GPD_MFP[3]) value and function mapping are as following list. (00, 0) = GPIO function is selected. (00, 1) = PS2_CLK function is selected. (10, 1) = I2C0_SCL function is selected. (11, 1) = ADC7 function is selected.
[25:24]	**PF2_MFP1**	**PF.2 Pin Alternate Function Selection** Bits PF2_MFP1 (ALT_MFP1[25:24]) and GPF_MFP[2] determine the PF.2 function. The PF2_MFP1 (ALT_MFP1[25:24]), GPF_MFP[2]) value and function mapping are as following list. (00, 0) = GPIO function is selected. (00, 1) = PS2_DAT function is selected. (10, 1) = I2C0_SDA function is selected. (11, 1) = ADC6 function is selected.
[23]	**PC8_MFP1**	**PC.8 Pin Alternate Function Selection** Bits PC8_MFP1 (ALT_MFP1[23]) and GPC_MFP[8] determine the PC.8 function. The PC8_MFP1 (ALT_MFP1[23]), GPC_MFP[8]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI1_SS0 function is selected. (1, 1) = PWM0 function is selected. (NUC123xxxAEx Only)
[22]	**Reserved**	Reserved.

[21]	**PD5_MFP1**	**PD.5 Pin Alternate Function Selection** Bits PD5_MFP1 (ALT_MFP1[21]) and GPD_MFP[5] determine the PD.5 function. The PD5_MFP1 (ALT_MFP1[21]), GPD_MFP[5]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = Reserved. (1, 0) = SPI2_MOSI1 function is selected. (1, 1) = ADC5 function is selected.
[20]	**PD4_MFP1**	**PD.4 Pin Alternate Function Selection** Bits PD4_MFP1 (ALT_MFP1[20]) and GPD_MFP[4] determine the PD.4 function. The PD4_MFP1 (ALT_MFP1[20]), GPD_MFP[4]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = Reserved. (1, 0) = SPI2_MISO1 function is selected. (1, 1) = ADC4 function is selected.
[19]	**PD3_MFP1**	**PD.3 Pin Alternate Function Selection** Bits PD3_MFP1 (ALT_MFP1[19]) and GPD_MFP[3] determine the PD.3 function. The PD3_MFP1 (ALT_MFP1[19]), GPD_MFP[3]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MOSI1 function is selected. (1, 0) = SPI2_MOSI0 function is selected. (1, 1) = ADC3 function is selected.
[18]	**PD2_MFP1**	**PD.2 Pin Alternate Function Selection** Bits PD2_MFP1 (ALT_MFP1[18]) and GPD_MFP[2] determine the PD.2 function. The PD2_MFP1 (ALT_MFP1[18]), GPD_MFP[2]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_MISO1 function is selected. (1, 0) = SPI2_MISO0 function is selected. (1, 1) = ADC2 function is selected.
[17]	**PD1_MFP1**	**PD.1 Pin Alternate Function Selection** Bits PD1_MFP1 (ALT_MFP1[17]) and GPD_MFP[1] determine the PD.1 function. The PD1_MFP1 (ALT_MFP1[17]), GPD_MFP[1]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = SPI0_SS1 function is selected. (1, 0) = SPI2_CLK function is selected. (1, 1) = ADC1 function is selected.

[16]	**PD0_MFP1**	**PD.0 Pin Alternate Function Selection** Bits PD0_MFP1 (ALT_MFP1[16]) and GPD_MFP[0] determine the PD.0 function. The PD0_MFP1 (ALT_MFP1[16]), GPD_MFP[0]) value and function mapping are as following list. (0, 0) = GPIO function is selected. (0, 1) = Reserved. (1, 0) = SPI2_SS0 function is selected. (1, 1) = ADC0 function is selected.
[15:0]	**Reserved**	Reserved.

GPIOA I/O Control Register (GPA_IOCR)

Register	Offset	R/W	Description	Reset Value
GPA_IOCR	GCR_BA+0xC0	R/W	GPIOA I/O 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				**GPA11_DS**	**GPA10_DS**	**Reserved**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**GPA11_DS**	**PA.11 Pin Driving Strength Selection** 0 = PA.11 strong driving strength mode Disabled. 1 = PA.11 strong driving strength mode Enabled.
[10]	**GPA10_DS**	**PA.10 Pin Driving Strength Selection** 0 = PA.10 strong driving strength mode Disabled. 1 = PA.10 strong driving strength mode Enabled.
[9:0]	**Reserved**	Reserved.

GPIOB I/O Control Register (GPB_IOCR)

Register	Offset	R/W	Description	Reset Value
GPB_IOCR	GCR_BA+0xC4	R/W	GPIOB I/O 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	**GPB14_DS**	**GPB13_DS**	**GPB12_DS**	**Reserved**			**GPB8_DS**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPB7_DS	**GPB6_DS**	**GPB5_DS**	**GPB4_DS**	**Reserved**			

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14]	**GPB14_DS**	**PB.14 Pin Driving Strength Selection** 0 = PB.14 strong driving strength mode Disabled. 1 = PB.14 strong driving strength mode Enabled.
[13]	**GPB13_DS**	**PB.13 Pin Driving Strength Selection** 0 = PB.13 strong driving strength mode Disabled. 1 = PB.13 strong driving strength mode Enabled.
[12]	**GPB12_DS**	**PB.12 Pin Driving Strength Selection** 0 = PB.12 strong driving strength mode Disabled. 1 = PB.12 strong driving strength mode Enabled.
[11:9]	**Reserved**	Reserved.
[8]	**GPB8_DS**	**PB.8 Pin Driving Strength Selection** 0 = PB.8 strong driving strength mode Disabled. 1 = PB.8 strong driving strength mode Enabled.
[7]	**GPB7_DS**	**PB.7 Pin Driving Strength Selection** 0 = PB.7 strong driving strength mode Disabled. 1 = PB.7 strong driving strength mode Enabled.
[6]	**GPB6_DS**	**PB.6 Pin Driving Strength Selection** 0 = PB.6 strong driving strength mode Disabled. 1 = PB.6 strong driving strength mode Enabled.
[5]	**GPB5_DS**	**PB.5 Pin Driving Strength Selection** 0 = PB.5 strong driving strength mode Disabled. 1 = PB.5 strong driving strength mode Enabled.

[4]	**GPB4_DS**	**PB.4 Pin Driving Strength Selection** 0 = PB.4 strong driving strength mode Disabled. 1 = PB.4 strong driving strength mode Enabled.
[3:0]	**Reserved**	Reserved.

GPIOD I/O Control Register (GPD_IOCR)

Register	Offset	R/W	Description	Reset Value
GPD_IOCR	GCR_BA+0xCC	R/W	GPIOD I/O 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				**GPD11_DS**	**GPD10_DS**	**GPD9_DS**	**GPD8_DS**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**GPD11_DS**	**PD.11 Pin Driving Strength Selection** 0 = PD.11 strong driving strength mode Disabled. 1 = PD.11 strong driving strength mode Enabled.
[10]	**GPD10_DS**	**PD.10 Pin Driving Strength Selection** 0 = PD.10 strong driving strength mode Disabled. 1 = PD.10 strong driving strength mode Enabled.
[9]	**GPD9_DS**	**PD.9 Pin Driving Strength Selection** 0 = PD.9 strong driving strength mode Disabled. 1 = PD.9 strong driving strength mode Enabled.
[8]	**GPD8_DS**	**PD.8 Pin Driving Strength Selection** 0 = PD.8 strong driving strength mode Disabled. 1 = PD.8 strong driving strength mode Enabled.
[7:0]	**Reserved**	Reserved.

Register Write-Protection Control Register (REGWRPROT)

Some of the system control registers need to be protected to avoid inadvertent write and disturb the chip operation. These system control registers are protected after the power-on reset till user to disable register protection. For user to program these protected registers, a register protection disable sequence needs to be followed by a special programming. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register REGWRPROT address at 0x5000_0100 continuously. Any different data value, different sequence or any other write to other address during these three data writing will abort the whole sequence.

After the protection is disabled, user can check the protection disable bit at address 0x5000_0100 bit0, 1 is protection disable, and 0 is protection enable. Then user can update the target protected register value and then write any data to the address “0x5000_0100” to enable register protection.

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

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[7:0]	**REGWRPROT**	**Register Write-protection Code (Write Only)** Some registers have write-protection function. Writing these registers has 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. Please refer to 6.2.7 register protection.

GPIOA Multiple Function High Byte Control Register (GPA_MFPH)

Register	Offset	R/W	Description	Reset Value
GPA_MFPH	GCR_BA+0x134	R/W	GPIOA Multiple Function High Byte Control Register (NUC123xxxAEx Only)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**GPA15_MFP**			**Reserved**	**GPA14_MFP**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**GPA13_MFP**			**Reserved**	**GPA12_MFP**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**GPA11_MFP**			**Reserved**	**GPA10_MFP**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**GPA15_MFP**	**PA.15 Pin Function Selection** 000 = The GPIO function is selected. 001 = The PWM3 function is selected. 010 = The CLKO (Clock Driver output) function is selected. 011 = The I2S_MCLK function is selected. Others = Reserved.
[27]	**Reserved**	Reserved.
[26:24]	**GPA14_MFP**	**PA.14 Pin Function Selection** 000 = The GPIO function is selected. 001 = The PWM2 function is selected. Others = Reserved.
[23]	**Reserved**	Reserved.
[22:20]	**GPA13_MFP**	**PA.13 Pin Function Selection** 000 = The GPIO function is selected. 001 = The PWM1 function is selected. Others = Reserved.
[19]	**Reserved**	Reserved.
[18:16]	**GPA12_MFP**	**PA.12 Pin Function Selection** 000 = The GPIO function is selected. 001 = The PWM0 function is selected. Others = Reserved.
[15]	**Reserved**	Reserved.

[14:12]	**GPA11_MFP**	**PA.11 Pin Function Selection** 000 = The GPIO function is selected. 001 = The I2C1_SCL function is selected. 010 = The SPI1_CLK function is selected. 011 = The SPI2_MOSI0 function is selected. Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**GPA10_MFP**	**PA.10 Pin Function Selection** 000 = The GPIO function is selected. 001 = The I2C1_SDA function is selected. 010 = The SPI1_MISO0 function is selected. 011 = The SPI2_MISO0 function is selected. Others = Reserved.
[7:0]	**Reserved**	Reserved.

Note: For NUC123xxxAEx, if GPx_MFPH and GPx_MFPL are used as pin multi-function setting, the GPx_MFP, ALT_MFP and ALT_MFP1 will become invalid.

GPIOB Multiple Function Low Byte Control Register (GPB_MFPL)

Register	Offset	R/W	Description	Reset Value
GPB_MFPL	GCR_BA+0x138	R/W	GPIOB Multiple Function Low Byte Control Register (NUC123xxxAEx Only)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**GPB7_MFP**			**Reserved**	**GPB6_MFP**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**GPB5_MFP**			**Reserved**	**GPB4_MFP**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**GPB3_MFP**			**Reserved**	**GPB2_MFP**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**GPB1_MFP**			**Reserved**	**GPB0_MFP**		

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**GPB7_MFP**	**PB.7 Pin Function Selection** 000 = The GPIO function is selected. 001 = The UART1_nCTS function is selected. 010 = The SPI2_MISO0 function is selected. Others = Reserved.
[27]	**Reserved**	Reserved.
[26:24]	**GPB6_MFP**	**PB.6 Pin Function Selection** 000 = The GPIO function is selected. 001 = The UART1_nRTS function is selected. 010 = The SPI2_MOSI0 function is selected. Others = Reserved.
[23]	**Reserved**	Reserved.
[22:20]	**GPB5_MFP**	**PB 5 Pin Function Selection** 000 = The GPIO function is selected. 001 = The UART1_TXD function is selected. 010 = The SPI2_CLK function is selected. Others = Reserved.
[19]	**Reserved**	Reserved.
[18:16]	**GPB4_MFP**	**PB.4 Pin Function Selection** 000 = The GPIO function is selected. 001 = The UART1_RXD function is selected. 010 = The SPI2_SS0 function is selected. 011 = The SPI1_SS1 function is selected. Others = Reserved.

[15]	**Reserved**	Reserved.
[14:12]	**GPB3_MFP**	**PB.3 Pin Function Selection** 000 = The GPIO function is selected. 001 = The UART0_nCTS function is selected. 010 = The TM3_EXT function is selected. Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**GPB2_MFP**	**PB.2 Pin Function Selection** 000 = The GPIO function is selected. 001 = The UART0_nRTS function is selected. 010 = The TM2_EXT function is selected. Others = Reserved.
[7]	**Reserved**	Reserved.
[6:4]	**GPB1_MFP**	**PB.1 Pin Function Selection** 000 = The GPIO function is selected. 001 = The UART0_TXD function is selected. Others = Reserved.
[3]	**Reserved**	Reserved.
[2:0]	**GPB0_MFP**	**PB.0 Pin Function Selection** 000 = The GPIO function is selected. 001 = The UART0_RXD function is selected. Others = Reserved.

Note: For NUC123xxxAEx, if GPx_MFPH and GPx_MFPL are used as pin multi-function setting, the GPx_MFP, ALT_MFP and ALT_MFP1 will become invalid.

GPIOB Multiple Function High Byte Control Register (GPB_MFPH)

Register	Offset	R/W	Description	Reset Value
GPB_MFPH	GCR_BA+0x13C	R/W	GPIOB Multiple Function High Byte Control Register (NUC123xxxAEx Only)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**GPB15_MFP**			**Reserved**	**GPB14_MFP**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**GPB13_MFP**			**Reserved**	**GPB12_MFP**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**GPB10_MFP**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**GPB9_MFP**			**Reserved**	**GPB8_MFP**		

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**GPB15_MFP**	**PB.15 Pin Function Selection** 000 = The GPIO function is selected. 001 = The INT1 function is selected. 010 = The TM0_EXT function is selected. Others = Reserved.
[27]	**Reserved**	Reserved.
[26:24]	**GPB14_MFP**	**PB.14 Pin Function Selection** 000 = The GPIO function is selected. 001 = The INT0 function is selected. Others = Reserved.
[23]	**Reserved**	Reserved.
[22:20]	**GPB13_MFP**	**PB.13 Pin Function Selection** 000 = The GPIO function is selected. Others = Reserved.
[19]	**Reserved**	Reserved.
[18:16]	**GPB12_MFP**	**PB.12 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI1_SS0 function is selected. 010 = The CLKO (Clock Driver output) function is selected. Others = Reserved.
[15:11]	**Reserved**	Reserved.

[10:8]	**GPB10_MFP**	**PB.10 Pin Function Selection** 000 = The GPIO function is selected. 001 = The TM2 function is selected. 010 = The SPI0_SS1 function is selected. Others = Reserved.
[7]	**Reserved**	Reserved.
[6:4]	**GPB9_MFP**	**PB.9 Pin Function Selection** 000 = The GPIO function is selected. 001 = The TM1 function is selected. 010 = The SPI1_SS1 function is selected. 011 = The PWM1 function is selected. Others = Reserved.
[3]	**Reserved**	Reserved.
[2:0]	**GPB8_MFP**	**PB.8 Pin Function Selection** 000 = The GPIO function is selected. 001 = The TM0 function is selected. Others = Reserved.

Note: For NUC123xxxAEx, if GPx_MFPH and GPx_MFPL is used as pin multi-function setting, the GPx_MFP, ALT_MFP and ALT_MFP1 will become invalid.

GPIOC Multiple Function Low Byte Control Register (GPC_MFPL)

Register	Offset	R/W	Description	Reset Value
GPC_MFPL	GCR_BA+0x140	R/W	GPIOC Multiple Function Low Byte Control Register (NUC123xxxAEx Only)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**GPC5_MFP**			**Reserved**	**GPC4_MFP**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**GPC3_MFP**			**Reserved**	**GPC2_MFP**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**GPC1_MFP**			**Reserved**	**GPC0_MFP**		

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22:20]	**GPC5_MFP**	**PC.5 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI0_MOSI1 function is selected. 010 = The UART0_TXD function is selected. Others = Reserved.
[19]	**Reserved**	Reserved.
[18:16]	**GPC4_MFP**	**PC.4 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI0_MISO1 function is selected. 010 = The UART0_RXD function is selected. Others = Reserved.
[15]	**Reserved**	Reserved.
[14:12]	**GPC3_MFP**	**PC.3 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI0_MOSI0 function is selected. 010 = The I2S_DO function is selected. Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**GPC2_MFP**	**PC.2 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI0_MISO0 function is selected. 010 = The I2S_DI function is selected. Others = Reserved.
[7]	**Reserved**	Reserved.

[6:4]	**GPC1_MFP**	**PC.1 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI0_CLK function is selected. 010 = The I2S_BCLK function is selected. Others = Reserved.
[3]	**Reserved**	Reserved.
[2:0]	**GPC0_MFP**	**PC.0 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI0_SS0 function is selected. 010 = The I2S_LRCLK function is selected. Others = Reserved.

Note: For NUC123xxxAEx, if GPx_MFPH and GPx_MFPL are used as pin multi-function setting, the GPx_MFP, ALT_MFP and ALT_MFP1 will become invalid.

GPIOC Multiple Function High Byte Control Register (GPC_MFPH)

Register	Offset	R/W	Description	Reset Value
GPC_MFPH	GCR_BA+0x144	R/W	GPIOC Multiple Function High Byte Control Register (NUC123xxxAEx Only)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**GPC13_MFP**			**Reserved**	**GPC12_MFP**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**GPC11_MFP**			**Reserved**	**GPC10_MFP**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**GPC9_MFP**			**Reserved**	**GPC8_MFP**		

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22:20]	**GPC13_MFP**	**PC.13 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI1_MOSI1 function is selected. 010 = The CLKO (Clock Driver output) function is selected. 011 = The PWM3 function is selected. Others = Reserved.
[19]	**Reserved**	Reserved.
[18:16]	**GPC12_MFP**	**PC.12 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI1_MISO1 function is selected. 010 = The I2S_MCLK function is selected. 011 = The PWM2 function is selected. Others = Reserved.
[15]	**Reserved**	Reserved.
[14:12]	**GPC11_MFP**	**PC.11 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI1_MOSI0 function is selected. Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**GPC10_MFP**	**PC.10 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI1_MISO0 function is selected. Others = Reserved.
[7]	**Reserved**	Reserved.

[6:4]	**GPC9_MFP**	**PC.9 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI1_CLK function is selected. Others = Reserved.
[3]	**Reserved**	Reserved.
[2:0]	**GPC8_MFP**	**PC.8 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI1_SS0 function is selected. 010 = The PWM0 function is selected. Others = Reserved.

Note: For NUC123xxxAEx, if GPx_MFPH and GPx_MFPL are used as pin multi-function setting, the GPx_MFP, ALT_MFP and ALT_MFP1 will become invalid.

GPIOD Multiple Function Low Byte Control Register (GPD_MFPL)

Register	Offset	R/W	Description	Reset Value
GPD_MFPL	GCR_BA+0x148	R/W	GPIOD Multiple Function Low Byte Control Register (NUC123xxxAEx Only)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**GPD5_MFP**			**Reserved**	**GPD4_MFP**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**GPD3_MFP**			**Reserved**	**GPD2_MFP**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**GPD1_MFP**			**Reserved**	**GPD0_MFP**		

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22:20]	**GPD5_MFP**	**PD.5 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI2_MOSI1 function is selected. 010 = The ADC5 function is selected. Others = Reserved.
[19]	**Reserved**	Reserved.
[18:16]	**GPD4_MFP**	**PD.4 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI2_MISO1 function is selected. 010 = The ADC4 function is selected. Others = Reserved.
[15]	**Reserved**	Reserved.
[14:12]	**GPD3_MFP**	**PD.3 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI0_MOSI1 function is selected. 010 = The SPI2_MOSI0 function is selected. 011 = The ADC3 function is selected. Others = Reserved.
[11]	**Reserved**	Reserved.

[10:8]	**GPD2_MFP**	**PD.2 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI0_MISO1 function is selected. 010 = The SPI2_MISO0 function is selected. 011 = The ADC2 function is selected. Others = Reserved.
[7]	**Reserved**	Reserved.
[6:4]	**GPD1_MFP**	**PD.1 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI0_SS1 function is selected. 010 = The SPI2_CLK2 function is selected. 011 = The ADC1 function is selected. Others = Reserved.
[3]	**Reserved**	Reserved.
[2:0]	**GPD0_MFP**	**PD.0 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI2_SS0 function is selected. 010 = The ADC0 function is selected. Others = Reserved.

Note: For NUC123xxxAEx, if GPx_MFPH and GPx_MFPL are used as pin multi-function setting, the GPx_MFP, ALT_MFP and ALT_MFP1 will become invalid.

GPIOD Multiple Function High Byte Control Register (GPD_MFPH)

Register	Offset	R/W	Description	Reset Value
GPD_MFPH	GCR_BA+0x14C	R/W	GPIOD Multiple Function High Byte Control Register (NUC123xxxAEx Only)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**GPD11_MFP**			**Reserved**	**GPD10_MFP**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**GPD9_MFP**			**Reserved**	**GPD8_MFP**		

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**GPD11_MFP**	**PD.11 Pin Function Selection** 000 = The GPIO function is selected. 001 = The INT1 function is selected. Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**GPD10_MFP**	**PD.10 Pin Function Selection** 000 = The GPIO function is selected. 001 = The CLKO function is selected. Others = Reserved.
[7]	**Reserved**	Reserved.
[6:4]	**GPD9_MFP**	**PD.9 Pin Function Selection** 000 = The GPIO function is selected. Others = Reserved.
[3]	**Reserved**	Reserved.
[2:0]	**GPD8_MFP**	**PD.8 Pin Function Selection** 000 = The GPIO function is selected. 001 = The SPI1_MOSI0 function is selected. Others = Reserved.

Note: For NUC123xxxAEx, if GPx_MFPH and GPx_MFPL are used as pin multi-function setting, the GPx_MFP, ALT_MFP and ALT_MFP1 will become invalid.

GPIOF Multiple Function Low Byte Control Register (GPF_MFPL)

Register	Offset	R/W	Description	Reset Value
GPF_MFPL	GCR_BA+0x158	R/W	GPIOF Multiple Function Low Byte Control Register (NUC123xxxAEx Only)	0x0000_11XX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**GPF3_MFP**			**Reserved**	**GPF2_MFP**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**GPF1_MFP**			**Reserved**	**GPF0_MFP**		

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**GPF3_MFP**	**PF.3 Pin Function Selection** 000 = The GPIO function is selected. 001 = The PS2_CLK function is selected. 010 = The I2C0_SCL function is selected. 011 = The ADC7 function is selected. Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**GPF2_MFP**	**PF.2 Pin Function Selection** 000 = The GPIO function is selected. 001 = The PS2_DAT function is selected. 010 = The I2C0_SDA function is selected. 011 = The ADC6 function is selected. Others = Reserved.
[7]	**Reserved**	Reserved.
[6:4]	**GPF1_MFP**	**PF.1 Pin Function Selection (Read Only)** 000 = The GPIO function is selected. 001 = The XT1_IN function is selected. Others = Reserved. These bits are controlled by CGPFMFP (Config0[27]). If CGPFMFP = 1, GPF1_MFP is 001. If CGPFMFP = 0, GPF1_MFP is 000.
[3]	**Reserved**	Reserved.

[2:0]	**GPF0_MFP**	**PF.0 Pin Function Selection (Read Only)** 000 = The GPIO function is selected. 001 = The XT1_OUT function is selected. Others = Reserved. These bits are controlled by CGPFMFP (Config[27]). If CGPFMFP = 1, GPF0_MFP is 001. If CGPFMFP = 0, GPF0_MFP is 000.

Note: For NUC123xxxAEx, if GPx_MFPH and GPx_MFPL are used as pin multi-function setting, the GPx_MFP, ALT_MFP and ALT_MFP1 will become invalid.

6.2.7 Register Protection

Some of the system control registers need to be protected to avoid inadvertent write and disturb the chip operation. These system control registers are protected after the power on reset till user to disable register protection. For user to program these protected registers, a register protection disable sequence needs to be followed by a special programming. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register REGWRPROT continuously. Any different data value, different sequence or any other write to other address during these three data writing will abort the whole sequence.

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

The protected registers are listed as following table.

Register	Bit	Description
IPRSTC1	[2] PDMA_RST	PDMA Controller Reset (Write Protect)
	[1] CPU_RST	CPU Kernel One-shot Reset (Write Protect)
	[0] CHIP_RST	CHIP One-shot Reset (Write Protect)
BODCR	[7] LVR_EN	Low Voltage Reset Enable Bit (Write Protect)
	[5] BOD_LPM	Brown-out Detector Low Power Mode (Write Protect)
	[3] BOD_RSTEN	Brown-out Reset Enable Bit (Write Protect)
	[2:1] BOD_VL	Brown-out Detector Threshold Voltage Selection (Write Protect)
	[0] BOD_EN	Brown-out Detector Enable Bit (Write Protect)
PORCR	[15:0] POR_DIS_CODE	Power-on-reset Enable Bits (Write Protect)
NMI_SEL	[8] NMI_EN	NMI Interrupt Enable Bit (Write Protect)
PWRCON	[8] PD_WAIT_CPU	Power-down Entry Conditions Control (Write Protect)
	[7] PWR_DOWN_EN	System Power-down Enable Bit (Write Protect)
	[5] PD_WU_INT_EN	Power-down Mode Wake-up Interrupt Enable Bit (Write Protect)
	[4] PD_WU_DLY	Enable the Wake-up Delay Counter (Write Protect)
	[3] OSC10K_EN	Internal 10 KHz Low Speed Oscillator Enable Bit (Write Protect)
	[2] OSC22M_EN	Internal 22.1184 MHz High Speed Oscillator Enable Bit (Write Protect)
	[0] XTL12M_EN	External 4~24 MHz High Speed Crystal Enable Bit (Write Protect)
APBCLK	[0] WDT_EN	Watchdog Timer Clock Enable Bit (Write Protect)
CLKSEL0	[5:3] STCLK_S	Cortex®-M0 SysTick Clock Source Selection (Write Protect)
	[2:0] HCLK_S	HCLK Clock Source Selection (Write Protect)
CLKSEL1	[1:0] WDT_S	Watchdog Timer Clock Source Selection (Write Protect)
ISPCON	[6] ISPFF	ISP Fail Flag (Write Protect)
	[5] LDUEN	LDROM Update Enable Bit (Write Protect)
	[4] CFGUEN	CONFIG Update Enable Bit (Write Protect)

	[3] APUEN	APROM Update Enable Bit (Write Protect)
	[1] BS	Boot Select (Write Protect)
	[0] ISPEN	ISP Enable Bit (Write Protect)
ISPTRG	[0] ISPGO	ISP Start Trigger (Write Protect)
FATCON	[6] MFOM	Middle Frequency Optimization Mode (Write Protect)
	[4] LFOM	Low Frequency Optimization Mode (Write Protect)
ISPSTA	[6] ISPFF	ISP Fail Flag (Write Protect)
TCSR0	[31] DBGACK_TMR	ICE Debug Mode Acknowledge Disable (Write Protect)
TCSR1	[31] DBGACK_TMR	ICE Debug Mode Acknowledge Disable (Write Protect)
TCSR2	[31] DBGACK_TMR	ICE Debug Mode Acknowledge Disable (Write Protect)
TCSR3	[31] DBGACK_TMR	ICE Debug Mode Acknowledge Disable (Write Protect)
WTCR	[31] DBGACK_WDT	ICE Debug Mode Acknowledge Disable (Write Protect)
	[10:8] WTIS	WDT Time-out Interval Selection (Write Protect)
	[7] WTE	WDT Enable Control (Write Protect)
	[6] WTIE	WDT Time-out Interrupt Enable Control (Write Protect)
	[5] WTWKF	WDT Time-out Wake-up Flag (Write Protect)
	[4] WTWKE	WDT Time-out Wake-up Function Control (Write Protect)
	[1] WTRE	WDT Time-out Reset Enable Control (Write Protect)
WTCRALT	[1:0] WTRDSEL	WDT Reset Delay Selection (Write Protect)

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

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

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

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

6.2.8.1 *System Timer Control Register Map*

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

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

6.2.8.2 *System Timer Control Register Description*

SysTick Control and Status (SYST_CSR)

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

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**COUNTFLAG**	**System Tick Counter Flag** Returns 1 if timer counted to 0 since last time this register was read. COUNTFLAG is set by a count transition from 1 to 0. COUNTFLAG is cleared on read or by a write to the Current Value register.
[15:3]	**Reserved**	Reserved.
[2]	**CLKSRC**	**System Tick Clock Source Selection** 0 = Clock source is (optional) external reference clock, SysTick clock source is defined by STCLK_S (CLKSEL0[5:3]). 1 = Core clock used for SysTick and SysTick clock source is from HCLK.
[1]	**TICKINT**	**System Tick Interrupt Enabled** 0 = Counting down to 0 does not cause the SysTick exception to be pended. Software can use COUNTFLAG to determine if a count to 0 has occurred. 1 = Counting down to 0 will cause the SysTick exception to be pended. Clearing the SysTick Current Value register by a write in software will not cause SysTick to be pended.
[0]	**ENABLE**	**System Tick Counter Enabled** 0 = Counter Disabled. 1 = Counter will operate in a multi-shot manner.

SysTick Reload Value Register (SYST_RVR)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**RELOAD**	**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	SCS _BA+0x18	R/W	SysTick Current Value Register	0xXXXX_XXXX

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CURRENT**	**System Tick Current Value** Current counter value. This is the value of the counter at the time it is sampled. The counter does not provide read-modify-write protection. The register is write-clear. A software write of any value will clear the register to 0.

6.2.9 Nested Vectored Interrupt Controller (NVIC)

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

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

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

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

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

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

6.2.9.1 *Exception Model and System Interrupt Map*

Table 6-8 lists the exception model supported by the NuMicro® NUC123 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 6-8 Exception Model

Vector Number	Interrupt Number (Bit In Interrupt Registers)	Interrupt Name	Source IP	Interrupt Description
0 ~ 15	-	-	-	System exceptions
16	0	**BOD_OUT**	Brown-out	Brown-out low voltage detected interrupt
17	1	**WDT_INT**	WDT	Watchdog/Window Watchdog Timer interrupt
18	2	**EINT0**	GPIO	External signal interrupt from PB.14 pin
19	3	**EINT1**	GPIO	External signal interrupt from PB.15 or PD.11 pin
20	4	**GPAB_INT**	GPIO	External signal interrupt from PA[15:0]/PB[13:0]
21	5	**GPCDF_INT**	GPIO	External interrupt from PC[15:0]/PD[15:0]/PF[3:0]
22	6	**PWMA_INT**	PWM0~3	PWM0, PWM1, PWM2 and PWM3 interrupt
23	7	**Reserved**	Reserved	Reserved
24	8	**TMR0_INT**	TMR0	Timer 0 interrupt
25	9	**TMR1_INT**	TMR1	Timer 1 interrupt
26	10	**TMR2_INT**	TMR2	Timer 2 interrupt
27	11	**TMR3_INT**	TMR3	Timer 3 interrupt
28	12	**UART0_INT**	UART0	UART0 interrupt
29	13	**UART1_INT**	UART1	UART1 interrupt
30	14	**SPI0_INT**	SPI0	SPI0 interrupt

31	15	**SPI1_INT**	SPI1	SPI1 interrupt
32	16	**SPI2_INT**	SPI2	SPI2 interrupt
33	17	**Reserved**	Reserved	Reserved
34	18	**I2C0_INT**	I^2C0	I^2C0 interrupt
35	19	**I2C1_INT**	I^2C1	I^2C1 interrupt
36	20	**Reserved**	Reserved	Reserved
37	21	**Reserved**	Reserved	Reserved
38	22	**Reserved**	Reserved	Reserved
39	23	**USB_INT**	USBD	USB 2.0 FS Device interrupt
40	24	**PS2_INT**	PS/2	PS/2 interrupt
41	25	**Reserved**	Reserved	Reserved
42	26	**PDMA_INT**	PDMA	PDMA interrupt
43	27	**I2S_INT**	I^2S	I^2S interrupt
44	28	**PWRWU_INT**	CLKC	Clock controller interrupt for chip wake-up from Power-down state
45	29	**ADC_INT**	ADC	ADC interrupt
46	30	**Reserved**	Reserved	Reserved
47	31	**Reserved**	Reserved	Reserved

Table 6-9 System Interrupt Map

6.2.9.2 *Vector Table*

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

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

Table 6-10 Vector Table 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 activate. If an interrupt is Active when it is disabled, it remains in its Active state until cleared by reset or an exception return. Clearing the enable bit prevents new activations of the associated interrupt.

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

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

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

6.2.9.4 *NVIC Control Registers*

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

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

IRQ0 ~ IRQ31 Set-Enable Control Register (NVIC_ISER)

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

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

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

IRQ0 ~ IRQ31 Clear-Enable Control Register (NVIC_ICER)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER	SCS _BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000

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

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

IRQ0 ~ IRQ31 Set-Pending Control Register (NVIC_ISPR)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR	SCS _BA+0x200	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000

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

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

IRQ0 ~ IRQ31 Clear-Pending Control Register (NVIC_ICPR)

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

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

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

IRQ0 ~ IRQ3 Interrupt Priority Register (NVIC_IPR0)

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

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

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

IRQ4 ~ IRQ7 Interrupt Priority Register (NVIC_IPR1)

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

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

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

IRQ8 ~ IRQ11 Interrupt Priority Register (NVIC_IPR2)

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

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

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

IRQ12 ~ IRQ15 Interrupt Priority Register (NVIC_IPR3)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR3	SCS _BA+0x40C	R/W	IRQ12 ~ IRQ15 Interrupt Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_15		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_14		**Reserved**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_13		**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_12		**Reserved**					

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

IRQ16 ~ IRQ19 Interrupt Priority Register (NVIC_IPR4)

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

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

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

IRQ20 ~ IRQ23 Interrupt Priority Register (NVIC_IPR5)

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

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

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

IRQ24 ~ IRQ27 Interrupt Priority Register (NVIC_IPR6)

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

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

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

IRQ28 ~ IRQ31 Interrupt Priority Register (NVIC_IPR7)

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

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

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

6.2.9.5 *Interrupt Source Control Registers*

Besides the interrupt control registers associated with the NVIC, the NuMicro® NUC123 series also implement some specific control registers to facilitate the interrupt functions, including "interrupt source identification", "NMI source selection" and "interrupt test mode", which are described below.

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

Register	Offset	R/W	Description	Reset Value
INT Base Address: **INT_BA = 0x5000_0300**				
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) interrupt source identity	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) interrupt source identity	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) interrupt source identity	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) interrupt source identity	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/B) interrupt source identity	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (GPC/D/F) interrupt source identity	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWMA) interrupt source identity	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (Reserved) interrupt source identity	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) interrupt source identity	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) interrupt source identity	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) interrupt source identity	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) interrupt source identity	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (URT0) interrupt source identity	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (URT1) interrupt source identity	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) interrupt source identity	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (SPI1) interrupt source identity	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	IRQ16 (SPI2) interrupt source identity	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (Reserved) interrupt source identity	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I^2C0) interrupt source identity	0xXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I^2C1) interrupt source identity	0xXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	R	IRQ20 (Reserved) interrupt source identity	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (Reserved) interrupt source identity	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (Reserved) interrupt source identity	0xXXXX_XXXX
IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USBD) interrupt source identity	0xXXXX_XXXX

IRQ24_SRC	INT_BA+0x60	R	IRQ24 (PS/2) interrupt source identity	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (Reserved) interrupt source identity	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (PDMA) interrupt source identity	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	IRQ27 (I^2S) interrupt source identity	0xXXXX_XXXX
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) interrupt source identity	0xXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) interrupt source identity	0xXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	R	IRQ30 (Reserved) interrupt source identity	0xXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	R	IRQ31 (Reserved) interrupt source identity	0xXXXX_XXXX
NMI_SEL	INT_BA+0x80	R/W	NMI source interrupt select control register	0x0000_0000
MCU_IRQ	INT_BA+0x84	R/W	MCU IRQ Number identity register	0x0000_0000

IRQ0 (BOD) Interrupt Source Identity (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							**BOD_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**BOD_INT**	**IRQ0 Source Identity** 0 = IRQ0 source is not from BOD interrupt (BOD_INT). 1 = IRQ0 source is from BOD interrupt (BOD_INT).

IRQ1 (WDT) Interrupt Source Identity (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							**WDT_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WDT_INT**	**IRQ1 Source Identity** 0 = IRQ1 source is not from watchdog interrupt (WDT _INT). 1 = IRQ1 source is from watchdog interrupt (WDT_INT).

IRQ2 (EINT0) Interrupt Source Identity (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							**EINT0**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**EINT0**	**IRQ2 Source Identity** 0 = IRQ2 source is not from external signal interrupt 0 from PB.14 (EINT0). 1 = IRQ2 source is from external signal interrupt 0 from PB.14 (EINT0).

IRQ3 (EINT1) Interrupt Source Identity (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							**EINT1**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**EINT1**	**IRQ3 Source Identity** 0 = IRQ3 source is not from external signal interrupt 1 from PB.15 or PD.11 (EINT1). 1 = IRQ3 source is from external signal interrupt 1 from PB.15 or PD.11 (EINT1).

IRQ4 (GPA/B) Interrupt Source Identity (IRQ4_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/B) 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						**GPB_INT**	**GPA_INT**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**GPB_INT**	**IRQ4 Source Identity** 0 = IRQ4 source is not from GPB interrupt (GPB_INT). 1 = IRQ4 source is from GPB interrupt (GPB_INT).
[0]	**GPA_INT**	**IRQ4 Source Identity** 0 = IRQ4 source is not from GPA interrupt (GPA_INT). 1 = IRQ4 source is from GPA interrupt (GPA_INT).

IRQ5 (GPC/D/F) Interrupt Source Identity (IRQ5_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (GPC/D/F) 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				**GPF_INT**	**Reserved**	**GPD_INT**	**GPC_INT**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[3]	**GPF_INT**	**IRQ5 Source Identity** 0 = IRQ5 source is not from GPF interrupt (GPF_INT). 1 = IRQ5 source is from GPF interrupt (GPF_INT).
[2]	**Reserved**	Reserved.
[1]	**GPD_INT**	**IRQ5 Source Identity** 0 = IRQ5 source is not from GPD interrupt (GPD_INT). 1 = IRQ5 source is from GPD interrupt (GPD_INT).
[0]	**GPC_INT**	**IRQ5 Source Identity** 0 = IRQ5 source is not from GPC interrupt (GPC_INT). 1 = IRQ5 source is from GPC interrupt (GPC_INT).

IRQ6 (PWMA) Interrupt Source Identity (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				**PWM3_INT**	**PWM2_INT**	**PWM1_INT**	**PWM0_INT**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PWM3_INT**	**IRQ6 Source Identity** 0 = IRQ6 source is not from PWM3 interrupt (PWM3_INT). 1 = IRQ6 source is from PWM3 interrupt (PWM3_INT).
[2]	**PWM2_INT**	**IRQ6 Source Identity** 0 = IRQ6 source is not from PWM2 interrupt (PWM2_INT). 1 = IRQ6 source is from PWM2 interrupt (PWM2_INT).
[1]	**PWM1_INT**	**IRQ6 Source Identity** 0 = IRQ6 source is not from PWM1 interrupt (PWM1_INT). 1 = IRQ6 source is from PWM1 interrupt (PWM1_INT).
[0]	**PWM0_INT**	**IRQ6 Source Identity** 0 = IRQ6 source is not from PWM0 interrupt (PWM0_INT). 1 = IRQ6 source is from PWM0 interrupt (PWM0_INT).

IRQ8 (TMR0) Interrupt Source Identity (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							**TMR0_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**TMR0_INT**	**IRQ8 Source Identity** 0 = IRQ8 source is not from Timer0 interrupt (TMR0_INT). 1 = IRQ8 source is from Timer0 interrupt (TMR0_INT).

IRQ9 (TMR1) Interrupt Source Identity (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							**TMR1_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**TMR1_INT**	**IRQ9 Source Identity** 0 = IRQ9 source is not from Timer1 interrupt (TMR1_INT). 1 = IRQ9 source is from Timer1 interrupt (TMR1_INT).

IRQ10 (TMR2) Interrupt Source Identity (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							**TMR2_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**TMR2_INT**	**IRQ10 Source Identity** 0 = IRQ10 source is not from Timer2 interrupt (TMR2_INT). 1 = IRQ10 source is from Timer2 interrupt (TMR2_INT).

IRQ11 (TMR3) Interrupt Source Identity (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							**TMR3_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**TMR3_INT**	**IRQ11 Source Identity** 0 = IRQ11 source is not from Timer3 interrupt (TMR3_INT). 1 = IRQ11 source is from Timer3 interrupt (TMR3_INT).

IRQ12 (UART0) Interrupt Source Identity (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							**UART0_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**UART0_INT**	**IRQ12 Source Identity** 0 = IRQ12 source is not from UART0 interrupt (UART0_INT). 1 = IRQ12 source is from UART0 interrupt (UART0_INT).

IRQ13 (UART1) Interrupt Source Identity (IRQ13_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (UART1) 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							**UART1_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**UART1_INT**	**IRQ13 Source Identity** 0 = IRQ13 source is not from UART1 interrupt (UART1_INT). 1 = IRQ13 source is from UART1 interrupt (UART1_INT).

IRQ14 (SPI0) Interrupt Source Identity (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							**SPI0_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SPI0_INT**	**IRQ14 Source Identity** 0 = IRQ14 source is not from SPI0 interrupt (SPI0_INT). 1 = IRQ14 source is from SPI0 interrupt (SPI0_INT).

IRQ15 (SPI1) Interrupt Source Identity (IRQ15_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (SPI1) 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							**SPI1_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SPI1_INT**	**IRQ15 Source Identity** 0 = IRQ15 source is not from SPI1 interrupt (SPI1_INT). 1 = IRQ15 source is from SPI1 interrupt (SPI1_INT).

IRQ16 (SPI2) Interrupt Source Identity (IRQ16_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ16_SRC	INT_BA+0x40	R	IRQ16 (SPI2) 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							**SPI2_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SPI2_INT**	**IRQ16 Source Identity** 0 = IRQ16 source is not from SPI2 interrupt (SPI2_INT). 1 = IRQ16 source is from SPI2 interrupt (SPI2_INT).

IRQ18 (I^2C0) Interrupt Source Identity (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							**I2C0_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**I2C0_INT**	**IRQ18 Source Identity** 0 = IRQ18 source is not from I^2C0 interrupt (I2C0_INT). 1 = IRQ18 source is from I^2C0 interrupt (I2C0_INT).

IRQ19 (I^2C1) Interrupt Source Identity (IRQ19_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I^2C1) 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							**I2C1_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**I2C1_INT**	**IRQ19 Source Identity** 0 = IRQ19 source is not from I^2C1 interrupt (I2C1_INT). 1 = IRQ19 source is from I^2C1 interrupt (I2C1_INT).

IRQ23 (USBD) Interrupt Source Identity (IRQ23_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USBD) 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							**USBD_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**USBD_INT**	**IRQ23 Source Identity** 0 = IRQ23 source is not from USBD interrupt (USBD_INT). 1 = IRQ23 source is from USBD interrupt (USBD_INT).

IRQ24 (PS/2) Interrupt Source Identity (IRQ24_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ24_SRC	INT_BA+0x60	R	IRQ24 (PS/2) 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							**PS2_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PS2_INT**	**IRQ24 Source Identity** 0 = IRQ24 source is not from PS/2 interrupt (PS2_INT). 1 = IRQ24 source is from PS/2 interrupt (PS2_INT).

IRQ26 (PDMA) Interrupt Source Identity (IRQ26_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (PDMA) 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							**PDMA_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PDMA_INT**	**IRQ26 Source Identity** 0 = IRQ26 source is not from PDMA interrupt (PDMA_INT). 1 = IRQ26 source is from PDMA interrupt (PDMA_INT).

IRQ27 (I^2S) Interrupt Source Identity (IRQ27_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ27_SRC	INT_BA+0x6C	R	IRQ27 (I^2S) 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							**I2S_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**I2S_INT**	**IRQ27 Source Identity** 0 = IRQ27 source is not from I^2S interrupt (I2S_INT). 1 = IRQ27 source is from I^2S interrupt (I2S_INT).

IRQ28 (PWRWU) Interrupt Source Identity (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							**PWRWU_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PWRWU_INT**	**IRQ28 Source Identity** 0 = IRQ28 source is not from PWRWU interrupt (PWRWU_INT). 1 = IRQ28 source is from PWREU interrupt (PWRWU_INT).

IRQ29 (ADC) Interrupt Source Identity (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							**ADC_INT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**ADC_INT**	**IRQ29 Source Identity** 0 = IRQ29 source is not from ADC interrupt (ADC_INT). 1 = IRQ29 source is from ADC interrupt (ADC_INT).

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**				

Bits	Description	
[31:8]	**Reserved**	Reserved.
[8]	**NMI_EN**	**NMI Interrupt Enable Bit (Write Protect)** 0 = NMI interrupt Disabled. 1 = NMI interrupt Enabled. **Note:** This bit is write protected bit. Refer to the REGWRPROT register.
[7:5]	**Reserved**	Reserved.
[4:0]	**NMI_SEL**	**NMI Interrupt Source Selection** The NMI interrupt to Cortex®-M0 can be selected from one of the peripheral interrupt by setting NMI_SEL with corresponding interrupt number.

MCU Interrupt Request Source Register (MCU_IRQ)

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

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

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

6.2.10 System Control Register

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

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

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

Register	Offset	R/W	Description	Reset Value
SCS Base Address: **SCS_BA = 0xE000_E000**				
CPUID	SCS_BA+0xD00	R	CPUID Register	0x410C_C200
ICSR	SCS_BA+0xD04	R/W	Interrupt Control and State Register	0x0000_0000
AIRCR	SCS_BA+0xD0C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000
SCR	SCS_BA+0xD10	R/W	System Control Register	0x0000_0000
SHPR2	SCS_BA+0xD1C	R/W	System Handler Priority Register 2	0x0000_0000
SHPR3	SCS_BA+0xD20	R/W	System Handler Priority Register 3	0x0000_0000

CPUID Register (CPUID)

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

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

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

Interrupt Control State Register (ICSR)

Register	Offset	R/W	Description	Reset Value
ICSR	SCS_BA+0xD04	R/W	Interrupt Control and State Register	0x0000_0000

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

Bits	Description	
[31]	**NMIPENDSET**	**NMI Set-pending Bit** Write Operation: 0 = No effect. 1 = Changes NMI exception state to pending. Read Operation: 0 = NMI exception is not pending. 1 = NMI exception is 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 Operation: 0 = No effect. 1 = Changes PendSV exception state to pending. Read Operation: 0 = PendSV exception is not pending. 1 = PendSV exception is pending. **Note:** Writing 1 to this bit is the only way to set the PendSV exception state to pending.
[27]	**PENDSVCLR**	**PendSV Clear-pending Bit** Write Operation: 0 = No effect. 1 = Removes the pending state from the PendSV exception. **Note:** This is a write only bit. When you want to clear PENDSV bit, you must “write 0 to PENDSVSET and write 1 to PENDSVCLR” at the same time.

[26]	**PENDSTSET**	**SysTick Exception Set-pending Bit** Write Operation: 0 = No effect. 1 = Changes SysTick exception state to pending. Read Operation: 0 = SysTick exception is not pending. 1 = SysTick exception is pending.
[25]	**PENDSTCLR**	**SysTick Exception Clear-pending Bit (Write Only)** Write Operation: 0 = No effect. 1 = Removes the pending state from the SysTick exception. **Note:** This is a write only bit. 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 (Read Only)** If set, a pending exception will be serviced on exit from the debug halt state.
[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 = The exception number of the currently active exception.

Application Interrupt and Reset Control Register (AIRCR)

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

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

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

System Control Register (SCR)

Register	Offset	R/W	Description	Reset Value
SCR	SCS_BA+0xD10	R/W	System Control Register	0x0000_0000

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

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

System Handler Priority Register 2 (SHPR2)

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

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

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

System Handler Priority Register 3 (SHPR3)

Register	Offset	R/W	Description	Reset Value
SHPR3	SCS_BA+0xD20	R/W	System Handler Priority Register 3	0x0000_0000

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

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

6.3 Clock Controller

6.3.1 Overview

The clock controller generates the clocks for the whole chip, including system clocks and all peripheral clocks. The clock controller also implements the power control function with the individually clock ON/OFF control, clock source selection and clock divider. The chip enters Power-down mode when Cortex®-M0 core executes the WFI instruction only if the PWR_DOWN_EN (PWRCON[7]) bit and PD_WAIT_CPU (PWRCON[8]) bit are both set to 1. After that, chip enters Power-down mode and wait for wake-up interrupt source triggered to leave Power-down mode. In the Power-down mode, the clock controller turns off the 4~24 MHz external high speed crystal oscillator and 22.1184 MHz internal high speed RC oscillator to reduce the overall system power consumption. The Figure 6-9 and Figure 6-10 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 FOUT), PLL source can be from 4~24 MHz external high speed crystal oscillator (HXT) or 22.1184 MHz internal high speed RC oscillator (HIRC))
- 22.1184 MHz internal high speed RC oscillator (HIRC)
- 10 kHz internal low speed RC oscillator (LIRC)

Each of these clock sources has certain stable time to wait for clock operating at stable frequency. When clock source is enabled, a stable counter start counting and correlated clock stable index (OSC22M_STB(CLKSTATUS[4]), OSC10K_STB(CLKSTATUS[3]), PLL_STB(CLKSTATUS[2]) and XTL12M_STB(CLKSTATUS[0])) are set to 1 after stable counter value reach a define value asshown in Table 6-11. System and peripheral can use the clock as its operating clock only when correlate clock stable index is set to 1. The clock stable index will auto clear when user disables the clock source (OSC10K_EN(PWRCON[3]), OSC22M_EN(PWRCON[2]), XTL12M_EN(PWRCON[0]) and PD(PLLCON[16])). Besides, the clock stable index of HXT, HIRC and PLL will auto clear when chip enter power-down and clock stable counter will re-counting after chip wake-up if correlate clock is enabled.

Clock Source	Clock Stable Count Value
HXT	4096 HXT clock
PLL	6144 PLL source (PLL source is HXT if PLL_SRC(PLLCON[19]) = 0, or HIRC if PLL_SRC(PLLCON[19]) = 1)
HIRC	256 HIRC clock
LIRC	1 LIRC

Table 6-11 Clock Stable Count Value Table

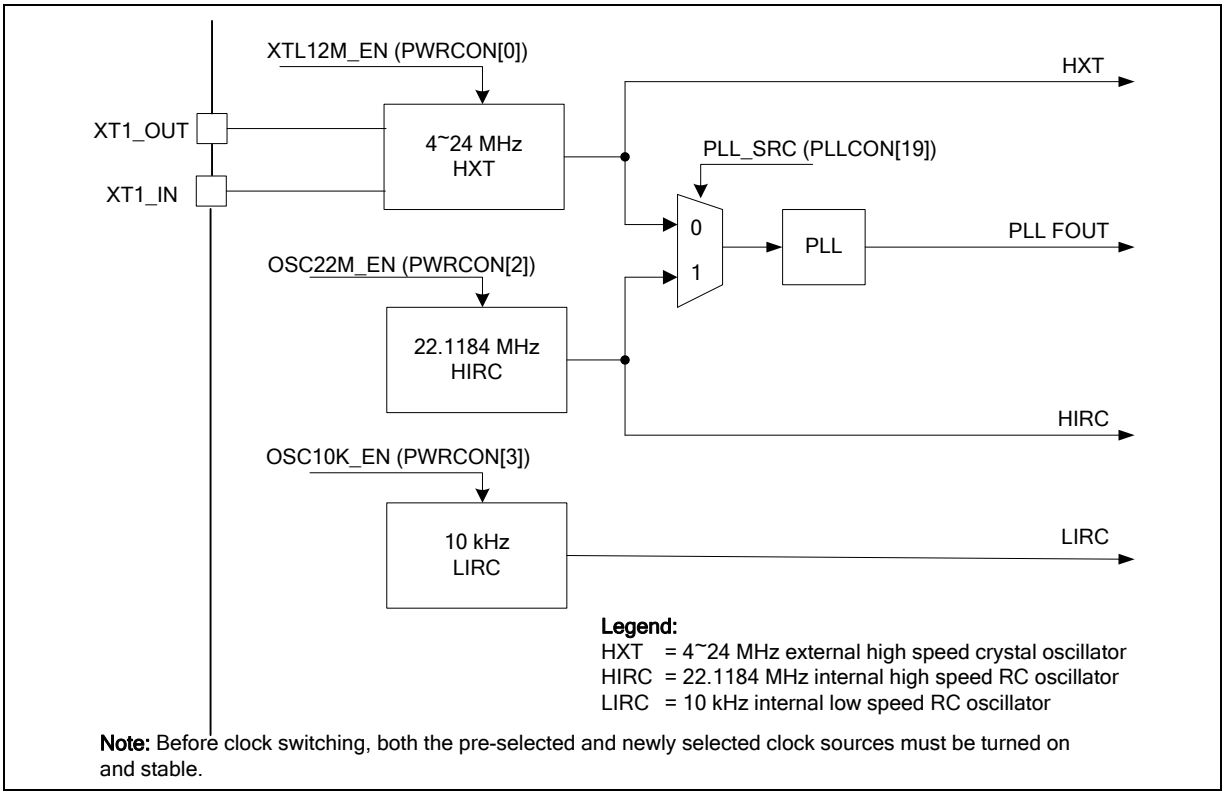
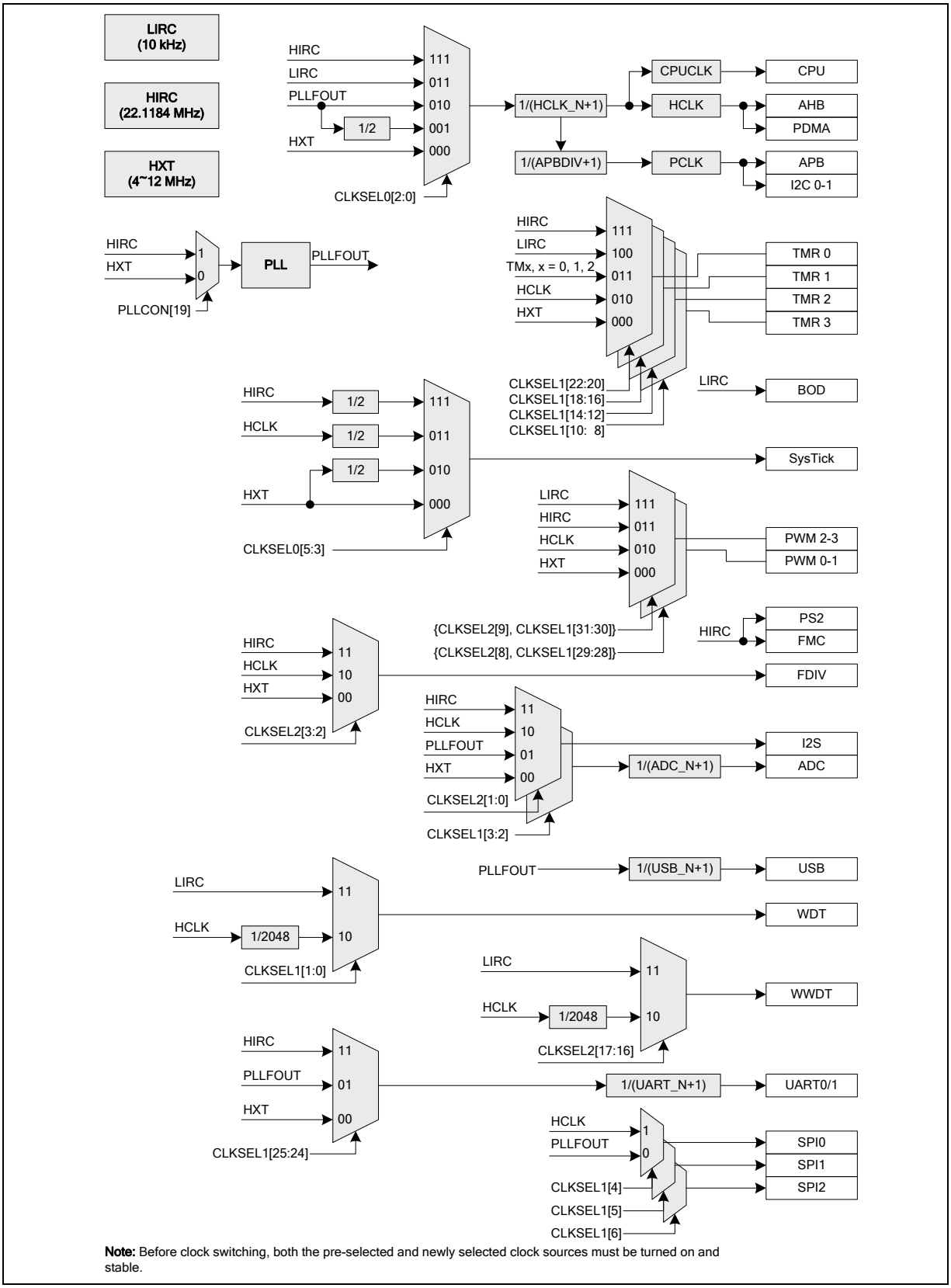


Figure 6-9 Clock Generator Global View Diagram

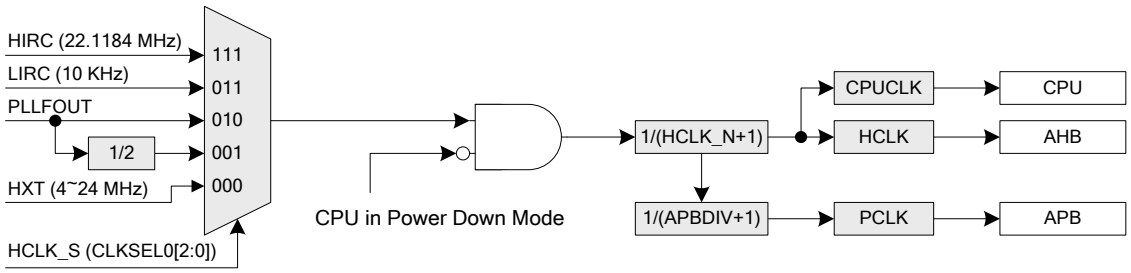


Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 6-10 Clock Generator Global View Diagram

6.3.2 System Clock and SysTick Clock

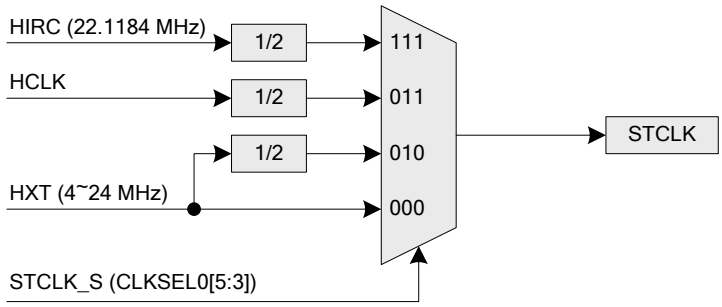
The system clock has 5 clock sources which were generated from clock generator block. The clock source switch depends on the register HCLK_S (CLKSEL0[2:0]). The block diagram is shown in Figure 6-11.



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

Figure 6-11 System Clock Block Diagram

The clock source of SysTick in Cortex®-M0 core can use CPU clock or external clock (SYST_CSR[2]). If using external clock, the SysTick clock (STCLK) has 4 clock sources. The clock source switch depends on the setting of the register STCLK_S (CLKSEL0[5:3]). The block diagram is shown in Figure 6-12.



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

Figure 6-12 SysTick Clock Control Block Diagram

6.3.3 Peripherals Clock

The peripherals clock had different clock source switch setting depending on different peripherals. Please refer to the CLKSEL1 and CLKSEL2 register description in 6.3.7.

6.3.4 Power-down Mode Clock

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

The clocks kept active are listed below:

- Clock Generator
  - Internal 10 kHz low speed oscillator clock

- WDT/Timer/PWM Peripherals Clock (when 10 kHz intertnal low speed RC oscillator (LIRC) is adopted as clock source)

6.3.5 Frequency Divider Output

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

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

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

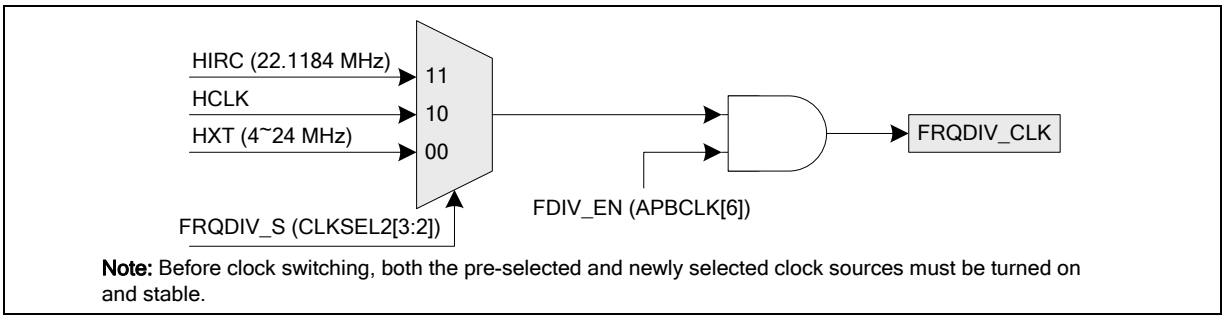


Figure 6-13 Clock Source of Frequency Divider

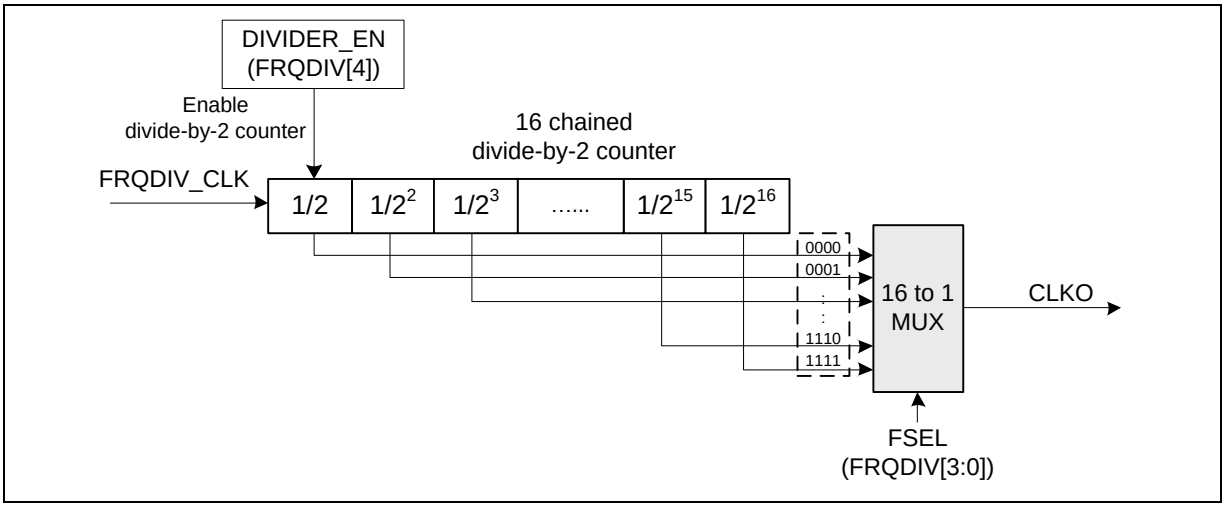


Figure 6-14 Block Diagram of Frequency Divider

6.3.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
CLK_BA = 0x5000_0200				
PWRCON	CLK_BA+0x00	R/W	System Power-down Control Register	0x0000_001X
AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_0004
APBCLK	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register	0x0000_000X
CLKSTATUS	CLK_BA+0x0C	R/W	Clock Status Monitor Register	0x0000_00XX
CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0000_003X
CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0xFFFF_FFFF
CLKSEL2	CLK_BA+0x1C	R/W	Clock Source Select Control Register 2	0x0002_00FF
CLKDIV	CLK_BA+0x18	R/W	Clock Divider Number Register	0x0000_0000
PLLCON	CLK_BA+0x20	R/W	PLL Control Register	0x0005_C22E
FRQDIV	CLK_BA+0x24	R/W	Frequency Divider Control Register	0x0000_0000
APBDIV	CLK_BA+0x2C	R/W	APB Divider Control Register	0x0000_0000

6.3.7 Register Description

System Power-down Control Register (PWRCON)

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

Register	Offset	R/W	Description	Reset Value
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 Conditions Control (Write Protect)** 0 = Chip entering Power-down mode when the PWR_DOWN_EN bit is set to 1. 1 = Chip entering Power-down mode when the both PD_WAIT_CPU and PWR_DOWN_EN bits are set to 1 and CPU run WFI instruction. **Note:** This bit is write protected bit. Refer to the REGWRPROT register.
[7]	**PWR_DOWN_EN**	**System Power-down Enable Bit (Write Protect)** When this bit is set to 1, the chip 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, the chip enters Power-down mode immediately after the PWR_DOWN_EN bit set. (b) If the PD_WAIT_CPU is 1, the chip keeps active till the CPU sleep mode is also active and then the chip enters Power-down mode. When chip wakes up from Power-down mode, this bit is auto cleared. 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 RC oscillator (HIRC) will be disabled in this mode, but the 10 kHz internal low speed RC oscillator (LIRC) is 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 the 10 kHz internal low speed RC oscillator (LIRC). 0 = Chip operating normally or chip in Idle mode because of WFI command. 1 = Chip entering the Power-down mode instantly or wait CPU sleep command WFI. **Note:** This bit is write protected bit. Refer to the REGWRPROT register.

[6]	**PD_WU_STS**	**Power-down Mode Wake-up Interrupt Status** Set by "Power-down wake-up event", which indicates that resuming from Power-down mode". The flag is set if the GPIO, USB, UART, WDT, TIMER, I^2C or BOD wake-up occurred. This bit can be cleared to 0 by software writing '1'. **Note:** This bit works only when PD_WU_INT_EN (PWRCON[5]) set to 1.
[5]	**PD_WU_INT_EN**	**Power-down Mode Wake-up Interrupt Enable Bit (Write Protect)** 0 = Power-down mode wake-up interrupt Disabled. 1 = Power-down mode wake-up interrupt Enabled. **Note1:** The interrupt will occur when both PD_WU_STS and PD_WU_INT_EN are high. **Note2:** This bit is write protected bit. Refer to the REGWRPROT register.
[4]	**PD_WU_DLY**	**Enable the Wake-up Delay Counter (Write Protect)** When the chip wakes up from Power-down mode, the clock control will delay certain clock cycles to wait system clock stable. The delayed clock cycle is 4096 clock cycles when chip work at external 4~24 MHz high speed crystal, and 256 clock cycles when chip work at internal 22.1184 MHz high speed oscillator. 0 = Clock cycles delay Disabled. 1 = Clock cycles delay Enabled. **Note:** This bit is write protected bit. Refer to the REGWRPROT register.
[3]	**OSC10K_EN**	**Internal 10 KHz Low Speed Oscillator Enable Bit (Write Protect)** 0 = Internal 10 kHz low speed oscillator Disabled. 1 = Internal 10 kHz low speed oscillator Enabled. **Note:** This bit is write protected bit. Refer to the REGWRPROT register.
[2]	**OSC22M_EN**	**Internal 22.1184 MHz High Speed Oscillator Enable Bit (Write Protect)** 0 = Internal 22.1184 MHz high speed oscillator Disabled. 1 = Internal 22.1184 MHz high speed oscillator Enabled. **Note:** This bit is write protected bit. Refer to the REGWRPROT register.
[1]	**Reserved**	Reserved.
[0]	**XTL12M_EN**	**External 4~24 MHz High Speed Crystal Enable Bit (Write Protect)** The bit default value is set by flash controller user configuration register CFOSC (Config0 [26:24]). When the default clock source is from external 4~24 MHz high speed crystal, this bit is set to 1 automatically. 0 = External 4~24 MHz high speed crystal Disabled. 1 = External 4~24 MHz high speed crystal Enabled. **Note:** This bit is write protected bit. Refer to the REGWRPROT register.

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

Sleep mode)					their clock source are selected as 10 kHz (LIRC).

Table 6-12 Power-Down Mode Control Table

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

AHB Devices Clock Enable Control Register (AHBCLK)

These bits are used to enable/disable clock for system clock PDMA clock and ISP clock.

Register	Offset	R/W	Description	Reset Value
AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_0004

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**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**	**PDMA_EN**	**Reserved**

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

APB Devices Clock Enable Control Register (APBCLK)

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

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

31	30	29	28	27	26	25	24
PS2_EN	**Reserved**	**I2S_EN**	**ADC_EN**	**USBD_EN**	**Reserved**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**PWM23_EN**	**PWM01_EN**	**Reserved**		**UART1_EN**	**UART0_EN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**SPI2_EN**	**SPI1_EN**	**SPI0_EN**	**Reserved**		**I2C1_EN**	**I2C0_EN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**FDIV_EN**	**TMR3_EN**	**TMR2_EN**	**TMR1_EN**	**TMR0_EN**	**Reserved**	**WDT_EN**

Bits	Description	
[31]	**PS2_EN**	**PS/2 Clock Enable Bit** 0 = PS/2 clock Disabled. 1 = PS/2 clock Enabled.
[30]	**Reserved**	Reserved.
[29]	**I2S_EN**	**I^2S Clock Enable Bit** 0 = I^2S Clock Disabled. 1 = I^2S Clock Enabled.
[28]	**ADC_EN**	**Analog-digital-converter (ADC) Clock Enable Bit** 0 = ADC clock Disabled. 1 = ADC clock Enabled.
[27]	**USBD_EN**	**USB 2.0 FS Device Controller Clock Enable Bit** 0 = USB clock Disabled. 1 = USB clock Enabled.
[26:22]	**Reserved**	Reserved.
[21]	**PWM23_EN**	**PWM_23 Clock Enable Bit** 0 = PWM23 clock Disabled. 1 = PWM23 clock Enabled.
[20]	**PWM01_EN**	**PWM_01 Clock Enable Bit** 0 = PWM01 clock Disabled. 1 = PWM01 clock Enabled.
[19:18]	**Reserved**	Reserved.
[17]	**UART1_EN**	**UART1 Clock Enable Bit** 0 = UART1 clock Disabled. 1 = UART1 clock Enabled.

[16]	**UART0_EN**	**UART0 Clock Enable Bit** 0 = UART0 clock Disabled. 1 = UART0 clock Enabled.
[15]	**Reserved**	Reserved.
[14]	**SPI2_EN**	**SPI2 Clock Enable Bit** 0 = SPI2 clock Disabled. 1 = SPI2 clock Enabled.
[13]	**SPI1_EN**	**SPI1 Clock Enable Bit** 0 = SPI1 clock Disabled. 1 = SPI1 clock Enabled.
[12]	**SPI0_EN**	**SPI0 Clock Enable Bit** 0 = SPI0 clock Disabled. 1 = SPI0 clock Enabled.
[11:10]	**Reserved**	Reserved.
[9]	**I2C1_EN**	**I^2C1 Clock Enable Bit** 0 = I^2C1 clock Disabled. 1 = I^2C1 clock Enabled.
[8]	**I2C0_EN**	**I^2C0 Clock Enable Bit** 0 = I^2C0 clock Disabled. 1 = I^2C0 clock Enabled.
[7]	**Reserved**	Reserved.
[6]	**FDIV_EN**	**Frequency Divider Output Clock Enable Bit** 0 = FDIV clock Disabled. 1 = FDIV clock Enabled.
[5]	**TMR3_EN**	**Timer3 Clock Enable Bit** 0 = Timer3 clock Disabled. 1 = Timer3 clock Enabled.
[4]	**TMR2_EN**	**Timer2 Clock Enable Bit** 0 = Timer2 clock Disabled. 1 = Timer2 clock Enabled.
[3]	**TMR1_EN**	**Timer1 Clock Enable Bit** 0 = Timer1 clock Disabled. 1 = Timer1 clock Enabled.
[2]	**TMR0_EN**	**Timer0 Clock Enable Bit** 0 = Timer0 clock Disabled. 1 = Timer0 clock Enabled.
[1]	**Reserved**	Reserved.
[0]	**WDT_EN**	**Watchdog Timer Clock Enable Bit (Write Protect)** 0 = Watchdog Timer Clock Disabled. 1 = Watchdog Timer Clock Enabled. **Note:** This bit is write protected bit. Refer to the REGWRPROT register.

Clock Status Monitor Register (CLKSTATUS)

The bits of this register are used to monitor if the chip clock source is stable or not, and if clock switch is 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 failure. **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:** On NUC123xxxANx, this bit can be cleared to 0 by software writing '1'. **Note3:** On NUC123xxxAEx, this bit is read only. After selected clock source is stable, hardware will switch system clock to selected clock automatically, and CLK_SW_FAIL will be cleared automatically by hardware.
[6:5]	**Reserved**	Reserved.
[4]	**OSC22M_STB**	**Internal 22.1184 MHz High Speed Oscillator Clock Source Stable Flag (Read Only)** 0 = Internal 22.1184 MHz high speed oscillator clock is not stable or disabled. 1 = Internal 22.1184 MHz high speed oscillator clock is stable and enabled.
[3]	**OSC10K_STB**	**Internal 10 KHz Low Speed Oscillator Clock Source Stable Flag (Read Only)** 0 = Internal 10 kHz low speed oscillator clock is not stable or disabled. 1 = Internal 10 kHz low speed oscillator clock is stable and enabled.
[2]	**PLL_STB**	**Internal PLL Clock Source Stable Flag (Read Only)** 0 = Internal PLL clock is not stable or disabled. 1 = Internal PLL clock is stable in normal mode.
[1]	**Reserved**	Reserved.
[0]	**XTL12M_STB**	**External 4~24 MHz High Speed Crystal Clock Source Stable Flag (Read Only)** 0 = External 4~24 MHz high speed crystal clock is not stable or disabled. 1 = External 4~24 MHz high speed crystal clock is stable and enabled.

Clock Source Select Control Register 0 (CLKSEL0)

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
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 (Write Protect)** If SYST_CSR[2]= 0, SysTick uses clock source listed below. 000 = Clock source from external 4~24 MHz high speed crystal clock. 010 = Clock source from external 4~24 MHz high speed crystal clock/2. 011 = Clock source from HCLK/2. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock/2. **Note:** These bits are write protected bit. Refer to the REGWRPROT register.
[2:0]	**HCLK_S**	**HCLK Clock Source Selection (Write Protect)** 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. 000 = Clock source from external 4~24 MHz high speed crystal clock. 001 = Clock source from PLL clock/2. 010 = Clock source from PLL clock. 011 = Clock source from internal 10 kHz low speed oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock. **Note1:** Before clock switching, the related clock sources (both pre-select and new-select) must be turn on. **Note2:** These bits are write protected bit. Refer to the REGWRPROT register.

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[1:0]		PWM01_S[1:0]		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	SPI2_S	SPI1_S	SPI0_S	ADC_S		WDT_S	

Bits	Description	
[31:30]	**PWM23_S[1:0]**	**PWM2 and PWM3 Clock Source Select Bit [1:0]** PWM2 and PWM3 use the same peripheral clock source, and both of them use the same prescaler. The peripheral clock source of PWM2 and PWM3 is defined by PWM23_S[2:0] and this field is combined by CLKSEL2[9] and CLKSEL1[31:30]. 000 = Clock source from external 4~24 MHz high speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from internal 22.1184 MHz high speed oscillator clock. 111 = Clock source from internal 10 kHz low speed oscillator clock.
[29:28]	**PWM01_S[1:0]**	**PWM0 and PWM1 Clock Source Select Bit [1:0]** PWM0 and PWM1 use the same peripheral clock source, and both of them use the same prescaler The peripheral clock source of PWM0 and PWM1 is defined by PWM01_S[2:0] and this field is combined by CLKSEL2[8] and CLKSEL1[29:28]. 000 = Clock source from external 4~24 MHz high speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from internal 22.1184 MHz high speed oscillator clock. 111 = Clock source from internal 10 kHz low speed oscillator clock.
[27:26]	**Reserved**	Reserved.
[25:24]	**UART_S**	**UART Clock Source Selection** 00 = Clock source from external 4~24 MHz high speed crystal clock. 01 = Clock source from PLL clock. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock.
[23]	**Reserved**	Reserved.

[22:20]	**TMR3_S**	**TIMER3 Clock Source Selection** 000 = Clock source from external 4~24 MHz high speed crystal clock. 010 = Clock source from HCLK. 011 = Reserved. 101 = Clock source from internal 10 kHz low speed oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock.
[19]	**Reserved**	Reserved.
[18:16]	**TMR2_S**	**TIMER2 Clock Source Selection** 000 = Clock source from external 4~24 MHz high speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from external clock source TM2. 101 = Clock source from internal 10 kHz low speed oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock.
[15]	**Reserved**	Reserved.
[14:12]	**TMR1_S**	**TIMER1 Clock Source Selection** 000 = Clock source from external 4~24 MHz high speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from external clock source TM1. 101 = Clock source from internal 10 kHz low speed oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock.
[11]	**Reserved**	Reserved.
[10:8]	**TMR0_S**	**TIMER0 Clock Source Selection** 000 = Clock source from external 4~24 MHz high speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from external clock source TM0. 101 = Clock source from internal 10 kHz low speed oscillator clock. 111 = Clock source from internal 22.1184 MHz high speed oscillator clock.
[7]	**Reserved**	Reserved.
[6]	**SPI2_S**	**SPI2 Clock Source Selection** 0 = Clock source from PLL clock. 1 = Clock source from HCLK.
[5]	**SPI1_S**	**SPI1 Clock Source Selection** 0 = Clock source from PLL clock. 1 = Clock source from HCLK.
[4]	**SPI0_S**	**SPI0 Clock Source Selection** 0 = Clock source from PLL clock. 1 = Clock source from HCLK.
[3:2]	**ADC_S**	**ADC Clock Source Selection** 00 = Clock source from external 4~24 MHz high speed crystal clock. 01 = Clock source from PLL clock. 10 = Clock source from HCLK. 11 = Clock source from internal 22.1184 MHz high speed oscillator clock.

[1:0]	**WDT_S**	**Watchdog Timer Clock Source Selection (Write Protect)** 10 = Clock source from HCLK/2048 clock. 11 = Clock source from internal 10 kHz low speed oscillator clock. **Note:** These bits are write protected bit. Refer to the REGWRPROT register.

Clock Source Select Control Register 2 (CLKSEL2)

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

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**WWDT_S**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**PWM23_S[2]**	**PWM01_S[2]**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**FRQDIV_S**		**I2S_S**	

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**WWDT_S**	**Windowed-watchdog Timer Clock Source Selection** 10 = Clock source from HCLK/2048 clock. 11 = Clock source from internal 10 kHz low speed oscillator clock.
[15:10]	**Reserved**	Reserved.
[9]	**PWM23_S[2]**	**PWM2 and PWM3 Clock Source Select Bit [2]** PWM2 and PWM3 use the same peripheral clock source, and both of them use the same prescaler. The peripheral clock source of PWM2 and PWM3 is defined by PWM23_S[2:0] and this field is combined by CLKSEL2[9] and CLKSEL1[31:30]. 000 = Clock source from external 4~24 MHz high speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from internal 22.1184 MHz high speed oscillator clock. 111 = Clock source from internal 10 kHz low speed oscillator clock.
[8]	**PWM01_S[2]**	**PWM0 and PWM1 Clock Source Select Bit [2]** PWM0 and PWM1 use the same peripheral clock source, and both of them use the same prescaler. The peripheral clock source of PWM0 and PWM1 is defined by PWM01_S[2:0] and this field is combined by CLKSEL2[8] and CLKSEL1[29:28]. 000 = Clock source from external 4~24 MHz high speed crystal clock. 010 = Clock source from HCLK. 011 = Clock source from internal 22.1184 MHz high speed oscillator clock. 111 = Clock source from internal 10 kHz low speed oscillator clock.
[7:4]	**Reserved**	Reserved.
[3:2]	**FRQDIV_S**	**Clock Divider Clock Source Selection** 00 = Clock source from external 4~24 MHz high speed crystal clock.

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

Clock Divider Number Register (CLKDIV)

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

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

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

PLL Control Register (PLLCON)

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

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

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19]	**PLL_SRC**	**PLL Source Clock Selection** 0 = PLL source clock from external 4~24 MHz high speed crystal. 1 = PLL source clock from internal 22.1184 MHz high speed oscillator.
[18]	**OE**	**PLL OE (FOUT Enable) 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 set to 1 in PWRCON register, the PLL will also enter Power-down mode. 0 = PLL is in Normal mode. 1 = PLL is in Power-down mode (default).
[15:14]	**OUT_DV**	**PLL Output Divider Control Bits** Refer to the formulas below the table.
[13:9]	**IN_DV**	**PLL Input Divider Control Bits** Refer to the formulas below the table.
[8:0]	**FB_DV**	**PLL Feedback Divider Control Bits** Refer to the formulas below the table.

Output Clock Frequency Setting:

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

Constraint:

1. $4MHz < FIN < 24MHz$
2. $800KHz < Fref = \frac{FIN}{2 \times NR} < 8MHz$
3. $100MHz < FCO = Fref \times 2 \times NF = FIN \times \frac{NF}{NR} < 200MHz$

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

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 Bit** 0 = Frequency Divider Disabled. 1 = Frequency Divider Enabled.
[3:0]	**FSEL**	**Divider Output Frequency Select Bits** The formula of output frequency is: $\mathbf{F_{out} = F_{in}/2^{(N+1)}}$. F_{in} is the input clock frequency. F_{out} is the frequency of divider output clock. N is the 4-bit value of FSEL[3:0].

APB Divider Control Register (APBDIV)

Register	Offset	R/W	Description	Reset Value
APBDIV	CLK_BA+0x2C	R/W	APB 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							**APBDIV**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**APBDIV**	**APB Divider Enable Bit** 0 = PCLK equal to HCLK. 1 = PCLK equal to HCLK / 2.

6.4 Flash Memory Controller (FMC)

6.4.1 Overview

The NuMicro® NUC123 series is equipped with 68/36 Kbytes on-chip embedded flash for application program memory (APROM) and Data Flash, and 4 Kbytes for ISP loader program memory (LDROM) that could be programmed boot loader to update APROM and Data Flash through In-System-Programming (ISP) procedure. The ISP function enables user to update embedded flash when chip is soldered on PCB. After chip is powered on, Cortex®-M0 CPU fetches code from APROM or LDROM decided by boot select (CBS (Config0[7:6]). User can also select to enable or disable In-Application-Programming (IAP) function through boot select (CBS (Config0[7:6]). Also, the NUC123 provides Data Flash for user, to store some application dependent data before chip is powered off.

6.4.2 Features

- Runs up to 72 MHz and optional up to 50 MHz with zero wait state for continuous address read access
- Supports 68/36 KB application program ROM (APROM)
- Supports 4KB loader ROM (LDROM)
- Supports Data Flash with configurable memory size
- Supports 8 bytes User Configuration block to control system initiation
- Supports 512 bytes page erase for all embedded flash
- Supports In-System-Programming (ISP) / In-Application-Programming (IAP) to update embedded flash memory

6.4.3 Block Diagram

The flash memory controller (FMC) consist of AHB slave interface, ISP control logic and flash macro interface timing control logic. The block diagram of flash memory controller is shown as follows.

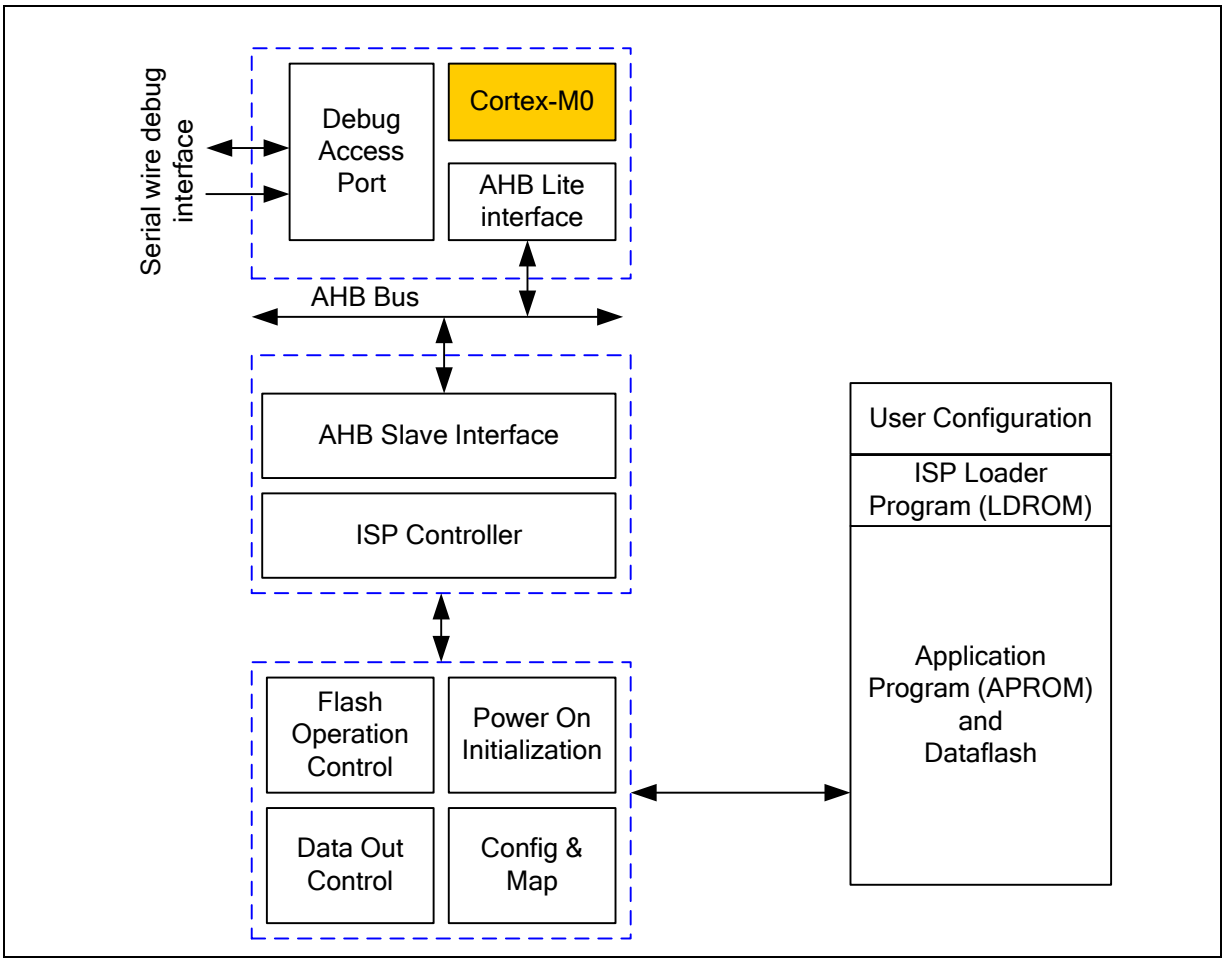


Figure 6-15 Flash Memory Control Block Diagram

6.4.4 Functional Description

6.4.4.1 *Memory Organization*

The flash memory consists of application program memory (APROM), Data Flash, ISP loader program memory (LDROM) and user configuration. User configuration block provides two words to control system logic, such as flash security lock, boot select, Brown-out voltage level, Data Flash base address, etc. It works like a fuse for power on setting. It is loaded from flash memory to its corresponding control registers during chip powered on. User can set these bits according to different application request. The Data Flash is fixed as 4 Kbytes size if DFVSEN (Config0[2]) is 1. User can also set Data Flash as a variable size by setting DFVSEN (Config0[2]) to 0. If Data Flash is configured as variable size, the start address and its size can be defined by DFBADR (Config1) and DFEN (Config0[0]).

Block Name	Size	Start Address	End Address
APROM	64 KB 32 KB	0x0000_0000	0x0000_FFFF (64 KB) 0x0000_7FFF (32 KB)
Data Flash	4 KB	0x0001_F000	0x0001_FFFF
LDROM	4 KB	0x0010_0000	0x0010_0FFF
User Configuration	2 words	0x0030_0000	0x0030_0007

Table 6-13 Memory Address Map (DFVSEN = 1)

Block Name	Size	Start Address	End Address
AP-ROM	68KB 36KB (68-0.5*N) or (36-0.5*N) KB	0x0000_0000	0x0001_0FFF (68KB, if DFEN=1) 0x0000_8FFF (36KB, if DFEN=1) DFBADR-1 (if DFEN=0)
Data Flash	0.5*N KB (if DFEN=0)	DFBADR (if DFEN=0)	0x0001_0FFF (68KB, if DFEN=0) 0x0000_8FFF (36KB, if DFEN=0)
LD-ROM	4 KB	0x0010_0000	0x0010_0FFF
User Configuration	2 words	0x0030_0000	0x0030_0007

Note: N is the page number of configured Data Flash. One page size is 512 bytes.

Table 6-14 Memory Address Map (DFVSEN = 0)

The flash memory organization is shown below:

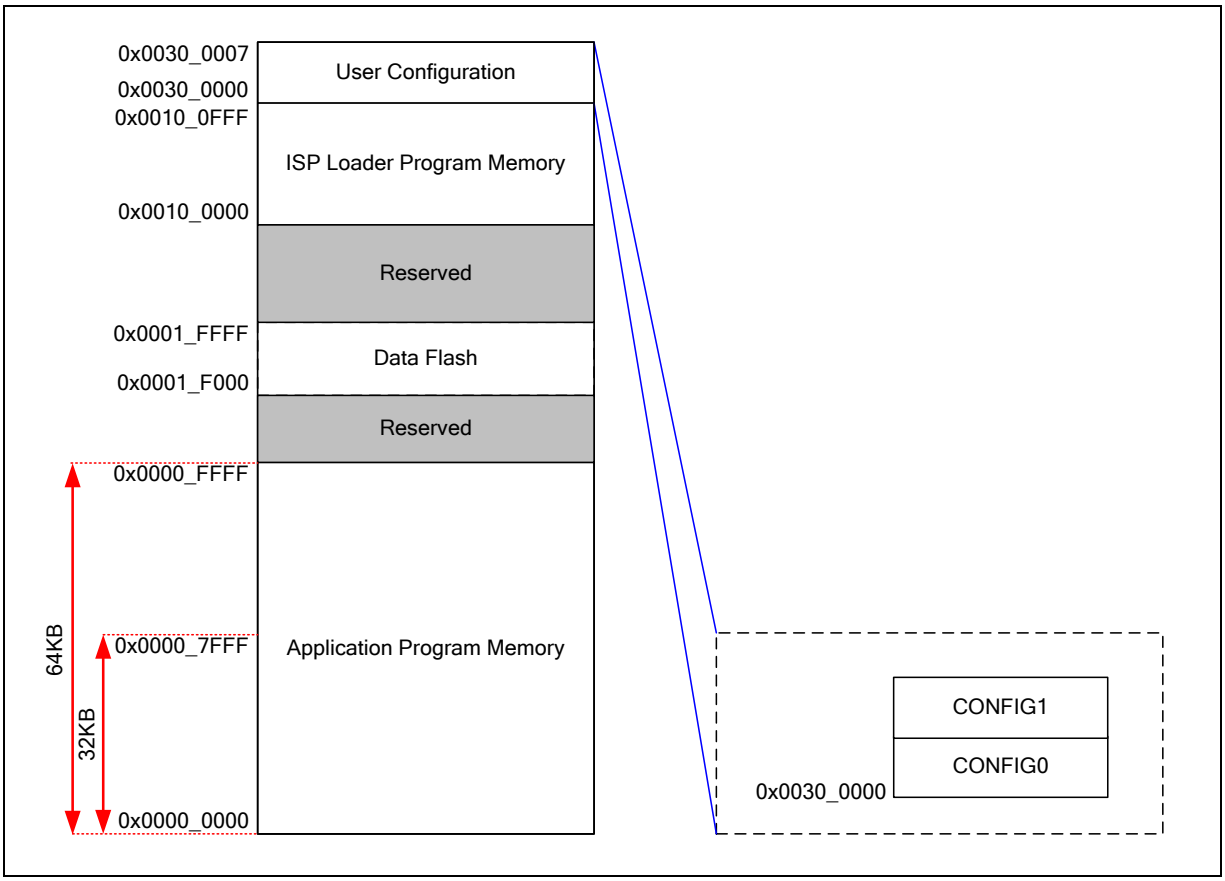


Figure 6-16 Flash Memory Organization (DFVSEN = 1)

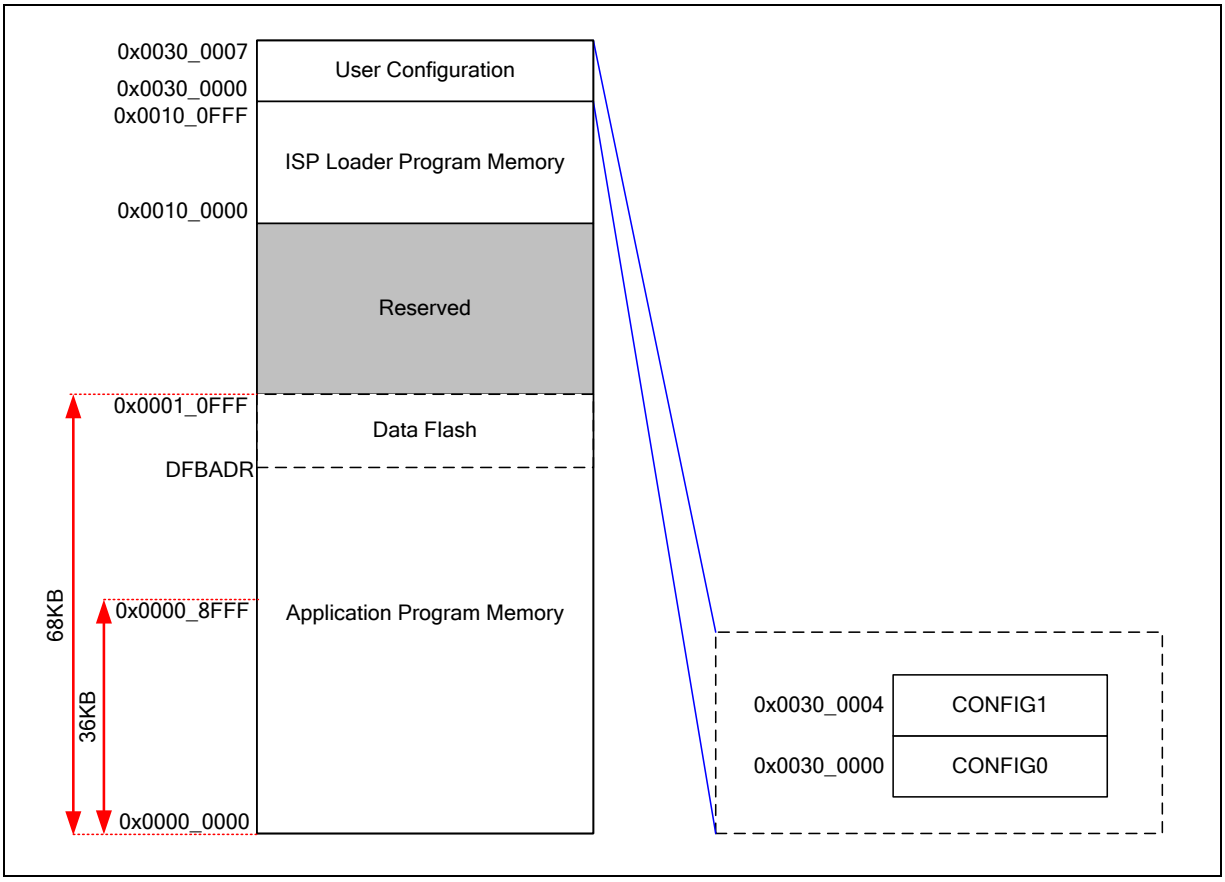


Figure 6-17 Flash Memory Organization (DFVSEN = 0)

6.4.4.2 *Boot Selection*

The NuMicro® NUC123 Series provides four booting sources for user to select, including LDROM with IAP, LDROM without IAP, APROM with IAP, and APROM without IAP. In any time, the booting source and system memory map are setting by CBS (CONFIG0[7:6]).

CBS	Boot Selection/System Memory Map	Vector Mapping Supporting
00b	LDROM with IAP mode.	Yes
01b	LDROM without IAP mode.	No
10b	APROM with IAP mode.	Yes
11b	APROM without IAP mode.	No

Table 6-15 Boot Selection Define Table

6.4.4.3 *In-Application-Programming (IAP)*

The NuMicro® NUC123 Series provides a new In-application-programming (IAP) function for user to switch the code executing between APROM and LDROM without a reset. User can enable the IAP function by re-booting chip and setting CBS (Config0[7:6]) as 10b or 00b.

When NUC123 boots from APROM with the IAP function enabled (CBS[1:0] = 10b), the executable range of code includes all of APROM and LDROM. The address space of APROM is kept as the original size but the address space of the 4 KB LDROM is mapped to 0x0010_0000~ 0x0010_0FFF.

When NUC123 boots from LDROM with the IAP function enabled (CBS[1:0] = 00b), the executable range of code includes all of LDROM and almost all of APROM except 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 still mapped to 0x0010_0000~0x0010_0FFF.

Please refer to the Figure 6-18 for the address map while IAP is activated.

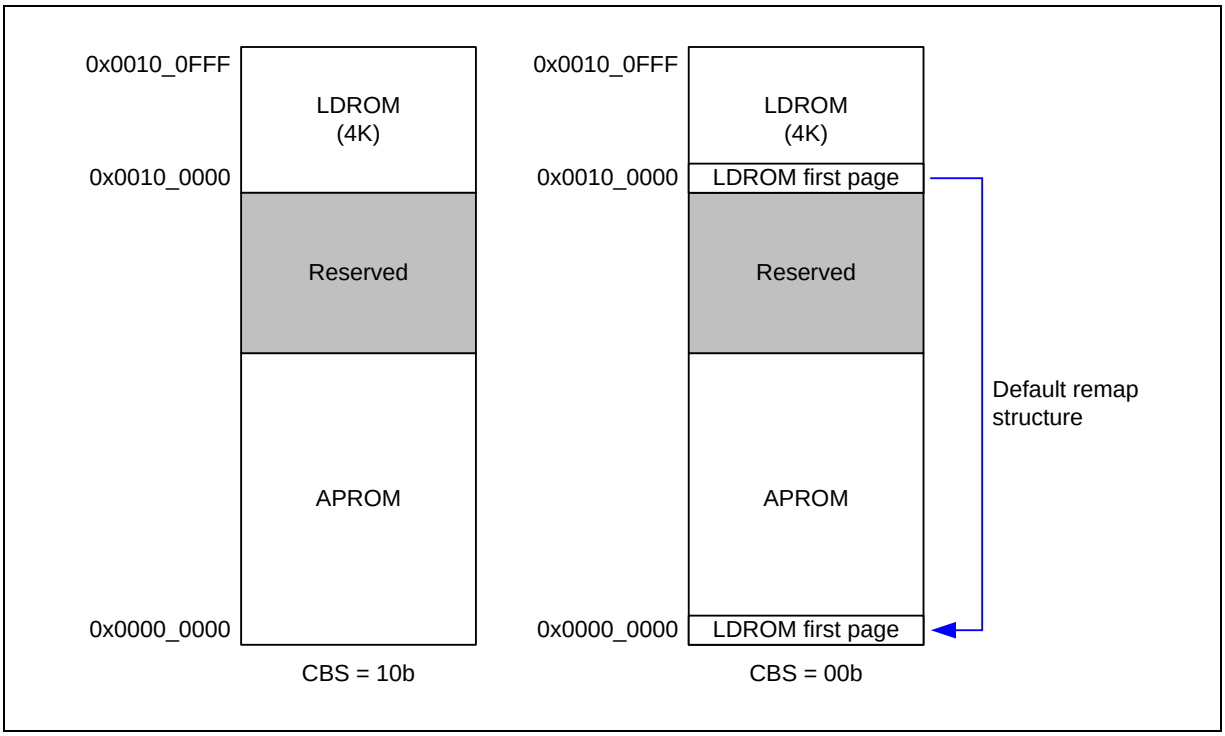


Figure 6-18 Executable Range of Code with IAP Function Enabled

When chip boots with the IAP function enabled, any other page within the executable range of code can be mirrored to the first page of executable code (0x0000_0000~0x0000_01FF) any time. User can change the remap address of the first executing page by filling the target remap address to ISPADR and then go through ISP procedure with the Vector Page Re-map command. After changing the remap address, user can check if the change is successful by reading the VECMAP (ISPSTA[20:9]). For NUC123xxxANx, VECMAP (ISPSTA[20:9]) will be reset by any reset source. For NUC123xxxAEx, VECMAP (ISPSTA[20:9]) will be reset by any reset source except software SYS reset happened (SYSRESETREQ (AIRCR[2]) = 1) and CPU reset happened (CPU_RST (IPRSTC1[1]) = 1).

6.4.4.4 *In System Program (ISP)*

A dedicated 4 KB ISP loader program memory (LDROM) is used to store ISP firmware. User can select to start program fetch from APROM or LDROM based on CBS[1] (Config0[7]).

In addition to set boot from APROM or LDROM, CBS (Config0[7:6]) also used to control system memory map after booting. When CBS = 11b to boot from APROM without IAP, the application in APROM will not be able to access LDROM by memory read. In other words, when CBS = 01b to boot from LDROM without IAP, the software executed in LDROM will not be able to access APROM by memory read. Figure 6-19 shows the memory map when boot from APROM and LDROM.

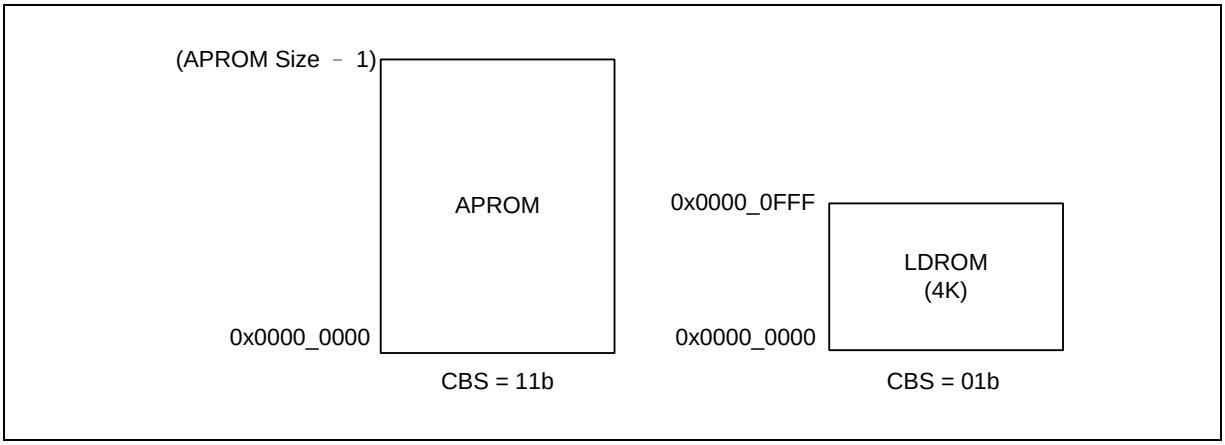


Figure 6-19 Program Executing Range for boot from APROM and boot from LDROM

The application program memory and Data Flash supports in system programming (ISP). NuMicro® NUC123 supports ISP mode allowing a device to be reprogrammed under software control. Furthermore, the capability to update the application firmware makes wide range of applications possible.

ISP is performed without removing the microcontroller from the system. Various interfaces enable LDROM firmware to get new program code easily. The most common method to perform ISP is via UART along with the firmware in LDROM. General speaking, PC transfers the new APROM code through serial port. Then LDROM firmware receives it and re-programs into APROM through ISP commands. The ISP firmware and PC application program for NuMicro® NUC123 Series enables user to easily perform ISP through Nuvoton ISP tool.

ISP Mode	ISPCMD	ISPADR	ISPDAT
FLASH Page Erase	0x22	Valid address of flash memory organization. It must be 512 bytes page alignment.	N/A

FLASH Program	0x21	Valid address of flash memory organization	Programming Data
FLASH Read	0x00	Valid address of flash memory organization	Return Data
Read Company ID	0x0B	0x0000_0000	0x0000_00DA
Read Unique ID	0x04	0x0000_0000	Unique ID Word 0
		0x0000_0004	Unique ID Word 1
		0x0000_0008	Unique ID Word 2
Vector Remap	0x2E	Valid address in APROM,LDROM or boot loader It must be 512 bytes alignment	N/A

Table 6-16 ISP Mode Command

6.4.4.4.1 ISP Procedure

The NuMicro® NUC123 Series supports booting from APROM or LDROM initially defined by user configuration bits (CBS). If user wants to update application program in APROM without IAP, he can write BS=1 and uses software reset to make chip boot from LDROM. The first step to start ISP function is write ISPEN bit to 1. Software is required to write REGWRPROT register in Global Control Register (GCR, 0x5000_0100) with 0x59, 0x16 and 0x88 before writing ISPCON register. This procedure is used to protect flash memory from destroying owning to unintended write during power on/off duration.

Several error conditions are checked after software sets ISPGO(ISPTRG[0]) to 1. If error condition occurs, ISP operation is not been started and ISPFF (ISPSTA[6]) will be set. ISPFF (ISPSTA[6]) is cleared by software, it will not be over written in next ISP operation. The next ISP procedure can be started even ISPFF (ISPSTA[6]) keeps at 1. It is recommended that software to check ISPFF (ISPSTA[6]) and clear it after each ISP operation.

When ISPGO (ISPTRG[0]) is set, CPU will wait for ISP operation finish, during this period; peripheral still keeps working as usual. If any interrupt request occur, CPU will not service it till ISP operation finish. When ISP operation is finished, the ISPGO (ISPTRG[0]) will be cleared by hardware automatically. User can know if ISP operation is finished by checking this bit. User should add ISB instruction next to the instruction which set 1 to ISPGO (ISPTRG[0]) to ensure correct execution of the instructions following ISP operation.

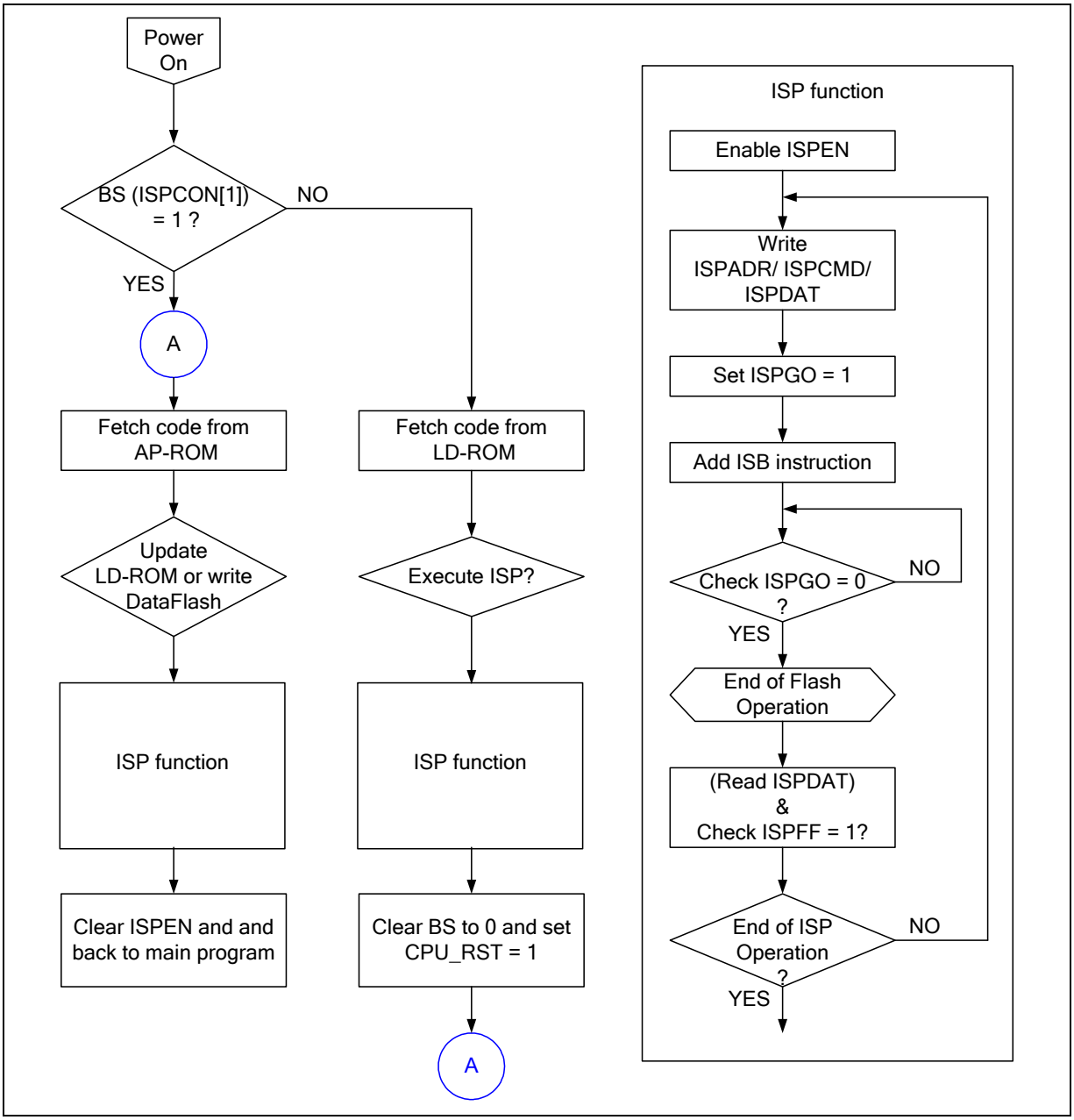


Figure 6-20 ISP Procedure Example

6.4.4.5 *Data Flash*

The NuMicro® NUC123 provides Data Flash for user to store data, which is read or written through ISP procedure. The size of each erase unit is 512 bytes. When a word will be changed, all 128 words need to be copied to another page or SRAM in advance.

When DFVSEN (Config0[2]) is set 1, the application program memory (APROM) is 64/32 Kbytes and the Data Flash size is 4 Kbytes. The start address of Data Flash is fixed at 0x0001_F000. When DFVSEN (Config0[2]) is set to 0, the application program memory (APROM) is 68/36 Kbytes and the Data Flash is shared with APROM with variable size defined by user. If DFEN (Config0[0]) is set to 1, there is no Data Flash and all 68/36 Kbytes size is used for APROM. If DFEN (Config0[0]) is set to 0, the Data Flash share with APROM and its base address is defined by DFBADR (Config1[19:0]). Under this setting, the application program memory size is (68/36-0.5*N)KB and Data Flash size is 0.5*N KB.

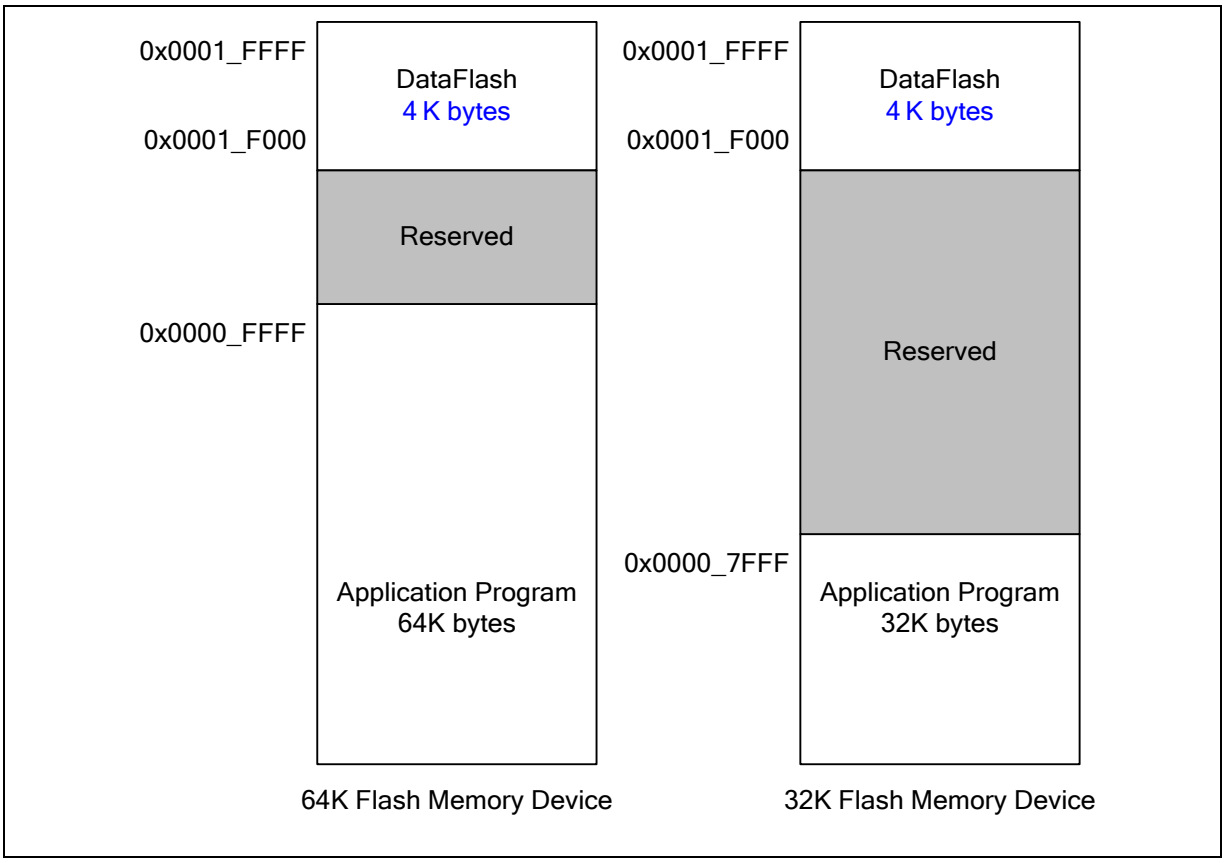


Figure 6-21 Flash Memory Structure (DFVSEN = 1)

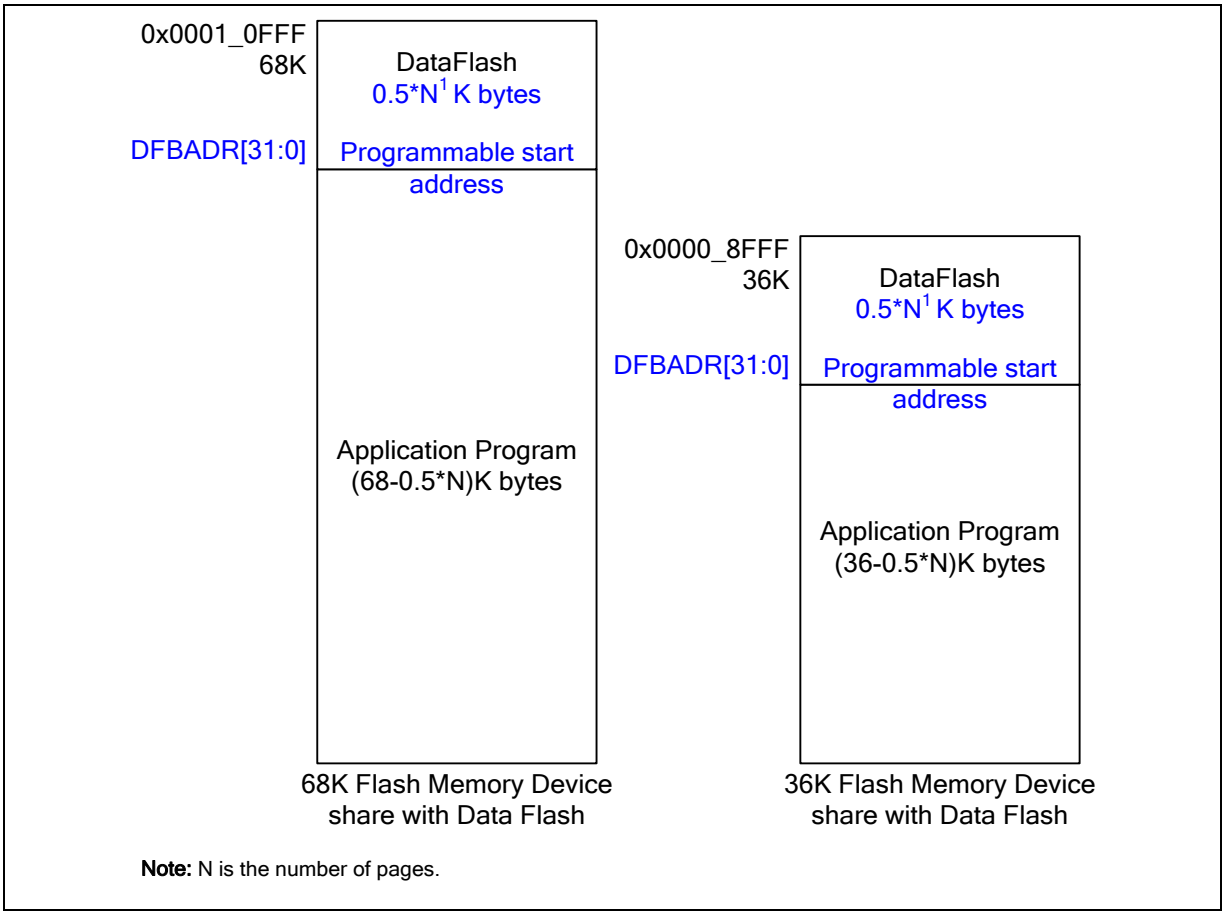


Figure 6-22 Flash Memory Structure (DFVSEN = 0)

6.4.4.6 *User Configuration*

Config0 (Address = 0x0030_0000)

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

Bits	Description	
[31]	**CWDTEN**	**Watchdog Hardware Enable Bit** When watchdog timer hardware enable function is enabled, the watchdog enable bit WTE (WTCR[7]) and watchdog reset enable bit WTRE (WTCR[1]) is set to 1 automatically after power on. The clock source of watchdog timer is force at LIRC and LIRC can't be disabled. 0 = WDT hardware enable function is active. WDT clock is always on except chip enters Power- down mode. When chip enter Power-down mode, WDT clock is always on if CWDTPDEN is 0 or WDT clock is controlled by OSC10K_EN (PWRCON[3]) if CWDTPDEN is 1. Please refer to bit field description of CWDTPDEN. 1 = WDT hardware enable function is inactive.
[30]	**CWDTPDEN**	**Watchdog Clock Power-down Enable Bit** 0 = Watchdog Timer clock kept enabled when chip enters Power-down. 1 = Watchdog Timer clock is controlled by OSC10K_EN (PWRCON[3]) when chip enters Power-down. **Note:** This bit only works if CWDTEN is set to 0
[29:28]	**Reserved**	Reserved.
[27]	**CGPFMFP**	**GPF Multi-function Selection** 0 = PF.0 & PF.1 pins are configured as GPIO function. 1 = PF.0 & PF.1 pins are used as external 4~24MHz crystal oscillator pin. **Note1:** For NUC123xxxANx, PF.0 and PF.1 multi-function is controlled by CGPFMFP (Config0[27]) when power-up, user can change PF.0 and PF.1 multi-function by writing GPF_MFP0 (GPF_MFP[1]) and GPF_MFP1 (GPF_MFP[0]). **Note2:** For NUC123xxxAEx, PF.0 and PF.1 multi-function can only be controlled by CGPFMFP (Config0[27]).
[26:24]	**CFOSC**	**CPU Clock Source Selection After Reset** The value of CFOSC will be loaded to CLKSEL0.HCLK_S[2:0] in system register after any reset occurs except CPU reset. 000 = 4 ~ 24 MHz external high speed crystal oscillators (HXT). 111 = 22.1184 MHz internal high speed RC oscillator (HIRC). Others = Reserved.
[23]	**CBODEN**	**Brown-out Detector Enable Bit** 0= Brown-out detect Enabled after powered on. 1= Brown-out detect Disabled after powered on.

[22:21]	**CBOV1-0**	**Brown-out Voltage Selection**For NUC123xxxAN 00 = Brown-out voltage is 2.2V. 01 = Brown-out voltage is 2.7V. 10 = Brown-out voltage is 3.8V. 11 = Brown-out voltage is 4.5V. For NUC123xxxAE 00 = Brown-out voltage is 2.2V. 01 = Brown-out voltage is 2.7V. 10 = Brown-out voltage is 3.7V. 11 = Brown-out voltage is 4.4V.
[20]	**CBORST**	**Brown-out Reset Enable Bit** 0 = Brown-out reset Enabled after powered on. 1 = Brown-out reset Disabled after powered on.
[19:8]	**Reserved**	Reserved.
[10]	**CIOINI**	**IO 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. **Note1:** This configuration is only supported for NUC123xxxAEx **Note2:** It is recommended to use 100 kΩ pull-up resistor on both ICE_DAT and ICE_CLK pin.
[9:8]	**Reserved**	Reserved.
[7:6]	**CBS**	**Chip Boot Selection** When CBS[0] = 0, the LDROM base address is mapping to 0x100000 and APROM base address is mapping to 0x0. User could access both APROM and LDROM without boot switching. In other words, if IAP mode is supported, the code in LDROM and APROM can be called by each other. 00 = Boot from LDROM with IAP mode. 01 = Boot from LDROM without IAP mode. 10 = Boot from APROM with IAP mode. 11 = Boot from APROM without IAP mode. **Note:** BS (ISPCON[1]) is only used to control boot switching when IAP mode is disabled and VECMAP (ISPSTA[20:9]) is only used to remap 0x0~0x1ff when IAP mode is enabled.
[5:3]	**Reserved**	Reserved.
[2]	**DFVSEN**	**DATA Flash Variable Size Enable Bit** 0 = Data flash size is variable and it base address is based on DFBADR (Config1). 1 = Data flash size is fixed as 4 Kbytes. **Note:** For NUC123xxxANx, CPU cannot read Data Flash directly if DFVSEN (Config0[2]) is set to 1, and the Data Flash can only be read through ISP procedure.
[1]	**LOCK**	**Security Lock Control** 0 = Flash memory content is locked. 1 = Flash memory content is not locked.
[0]	**DFEN**	**Data Flash Enable Bit** 0 = Data flash Enabled. 1 = Data flash Disabled. **Note:** This bit only works if DFVSEN is set to 0. When DFVSEN is 0 and DFEN is 1, there is no Data Flash and APROM size is 68 Kbytes. When DFVSEN is 0 and DFEN is 0, the Data Flash is shared with APROM within 68 Kbytes, and the base address of Data Flash is decided by DFBADR (Config1).

Config1 (Address = 0x0030_0004)

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:0]	**DFBADR**	**Data Flash Base Address (this Register Works Only When DFEN Set to 0)** If DFEN is set to 0, the Data Flash base address is defined by user. Since on-chip flash erase unit is 512 bytes, it is mandatory to keep bit 8-0 as 0.

6.4.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
Base Address (FMC_BA) : 0x5000_C000				
ISPCON	FMC_BA+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 Register	0x0000_0000
DFBADR	FMC_BA+0x14	R	Data Flash Start Address (DFVSEN = 1)	0x0001_F000
DFBADR	FMC_BA+0x14	R	Data Flash Start Address (DFVSEN = 0)	CONFIG1
FATCON	FMC_BA+0x18	R/W	Flash Access Window Control Register	0x0000_0000
ISPSTA	FMC_BA+0x40	R	ISP Status Register	0x0000_0000

6.4.6 Register Description

ISP Control Register (ISPCON)

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

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is set by hardware when a triggered ISP meets any of the following conditions: (1) APROM writes to itself if APUEN is set to 0. (2) LDROM writes to itself if LDUEN is set to 0. (3) CONFIG is erased/programmed if CFGUEN is set to 0. (4) Page Erase command at LOCK mode with ICE connection (5) Erase or Program command at brown-out detected (6) Destination address is illegal, such as over an available range. (7) Invalid ISP commands Write 1 to clear. **Note:** This bit can be cleared by software writing '1'.
[5]	**LDUEN**	**LDROM Update Enable Bit (Write Protect)** LDROM update enable bit. 0 = LDROM cannot be updated. 1 = LDROM can be updated.
[4]	**CFGUEN**	**CONFIG Update Enable Bit (Write Protect)** 0 = CONFIG cannot be updated. 1 = CONFIG can be updated.
[3]	**APUEN**	**APROM Update Enable Bit (Write Protect)** 0 = APROM cannot be updated when the chip runs in APROM. 1 = APROM can be updated when the chip runs in APROM.
[2]	**Reserved**	Reserved.

[1]	**BS**	**Boot Select (Write Protect)** Set/clear this bit to select next booting from LDROM/APROM without IAP, respectively. This bit also functions as chip booting status flag, which can be used to check where chip booted from. This bit is initiated with the inversed value of CBS in Config0 after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened 0 = Boot from APROM. 1 = Boot from LDROM.
[0]	**ISPEN**	**ISP Enable Bit (Write Protect)** ISP function enable bit. Set this bit to enable ISP function. 0 = ISP function Disabled. 1 = ISP function Enabled.

ISP Address (ISPADR)

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

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

Bits	Description	
[31:0]	**ISPADR**	**ISP Address** The NuMicro® NUC123 series is equipped with an embedded flash, and it supports word program only. ISPADR[1:0] must be kept 00b for ISP operation.

ISP Data Register (ISPDAT)

Register	Offset	R/W	Description	Reset Value
ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000

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

Bits	Description	
[31:0]	**ISPDAT**	**ISP Data** Write data to this register before ISP program operation. Read data from this register after ISP read operation.

ISP Command (ISPCMD)

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

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**CMD**	**ISP Command** ISP command table is shown below: 0x00= FLASH Read. 0x04= Read Unique ID. 0x0B= Read Company ID. 0x21= FLASH Program. 0x22= FLASH Page Erase. 0x2E= Vector Remap. The other commands are invalid.

ISP Trigger Control Register (ISPTRG)

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

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**ISPGO**	**ISP Start Trigger (Write Protect)** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. 0 = ISP operation is finished. 1 = ISP is progressed. **Note:** To make sure ISP function has been finished before CPU goes ahead, ISB (Instruction Synchronization Barrier) instruction is used right after ISPGO (ISPTRG[0]) setting.

Data Flash Base Address Register (DFBADR)

Register	Offset	R/W	Description	Reset Value
DFBADR	FMC_BA+0x14	R	Data flash Base Address	0x0001_F000 CONFIG1

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

Bits	Description	
[31:0]	**DFBADR**	**Data Flash Base Address (Read Only)** This register indicates Data Flash start address. It is a read only register. When DFVSEN is set to 0, the Data Flash is shared with APROM. The Data Flash size is defined by user configuration and the content of this register is loaded from Config1. When DFVSEN is set to 1, the Data Flash size is fixed as 4K and the start address can be read from this register is fixed at 0x0001_F000.

Flash Access Time Control Register (FATCON)

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

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**MFOM**	**Middle Frequency Optimization Mode (Write Protect)** When chip operation frequency is lower than 50 MHz, chip can work more efficiently by setting this bit to 1 0 = Middle Frequency Optimization mode Disabled. 1 = Middle Frequency Optimization mode Enabled.
[5]	**Reserved**	Reserved.
[4]	**LFOM**	**Low Frequency Optimization Mode (Write Protect)** When chip operation frequency is lower than 25 MHz, chip can work more efficiently by setting this bit to 1 0 = Low Frequency Optimization mode Disabled. 1 = Low Frequency Optimization mode Enabled.
[3:0]	**Reserved**	Reserved.

ISP Status Register (ISPSTA)

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

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

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20:9]	**VECMAP**	**Vector Page Mapping Address (Read Only)** The current flash address space 0x0000_0000~0x0000_01FF is mapping to address {VECMAP[11:0], 9'h000} ~ {VECMAP[11:0], 9'h1FF} **Note:** The VECMAP can keep value when software SYS reset happened (SYSRESETREQ (AIRCR[2]) = 1) or CPU reset happened (CPU_RST (IPRSTC1[1]) = 1) (NUC123xxxAEx Only).
[8:7]	**Reserved**	Reserved.
[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is the mirror of ISPFF (ISPCON[6]), it needs to be cleared by writing 1 to ISPCON[6] or FMC_ISPSTA[6]. This bit is set by hardware when a triggered ISP meets any of the following conditions: (1) APROM writes to itself if APUEN is set to 0. (2) LDROM writes to itself if LDUEN is set to 0. (3) CONFIG is erased/programmed if CFGUEN is set to 0. (4) Page Erase command at LOCK mode with ICE connection (5) Erase or Program command at brown-out detected (6) Destination address is illegal, such as over an available range. (7) Invalid ISP commands Write 1 to clear.
[5:3]	**Reserved**	Reserved.

[2:1]	**CBS**	**Chip Boot Selection of CONFIG (Read Only)** When CBS[0] = 0, the LDROM base address is mapping to 0x100000 and APROM base address is mapping to 0x0. User could access both APROM and LDROM without boot switching. In other words, if IAP mode is supported, the code in LDROM and APROM can be called by each other. 00 = Boot from LDROM with IAP mode. 01 = Boot from LDROM without IAP mode. 10 = Boot from APROM with IAP mode. 11 = Boot from APROM without IAP mode. **Note:** BS (ISPCON[1]) is only used to control boot switching when CBS[0] = 1. VECMAP (ISPSTA[20:9]) is only used to remap 0x0~0x1ff when CBS[0] = 0.
[0]	**ISPGO**	**ISP Start Trigger** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. 0 = ISP operation is finished. 1 = ISP is progressed.

6.5 General Purpose I/O (GPIO)

6.5.1 Overview

The NuMicro® NUC123 series has up to 47 General Purpose I/O pins shared with other function pins depending on the chip configuration. These 47 pins are arranged in 5 ports named GPIOA, GPIOB, GPIOC, GPIOD and GPIOF. GPIOA has 6 pins on PA[15:10]. GPIOB has 15 pins on PB[15:12] and PB[10:0]. GPIOC has 12 pins on PC[13:8] and PC[5:0]. GPIOD has 10 pins on PD[11:8] and PD[5:0]. GPIOF has 4 pins on PF[3:0]. Each one of the 47 pins is independent and has the corresponding register bits to control the pin mode function and data.

The I/O type of each of I/O pins can be configured by software individually as input, output, open-drain or quasi-bidirectional mode. After the chip is reset, the I/O mode of all pins are depending on CIOINI (Config0[10]) (NUC123xxxAEx Only). Each I/O pin has a very weakly individual pull-up resistor which is about 110 kΩ~300 kΩ for V_{DD} is from 5.0 V to 2.5 V.

6.5.2 Features

- Four I/O modes:
  - Quasi bi-direction
  - Push-Pull output
  - Open-Drain output
  - Input only with high impendence
- TTL/Schmitt trigger input selectable by GPx_TYPE[15:0] in GPx_MFP[31:16]
- I/O pin can be configured as interrupt source with edge/level setting
- Supports High Driver and High Sink I/O mode
- Configurable default I/O mode of all pins after reset by CIOINI (Config0[10]) setting (NUC123xxxAEx Only)
  - If CIOINI (Config[10]) is 0, all GPIO pins in input tri-state mode after chip reset
  - If CIOINI (Config[10]) is 1, all GPIO pins in Quasi-bidirectional mode after chip reset
- I/O pin internal pull-up resistor enabled only in Quasi-bidirectional I/O mode
- Enabling the pin interrupt function will also enable the wake-up function

6.5.3 Block Diagram

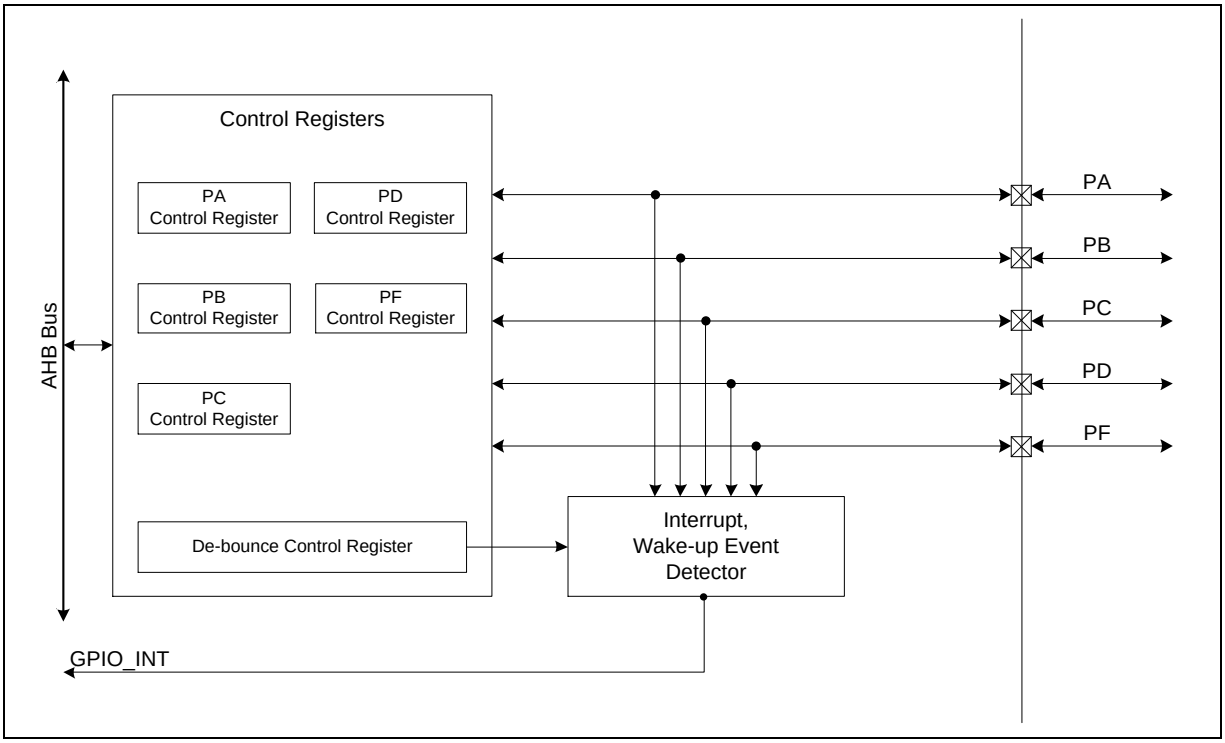


Figure 6-13 GPIO Controller Block Diagram

6.5.4 Basic Configuration

The GPIO pin functions are configured in GPA_MFP, GPB_MFP, GPC_MFP, GPD_MFP, GPF_MFP, ALT_MFP and ALT_MFP1 registers. NUC123xxxAEx provides the alternative of configuring the GPIO pins in GPA_MFPH, GPB_MFPL, GPB_MFPH, GPC_MFPL, GPC_MFPH, GPD_MFPL, GPD_MFPH and GPF_MFPL registers. (For NUC123xxxAEx, if GPx_MFPH and GPx_MFPL are used as pin multi-function setting, the GPx_MFP, ALT_MFP and ALT_MFP1 will become invalid).

6.5.5 Functional Description

6.5.5.1 *Input Mode*

Set PMDn (GPIOx_PMD [2n+1:2n]) to 00 as the GPIOx.n pin is in Input mode and the I/O pin is in tri-state (high impedance) without output drive capability. The PIN (GPIOx_PIN[n]) value reflects the status of the corresponding port pins.

6.5.5.2 *Push-pull Output Mode*

Set PMDn (GPIOx_PMD [2n+1:2n]) to 01 as GPIOx.n pin is in Output mode and the I/O pin supports digital output function with source/sink current capability. The bit value in the corresponding bit [n] of DOUT (GPIOx_DOUT[n]) is driven on the pin.

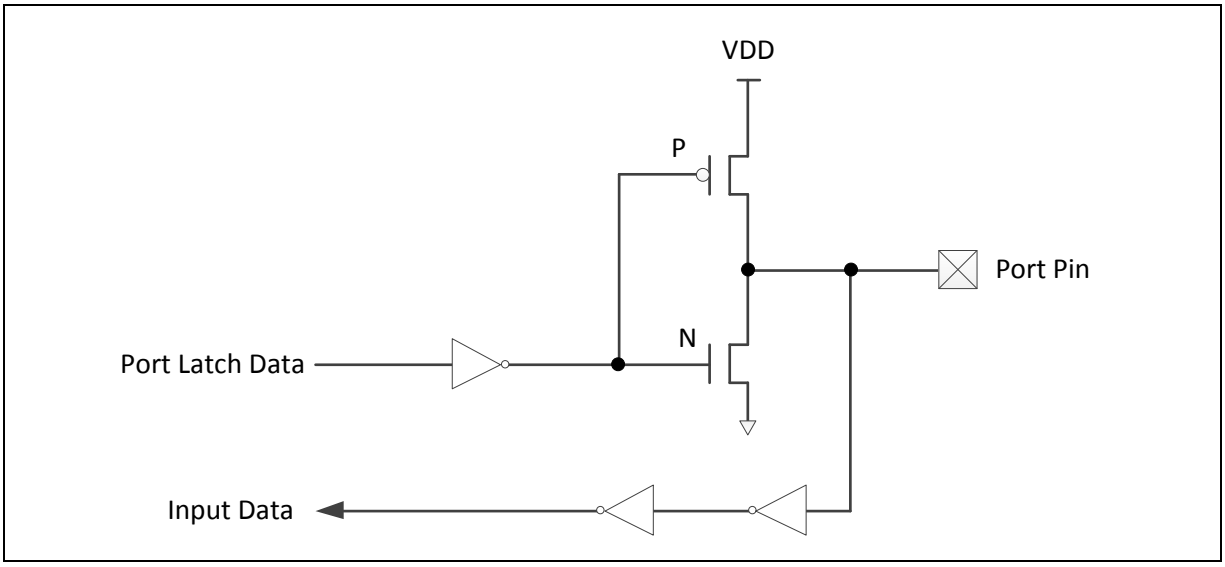


Figure 6-14 Push-Pull Output

6.5.5.3 *Open-Drain Mode*

Set (GPIOx_PMD [2n+1:2n]) to 10 as GPIOx.n pin is in Open-Drain mode and the digital output function of I/O pin supports only sink current capability, an additional pull-up register is needed for driving high state. If the bit value in the corresponding DOUT (GPIOx_DOUT[n]) bit is 0, the pin drive a "low" output on the pin. If the bit value in the corresponding DOUT (GPIOx_DOUT[n]) bit is 1, the pin output drives high that is controlled by external pull high resistor.

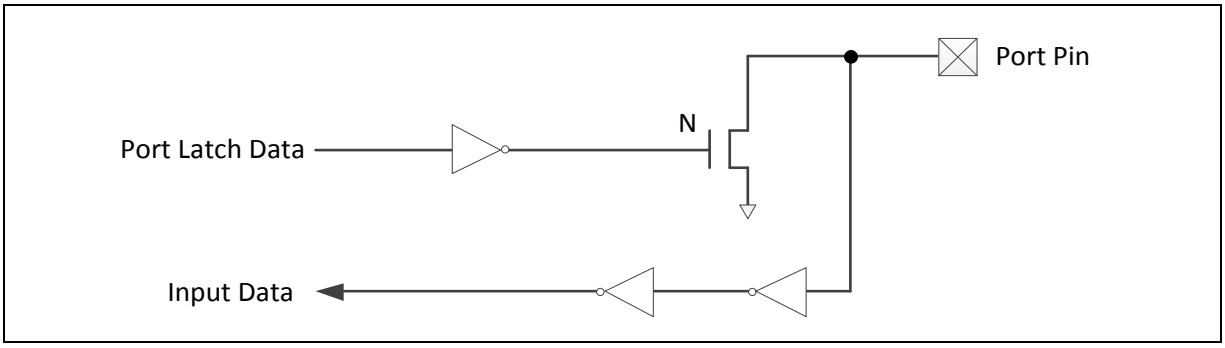


Figure 6-15 Open-Drain Output

6.5.5.4 *Quasi-bidirectional Mode*

Set (GPIOx_PMD [2n+1:2n]) to 11 as GPIOx.n pin is in Quasi-bidirectional mode and the I/O pin supports digital output and input function at the same time but the source current is only up to hundreds uA. Before the digital input function is performed the corresponding DOUT (GPIOx_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 DOUT (GPIOx_DOUT[n]) bit is 0, the pin drive a "low" output on the pin. If the bit value in the corresponding DOUT (GPIOx_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, the pin will drive strong high with 2 clock cycles on the pin and then disable the strong output drive and then the pin status is control by internal pull-up resistor. Note that the source current capability in quasi-bidirectional mode is only about 200 uA to 30 uA for V_{DD} is form 5.0 V to 2.5 V.

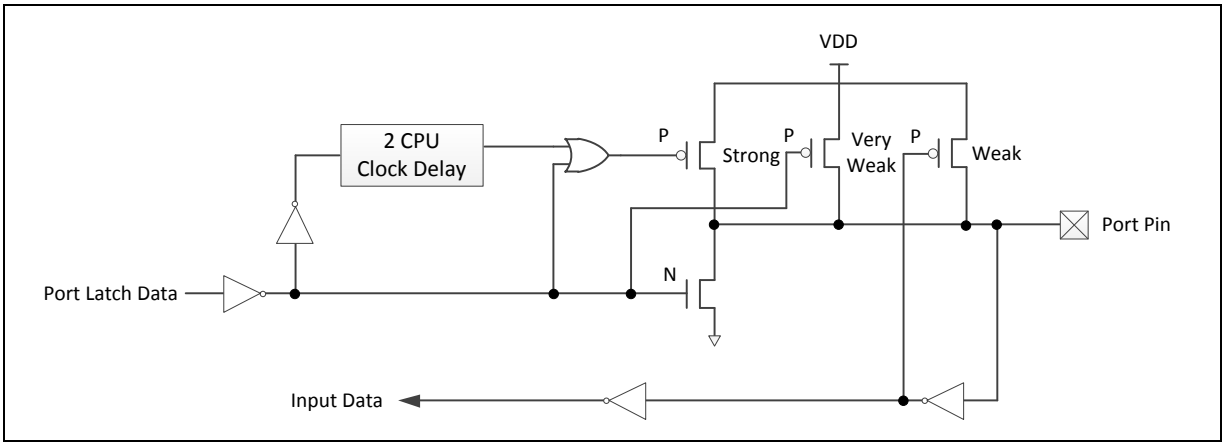


Figure 6-16 Quasi-bidirectional I/O Mode

6.5.5.5 *GPIO Interrupt and Wake-up Function*

Each GPIO pin can be set as chip interrupt source by setting correlative IR_EN (GPIOx_IEN[n+16])/ IF_EN (GPIOx_IEN[n]) bit and IMD (GPIOx_IMD[n]). 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 DBCLKSRC (DBNCECON[4]) and DBCLKSEL (DBNCECON [3:0]) 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.but there are two things need to be noticed (NUC123xxxANx Only) if using GPIO as chip wake-up source.

1. To ensure the I/O status before entering Idle/Power-down mode

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

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

2. To disable the I/O de-bounce function before entering Idle/Power-down mode

If the specified wake-up I/O pin with enabling input signal de-bounce function, system will encounter two GPIO interrupt events while the system is woken up by this GPIO pin. One interrupt event is caused by wake-up function, the other is caused by I/O input de-bounce function. User should be disable the de-bounce function before entering Idle/Power-down mode to avoid the second interrupt event occurred after system waken up.

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				
GPIOA_PMD	GP_BA+0x000	R/W	GPIO Port A I/O Mode Control	0xXXXX_XXXX
GPIOA_OFFD	GP_BA+0x004	R/W	GPIO Port A Digital Input Path Disable Control	0x0000_0000
GPIOA_DOUT	GP_BA+0x008	R/W	GPIO Port A Data Output Value	0x0000_FFFF
GPIOA_DMASK	GP_BA+0x00C	R/W	GPIO Port A Data Output Write Mask	0x0000_0000
GPIOA_PIN	GP_BA+0x010	R	GPIO Port A Pin Value	0x0000_XXXX
GPIOA_DBEN	GP_BA+0x014	R/W	GPIO Port A De-bounce Enable Control Register	0x0000_0000
GPIOA_IMD	GP_BA+0x018	R/W	GPIO Port A Interrupt Mode Control	0x0000_0000
GPIOA_IEN	GP_BA+0x01C	R/W	GPIO Port A Interrupt Enable Control Register	0x0000_0000
GPIOA_ISRC	GP_BA+0x020	R/W	GPIO Port A Interrupt Source Flag	0x0000_0000
GPIOB_PMD	GP_BA+0x040	R/W	GPIO Port B I/O Mode Control	0xXXXX_XXXX
GPIOB_OFFD	GP_BA+0x044	R/W	GPIO Port B Digital Input Path Disable Control	0x0000_0000
GPIOB_DOUT	GP_BA+0x048	R/W	GPIO Port B Data Output Value	0x0000_FFFF
GPIOB_DMASK	GP_BA+0x04C	R/W	GPIO Port B Data Output Write Mask	0x0000_0000
GPIOB_PIN	GP_BA+0x050	R	GPIO Port B Pin Value	0x0000_XXXX
GPIOB_DBEN	GP_BA+0x054	R/W	GPIO Port B De-bounce Enable Control Register	0x0000_0000
GPIOB_IMD	GP_BA+0x058	R/W	GPIO Port B Interrupt Mode Control	0x0000_0000
GPIOB_IEN	GP_BA+0x05C	R/W	GPIO Port B Interrupt Enable Control Register	0x0000_0000
GPIOB_ISRC	GP_BA+0x060	R/W	GPIO Port B Interrupt Source Flag	0x0000_0000
GPIOC_PMD	GP_BA+0x080	R/W	GPIO Port C I/O Mode Control	0xXXXX_XXXX
GPIOC_OFFD	GP_BA+0x084	R/W	GPIO Port C Digital Input Path Disable Control	0x0000_0000
GPIOC_DOUT	GP_BA+0x088	R/W	GPIO Port C Data Output Value	0x0000_FFFF
GPIOC_DMASK	GP_BA+0x08C	R/W	GPIO Port C Data Output Write Mask	0x0000_0000
GPIOC_PIN	GP_BA+0x090	R	GPIO Port C Pin Value	0x0000_XXXX
GPIOC_DBEN	GP_BA+0x094	R/W	GPIO Port C De-bounce Enable Control Register	0x0000_0000
GPIOC_IMD	GP_BA+0x098	R/W	GPIO Port C Interrupt Mode Control	0x0000_0000
GPIOC_IEN	GP_BA+0x09C	R/W	GPIO Port C Interrupt Enable Control Register	0x0000_0000
GPIOC_ISRC	GP_BA+0x0A0	R/W	GPIO Port C Interrupt Source Flag	0x0000_0000

Register	Offset	R/W	Description	Reset Value
GPIOD_PMD	GP_BA+0x0C0	R/W	GPIO Port D I/O Mode Control	0xXXXX_XXXX
GPIOD_OFFD	GP_BA+0x0C4	R/W	GPIO Port D Digital Input Path Disable Control	0x0000_0000
GPIOD_DOUT	GP_BA+0x0C8	R/W	GPIO Port D Data Output Value	0x0000_FFFF
GPIOD_DMASK	GP_BA+0x0CC	R/W	GPIO Port D Data Output Write Mask	0x0000_0000
GPIOD_PIN	GP_BA+0x0D0	R	GPIO Port D Pin Value	0x0000_XXXX
GPIOD_DBEN	GP_BA+0x0D4	R/W	GPIO Port D De-bounce Enable Control Register	0x0000_0000
GPIOD_IMD	GP_BA+0x0D8	R/W	GPIO Port D Interrupt Mode Control	0x0000_0000
GPIOD_IEN	GP_BA+0x0DC	R/W	GPIO Port D Interrupt Enable Control Register	0x0000_0000
GPIOD_ISRC	GP_BA+0x0E0	R/W	GPIO Port D Interrupt Source Flag	0x0000_0000
GPIOF_PMD	GP_BA+0x140	R/W	GPIO Port F I/O Mode Control	0x0000_00XX
GPIOF_OFFD	GP_BA+0x144	R/W	GPIO Port F Digital Input Path Disable Control	0x0000_0000
GPIOF_DOUT	GP_BA+0x148	R/W	GPIO Port F Data Output Value	0x0000_000F
GPIOF_DMASK	GP_BA+0x14C	R/W	GPIO Port F Data Output Write Mask	0x0000_0000
GPIOF_PIN	GP_BA+0x150	R	GPIO Port F Pin Value	0x0000_000X
GPIOF_DBEN	GP_BA+0x154	R/W	GPIO Port F De-bounce Enable Control Register	0x0000_0000
GPIOF_IMD	GP_BA+0x158	R/W	GPIO Port F Interrupt Mode Control	0x0000_0000
GPIOF_IEN	GP_BA+0x15C	R/W	GPIO Port F Interrupt Enable Control Register	0x0000_0000
GPIOF_ISRC	GP_BA+0x160	R/W	GPIO Port F Interrupt Source Flag	0x0000_0000
DBNCECON	GP_BA+0x180	R/W	GPIO Interrupt De-bounce Control Register	0x0000_0020
GPIOA10_DOUT	GP_BA+0x228	R/W	GPIO PA.10 Pin Data Input/Output	0x0000_000X
GPIOA11_DOUT	GP_BA+0x22C	R/W	GPIO PA.11 Pin Data Input/Output	0x0000_000X
GPIOA12_DOUT	GP_BA+0x230	R/W	GPIO PA.12 Pin Data Input/Output	0x0000_000X
GPIOA13_DOUT	GP_BA+0x234	R/W	GPIO PA.13 Pin Data Input/Output	0x0000_000X
GPIOA14_DOUT	GP_BA+0x238	R/W	GPIO PA.14 Pin Data Input/Output	0x0000_000X
GPIOA15_DOUT	GP_BA+0x23C	R/W	GPIO PA.15 Pin Data Input/Output	0x0000_000X
GPIOB0_DOUT	GP_BA+0x240	R/W	GPIO PB.0 Pin Data Input/Output	0x0000_000X
GPIOB1_DOUT	GP_BA+0x244	R/W	GPIO PB.1 Pin Data Input/Output	0x0000_000X
GPIOB2_DOUT	GP_BA+0x248	R/W	GPIO PB.2 Pin Data Input/Output	0x0000_000X
GPIOB3_DOUT	GP_BA+0x24C	R/W	GPIO PB.3 Pin Data Input/Output	0x0000_000X
GPIOB4_DOUT	GP_BA+0x250	R/W	GPIO PB.4 Pin Data Input/Output	0x0000_000X

Register	Offset	R/W	Description	Reset Value
GPIOB5_DOUT	GP_BA+0x254	R/W	GPIO PB.5 Pin Data Input/Output	0x0000_000X
GPIOB6_DOUT	GP_BA+0x258	R/W	GPIO PB.6 Pin Data Input/Output	0x0000_000X
GPIOB7_DOUT	GP_BA+0x25C	R/W	GPIO PB.7 Pin Data Input/Output	0x0000_000X
GPIOB8_DOUT	GP_BA+0x260	R/W	GPIO PB.8 Pin Data Input/Output	0x0000_000X
GPIOB9_DOUT	GP_BA+0x264	R/W	GPIO PB.9 Pin Data Input/Output	0x0000_000X
GPIOB10_DOUT	GP_BA+0x268	R/W	GPIO PB.10 Pin Data Input/Output	0x0000_000X
GPIOB12_DOUT	GP_BA+0x270	R/W	GPIO PB.12 Pin Data Input/Output	0x0000_000X
GPIOB13_DOUT	GP_BA+0x274	R/W	GPIO PB.13 Pin Data Input/Output	0x0000_000X
GPIOB14_DOUT	GP_BA+0x278	R/W	GPIO PB.14 Pin Data Input/Output	0x0000_000X
GPIOB15_DOUT	GP_BA+0x27C	R/W	GPIO PB.15 Pin Data Input/Output	0x0000_000X
GPIOC0_DOUT	GP_BA+0x280	R/W	GPIO PC.0 Pin Data Input/Output	0x0000_000X
GPIOC1_DOUT	GP_BA+0x284	R/W	GPIO PC.1 Pin Data Input/Output	0x0000_000X
GPIOC2_DOUT	GP_BA+0x288	R/W	GPIO PC.2 Pin Data Input/Output	0x0000_000X
GPIOC3_DOUT	GP_BA+0x28C	R/W	GPIO PC.3 Pin Data Input/Output	0x0000_000X
GPIOC4_DOUT	GP_BA+0x290	R/W	GPIO PC.4 Pin Data Input/Output	0x0000_000X
GPIOC5_DOUT	GP_BA+0x294	R/W	GPIO PC.5 Pin Data Input/Output	0x0000_000X
GPIOC8_DOUT	GP_BA+0x2A0	R/W	GPIO PC.8 Pin Data Input/Output	0x0000_000X
GPIOC9_DOUT	GP_BA+0x2A4	R/W	GPIO PC.9 Pin Data Input/Output	0x0000_000X
GPIOC10_DOUT	GP_BA+0x2A8	R/W	GPIO PC.10 Pin Data Input/Output	0x0000_000X
GPIOC11_DOUT	GP_BA+0x2AC	R/W	GPIO PC.11 Pin Data Input/Output	0x0000_000X
GPIOC12_DOUT	GP_BA+0x2B0	R/W	GPIO PC.12 Pin Data Input/Output	0x0000_000X
GPIOC13_DOUT	GP_BA+0x2B4	R/W	GPIO PC.13 Pin Data Input/Output	0x0000_000X
GPIOD0_DOUT	GP_BA+0x2C0	R/W	GPIO PD.0 Pin Data Input/Output	0x0000_000X
GPIOD1_DOUT	GP_BA+0x2C4	R/W	GPIO PD.1 Pin Data Input/Output	0x0000_000X
GPIOD2_DOUT	GP_BA+0x2C8	R/W	GPIO PD.2 Pin Data Input/Output	0x0000_000X
GPIOD3_DOUT	GP_BA+0x2CC	R/W	GPIO PD.3 Pin Data Input/Output	0x0000_000X
GPIOD4_DOUT	GP_BA+0x2D0	R/W	GPIO PD.4 Pin Data Input/Output	0x0000_000X
GPIOD5_DOUT	GP_BA+0x2D4	R/W	GPIO PD.5 Pin Data Input/Output	0x0000_000X
GPIOD8_DOUT	GP_BA+0x2E0	R/W	GPIO PD.8 Pin Data Input/Output	0x0000_000X
GPIOD9_DOUT	GP_BA+0x2E4	R/W	GPIO PD.9 Pin Data Input/Output	0x0000_000X

Register	Offset	R/W	Description	Reset Value
GPIOD10_DOUT	GP_BA+0x2E8	R/W	GPIO PD.10 Pin Data Input/Output	0x0000_000X
GPIOD11_DOUT	GP_BA+0x2EC	R/W	GPIO PD.11 Pin Data Input/Output	0x0000_000X
GPIOF0_DOUT	GP_BA+0x340	R/W	GPIO PF.0 Pin Data Input/Output	0x0000_000X
GPIOF1_DOUT	GP_BA+0x344	R/W	GPIO PF.1 Pin Data Input/Output	0x0000_000X
GPIOF2_DOUT	GP_BA+0x348	R/W	GPIO PF.2 Pin Data Input/Output	0x0000_000X
GPIOF3_DOUT	GP_BA+0x34C	R/W	GPIO PF.3 Pin Data Input/Output	0x0000_000X

6.5.7 Register Description

GPIO Port A-F I/O Mode Control (GPIOx_PMD)

Register	Offset	R/W	Description	Reset Value
GPIOA_PMD	GP_BA+0x000	R/W	GPIO Port A I/O Mode Control	0xXXXX_XXXX
GPIOB_PMD	GP_BA+0x040	R/W	GPIO Port B I/O Mode Control	0xXXXX_XXXX
GPIOC_PMD	GP_BA+0x080	R/W	GPIO Port C I/O Mode Control	0xXXXX_XXXX
GPIOD_PMD	GP_BA+0x0C0	R/W	GPIO Port D I/O Mode Control	0xXXXX_XXXX
GPIOF_PMD	GP_BA+0x140	R/W	GPIO Port F I/O Mode Control	0x0000_00XX

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

Bits	Description	
[2n+1:2n] n=0,1..15	**PMDn**	**Port A-f I/O Pin[N] Mode Control** Determine each I/O type of GPIOx.n pins. 00 = GPIO port [n] pin is in input mode. 01 = GPIO port [n] pin is in Push-pull Output mode. 10 = GPIO port [n] pin is in Open-drain Output mode. 11 = GPIO port [n] pin is in Quasi-bidirectional mode. **Note1:** The default value is 0xFFFF_FFFF and all pins will be quasi-bidirectional mode after chip is powered on (NUC123xxxANx Only). **Note2:** The initial value of this field is defined by CIOINI (Config0 [10]). (NUC123xxxAEx Only) If CIOINI is set to 0, the default value is 0x0000_0000 and all pins will be input tri-state mode after chip powered on. If CIOINI is set to 1, the default value is 0xFFFF_FFFF and all pins will be quasi-bidirectional mode after chip powered on. **Note3:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port F. Others are reserved.

GPIO Port A-F Digital Input Path Disable Control (GPIOx_OFFD)

Register	Offset	R/W	Description	Reset Value
GPIOA_OFFD	GP_BA+0x004	R/W	GPIO Port A Digital Input Path Disable Control	0x0000_0000
GPIOB_OFFD	GP_BA+0x044	R/W	GPIO Port B Digital Input Path Disable Control	0x0000_0000
GPIOC_OFFD	GP_BA+0x084	R/W	GPIO Port C Digital Input Path Disable Control	0x0000_0000
GPIOD_OFFD	GP_BA+0x0C4	R/W	GPIO Port D Digital Input Path Disable Control	0x0000_0000
GPIOF_OFFD	GP_BA+0x144	R/W	GPIO Port F Digital Input Path Disable Control	0x0000_0000

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

Bits	Description	
[n+16] n=0,1..15	**OFFD[n]**	**Port A-f Pin[N] Digital Input Path Disable Control** Each of these bits is used to control if the input path of corresponding Px.n pin is disabled. If input is analog signal, user 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). **Notes:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port F. Others are reserved.
[0:15]	**Reserved**	Reserved.

GPIO Port A-F Data Output Value (GPIOx_DOUT)

Register	Offset	R/W	Description	Reset Value
GPIOA_DOUT	GP_BA+0x008	R/W	GPIO Port A Data Output Value	0x0000_FFFF
GPIOB_DOUT	GP_BA+0x048	R/W	GPIO Port B Data Output Value	0x0000_FFFF
GPIOC_DOUT	GP_BA+0x088	R/W	GPIO Port C Data Output Value	0x0000_FFFF
GPIOD_DOUT	GP_BA+0x0C8	R/W	GPIO Port D Data Output Value	0x0000_FFFF
GPIOF_DOUT	GP_BA+0x148	R/W	GPIO Port F Data Output Value	0x0000_000F

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DOUT[n]**	**Port A-f 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:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port F. Others are reserved.

GPIO Port A-F Data Output Write Mask (GPIOx _DMASK)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DMASK[n]**	**Port A-f Pin[N] Data Output Write Mask** These bits are used to protect the corresponding DOUT (GPIOx_DOUT[n]) bit. When the DATMSK (GPIOx _DATMSK[n]) bit is set to 1, the corresponding DOUT (GPIOx _DOUT[n]) bit is protected. If the write signal is masked, writing data to the protect bit is ignored. 0 = Corresponding DOUT (GPIOx_DOUT[n]) bit can be updated. 1 = Corresponding DOUT (GPIOx_DOUT[n]) bit protected. **Note1:** This function only protect corresponding DOUT (GPIOx_DOUT[n]) bit, and will not protect corresponding bit control register GPIOxn_DOUT. **Note2:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port F. Others are reserved.

GPIO Port A-F Pin Value (GPIOx _PIN)

Register	Offset	R/W	Description	Reset Value
GPIOA_PIN	GP_BA+0x010	R	GPIO Port A Pin Value	0x0000_XXXX
GPIOB_PIN	GP_BA+0x050	R	GPIO Port B Pin Value	0x0000_XXXX
GPIOC_PIN	GP_BA+0x090	R	GPIO Port C Pin Value	0x0000_XXXX
GPIOD_PIN	GP_BA+0x0D0	R	GPIO Port D Pin Value	0x0000_XXXX
GPIOF_PIN	GP_BA+0x150	R	GPIO Port F Pin Value	0x0000_000X

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**PIN[n]**	**Port A-f 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:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port F. Others are reserved.

GPIO Port A-F De-bounce Enable (GPIOx _DBEN)

Register	Offset	R/W	Description	Reset Value
GPIOA_DBEN	GP_BA+0x014	R/W	GPIO Port A De-Bounce Enable Control Register	0x0000_0000
GPIOB_DBEN	GP_BA+0x054	R/W	GPIO Port B De-Bounce Enable Control Register	0x0000_0000
GPIOC_DBEN	GP_BA+0x094	R/W	GPIO Port C De-Bounce Enable Control Register	0x0000_0000
GPIOD_DBEN	GP_BA+0x0D4	R/W	GPIO Port D De-Bounce Enable Control Register	0x0000_0000
GPIOF_DBEN	GP_BA+0x154	R/W	GPIO Port F De-Bounce Enable Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DBEN[n]**	**Port A-f Pin[N] Input Signal De-bounce Enable Bit** The 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 DBCLKSRC (DBNCECON [4]), one de-bounce sample cycle period is controlled by DBCLKSEL (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. **Note:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port F. Others are reserved.

GPIO Port A-F Interrupt Mode Control (GPIOx _IMD)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**IMD[n]**	**Port A-f Pin[N] Edge or Level Detection Interrupt Trigger Type Control** IMD (GPIOx_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 IR_EN (GPIOx_IEN[n+16])/IF_EN (GPIOx_IEN[n]). 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:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port F. Others are reserved.

GPIO Port A-F Interrupt Enable Control (GPIOx _IEN)

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

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

Bits	Description	
[n+16] n=0,1..15	**IR_EN[n]**	**Port A-f Pin[N] Rising Edge or High Level Interrupt Trigger Type Enable Bit** The IR_EN (GPIOx_IEN[n+16]) 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 (GPIOx_IEN[n+16]) bit to 1 : If the interrupt is level trigger (IMD (GPIOx_IMD[n]) bit is set to 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 (Px_IMD[n]) bit is set to 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:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port F. Others are reserved.

[n] n=0,1..15	**IF_EN[n]**	**Port A-f Pin[N] Falling Edge or Low Level Interrupt Trigger Type Enable Bit** The IF_EN (GPIOx_IEN[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 (Px_IEN[n]) bit to 1 : If the interrupt is level trigger (IMD (GPIOx_IMD[n]) bit is set to 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 (GPIOx_IMD[n]) bit is set to 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:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port D. Others are reserved.

GPIO Port A-F Interrupt Trigger Source (GPIOx_ISRC)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**ISRC[n]**	**Port A-f Pin[N] Interrupt Source Flag** Write Operation : 0 = No action. 1 = Clear the corresponding pending interrupt. Read Operation : 0 = No interrupt at Px.n. 1 = Px.n generates an interrupt. **Note:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port F. Others are reserved.

GPIO Interrupt De-bounce Control Register (DBNCECON)

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

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**ICLK_ON**	**Interrupt Clock on Mode** 0 = Edge detection circuit is active only if I/O pin corresponding IR_EN (GPIOx_IEN[n+16])/IF_EN (Px_IEN[n]) bit is set to 1. 1 = All I/O pins edge detection circuit is always active after reset. **Note:** It is recommended to disable this bit to save system power if no special application concern.
[4]	**DBCLKSRC**	**De-bounce Counter Clock Source Selection** 0 = De-bounce counter clock source is the HCLK. 1 = De-bounce counter clock source is the internal 10 kHz 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 (GPIOxn_DOUT)

Register	Offset	R/W	Description	Reset Value
GPIOAN_DOUT	GP_BA+0x200 +0x04*n	R/W	GPIO PA.n Pin Data Input/Output	0x0000_000X
GPIOBN_DOUT	GP_BA+0x240 +0x04*n	R/W	GPIO PB.n Pin Data Input/Output	0x0000_000X
GPIOCN_DOUT	GP_BA+0x280 +0x04*n	R/W	GPIO PC.n Pin Data Input/Output	0x0000_000X
GPIODN_DOUT	GP_BA+0x2C0 +0x04*n	R/W	GPIO PD.n Pin Data Input/Output	0x0000_000X
GPIOFN_DOUT	GP_BA+0x340 +0x04*n	R/W	GPIO PF.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							**GPIOxn_DOUT**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**GPIOxn_DOUT**	**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 GPIOA0_DOUT will reflect the written value to bit DOUT (GPIOA_DOUT[0]), read GPIOA0_DOUT will return the value of PIN (GPIOA_PIN[0]). **Note1:** The writing operation will not be affected by register DMASK (GPIOx_DMASK[n]). **Note2:** n = 10~15 for port A. Others are reserved. n = 0~10, 12~15 for port B. Others are reserved. n = 0~5, 8~13 for port C. Others are reserved. n = 0~5, 8~11 for port D. Others are reserved. n = 0~3 for port F. Others are reserved.

6.6 PDMA Controller (PDMA)

6.6.1 Overview

The NuMicro® NUC123 contains a six-channel peripheral direct memory access (PDMA) controller and a cyclic redundancy check (CRC) generator.

The PDMA can transfer data to and from memory or transfer data to and from APB devices. For PDMA channel (PDMA CH0~CH5), there is one-word buffer as transfer buffer between the Peripherals APB devices and Memory. The CPU can recognize the completion of a PDMA operation by software polling or when it receives an internal PDMA interrupt. The PDMA controller can increase source or destination address or fixed them as well.

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

6.6.2 Features

- Supports six PDMA channels and one CRC channel; each PDMA channel can support a unidirectional transfer
- AMBA AHB master/slave interface compatible, for data transfer and register read/write
- Hardware round robin priority scheme. PDMA channel 0 has the highest priority
- PDMA
  - Peripheral-to-memory, memory-to-peripheral, and memory-to-memory transfer
  - Supports word/half-word/byte transfer data width from/to peripheral
  - Supports address direction: increment, fixed
  - Supports software, SPI, UART, ADC, PWM and I^2S request
- Cyclic Redundancy Check (CRC)
  - Supports four common polynomials CRC-CCITT, CRC-8, CRC-16, and CRC-32
    - CRC-CCITT: $X^{16} + X^{12} + X^5 + 1$
    - CRC-8: $X^8 + X^2 + X + 1$
    - CRC-16: $X^{16} + X^{15} + X^2 + 1$
    - CRC-32: $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
  - Programmable seed value
  - Supports programmable order reverse setting for input data and CRC checksum
  - Supports programmable 1's complement setting for input data and CRC checksum.
  - Supports CPU PIO mode or PDMA transfer mode
  - Supports 8/16/32-bit of data width in CPU PIO mode
    - 8-bit write mode: 1-AHB clock cycle operation
    - 16-bit write mode: 2-AHB clock cycle operation
    - 32-bit write mode: 4-AHB clock cycle operation
  - Supports byte alignment transfer length in CRC PDMA mode

6.6.3 Block Diagram

The PDMA block diagram, PDMA clock control diagram and CRC block diagram are shown as follows.

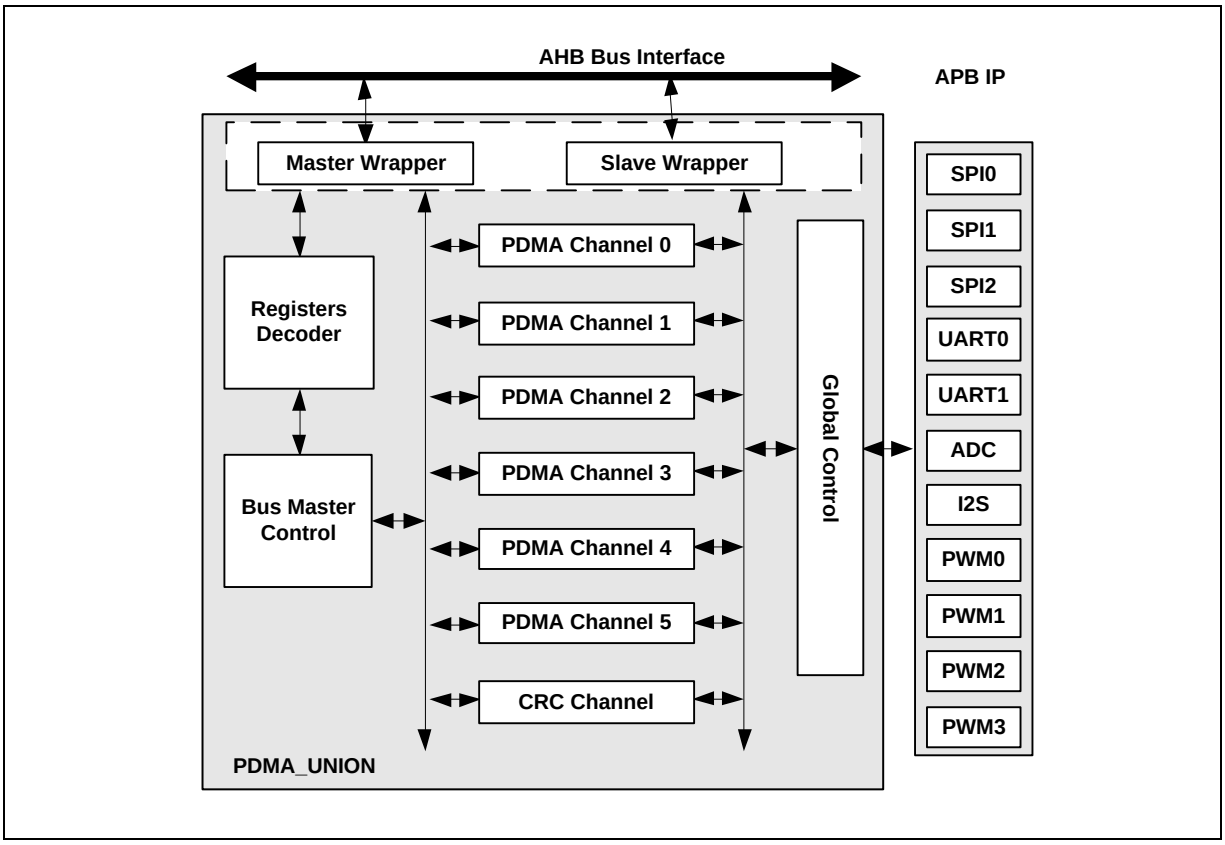


Figure 6-23 PDMA Controller Block Diagram

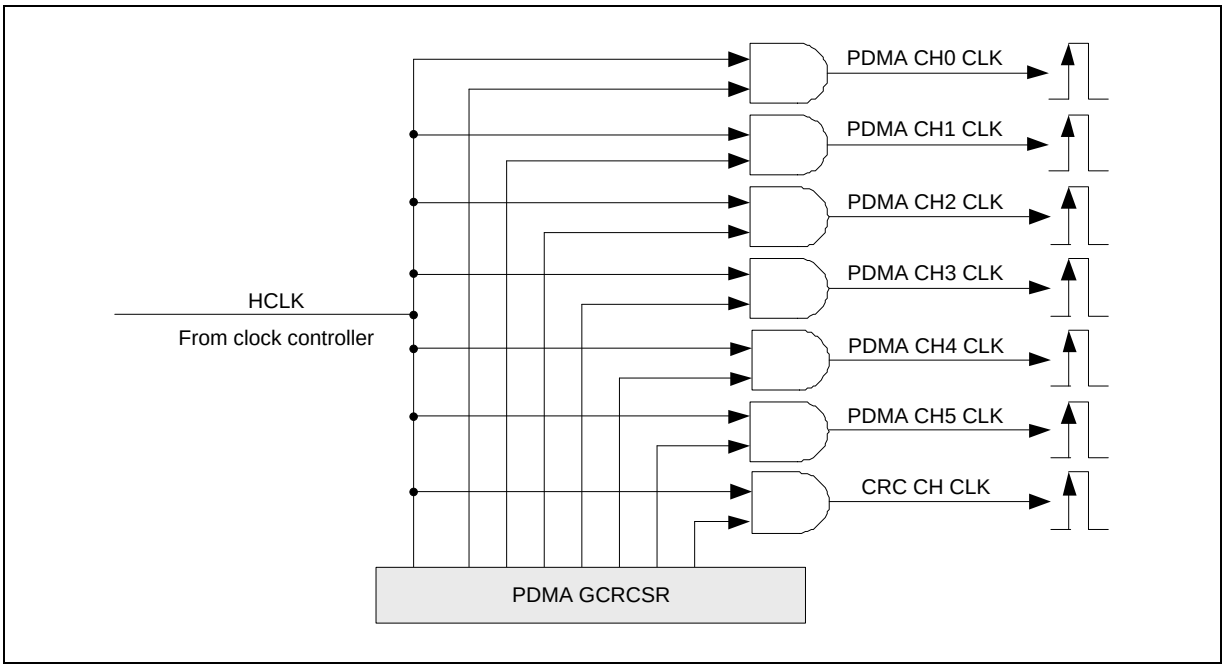


Figure 6-24 PDMA Clock Controller Block Diagram

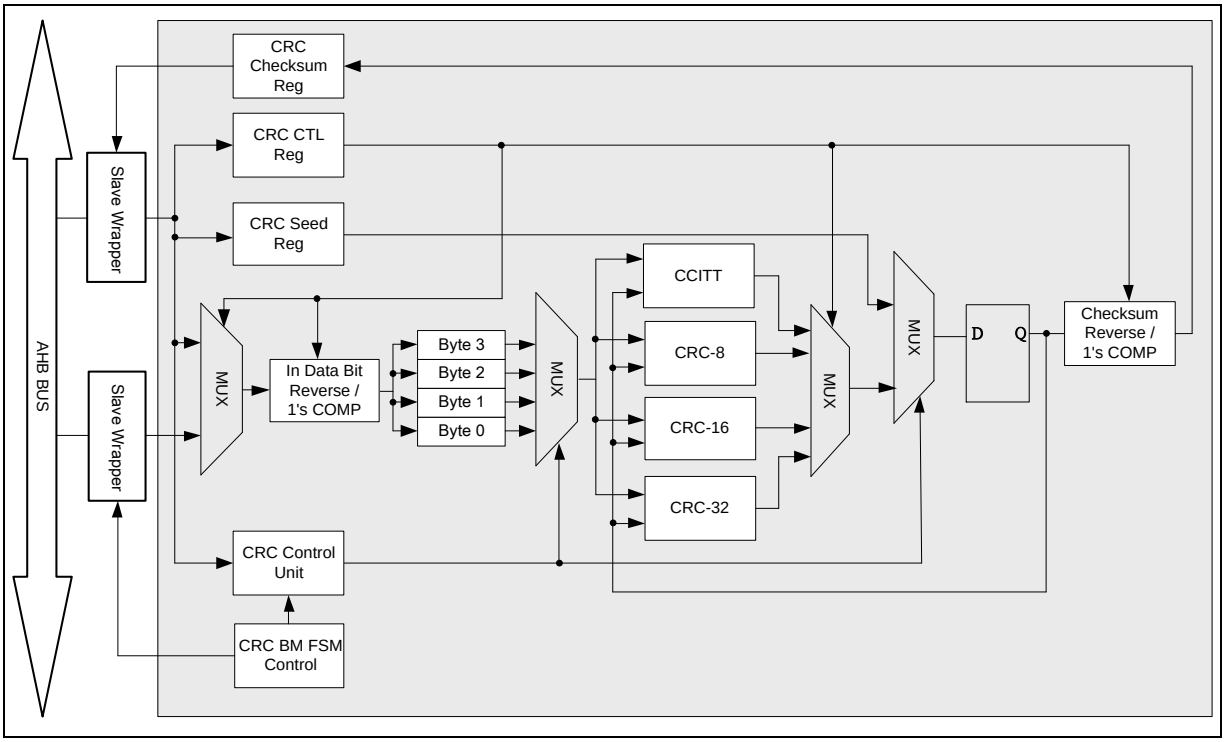


Figure 6-25 CRC Generator Block Diagram

6.6.4 Basic Configuration

The clock source of PDMA can be enabled in PDMA_EN (AHBCLK[1]).

6.6.5 Functional Description

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

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

6.6.5.1 *PDMA*

The PDMA controller has six channels to support Peripheral-to-Memory or Memory-to-Peripheral or Memory-to-Memory data transfer. Every PDMA channel behavior is not pre-defined, so user must configure the channel service settings of PDMA_PDSSR0, PDMA_PDSSR1 and PDMA_PDSSR2 registers before start the related PDMA channel. PDMA controller only services a channel in a time, as the result, six channels using round robin priority scheme, and channel 0 to channel5 corresponding priority is from high to low.

The PDMA controller supports independent address control for source and destination address. By setting SAD_SEL (PDMA_CSRx[5:4]) to control transfer source address increment or fixed, and setting DAD_SEL (PDMA_CSRx[7:6]) to control transfer destination address increment or fixed. The unit of transfer data between peripheral and memory can be selected from APB_TWS (PDMA_CSRx[20:19]).

The PDMA controller will generate interrupt signal only in two cases: when PDMA finished all transfer,

then BLKD_IF (PDMA_ISRx[1]) will be set and if BLKD_IE (PDMA_IERx[1]) is enabled. Another case is when PDMA received transfer error response, then TABORT_IF (PDMA_ISRx[0]) will be set and if TABORT_IE (PDMA_IERx[0]) is enabled.

Software must enable PDMA channel by setting PDMACEN bit and then write a valid source address to the PDMA_SARx register, a destination address to the PDMA_DARx register, and a transfer count to the PDMA_BCRx register. Next, trigger the TRIG_EN (PDMA_CSRx[23]). PDMA will continue the transfer until PDMA_CBCRx comes down to zero, If an error occurs during the PDMA operation, the channel stops unless software clears the error condition and sets the SW_RST (PDMA_CSRx [1]) to reset the PDMA channel and set PDMACEN (PDMA_CSRx[0]) and TRIG_EN (PDMA_CSRx[23]) bits field to start again.

6.6.5.2 *CRC*

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

The CRC generator supports CPU PIO mode (CRCCEN (CRC_CTL[0]) = 1, TRIG_EN (CRC_CTL[23]) = 0) and PDMA transfer mode (CRCCEN (CRC_CTL[0]) = 1, TRIG_EN (CRC_CTL[23]) = 1). The following sequence is a program sequence example.

Procedure when operation in CPU PIO mode:

1. Enable CRC generator by setting CRCCEN (CRC_CTL[0]).
2. Initial Setting. Setting the data format (WDATA_RVS (CRC_CTL[24]), CHECKSUM_RVS (CRC_CTL[25]), WDATA_COM (CRC_CTL[26]) and CHECKSUM_COM (CRC_CTL[27])), initial seed value (CRC_SEED register) and select the data length by setting CPU_WDLEN (CRC_CTL[29:28]).
3. Setting CRC reset to load the initial seed value to CRC circuit by setting CRC_RST (CRC_CTL[1]).
4. Write data to CRC_WDATA register to perform CRC calculation.
5. Get the CRC checksum result by reading CRC_CHECKSUM register.

Procedure when operation in CRC PDMA mode:

1. Enable CRC generator by setting CRCCEN (CRC_CTL[0]).
2. Initial Setting. Setting the data format (WDATA_RVS (CRC_CTL[24]), CHECKSUM_RVS (CRC_CTL[25]), WDATA_COM (CRC_CTL[26]) and CHECKSUM_COM (CRC_CTL[27])), initial seed value (CRC_SEED register).
3. Give a valid source address and transfer count by setting CRC_DMASAR and CRC_DMABCR registers.
4. Enable TRIG_EN (CRC_CTL[23]) and then hardware will reset the seed value and then read memory data to perform CRC calculation.
5. Wait CRC PDMA transfer and CRC calculation done and then get the CRC checksum result by reading CRC_CHECKSUM register.

6.6.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
PDMA_BA_ch0 = 0x5000_8000 **PDMA_BA_ch1 = 0x5000_8100** **PDMA_BA_ch2 = 0x5000_8200** **PDMA_BA_ch3 = 0x5000_8300** **PDMA_BA_ch4 = 0x5000_8400** **PDMA_BA_ch5 = 0x5000_8500**				
PDMA_CSRX	PDMA_BA_chx+0x00	R/W	PDMA Control Register	0x0000_0000
PDMA_SARX	PDMA_BA_chx+0x04	R/W	PDMA Transfer Source Address Register	0x0000_0000
PDMA_DARX	PDMA_BA_chx+0x08	R/W	PDMA Transfer Destination Address Register	0x0000_0000
PDMA_BCRX	PDMA_BA_chx+0x0C	R/W	PDMA Transfer Byte Count Register	0x0000_0000
PDMA_POINTX	PDMA_BA_chx+0x10	R	PDMA Internal Buffer Pointer	0xXXXX_0000
PDMA_CSARX	PDMA_BA_chx+0x14	R	PDMA Current Source Address Register	0x0000_0000
PDMA_CDARX	PDMA_BA_chx+0x18	R	PDMA Current Destination Address Register	0x0000_0000
PDMA_CBCRX	PDMA_BA_chx+0x1C	R	PDMA Current Transfer Byte Count Register	0x0000_0000
PDMA_IERX	PDMA_BA_chx+0x20	R/W	PDMA Interrupt Enable Register	0x0000_0001
PDMA_ISRX	PDMA_BA_chx+0x24	R/W	PDMA Interrupt Status Register	0x0000_0000
PDMA_SBUF_CX	PDMA_BA_chx+0x80	R	PDMA Shared Buffer FIFO	0x0000_0000
CRC_BA = 0x5000_8E00				
CRC_CTL	CRC_BA+0x00	R/W	CRC Control Register	0x2000_0000
CRC_DMASAR	CRC_BA+0x04	R/W	CRC PDMA Transfer Source Address Register	0x0000_0000
CRC_DMABCR	CRC_BA+0x0C	R/W	CRC PDMA Transfer Byte Count Register	0x0000_0000
CRC_DMACSAR	CRC_BA+0x14	R/W	CRC PDMA Current Source Address Register	0x0000_0000
CRC_DMACBCR	CRC_BA+0x1C	R/W	CRC PDMA Current Transfer Byte Count Register	0x0000_0000
CRC_DMAIER	CRC_BA+0x20	R/W	CRC PDMA Interrupt Enable Register	0x0000_0001
CRC_DMAISR	CRC_BA+0x24	R/W	CRC PDMA Interrupt Status Register	0x0000_0000
CRC_WDATA	CRC_BA+0x80	R/W	CRC Write Data Register	0x0000_0000
CRC_SEED	CRC_BA+0x84	R/W	CRC Seed Register	0xFFFF_FFFF
CRC_CHECKSUM	CRC_BA+0X88	R	CRC Check Sum Register	0x0000_0000
PDMA_GCR_BA = 0x5000_8F00				
PDMA_GCRCTL	PDMA_GCR_BA+0x00	R/W	PDMA Global Control Register	0x0000_0000

PDMA_PDSSR0	PDMA_GCR_BA+0x04	R/W	PDMA Service Selection Control Register 0	0x00FF_FFFF
PDMA_PDSSR1	PDMA_GCR_BA+0x08	R/W	PDMA Service Selection Control Register 1	0x0FFF_FFFF
PDMA_GCRISR	PDMA_GCR_BA+0x0C	R/W	PDMA Global Interrupt Status Register	0x0000_0000
PDMA_PDSSR2	PDMA_GCR_BA+0x10	R/W	PDMA Service Selection Control Register 2	0x00FF_FFFF

6.6.7 Register Description

PDMA Control Register (PDMA_CSRx)

Register	Offset	R/W	Description	Reset Value
PDMA_CSR0	PDMA_BA_ch0+0x00	R/W	PDMA Control Register CH0	0x0000_0000
PDMA_CSR1	PDMA_BA_ch1+0x00	R/W	PDMA Control Register CH1	0x0000_0000
PDMA_CSR2	PDMA_BA_ch2+0x00	R/W	PDMA Control Register CH2	0x0000_0000
PDMA_CSR3	PDMA_BA_ch3+0x00	R/W	PDMA Control Register CH3	0x0000_0000
PDMA_CSR4	PDMA_BA_ch4+0x00	R/W	PDMA Control Register CH4	0x0000_0000
PDMA_CSR5	PDMA_BA_ch5+0x00	R/W	PDMA Control Register CH5	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**TRIG_EN**	**PDMA Software Trigger Enable Bit** 0 = No effect. 1 = PDMA data read or write transfer Enabled. **Note1:** When PDMA transfer completed, this bit will be cleared automatically. **Note2:** If the bus error occurs, all PDMA transfer will be stopped. Software must reset all PDMA channel, and then trigger again.
[22:21]	**Reserved**	Reserved.
[20:19]	**APB_TWS**	**Peripheral Transfer Width Selection** 00 = One word (32-bit) is transferred for every PDMA operation. 01 = One byte (8-bit) is transferred for every PDMA operation. 10 = One half-word (16-bit) is transferred for every PDMA operation. 11 = Reserved. **Note:** This field is meaningful only when MODE_SEL is Peripheral to Memory mode (Peripheral-to-Memory) or Memory to Peripheral mode (Memory-to-Peripheral).
[18:8]	**Reserved**	Reserved.

[7:6]	**DAD_SEL**	**Transfer Destination Address Direction Selection** 00 = Transfer destination address is increasing successively. 01 = Reserved. 10 = Transfer destination address is fixed (This feature can be used when data transferred from multiple sources to a single destination). 11 = Reserved.
[5:4]	**SAD_SEL**	**Transfer Source Address Direction Selection** 00 = Transfer source address is increasing successively. 01 = Reserved. 10 = Transfer source address is fixed (This feature can be used when data where transferred from a single source to multiple destinations). 11 = Reserved.
[3:2]	**MODE_SEL**	**PDMA Mode Selection** 00 = Memory to Memory mode (Memory-to-Memory). 01 = Peripheral to Memory mode (Peripheral-to-Memory). 10 = Memory to Peripheral mode (Memory-to-Peripheral).
[1]	**SW_RST**	**Software Engine Reset** 0 = No effect. 1 = Reset the internal state machine, pointers and internal buffer. The contents of control register will not be cleared. This bit will be automatically cleared after one AHB clock cycle.
[0]	**PDMACEN**	**PDMA Channel Enable Bit** Setting this bit to 1 enables PDMA's operation. If this bit is cleared, PDMA will ignore all PDMA request and force Bus Master into IDLE state.

PDMA Transfer Source Address Register (PDMA_SARx)

Register	Offset	R/W	Description	Reset Value
PDMA_SAR0	PDMA_BA_ch0+0x04	R/W	PDMA Transfer Source Address Register CH0	0x0000_0000
PDMA_SAR1	PDMA_BA_ch1+0x04	R/W	PDMA Transfer Source Address Register CH1	0x0000_0000
PDMA_SAR2	PDMA_BA_ch2+0x04	R/W	PDMA Transfer Source Address Register CH2	0x0000_0000
PDMA_SAR3	PDMA_BA_ch3+0x04	R/W	PDMA Transfer Source Address Register CH3	0x0000_0000
PDMA_SAR4	PDMA_BA_ch4+0x04	R/W	PDMA Transfer Source Address Register CH4	0x0000_0000
PDMA_SAR5	PDMA_BA_ch5+0x04	R/W	PDMA Transfer Source Address Register CH5	0x0000_0000

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

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

PDMA Transfer Destination Address Register (PDMA_DARx)

Register	Offset	R/W	Description	Reset Value
PDMA_DAR0	PDMA_BA_ch0+0x08	R/W	PDMA Transfer Destination Address Register CH0	0x0000_0000
PDMA_DAR1	PDMA_BA_ch1+0x08	R/W	PDMA Transfer Destination Address Register CH1	0x0000_0000
PDMA_DAR2	PDMA_BA_ch2+0x08	R/W	PDMA Transfer Destination Address Register CH2	0x0000_0000
PDMA_DAR3	PDMA_BA_ch3+0x08	R/W	PDMA Transfer Destination Address Register CH3	0x0000_0000
PDMA_DAR4	PDMA_BA_ch4+0x08	R/W	PDMA Transfer Destination Address Register CH4	0x0000_0000
PDMA_DAR5	PDMA_BA_ch5+0x08	R/W	PDMA Transfer Destination Address Register CH5	0x0000_0000

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

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

PDMA Transfer Byte Count Register (PDMA_BCRx)

Register	Offset	R/W	Description	Reset Value
PDMA_BCR0	PDMA_BA_ch0+0x0C	R/W	PDMA Transfer Byte Count Register CH0	0x0000_0000
PDMA_BCR1	PDMA_BA_ch1+0x0C	R/W	PDMA Transfer Byte Count Register CH1	0x0000_0000
PDMA_BCR2	PDMA_BA_ch2+0x0C	R/W	PDMA Transfer Byte Count Register CH2	0x0000_0000
PDMA_BCR3	PDMA_BA_ch3+0x0C	R/W	PDMA Transfer Byte Count Register CH3	0x0000_0000
PDMA_BCR4	PDMA_BA_ch4+0x0C	R/W	PDMA Transfer Byte Count Register CH4	0x0000_0000
PDMA_BCR5	PDMA_BA_ch5+0x0C	R/W	PDMA Transfer Byte Count Register CH5	0x0000_0000

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

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

PDMA Internal Buffer Pointer Register (PDMA_POINTx)

Register	Offset	R/W	Description	Reset Value
PDMA_POINT0	PDMA_BA_ch0+0x10	R	PDMA Internal Buffer Pointer Register CH0	0xXXXX_0000
PDMA_POINT1	PDMA_BA_ch1+0x10	R	PDMA Internal Buffer Pointer Register CH1	0xXXXX_0000
PDMA_POINT2	PDMA_BA_ch2+0x10	R	PDMA Internal Buffer Pointer Register CH2	0xXXXX_0000
PDMA_POINT3	PDMA_BA_ch3+0x10	R	PDMA Internal Buffer Pointer Register CH3	0xXXXX_0000
PDMA_POINT4	PDMA_BA_ch4+0x10	R	PDMA Internal Buffer Pointer Register CH4	0xXXXX_0000
PDMA_POINT5	PDMA_BA_ch5+0x10	R	PDMA Internal Buffer Pointer Register CH5	0xXXXX_0000

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

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

PDMA Current Source Address Register (PDMA_CSARx)

Register	Offset	R/W	Description	Reset Value
PDMA_CSAR0	PDMA_BA_ch0+0x14	R	PDMA Current Source Address Register CH0	0x0000_0000
PDMA_CSAR1	PDMA_BA_ch1+0x14	R	PDMA Current Source Address Register CH1	0x0000_0000
PDMA_CSAR2	PDMA_BA_ch2+0x14	R	PDMA Current Source Address Register CH2	0x0000_0000
PDMA_CSAR3	PDMA_BA_ch3+0x14	R	PDMA Current Source Address Register CH3	0x0000_0000
PDMA_CSAR4	PDMA_BA_ch4+0x14	R	PDMA Current Source Address Register CH4	0x0000_0000
PDMA_CSAR5	PDMA_BA_ch5+0x14	R	PDMA Current Source Address Register CH5	0x0000_0000

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

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

PDMA Current Destination Address Register (PDMA_CDARx)

Register	Offset	R/W	Description	Reset Value
PDMA_CDAR0	PDMA_BA_ch0+0x18	R	PDMA Current Destination Address Register CH0	0x0000_0000
PDMA_CDAR1	PDMA_BA_ch1+0x18	R	PDMA Current Destination Address Register CH1	0x0000_0000
PDMA_CDAR2	PDMA_BA_ch2+0x18	R	PDMA Current Destination Address Register CH2	0x0000_0000
PDMA_CDAR3	PDMA_BA_ch3+0x18	R	PDMA Current Destination Address Register CH3	0x0000_0000
PDMA_CDAR4	PDMA_BA_ch4+0x18	R	PDMA Current Destination Address Register CH4	0x0000_0000
PDMA_CDAR5	PDMA_BA_ch5+0x18	R	PDMA Current Destination Address Register CH5	0x0000_0000

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

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

PDMA Current Byte Count Register (PDMA_CBCRx)

Register	Offset	R/W	Description	Reset Value
PDMA_CBCR0	PDMA_BA_ch0+0x1C	R	PDMA Current Byte Count Register CH0	0x0000_0000
PDMA_CBCR1	PDMA_BA_ch1+0x1C	R	PDMA Current Byte Count Register CH1	0x0000_0000
PDMA_CBCR2	PDMA_BA_ch2+0x1C	R	PDMA Current Byte Count Register CH2	0x0000_0000
PDMA_CBCR3	PDMA_BA_ch3+0x1C	R	PDMA Current Byte Count Register CH3	0x0000_0000
PDMA_CBCR4	PDMA_BA_ch4+0x1C	R	PDMA Current Byte Count Register CH4	0x0000_0000
PDMA_CBCR5	PDMA_BA_ch5+0x1C	R	PDMA Current Byte Count Register CH5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PDMA_CBCR**	**PDMA Current Byte Count Register (Read Only)** This field indicates the current remained byte count of PDMA. **Note:** SW_RST will clear this register value.

PDMA Interrupt Enable Register (PDMA_IERx)

Register	Offset	R/W	Description	Reset Value
PDMA_IER0	PDMA_BA_ch0+0x20	R/W	PDMA Interrupt Enable Register CH0	0x0000_0001
PDMA_IER1	PDMA_BA_ch1+0x20	R/W	PDMA Interrupt Enable Register CH1	0x0000_0001
PDMA_IER2	PDMA_BA_ch2+0x20	R/W	PDMA Interrupt Enable Register CH2	0x0000_0001
PDMA_IER3	PDMA_BA_ch3+0x20	R/W	PDMA Interrupt Enable Register CH3	0x0000_0001
PDMA_IER4	PDMA_BA_ch4+0x20	R/W	PDMA Interrupt Enable Register CH4	0x0000_0001
PDMA_IER5	PDMA_BA_ch5+0x20	R/W	PDMA Interrupt Enable Register CH5	0x0000_0001

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

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

PDMA Interrupt Status Register (PDMA_ISRx)

Register	Offset	R/W	Description	Reset Value
PDMA_ISR0	PDMA_BA_ch0+0x24	R/W	PDMA Interrupt Status Register CH0	0x0000_0000
PDMA_ISR1	PDMA_BA_ch1+0x24	R/W	PDMA Interrupt Status Register CH1	0x0000_0000
PDMA_ISR2	PDMA_BA_ch2+0x24	R/W	PDMA Interrupt Status Register CH2	0x0000_0000
PDMA_ISR3	PDMA_BA_ch3+0x24	R/W	PDMA Interrupt Status Register CH3	0x0000_0000
PDMA_ISR4	PDMA_BA_ch4+0x24	R/W	PDMA Interrupt Status Register CH4	0x0000_0000
PDMA_ISR5	PDMA_BA_ch5+0x24	R/W	PDMA Interrupt Status Register CH5	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**BLKD_IF**	**Block Transfer Done Interrupt Flag** This bit indicates that PDMA has finished all transfer. 0 = Not finished. 1 = Done. **Note:** This bit can be cleared to 0 by software writing '1'.
[0]	**TABORT_IF**	**PDMA Read/Write Target Abort Interrupt Flag** 0 = No bus ERROR response received. 1 = Bus ERROR response received. **Note:** This bit can be cleared to 0 by software writing '1'.

Note: TABORT_IF (PDMA_ISR[0]) indicates if bus master has received ERROR response or not. If bus master received ERROR response, it means that target abort is happened. PDMAC will stop transfer and respond this event to software then go to IDLE state. When target abort occurred, software must reset PDMA, and then transfer those data again.

PDMA Shared Buffer FIFO 0 (PDMA_SBUF_cx)

Register	Offset	R/W	Description	Reset Value
PDMA_SBUF_C0	PDMA_BA_ch0+0x080	R	PDMA Shared Buffer FIFO Register CH0	0x0000_0000
PDMA_SBUF_C1	PDMA_BA_ch1+0x180	R	PDMA Shared Buffer FIFO Register CH1	0x0000_0000
PDMA_SBUF_C2	PDMA_BA_ch2+0x280	R	PDMA Shared Buffer FIFO Register CH2	0x0000_0000
PDMA_SBUF_C3	PDMA_BA_ch3+0x380	R	PDMA Shared Buffer FIFO Register CH3	0x0000_0000
PDMA_SBUF_C4	PDMA_BA_ch4+0x480	R	PDMA Shared Buffer FIFO Register CH4	0x0000_0000
PDMA_SBUF_C5	PDMA_BA_ch5+0x580	R	PDMA Shared Buffer FIFO Register CH5	0x0000_0000

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

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

CRC Control Register (CRC_CTL)

Register	Offset	R/W	Description	Reset Value
CRC_CTL	CRC_BA+0x00	R/W	CRC Control Register	0x2000_0000

31	30	29	28	27	26	25	24
CRC_MODE		CPU_WDLEN		CHECKSUM_COM	WDATA_COM	CHECKSUM_RVS	WDATA_RVS
23	22	21	20	19	18	17	16
TRIG_EN	Reserved						
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CRC_RST	CRCCEN

Bits	Description	
[31:30]	**CRC_MODE**	**CRC Polynomial Mode** 00 = CRC-CCITT Polynomial mode. 01 = CRC-8 Polynomial mode. 10 = CRC-16 Polynomial mode. 11 = CRC-32 Polynomial mode.
[29:28]	**CPU_WDLEN**	**CPU Write Data Length** When operation in CPU PIO mode (CRCCEN= 1, TRIG_EN = 0), this field indicates the write data length. 00 = Data length is 8-bit mode. 01 = Data length is 16-bit mode. 1x = Data length is 32-bit mode. **Note1:** This field is used for CPU PIO mode. **Note2:** When the data length is 8-bit mode, the valid data is CRC_WDATA [7:0]; if the data length is 16-bit mode, the valid data is CRC_WDATA [15:0].
[27]	**CHECKSUM_COM**	**Checksum Complement** 0 = No 1's complement for CRC checksum. 1 = 1's complement for CRC checksum.
[26]	**WDATA_COM**	**Write Data Complement** 0 = No 1's complement for CRC write data in. 1 = 1's complement for CRC write data in.
[25]	**CHECKSUM_RVS**	**Checksum Reverse** 0 = No bit order reverse for CRC checksum. 1 = Bit order reverse for CRC checksum. **Note:** If the checksum data is 0XDD7B0F2E, the bit order reversed for CRC checksum is 0x74F0DEBB.

[24]	**WDATA_RVS**	**Write Data Order Reverse** 0 = No bit order reversed for CRC write data in. 1 = Bit order reversed for CRC write data in (per byte). **Note:** If the write data is 0xAABBCCDD, the bit order reverse for CRC write data in is 0x55DD33BB.
[23]	**TRIG_EN**	**CRC Software Trigger Enable Bit** 0 = No effect. 1 = CRC PDMA data read or write transfer Enabled. **Note1:** If this bit assert indicates the CRC engine operation in CRC PDMA mode, do not fill in any data in CRC_WDATA register. **Note2:** When CRC PDMA transfer is completed, this bit will be cleared automatically. **Note3:** If the bus error occurs, all CRC PDMA transfer will be stopped. Software must reset all PDMA channel, and then trigger again.
[22:2]	**Reserved**	Reserved.
[1]	**CRC_RST**	**CRC Engine Reset** 0 = No effect. 1 = Reset the internal CRC state machine and internal buffer. The contents of control register will not be cleared. This bit will automatically be cleared after one AHB clock cycles. **Note:** When operated in CPU PIO mode, setting this bit will reload the initial seed value.
[0]	**CRCCEN**	**CRC Channel Enable Bit** Setting this bit to 1 enables CRC's operation. When operation in CRC PDMA mode (TRIG_EN = 1), if user clears this bit, the PDMA operation will be continuous until all CRC PDMA operation is done, and the TRIG_EN bit will asserted until all CRC PDMA operation done. But in this case, the BLKD_IF (CRC_DMAISR[1]) flag will inactive, user can read CRC result by reading CRC_CHECKSUM register when TRIG_EN = 0. When operation in CRC PDMA mode (TRIG_EN = 1), if user wants to stop the transfer immediately, user can write 1 to CRC_RST bit to stop the transmission.

CRC PDMA Transfer Source Address Register (CRC_DMASAR)

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

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

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

CRC PDMA Transfer Byte Count Register (CRC_DMABCR)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CRC_DMABCR**	**CRC PDMA Transfer Byte Count Register** This field indicates a 16-bit transfer byte count number of CRC PDMA.

CRC PDMA Current Source Address Register (CRC_DMACSAR)

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

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

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

CRC PDMA Current Byte Count Register (CRC_DMACBCR)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CRC_DMACBCR**	**CRC PDMA Current Byte Count Register (Read Only)** This field indicates the current remained byte count of CRC_DMA. **Note:** CRC_RST will clear this register value.

CRC PDMA Interrupt Enable Register (CRC_DMAIER)

Register	Offset	R/W	Description	Reset Value
CRC_DMAIER	CRC_BA+0x20	R/W	CRC PDMA Interrupt Enable Control Register	0x0000_0001

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

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

CRC PDMA Interrupt Status Register (CRC_DMAISR)

Register	Offset	R/W	Description	Reset Value
CRC_DMAISR	CRC_BA+0x24	R/W	CRC PDMA Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**BLKD_IF**	**Block Transfer Done Interrupt Flag** This bit indicates that CRC PDMA has finished all transfer. 0 = Not finished. 1 = Done. **Note:** This bit can be cleared to 0 by software writing '1'.
[0]	**TABORT_IF**	**CRC PDMA Read/Write Target Abort Interrupt Flag** 0 = No bus ERROR response received. 1 = Bus ERROR response received. **Note:** This bit can be cleared to 0 by software writing '1'.

Note: TABORT_IF (CRC_DMAISR[0]) indicate if bus master received ERROR response or not. If bus master received ERROR response, it means that target abort is happened. PDMA will stop transfer and respond this event to software then go to IDLE state. When target abort occurred, software must reset PDMA, and then transfer those data again.

CRC Write Data Register (CRC_WDATA)

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

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

Bits	Description	
[31:0]	**CRC_WDATA**	**CRC Write Data Register** When operated in CPU PIO mode (CRCCEN (CRC_CTL[0]) = 1, TRIG_EN (CRC_CTL[23]) = 0), software can write data to this field to perform CRC operation;. When operated in CRC PDMA mode (CRCCEN (CRC_CTL[0]) = 1, CRC_CTL [TRIG_EN] = 1), this field will be used for PDMA internal buffer. **Note1:** When operated in CRC PDMA mode, so don't filled any data in this field. **Note2:** The WDATA_COM (CRC_CTL[26]) and WDATA_RVS (CRC_CTL[24]) bit setting will affect this field; for example, if WDATA_RVS = 1, if the write data in CRC_WDATA register is 0xAABBCCDD, the read data from CRC_WDATA register will be 0x55DD33BB.

CRC Seed Register (CRC_SEED)

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

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

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

CRC Checksum Register (CRC_CHECKSUM)

Register	Offset	R/W	Description	Reset Value
CRC_CHECKSUM	CRC_BA+0x88	R	CRC Checksum Register	0x0000_0000

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

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

PDMA Global Control Register (PDMA_GCRCTL)

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

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**CRC_CLK_EN**	**CRC Controller Clock Enable Control** 0 = CRC controller clock Disabled. 1 = CRC controller clock Enabled.
[23:14]	**Reserved**	Reserved.
[13]	**CLK5_EN**	**PDMA Controller Channel 5 Clock Enable Control** 0 = PDMA channel 5 clock Disabled. 1 = PDMA channel 5 clock Enabled.
[12]	**CLK4_EN**	**PDMA Controller Channel 4 Clock Enable Control** 0 = PDMA channel 4 clock Disabled. 1 = PDMA channel 4 clock Enabled.
[11	**CLK3_EN**	**PDMA Controller Channel 3 Clock Enable Control** 0 = PDMA channel 3 clock Disabled. 1 = PDMA channel 3 clock Enabled.
[10	**CLK2_EN**	**PDMA Controller Channel 2 Clock Enable Control** 0 = PDMA channel 2 clock Disabled. 1 = PDMA channel 2 clock Enabled.
[9]	**CLK1_EN**	**PDMA Controller Channel 1 Clock Enable Control** 0 = PDMA channel 1 clock Disabled. 1 = PDMA channel 1 clock Enabled.
[8]	**CLK0_EN**	**PDMA Controller Channel 0 Clock Enable Control** 0 = PDMA channel 0 clock Disabled. 1 = PDMA channel 0 clock Enabled.
[7:0]	**Reserved**	Reserved.

PDMA Service Selection Control Register 0 (DMA_PDSSR0)

Register	Address	R/W	Description	Reset Value
PDMA_PDSSR0	PDMA_GCR_BA+0x04	R/W	PDMA Service Selection Control Register 0	0x00FF_FFFF

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**SPI2_TXSEL**	**PDMA SPI2 TX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral SPI2 TX. Software can configure the TX channel setting by SPI2_TXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.
[19:16]	**SPI2_RXSEL**	**PDMA SPI2 RX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral SPI2 RX. Software can configure the RX channel setting by SPI2_RXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.
[15:12]	**SPI1_TXSEL**	**PDMA SPI1 TX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral SPI1 TX. Software can configure the TX channel setting by SPI1_TXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.
[11:8]	**SPI1_RXSEL**	**PDMA SPI1 RX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral SPI1 RX. Software can configure the RX channel setting by SPI1_RXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.
[7:4]	**SPI0_TXSEL**	**PDMA SPI0 TX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral SPI0 TX. Software can configure the TX channel setting by SPI0_TXSEL. The channel configuration is the same as SPI0_RXSEL field. Please refer to the explanation of SPI0_RXSEL.

[3:0]	**SPI0_RXSEL**	**PDMA SPI0 RX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral SPI0 RX. Software can change the channel RX setting by SPI0_RXSEL. 0000 = CH0. 0001 = CH1. 0010 = CH2. 0011 = CH3. 0100 = CH4. 0101 = CH5. Others = Reserved. **Note:** For example, SPI0_RXSEL = 0100 means SPI0_RX is connected to PDMA_CH4.

PDMA Service Selection Control Register 1 (PDMA_PDSSR1)

Register	Address	R/W	Description	Reset Value
PDMA_PDSSR1	PDMA_GCR_BA+0x08	R/W	PDMA Service Selection Control Register 1	0x0FFF_FFFF

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

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

[3:0]	**UART0_RXSEL**	**PDMA UART1 RX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral UART0 RX software can change the channel RX setting by UART0_RXSEL. 0000 = CH0. 0001 = CH1. 0010 = CH2. 0011 = CH3. 0100 = CH4. 0101 = CH5. Others = Reserved. **Note:** For example, UART0_RXSEL = 0100 means UART0_RX is connected to PDMA_CH4.

PDMA Global Interrupt Status Register (PDMA_GCRISR)

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

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

Bits	Description	
[31]	**INTR**	**Interrupt Pin Status (Read Only)** This bit is the Interrupt status of PDMA controller.
[30:17]	**Reserved**	Reserved.
[16]	**CRC_INTR**	**Interrupt Pin Status of CRC Controller (Read Only)** This bit is the Interrupt status of CRC controller.
[15:6]	**Reserved**	Reserved.
[5]	**INTR5**	**Interrupt Pin Status of Channel 5 (Read Only)** This bit is the Interrupt status of PDMA channel 5.
[4]	**INTR4**	**Interrupt Pin Status of Channel 4 (Read Only)** This bit is the Interrupt status of PDMA channel 4.
[3]	**INTR3**	**Interrupt Pin Status of Channel 3 (Read Only)** This bit is the Interrupt status of PDMA channel 3.
[2]	**INTR2**	**Interrupt Pin Status of Channel 2 (Read Only)** This bit is the Interrupt status of PDMA channel 2.
[1]	**INTR1**	**Interrupt Pin Status of Channel 1 (Read Only)** This bit is the Interrupt status of PDMA channel 1.
[0]	**INTR0**	**Interrupt Pin Status of Channel 0 (Read Only)** This bit is the Interrupt status of PDMA channel 0.

PDMA Service Selection Control Register 2 (DMA_PDSSR2)

Register	Offset	R/W	Description	Reset Value
PDMA_PDSSR2	PDMA_BA_GCR+0x10	R/W	PDMA Service Selection Control Register 2	0x00FF_FFFF

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PWM3_RXSEL				**PWM2_RXSEL**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PWM1_RXSEL				**PWM0_RXSEL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2S_TXSEL				**I2S_RXSEL**			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**PWM3_RXSEL**	**PDMA PWM3 RX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral PWM3 RX for the capture function. Software can configure the capture channel setting by PWM3_RXSEL. The channel configuration is the same as I2S_RXSEL field. Please refer to the explanation of I2S_RXSEL.
[19:16]	**PWM2_RXSEL**	**PDMA PWM2 RX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral PWM2 RX for the capture function. Software can configure the capture channel setting by PWM2_RXSEL. The channel configuration is the same as I2S_RXSEL field. Please refer to the explanation of I2S_RXSEL.
[15:12]	**PWM1_RXSEL**	**PDMA PWM1 RX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral PWM1 RX for the capture function. Software can configure the capture channel setting by PWM1_RXSEL. The channel configuration is the same as I2S_RXSEL field. Please refer to the explanation of I2S_RXSEL.
[11:8]	**PWM0_RXSEL**	**PDMA PWM0 RX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral PWM0 RX for the capture function. Software can configure the capture channel setting by PWM0_RXSEL. The channel configuration is the same as I2S_RXSEL field. Please refer to the explanation of I2S_RXSEL.
[7:4]	**I2S_TXSEL**	**PDMA I^2S TX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral I^2S TX. Software can configure the TX channel setting by I2S_TXSEL. The channel configuration is the same as I2S_RXSEL field. Please refer to the explanation of I2S_RXSEL.

[3:0]	**I2S_RXSEL**	**PDMA I²S RX Selection** This filed defines which PDMA channel is connected to the on-chip peripheral I²S RX. Software can change the channel RX setting by I2S_RXSEL 0000 = CH0. 0001 = CH1. 0010 = CH2. 0011 = CH3. 0100 = CH4. 0101 = CH5. Others = Reserved. **Note:** For example, I2S_RXSEL = 0100, which means I2S_RX is connected to PDMA_CH4.

6.7 Timer Controller (TMR)

6.7.1 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.

6.7.2 Features

- Four sets of 32-bit timers with 24-bit up counter and one 8-bit prescale counter
- Independent clock source for each timer
- Provides one-shot, periodic, toggle-output and continuous counting operation modes
- 24-bit up counter value is readable through TDR (TDR[23:0])
- Supports event counting function
- 24-bit capture value is readable through TCAP (TCAP[23:0])
- Supports external capture pin event for interval measurement
- Supports external capture pin event to reset 24-bit up counter
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated

6.7.3 Block Diagram

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

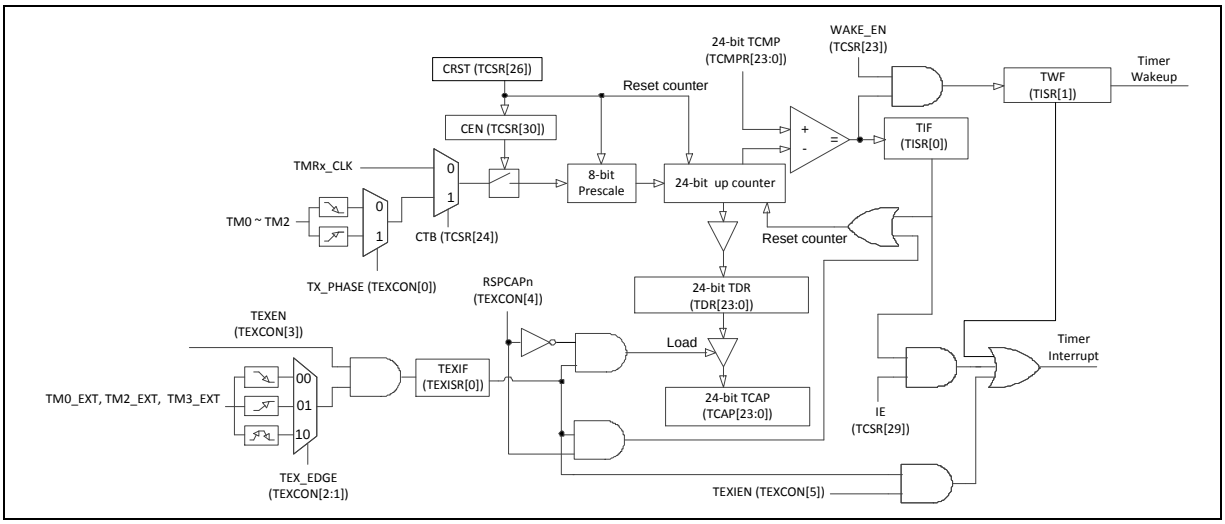
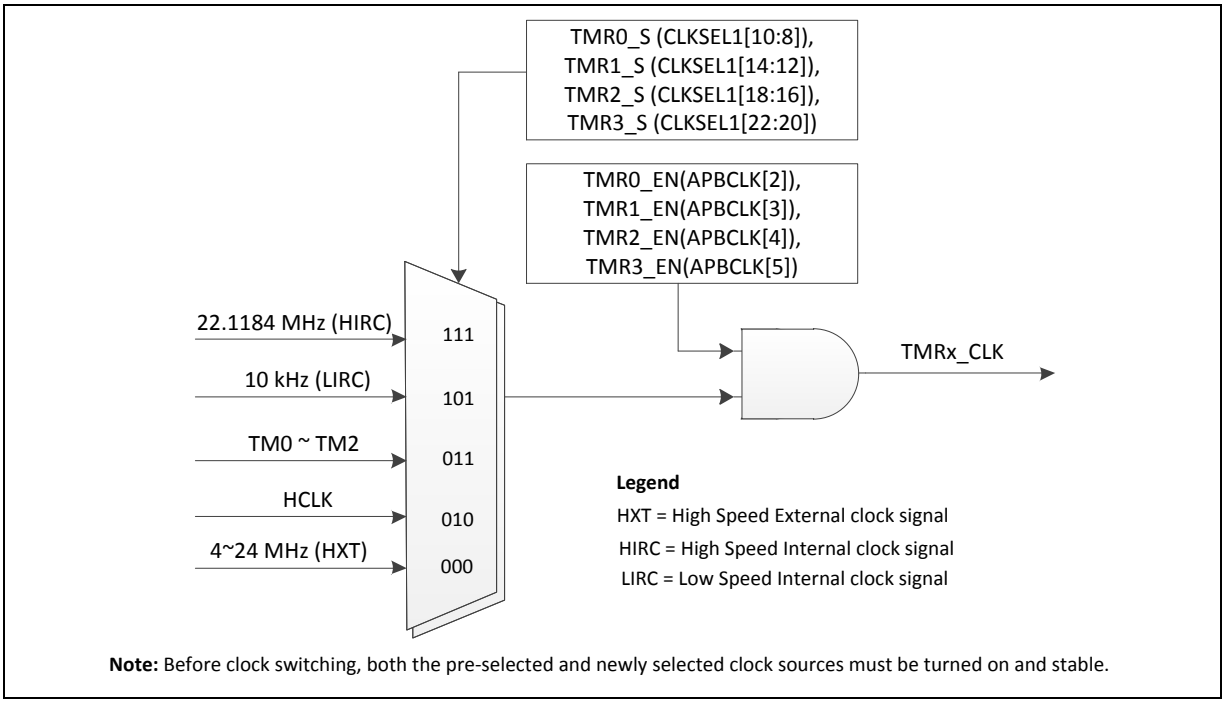


Figure 6-26 Timer Controller Block Diagram



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

Figure 6-27 Clock Source of Timer Controller

6.7.4 Basic Configuration

The timer pins are configured in GPB_MFP and ALT_MFP registers. NUC123xxxAEx provides the alternative of configuring the timer pins in GPB_MFPH and GPB_MFPL registers. (For NUC123xxxAEx, if GPB_MFPH and GPB_MFPL are used as pin multi-function setting, the GPB_MFP and ALT_MFP will become invalid).

The peripheral clock source of Timer0 ~ Timer3 can be enabled in TMRx_EN (APBCLK[5:2]) and selected as different frequency in TMR0_S (CLKSEL1[10:8]) for Timer0, TMR1_S (CLKSEL1[14:12]) for Timer1, TMR2_S (CLKSEL1[18:16]) for Timer2 and TMR3_S (CLKSEL1[22:20]) for Timer3.

6.7.5 Functional Description

Timer controller provides one-shot, period, toggle and continuous counting operation modes. It also provides the event counting function to count the event from external pin and input capture function to capture or reset timer counter value. Each operating function mode is described below.

6.7.5.1 *Timer Interrupt Flag*

Timer controller supports two interrupt flags; one is TIF (TISR[0]) and its set while timer counter value TDR (TDR[23:0]) matches the timer compared value TCMP (TCMPR[23:0]), the other is TEXIF (TEXISR[0]) and its set when the transition on the TMx_EXT pin associated TEX_EDGE (TEXCON[2:1]) setting.

6.7.5.2 *Timer Counting Mode*

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

6.7.5.3 *One–shot Mode*

If timer controller is configured at one-shot mode (TCSR[28:27] is 00) and CEN (TCSR[30]) is set, the timer counter starts up counting. Once the TDR (TDR[23:0]) value reaches TCMP (TCMPR[23:0]) value, the TIF (TISR[0]) will be set to 1, TDR value and CEN bit is cleared automatically by timer controller then timer counting operation stops. In the meantime, if the IE (TCSR[29]) is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also.

6.7.5.4 *Periodic Mode*

If timer controller is configured at periodic mode (TCSR[28:27] is 01) and CEN (TCSR[30]) is set, the timer counter starts up counting. Once the TDR (TDR[23:0]) value reaches TCMP (TCMPR[23:0]) value, the TIF (TISR[0]) will be set to 1, TDR value will be cleared automatically by timer controller and timer counter operates counting again. 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. In this mode, timer controller operates counting and compares with TCMP value periodically until the CEN bit is cleared by user.

6.7.5.5 *Toggle-Output Mode*

If timer controller is configured at toggle-output mode (TCSR[28:27] is 10) and CEN (TCSR[30]) is set, the timer counter starts up counting. The counting operation of toggle-output mode is almost the same as periodic mode, except toggle-output mode has associated TM0 ~ TM2 pin to output signal while specify TIF (TISR[0]) is set. Thus, the toggle-output signal on TM0 ~ TM2 pin is high and changing back and forth with 50% duty cycle.

6.7.5.6 *Continuous Counting Mode*

If timer controller is configured at continuous counting mode (TCSR[28:27] is 11) and CEN (TCSR[30]) is set, the timer counter starts up counting. Once the TDR (TDR[23:0]) value reaches TCMP (TCMPR[23:0]) value, the TIF (TISR[0]) will be set to 1 and TDR value keeps up counting. In the meantime, if the IE (TCSR[29]) 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 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 user programs TCMP value as 200 and clears TIF, the TIF will set to 1 again when TDR value reaches to 200. At last, user programs TCMP as 500 and clears TIF, the TIF will set to 1 again when TDR value reaches to 500.

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

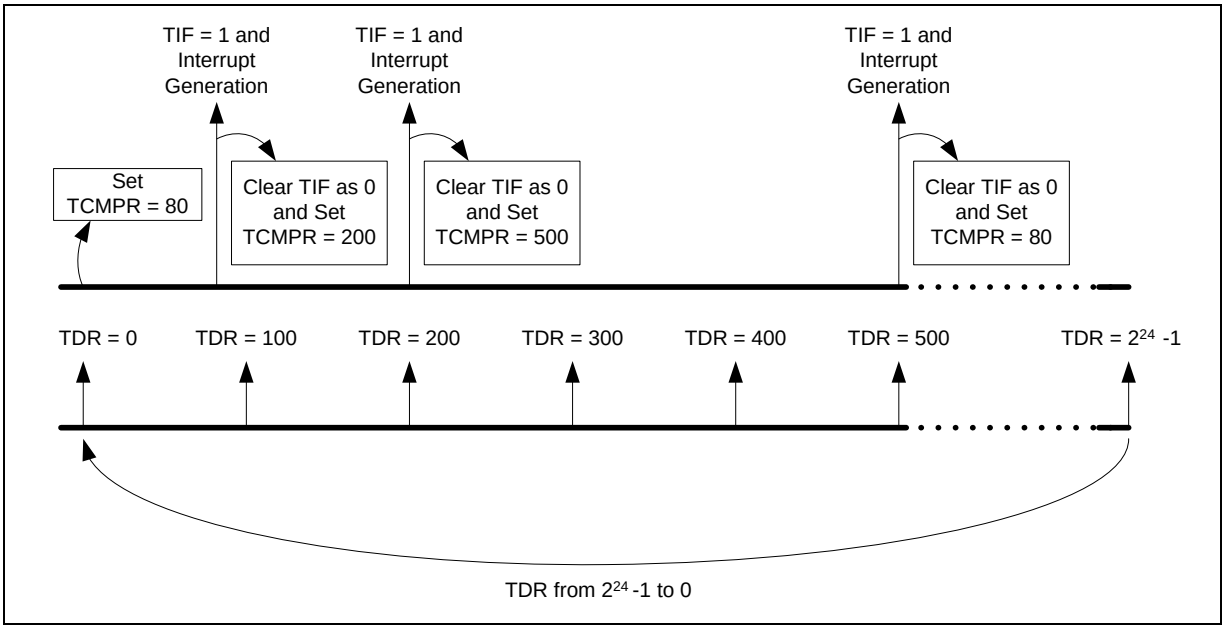


Figure 6-28 Continuous Counting Mode

6.7.5.7 *Event Counting Mode*

Timer controller also provides an application which can count the input event from TMx (x= 0~2) pin and the number of event will reflect to TDR (TDR[23:0]) value. It is also called as event counting function. In this function, CTB (TCSR[24]) should be set and the timer peripheral clock source should be set as HCLK.

User can enable or disable TMx pin de-bounce circuit by setting TCDB (TEXCON[7]). The input event frequency should be less than 1/3 HCLK if TMx pin de-bounce disabled or less than 1/8 HCLK if TMx pin de-bounce enabled to assure the returned TDR value is correct, and user can also select edge detection phase of TMx pin by setting 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 counter value TDR (TDR[23:0]) for TMx pin.

6.7.5.8 *External Capture Mode*

The event capture function is used to load TDR (TDR[23:0]) value to TCAP (TCAP[23:0]) value while edge transition detected on TMx_EXT (x= 0,2,3) pin. In this mode, RSTCAPn (TEXCON[4]) should be as 0 for select TMx_EXT transition is using to trigger event capture function and the timer peripheral clock source should be set as HCLK.

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

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

Users must consider the Timer will keep register TCAP unchanged and drop the new capture value, if the CPU does not clear the TEXIF status.

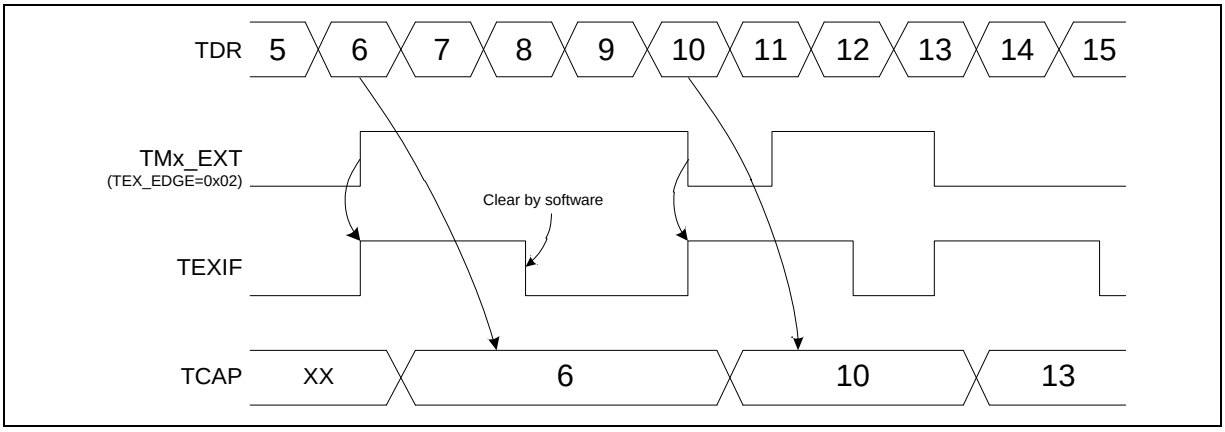


Figure 6-29 External Capture Mode

6.7.5.9 *External Reset Counter Mode*

Timer controller also provides reset counter function to reset TDR (TDR[23:0]) value while edge transition detected on TMx_EXT (x= 0, 2, 3). In this mode, most the settings are the same as event capture mode except RSTCAPn (TEXCON[4]) should be as 1 for select TMx_EXT transition is using to trigger reset counter value.

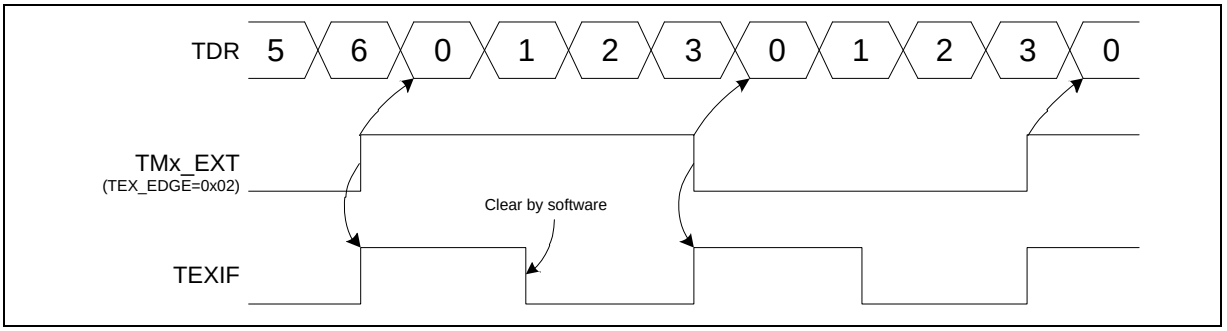


Figure 6-30 External Reset Counter Mod

6.7.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
TMR_BA01 = 0x4001_0000 **TMR_BA23 = 0x4011_0000**				
TCSR0	TMR_BA01+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TCMPR0	TMR_BA01+0x04	R/W	Timer0 Compare Register	0x0000_0000
TISR0	TMR_BA01+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TDR0	TMR_BA01+0x0C	R	Timer0 Data Register	0x0000_0000
TCAP0	TMR_BA01+0x10	R	Timer0 Capture Data Register	0x0000_0000
TEXCON0	TMR_BA01+0x14	R/W	Timer0 External Control Register	0x0000_0000
TEXISR0	TMR_BA01+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TCSR1	TMR_BA01+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TCMPR1	TMR_BA01+0x24	R/W	Timer1 Compare Register	0x0000_0000
TISR1	TMR_BA01+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TDR1	TMR_BA01+0x2C	R	Timer1 Data Register	0x0000_0000
TCAP1	TMR_BA01+0x30	R	Timer1 Capture Data Register	0x0000_0000
TEXCON1	TMR_BA01+0x34	R/W	Timer1 External Control Register	0x0000_0000
TEXISR1	TMR_BA01+0x38	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TCSR2	TMR_BA23+0x00	R/W	Timer2 Control and Status Register	0x0000_0005
TCMPR2	TMR_BA23+0x04	R/W	Timer2 Compare Register	0x0000_0000
TISR2	TMR_BA23+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TDR2	TMR_BA23+0x0C	R	Timer2 Data Register	0x0000_0000
TCAP2	TMR_BA23+0x10	R	Timer2 Capture Data Register	0x0000_0000
TEXCON2	TMR_BA23+0x14	R/W	Timer2 External Control Register	0x0000_0000
TEXISR2	TMR_BA23+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TCSR3	TMR_BA23+0x20	R/W	Timer3 Control and Status Register	0x0000_0005
TCMPR3	TMR_BA23+0x24	R/W	Timer3 Compare Register	0x0000_0000
TISR3	TMR_BA23+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000
TDR3	TMR_BA23+0x2C	R	Timer3 Data Register	0x0000_0000
TCAP3	TMR_BA23+0x30	R	Timer3 Capture Data Register	0x0000_0000
TEXCON3	TMR_BA23+0x34	R/W	Timer3 External Control Register	0x0000_0000
TEXISR3	TMR_BA23+0x38	R/W	Timer3 External Interrupt Status Register	0x0000_0000

6.7.7 Register Description

Timer Control Register (TCSR)

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

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

Bits	Description	
[31]	**DBGACK_TMR**	**ICE Debug Mode Acknowledge Disable (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. **Note:** This bit is write protected. Refer to the REGWRPROT register.
[30]	**CEN**	**Timer Counting Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. **Note1:** In stop status, and then set CEN to 1 will enable the 24-bit up counter to keep counting from the last stop counting value. **Note2:** This bit is auto-cleared by hardware in one-shot mode (TCSR[28:27] = 00) when the timer interrupt flag TIF (TISR[0]) is generated.
[29]	**IE**	**Timer Interrupt Enable Bit** 0 = Timer Interrupt Disabled. 1 = Timer Interrupt Enabled. **Note:** 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 Counting Mode Selection** 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 Counter Reset Bit** Setting this bit will reset the 24-bit up counter value TDR and also force CEN (TCSR[30]) to 0 if CACT (TCSR[25]) is 1. 0 = No effect. 1 = Reset internal 8-bit prescale counter, 24-bit up counter value and CEN bit.
[25]	**CACT**	**Timer Active Status Bit (Read Only)** This bit indicates the up-timer status. 0 = Timer is not active. 1 = Timer is active.
[24]	**CTB**	**Event Counter Mode Enable Bit** This bit is for external counting pin function enabled. 0 = Event counter mode Disabled. 1 = Event counter mode Enabled. **Note:** When timer is used as an event counter, this bit should be set to 1 and select HCLK HCLK as timer clock source.
[23]	**WAKE_EN**	**Wake-up Function Enable Bit** If this bit is set to 1, while timer interrupt flag TIF (TISR[0]) is 1 and IE (TCSR[29]) is enabled, the timer interrupt signal will generate a wake-up trigger event to CPU. 0 = Wake-up function Disabled if timer interrupt signal generated. 1 = Wake-up function Enabled if timer interrupt signal generated.
[22:8]	**Reserved**	Reserved.
[7:0]	**PRESCALE**	**Prescale Counter** Timer input clock or event source is divided by (PRESCALE+1) before it is fed to the timer up counter. If this field is 0 (PRESCALE = 0), then there is no scaling.

Timer Compare Register (TCMPR)

Register	Offset	R/W	Description	Reset Value
TCMPR0	TMR_BA01+0x04	R/W	Timer0 Compare Register	0x0000_0000
TCMPR1	TMR_BA01+0x24	R/W	Timer1 Compare Register	0x0000_0000
TCMPR2	TMR_BA23+0x04	R/W	Timer2 Compare Register	0x0000_0000
TCMPR3	TMR_BA23+0x24	R/W	Timer3 Compare Register	0x0000_0000

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

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

Timer Interrupt Status Register (TISR)

Register	Offset	R/W	Description	Reset Value
TISR0	TMR_BA01+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TISR1	TMR_BA01+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TISR2	TMR_BA23+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TISR3	TMR_BA23+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TWF**	**Timer Wake-up Flag** This bit indicates the interrupt wake-up flag status of timer. 0 = Timer does not cause CPU wake-up. 1 = CPU wake-up from Idle or Power-down mode if timer time-out interrupt signal generated. **Note:** This bit can be cleared by software writing '1'.
[0]	**TIF**	**Timer Interrupt Flag** This bit indicates the interrupt flag status of Timer while 24-bit timer up counter TDR value reaches to TCMP (TCMPR[23:0]) value. 0 = No effect. 1 = TDR value matches the TCMP value. **Note:** This bit can be cleared by software writing '1'.

Timer Data Register (TDR)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**TDR**	**Timer Data Register** This field can be reflected the internal 24-bit timer counter value or external event input counter value from TMx (x=0~2) pin. If CTB (TCSR[24]) is 0, user can read TDR value for getting current 24- bit counter value . If CTB (TCSR[24]) is 1, user can read TDR value for getting current 24- bit event input counter value.

Timer Capture Data Register (TCAP)

Register	Offset	R/W	Description	Reset Value
TCAP0	TMR_BA01+0x10	R/W	Timer0 Capture Data Register	0x0000_0000
TCAP1	TMR_BA01+0x30	R/W	Timer1 Capture Data Register	0x0000_0000
TCAP2	TMR_BA23+0x10	R/W	Timer2 Capture Data Register	0x0000_0000
TCAP3	TMR_BA23+0x30	R/W	Timer3 Capture Data Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**TCAP**	**Timer Capture Data Register** When TEXEN (TEXCON[3]) bit is set, RSTCAPn (TEXCON[4]) bit is 0, and a transition on TMx_EXT (x=0, 2, 3) pin matched the TEX_EDGE (TEXCON[2:1]) setting, TEXIF (TEXISR[0]) will set to 1 and the current timer counter value TDR will be auto-loaded into this TCAP field.

Timer External Control Register (TEXCON)

Register	Offset	R/W	Description	Reset Value
TEXCON0	TMR_BA01+0x14	R/W	Timer0 External Control Register	0x0000_0000
TEXCON1	TMR_BA01+0x34	R/W	Timer1 External Control Register	0x0000_0000
TEXCON2	TMR_BA23+0x14	R/W	Timer2 External Control Register	0x0000_0000
TEXCON3	TMR_BA23+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**	**RSTCAPn**	**TEXEN**	**TEX_EDGE**		**TX_PHASE**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**TCDB**	**Timer Counter Pin De-bounce Enable Bit** 0 = TMx (x= 0~2) pin de-bounce Disabled. 1 = TMx (x= 0~2) pin de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of TMx pin is detected with de-bounce circuit.
[6]	**TEXDB**	**Timer External Capture Pin De-bounce Enable Bit** 0 = TMx_EXT (x= 0, 2, 3) pin de-bounce Disabled. 1 = TMx_EXT (x= 0, 2, 3) pin de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of TMx_EXT pin is detected with de-bounce circuit.
[5]	**TEXIEN**	**Timer External Capture Interrupt Enable Bit** 0 = TMx_EXT (x= 0, 2, 3) pin detection Interrupt Disabled. 1 = TMx_EXT (x= 0, 2, 3) pin detection Interrupt Enabled. **Note:** TEXIEN is used to enable timer external interrupt. If TEXIEN enabled, timer will rise an interrupt when TEXIF (TEXISR[0]) is 1. For example, while TEXIEN = 1, TEXEN = 1, and TEX_EDGE = 00, a 1 to 0 transition on the TMx_EXT pin will cause the TEXIF to be set then the interrupt signal is generated and sent to NVIC to inform CPU.
[4]	**RSTCAPn**	**Capture Function Selection** 0 = External Capture Mode Enabled. 1 = External Reset Mode Enabled. **Note1:** When RSTCAPn is 0, transition on TMx_EXT (x= 0, 2, 3) pin is using to save the 24-bit timer counter value.

		Note2: When RSTCAPn is 1, transition on TMx_EXT (x= 0, 2, 3) pin is using to reset the 24-bit timer counter value.
[3]	**TEXEN**	**Timer External Capture Pin Enable Bit** This bit enables the TMx_EXT pin. 0 =TMx_EXT (x= 0, 2, 3) pin Disabled. 1 =TMx_EXT (x= 0, 2, 3) pin Enabled.
[2:1]	**TEX_EDGE**	**Timer External Capture Pin Edge Detect** 00 = A Falling edge on TMx_EXT (x= 0, 2, 3) pin will be detected. 01 = A Rising edge on TMx_EXT (x= 0, 2, 3) pin will be detected. 10 = Either Rising or Falling edge on TMx_EXT (x= 0, 2, 3) pin will be detected. 11 = Reserved.
[0]	**TX_PHASE**	**Timer External Count Phase** This bit indicates the detection phase of external counting pin TMx (x= 0~2). 0 = A Falling edge of external counting pin will be counted. 1 = A Rising edge of external counting pin will be counted.

Timer External Interrupt Status Register (TISR)

Register	Offset	R/W	Description	Reset Value
TEXISR0	TMR_BA01+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TEXISR1	TMR_BA01+0x38	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TEXISR2	TMR_BA23+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TEXISR3	TMR_BA23+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 timer external capture interrupt flag status. 0 = TMx_EXT (x= 0, 2, 3) pin interrupt did not occur. 1 = TMx_EXT (x= 0, 2, 3) pin interrupt occurred. **Note1:** This bit is cleared by writing 1 to it. **Note2:** When TEXEN (TEXCON[3]) bit is set, RSTCAPn (TEXCON[4]) bit is 0, and a transition on TMx_EXT (x= 0, 2, 3) pin matched the TEX_EDGE (TEXCON[2:1]) setting, this bit will set to 1 by hardware. **Note3:** There is a new incoming capture event detected before CPU clearing the TEXIF status. If the above condition occurred, the Timer will keep register TCAP unchanged and drop the new capture value.

6.8 PWM Generator and Capture Timer (PWM)

6.8.1 Overview

The NuMicro® NUC123 series has 1 set of PWM group supporting 1 set of PWM generators which can be configured as 4 independent PWM outputs, PWM0~PWM3, or as 2 complementary PWM pairs, (PWM0, PWM1), (PWM2, PWM3) with two programmable dead-zone generators. PWM output function can be alternated to capture function.

Each PWM generator has one 8-bit prescaler, one clock divider with 5 divided frequencies (1, 1/2, 1/4, 1/8, 1/16), two PWM Timers including two clock selectors, two 16-bit PWM down-counters for PWM period control, two 16-bit comparators for PWM duty control and one dead-zone generator. The PWM generators provide four independent PWM interrupt flags which are set by hardware when the corresponding PWM period down counter reaches zero.

Each PWM interrupt source with its corresponding enable bit can cause CPU to request PWM interrupt. The PWM generators can be configured as one-shot mode to produce only one PWM cycle signal or auto-reload mode to output PWM waveform continuously. PWM can be used to trigger ADC when operation in center-aligned mode.

6.8.2 Features

PWM function:

- Up to 1 PWM group (PWMA) to support 4 PWM channels or 2 PWM paired channels
- Supports 8-bit prescaler from 1 to 255
- Up to 16-bit resolution PWM timer
- PWM timer supports down and up-down operation type
- One-shot or Auto-reload mode PWM
- PWM Interrupt request synchronized with PWM period or duty
- Supports dead-zone generator with 8-bit resolution for 2 PWM paired channels
- Supports trigger ADC on center point in center-aligned mode

Capture function:

- Supports 4 Capture input channels shared with 4 PWM output channels
- Supports rising or falling capture condition
- Supports rising or falling capture interrupt
- Supports PDMA transfer function for each channel

6.8.3 Block Diagram

Figure 6-31 to Figure 6-34 illustrate the architecture of PWM in pair (i.e. PWM-Timer 0/1 are in one pair and PWM-Timer 2/3 are in the other one).

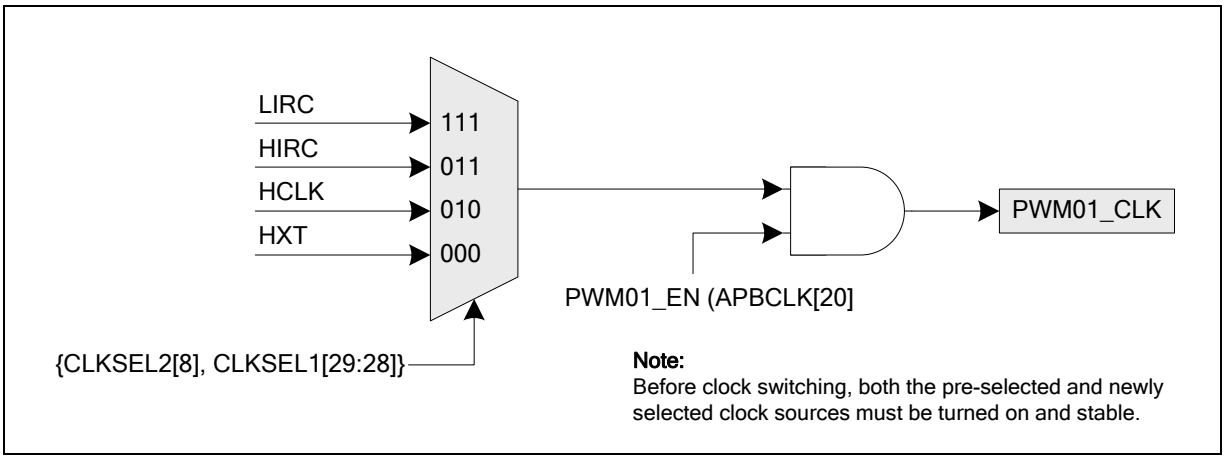


Figure 6-31 PWM Generator 0 Clock Source Control

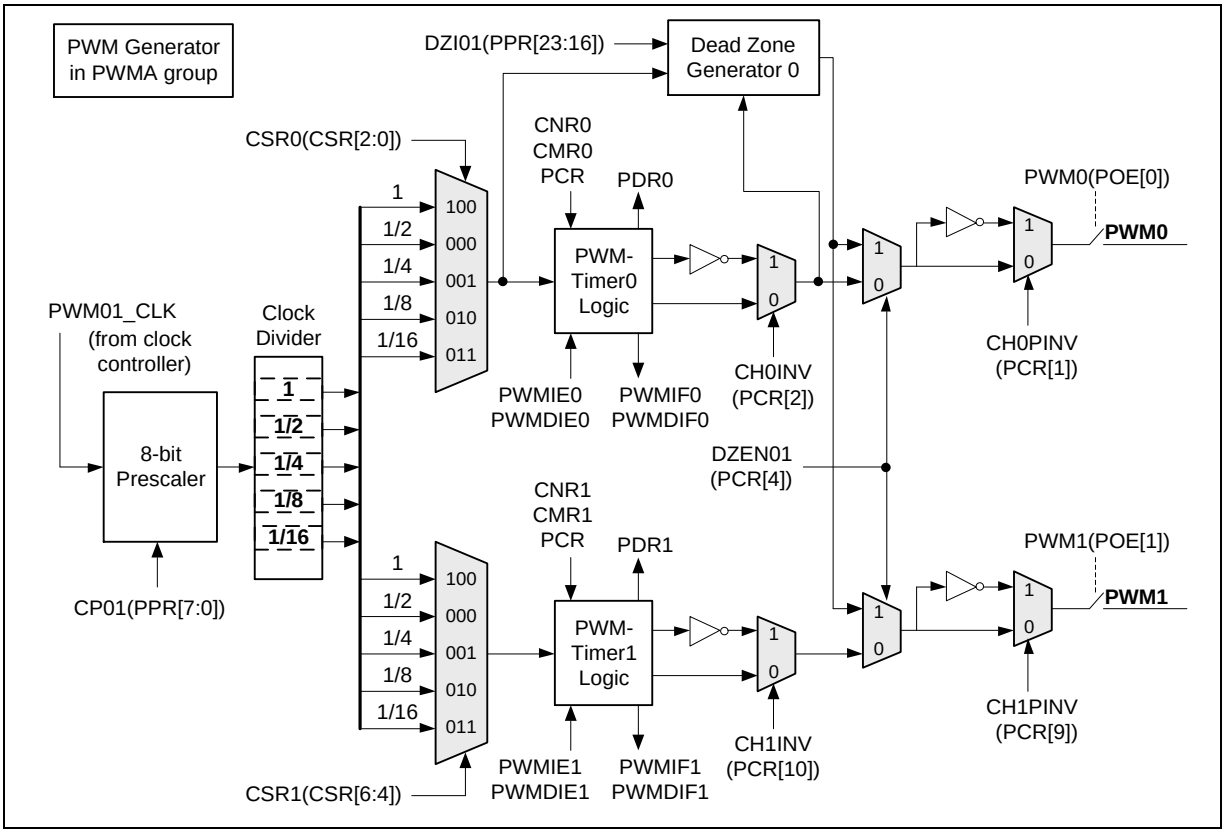


Figure 6-32 PWM Generator 0 Architecture Diagram

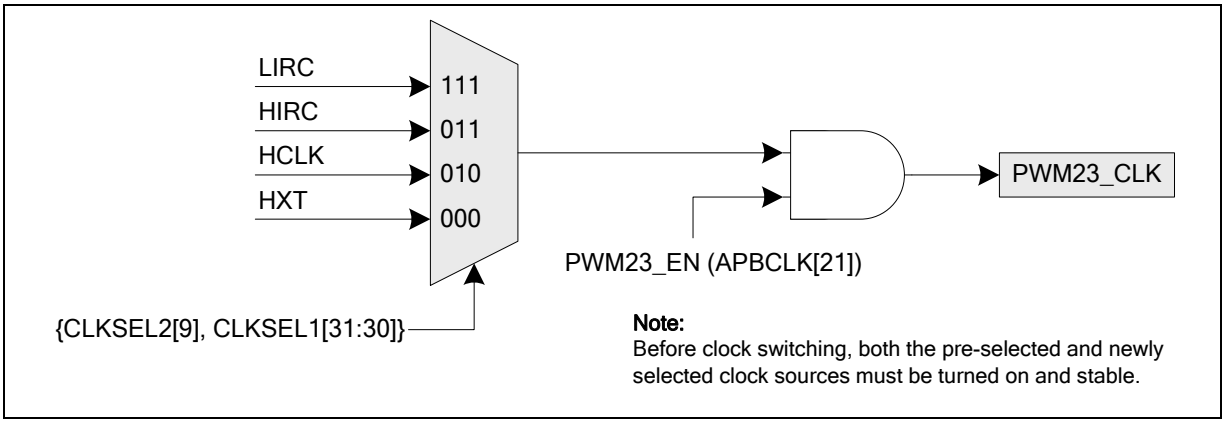


Figure 6-33 PWM Generator 2 Clock Source Control

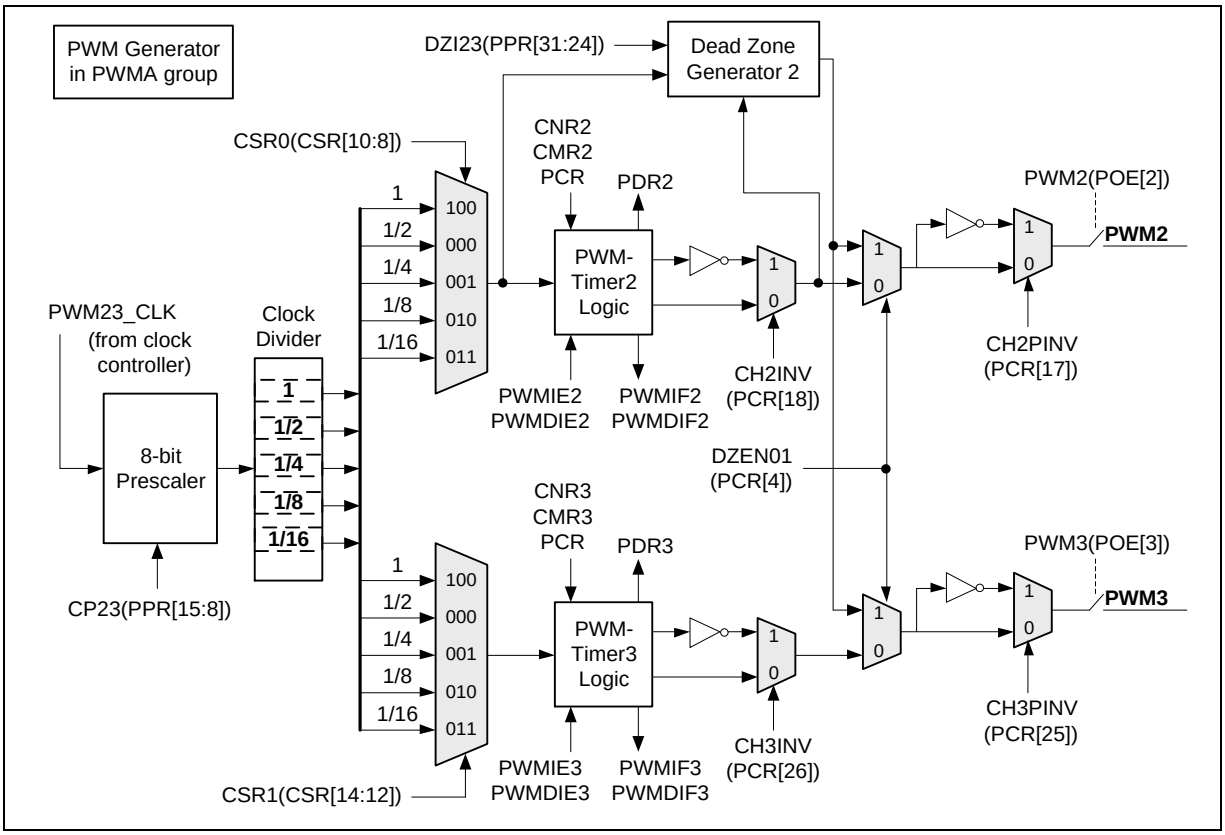


Figure 6-34 PWM Generator 2 Architecture Diagram

6.8.4 Basic Configuration

The PWM pin functions are configured in GPA_MFP, GPB_MFP, GPC_MFP, ALT_MFP and ALT_MFP1 Registers. NUC123xxxAEx provides the alternative of configuring the PWM pins in GPA_MFPH, GPB_MFPH and GPC_MFPH registers. (For NUC123xxxAEx, if GPA_MFPH, GPB_MFPH and GPC_MFPH are used as pin multi-function setting, the GPA_MFP, GPB_MFP, GPC_MFP, ALT_MFP and ALT_MFP1 will become invalid).

The peripheral clock source of PWM01 can be enabled in PWM01_EN (APBCLK[20]), PWM23 can be enabled in PWM23_EN (APBCLK[21]) and selected as different frequency in PWM01_S (CLKSEL2[8] and CLKSEL1[29:28]) for PWM01, PWM23_S (CLKSEL2[9] and CLKSEL1[31:30]) for PWM23.

6.8.5 Functional Description

6.8.5.1 PWM-Timer Operation

The PWM controller supports two operation modes: Edge-aligned and Center-aligned mode.

Edge-aligned PWM (Down-counter)

In Edge-aligned PWM output mode, the 16 bits PWM counter will start down-counting from period value (CNRn register) to zero to finish a PWM period, then restart down-counting from period value to zero again if auto-reload mode is enabled (CHnMOD bit is 1 in PCR register).The value of PWM counter will be compared with comparator value (CMRn register) to control output level of PWM generator. The PWM generator will output low when the value of PWM counter is larger than comparator value and output high when the value of PWM counter is equal or smaller than comparator value.

The PWM period interrupt (PWMIFn (PIIR[3:0])) will be triggered by setting PWMIEn (PIER[3:0]) to 1, and PWM duty interrupt (PWMDIFn(PIIR[11:8])) will be triggered by setting PWMDIEn (PIER[11:8]) to 1.

The PWM period and duty control are configured by PWM down-counter register (CNR) and PWM comparator register (CMR). The new period and comparator value will take effect at the start of next period. The PWM-timer timing operation is shown in Figure 6-36. The pulse width modulation follows the formula below and the legend of PWM-Timer Comparator is shown as Figure 6-35. Note that the corresponding GPIO pins must be configured as PWM function (enabling POE and disabling CAPENR) for the corresponding PWM channel.

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

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

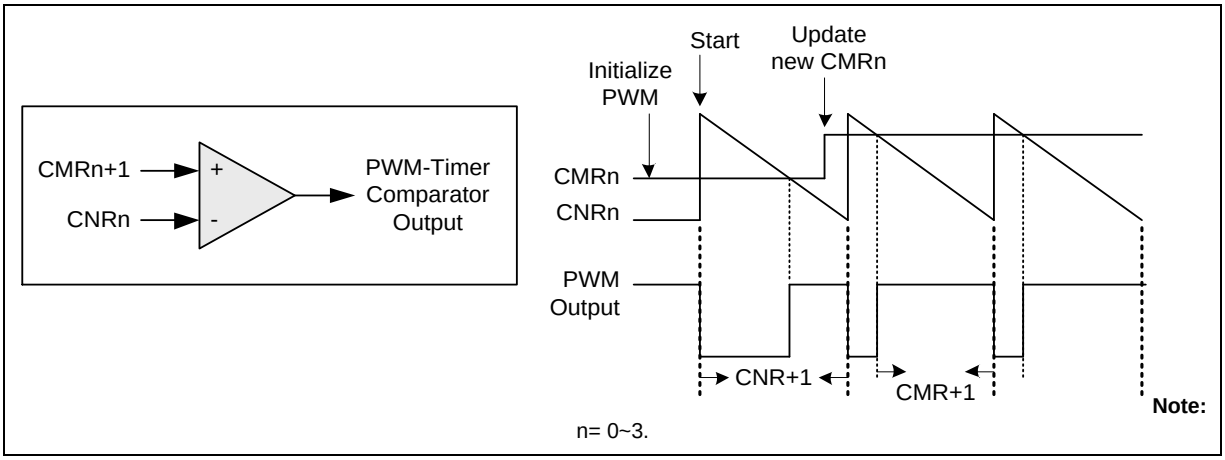


Figure 6-35 Legend of Internal Comparator Output of PWM-Timer

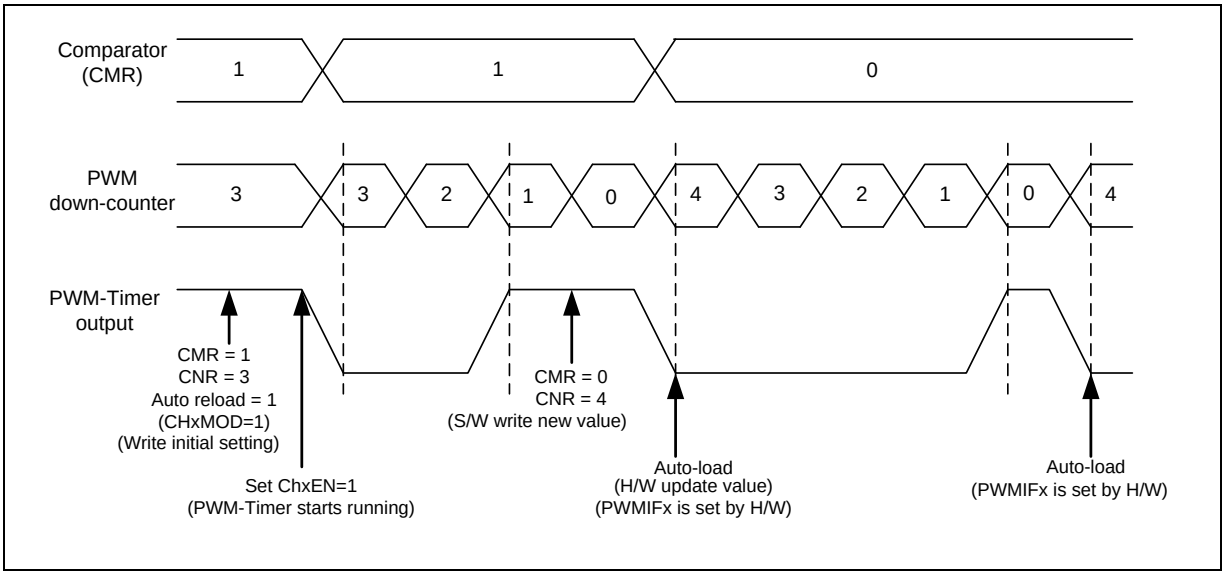


Figure 6-36 PWM-Timer Operation Timing

Center-aligned PWM (up/down-counter)

The Center-aligned PWM signals are produced by the module when the PWM time base is configured in Up/Down Counting mode. The PWM counter will start counting-up from zero to the value of CNRn register and then start counting down to zero to finish a PWM period, then restart next PWM period again if auto-reload mode is enabled (CHnMOD bit is 1 in PCR register).The value of PWM counter will be compared with comparator value (CMRn register) to control output level of PWM generator. The PWM generator will output low when the value of PWM counter is larger than comparator value and output high when the value of PWM counter is equal or smaller than comparator value. Once the PWM counter underflows the new period and comparator value will take effect when PWM timer is operating at auto-reload mode.

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

- PWM frequency = PWMnm_CLK/[(prescale+1)*(clock divider)*(CNR+1)]; where nm could be 01, 23 depending on selected PWM channel.

- Duty ratio = [(2 x CMR) + 1]/[2 x (CNR+1)]
- CMR > CNR: PWM output is always high
- CMR <= CNR: PWM low width= 2 x (CNR-CMR) + 1 unit[1]; PWM high width = (2 x CMR) + 1 unit
- CMR = 0: PWM low width = 2 x CNR + 1 unit; PWM high width = 1 unit

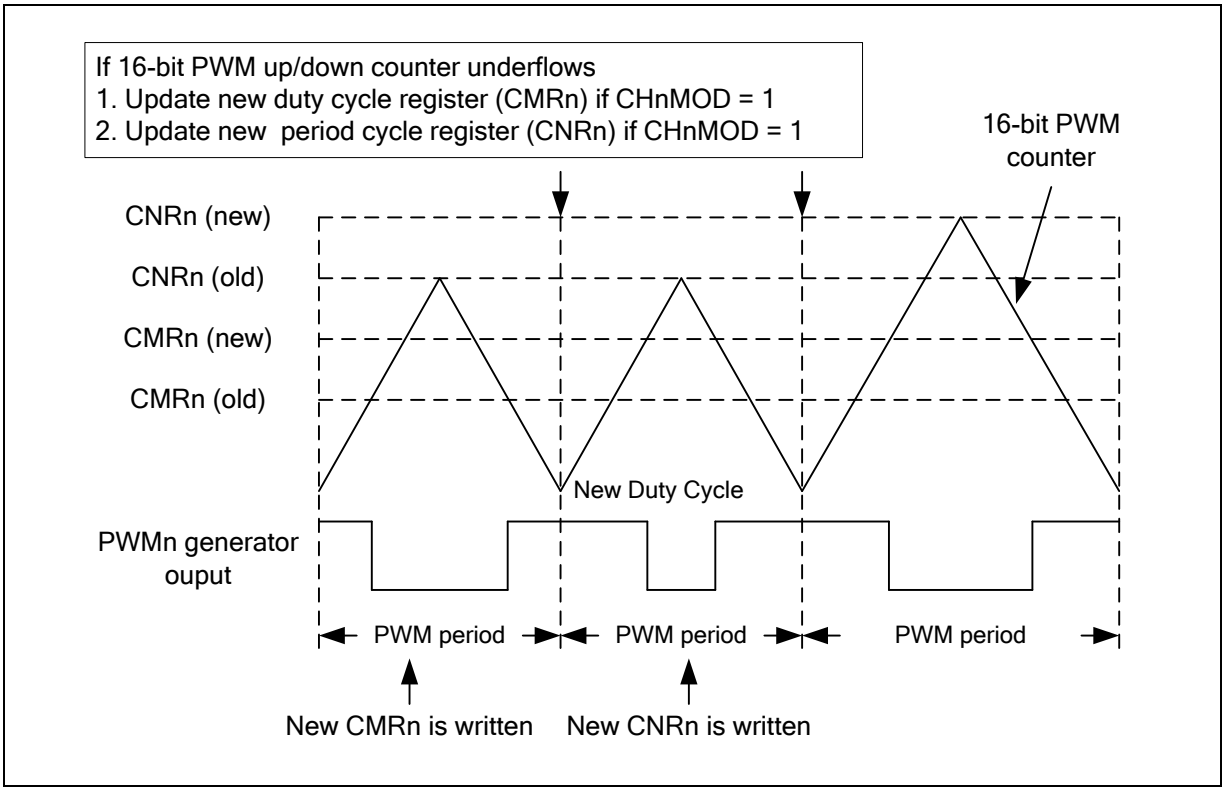


Figure 6-37 Center-Aligned Mode Output Waveform

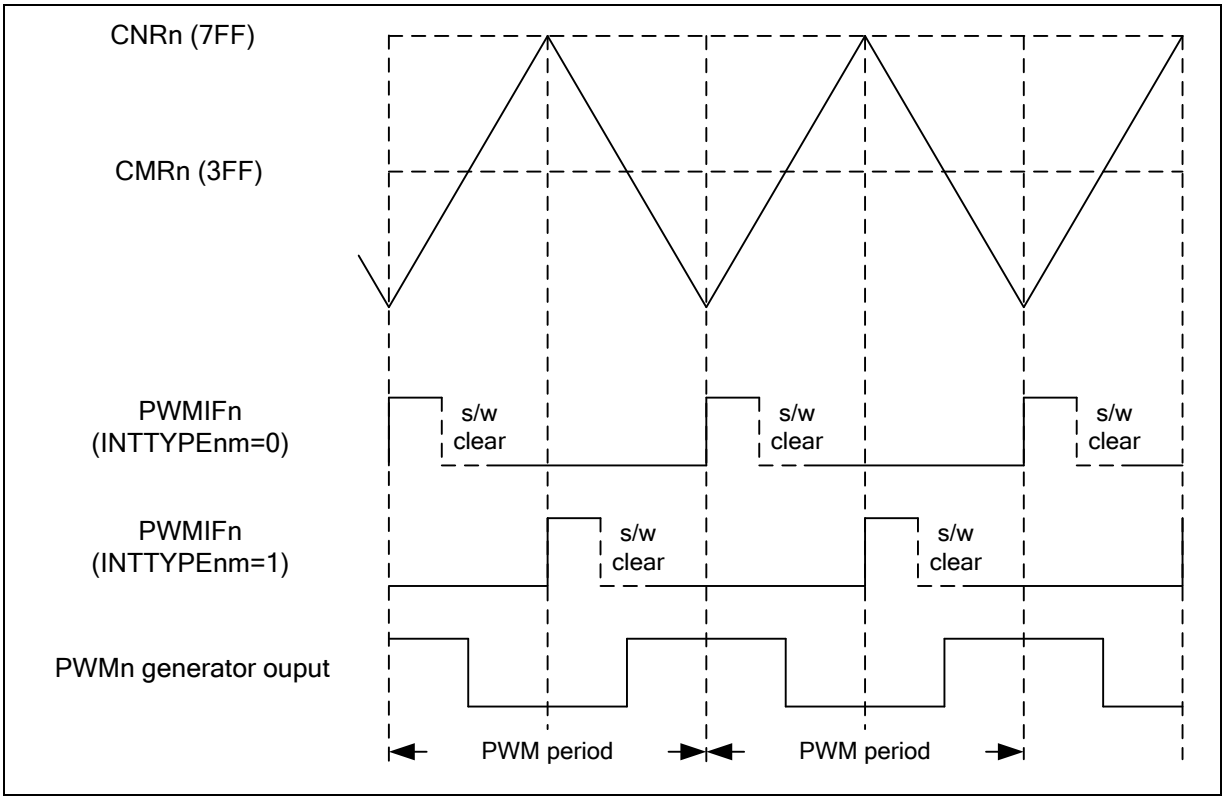


Figure 6-38 PWM Center Aligned Interrupt Generate Timing Waveform

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

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 reload value is updated at the start of next period without affecting current timer operation and avoids glitch at PWM outputs. The PWM counter value can be written into CNRn and the current PWM counter value can be read from PDRn.

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

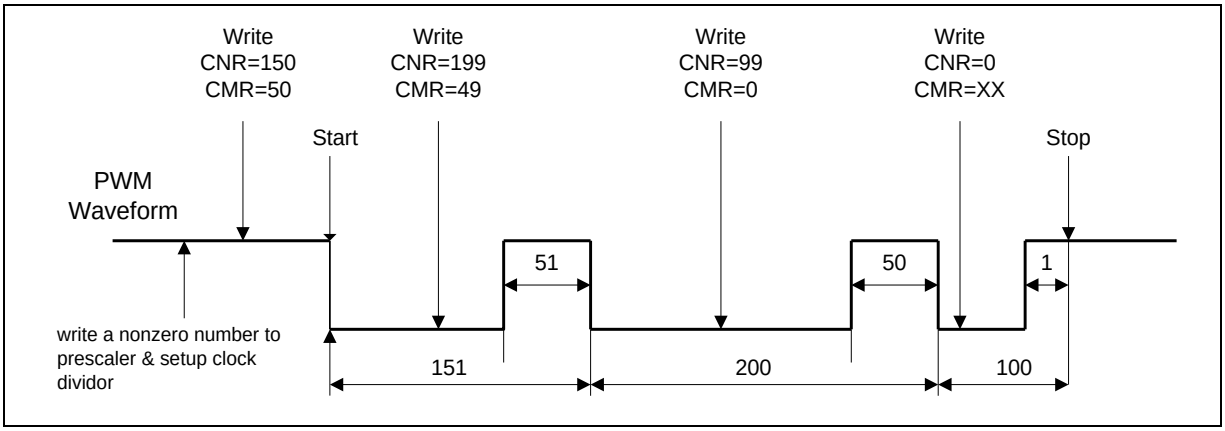


Figure 6-39 PWM Double Buffering Illustration

6.8.5.3 *Modulate Duty Ratio*

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

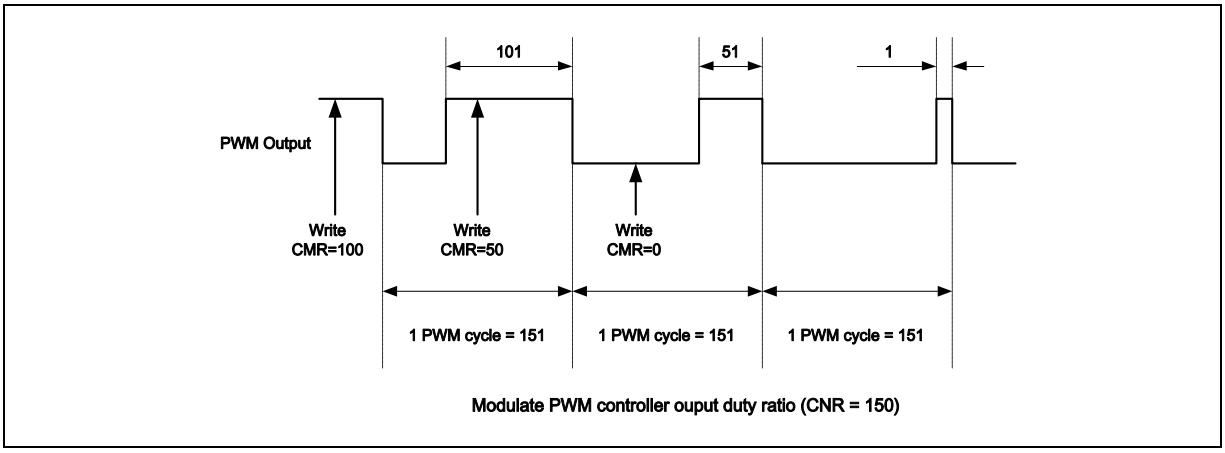


Figure 6-40 PWM Controller Output Duty Ratio in Edge-aligned Mode

6.8.5.4 *Dead-zone Generator*

The dead-zone generator inserts an “off” period called “dead-zone” between the turnings off of one pin to the turning on of the complementary pin of the paired pins. This is to prevent damage to the power switching devices that will be connected to the PWM output pins. The complementary output pair

mode has an 8-bit down counter DZInm (PPR[31:16], where nm could be 01, 23) used to produce the dead-zone insertion. The complementary outputs are delayed until the timer counts down to zero.

The dead-zone can be calculated from the following formula:

dead-zone = PWM_CLK * (DZInm[7:0]+1).

The timing diagram below indicates the dead-zone insertion for pair of PWM signals.

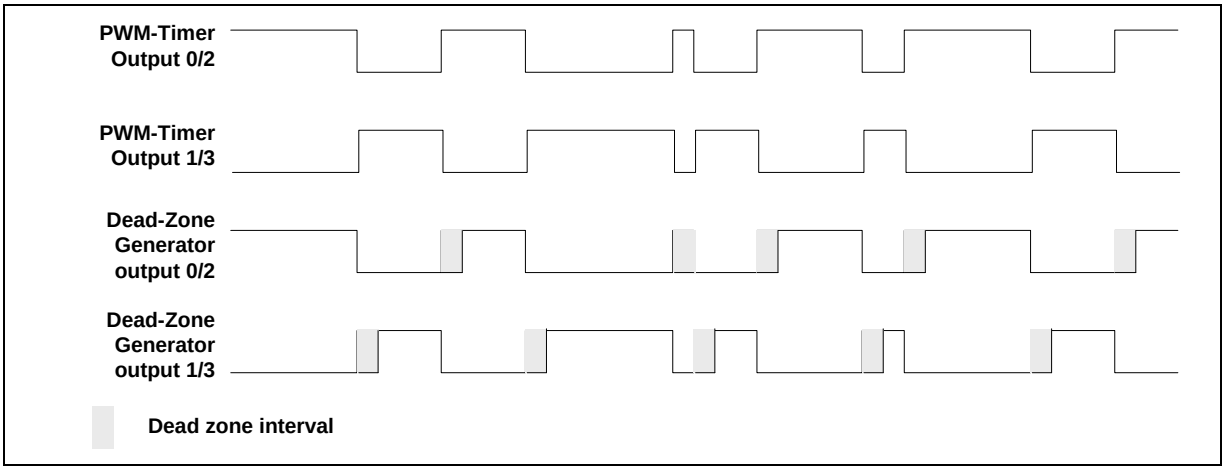


Figure 6-41 Paired-PWM Output with Dead-zone Generation Operation

In Power inverter applications, a dead-zone insertion avoids the upper and lower switches of the half bridge from being active at the same time. Hence the dead-zone control is crucial to proper operation of a system. Some amount of time must be provided between turning off of one PWM output in a complementary pair and turning on the other transistor as the power output devices cannot switch instantaneously.

6.8.5.5 *Polarity Control*

Each PWM port of PWM0 ~ PWM3 has independent polarity control to configure the polarity of active state of PWM output. By default, the PWM output is active high.

The Figure 6-42 shows the initial state before PWM starts with different polarity settings.

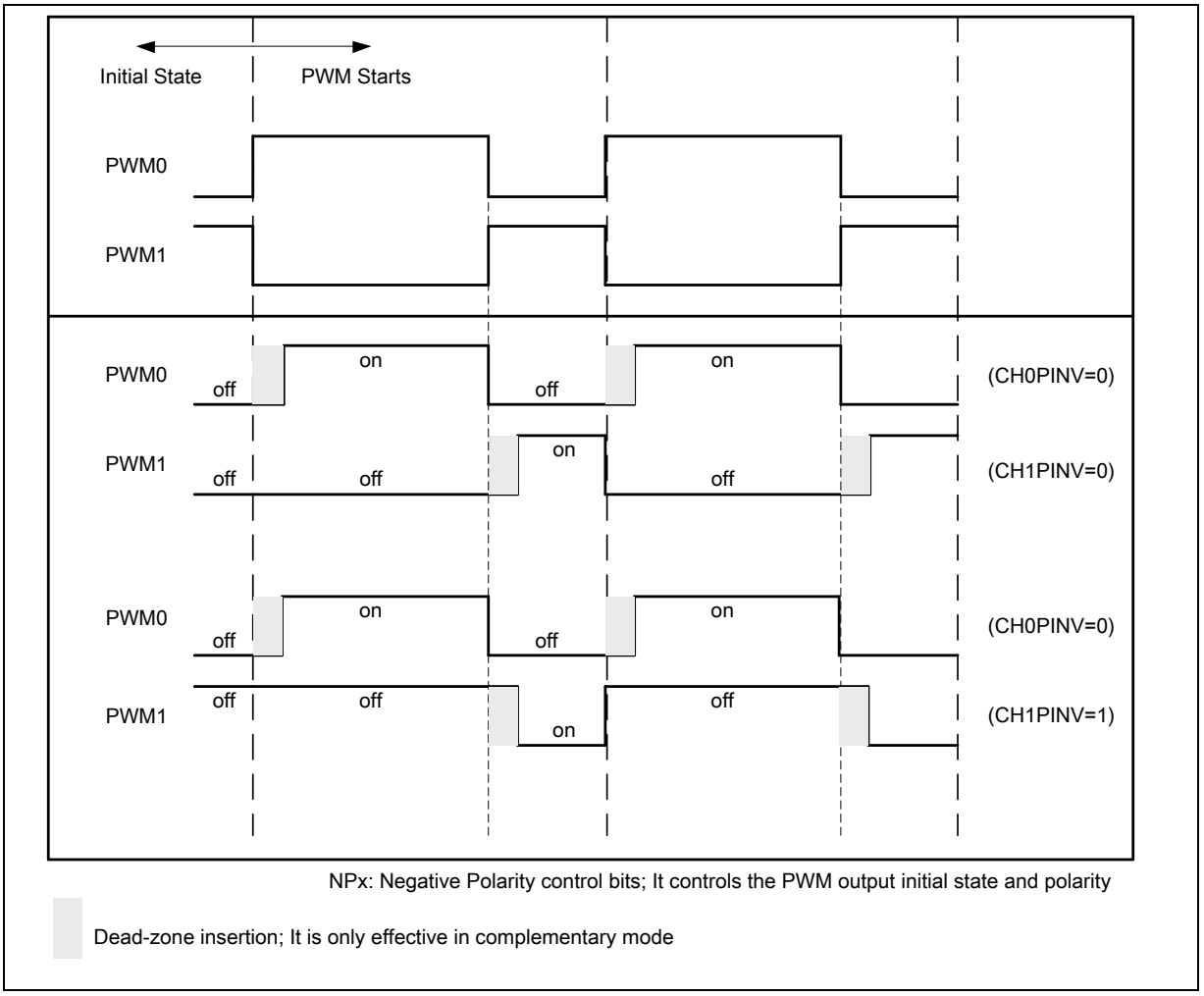


Figure 6-42 Initial State and Polarity Control with Rising Edge Dead-zone Insertion

6.8.5.6 *PWM PDMA function*

PWM supports PDMA transfer function for each channel when operating in Capturing mode. Take channel 0 for example, when the corresponding PDMA enable bit (defined in CAPCTL register) is set, the capturing module will issue a request to PDMA controller when the preceding capture event happened. The PDMA controller will issue ack to capture module and read back CAP0RFPDMA register to memory. By setting CAP0PDMAMOD, PDMA can transfer rising latched data or falling latched data or both of them to memory. When using PDMA to transfer both falling and rising data, remember to set CAP0RFORDER in PWM_CAPCTL to decide the order of transferring data (whether the falling edge latched is first or rising edged latched is first).

6.8.5.7 *PWM Center-aligned Trigger ADC Function*

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

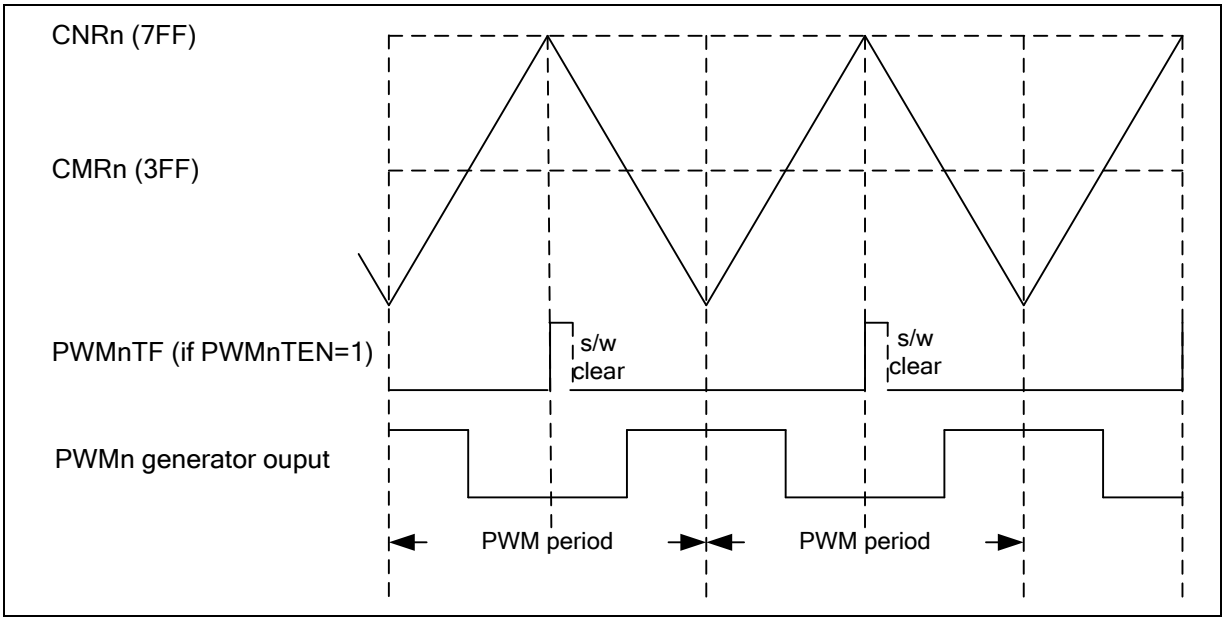


Figure 6-43 Initial State and Polarity Control with Rising Edge Dead-zone Insertion

6.8.5.8 *PWM-Timer Interrupt Architecture*

There are four PWM interrupts, PWM0_INT~PWM3_INT, which are grouped into PWMA_INT for Advanced Interrupt Controller (AIC). That is, PWM 0 and Capture 0 share one interrupt, and PWM1 and Capture 1 share the same interrupt. Therefore, PWM function and Capture function in the same channel cannot be used at the same time. Below demonstrates the architecture of PWM-Timer interrupts.

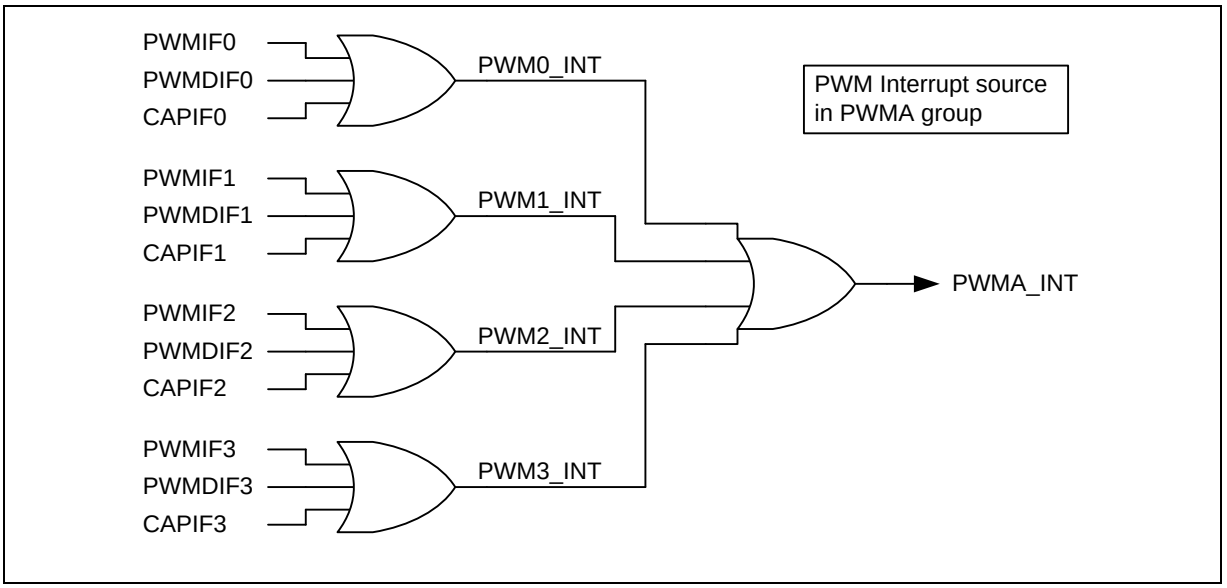


Figure 6-44 PWM Group A PWM-Timer Interrupt Architecture Diagram

6.8.5.9 *Capture Operation*

The alternate feature of the PWM-timer is digital input Capture function. The Capture 0 and PWM 0 share one timer included in PWM 0, and the Capture 1 and PWM 1 share another timer. Therefore

user must setup the PWM-timer before enable Capture feature. After capture feature is enabled, the capture always latches PWM-counter to Capture Rising Latch Register CRLRn when input channel has a rising transition and latches PWM-counter to Capture Falling Latch Register CFLRn when input channel has a falling transition. Capture channel 0 interrupt is programmable by setting CRL_IE0 (CCR0[1]) (Rising latch Interrupt enable) and CFL_IE0 (CCR0[2]) (Falling latch Interrupt enable) to decide the condition of interrupt occur. Capture channel 1 has the same feature by setting CRL_IE1 (CCR0[17]) and CFL_IE1 (CCR0[18]), and etc. Whenever the Capture controller issues a capture interrupt, the corresponding PWM counter will be reloaded with CNRn at this moment. Note that the corresponding GPIO pins must be configured as capture function (disabling POE and enabling CAPENR) for the corresponding capture channel.

The maximum captured frequency that PWM can capture is confined by the capture interrupt latency. When capture interrupt occurred, software will perform at least three steps including: Read PIIR to get interrupt source and Read CRLRn/CFLRn (n=0~3) to get capture value and finally write 1 to clear PIIR to zero. If interrupt latency takes time T0 to finish, the capture signal must not transition during this interval (T0). In this case, the maximum capture frequency will be 1/T0.

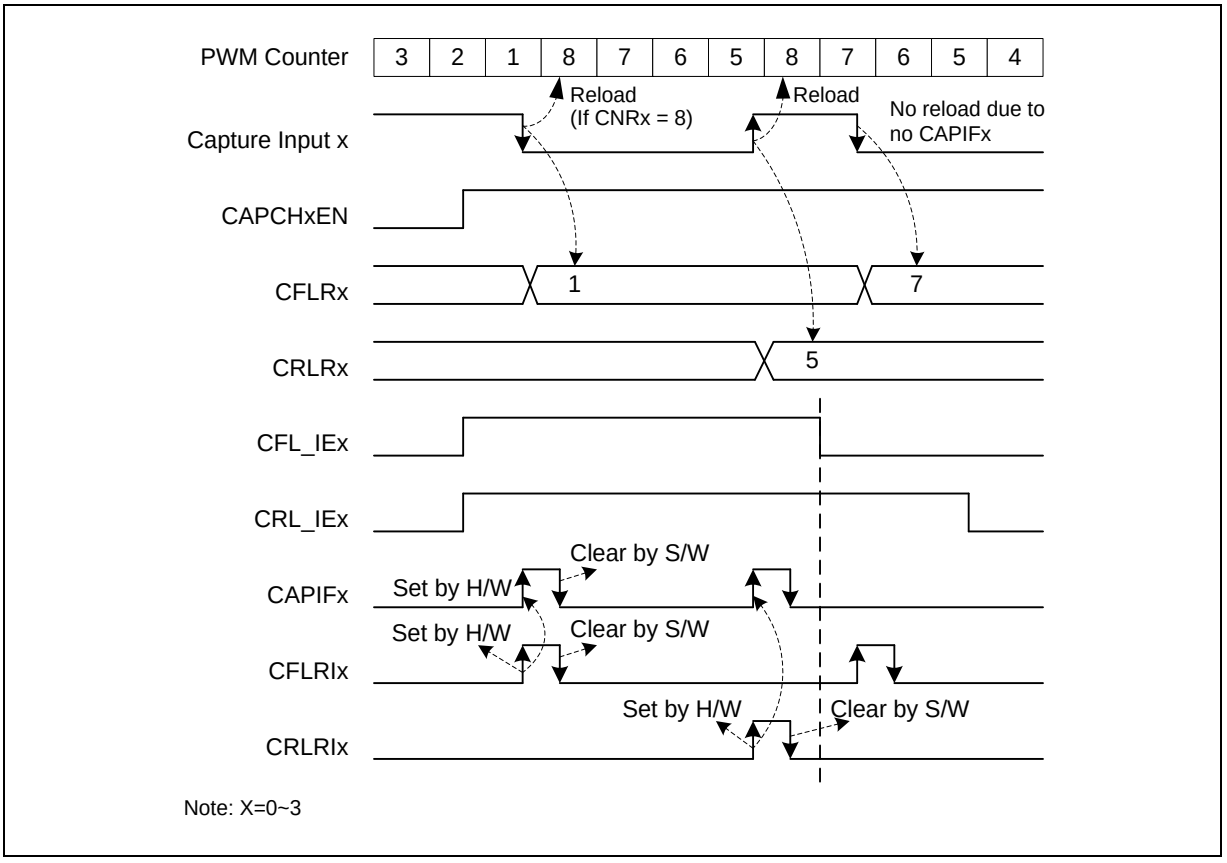


Figure 6-45 Capture Operation Timing

In this case, CNR is 8:

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

6.8.5.10 *PWM-Timer Start Procedure*

The following procedure is recommended for starting a PWM drive:

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

6.8.5.11 *PWM-Timer Stop Procedure*

Method 1:

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

Method 2:

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

Method 3:

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

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

6.8.5.12 *Capture Start Procedure*

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

6.8.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
PWMA_BA = 0x4004_0000 (PWM group A)				
PPR	PWMA_BA+0x00	R/W	PWM Group A Prescaler Register	0x0000_0000
CSR	PWMA_BA+0x04	R/W	PWM Group A Clock Select Register	0x0000_0000
PCR	PWMA_BA+0x08	R/W	PWM Group A Control Register	0x0000_0000
CNR0	PWMA_BA+0x0C	R/W	PWM Group A Counter Register 0	0x0000_0000
CMR0	PWMA_BA+0x10	R/W	PWM Group A Comparator Register 0	0x0000_0000
PDR0	PWMA_BA+0x14	R	PWM Group A Data Register 0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWM Group A Counter Register 1	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWM Group A Comparator Register 1	0x0000_0000
PDR1	PWMA_BA+0x20	R	PWM Group A Data Register 1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM Group A Counter Register 2	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM Group A Comparator Register 2	0x0000_0000
PDR2	PWMA_BA+0x2C	R	PWM Group A Data Register 2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM Group A Counter Register 3	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM Group A Comparator Register 3	0x0000_0000
PDR3	PWMA_BA+0x38	R	PWM Group A Data Register 3	0x0000_0000
PIER	PWMA_BA+0x40	R/W	PWM Group A Interrupt Enable Register	0x0000_0000
PIIR	PWMA_BA+0x44	R/W	PWM Group A Interrupt Indication Register	0x0000_0000
CCR0	PWMA_BA+0x50	R/W	PWM Group A Capture Control Register 0	0x0000_0000
CCR2	PWMA_BA+0x54	R/W	PWM Group A Capture Control Register 2	0x0000_0000
CRLR0	PWMA_BA+0x58	R	PWM Group A Channel 0 Capture Rising Latch Register	0x0000_0000
CFLR0	PWMA_BA+0x5C	R	PWM Group A Channel 0 Capture Falling Latch Register	0x0000_0000
CRLR1	PWMA_BA+0x60	R	PWM Group A Channel 1 Capture Rising Latch Register	0x0000_0000
CFLR1	PWMA_BA+0x64	R	PWM Group A Channel 1 Capture Falling Latch Register	0x0000_0000
CRLR2	PWMA_BA+0x68	R	PWM Group A Channel 2 Capture Rising Latch Register	0x0000_0000
CFLR2	PWMA_BA+0x6C	R	PWM Group A Channel 2 Capture Falling Latch Register	0x0000_0000
CRLR3	PWMA_BA+0x70	R	PWM Group A Channel 3 Capture Rising Latch Register	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM Group A Channel 3 Capture Falling Latch Register	0x0000_0000

CAPENR	PWMA_BA+0x78	R/W	PWM Group A Capture Input 0~3 Enable Register	0x0000_0000
POE	PWMA_BA+0x7C	R/W	PWM Group A Output Enable for channel 0~3	0x0000_0000
TCON	PWMA_BA+0x80	R/W	PWM Group A Trigger Control Register	0x0000_0000
TSTATUS	PWMA_BA+0x84	R/W	PWM Group A Trigger Status Register	0x0000_0000
SYNCBUSY0	PWMA_BA+0x88	R	PWM Group A Channel 0 Synchronous Busy Status Register	0x0000_0000
SYNCBUSY1	PWMA_BA+0x8C	R	PWM Group A Channel 1 Synchronous Busy Status Register	0x0000_0000
SYNCBUSY2	PWMA_BA+0x90	R	PWM Group A Channel 2 Synchronous Busy Status Register	0x0000_0000
SYNCBUSY3	PWMA_BA+0x94	R	PWM Group A Channel 3 Synchronous Busy Status Register	0x0000_0000
CAPPDMACTL	PWMA_BA+0xC0	R/W	PWM Group A PDMA Control Register	0x0000_0000
CAP0PDMA	PWMA_BA+0xC4	R	PWM Group A Channel0 PDMA Data Register	0x0000_0000
CAP1PDMA	PWMA_BA+0xC8	R	PWM Group A Channel1 PDMA Data Register	0x0000_0000
CAP2PDMA	PWMA_BA+0xCC	R	PWM Group A Channel2 PDMA Data Register	0x0000_0000
CAP3PDMA	PWMA_BA+0xD0	R	PWM Group A Channel3 PDMA Data Register	0x0000_0000

6.8.7 Register Description

PWM Pre-Scale Register (PPR)

Register	Offset	R/W	Description	Reset Value
PPR	PWMA_BA+0x00	R/W	PWM Group A Pre-scale Register	0x0000_0000

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

Bits	Description	
[31:24]	**DZI23**	**Dead-zone Interval for Pair of Channel2 and Channel3 (PWM2 and PWM3 Pair for PWM Group A)** These 8 bits determine dead-zone length. The unit time of dead-zone length is received from corresponding CSR bits.
[23:16]	**DZI01**	**Dead-zone Interval for Pair of Channel 0 and Channel 1 (PWM0 and PWM1 Pair for PWM Group A)** These 8 bits determine dead-zone length. The unit time of dead-zone length is received from corresponding CSR bits.
[15:8]	**CP23**	**Clock Prescaler 2 (PWM Timer2 / 3 for Group A)** Clock input is divided by (CP23 + 1) before it is fed to the corresponding PWM-timer. If CP23 = 0, the clock prescaler 2 output clock will be stopped. Thus the corresponding PWM timer will also be stopped.
[7:0]	**CP01**	**Clock Prescaler 0 (PWM-timer 0 / 1 for Group A)** Clock input is divided by (CP01 + 1) before it is fed to the corresponding PWM-timer. If CP01 = 0, the clock prescaler 0 output clock will be stopped. Thus the 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 Group A Clock Selector Register	0x0000_0000

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

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**CSR3**	**PWM Timer 3 Clock Source Selection (PWM Timer 3 for Group A)** Select clock input for PWM timer. 000 = Input clock divided by 2. 001 = Input clock divided by 4. 010 = Input clock divided by 8. 011 = Input clock divided by 16. 100 = Input clock divided by 1.
[11]	**Reserved**	Reserved.
[10:8]	**CSR2**	**PWM Timer 2 Clock Source Selection (PWM Timer 2 for Group A)** Select clock input for PWM timer.Please refer to CSR3.
[7]	**Reserved**	Reserved.
[6:4]	**CSR1**	**PWM Timer 1 Clock Source Selection (PWM Timer 1 for Group A)** Select clock input for PWM timer. Please refer to CSR3.
[3]	**Reserved**	Reserved.
[2:0]	**CSR0**	**PWM Timer 0 Clock Source Selection (PWM Timer 0 for Group A)** Select clock input for PWM timer. Please refer to CSR3.

PWM Control Register (PCR)

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

31	30	29	28	27	26	25	24
PWMTYPE23	**PWMTYPE01**	**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]	**PWMTYPE23**	**PWM23 Aligned Type Selection Bit (PWM2 and PWM3 Pair for PWM Group A)** 0 = Edge-aligned type. 1 = Center-aligned type.
[30]	**PWMTYPE01**	**PWM01 Aligned Type Selection Bit (PWM0 and PWM1 Pair for PWM Group A)** 0 = Edge-aligned type. 1 = Center-aligned type.
[30:28]	**Reserved**	Reserved.
[27]	**CH3MOD**	**PWM-timer 3 Auto-reload/One-shot Mode (PWM Timer 3 for Group A)** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause CNR3 and CMR3 cleared.
[26]	**CH3INV**	**PWM-timer 3 Output Inverter Enable (PWM Timer 3 for Group A)** 0 = Inverter Disabled. 1 = Inverter Enabled.
[25]	**CH3PINV**	**PWM-timer 3 Output Polar Inverse Enable Bit (PWM Timer 3 for Group A)** 0 = PWM3 output polar inverse Disabled. 1 = PWM3 output polar inverse Enabled.
[24]	**CH3EN**	**PWM-timer 3 Enable Bit (PWM Timer 3 for Group A)** 0 = Corresponding PWM-Timer running Stopped. 1 = Corresponding PWM-Timer start run Enabled.
[23:20]	**Reserved**	Reserved.
[19]	**CH2MOD**	**PWM-timer 2 Auto-reload/One-shot Mode (PWM Timer 2 for Group A)** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause CNR2 and CMR2 cleared.

[18]	**CH2INV**	**PWM-timer 2 Output Inverter Enable Bit (PWM Timer 2 for Group A)** 0 = Inverter Disabled. 1 = Inverter Enabled.
[17]	**CH2PINV**	**PWM-timer 2 Output Polar Inverse Enable Bit (PWM Timer 2 for Group A)** 0 = PWM2 output polar inverse Disabled. 1 = PWM2 output polar inverse Enabled.
[16]	**CH2EN**	**PWM-timer 2 Enable Bit (PWM Timer 2 for Group A)** 0 = Corresponding PWM-Timer running Stopped. 1 = Corresponding PWM-Timer start run Enabled.
[15:12]	**Reserved**	Reserved.
[11]	**CH1MOD**	**PWM-timer 1 Auto-reload/One-shot Mode (PWM Timer 1 for Group A)** 0 = One-shot mode. 1 = Auto-load mode. **Note:** If there is a transition at this bit, it will cause CNR1 and CMR1 cleared.
[10]	**CH1INV**	**PWM-timer 1 Output Inverter Enable Bit (PWM Timer 1 for Group A)** 0 = Inverter Disabled. 1 = Inverter Enabled.
[9]	**CH1PINV**	**PWM-timer 1 Output Polar Inverse Enable Bit (PWM Timer 1 for Group A)** 0 = PWM1 output polar inverse Disabled. 1 = PWM1 output polar inverse Enabled.
[8]	**CH1EN**	**PWM-timer 1 Enable Bit (PWM Timer 1 for Group A)** 0 = Corresponding PWM-Timer running Stopped. 1 = Corresponding PWM-Timer start run Enabled.
[7:6]	**Reserved**	Reserved.
[5]	**DZEN23**	**Dead-zone 2 Generator Enable Bit (PWM2 and PWM3 Pair for PWM Group A)** 0 = Dead-zone 2 generator Disabled. 1 = Dead-zone 2 generator Enabled. **Note:** When dead-zone generator is enabled, the pair of PWM2 and PWM3 becomes a complementary pair for PWM group A.
[4]	**DZEN01**	**Dead-zone 0 Generator Enable Bit (PWM0 and PWM1 Pair for PWM Group a)** 0 = Dead-zone 0 generator Disabled. 1 = Dead-zone 0 generator Enabled. **Note:** When dead-zone generator is enabled, the pair of PWM0 and PWM1 becomes a complementary pair for PWM group A.
[3]	**CH0MOD**	**PWM-timer 0 Auto-reload/One-shot Mode (PWM Timer 0 for Group A)** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause CNR0 and CMR0 cleared.
[2]	**CH0INV**	**PWM-timer 0 Output Inverter Enable Bit (PWM Timer 0 for Group A)** 0 = Inverter Disabled. 1 = Inverter Enabled.
[1]	**CH0PINV**	**PWM-timer 0 Output Polar Inverse Enable Bit (PWM Timer 0 for Group A)** 0 = PWM0 output polar inverse Disabled. 1 = PWM0 output polar inverse Enabled.

[0]	**CH0EN**	**PWM-timer 0 Enable Bit (PWM Timer 0 for Group A)** 0 = Corresponding PWM-Timer running Stopped. 1 = Corresponding PWM-Timer start run Enabled.

PWM Counter Register 0-3 (CNR0-3)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNR**	**PWM Timer Loaded Value** CNR determines the PWM period. PWM frequency = PWMnm_CLK/[(prescale+1)*(clock divider)*(CNR+1)]; where nm could be 01 or 23, depending on the selected PWM channel. For Edge-aligned mode: ● 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 mode: ● 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 to CNR will take effect in the next PWM cycle. **Note2:** When PWM operating at center-aligned type, CNR value should be set between 0x0000 to 0xFFFE. If CNR equal to 0xFFFF, the PWM will work unpredictable. **Note3:** When CNR value is set to 0, PWM output is always high.

PWM Comparator Register 0-3 (CMR0-3)

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

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

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

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

Register	Offset	R/W	Description	Reset Value
PDR0	PWMA_BA0+0x14	R	PWM Group A Data Register 0	0x0000_0000
PDR1	PWMA_BA0+0x20	R	PWM Group A Data Register 1	0x0000_0000
PDR2	PWMA_BA0+0x2C	R	PWM Group A Data Register 2	0x0000_0000
PDR3	PWMA_BA0+0x38	R	PWM Group A Data Register 3	0x0000_0000

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

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

PWM Interrupt Enable Register (PIER)

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**INTTYPE23**	**INTTYPE01**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**PWMDIE3**	**PWMDIE2**	**PWMDIE1**	**PWMDIE0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PWMIE3**	**PWMIE2**	**PWMIE1**	**PWMIE0**

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**INTTYPE23**	**PWM23 Interrupt Type Selection Bit (PWM2 and PWM3 Pair for PWM Group A)** 0 = PWMIFn will be set if PWM counter underflow. 1 = PWMIFn will be set if PWM counter matches CNRn register. **Note:** This bit is effective when PWM is in center-aligned mode only.
[16]	**INTTYPE01**	**PWM01 Interrupt Type Selection Bit (PWM0 and PWM1 Pair for PWM Group A)** 0 = PWMIFn will be set if PWM counter underflow. 1 = PWMIFn will be set if PWM counter matches CNRn register. **Note:** This bit is effective when PWM is in center-aligned mode only.
[11]	**PWMDIE3**	**PWM Channel 3 Duty Interrupt Enable Bit** 0 = PWM channel 3 duty interrupt Disabled. 1 = PWM channel 3 duty interrupt Enabled.
[10]	**PWMDIE2**	**PWM Channel 2 Duty Interrupt Enable Bit** 0 = PWM channel 2 duty interrupt Disabled. 1 = PWM channel 2 duty interrupt Enabled.
[9]	**PWMDIE1**	**PWM Channel 1 Duty Interrupt Enable Bit** 0 = PWM channel 1 duty interrupt Disabled. 1 = PWM channel 1 duty interrupt Enabled.
[8]	**PWMDIE0**	**PWM Channel 0 Duty Interrupt Enable Bit** 0 = PWM channel 0 duty interrupt Disabled. 1 = PWM channel 0 duty interrupt Enabled.
[7:4]	**Reserved**	Reserved.
[3]	**PWMIE3**	**PWM Channel 3 Interrupt Enable Bit** 0 = PWM channel 3 interrupt Disabled. 1 = PWM channel 3 interrupt Enabled.

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

PWM Interrupt Indication Register (PIIR)

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

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**PWMDIF3**	**PWM Channel 3 Duty Interrupt Flag** Flag is set by hardware when channel 3 PWM counter down count and reaches CMR3. **Note1:** If CMR is equal to CNR, this flag does not work. **Note2:** This bit can be cleared to 0 by software writing '1'.
[10]	**PWMDIF2**	**PWM Channel 2 Duty Interrupt Flag** Flag is set by hardware when channel 2 PWM counter down count and reaches CMR2. **Note1:** If CMR is equal to CNR, this flag does not work. **Note2:** This bit can be cleared to 0 by software writing '1'.
[9]	**PWMDIF1**	**PWM Channel 1 Duty Interrupt Flag** Flag is set by hardware when channel 1 PWM counter down count and reaches CMR1. **Note1:** If CMR is equal to CNR, this flag does not work. **Note2:** This bit can be cleared to 0 by software writing '1'.
[8]	**PWMDIF0**	**PWM Channel 0 Duty Interrupt Flag** Flag is set by hardware when channel 0 PWM counter down count and reaches CMR0. **Note1:** If CMR is equal to CNR, this flag does not work. **Note2:** This bit can be cleared to 0 by software writing '1'.
[7:4]	**Reserved**	Reserved.
[3]	**PWMIF3**	**PWM Channel 3 Interrupt Flag** This bit is set by hardware when PWM3 counter reaches the requirement of interrupt (depending on INTTYPE23 bit of PIER register) if PWM3 interrupt enable bit (PWMIE3) is 1. **Note:** This bit can be cleared to 0 by software writing '1'.

[2]	**PWMIF2**	**PWM Channel 2 Interrupt Flag** This bit is set by hardware when PWM2 counter reaches the requirement of interrupt (depending on INTTYPE23 bit of PIER register) if PWM2 interrupt enable bit (PWMIE2) is 1. **Note:** This bit can be cleared to 0 by software writing '1'.
[1]	**PWMIF1**	**PWM Channel 1 Interrupt Flag** This bit is set by hardware when PWM1 counter reaches the requirement of interrupt (depending on INTTYPE01 bit of PIER register) if PWM1 interrupt enable bit (PWMIE1) is 1. **Note:** This bit can be cleared to 0 by software writing '1'.
[0]	**PWMIF0**	**PWM Channel 0 Interrupt Flag** This bit is set by hardware when PWM0 counter reaches the requirement of interrupt (depending on INTTYPE01 bit of PIER register) if PWM0 interrupt enable bit (PWMIE0) is 1. **Note:** This bit can be cleared to 0 by software writing '1'.

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

Capture Control Register 0 (CCR0)

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

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

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

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

Capture Control Register 2 (CCR2)

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

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

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

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

Capture Rising Latch Register 0-3 (CRLR0-3)

Register	Offset	R/W	Description	Reset Value
CRLR0	PWMA_BA+0x58	R	PWM Group A Channel 0 Capture Rising Latch Register	0x0000_0000
CRLR1	PWMA_BA+0x60	R	PWM Group A Channel 1 Capture Rising Latch Register	0x0000_0000
CRLR2	PWMA_BA+0x68	R	PWM Group A Channel 2 Capture Rising Latch Register	0x0000_0000
CRLR3	PWMA_BA+0x70	R	PWM Group A Channel 3 Capture Rising Latch 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**
CRLR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CRLR							

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

Capture Falling Latch Register 0-3 (CFLR0-3)

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

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

Capture Input Enable Register (CAPENR)

Register	Offset	R/W	Description	Reset Value
CAPENR	PWMA_BA+0x78	R/W	PWM Group A Capture Input 0~3 Enable Register	0x0000_0000

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

Bits	Description	
[3:0]	**CAPENR**	**Capture Input Enable Register** There are four capture inputs from pad. Bit0~Bit3 are used to control each input enable or disable. If enabled, PWMn multi-function pin input will affect its input capture function; If Disabled, PWMn multi-function pin input does not affect input capture function. xxx1 = Capture channel 0 is from pin PA.12. xx1x = Capture channel 1 is from pin PA.13. x1xx = Capture channel 2 is from pin PA.14. 1xxx = Capture channel 3 is from pin PA.15.

PWM Output Enable Register (POE)

Register	Offset	R/W	Description	Reset Value
POE	PWMA_BA+0x7C	R/W	PWM Group A Output Enable Register for channel 0~3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PWM3**	**PWM2**	**PWM1**	**PWM0**

Bits	Description	
[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 Group A 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PWM3TEN**	**PWM2TEN**	**PWM1TEN**	**PWM0TEN**

Bits	Description	
[3]	**PWM3TEN**	**Channel 3 Center-aligned Trigger Enable Register** PWM can trigger ADC to start conversion when PWM counter counts up to CNR if this bit is set to 1. 0 = PWM channel 3 trigger ADC function Disabled. 1 = PWM channel 3 trigger ADC function Enabled. **Note:** This function is only supported when PWM operating in Center-aligned mode.
[2]	**PWM2TEN**	**Channel 2 Center-aligned Trigger Enable Register** PWM can trigger ADC to start conversion when PWM counter counts up to CNR if this bit is set to 1.0 = PWM channel 2 trigger ADC function Disabled. 1 = PWM channel 2 trigger ADC function Enabled. **Note:** This function is only supported when PWM operating in Center-aligned mode.
[1]	**PWM1TEN**	**Channel 1 Center-aligned Trigger Enable Register** PWM can trigger ADC to start conversion when PWM counter counts up to CNR if this bit is set to 1. 0 = PWM channel 1 trigger ADC function Disabled. 1 = PWM channel 1 trigger ADC function Enabled. **Note:** This function is only supported when PWM operating in Center-aligned mode
[0]	**PWM0TEN**	**Channel 0 Center-aligned Trigger Enable Register** PWM can trigger ADC to start conversion when PWM counter counts up to CNR if this bit is set to 1. 0 = PWM channel 0 trigger ADC function Disabled. 1 = PWM channel 0 trigger ADC function Enabled. **Note:** This function is only supported when PWM operating in Center-aligned mode.

PWM Trigger Status Register (TSTATUS)

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

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

Bits	Description	
[3]	**PWM3TF**	**Channel 3 Center-aligned Trigger Flag** For Center-aligned operating mode, this bit is set to 1 by hardware when PWM counter up count to CNR if PWM3TEN bit is set to 1. After this bit is set to 1, ADC will start conversion if ADC triggered source is selected by PWM. **Note:** This bit can be cleared to 0 by software writing '1'.
[2]	**PWM2TF**	**Channel 2 Center-aligned Trigger Flag** For Center-aligned operating mode, this bit is set to 1 by hardware when PWM counter up count to CNR if PWM2TEN bit is set to 1. After this bit is set to 1, ADC will start conversion if ADC triggered source is selected by PWM. **Note:** This bit can be cleared to 0 by software writing '1'.
[1]	**PWM1TF**	**Channel 1 Center-aligned Trigger Flag** For Center-aligned operating mode, this bit is set to 1 by hardware when PWM counter up count to CNR if PWM1TEN bit is set to 1. After this bit is set to 1, ADC will start conversion if ADC triggered source is selected by PWM. **Note:** This bit can be cleared to 0 by software writing '1'.
[0]	**PWM0TF**	**Channel 0 Center-aligned Trigger Flag** For Center-aligned operating mode, this bit is set to 1 by hardware when PWM counter up count to CNR if PWM0TEN bit is set to 1. After this bit is set to 1, ADC will start conversion if ADC triggered source is selected by PWM. **Note:** This bit can be cleared to 0 by software writing '1'.

PWM0 Synchronous Busy Status Register 0-3 (SYNCBUSY0-3)

Register	Offset	R/W	Description	Reset Value
SYNCBUSY0	PWMA_BA+0x88	R	PWM Group A Channel 0 Synchronous Busy Status Register	0x0000_0000
SYNCBUSY1	PWMA_BA+0x8C	R	PWM Group A Channel 1 Synchronous Busy Status Register	0x0000_0000
SYNCBUSY2	PWMA_BA+0x90	R	PWM Group A Channel 2 Synchronous Busy Status Register	0x0000_0000
SYNCBUSY3	PWMA_BA+0x94	R	PWM Group A Channel 3 Synchronous Busy Status Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**S_BUSY**	**PWM Synchronous Busy** When software writes CNRn/CMRn/PPR or switch PWMn operation mode (PCR[3, 11, 19, 27]), PWM will have a busy time to update these values completely because PWM clock may be different from system clock domain. Software needs to check this busy status before writes CNRn/CMRn/PPR or switch PWMn operation mode (PCR[3, 11, 19, 27]) to make sure previous setting has been update completely. This bit will be set when software write CNRn/CMRn/PPR or switch PWMn operation mode (PCR[3, 11, 19, 27]) and will be cleared by hardware automatically when PWM update these value completely.

PWM PDMA control Register (CAPPDMACTL)

Register	Offset	R/W	Description	Reset Value
CAPPDMACTL	PWMA_BA+0xC0	R/W	PWM Group A PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
				CAP3RFORDER	**CAP3PDMAMOD**		**CAP3PDMAEN**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
				CAP2RFORDER	**CAP2PDMAMOD**		**CAP2PDMAEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
				CAP1RFORDER	**CAP1PDMAMOD**		**CAP1PDMAEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				CAP0RFORDER	**CAP0PDMAMOD**		**CAP0PDMAEN**

Bits	Description	
[27]	**CAP3RFORDER**	**Capture Channel 3 Rising/Falling Order** Set this bit to determine whether the CRLR3 or CFLR3 is the first captured data transferred to memory through PDMA when CAP3PDMAMOD =11. 0 = CFLR3 is the first captured data to memory. 1 = CRLR3 is the first captured data to memory.
[26:25]	**CAP3PDMAMOD**	**Select CRLR3 or CFLR3 to Do PDMA Transfer** 00 = Reserved. 01 = CRLR3. 10 = CFLR3. 11 = Both CRLR3 and CFLR3.
[24]	**CAP3PDMAEN**	**Channel 3 PDMA Enable Bit** 0 = Channel 3 PDMA function Disabled. 1 = Channel 3 PDMA function Enabled for the channel 3 captured data and transfer to memory.
[19]	**CAP2RFORDER**	**Capture Channel 2 Rising/Falling Order** Set this bit to determine whether the CRLR2 or CFLR2 is the first captured data transferred to memory through PDMA when CAP2PDMAMOD =11. 0 = CFLR2 is the first captured data to memory. 1 = CRLR2 is the first captured data to memory.
[18:17]	**CAP2PDMAMOD**	**Select CRLR2 or CFLR2 to Do PDMA Transfer** 00 = Reserved. 01 = CRLR2. 10 = CFLR2. 11 = Both CRLR2 and CFLR2.
[16]	**CAP2PDMAEN**	**Channel 2 PDMA Enable Bit** 0 = Channel 2 PDMA function Disabled. 1 = Channel 2 PDMA function Enabled for the channel 2 captured data and transfer to memory.

[11]	**CAP1RFORDER**	**Capture Channel 1 Rising/Falling Order** Set this bit to determine whether the CRLR1 or CFLR1 is the first captured data transferred to memory through PDMA when CAP1PDMAMOD = 11. 0 = CFLR1 is the first captured data to memory. 1 = CRLR1 is the first captured data to memory.
[10:9]	**CAP1PDMAMOD**	**Select CRLR1 or CFLR1 to Transfer PDMA** 00 = Reserved. 01 = CRLR1. 10 = CFLR1. 11 = both CRLR1 and CFLR1.
[8]	**CAP1PDMAEN**	**Channel 1 PDMA Enable Bit** 0 = Channel 1 PDMA function Disabled. 1 = Channel 1 PDMA function Enabled for the channel 1 captured data and transfer to memory.
[3]	**CAP0RFORDER**	**Capture Channel 0 Rising/Falling Order** Set this bit to determine whether the CRLR0 or CFLR0 is the first captured data transferred to memory through PDMA when CAP0PDMAMOD =11. 0 = CFLR0 is the first captured data to memory. 1 = CRLR0 is the first captured data to memory.
[2:1]	**CAP0PDMAMOD**	**Select CRLR0 or CFLR0 to Transfer PDMA** 00 = Reserved. 01 = CRLR0. 10 = CFLR0. 11 = Both CRLR0 and CFLR0.
[0]	**CAP0PDMAEN**	**Channel 0 PDMA Enable Bit** 0 = Channel 0 PDMA function Disabled. 1 = Channel 0 PDMA function Enabled for the channel 0 captured data and transfer to memory.

PWM PDMA DATA Register 0-3 (CAP0-3PDMA)

Register	Offset	R/W	Description	Reset Value
CAP0PDMA	PWMA_BA+0xC4	R	PWM Group A Channel 0 PDMA DATA Register	0x0000_0000
CAP1PDMA	PWMA_BA+0xC8	R	PWM Group A Channel 1 PDMA DATA Register	0x0000_0000
CAP2PDMA	PWMA_BA+0xCC	R	PWM Group A Channel 2 PDMA DATA Register	0x0000_0000
CAP3PDMA	PWMA_BA+0xD0	R	PWM Group A Channel 3 PDMA 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**
CAPnRFPDMA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CAPnRFPDMA							

Bits	Description	
[15:0]	**CAPnRFPDMA**	**PDMA Data Register for PWM Channel N** It is the capturing value(CFLRn/CRLRn) for PWM channel n.

6.9 Watchdog Timer (WDT)

6.9.1 Overview

The purpose of Watchdog Timer (WDT) 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.

6.9.2 Features

- 18-bit free running up counter for WDT time-out interval
- Selectable time-out interval (2^4 ~ 2^{18}) and the time-out interval is 1.6 ms ~ 26.214 s if WDT_CLK = 10 kHz.
- System kept in reset state for a period of (1 / WDT_CLK) * 63
- Supports selectable WDT reset delay period, including 1026、130、18 or 3 WDT_CLK reset delay period
- Supports to force WDT enabled after chip powered on or reset by setting CWDTEN in Config0 register
- Supports WDT time-out wake-up function only if WDT clock source is selected as 10 kHz.

6.9.3 Block Diagram

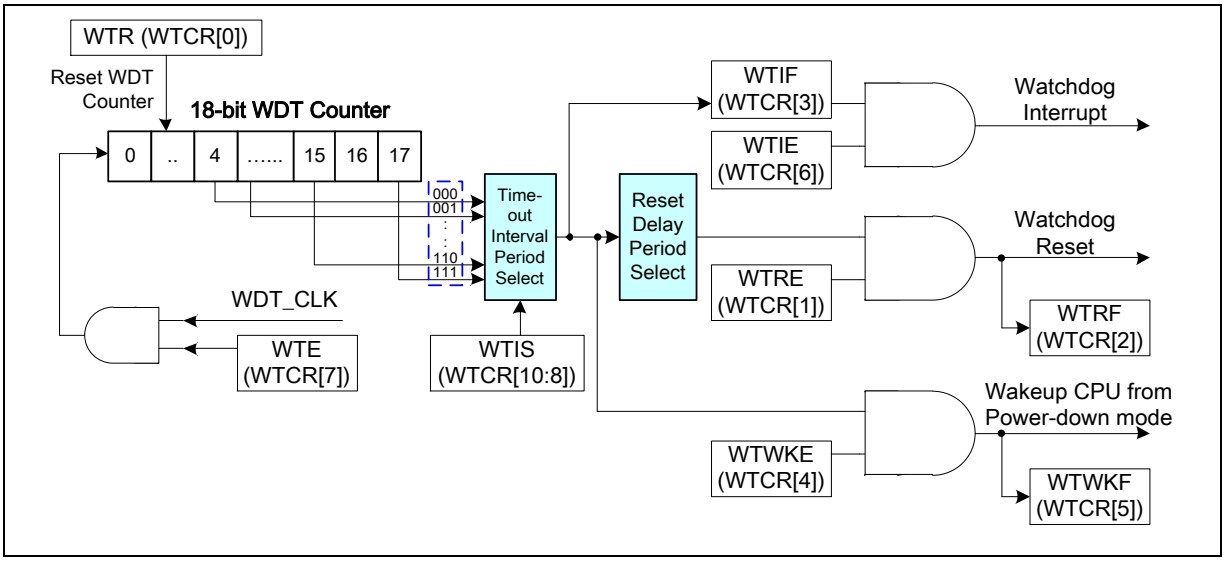


Figure 6-46 Watchdog Timer Block Diagram

Note1: WDT resets CPU and lasts 63 WDT_CLK.

Note2: Chip can be woken-up by WDT time-out interrupt signal generated only, if WDT clock source is selected to 10kHz oscillator.

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

6.9.4 Clock Control

The WDT clock control are shown as Figure 6-47.

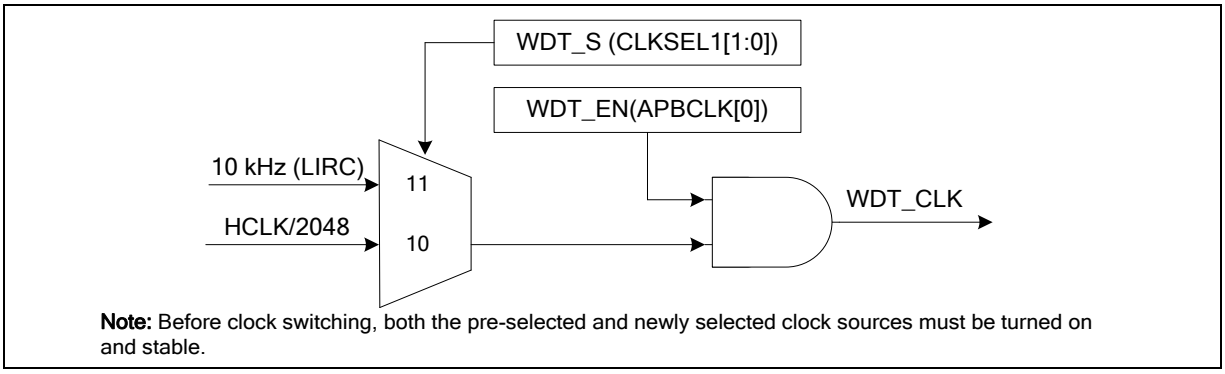


Figure 6-47 Watchdog Timer Clock Control

6.9.5 Basic Configuration

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

WDT controller also can be forced enabled and active in 10 kHz after chip powered on or reset while CWDTEN (Config0[31]) is configure to 0.

6.9.6 Functional Description

The WDT includes an 18-bit free running up counter with programmable time-out intervals. Table 6-17 shows the WDT time-out interval period selection and Figure 6-48 shows the WDT time-out interval and reset period timing.

6.9.6.1 *WDT Time-out Interrupt*

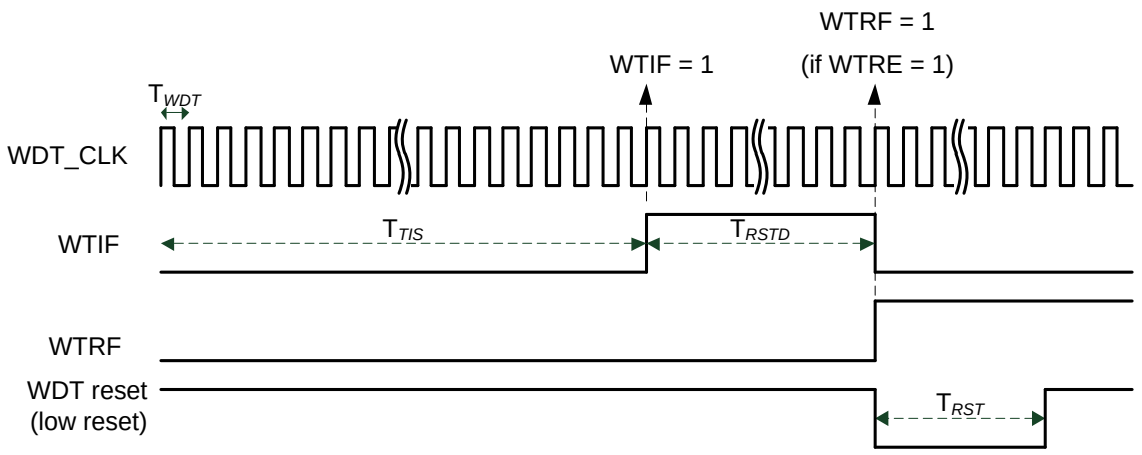
Setting WTE (WTCR[7]) to 1 will enable the WDT function and the WDT counter to start counting up. There are eight time-out interval period can be selected by setting WTIS (WTCR[10:8]). When the WDT up counter reaches the WTIS (WTCR[10:8]) settings, WDT time-out interrupt will occur then WDT time-out interrupt flag WTIF (WTCR[3]) will be set to 1 immediately.

6.9.6.2 *WDT Reset Delay Period and Reset System*

There is a specified T_{RSTD} reset delay period follows the WTIF (WTCR[3]) is setting to 1. User should set WTR (WTCR[0]) to reset the 18-bit WDT up counter value to avoid generate WDT time-out reset signal before the T_{RSTD} reset delay period expires. Moreover, user should set WTRDSEL (WTCRALT[1:0]) to select reset delay period to clear WDT counter. If the WDT up counter value has not been cleared after the specific T_{RSTD} delay period expires, the WDT control will set WTRF (WTCR[2]) to 1 if WTRE (WTCR[1]) bit is enabled, then chip enters to reset state immediately. Refer to Figure 6-48, T_{RST} reset period will keep last 63 WDT clocks then chip restart executing program from reset vector (0x0000_0000). The WTRF (WTCR[2]) will keep 1 after WDT time-out reset the chip, user can check WTRF (WTCR[2]) by software to recognize the system has been reset by WDT time-out reset 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 6-17 Watchdog Timer Time-out Interval Period Selection



- T_{WDT} : *Watchdog Clock Time Period*
- T_{TIS} : *Watchdog Time-out Interval Period ($(2^4 \sim 2^{18}) * T_{WDT}$)*
- T_{RSTD} : *Watchdog Reset Delay Period*
  *- Selectable 3/18/130/1026 * T_{WDT} delay period controlled by WTRDSEL (WTCRALT[1:0])*
- T_{RST} : *Watchdog Reset Period (63 * T_{WDT})*

Figure 6-48 Watchdog Timer Time-out Interval and Reset Period Timing

6.9.6.3 *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 (WTCR[4]) enabled. Notice that user should set OSC10K_EN (PWRCON[3]) before system entries Power-down mode because the system peripheral clock are disabled when system is Power-down mode. In the meanwhile, the WTWKF (WTCR[5]) will set to 1 automatically, user can check WTWKF (WTCR[5]) status by software to recognize the system has been waken-up by WDT time-out interrupt or not.

6.9.7 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	WDT Control Register	0x0000_0700
WTCRALT	WDT_BA+0x04	R/W	WDT Alternative Control Register	0x0000_0000

6.9.8 Register Description

WDT Control Register (WTCR)

Register	Offset	R/W	Description	Reset Value
WTCR	WDT_BA+0x00	R/W	WDT 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 (Write Protect)** 0 = ICE debug mode acknowledgement affects 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. **Note:** This bit is write protected. Refer to the REGWRPROT register.
[30:11]	**Reserved**	Reserved.
[10:8]	**WTIS**	**WDT Time-out Interval Selection (Write Protect)** These three bits select the time-out interval period for the WDT. 000 = 2^4 * WDT_CLK. 001 = 2^6 * WDT_CLK. 010 = 2^8 * WDT_CLK. 011 = 2^{10} * WDT_CLK. 100 = 2^{12} * WDT_CLK. 101 = 2^{14} * WDT_CLK. 110 = 2^{16} * WDT_CLK. 111 = 2^{18} * WDT_CLK. **Note:** This bit is write protected. Refer to the REGWRPROT register.
[7]	**WTE**	**WDT Enable Control (Write Protect)** 0 = WDT Disabled (This action will reset the internal up counter value). 1 = WDT Enabled. **Note1:** This bit is write protected. Refer to the REGWRPROT register. **Note2:** If CWDTEN (Config0[31]) bits is configure to 0, this bit is forced as 1 and user cannot change this bit to 0.
[6]	**WTIE**	**WDT 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. **Note:** This bit is write protected. Refer to the REGWRPROT register.
[5]	**WTWKF**	**WDT Time-out Wake-up Flag (Write Protect)** 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. **Note1:** This bit is write protected. Refer to the REGWRPROT register. **Note2:** This bit is cleared by writing 1 to it.
[4]	**WTWKE**	**WDT Time-out Wake-up Function Control (Write Protect)** If this bit is set to 1, while WDT time-out interrupt flag IF (WTCR[3]) is generated to 1 and interrupt enable bit WTIE (WTCR[6]) is enabled, the WDT time-out interrupt signal will generate a wake-up trigger event to chip. 0 = Wake-up trigger event Disabled if WDT time-out interrupt signal generated. 1 = Wake-up trigger event Enabled if WDT time-out interrupt signal generated. **Note1:** This bit is write protected. Refer to the REGWRPROT register. **Note2:** Chip can be woken-up by WDT time-out interrupt signal generated only if WDT clock source is selected to 10 kHz internal low speed RC oscillator (LIRC).
[3]	**WTIF**	**WDT 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**	**WDT 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**	**WDT 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:** This bit is write protected. Refer to the REGWRPROT register.
[0]	**WTR**	**Reset WDT Up Counter (Write Protect)** 0 = No effect. 1 = Reset the internal 18-bit WDT up counter value. **Note1:** This bit is write protected. Refer to the REGWRPROT register. **Note2:** This bit will be automatically cleared by hardware.

WDT Alternative Control Register (WTCRALT)

Register	Offset	R/W	Description	Reset Value
WTCRALT	WDT_BA+0x04	R/W	WDT 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**	**WDT Reset Delay Selection (Write Protect)** When WDT time-out happened, user has a time named WDT Reset Delay Period to clear WDT counter by setting WTR (WTCR[0]) to prevent WDT time-out reset happened. User can select a suitable setting of WTRDSEL for different WDT Reset Delay Period. 00 = WDT Reset Delay Period is 1026 * WDT_CLK. 01 = WDT Reset Delay Period is 130 * WDT_CLK. 10 = WDT Reset Delay Period is 18 * WDT_CLK. 11 = WDT Reset Delay Period is 3 * WDT_CLK. **Note1:** This bit is write protected. Refer to the REGWRPROT register. **Note2:** This register will be reset to 0 if WDT time-out reset happened.

6.10 Window Watchdog Timer (WWDT)

6.10.1 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. The 6-bit down counter value will stop to update when chip is in Idle or Power-down mode.

6.10.2 Features

- 6-bit down counter value WWDTCVAL (WWDTCVR[5:0]) and 6-bit compare value WINCMP (WWDTCR[21:16]) to make the WWDT time-out window period flexible
- Supports 4-bit value PERIODSEL (WWDTCR[11:8]) to programmable maximum 11-bit prescale counter period of WWDT counter

6.10.3 Block Diagram

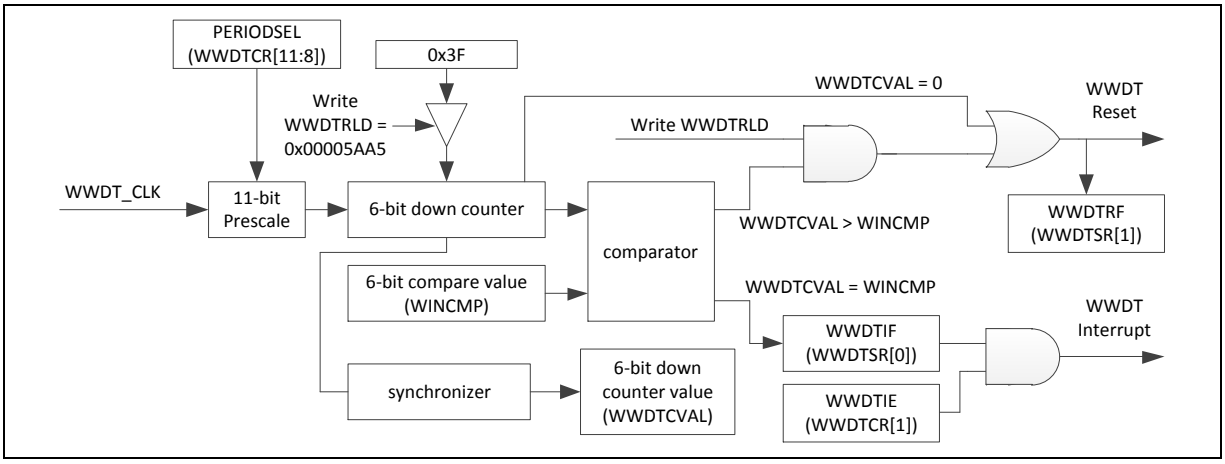


Figure 6-49 Window Watchdog Timer Block Diagram

6.10.4 Clock Diagram

The Window Watchdog Timer block diagram is shown in Figure 6-50.

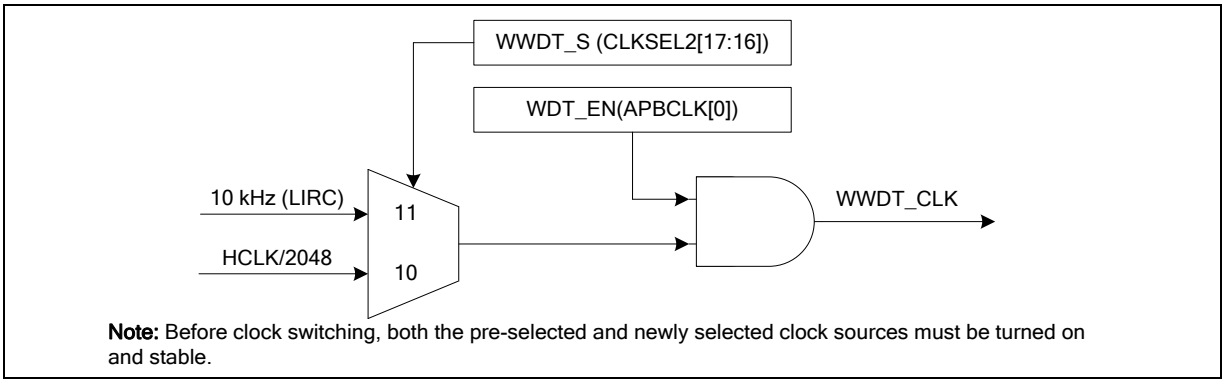


Figure 6-50 Window Watchdog Timer Clock Control

6.10.5 Basic Configuration

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

6.10.6 Functional Description

The WWDT includes a 6-bit down counter with programmable prescaler to define different time-out intervals. The clock source of 6-bit WWDT is based on system clock divide 2048 or internal 10 kHz oscillator with a programmable 11-bit prescaler. The programmable 11-bit prescaler is controlled by register PERIODSEL (WWDTCR[11:8]) and the correlate of PERIODSEL and prescaler value is listed in Table 6-18.

PERIODSEL	Prescaler Value	Time-Out Period	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 6-18 WWDT Prescaler Value Selection

6.10.6.1 *WWDT Counting*

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

6.10.6.2 *WWDT Compare Match Interrupt*

During down counting by the WWDT counter, the WWDTIF (WWDTSR[0]) is set to 1 while the WWDT counter value (WWDTCVAL) is equal to window compare value (WINCMP) and WWDTIF can be cleared by user; if WWDTIE (WWDTCR[1]) is also set to 1 by user, the WWDT compare match interrupt signal is generated also while WWDTIF is set to 1 by hardware.

6.10.6.3 *WWDT Reset System*

When WWDTIF (WWDTSR[0]) is generated, user must reload WWDT counter value to 0x3F by writing 0x00005AA5 to WWDTRLD register, and also to prevent WWDT counter value reached to 0 and generate WWDT reset system signal to info system reset. If current WWDTCVAL (WWDTCVR[5:0]) is larger than WINCMP (WWDTCR[21:16]) and user writes 0x00005AA5 to the WWDTRLD register, the WWDT reset system signal will be generated immediately to cause chip reset also.

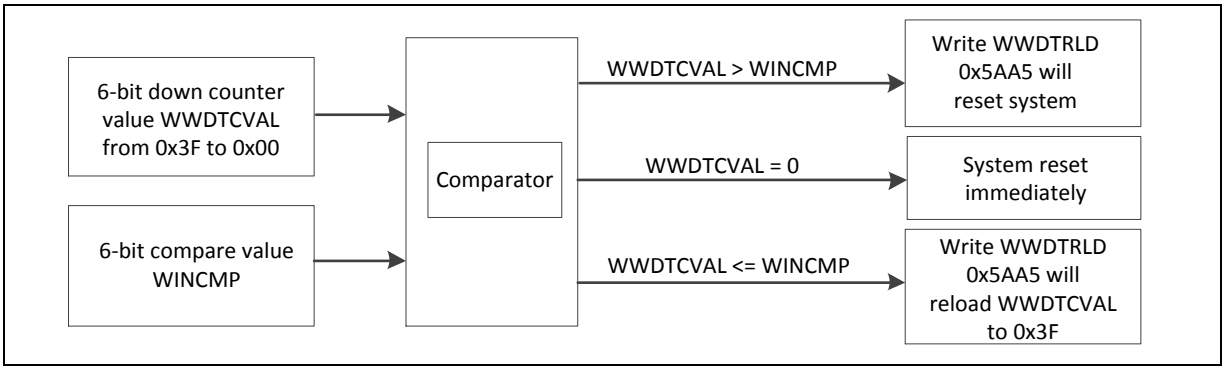


Figure 6-51 WWDT Reset and Reload Behavior

6.10.6.4 *WWDT Window Setting Limitation*

When user writes 0x00005AA5 to WWDTRLD register to reload WWDT counter value to 0x3F, it needs 3 WWDT clocks to sync the reload command to actually perform reload action. Notice that if user set PERIODSEL (WWDTCR[11:8]) to 0000, the counter prescale value should be as 1, and the WINCMP (WWDTCR[21:16]) must be larger than 2. Otherwise, writing WWDTRLD register to reload WWDT counter value to 0x3F is unavailable, WWDTIF (WWDTSR[0]) 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 6-19 WWDT Prescaler Value Selection

6.10.7 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	WWDT Timer Reload Counter Register	0x0000_0000
WWDTCR	WWDT_BA+0x04	R/W	WWDT Timer Control Register	0x003F_0800
WWDTSR	WWDT_BA+0x08	R/W	WWDT Timer Status Register	0x0000_0000
WWDTCVR	WWDT_BA+0x0C	R	WWDT Counter Value Register	0x0000_003F

6.10.8 Register Description

WWDT Reload Counter Register (WWDTRLD)

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

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

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

WWDT Control Register (WWDTCR)

Register	Offset	R/W	Description	Reset Value
WWDTCR	WWDT_BA+0x04	R/W	WWDT Control Register	0x003F_0800

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

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

Bits	Description	
[31]	**DBGACK_WWDT**	**ICE Debug Mode Acknowledge Disable Bit** 0 = ICE debug mode acknowledgement effects WWDT counting. WWDT down counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. WWDT down counter will keep going no matter CPU is held by ICE or not.
[30:22]	**Reserved**	Reserved.
[21:16]	**WINCMP**	**WWDT Window Compare Register** Set this register to adjust the valid reload window. **Note:** Software can only write WWDTRLD when WWDT counter value between 0 and WINCMP. If software writes WWDTRLD when WWDT counter value is larger than WWCMP, WWDT will generate RESET signal.
[15:12]	**Reserved**	Reserved.
[11:8]	**PERIODSEL**	**WWDT Pre-scale Period Selection** These three bits select the pre-scale for the WWDT counter period. 0000 = Pre-scale is 1; Max time-out period is 1 * 64 * WWDT_CLK. 0001 = Pre-scale is 2; Max time-out period is 2 * 64 * WWDT_CLK. 0010 = Pre-scale is 4; Max time-out period is 4 * 64 * WWDT_CLK. 0011 = Pre-scale is 8; Max time-out period is 8 * 64 * WWDT_CLK. 0100 = Pre-scale is 16; Max time-out period is 16 * 64 * WWDT_CLK. 0101 = Pre-scale is 32; Max time-out period is 32 * 64 * WWDT_CLK. 0110 = Pre-scale is 64; Max time-out period is 64 * 64 * WWDT_CLK. 0111 = Pre-scale is 128; Max time-out period is 128 * 64 * WWDT_CLK. 1000 = Pre-scale is 192; Max time-out period is 192 * 64 * WWDT_CLK. 1001 = Pre-scale is 256; Max time-out period is 256 * 64 * WWDT_CLK. 1010 = Pre-scale is 384; Max time-out period is 384 * 64 * WWDT_CLK. 1011 = Pre-scale is 512; Max time-out period is 512 * 64 * WWDT_CLK. 1100 = Pre-scale is 768; Max time-out period is 768 * 64 * WWDT_CLK.

		1101 = Pre-scale is 1024; Max time-out period is 1024 * 64 * WWDT_CLK. 1110 = Pre-scale is 1536; Max time-out period is 1536 * 64 * WWDT_CLK. 1111 = Pre-scale is 2048; Max time-out period is 2048 * 64 * WWDT_CLK.
[7:2]	**Reserved**	Reserved.
[1]	**WWDTIE**	**WWDT Interrupt Enable Bit** Set this bit to enable the Watchdog timer interrupt function. 0 = Watchdog timer interrupt function Disabled. 1 = Watchdog timer interrupt function Enabled.
[0]	**WWDTEN**	**WWDT Enable Bit** Set this bit to enable the Window Watchdog timer. 0 = Window Watchdog timer function Disabled. 1 = Window Watchdog timer function Enabled.

WWDT Status Register (WWDTSR)

Register	Offset	R/W	Description	Reset Value
WWDTSR	WWDT_BA+0x08	R/W	WWDT Status Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WWDTRF**	**WWDT Reset Flag** 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 can be cleared by software writing '1'.
[0]	**WWDTIF**	**WWDT Compare Match Interrupt Flag** This bit indicates the interrupt flag status of WWDT while WWDT counter value matches WINCMP (WWDTCR[21:16]). 0 = No effect. 1 = WWDT counter value matches WINCMP. **Note:** This bit can be cleared by software writing '1'.

WWDT Counter Value Register (WWDTCVR)

Register	Offset	R/W	Description	Reset Value
WWDTCVR	WWDT_BA+0x0C	R	WWDT Value Register	0x0000_003F

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**WWDTCVAL**	**WWDT Counter Value (Read Only)** This register reflects the down counter value of window watchdog.

6.11 UART Interface Controller (UART)

6.11.1 Overview

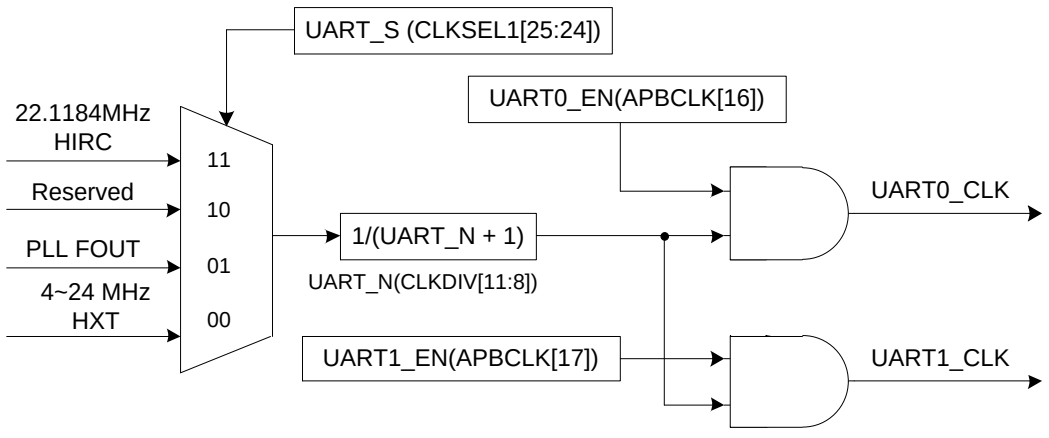
The NuMicro® NUC123 series provides two channels of Universal Asynchronous Receiver/ Transmitters (UART). UART Controller performs Normal Speed UART and supports flow control function. The UART Controller performs a serial-to-parallel conversion on data received from the peripheral and a parallel-to-serial conversion on data transmitted from the CPU. Each UART Controller channel supports six types of interrupts. The UART controller also supports IrDA SIR and RS-485.

6.11.2 Features

- Full duplex, asynchronous communications
- Separates receive / transmit 16/16 bytes entry FIFO for data payloads
- Supports hardware auto flow control/flow control
- Programmable receiver buffer trigger level
- Supports programmable baud-rate generator for each channel individually
- Supports nCTS wake-up function
- Supports 8-bit receiver buffer time-out detection function
- UART0/UART1 served by the DMA controller
- Programmable transmitting data delay time between the last stop and the next start bit by setting DLY (UA_TOR [15:8])
- 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 RS-485 function mode.
  - Supports RS-485 9-bit mode
  - Supports hardware or software direct enable to program nRTS pin to control RS-485 transmission direction

6.11.3 Block Diagram

The UART clock control and block diagram are shown in following.



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

Figure 6-52 UART Clock Control Diagram

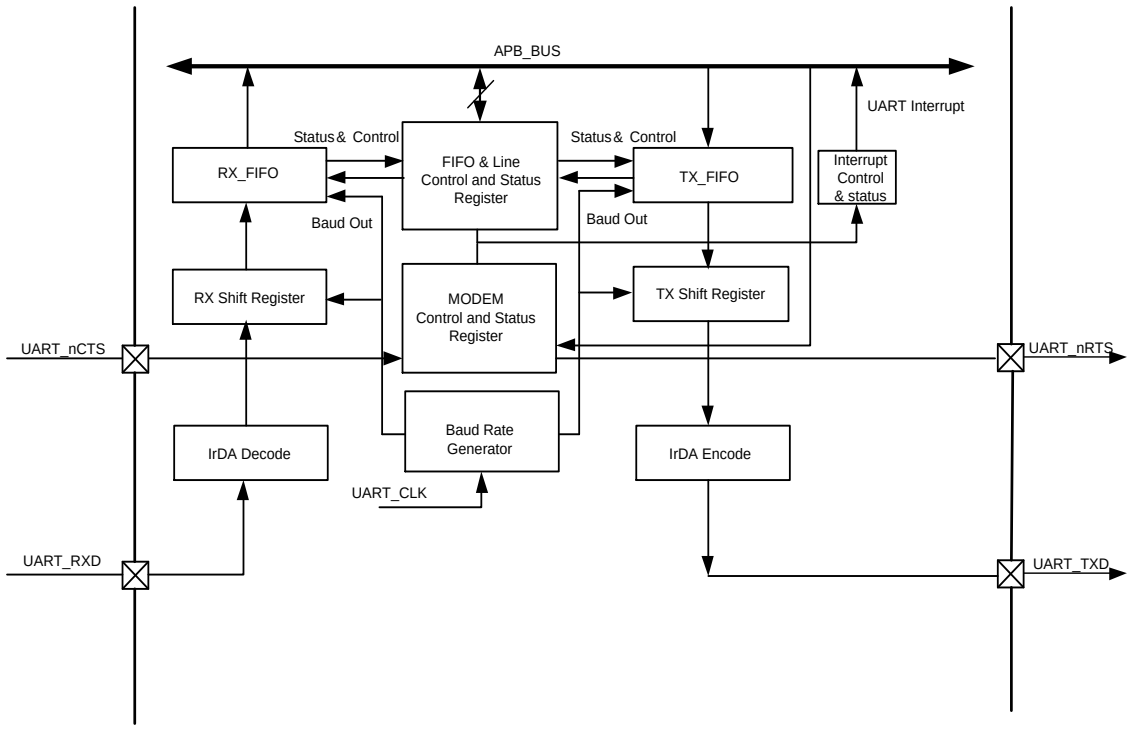


Figure 6-53 UART Block Diagram

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 BIF (UA_FSR[6]), FEF (UA_FSR[5]), PEF (UA_FSR[4]) per byte) to reduce the number of interrupts presented to the CPU.

TX shift Register

This block is responsible for shifting out the transmitting data serially.

RX shift Register

This block is responsible for shifting in the receiving data serially.

Modem Control and Status 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.

FIFO & Line 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.

Interrupt Control and Status Register

There are six 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) and Buffer error interrupt (BUF_ERR_INT).

6.11.4 Basic Configuration

The basic configurations of UART0 are as follows:

- UART0 pins are configured in ALT_MFP and GPB_MFP or GPC_MFP registers. **NUC123xxxAEx provides the alternative of configuring the** UART0 pins in GPB_MFPL or GPC_MFPL registers. (For NUC123xxxAEx, if GPB_MFPL or GPC_MFPL is used as pin multi-function setting, the ALT_MFP, GPB_MFP and GPC_MFP will become invalid).
- Enable UART0 peripheral clock in UART0_EN (APBCLK[16]).
- Reset UART0 controller in UART0_RST (IPRSTC2[16]).

The basic configurations of UART1 are as follows:

- UART1 pins are configured in ALT_MFP and GPB_MFP register. NUC123xxxAEx provides the alternative of configuring the UART1 pins in GPB_MFPL register.(For NUC123xxxAEx, if GPB_MFPL **is used** as pin multi-function setting, the ALT_MFP and GPB_MFP will become invalid).
- Enable UART1 peripheral clock in UART1_EN (APBCLK[17]).
- Reset UART1 controller in UART1_RST (IPRSTC2[17]).

6.11.5 Functional Description

The UART Controller supports three function modes including UART, IrDA and RS-485 mode. User can select a function by setting the UA_FUN_SEL register. The three function modes will be described in following section.

6.11.5.1 *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 Table 6-20 and Table 6-21 and Table 6-22 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. More detail register description is shown in UA_BAUD register. There are three setting mode. Mode 0 is set by UA_BAUD[29:28] with 00. Mode 1 is set by UA_BAUD[29:28] with 10. Mode 2 is set by UA_BAUD[29:28] with 11.

Mode	DIV_X_EN	DIV_X_ONE	Baud Rate Equation
Mode 0	0	0	UART_CLK / [16 * (BRD+2)]
Mode 1	1	0	UART_CLK / [(DIVIDER _X+1) * (BRD+2)], DIVIDER_X must >= 8
Mode 2 (NUC123xxxANx Only)	1	1	UART_CLK / (BRD+2) If UART_CLK <= HCLK, BRD must >= 9. If HCLK < UART_CLK =< 2*HCLK, BRD must >=15. If 2*HCLK < UART_CLK =< 3*HCLK, BRD must >=21. If UART_CLK > 3*HCLK, it is unsupported.
Mode 2 (NUC123xxxAEx Only)	1	1	UART_CLK / (BRD+2) If UART_CLK <= 3*HCLK, BRD must >= 9. If UART_CLK > 3*HCLK, BRD must >= 3*N – 1.

			N is the smallest integer larger than or equal to the ratio of UART_CLK /HCLK. For example, If 3*HCLK < UART_CLK =< 4*HCLK, BRD must >=11. If 4*HCLK < UART_CLK =< 5*HCLK, BRD must >=14.

Table 6-20 Baud Rate Equation Table

UART Peripheral Clock = 22.1184 MHz			
Baud Rate	**Mode 0**	**Mode 1**	**Mode 2**
921600	Not support	BRD=0, DIVIDER_X=11	BRD=22
460800	BRD=1	BRD=1, DIVIDER_X =15 BRD=2, DIVIDER_X =11	BRD=46
230400	BRD =4	BRD =4, DIVIDER_X =15 BRD =6, DIVIDER_X =11	BRD =94
115200	BRD =10	BRD =10, DIVIDER_X =15 BRD =14, DIVIDER_X =11	BRD =190
57600	BRD =22	BRD =22, DIVIDER_X =15 BRD =30, DIVIDER_X =11	BRD =382
38400	BRD =34	BRD =62, DIVIDER_X =8 BRD =46, DIVIDER_X =11 BRD =34, E DIVIDER_X =15	BRD =574
19200	BRD =70	BRD =126, DIVIDER_X =8 BRD =94, DIVIDER_X =11 BRD =70, DIVIDER_X =15	BRD =1150
9600	BRD =142	BRD =254, DIVIDER_X =8 BRD =190, DIVIDER_X =11 BRD =142, DIVIDER_X =15	BRD =2302
4800	BRD =286	BRD =510, DIVIDER_X =8 BRD =382, DIVIDER_X =11 BRD =286, DIVIDER_X =15	BRD =4606

Table 6-21 UART Controller Baud Rate Parameter Setting Example Table

UART Peripheral Clock = 22.1184 MHz			
Baud Rate	**UA_BAUD Value**		
	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 6-22 UART Controller Baud Rate Register Setting Example Table

6.11.5.2 *UART Controller Transmit Delay Time Value*

The UART Controller programs DLY (UA_TOR[15:8]) to control the transfer delay time between the last stop bit and next start bit in transmission. The unit is baud. The operation is shown in below.

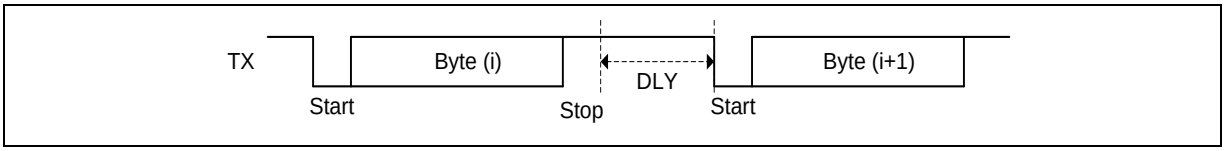


Figure 6-54 Transmit Delay Time Operation

6.11.5.3 *UART Controller FIFO Control and Status*

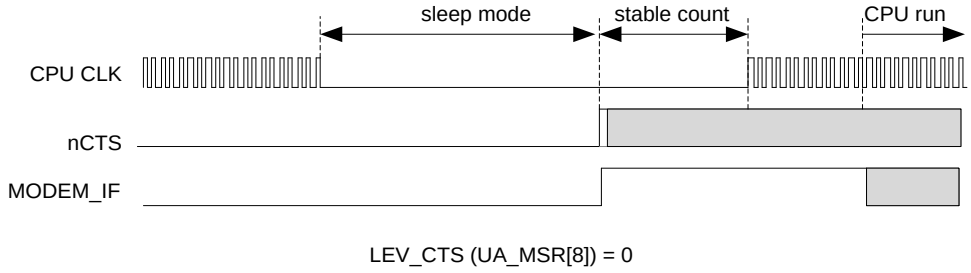
The UART Controller is built-in with a 16 bytes transmitter FIFO (TX_FIFO) and a 16 bytes receiver FIFO (RX_FIFO) that reduces the number of interrupts presented to the CPU. The CPU can read the status of the UART at any time during operation. The reported status information includes condition of the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

If there is any overrun event in transmitter or receiver FIFO, the buffer error flag BUF_ERR_IF (UA_ISR[5]) will be set automatically.

6.11.5.4 *UART Controller Wake-up Function*

The UART controller supports wake-up system function. The wake-up function includes nCTS. When the system is in Power-down, the UART can wake-up system by nCTS pin. The following diagram demonstrates the wake-up function.

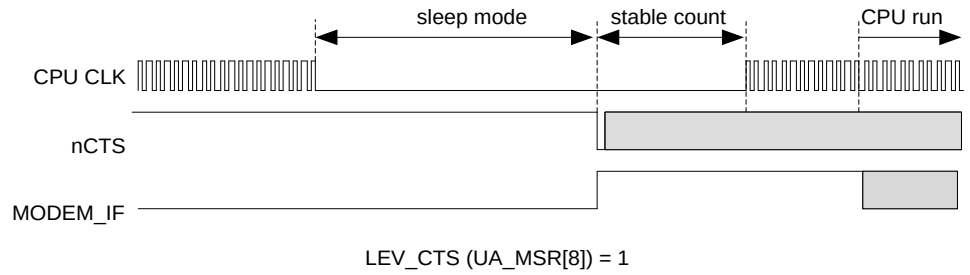
nCTS Wake-up Case 1 (nCTS transition from low to high)



Note: Stable count means HCLK source recovery stable count

Figure 6-55 UART nCTS Wake-up Case1

nCTS Wake-up Case 2 (nCTS transition from high to low)



Note: Stable count means HCLK source recovery stable count

Figure 6-56 UART nCTS Wake-up Case2

6.11.5.5 *UART Controller Interrupt and Status*

Each UART Controller supports six 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 error) (RLS_INT)
- MODEM status interrupt (MODEM_INT)
- Receiver buffer time-out interrupt (TOUT_INT)
- Buffer error interrupt (BUF_ERR_INT)

The Table 6-23 and Table 6-24 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.

UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator To Interrupt Controller	Interrupt Flag	Flag Cleared By
Buffer Error Interrupt	BUF_ERR_IEN	HW_BUF_ERR_INT	**HW_BUF_ERR_IF** = TX_OVER_IF	Write '1' to TX_OVER_IF
			HW_BUF_ERR_IF = RX_OVER_IF	Write '1' to RX_OVER_IF
Receiver Buffer Time-	RTO_IEN	HW_TOUT_INT	**HW_TOUT_IF**	Read UA_RBR

out Interrupt				
Modem Status Interrupt	MODEM_IEN	HW_MODEM_INT	**HW_MODEM_IF** = DCTSF	Write '1' to DCTSF
Receive Line Status Interrupt	RLS_IEN	HW_RLS_INT	**HW_RLS_IF** = BIF	Write '1' to BIF
			HW_RLS_IF =FEF	Write '1' to FEF
			HW_RLS_IF = PEF	Write '1' to PEF
			HW_RLS_IF = RS485_ADD_DETF	Write '1' to RS485_ADD_DETF
Transmit Holding Register Empty Interrupt	THRE_IEN	HW_THRE_INT	**HW_THRE_IF**	Write UA_THR
Receive Data Available Interrupt	RDA_IEN	HW_RDA_INT	**HW_RDA_IF**	Read UA_RBR

Table 6-23 UART Interrupt Sources and Flags Table in DMA Mode

UART Interrupt Source	Interrupt Enable Bit	Interrupt Indicator To Interrupt Controller	Interrupt Flag	Flag Cleared By
Buffer Error Interrupt	BUF_ERR_IEN	BUF_ERR_INT	**BUF_ERR_IF** = TX_OVER_IF	Write '1' to TX_OVER_IF
			BUF_ERR_IF = RX_OVER_IF	Write '1' to RX_OVER_IF
Receiver Buffer Time-out Interrupt	RTO_IEN	TOUT_INT	**TOUT_IF**	Read UA_RBR
Modem Status Interrupt	MODEM_IEN	MODEM_INT	**MODEM_IF** = DCTSF	Write '1' to DCTSF
Receive Line Status Interrupt	RLS_IEN	RLS_INT	**RLS_IF** = BIF	Write '1' to BIF
			RLS_IF = FEF	Write '1' to FEF
			RLS_IF = PEF	Write '1' to PEF
			RLS_IF = RS485_ADD_DETF	Write '1' to RS485_ADD_DETF
Transmit Holding Register Empty Interrupt	THRE_IEN	THRE_INT	**THRE_IF**	Write UA_THR
Receive Data Available Interrupt	RDA_IEN	RDA_INT	**RDA_IF**	Read UA_RBR

Table 6-24 UART Interrupt Sources and Flags Table in Software Mode

6.11.5.6 *UART Function Mode*

The UART Controller provides UART function (Setting FUN_SEL (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 UART supports hardware auto-flow control that provides programmable nRTS flow control trigger

level. When number of data bytes in RX FIFO is equal to or greater than RTS_TRI_LEV (UA_FCR[19:16]), the nRTS is de-asserted.

UART Line Control Function

The UART Controller supports fully programmable serial-interface characteristics by setting the UA_LCR register. User can program UA_LCR register for the word length, stop bit and parity bit setting. The Table 6-25 and Table 6-26 list the UART word, stop bit length and the 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 6-25 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 total count an even number.
Forced Mark 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 6-26 UART Line Control of Parity Bit Setting

UART Auto-Flow Control Function

The UART supports auto-flow control function that uses two signals, nCTS (clear-to-send) and nRTS (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 nRTS to external device. When the number of bytes stored in the RX FIFO equals the value of RTS_TRI_LEV (UA_FCR[19:16]), the nRTS is de-asserted. The UART sends data out when UART detects nCTS is asserted from external device. If the valid asserted nCTS is not detected, the UART

will not send data out.

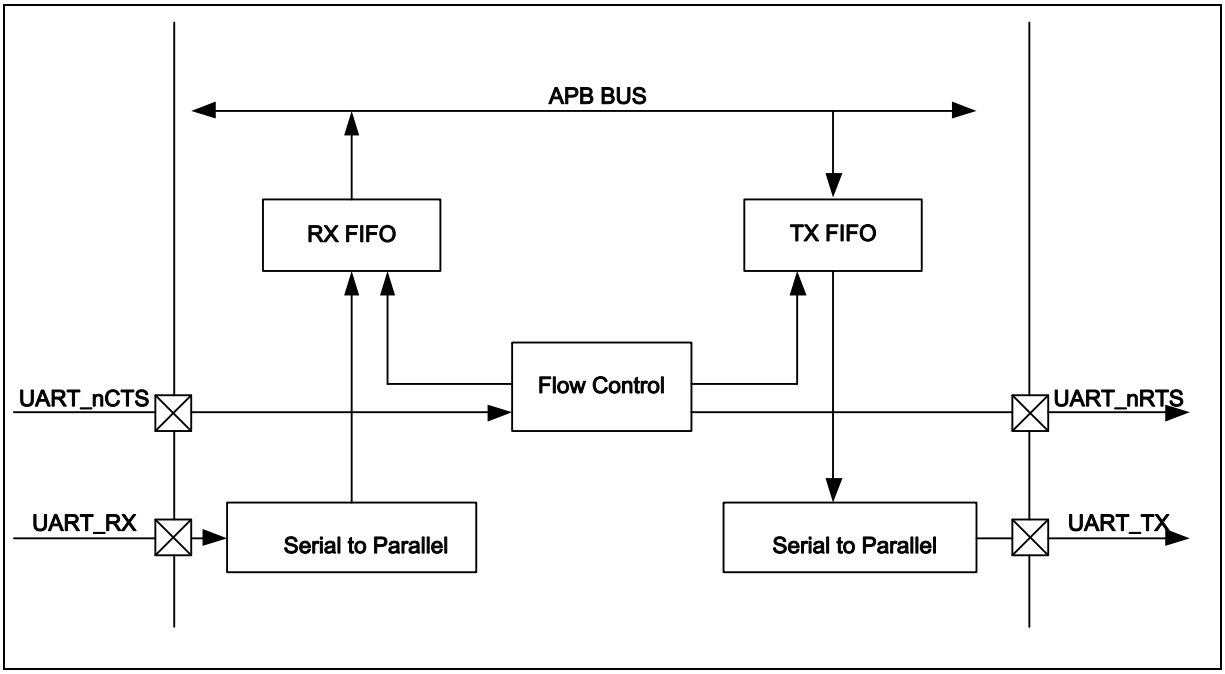


Figure 6-57 Auto-Flow Control Block Diagram

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

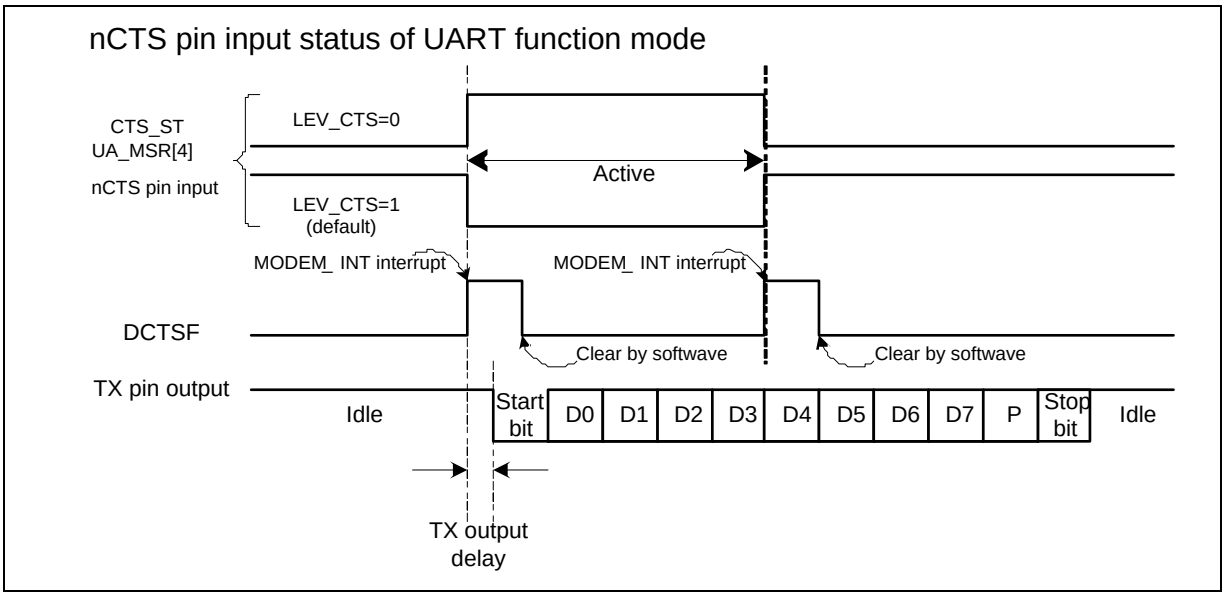


Figure 6-58 UART nCTS Auto-Flow Control Enabled

As shown in the Figure 6-59, in UART nRTS auto-flow control mode (AUTO_RTS_EN(UA_IER[12])=1), the nRTS internal signal is controlled by UA_FCR controller with RTS_TRI_LEV (UA_FCR[19:16]) trigger level.

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

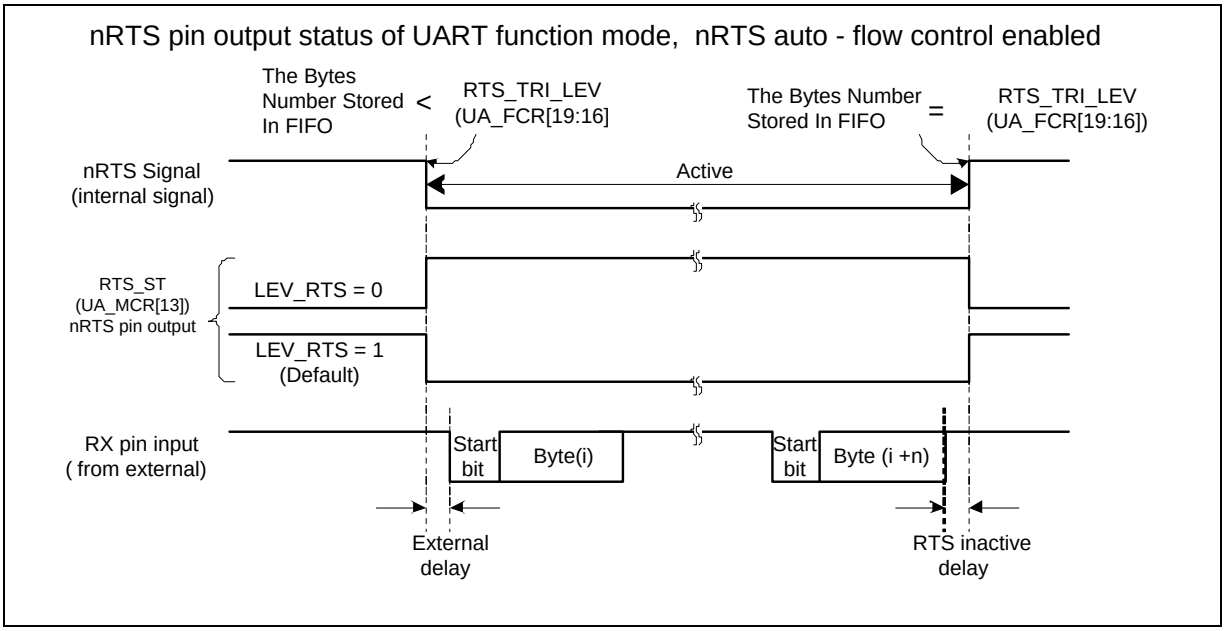


Figure 6-59 UART nRTS Auto-Flow Control Enabled

As shown in the Figure 6-60, in software mode (AUTO_RTS_EN (UA_IER[12])=0), the nRTS flow is directly controlled by software programming of RTS(UA_MCR[1]) control bit.

Setting LEV_RTS (UA_MCR[9]) can control the nRTS 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 nRTS pin output voltage logic status.

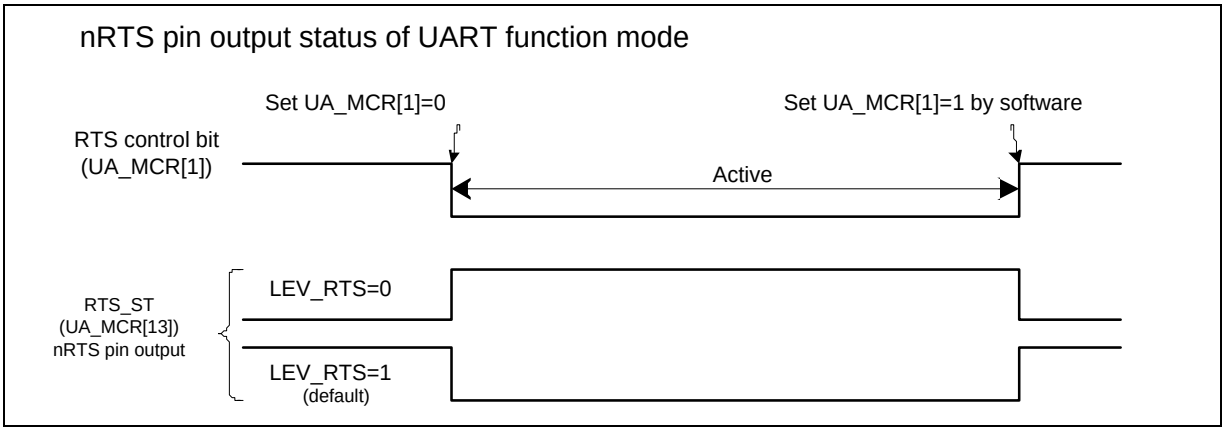


Figure 6-60 UART nRTS Auto-Flow with Software Control

6.11.5.7 *IrDA Function Mode*

The UART Controller also provides Serial IrDA (SIR, Serial Infrared) function (Setting FUN_SEL (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]) must be cleared.

Baud Rate = Clock / (16 * (BRD+2)), where BRD is Baud Rate Divider in UA_BAUD register.

The IrDA control block diagram is shown as follows.

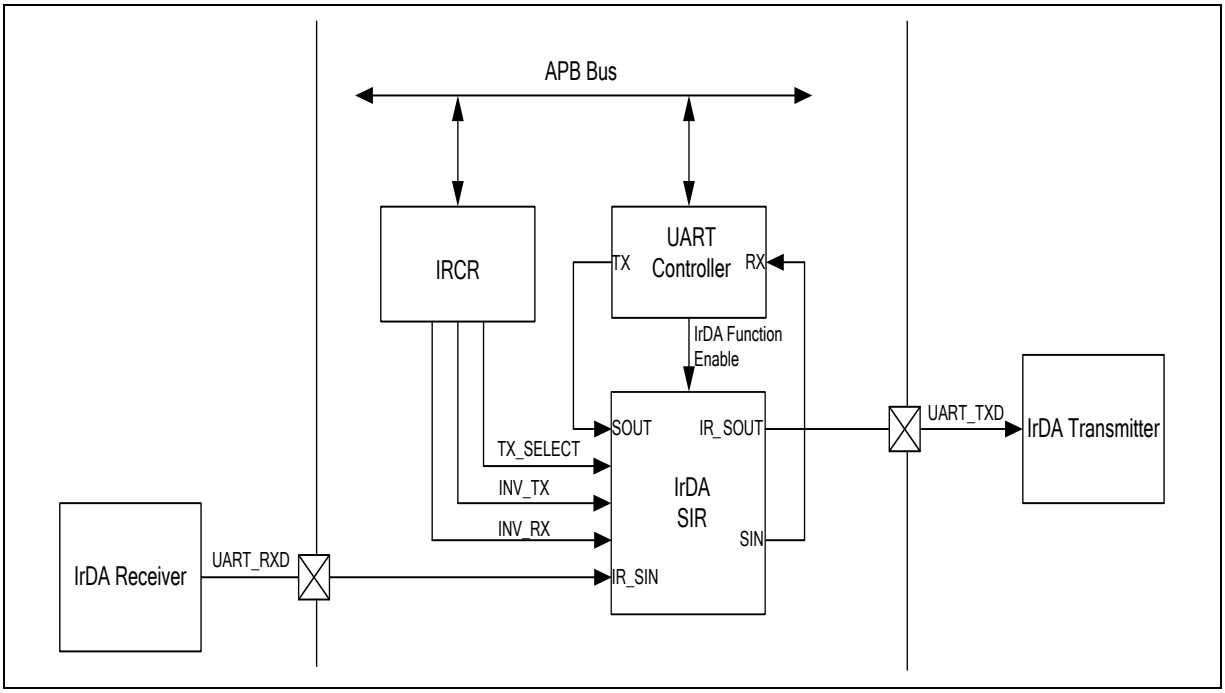


Figure 6-61 IrDA Block Diagram

6.11.5.7.1 IrDA SIR Transmit Encoder

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

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

6.11.5.7.2 IrDA SIR Receive Decoder

The IrDA SIR Receive Decoder demodulates the Return-to-Zero bit stream from the input detector and outputs the NRZ serial bits stream to the UART received data input. The decoder input is normally high in the idle state and the start bit is detected when the decoder input is LOW.

6.11.5.7.3 IrDA SIR Operation

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

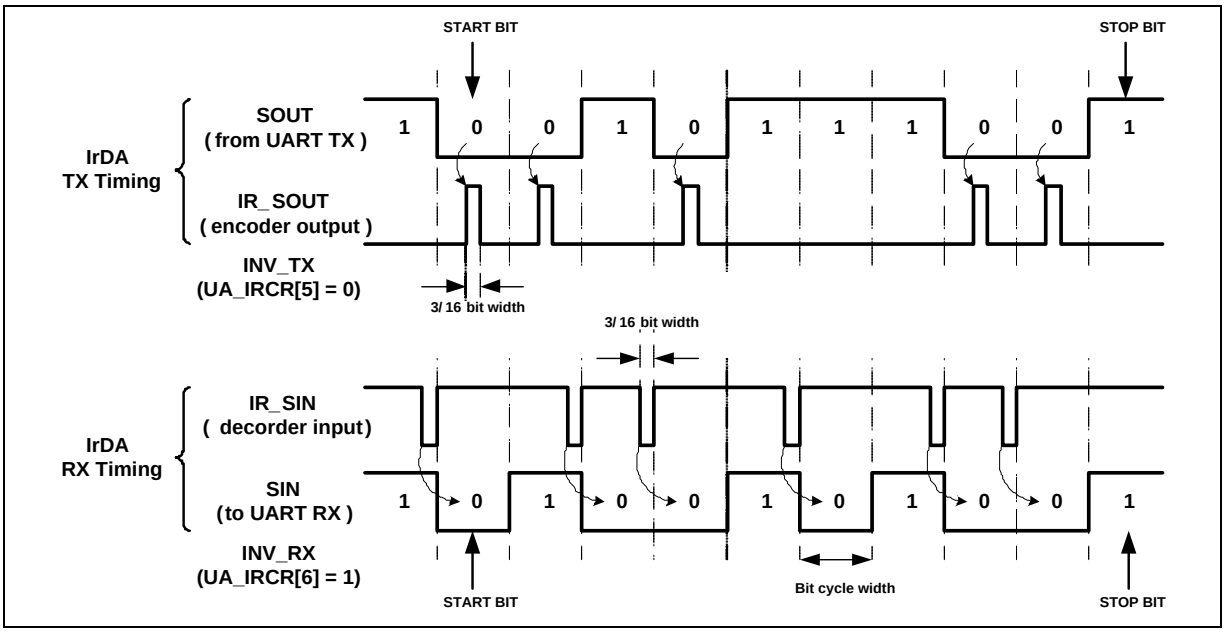


Figure 6-62 IrDA TX/RX Timing Diagram

6.11.5.8 *RS-485 Function Mode*

Another alternate function of UART Controller is RS-485 function (user must set FUN_SEL (UA_FUN_SEL[1:0]) to '11' to enable RS-485 function), and direction control provided by nRTS pin from an asynchronous serial port. The RS-485 transceiver control is implemented by using the nRTS control signal to enable the RS-485 driver. Many characteristics of the RX and TX are same as UART in RS-485 mode.

The UART controller can be configured as an RS-485 addressable slave and the RS-485 master transmitter will identify an address character by setting the parity (9-th bit) to 1. For data characters, the parity is set to 0. Software can use UA_LCR register to control the 9-th bit (When the PBE (UA_LCR[6]), EPE (UA_LCR[5]) and SPE (UA_LCR[4]) 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.

Note 1: When RS485 NMM or ADD mode is selected, the RS485 clock operating frequency should be less than or equal to half of PCLK clock operation frequency. Otherwise, RS485 cannot receive correct data. (NUC123xxxANx Only)

Note 2: RS485 ADD mode only support in Mode 2 (NUC123xxxANx Only).

The Controller supports three operation modes that is as following.

RS-485 Normal Multidrop Operation Mode (NMM)

In RS-485 Normal Multidrop Operation Mode (RS485_NMM (UA_ALT_CSR[8]) = 1), at 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 disable 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 the 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 (RS485_AAD (UA_ALT_CSR[9]) = 1), the receiver will ignore any data until an address byte is detected (bit9 =1) and the address byte data match the ADDR_MATCH (UA_ALT_CSR [31:24]) value. The address byte data will be stored in the RX-FIFO. The all received byte data will be accepted and stored in the RX -FIFO until an address byte data not match the ADDR_MATCH (UA_ALT_CSR[31:24]) value.

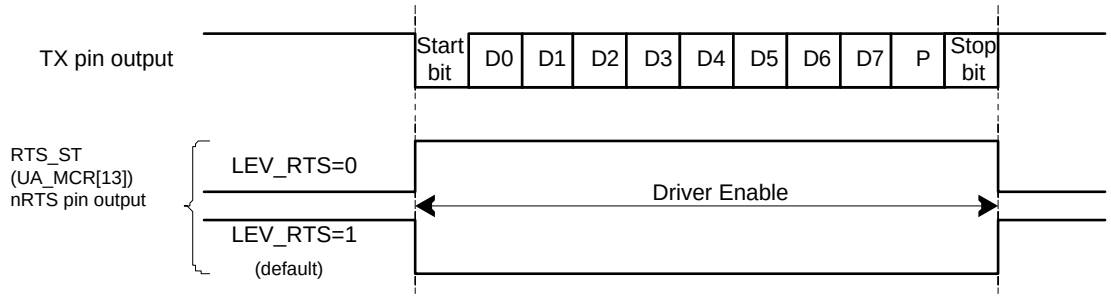
RS-485 Auto Direction Mode (AUD)

Another option function of RS-485 controllers is RS-485 auto direction control function (RS485_AUD (UA_ALT_CSR[10) = 1). The RS-485 transceiver control is implemented by using the nRTS control signal from an asynchronous serial port. The nRTS line is connected to the RS-485 transceiver enable pin such that setting the nRTS line to high (logic 1) enables the RS-485 transceiver. Setting the nRTS line to low (logic 0) puts the transceiver into the tri-state condition to disabled. User can setting LEV_RTS (UA_MCR[9]) to change the nRTS driving level.

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

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

nRTS pin output status of RS-485 function mode (RS-485 AUD mode enabled)



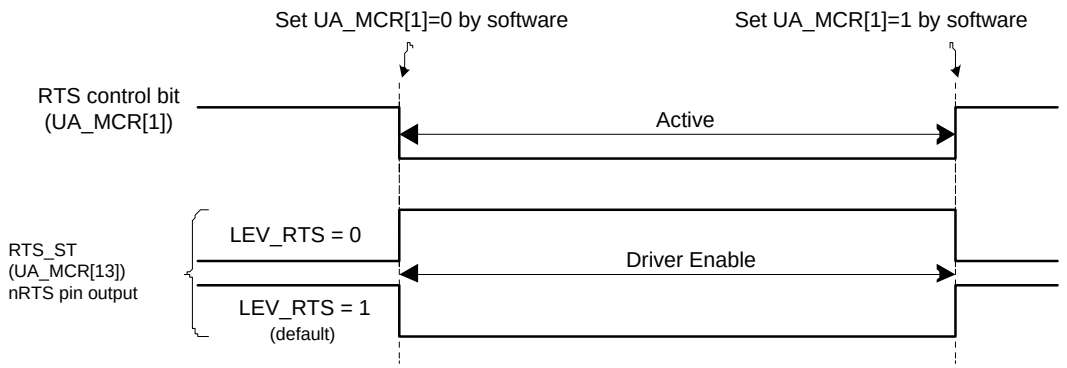
Note: RS485_AUD(UA_ALT_CSR[10]) = 1, the nRTS pin output by hardware control only.

Figure 6-63 RS-485 nRTS Driving Level in Auto Direction Mode

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

Setting LEV_RTS (UA_MCR[9]) can control the nRTS 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 nRTS pin output voltage logic status.

nRTS pin output status of RS-485 function mode

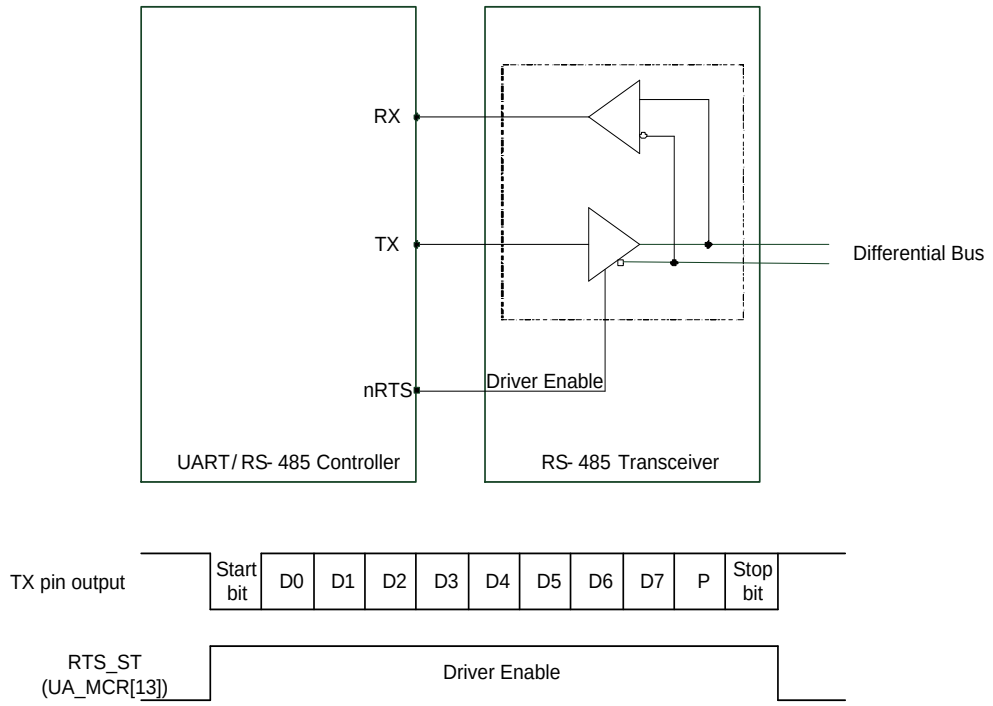


Note: RS485_AUD (UA_ALT_CSR[10]) = 0, support nRTS control bit by software control only.

Figure 6-64 RS-485 nRTS Driving Level with Software Control

Programming Sequence Example:

1. Program FUN_SEL in UA_FUN_SEL to select RS-485 function.
2. Program the RX_DIS (UA_FCR[8]) to determine enable or disable RS-485 receiver.
3. Program the RS485_NMM (UA_ALT_CSR[8]) or RS485_AAD (UA_ALT_CSR[9]) mode.
4. If the RS485_AAD (UA_ALT_CSR[9]) mode is selected, the ADDR_MATCH (UA_ALT_CSR[31:24]) is programmed for auto address match value.
5. Determine auto direction control by programming RS485_AUD (UA_ALT_CSR[10]).



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

Figure 6-65 Structure of RS-485 Frame

6.11.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UART Base Address : **UARTx_BA = 0x4005_0000 + (0x10_0000 * x)** **x= 0,1**				
UA_RBR	UARTx_BA+0x00	R	UART Receive Buffer Register	Undefined
UA_THR	UARTx_BA+0x00	W	UART Transmit Holding Register	Undefined
UA_IER	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000
UA_FCR	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101
UA_LCR	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000
UA_MCR	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200
UA_MSR	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110
UA_FSR	UARTx_BA+0x18	R/W	UART FIFO Status Register	0x1040_4000
UA_ISR	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0000_0002
UA_TOR	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000
UA_BAUD	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000
UA_IRCR	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040
UA_ALT_CSR	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_0000
UA_FUN_SEL	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000

6.11.7 Register Description

UART Receive Buffer Register (UA_RBR)

Register	Offset	R/W	Description	Reset Value
UA_RBR	UARTx_BA+0x00	R	UART Receive Buffer Register	Undefined

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

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

UART Transmit Holding Register (UA_THR)

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

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**THR**	**Transmit Holding Register** By writing one byte to this register, the data byte will be stored in transmitter FIFO. The UART Controller will send out the data stored in transmitter FIFO top location through the UART_TXD.

UART Interrupt Enable Register (UA_IER)

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

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DMA_RX_EN	DMA_TX_EN	AUTO_CTS_EN	AUTO_RTS_EN	TIME_OUT_EN	Reserved		
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	
[31:16]	**Reserved**	Reserved.
[15]	**DMA_RX_EN**	**RX DMA Enable Bit** This bit can enable or disable RX DMA service. 0 = RX DMA Disabled. 1 = RX DMA Enabled. **Note:** If RLS_IEN (UA_IER[2]) is enabled and HW_RLS_INT(UA_ISR[26]) is set to 1, the RLS (Receive Line Status) Interrupt is caused. If RLS interrupt is caused by Break Error Flag BIF(UA_FSR[6]), Frame Error Flag FEF(UA_FSR[5]) or Parity Error Flag PEF(UA_FSR[4]) , UART PDMA receive request operation is stop. Clear Break Error Flag BIF or Frame Error Flag FEF or Parity Error Flag PEF by writing "1" to corresponding BIF, FEF and PEF to make UART PDMA receive request operation continue.
[14]	**DMA_TX_EN**	**TX DMA Enable Bit** This bit can enable or disable TX DMA service. 0 = TX DMA Disabled. 1 = TX DMA Enabled. **Note:** If RLS_IEN (UA_IER[2]) is enabled and HW_RLS_INT(UA_ISR[26]) is set to 1, the RLS (Receive Line Status) Interrupt is caused. If RLS interrupt is caused by Break Error Flag BIF(UA_FSR[6]), Frame Error Flag FEF(UA_FSR[5]) or Parity Error Flag PEF(UA_FSR[4]) , UART PDMA transmit request operation is stop. Clear Break Error Flag BIF or Frame Error Flag FEF or Parity Error Flag PEF by writing "1" to corresponding BIF, FEF and PEF to make UART PDMA transmit request operation continue.
[13]	**AUTO_CTS_EN**	**nCTS Auto Flow Control Enable Bit** 0 = nCTS auto flow control Disabled. 1 = nCTS auto flow control Enabled. **Note:** When nCTS auto-flow is enabled, the UART will send data to external device when nCTS input assert (UART will not send data to device until nCTS is asserted).
[12]	**AUTO_RTS_EN**	**nRTS Auto Flow Control Enable Bit** 0 = nRTS auto flow control Disabled. 1 = nRTS auto flow control Enabled. **Note:** When nRTS auto-flow is enabled, if the number of bytes in the RX FIFO is equal to

		the RTS_TRI_LEV (UA_FCR [19:16]), the UART will de-assert nRTS signal.
[11]	**TIME_OUT_EN**	**Receive Buffer Time-out Counter Enable Bit** 0 = Receive Buffer Time-out counter Disabled. 1 = Receive Buffer Time-out counter Enabled.
[10:7]	**Reserved**	Reserved.
[6]	**WAKE_EN**	**UART Wake-up Function Enable Bit** 0 = UART wake-up function Disabled. 1 = UART wake-up function Enabled, when chip is in Power-down mode, an external nCTS change will wake-up chip from Power-down mode.
[5]	**BUF_ERR_IEN**	**Buffer Error Interrupt Enable Bit** 0 = Buffer error interrupt Disabled. 1 = Buffer error interrupt Enabled.
[4]	**RTO_IEN**	**RX Time-out Interrupt Enable Bit** 0 = RX time-out interrupt Disabled. 1 = RX time-out interrupt Enabled.
[3]	**MODEM_IEN**	**Modem Status Interrupt Enable Bit** 0 = Modem status interrupt Disabled. 1 = Modem status interrupt Enabled.
[2]	**RLS_IEN**	**Receive Line Status Interrupt Enable Bit** 0 = Receive Line Status interrupt Disabled. 1 = Receive Line Status interrupt Enabled.
[1]	**THRE_IEN**	**Transmit Holding Register Empty Interrupt Enable Bit** 0 = Transmit holding register empty interrupt Disabled. 1 = Transmit holding register empt interrupt Enabled.
[0]	**RDA_IEN**	**Receive Data Available Interrupt Enable Bit** 0 = Receive data available interrupt Disabled. 1 = Receive data available interrupt Enabled.

UART FIFO Control Register (UA_FCR)

Register	Offset	R/W	Description	Reset Value
UA_FCR	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**	**nRTS Trigger Level for Auto-flow Control Use** 0000 = nRTS Trigger Level is 1 byte. 0001 = nRTS Trigger Level is 4 bytes. 0010 = nRTS Trigger Level is 8 bytes. 0011 = nRTS Trigger Level is 14 bytes. Others = Reserved. **Note:** This field is used for auto nRTS flow control.
[15:9]	**Reserved**	Reserved.
[8]	**RX_DIS**	**Receiver Disable Bit** The receiver is enabled or disabled. 0 = Receiver Enabled. 1 = Receiver Disabled. **Note 1:** This field is used for RS-485 Normal Multi-drop mode. It should be programmed before RS485_NMM (UA_ALT_CSR [8]) is programmed. **Note 2:** Refer to **RS-485 Function Mode** section for detail information.
[7:4]	**RFITL**	**RX FIFO Interrupt Trigger Level** When the number of bytes in the received FIFO is equal to the RFITL, the RDA_IF (UA_ISR[0]) will be set (if RDA_IEN(UA_IER [0]) enabled, an interrupt will be generated). 0000 = RX FIFO Interrupt Trigger Level is 1 byte. 0001 = RX FIFO Interrupt Trigger Level is 4 bytes. 0010 = RX FIFO Interrupt Trigger Level is 8 bytes. 0011 = RX FIFO Interrupt Trigger Level is 14 bytes. Others = Reserved.
[3]	**Reserved**	Reserved.
[2]	**TFR**	**TX Field Software Reset** When TFR is set, all the byte in the transmit FIFO and TX internal state machine are cleared.

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

UART Line Control Register (UA_LCR)

Register	Offset	R/W	Description	Reset Value
UA_LCR	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** 0 = Break Control Disabled. 1 = Break Control Enabled. **Note:** When this bit is set to logic 1, the serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX and has no effect on the transmitter logic.
[5]	**SPE**	**Stick Parity Enable Bit** 0 = Stick parity Disabled. 1 = Stick parity Enabled. **Note:** If PBE (UA_LCR[3]) and EPE (UA_LCR[4]) are logic 1, the parity bit is transmitted and checked as logic 0. If PBE (UA_LCR[3]) is 1 and EPE (UA_LCR[4]) is 0 then the parity bit is transmitted and checked as 1.
[4]	**EPE**	**Even Parity Enable Bit** 0 = Odd number of logic 1's is transmitted and checked in each word. 1 = Even number of logic 1's is transmitted and checked in each word. **Note:** This bit has effect only when PBE (UA_LCR[3]) is set.
[3]	**PBE**	**Parity Bit Enable Bit** 0 = No parity bit. 1 = Parity bit generated Enabled. **Note:** 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-bit word length, 2 "STOP bit" is generated in the transmitted data.
[1:0]	**WLS**	**Word Length Selection** This field sets UART word length. 00 = 5 bits. 01 = 6 bits.

		10 = 7 bits. 11 = 8 bits.

UART MODEM Control Register (UA_MCR)

Register	Offset	R/W	Description	Reset Value
UA_MCR	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**	**nRTS Pin Status (Read Only)** This bit mirror from nRTS pin output of voltage logic status. 0 = nRTS pin output is low level voltage logic state. 1 = nRTS pin output is high level voltage logic state.
[12:10]	**Reserved**	Reserved.
[9]	**LEV_RTS**	**nRTS Pin Active Level** This bit defines the active level state of nRTS pin output. 0 = nRTS pin output is high level active. 1 = nRTS pin output is low level active. (Default) **Note1:** Refer to Figure 6-59 and Figure 6-60 for UART function mode. **Note2:** Refer to Figure 6-63 and Figure 6-64 for RS-485 function mode.
[8:2]	**Reserved**	Reserved.
[1]	**RTS**	**nRTS (Request-to-send) Signal Control** This bit is direct control internal nRTS signal active or not, and then drive the nRTS pin output with LEV_RTS bit configuration. 0 = nRTS signal is active. 1 = nRTS signal is inactive. **Note1:** This nRTS signal control bit is not effective when nRTS auto-flow control is enabled in UART function mode. **Note2:** This nRTS 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	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**	**nCTS Pin Active Level** This bit defines the active level state of nCTS pin input. 0 = nCTS pin input is high level active. 1 = nCTS pin input is low level active. (Default)
[7:5]	**Reserved**	Reserved.
[4]	**CTS_ST**	**nCTS Pin Status (Read Only)** This bit mirror from nCTS pin input of voltage logic status. 0 = nCTS pin input is low level voltage logic state. 1 = nCTS pin input is high level voltage logic state. **Note:** This bit echoes when UART Controller peripheral clock is enabled, and nCTS multi-function port is selected.
[3:1]	**Reserved**	Reserved.
[0]	**DCTSF**	**Detect nCTS Status Change Flag** This bit is set whenever nCTS input has change state, and it will generate Modem interrupt to CPU when MODEM_IEN (UA_IER[3]) is set to 1. 0 = nCTS input has not change state. 1 = nCTS input has change state. **Note:** This bit is can be cleared by writing “1” to it.

UART FIFO Status Register (UA_FSR)

Register	Offset	R/W	Description	Reset Value
UA_FSR	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 or the STOP bit of the last byte has not been transmitted. 1 = TX FIFO is empty and the STOP bit of the last byte has been transmitted. **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 can be cleared by software writing '1'.
[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 using level of TX FIFO Buffer equal to 16; otherwise, it is cleared by hardware.
[22]	**TX_EMPTY**	**Transmitter FIFO Empty (Read Only)** This bit indicates TX FIFO is 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 UA_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 indicates RX FIFO full or not. 0 = RX FIFO is not full. 1 = RX FIFO is full. **Note:** This bit is set when the using level of RX FIFO Buffer equal to 16; otherwise, it is cleared by hardware.
[14]	**RX_EMPTY**	**Receiver FIFO Empty (Read Only)** This bit indicates 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** 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. **Note:** This bit can be cleared by writing ‘1’ to it.
[5]	**FEF**	**Framing Error Flag** 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. **Note:** This bit is can be cleared by writing ‘1’ to it.
[4]	**PEF**	**Parity Error Flag** 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. **Note:** This bit can be cleared by writing ‘1’ to it.
[3]	**RS485_ADD_DETF**	**RS-485 Address Byte Detection Flag**

		0 = Receiver detects a data that is not an address bit (bit 9 ='0'). 1 = Receiver detects a data that is an address bit (bit 9 ='1'). **Note1:** This field is used for RS-485 function mode and RS485_ADD_EN (UA_ALT_CSR[15]) is set to 1 to enable Address detection mode . **Note 2:** This bit can be 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, this bit will be set. 0 = RX FIFO is not overflow. 1 = RX FIFO is overflow. **Note:** This bit can be cleared by writing '1' to it.

UART Interrupt Status Control Register (UA_ISR)

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

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

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**HW_BUF_ERR_INT**	**in DMA Mode, Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUF_ERR_IEN (UA_IER[5]) and HW_BUF_ERR_IF (UA_ISR[21]) are both set to 1. 0 = No buffer error interrupt is generated in DMA mode. 1 = Buffer error interrupt is generated in DMA mode.
[28]	**HW_TOUT_INT**	**in DMA Mode, Time-out Interrupt Indicator (Read Only)** This bit is set if RTO_IEN (UA_IER[4]) and HW_TOUT_IF (UA_ISR[20]) are both set to 1. 0 = No Tout interrupt is generated in DMA mode. 1 = Tout interrupt is generated in DMA mode.
[27]	**HW_MODEM_INT**	**in DMA Mode, MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEM_IEN (UA_IER[3]) and HW_MODEM_IF (UA_ISR[19]) are both set to 1. 0 = No Modem interrupt is generated in DMA mode. 1 = Modem interrupt is generated in DMA mode.
[26]	**HW_RLS_INT**	**in DMA Mode, Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLS_IEN (UA_IER[2]) and HW_RLS_IF (UA_ISR[18]) are both set to 1. 0 = No RLS interrupt is generated in DMA mode. 1 = RLS interrupt is generated in DMA mode.
[25:22]	**Reserved**	Reserved.
[21]	**HW_BUF_ERR_IF**	**in DMA Mode, Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX or RX FIFO overflows (TX_OVER_IF (UA_FSR[24]) or RX_OVER_IF (UA_FSR[0]) is set). When BUF_ERR_IF (UA_ISR[5]) is set, the transfer maybe is not correct. If BUF_ERR_IEN (UA_IER [5]) is enabled, the buffer error interrupt will be generated.

		0 = No buffer error interrupt flag is generated in DMA mode. 1 = Buffer error interrupt flag is generated in DMA mode. **Note:** This bit is cleared when both TX_OVER_IF (UA_FSR[24]) and RX_OVER_IF (UA_FSR[0]) are cleared.
[20]	**HW_TOUT_IF**	**in DMA Mode, Time-out Interrupt Flag (Read Only)** This bit is set when the RX FIFO is not empty and no activities occurred in the RX FIFO and the time-out counter equal to TOIC (UA_TOR[7:0]). If RTO_IEN (UA_IER [4]) is enabled, the Time-out interrupt will be generated. 0 = No Time-out interrupt flag is generated in DMA mode. 1 = Time-out interrupt flag is generated in DMA mode. **Note:** This bit is read only and user can read UA_RBR (RX is in active) to clear it.
[19]	**HW_MODEM_IF**	**in DMA Mode, MODEM Interrupt Flag (Read Only)** This bit is set when the nCTS pin has state change (DCTSF (UA_MSR[0])=1). If MODEM_IEN (UA_IER [3]) is enabled, the Modem interrupt will be generated. 0 = No Modem interrupt flag is generated in DMA mode. 1 = Modem interrupt flag is generated in DMA mode. **Note:** This bit is read only and reset to 0 when bit DCTSF (UA_MSR[0]) is cleared by a write 1 on DCTSF.
[18]	**HW_RLS_IF**	**in DMA Mode, Receive Line Status Flag (Read Only)** This bit is set when the RX receive data have parity error, framing error or break error (at least one of 3 bits, BIF (UA_FSR[6]), FEF (UA_FSR[5]) and PEF (UA_FSR[4]), is set). If RLS_IEN (UA_IER [2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated in DMA mode. 1 = RLS interrupt flag is generated in DMA mode. **Note 1:** In RS-485 function mode, this field includes “receiver detect any address byte received address byte character (bit9 = ‘1’) bit". **Note 2:** This bit is read only and reset to 0 when all bits of BIF (UA_FSR[6]), FEF (UA_FSR[5]) and PEF (UA_FSR[4]) are cleared. **Note3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF (UA_FSR[6]) , FEF(UA_FSR[5]) and PEF(UA_FSR[4]) and RS485_ADD_DETF (UA_FSR[3]) are cleared.
[17:14]	**Reserved**	Reserved.
[13]	**BUF_ERR_INT**	**Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUF_ERR_IEN (UA_IER[5] and BUF_ERR_IF (UA_ISR[5]) are both set to 1. 0 = No buffer error interrupt is generated. 1 = Buffer error interrupt is generated.
[12]	**TOUT_INT**	**Time-out Interrupt Indicator (Read Only)** This bit is set if RTO_IEN (UA_IER[4]) and TOUT_IF (UA_ISR[4]) 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 (UA_IER[3] and MODEM_IF (UA_ISR[3]) are both set to 1. 0 = No Modem interrupt is generated. 1 = Modem interrupt is generated.
[10]	**RLS_INT**	**Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLS_IEN (UA_IER[2] and RLS_IF (UA_ISR[2]) are both set to 1. 0 = No RLS interrupt is generated. 1 = RLS interrupt is generated.

[9]	**THRE_INT**	**Transmit Holding Register Empty Interrupt Indicator (Read Only)** This bit is set if THRE_IEN (UA_IER[1] and THRE_IF (UA_ISR[1]) are both set to 1. 0 = No THRE interrupt is generated. 1 = THRE interrupt is generated.
[8]	**RDA_INT**	**Receive Data Available Interrupt Indicator (Read Only)** This bit is set if RDA_IEN (UA_IER[0] and RDA_IF (UA_ISR[0]) are both set to 1. 0 = No RDA interrupt is generated. 1 = RDA interrupt is generated.
[7:5]	**Reserved**	Reserved.
[5]	**BUF_ERR_IF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX or RX FIFO overflows (TX_OVER_IF (UA_FSR[24]) or RX_OVER_IF (UA_FSR[0]) is set). When BUF_ERR_IF (UA_ISR[5]) is set, the transfer maybe is not correct. If BUF_ERR_IEN (UA_IER [5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated. 1 = Buffer error interrupt flag is generated. **Note:** This bit is cleared if both of RX_OVER_IF (UA_FSR[0]) and TX_OVER_IF (UA_FSR[24]) are cleared to 0 by writing 1 to RX_OVER_IF (UA_FSR[0]) and TX_OVER_IF (UA_FSR[24]).
[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 (UA_TOR[7:0]). If RTO_IEN (UA_IER [4]) is enabled, the Time-out 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 nCTS pin has state change (DCTSF (UA_MSR[0]) =1). If MODEM_IEN (UA_IER [3]) is enabled, the Modem interrupt will be generated. 0 = No Modem interrupt flag is generated. 1 = Modem interrupt flag is generated. **Note:** This bit is read only and reset to 0 when bit DCTSF is cleared by a write 1 on DCTSF.
[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 (UA_FSR[6]), FEF (UA_FSR[5]) and PEF (UA_FSR[4]), is set). If RLS_IEN (UA_IER [2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated. 1 = RLS interrupt flag is generated. **Note1:** In RS-485 function mode, this field is set include “receiver detect and received address byte character (bit9 = ‘1’) bit". At the same time, the bit of RS485_ADD_DETF (UA_FSR[3]) is also set. **Note2:** This bit is read only and reset to 0 when all bits of BIF (UA_FSR[6]), FEF(UA_FSR[5]) and PEF(UA_FSR[4]) are cleared. **Note3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF (UA_FSR[6]) , FEF(UA_FSR[5]) and PEF(UA_FSR[4]) and RS485_ADD_DET (UA_FSR[3]) are cleared.
[1]	**THRE_IF**	**Transmit Holding Register Empty Interrupt Flag (Read Only)** This bit is set when the last data of TX FIFO is transferred to Transmitter Shift Register. If THRE_IEN (UA_IER [1]) is enabled, the THRE interrupt will be generated. 0 = No THRE interrupt flag is generated.

		1 = THRE interrupt flag is generated. **Note:** This bit is read only and it will be cleared when writing data into UA_THR (TX FIFO not empty).
[0]	**RDA_IF**	**Receive Data Available Interrupt Flag (Read Only)** When the number of bytes in the RX FIFO is equal to the RFITL (UA_FCR[7:4]), the RDA_IF (UA_ISR[0]) will be set. If RDA_IEN (UA_IER [0]) is enabled, the RDA interrupt will be generated. 0 = No RDA interrupt flag is generated. 1 = RDA interrupt flag is generated. **Note:** This bit is read only and it will be cleared when the number of unread bytes of RX FIFO drops below the threshold level (RFITL (UA_FCR[7:4])).

UART_Time-out Register (UA_TOR)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**DLY**	**TX Delay Time Value** This field is used to programming the transfer delay time between the last stop bit and next start bit. The unit is bit time.
[7:0]	**TOIC**	**Time-out Interrupt Comparator** The time-out counter resets and starts counting (counting clock = baud rate) whenever the RX FIFO receives a new data word if time out counter is enabled by setting TIME_OUT_EN(UA_IER[11]). Once the content of time-out counter is equal to that of time-out interrupt comparator (TOIC (UA_TOR[7:0])), a receiver time-out interrupt (TOUT_INT (UA_ISR[12])) is generated if RTO_IEN (UA_IER [4]). A new incoming data word or RX FIFO empty clears TOUT_IF (UA_ISR[4]). 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. Thus, for example, if TOIC is set as 40, the time-out interrupt is generated after four characters are not received when 1 stop bit and no parity check is set for UART transfer.

UART Baud Rate Divider Register (UA_BAUD)

Register	Offset	R/W	Description	Reset Value
UA_BAUD	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000

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

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**DIV_X_EN**	**Divider X Enable Bit** The BRD = Baud Rate Divider, and the baud rate equation is Baud Rate = Clock / [M * (BRD + 2)]; The default value of M is 16. 0 = Divider X Disabled (the equation of M = 16). 1 = Divider X Enabled (the equation of M = X+1, but DIVIDER_X [27:24] must >= 8). **Note 1:** The detail description is shown in **UART Controller Baud Rate Generator** section. **Note 2:** In IrDA mode must be operated in mode 0.
[28]	**DIV_X_ONE**	**Divider X Equal to 1** 0 = Divider M is X +1 (the equation of M = X+1, but DIVIDER_X[27:24] must >= 8). 1 = Divider M is 1. **Note:** The detail description is shown in **UART Controller Baud Rate Generator** section.
[27:24]	**DIVIDER_X**	**Divider X** The baud rate divider M is X+1. **Note 1:** This field is used for baud rate calculation in mode 1 and has no effect for baud rate calculation in mode 0 and mode 2. **Note 2:** The detail description is shown in **UART Controller Baud Rate Generator** section.
[23:16]	**Reserved**	Reserved.
[15:0]	**BRD**	**Baud Rate Divider** The field indicates the baud rate divider. This filed is used in baud rate calculation. **Note:** The detail description is shown in **UART Controller Baud Rate Generator** section.

UART IrDA Control Register (UA_IRCR)

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

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**INV_RX**	**IrDA Inverse Receive Input Signal** 0 = None inverse receiving input signal. 1 = Inverse receiving input signal. (Default)
[5]	**INV_TX**	**IrDA Inverse Transmitting Output Signal** 0 = None inverse transmitting signal. (Default) 1 = Inverse transmitting output signal.
[4:2]	**Reserved**	Reserved.
[1]	**TX_SELECT**	**IrDA Receiver/Transmitter Selection Enable Bit** 0 = IrDA Transmitter Disabled and Receiver Enabled. (Default) 1 = IrDA Transmitter Enabled and Receiver Disabled.
[0]	**Reserved**	Reserved.

UART Alternate Control/Status Register (UA_ALT_CSR)

Register	Offset	R/W	Description	Reset Value
UA_ALT_CSR	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**
Reserved							

Bits	Description	
[31:24]	**ADDR_MATCH**	**Address Match Value** This field contains the RS-485 address match values. **Note:** This field is used for RS-485 Auto Address Detection mode.
[23:16]	**Reserved**	Reserved.
[15]	**RS485_ADD_EN**	**RS-485 Address Detection Enable Bit** This bit is used to enable RS-485 Address Detection mode. 0 = Address detection mode Disabled. 1 = Address detection mode Enabled. **Note:** This field is used for RS-485 any operation mode.
[14:11]	**Reserved**	Reserved.
[10]	**RS485_AUD**	**RS-485 Auto Direction Mode (AUD)** 0 = RS-485 Auto Direction Operation mode (AUD) Disabled. 1 = RS-485 Auto Direction Operation mode (AUD) Enabled. **Note:** It can be active with RS-485_AAD or RS-485_NMM operation mode.
[9]	**RS485_AAD**	**RS-485 Auto Address Detection Operation Mode (AAD)** 0 = RS-485 Auto Address Detection Operation mode (AAD) Disabled. 1 = RS-485 Auto Address Detection Operation mode (AAD) Enabled. **Note:** It can't be active with RS-485_NMM operation mode.
[8]	**RS485_NMM**	**RS-485 Normal Multi-drop Operation Mode (NMM)** 0 = RS-485 Normal Multi-drop Operation mode (NMM) Disabled. 1 = RS-485 Normal Multi-drop Operation mode (NMM) Enabled. **Note:** It can't be active with RS-485_AAD operation mode.
[7:0]	**Reserved**	Reserved.

UART Function Select Register (UA_FUN_SEL)

Register	Offset	R/W	Description	Reset Value
UA_FUN_SEL	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 Selection Enable Bits** 00 = UART Function Enabled. 01 = Reserved. 10 = IrDA Function Enabled. 11 = RS-485 Function Enabled.

6.12 PS/2 Device Controller (PS2D)

6.12.1 Overview

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

6.12.2 Features

- Host communication inhibit and request to send detection
- Reception frame error detection
- Programmable 1 to 16 bytes transmit buffer to reduce CPU intervention
- Double buffer for data reception
- S/W override bus

6.12.3 Basic Configuration

The basic configurations of PS/2 are as follows:

- PS/2 pins are configured in the GPF_MFP and ALT_MFP1. NUC123xxxAEx provides the alternative of configuring the in the GPF_MFPL. (For NUC123xxxAEx, if GPF_MFPL is used as pin multi-function setting, the GPF_MFP and ALT_MFP1 will become invalid).
- Enable PS/2 peripheral clock in PS2_EN (APBCLK[31]).
- Reset PS/2 controller in PS2_RST (IPRST2[23]).

6.12.4 Block Diagram

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

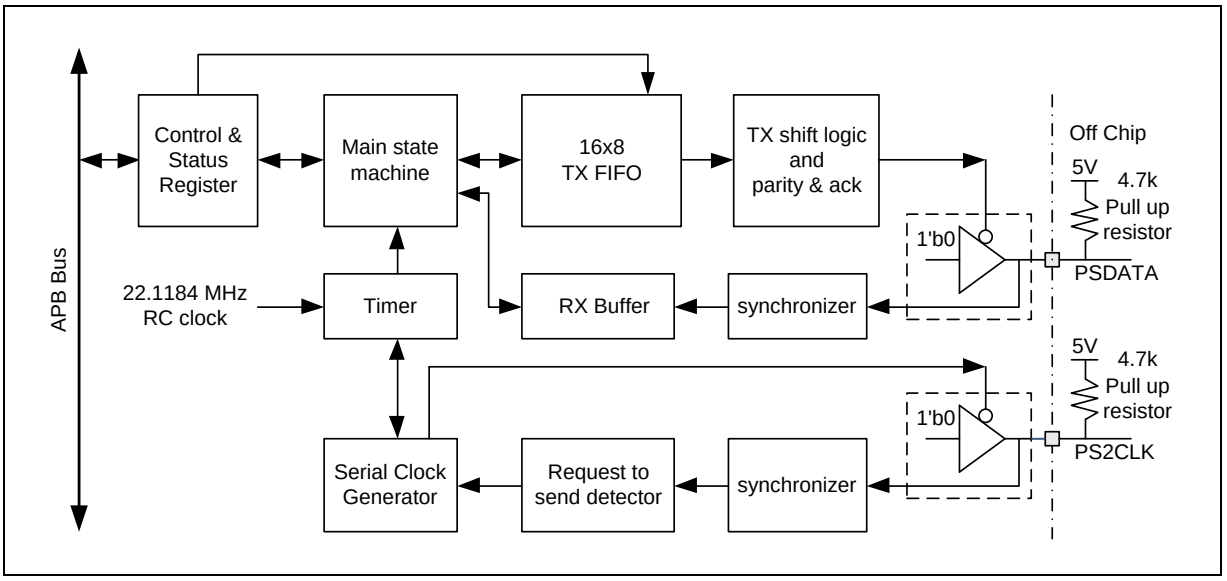


Figure 6-66 PS/2 Device Block Diagram

6.12.5 Functional Description

6.12.5.1 *Communication*

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

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

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

Table 6-27 PS/2 Bus State Define Table

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

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

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

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

Device-to-Host

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

- 1 start bit. This is always 0
- 8 DATA bits, least significant bit first
- 1 parity bit (odd parity)
- 1 stop bit. This is always 1

The device writes a bit on the DATA line when CLK is high, and it is read by the host when CLK is low, which is illustrated in Figure 6-67.

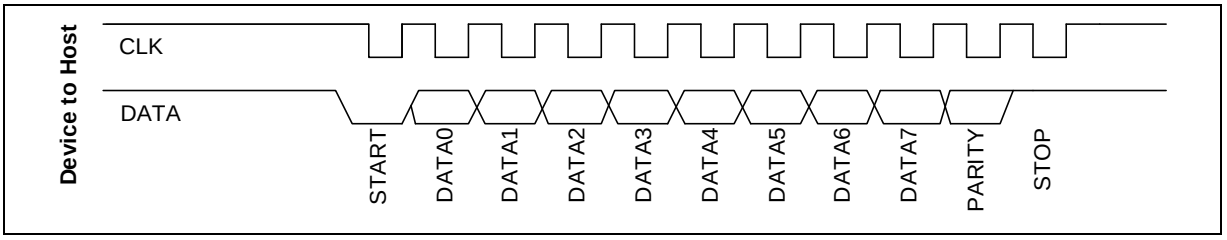


Figure 6-67 Data Format of Device-to-Host

Host-to-Device:

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

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

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

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

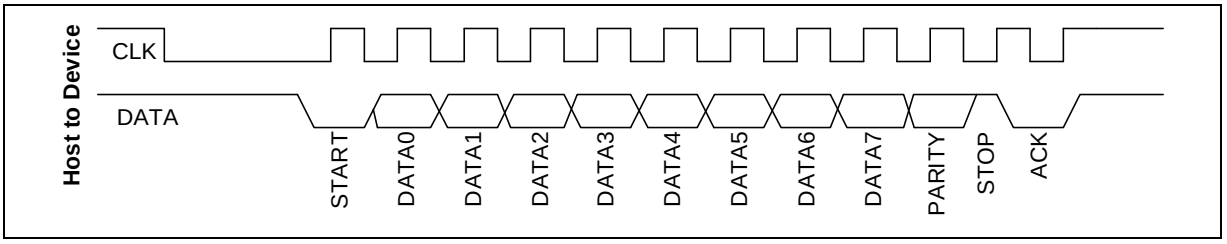


Figure 6-68 Data Format of Host-to-Device

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

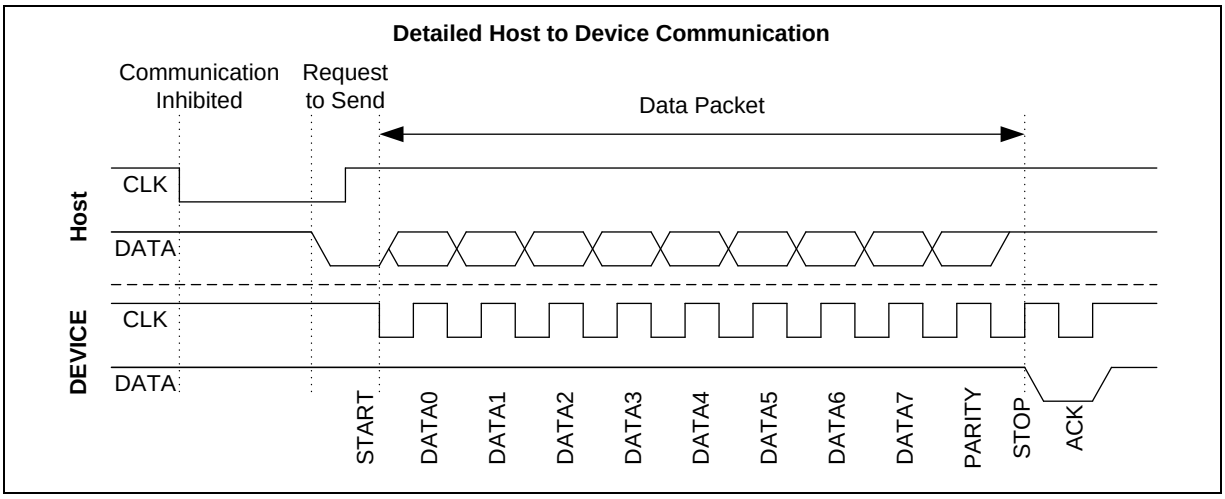


Figure 6-69 PS/2 Bit Data Format

6.12.5.2 *PS/2 Bus Timing Specification*

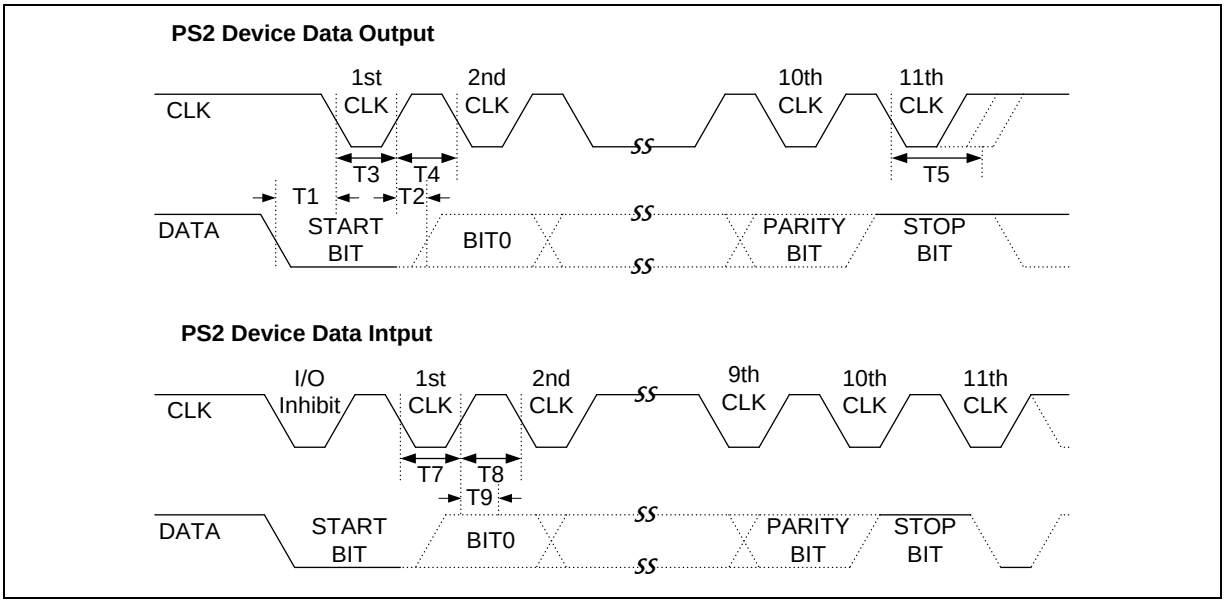


Figure 6-70 PS/2 Bus Timing

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

Table 6-28 PS/2 Bus Timing Parameter Define Table

6.12.5.3 *TX FIFO Operation*

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

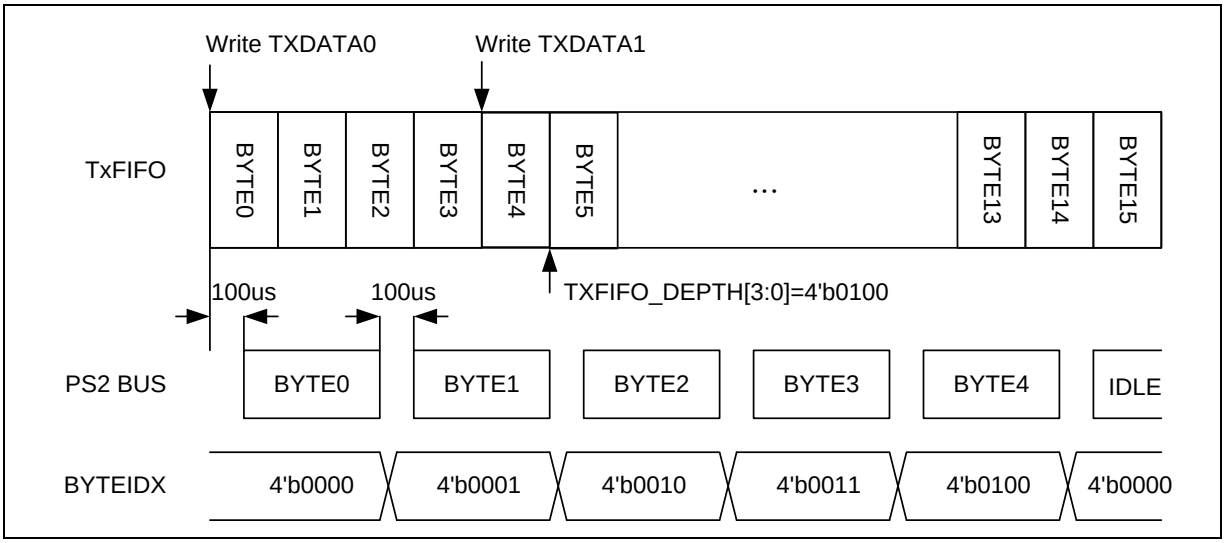


Figure 6-71 PS/2 Data Format

6.12.6 Register Map

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

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

6.12.7 Register Description

PS/2 Control Register (PS2CON)

Register	Offset	R/W	Description	Reset Value
PS2CON	PS2_BA+0x00	R/W	PS/2 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**FPS2DAT**	**FPS2CLK**	**OVERRIDE**	**CLRFIFO**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACK	**TXFIFO_DEPTH**				**RXINTEN**	**TXINTEN**	**PS2EN**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**FPS2DAT**	**Force PS2DATA Line** It forces PS2DATA high or low regardless of the internal state of the device controller if OVERRIDE is set to high. 0 = Force PS2DATA low. 1 = Force PS2DATA high.
[10]	**FPS2CLK**	**Force PS2CLK Line** It forces PS2CLK line high or low regardless of the internal state of the device controller if OVERRIDE is set to high. 0 = Force PS2CLK line low. 1 = Force PS2CLK line high.
[9]	**OVERRIDE**	**Software Override PS/2 CLK/DATA Pin State** 0 = PS2CLK and PS2DATA pins are controlled by internal state machine. 1 = PS2CLK and PS2DATA pins are controlled by software.
[8]	**CLRFIFO**	**Clear TX FIFO** Write 1 to this bit to terminate device to host transmission. The TXEMPTY bit in PS2STATUS bit will be set to 1 and pointer BYTEIDEX is reset to 0 regardless there is residue data in buffer or not. The buffer content is not been cleared. 0 = Not active. 1 = Clear FIFO.
[7]	**ACK**	**Acknowledge Enable Bit** 0 = Always sends acknowledge to host at 12th clock for host to device communication. 1 = If parity error or stop bit is not received correctly, acknowledge bit will not be sent to host at 12th clock.
[6:3]	**TXFIFO_DEPTH**	**Transmit Data FIFO Depth** There is 16-byte buffer for data transmit. Software can define the FIFO depth from 1 to 16 bytes depending on the application.

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

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

Register	Offset	R/W	Description	Reset Value
PS2TXDATA0	PS2_BA+0x04	R/W	PS/2 Transmit Data Register0	0x0000_0000
PS2TXDATA1	PS2_BA+0x08	R/W	PS/2 Transmit Data Register1	0x0000_0000
PS2TXDATA2	PS2_BA+0x0C	R/W	PS/2 Transmit Data Register2	0x0000_0000
PS2TXDATA3	PS2_BA+0x10	R/W	PS/2 Transmit Data Register3	0x0000_0000

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

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

PS/2 Receiver DATA Register (PS2RXDATA)

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PS2RXDATA							

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

PS/2 Status Register (PS2STATUS)

Register	Offset	R/W	Description	Reset Value
PS2STATUS	PS2_BA+0x18	R/W	PS/2 Status Register	0x0000_0000

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**BYTEIDX**	**Byte Index (Read Only)** It indicates which data byte in transmit data shift register. When all data in FIFO is transmitted and it will be cleared to 0.

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

[7]	**TXEMPTY**	**TX FIFO Empty (Read Only)** When software writes any data to PS2TXDATA0-3 the TXEMPTY bit is cleared to 0 immediately if PS2EN is enabled. When transmitted data byte number is equal to FIFODEPTH then TXEMPTY bit is set to 1. 0 = There is data to be transmitted. 1 = FIFO is empty.
[6]	**RXOVF**	**RX Buffer Overwrite** 0 = No overwrite. 1 = Data in PS2RXDATA register is overwritten by new received data. **Note:** This bit can be cleared by software writing '1'.
[5]	**TXBUSY**	**Transmit Busy (Read Only)** This bit indicates that the PS/2 device is currently sending data.

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

PS/2 Interrupt Identification Register (PS2INTID)

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[1]	**TXINT**	**Transmit Interrupt** This bit is set to 1 after STOP bit is transmitted. Interrupt occurs if TXINTEN bit is set to 1. 0 = No interrupt. 1 = Transmit interrupt occurs. **Note:** This bit can be cleared by software writing '1'.
[0]	**RXINT**	**Receive Interrupt** This bit is set to 1 when acknowledge bit is sent for Host to device communication. Interrupt occurs if RXINTEN bit is set to 1. 0 = No interrupt. 1 = Receive interrupt occurs. **Note:** This bit can be cleared by software writing '1'.

6.13 I^2C Serial Interface Controller (Master/Slave) (I^2C)

6.13.1 Overview

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

6.13.2 Features

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

6.13.3 Basic Configuration

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

- I2C0 pins are configured in GPF_MFP and ALT_MFP1. NUC123xxxAEx provides the alternative of configuring the I2C0 pins in GPF_MFPL. (For NUC123xxxAEx, if GPF_MFPL is used as pin multi-function setting, the GPF_MFP and ALT_MFP1 will become invalid).
- Enable I2C0 peripheral clock in I2C0_EN(APBCLK [8]).
- Reset I2C0 controller in I2C0_RST(IPRSTC2 [8]).

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

- I2C1 pins are configured in GPA_MFP and ALT_MFP. NUC123xxxAEx provides the alternative of configuring the I2C1 pins in GPA_MFPH. (For NUC123xxxAEx, if GPA_MFPH is used as pin multi-function setting, the GPA_MFP and ALT_MFP will become invalid).
- Enable I2C1 peripheral clock in I2C1_EN(APBCLK [9]).
- Reset I2C1 controller in I2C1_RST(IPRSTC2 [9]).

6.13.4 Block Diagram

The block diagram of I^2C controller is shown below.

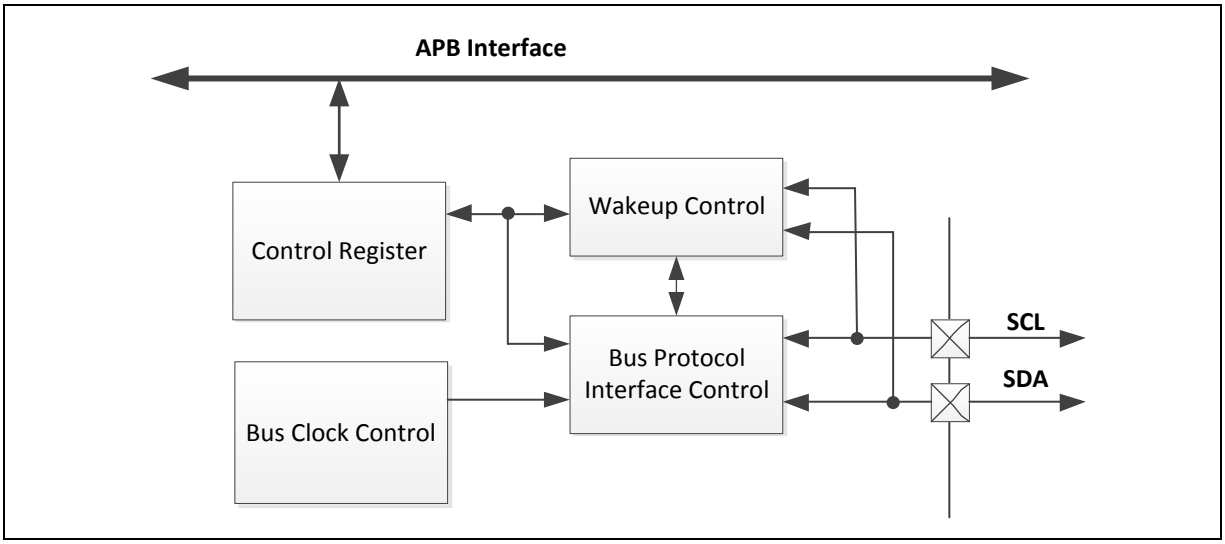


Figure 6-72 I^2C Controller Block Diagram

6.13.5 Functional Description

On I^2C 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 Figure 6-73 for more detailed I^2C BUS Timing.

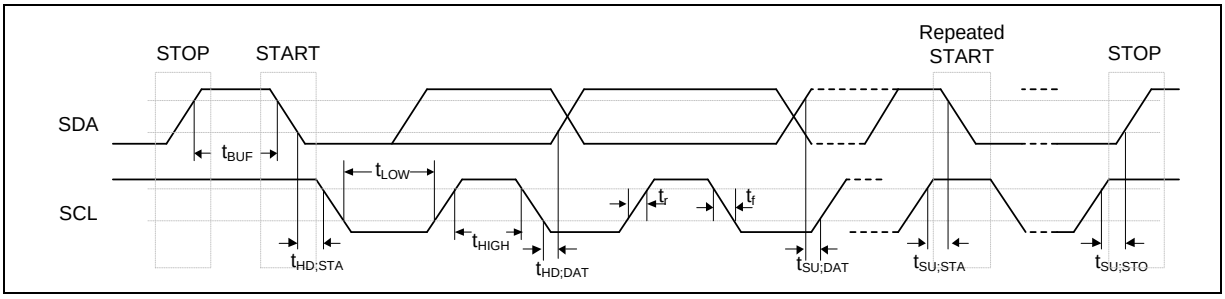


Figure 6-73 I²C Bus Timing

The device's on-chip I²C provides the serial interface that meets the I²C bus standard mode specification. The I²C port handles byte transfers autonomously. To enable this port, the bit IPEN (I2CON[0]) should be set to '1'. The I²C hardware interfaces to the I²C bus via two pins: SDA and SCL. When I/O pins are used as I²C ports, user must set the pins function to I²C in advance.

Note: Pull-up resistor is needed for I²C operation as the SDA and SCL are open-drain pins

6.13.5.1 *I²C Protocol*

The Figure 6-74 shows the typical I²C protocol. Normally, a standard communication consists of four parts :

1. START or Repeated START signal generation
2. Slave address transfer
3. Data transfer
4. STOP signal generation

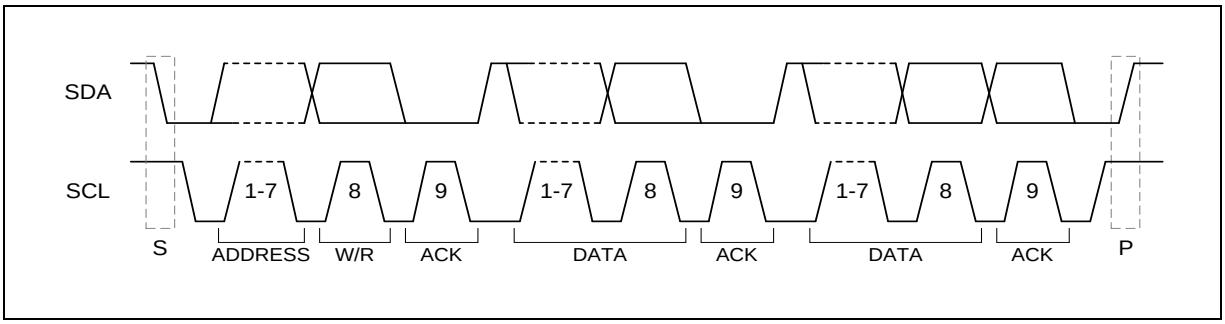


Figure 6-74 I²C Protocol

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 as the S-bit, is defined as a HIGH to LOW transition on the SDA line while SCL is HIGH. The START signal denotes the beginning of a new data transfer.

6.13.5.1.1 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 transfer.

A Repeated START (Sr) is a START signal without first generating a STOP signal. 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.

6.13.5.1.2 STOP signal

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

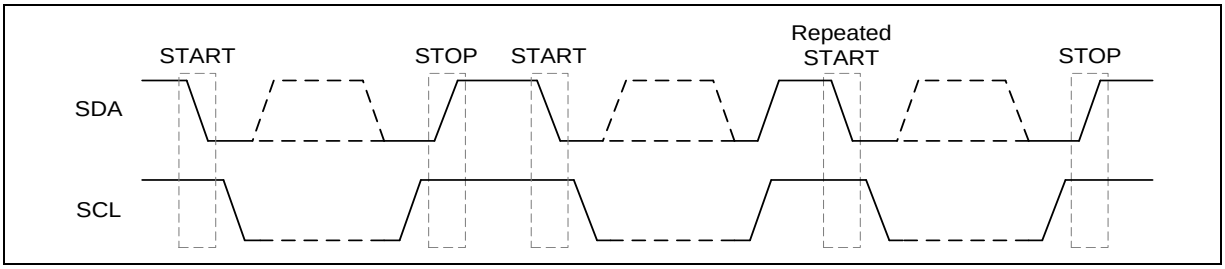


Figure 6-75 I^{2}C START and STOP Conditions

6.13.5.1.3 Slave Address Transfer

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

6.13.5.1.4 Data Transfer

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

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

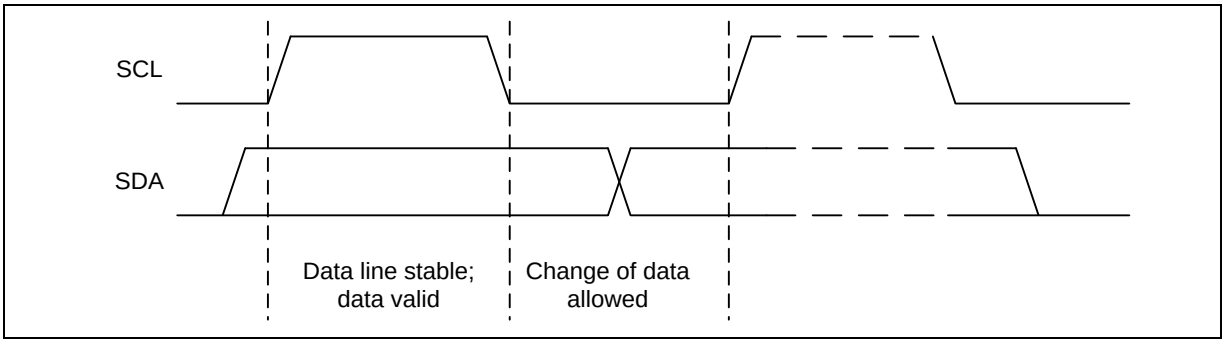


Figure 6-76 Bit Transfer on the I^{2}C Bus

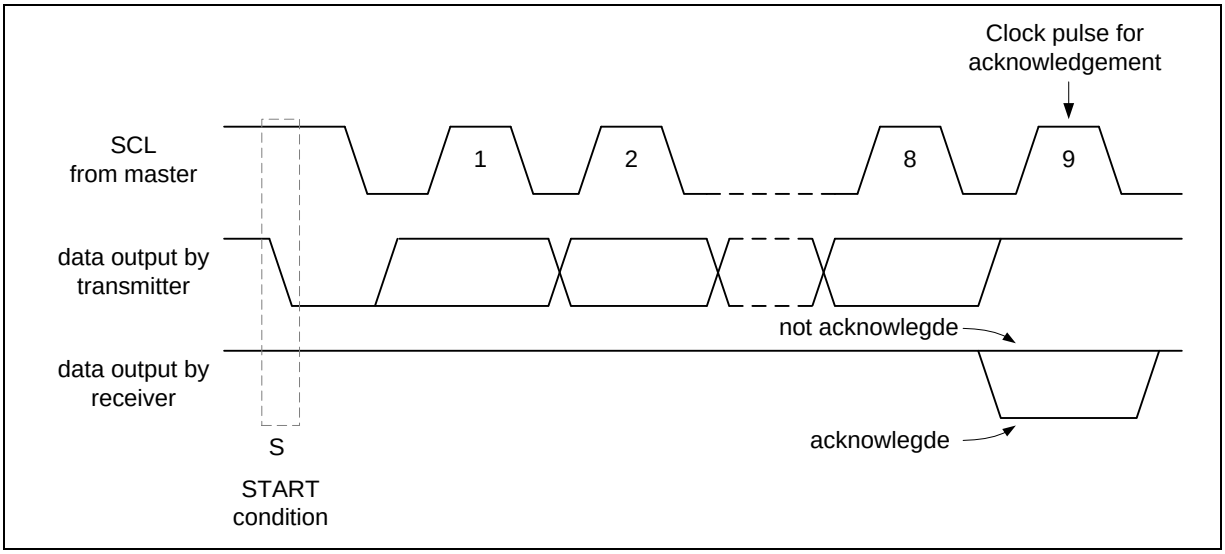


Figure 6-77 Acknowledge on the I²C Bus

6.13.5.1.5 Data transfer on the I2C-bus

The Figure 6-78 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.

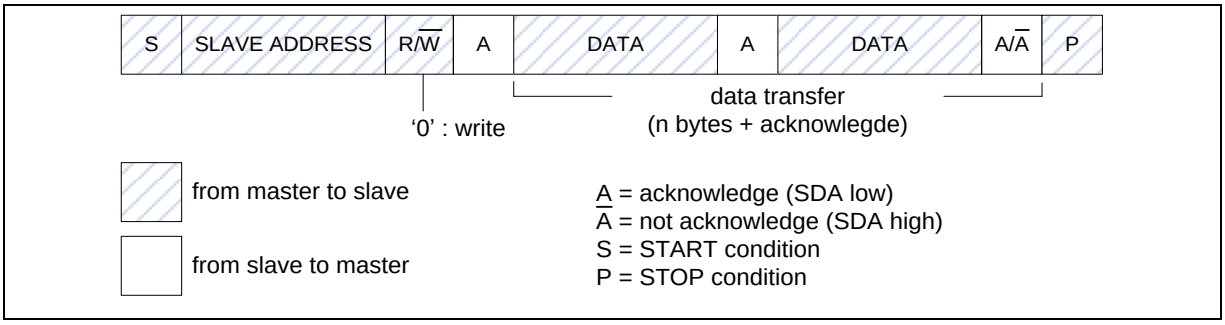


Figure 6-78 Master Transmits Data to Slave

The Figure 6-79 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.

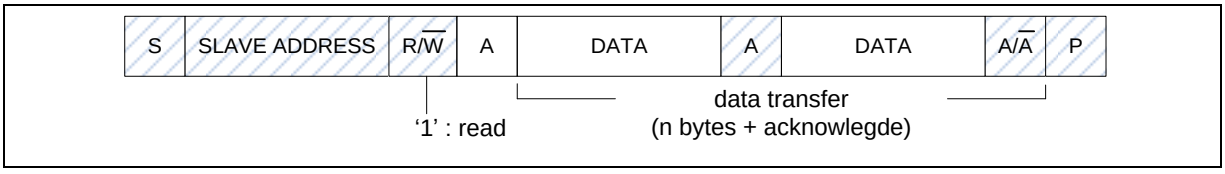


Figure 6-79 Master Reads Data from Slave

6.13.5.2 *Operation Mode*

The on-chip I²C 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 (I2CON[2]) 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 (I2CON[5:4]) and AA (I2CON[2]) are used to control the next state of the I^{2}C hardware after SI (I2CON[3]) flag is cleared. Upon completion of the new action, a new status code will be updated in I2CSTATUS register and the SI flag (I2CON[3]) 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 Figure 6-80 shows the current I^{2}C status code is 0x08, and then set DATA=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 AA, the I2CSTATUS will be updated by status code 0x18.

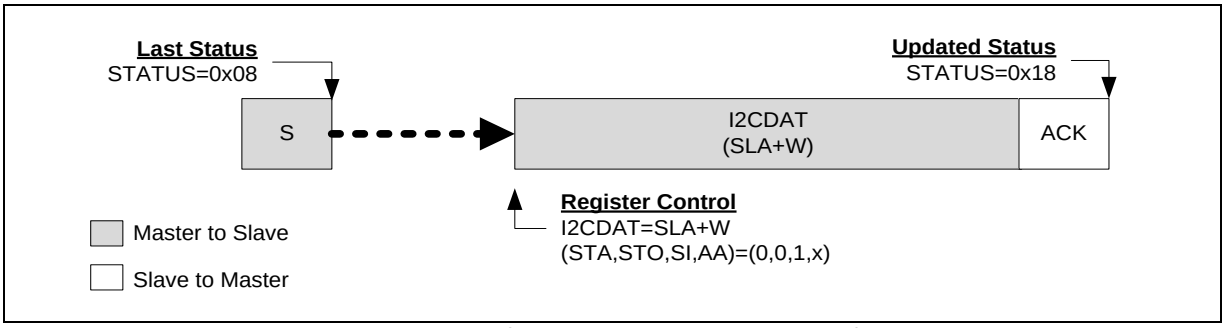


Figure 6-80 Control I^{2}C Bus according to Current I^{2}C Status

6.13.5.3 *Master Mode*

In the Figure 6-81 and Figure 6-82, 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 or Master receiver mode after START signal has been sent successfully and new status code would be 0x08. Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^{2}C protocol.

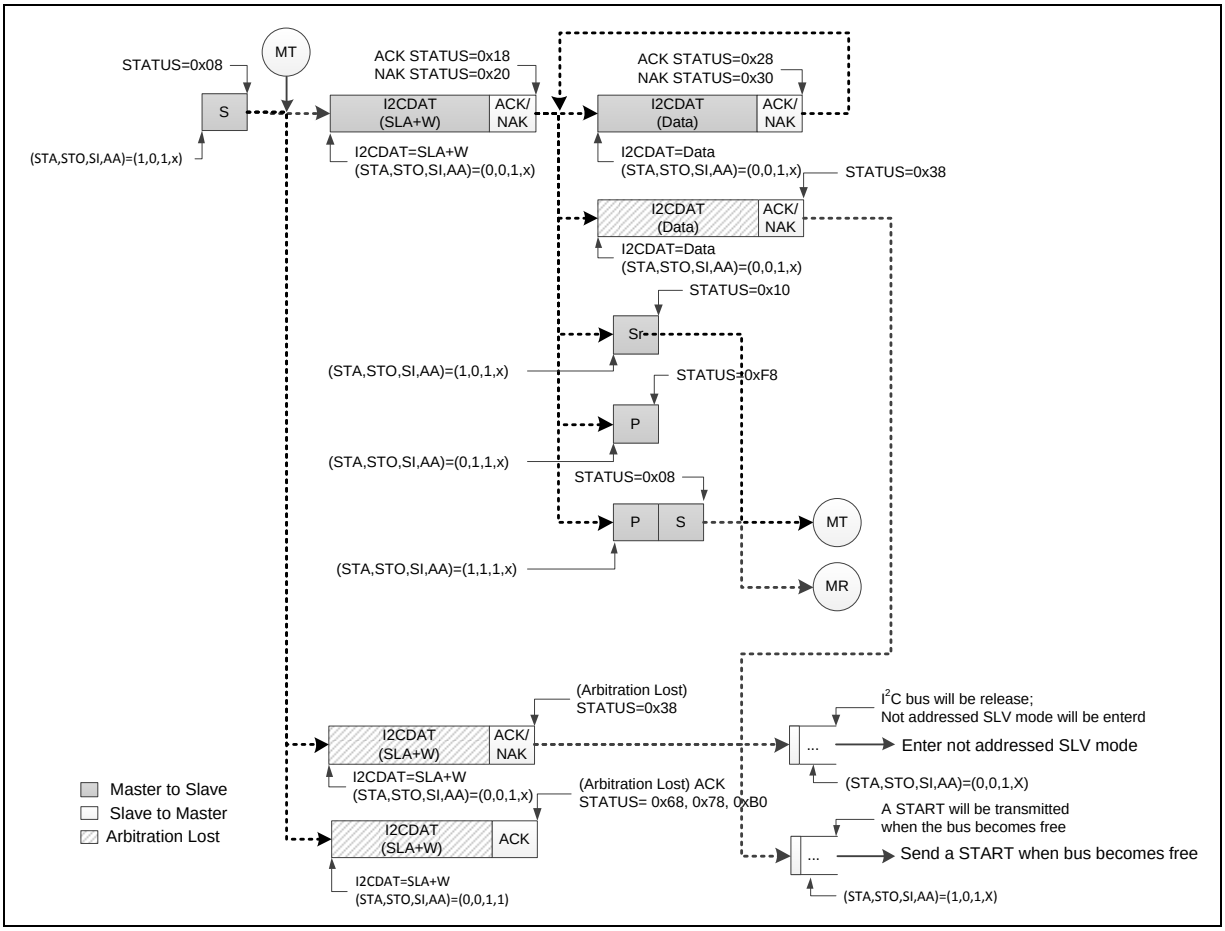


Figure 6-81 Master Transmitter Mode Control Flow

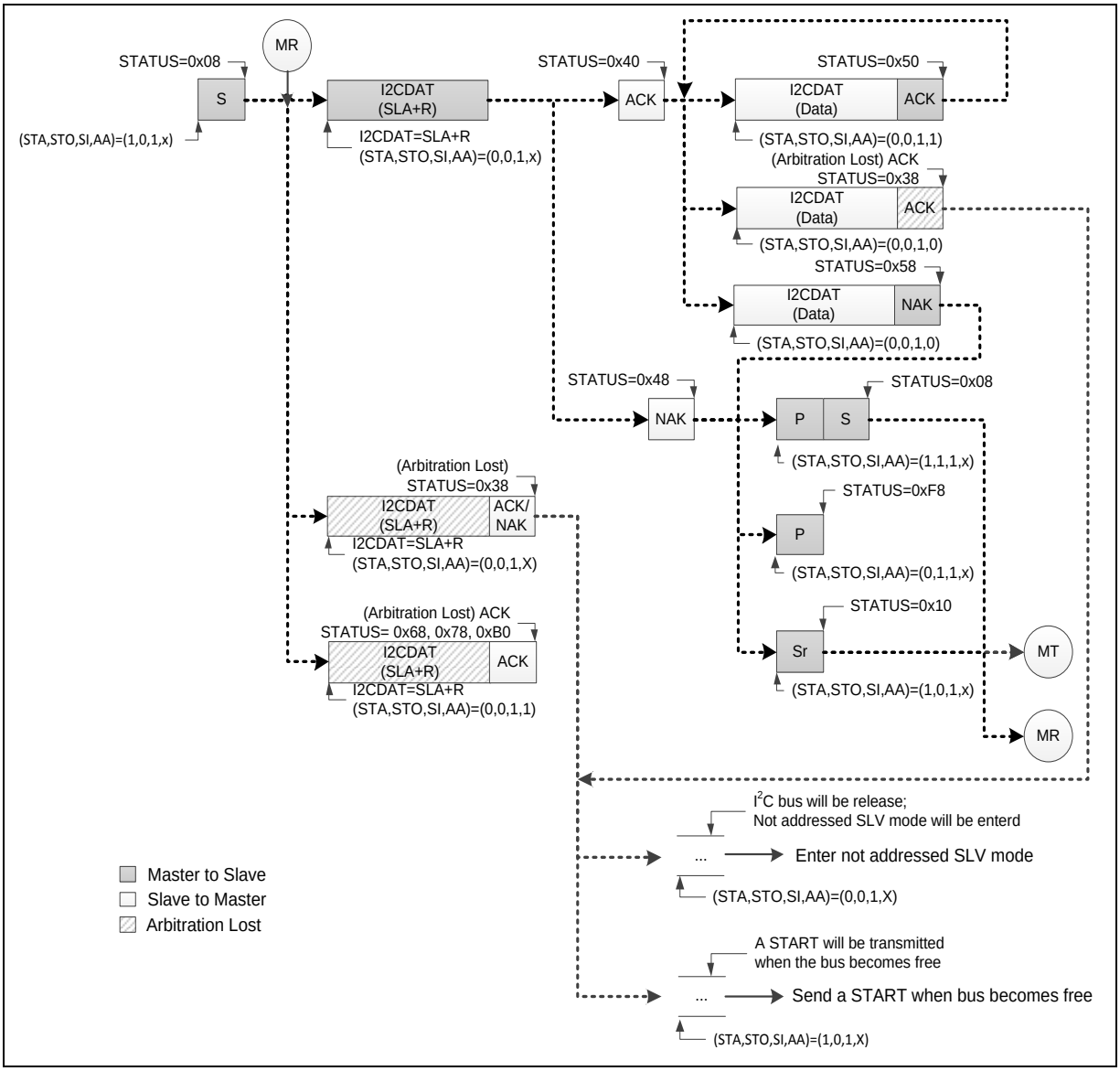


Figure 6-82 Master Receiver Mode Control Flow

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.

6.13.5.4 *Slave Mode*

When reset default, I^{2}C is not addressed and will not recognize the address on I^{2}C bus. User can set slave address by I2CADDRx and set (STA, STO, SI, AA) = (0, 0, 1, 1) to let I^{2}C recognize the address sent by master. The follow figure shows all the possible flow for I^{2}C in Slave mode. Users need to follow a proper flow to implement their own I^{2}C protocol.

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. 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 I^{2}C communication, the SCL clock will be released when writing '1' to clear SI flag in Slave mode.

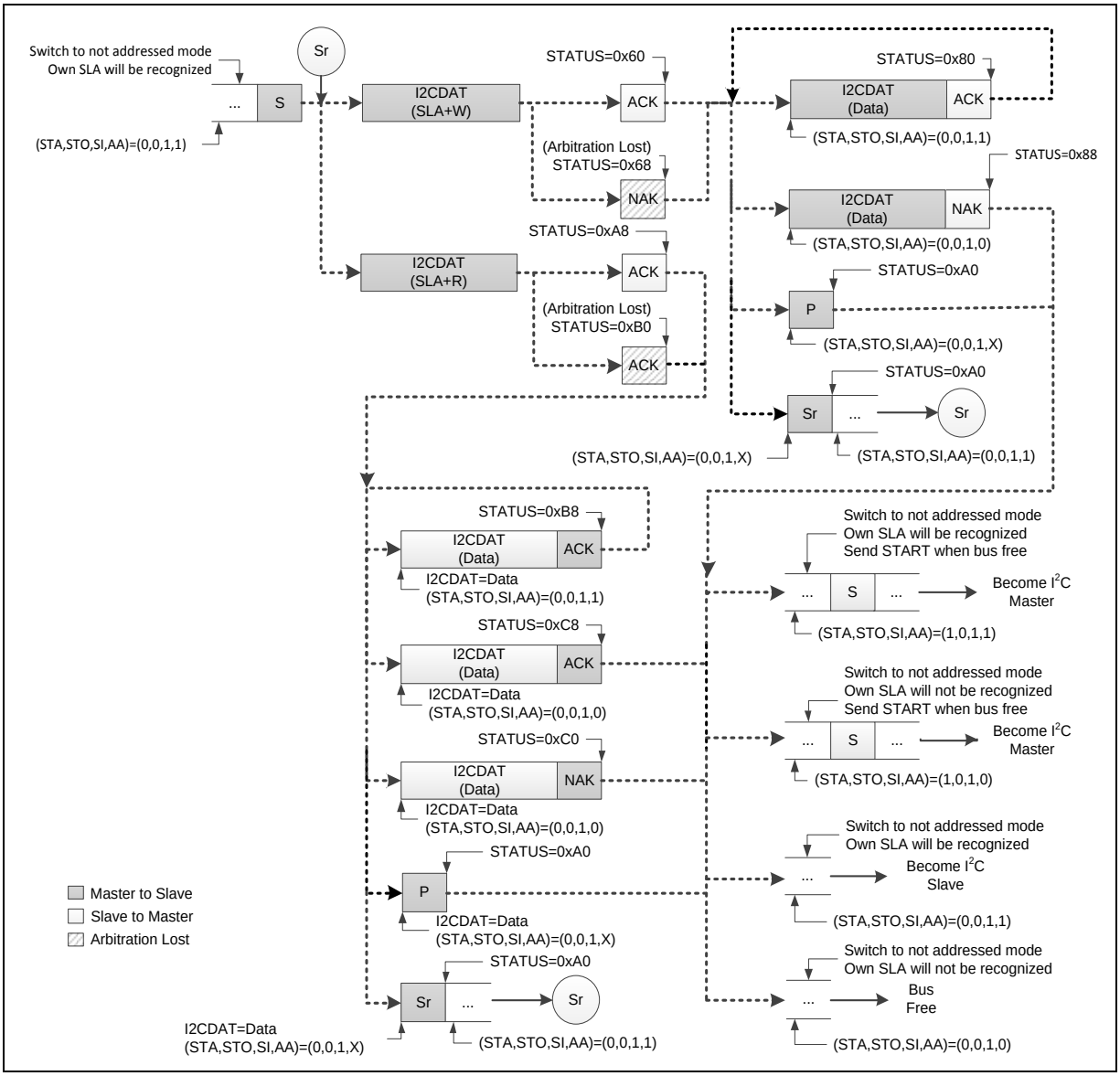


Figure 6-83 Slave Mode Control Flow

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.

6.13.5.5 *General Call (GC) Mode*

If the GC (I2CADDR[0]) bit is set to 1, 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.

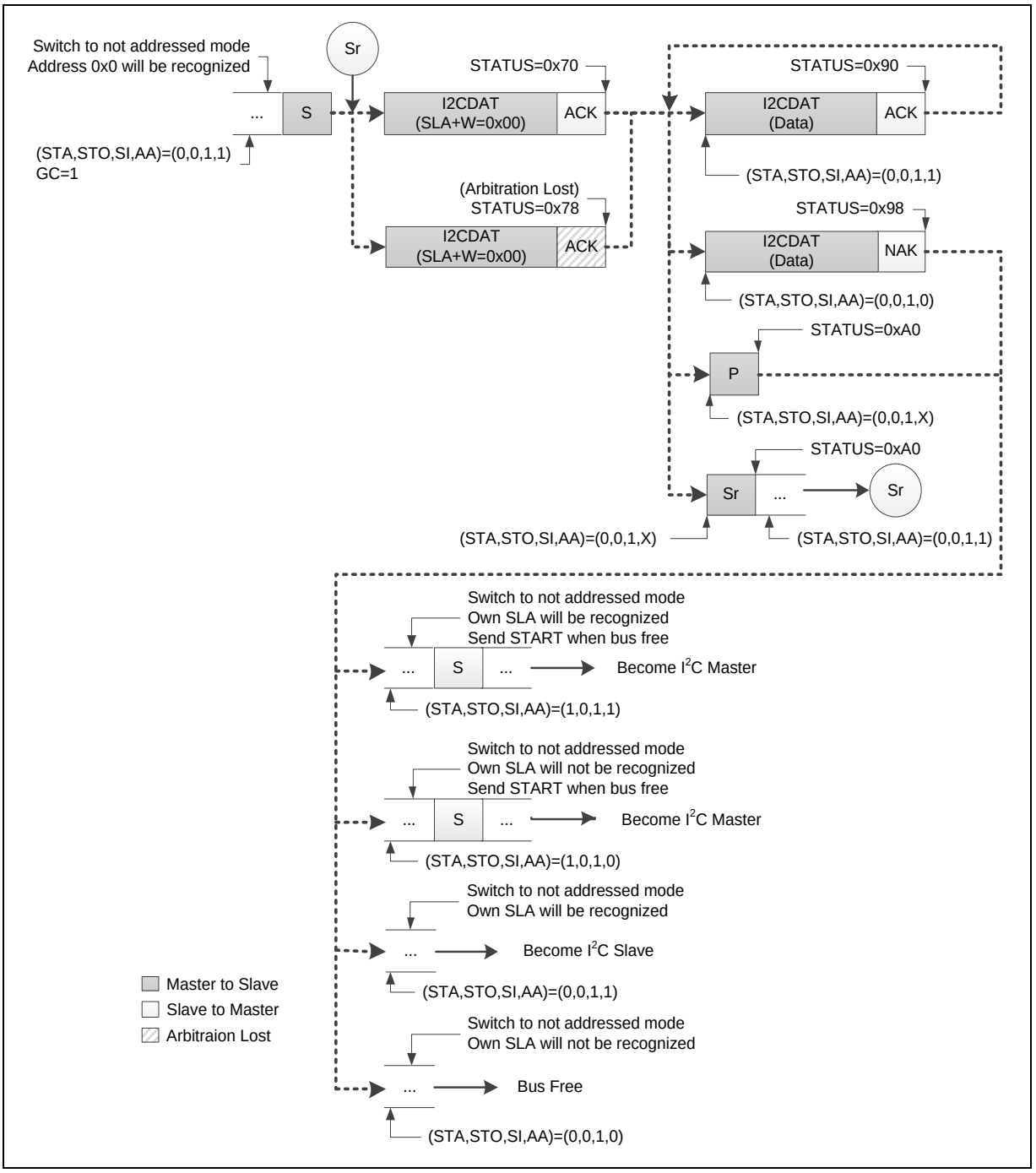


Figure 6-84 GC Mode

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 the above figure when getting 0xA0 status.

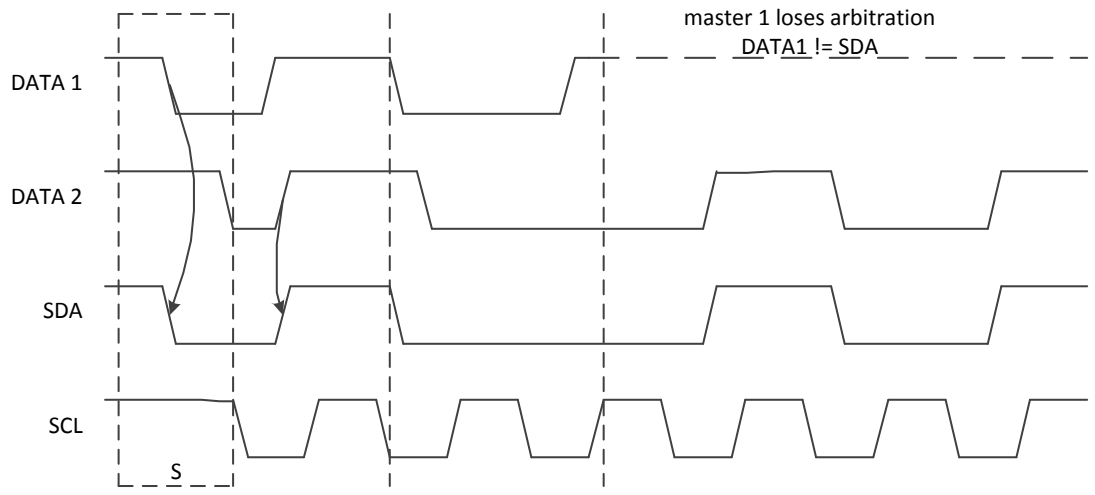
Note: After slave gets status of 0x98 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I^{2}C signal or address from master. At this time, I^{2}C controller should be reset to leave this status.

.

6.13.5.6 *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 supports multi-master by including collision detection and arbitration to prevent data corruption.

If for some reason two masters initiate command at the same time, the arbitration procedure determines which master wins and can continue with the command. Arbitration is performed on the SDA signal while the SCL signal is high. Each master checks if the SDA signal on the bus corresponds to the generated SDA signal. If the SDA signal on the bus is low but it should be high, then this master has lost arbitration. The device that has lost arbitration can generate SCL pulses until the byte ends and must then release the bus and go into slave mode. The arbitration procedure can continue until all the data is transferred. This means that in multi-master system each master must monitor the bus for collisions and act accordingly.



Arbitration procedure of two masters

Figure 6-85 Arbitration Lost

- When I2CSTATUS = 0x38, an “Arbitration Lost” is received. Arbitration lost event maybe occur during the send START bit, data bits or STOP bit. User could set (STA, STO, SI, AA) = (1, 0, 1, X) to send START again when bus free, or set (STA, STO, SI, AA) = (0, 0, 1, X) 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

6.13.5.7 *Protocol Registers*

To control I^{2}C port through the following 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). All bit 31~ bit 8 of these I²C special function registers are reserved. These bits do not have any functions and are all 0 if read back.

6.13.5.7.1 Address Registers (I2CADDR)

I²C port is equipped with four slave address registers I2CADDRn (n=0~3). The contents of the register are irrelevant when I²C is in Master mode. In the Slave mode, the bit field I2CADDRn[7:1] must be loaded with the chip's own slave address. The I²C hardware will react if the contents of I2CADDRn are matched with the received slave address.

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

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

6.13.5.7.2 Data Registers (I2CDAT)

This register contains a byte of serial data to be transmitted or a byte which just has been received. The CPU can read from or write to this 8-bit (I2CDAT [7:0]) directly while it is not in the process of shifting a byte. When I²C is in a defined state and the serial interrupt flag (SI) is set, data in I2CDAT [7:0] remains stable. While data is being shifted out, data on the bus is simultaneously being shifted in; I2CDAT [7:0] always contains the last data byte present on the bus.

The acknowledge bit is controlled by the I²C hardware and cannot be accessed by the CPU. Serial data is shifted through into I2CDAT [7:0] on the rising edges of serial clock 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. In order to monitor bus status while sending data, the bus data will be shifted to I2CDAT[7:0] when sending I2CDAT[7:0] to bus. In the case of sending data, serial data bits are shifted out from I2CDAT [7:0] on the falling edges of SCL clock, and is shifted into I2CDAT [7:0] on the rising edges of SCL clock.

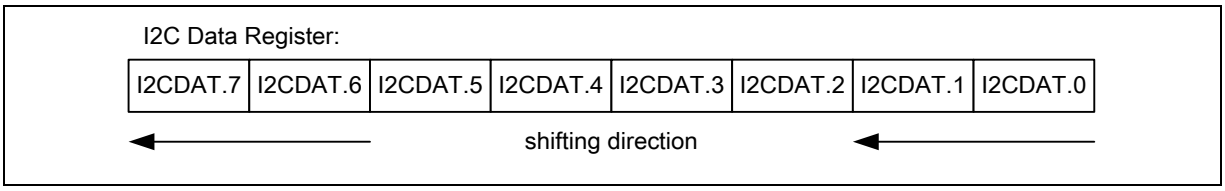


Figure 6-86 I²C Data Shifting Direction

6.13.5.7.3 Control Register (I2CON)

The CPU can read from and write to I2CON [7:0] directly. When the I²C port is enabled by setting ENSI (I2CON [6]) to high, the internal states will be controlled by I2CON and I²C logic hardware.

There are two bits are affected by hardware: the SI (I2CON[3]) bit is set when the I²C hardware requests a serial interrupt, and the STO (I2CON[4]) bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when ENSI = 0.

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

6.13.5.7.4 Status Register (I2CSTATUS)

I2CSTATUS [7:0] is an 8-bit read-only register. The three least significant bits are always 0. The bit field I2CSTATUS [7:3] contain the status code and there are 26 possible status codes. All states are listed in section 5.6.6. When I2CSTATUS [7:0] contains F8H, no serial interrupt is requested. All other I2CSTATUS [7:3] values correspond to defined I^2C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2CSTATUS[7:0] one cycle after SI is set by hardware and is still present one cycle after SI has been 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 illegal position in the 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^2C from bus error, STO should be set and SI should be clear to enter not addressed Slave mode. Then STO is cleared to release bus and to wait new communication. I^2C bus cannot recognize stop condition during this action when 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 6-29 I^2C Status Code Description

6.13.5.7.5 I^2C Clock Baud Rate Bits (I2CLK)

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

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

6.13.5.7.6 The I²C Time-out Counter Register (I2CTOC)

There is a 14-bit time-out counter which can be used to deal with the I²C bus hang-up. If the time-out counter is enabled, the counter starts counting up until it overflows (TIF (I2CTOC[0])=1) and generates I²C interrupt to CPU or stops counting by clearing ENTI (I2CTOC[2]) to 0. When time-out counter is enabled, setting SI (I2CON[3]) flag to high will reset counter and re-start counting up after SI is cleared. If I²C bus hangs up, it causes the I2CSTATUS and flag SI are not updated for a period, the 14-bit time-out counter may overflow and acknowledge CPU the I²C interrupt. Refer to the Figure 6-87 for the 14-bit time-out counter. User may write 1 to clear TIF to 0.

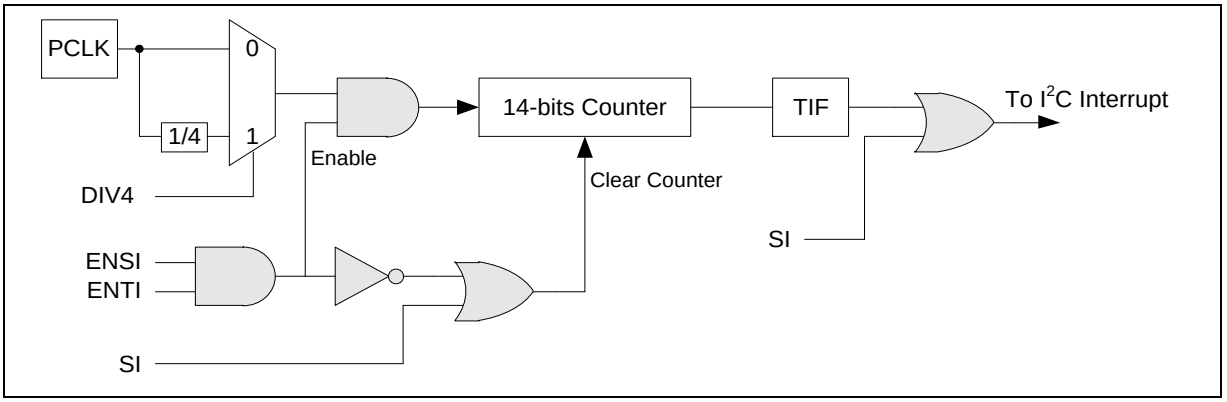


Figure 6-87: I²C Time-out Count Block Diagram

6.13.5.7.7 Slave Address Mask Register (I2CADM)

The I²C bus controllers support multiple address recognition with four address mask registers I2CADMn (n=0~3). When the bit in the address mask register is set to 1 it means the received corresponding address bit is “Don’t-care”. If the bit is set to 0, that means the received corresponding register bit should be exact the same as address register.

6.13.5.7.8 The I²C wake-up control Register (I2CWKUPCON)

When chip enters Power-down mode, other I²C master can wake-up our chip by addressing our I²C device, user must configure the related setting before entering sleep mode. When the chip is woken-up by address match with one of the four address register, the following data will be abandoned at this time.

6.13.5.7.9 The I²C wake-up status Register (I2CWKUPSTS)

When system is woken up by other I²C master device, WKUPIF is set to indicate this event. User needs write “1” to clear this bit.

6.13.6 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: **I2Cn_BA = 0x4002_0000 + (0x100000 *n)** **n= 0,1**				
I2CON	I2Cx_BA+0x00	R/W	I^{2}C Control Register	0x0000_0000
I2CADDR0	I2Cx_BA+0x04	R/W	I^{2}C Slave Address Register0	0x0000_0000
I2CDAT	I2Cx_BA+0x08	R/W	I^{2}C DATA Register	0x0000_0000
I2CSTATUS	I2Cx_BA+0x0C	R	I^{2}C Status Register	0x0000_00F8
I2CLK	I2Cx_BA+0x10	R/W	I^{2}C Clock Divided Register	0x0000_0000
I2CTOC	I2Cx_BA+0x14	R/W	I^{2}C Time-out Control Register	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
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
I2CWKUPCON	I2Cx_BA+0x3C	R/W	I^{2}C Wake-up Control Register	0x0000_0000
I2CWKUPSTS	I2Cx_BA+0x40	R	I^{2}C Wake-up Status Register	0x0000_0000

6.13.7 Register Description

I²C Control Register (I2CON)

Register	Offset	R/W	Description	Reset Value
I2CON	I2Cn_BA+0x00	R/W	I²C Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EI	**ENS1**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**EI**	**Interrupt Enable Bit** Set to enable I²C interrupt. 0 = I²C interrupt Disabled. 1 = I²C interrupt Enabled.
[6]	**ENS1**	**I²C Controller Enable Bit** Set to enable I²C serial function controller. When ENS1=1 the I²C serial function enable. The multi-function pin function must set to SDA, and SCL of I²C function first. 0 = I²C function Disabled. 1 = I²C function Enabled.
[5]	**STA**	**I²C START Control** Setting STA to logic 1 to enter Master mode, the I²C hardware sends a START or repeat START condition to bus when the bus is free.
[4]	**STO**	**I²C STOP Control** In Master mode, set this bit to 1 to transmit a STOP condition to bus then the controller will check the bus condition if a STOP condition is detected and this bit will be cleared by hardware automatically.
[3]	**SI**	**I²C Interrupt Flag** When a new I²C state is present in the I2CSTATUS register, this bit will be set automatically, and if the EI (I2CON [7]) bit is set, the I²C interrupt is requested. SI must be cleared by software. Clear SI by writing 1 to this bit.
[2]	**AA**	**Assert Acknowledge Control Bit** When AA =1 prior to address or data is received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.). A slave is acknowledging the address sent from master, 2.) The receiver devices are acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line.

[1:0]	**Reserved**	Reserved.

I²C Data Register (I2CDAT)

Register	Offset	R/W	Description	Reset Value
I2CDAT	I2Cn_BA+0x08	R/W	I²C Data Register	0x0000_0000

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

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

I²C Status Register (I2CSTATUS)

Register	Offset	R/W	Description	Reset Value
I2CSTATUS	I2Cn_BA+0x0C	R/W	I²C Status Register	0x0000_00F8

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**I2CSTATUS**	**I²C Status Register (Read Only)** The three least significant bits are always 0. The five most significant bits contain the status code. There are 28 possible status codes. When the content of I2CSTATUS is F8H, no serial interrupt is requested. Others I2CSTATUS values correspond to defined I²C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2CSTATUS one cycle after SI is set by hardware and is still present one cycle after SI has been reset by software. In addition, states 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the formation frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit.

I²C Clock Divided Register (I2CLK)

Register	Offset	R/W	Description	Reset Value
I2CLK	I2Cn_BA+0x10	R/W	I²C Clock Divided Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**I2CLK**	**I²C Clock Divided Register** Indicates the I²C clock rate: Data Baud Rate of I²C = (system clock) / (4x (I2CLK+1)). **Note:** The minimum value of CLK_DIV is 4.

I²C Time-Out Counter Register (I2CTOC)

Register	Offset	R/W	Description	Reset Value
I2CTOC	I2Cn_BA+0x14	R/W	I²C Time-Out Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ENTI**	**DIV4**	**TIF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ENTI**	**Time-out Counter Enable Bit** When Enabled, the 14-bit time-out counter will start counting when SI is cleared. Setting the flag SI to '1' will reset counter and re-start up counting after SI is cleared. 0 = Time-out counter Disabled. 1 = Time-out counter Enabled.
[1]	**DIV4**	**Time-out Counter Input Clock Divided by 4** When Enabled, the time-out period is extend 4 times. 0 = Time-out counter input clock divided by 4Disabled. 1 = Time-out counter input clock divided by 4Enabled.
[0]	**TIF**	**Time-out Flag** This bit is set by hardware when I²C time-out happened and it can interrupt CPU if I²C interrupt enable bit (EI) is set to 1. **Note:** This bit can be cleared by software writing '1'.

I²C Slave Address Register (I2CADDRx)

Register	Offset	R/W	Description	Reset Value
I2CADDR0	I2Cn_BA+0x04	R/W	I²C Slave Address Register0	0x0000_0000
I2CADDR1	I2Cn_BA+0x18	R/W	I²C Slave Address Register1	0x0000_0000
I2CADDR2	I2Cn_BA+0x1C	R/W	I²C Slave Address Register2	0x0000_0000
I2CADDR3	I2Cn_BA+0x20	R/W	I²C Slave Address Register3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CADDR							**GC**

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

I²C Slave Address Mask Register (I2CADMx)

Register	Offset	R/W	Description	Reset Value
I2CADM0	I2Cn_BA+0x24	R/W	I²C Slave Address Mask Register0	0x0000_0000
I2CADM1	I2Cn_BA+0x28	R/W	I²C Slave Address Mask Register1	0x0000_0000
I2CADM2	I2Cn_BA+0x2C	R/W	I²C Slave Address Mask Register2	0x0000_0000
I2CADM3	I2Cn_BA+0x30	R/W	I²C Slave Address Mask Register3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I2CADM							**Reserved**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**I2CADM**	**I²C Address Mask Register** 0 = Mask Disabled (the received corresponding register bit should be exactly the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don't care.). I²C bus controllers support multiple address recognition with four address mask register. When the bit in the address mask register is set to one, it means the received corresponding address bit is don't-care. If the bit is set to zero, that means the received corresponding register bit should be exact the same as address register.
[0]	**Reserved**	Reserved.

I²C Wake-up Control Register (I2WKUPCON)

Register	Offset	R/W	Description	Reset Value
I2CWKUPCON	I2Cn_BA+0x3C	R/W	I²C Wake-up Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKUPEN**	**I²C Wake-up Function Enable Bit** 0 = I²C wake-up function Disabled. 1 = I²C wake-up function Enabled.

I²C Wake-up StatusRegister (I2WKUPSTS)

Register	Offset	R/W	Description	Reset Value
I2CWKUPSTS	I2C_BA+0x40	R	I²C wake-up status register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKUPIF**	**Wake-up Interrupt Flag** When chip is woken up from Power-down mode by I²C, this bit is set to 1. **Note:** This bit can be cleared by software writing '1'.

6.14 Serial Peripheral Interface (SPI)

6.14.1 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. This NuMicro® NUC123 series contains up to three sets of SPI controllers performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. Each set of SPI controller can be configured as a Master or a Slave device.

This controller supports variable serial clock function for special application and it also supports 2-bit Transfer mode. The controller also supports PDMA function to access the data buffer and also supports Dual I/O transfer mode.

6.14.2 Features

- Up to three sets of SPI controllers
- Supports Master or Slave mode operation
- Supports 2-bit Transfer mode
- Supports Dual I/O transfer mode
- Configurable bit length of a transfer word from 8 to 32-bit
- Provide separate 8-layer depth transmit and receive FIFO buffers
- Supports MSB first or LSB first transfer sequence
- Up to two slave select lines in Master mode
- Supports Byte Reorder function
- Supports configurable suspend interval in Master mode
- Variable output serial clock frequency in Master mode
- Supports PDMA transfer
- Supports 3-Wire, no slave select signal, bi-direction interface

6.14.3 Block Diagram

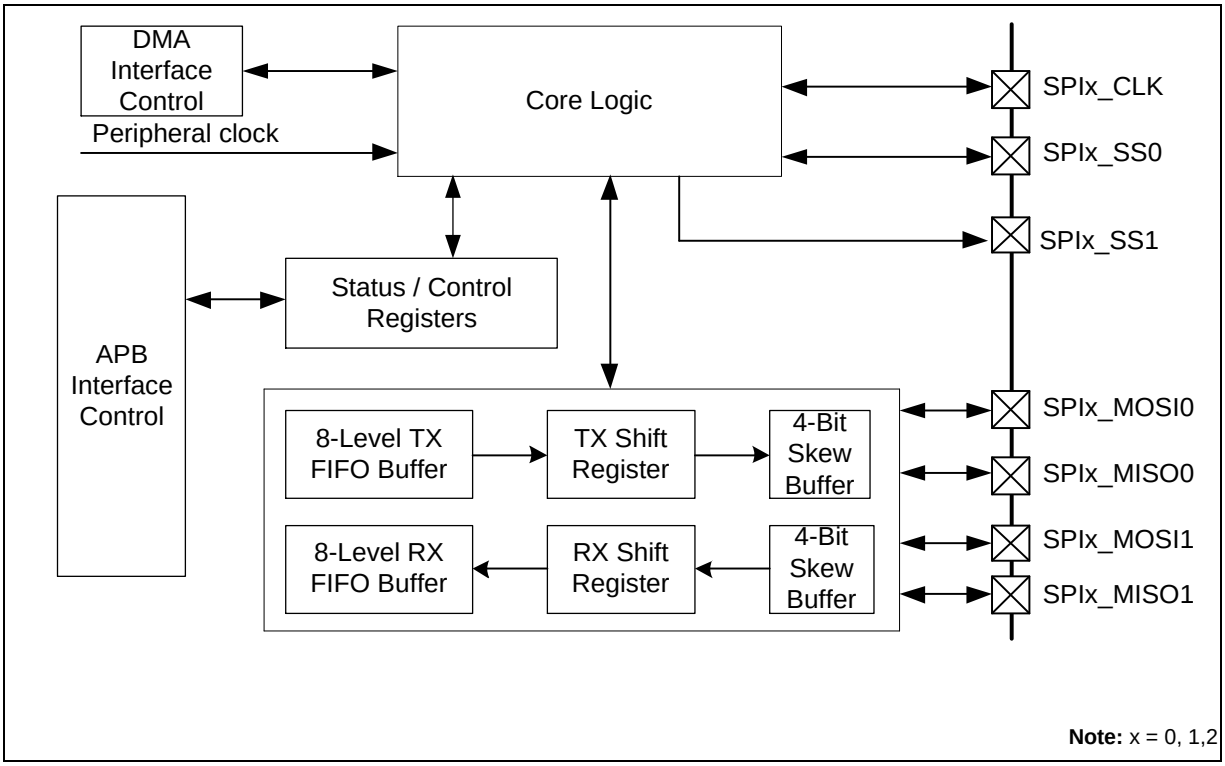


Figure 6-88 SPI Block Diagram

TX FIFO Buffer:

The transmit FIFO buffer is an 8-level depth, 32-bit wide, first-in, first-out register buffer. The data can be written to the transmit FIFO buffer in advance through software by writing the SPI_TX register.

RX FIFO Buffer:

The received FIFO buffer is also an 8-level depth, 32-bit wide, first-in, first-out register buffer. The receive control logic will store the received data to this buffer. The FIFO buffer data can be read from SPI_RX register by software.

TX Shift Register:

The transmit shift register is a 32-bit wide register buffer. The transmit data is loaded from the TX FIFO buffer and shifted out bit-by-bit to the skew buffer.

RX Shift Register:

The receive shift register is also a 32-bit wide register buffer. The receive data is shifted in bit-by-bit from the skew buffer and is loaded into RX FIFO buffer when a transaction done.

Skew Buffer:

The skew buffer is a 4-bit buffer.

For transmitting, it is written from shift register by peripheral clock and read out by SPI bus clock. The first three bits of TX shift register will be loaded to skew buffer in the beginning of transmission. The remaining bits will be shifted into skew buffer as any bit in skew buffer is shifted out to SPI bus.

For receiving, the serial data captured from SPI bus is written into the skew buffer basing on the SPI bus clock. These captured data will be read out and written into the RX shift register basing on the SPI peripheral clock.

6.14.4 Basic Configuration

The basic configurations of SPI0 are as follows:

- SPI0 pins are configured in GPB_MFP, GPC_MFP, GPD_MFP and ALT_MFP registers. NUC123xxxAEx provides the alternative of configuring the SPI0 pins in GPB_MFPH, GPC_MFPL and GPD_MFPL registers. (For NUC123xxxAEx, if GPB_MFPH, GPC_MFPL and GPD_MFPL are used as pin multi-function setting, the GPB_MFP, GPC_MFP, GPD_MFP and ALT_MFP will become invalid).
- Enable SPI0 peripheral clock in SPI0_EN (APBCLK[12]).
- Reset SPI0 controller in SPI0_RST (IPRSTC2[12]).

The basic configurations of SPI1 are as follows:

- SPI1 pins are configured in GPA_MFP, GPB_MFP, GPC_MFP, GPD_MFP and ALT_MFP registers. NUC123xxxAEx provides the alternative of configuring the SPI1 pins in GPA_MFPH, GPB_MFPL, GPB_MFPH, GPC_MFPH and GPD_MFPH registers. (For NUC123xxxAEx, if GPA_MFPH, GPB_MFPL, GPB_MFPH, GPC_MFPH and GPD_MFPH are used as pin multi-function setting, the GPA_MFP, GPB_MFP, GPC_MFP, GPD_MFP and ALT_MFP will become invalid).

- Enable SPI1 peripheral clock in SPI1_EN (APBCLK [13]).
- Reset SPI1 controller in SPI1_RST (IPRSTC2[13]).

The basic configurations of SPI2 are as follows:

- SPI2 pins are configured in GPA_MFP, GPB_MFP, GPD_MFP and ALT_MFP registers. NUC123xxxAEx provides the alternative of configuring the SPI2 pins in GPA_MFPH, GPB_MFPL and GPD_MFPL registers. (For NUC123xxxAEx, if GPA_MFPH, GPB_MFPL and GPD_MFPL are used as pin multi-function setting, the GPA_MFP, GPB_MFP, GPD_MFP and ALT_MFP will become invalid).
- Enable SPI2 peripheral clock in SPI2_EN (APBCLK [14]).
- Reset SPI2 controller in SPI2_RST (IPRSTC2[14]).

6.14.5 Functional Description

6.14.5.1 *Terminology*

SPI Peripheral clock and SPI Bus Clock

The SPI controller needs the peripheral clock to drive the SPI logic unit to perform the data transfer. The peripheral clock rate is determined by the settings of clock source which can be PLL out or the PCLK, BCn (SPI_CNTRL2[31]) option and clock divisor (SPI_DIVIDER). SPIx_S (x=0,1,2) of CLKSEL1 register determines the clock source of the SPI peripheral clock. Set the BCn (SPI_CNTRL2[31]) bit to 0 for the compatible SPI clock rate calculation of previous products. The DIVIDER (SPI_DIVIDER[7:0]) setting determines the divisor of the clock rate calculation.

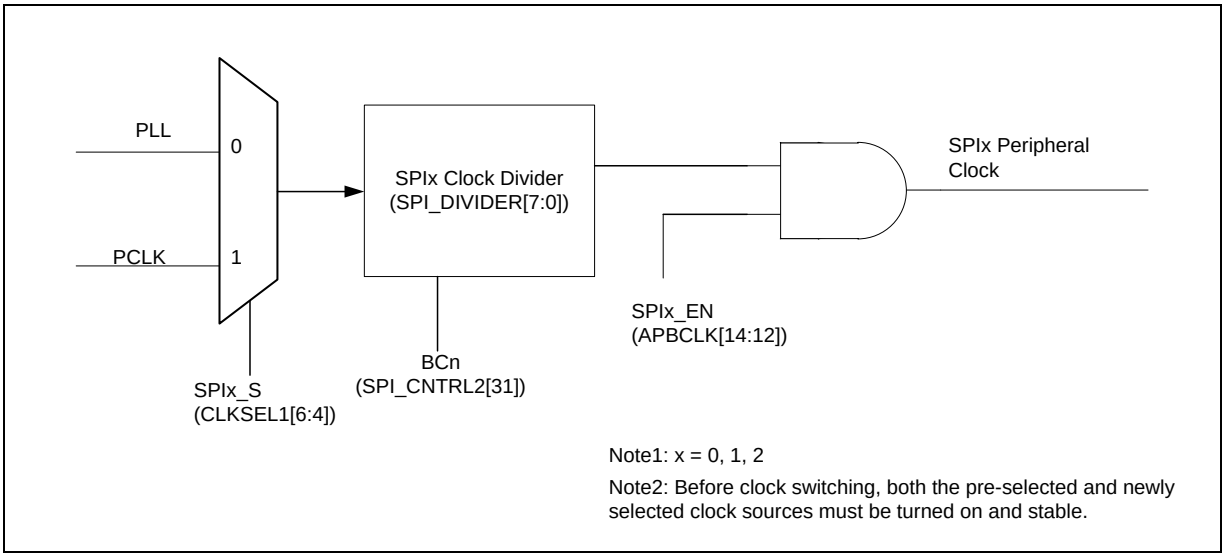


Figure 6-89 SPI Peripheral Clock

In Master mode, if the variable clock function is disabled, the frequency of the SPI bus clock is equal to the SPI peripheral clock rate. In general, the SPI bus clock is denoted as SPI clock. In Slave mode, the SPI bus clock is provided by an off-chip Master device. The SPI peripheral clock rate of Slave device must be faster than the bus clock rate of the Master device connected together. The frequency of SPI peripheral clock cannot be faster than the system clock rate regardless of Master or Slave mode.

Master/Slave mode

This SPI controller can be set as Master or Slave mode by setting the SLAVE (SPI_CNTRL[18]) to communicate with the off-chip SPI Slave or Master device. The application block diagrams are shown below.

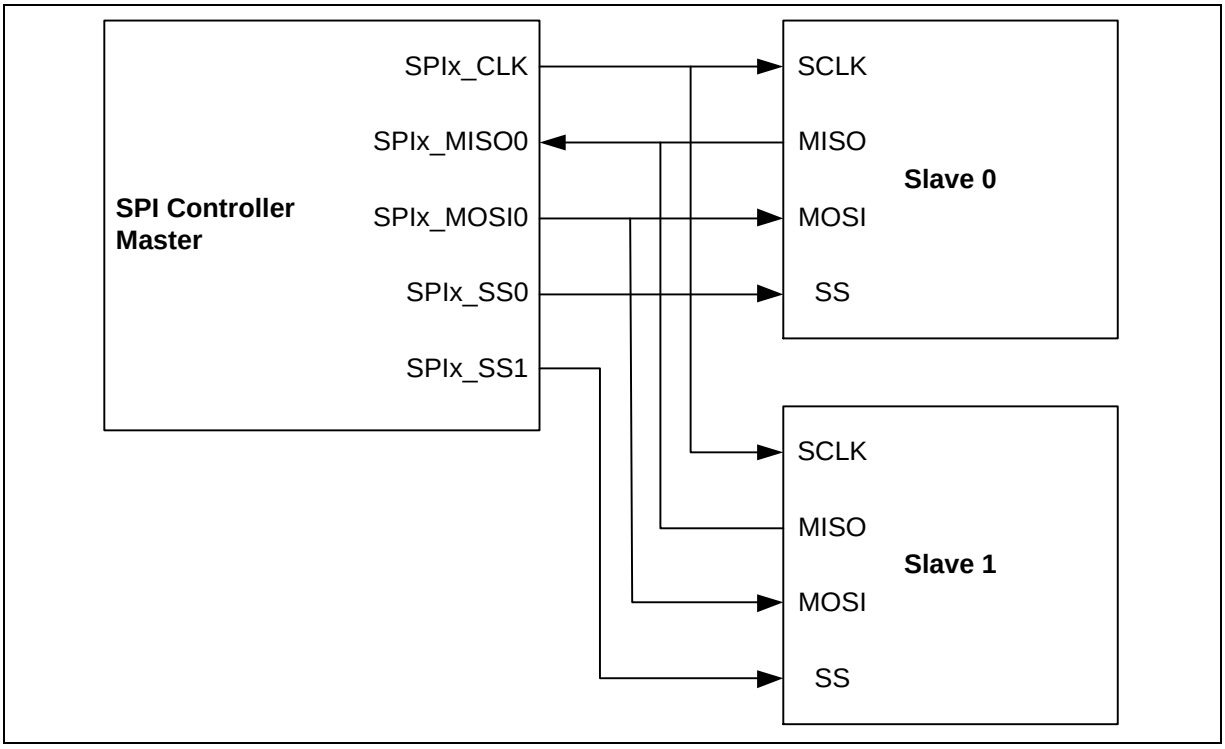


Figure 6-90 SPI Master Mode Application Block Diagram

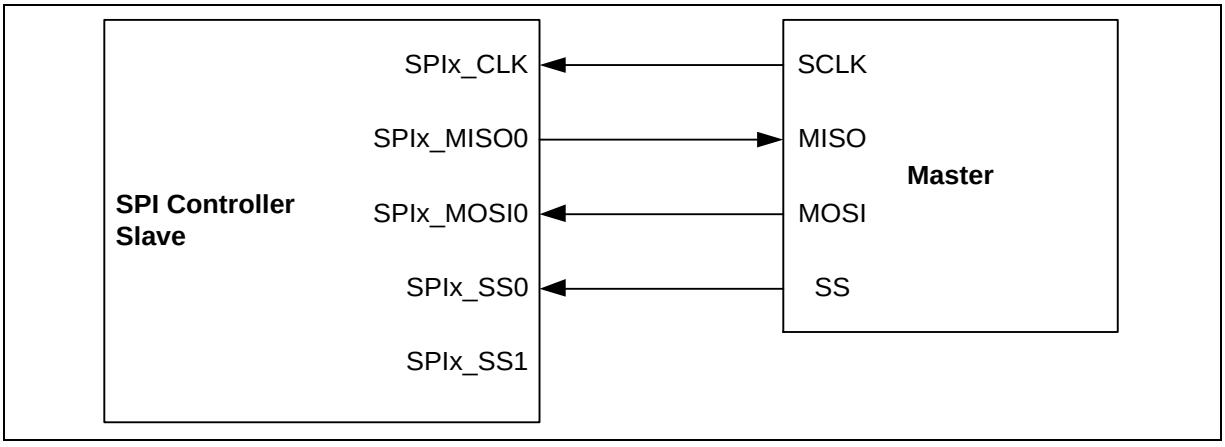


Figure 6-91 SPI Slave Mode Application Block Diagram

Slave Selection

In Master mode, this SPI controller can drive up to two off-chip Slave devices through the slave select output pins SPIx_SS0 and SPIx_SS1. In Slave mode, the off-chip Master device drives the slave select signal from the SPIx_SS0 input pin to this SPI controller. The duration between the slave select active edge and the first SPI clock input shall over 3 SPI peripheral clock cycles of Slave.

In Master/Slave mode, the active state of slave select signal can be programmed to low active or high

active in SS_LVL (SPI_SSR[2]). The selection of slave select condition depends on what type of device is connected.

In Slave mode, the SS_LTRIG (SPI_SSR[4]) defines the slave select signal SPIx_SS0/1 is level trigger or edge trigger. If the SS_LTRIG bit is configured as level trigger, the LTRIG_FLAG bit (SPI_SSR[5]) is used to indicate if the received bits among one transaction meets the requirement defined in TX_BIT_LEN (SPI_CNTRL[7:3]). To recognize the inactive state of the slave selection signal, the inactive period of the slave selection signal must be larger than or equal to 3 peripheral clock cycles between two successive transactions.

Level-trigger / Edge-trigger

In Slave mode, the slave select signal can be configured as level-trigger or edge-trigger. In edge-trigger, the data transfer starts from an active edge and ends on an inactive edge. If Master does not send an inactive edge to Slave, the transfer procedure will not be completed and the unit transfer interrupt flag of Slave will not be set. In level-trigger, the following two conditions will terminate the transfer procedure and the unit transfer interrupt flag of Slave will be set. The first condition is that if the number of transferred bits matches the setting of TX_BIT_LEN (SPI_CNTRL[7:3]), 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 is in progress, it will force Slave device to terminate the current transfer no matter how many bits have been transferred and the unit transfer interrupt flag will be set. User can read the status of LTRIG_FLAG (SPI_SSR[5]) to check if the data has been transferred completely.

Timing Condition

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

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

RX_NEG (SPI_CNTRL[1]) defines the data received either on negative edge or on positive edge of SPI bus clock.

Note: The settings of TX_NEG and RX_NEG are mutual exclusive. In other words, do not transmit and receive data at the same clock edge.

Transmit/Receive Bit Length

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

When SPI controller finishes a transaction, i.e. receives or transmits a special count of bits defined in TX_BIT_LEN (SPI_CNTRL[7:3]), the unit transfer interrupt flag will be set to 1.

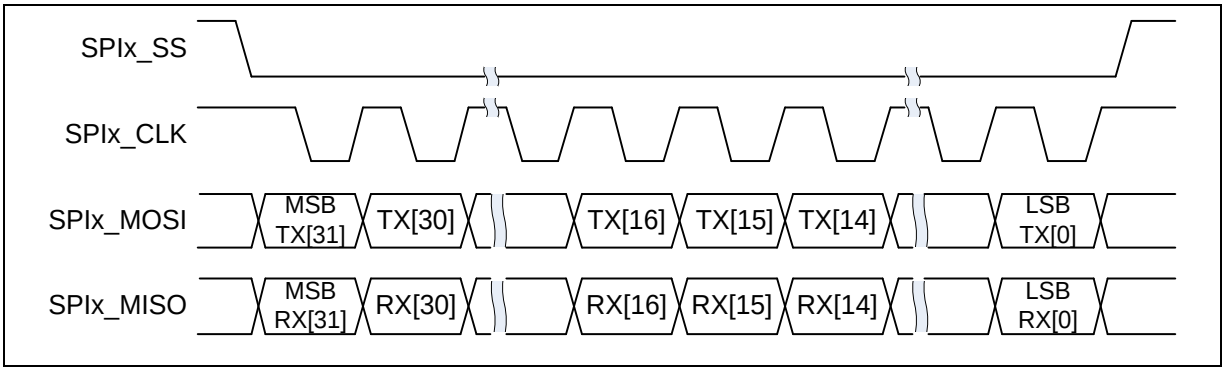


Figure 6-92 32-Bit in One Transaction (Master Mode)

LSB/MSB First

LSB (SPI_CNTRL[10]) defines the bit transfer sequence in a transaction. If the LSB (SPI_CNTRL[10]) is set to 1, the transfer sequence is LSB first. The bit 0 will be transferred firstly. If the LSB (SPI_CNTRL[10]) is cleared to 0, the transfer sequence is MSB first.

6.14.5.2 *Automatic Slave Selection*

In Master mode, if AUTOSS (SPI_SSR[3]) is set, the slave select signals will be generated automatically and output to SPISSx0 and SPISSx1 pins according to SSR[0] (SPI_SSR[0]) and SSR[1] (SPI_SSR[1]) whether be enabled or not. The slave selection signal will be set to active state automatically when the SPI data transfer is started by writing to TX FIFO in FIFO mode or by setting the GO_BUSY (SPI_CNTRL[0]) when the FIFO mode is disabled. It will be set to inactive state when SPI bus is idle. If SPI bus is not idle, i.e. TX FIFO, TX shift register or TX skew buffer is not empty, the slave selection signal will be set to inactive state between transactions if the value of SP_CYCLE (SPI_CTL[15:12]) is greater than or equal to 3.

In Master mode, if the value of SP_CYCLE[3:0] is less than 3 and the AUTOSS is set as 1, the slave select signal will keep at active state between two successive transactions.

If the AUTOSS bit is cleared, the slave select output signals will be asserted/de-asserted by manual setting/clearing the related bits of SPI_SSR[1:0]. The active state of the slave select output signals is specified in SS_LVL bit (SPI_SSR[2]).

In Slave mode, to recognize the inactive state of the slave select signal, the inactive period of the slave select signal must be larger than or equal to 6 peripheral clock periods between two successive transactions.

The duration between the slave selection signal active edge and the first SPI bus clock edge is 1 SPI bus clock cycle and the duration between the last SPI bus clock and the slave selection signal inactive edge is 1.5 SPI bus clock cycle.

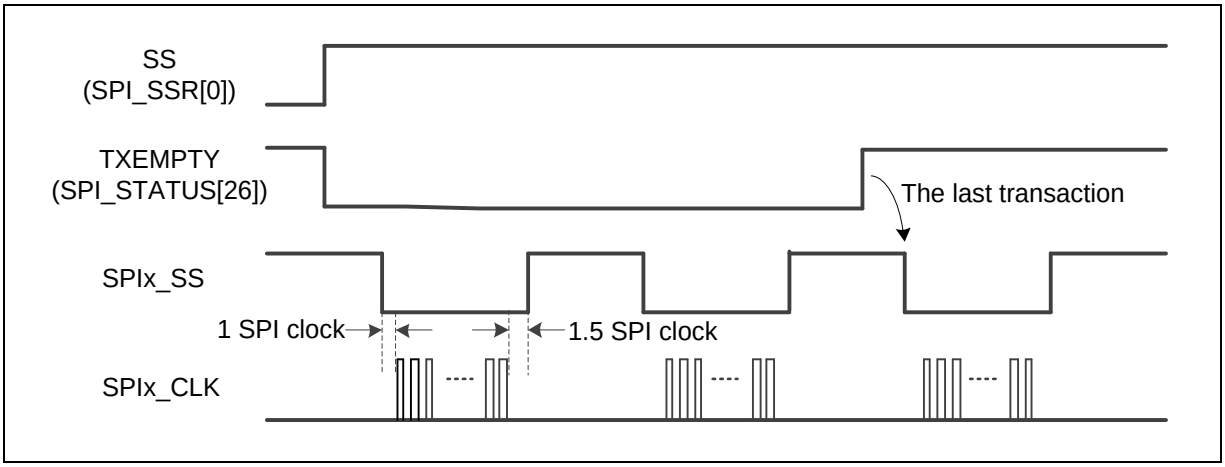


Figure 6-93 Automatic Slave Selection (SS_LVL = 0, SP_CYCLE > 0x2)

6.14.5.3 *Variable Clock Function*

In Master mode, if the VARCLK_EN bit (SPI_CNTRL[23]) is set to 1, the output of serial clock can be programmed as variable frequency pattern. The serial clock period of each cycle depends on the setting of the SPI_VARCLK register. When the variable clock function is enabled, the TX_BIT_LEN setting must be set as 0x10 to configure the data transfer as 16-bit transfer mode. The VARCLK[31] determines the clock period of the first clock cycle. If VARCLK[31] is 0, the first clock cycle depends on the DIVIDER setting; if it is 1, the first clock cycle depends on the DIVIDER2 setting. Two

successive bits in VARCLK[30:1] defines one clock cycle. The bit field VARCLK[30:29] defines the second clock cycle of SPI serial clock of a transaction, and the bit field VARCLK[28:27] defines the third clock cycle and so on. The VARCLK[0] is unmeaning. The Figure 6-94 shows the timing relationship among the SPI bus clock, VARCLK setting, DIVIDER setting and DIVIDER2 setting.

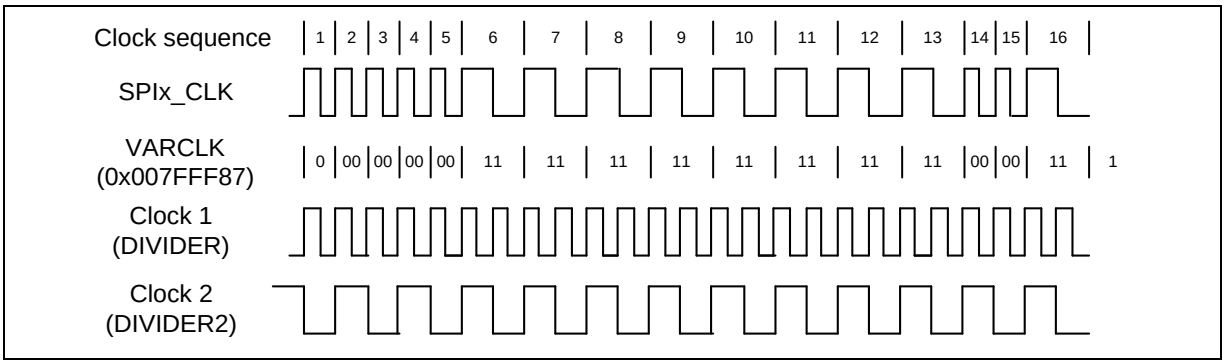


Figure 6-94 Variable Serial Clock Frequency

6.14.5.4 *Word Suspend*

These four bits field of SP_CYCLE (SPI_CNTRL[15:12]) provide a configurable suspend interval, 0.5 ~ 15.5 serial clock periods, between two successive transaction words in Master mode. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SP_CYCLE is 0x3 (3.5 serial clock cycles). This SP_CYCLE setting will not take effect to the word suspend interval if the software disables the FIFO mode.

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

6.14.5.5 *Byte Reorder*

When the transfer is set as MSB first (LSB = 0) and the REORDER (SPI_CNTRL[19]) bit is set to 1, the data stored in the TX buffer and RX buffer will be rearranged in the order as [BYTE0, BYTE1, BYTE2, BYTE3] in 32-bit transfer mode (TX_BIT_LEN = 0). The sequence of transmitted/received data will be BYTE0, BYTE1, BYTE2, and then BYTE3. If the TX_BIT_LEN is set as 24-bit transfer mode, the data in TX buffer and RX buffer will be rearranged as [unknown byte, BYTE0, BYTE1, BYTE2]. The SPI controller will transmit/receive data with the sequence of BYTE0, BYTE1 and then BYTE2. Each byte will be transmitted/received with MSB first. The rule of 16-bit mode is the same as above. Byte reorder function is only available when TX_BIT_LEN is configured as 16, 24, and 32 bits.

Note: the Byte Reorder function is not supported when the variable serial clock function is enabled.

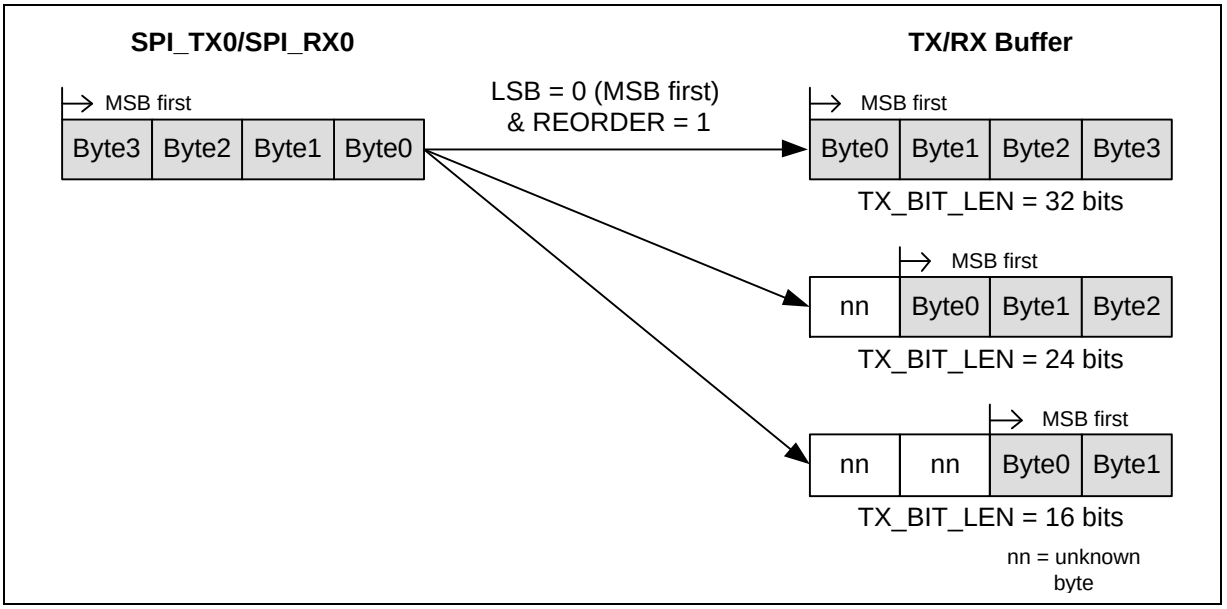


Figure 6-95 Byte Reorder

6.14.5.6 *Byte Suspend*

In Master mode, if REORDER (SPI_CNTRL[19]) is set to 1, the hardware will insert a suspend interval of 0.5 ~ 15.5 serial clock periods between two successive bytes in a transaction word. Both settings of byte suspend interval and word suspend interval are configured in SP_CYCLE (SPI_CNTRL[15:12]).

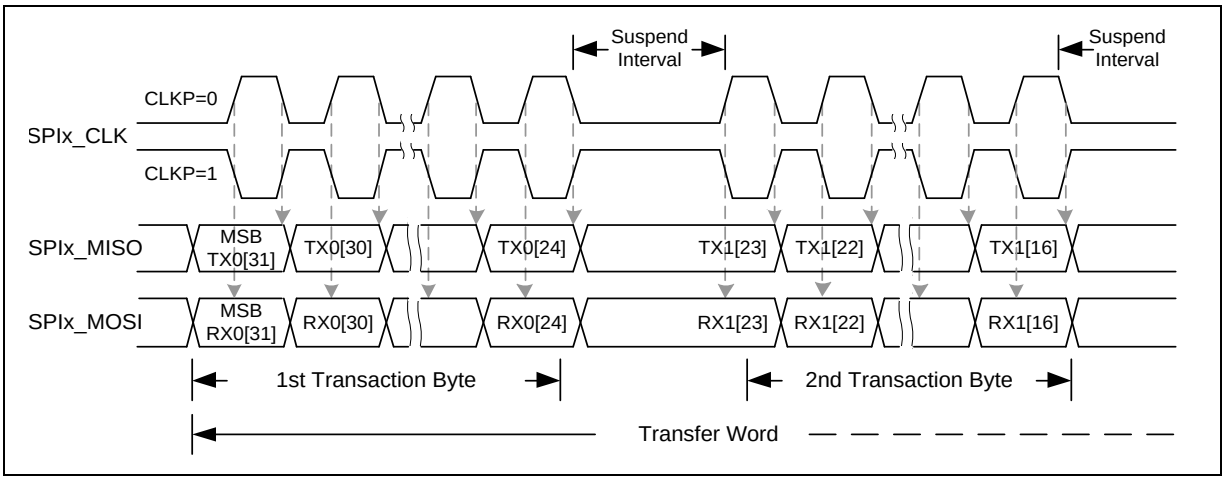


Figure 6-96 Timing Waveform for Byte Suspend

6.14.5.7 *Slave 3-Wire Mode*

When NOSLVSEL (SPI_CNTRL2[8]) is set by software to enable the Slave 3-Wire mode, the SPI controller can work with no slave selection signal in Slave mode. The NOSLVSEL only takes effect in Slave mode. Only three pins, SPIx_CLK, SPIx_MISO, and SPIx_MOSI, are required to communicate with a SPI Master. The SPIx_SS0 pin can be configured as a GPIO. When the NOSLVSEL is set to 1, the SPI Slave will be ready to transmit/receive data after the GO_BUSY (SPI_CNTRL[0]) is set to 1.

If there is no transfer done interrupt over the time period which is defined by user after the transfer start, the user can set the SLV_ABORT (SPI_CNTRL2[9]) to force the transfer done.

In 3-wire mode, the SS_LTRIG (SPI_SSR[4]) shall be set as 1.

6.14.5.8 *PDMA Transfer Function*

When TX_DMA_GO (SPI_DMA[0]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically. Hardware will clear this bit to 0 automatically after PDMA transfer done.

If uses SPI transmit PDMA function to transfer data, the software should not set the GO_BUSY (SPI_CNTRL[0]) bit to 1. The PDMA control logic of SPI controller will set it automatically whenever necessary.

In Slave mode and the FIFO mode is disabled, the minimal suspend interval between two successive transactions must be larger than (8 SPI serial clock periods + 14 APB clock periods) for edge-trigger mode or (9.5 serial clock periods + 14 APB clock periods) for level-trigger mode. If the 2-bit transfer mode is enabled, additional 18 APB clock periods for the above conditions is required.

When RX_DMA_GO (SPI_DMA[1]) is set to 1, the controller will start the PDMA reception process. SPI controller will issue request to PDMA controller automatically when there is data in the RX FIFO buffer. Hardware will clear this bit to 0 automatically after PDMA transfer is done.

If the software uses the receive PDMA function to access the received data of SPI and does not use the transmit PDMA function, the GO_BUSY bit shall be set by software. Enable the FIFO mode is recommended if the software uses more than one PDMA channel to transfer data.

In Slave mode and the FIFO mode is disabled, if the software only uses one PDMA channel for SPI receive PDMA function and the other PDMA channels are not in use, the minimal suspend interval between two successive transactions must be larger than (9 SPI Slave peripheral clock periods + 4 APB clock periods) for edge-trigger mode or (9.5 SPI Slave peripheral clock periods + 4 APB clock periods) for level-trigger mode.

6.14.5.9 *Two-Bit Transfer Mode*

The SPI controller also supports 2-bit transfer mode when set the TWOB (SPI_CNTRL[22]) to 1. In 2-bit transfer mode, the SPI controller performs full duplex data transfer. In other words, it can transmit and receives two-bit serial data simultaneously.

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

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

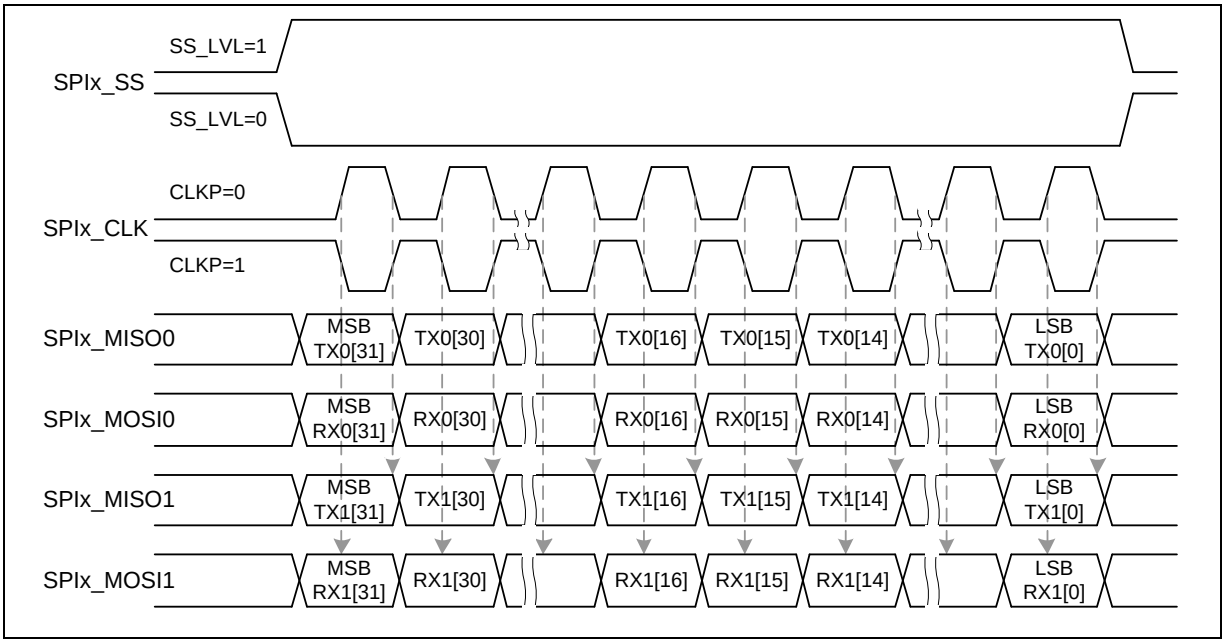


Figure 6-97 Two-Bit Transfer Mode (Slave Mode)

6.14.5.10 *Dual I/O Mode*

The SPI controller also supports Dual I/O transfer when set the DUAL_IO_EN (SPI_CNTRL2[13]) to 1. Many general SPI flashes support Dual I/O transfer. The DUAL_IO_DIR (SPI_CNTRL2[12]) is used to define the direction of the transfer data. When set the DUAL_IO_DIR to 1, the controller will send the data to external device. When the DUAL_IO_DIR is cleared to 0, the controller will read the data from the external device. This function is only available when the transfer bit length is even number.

The Dual I/O mode is not supported when the Slave 3-wire mode or the Byte Reorder function is enabled.

For Dual I/O mode, if both the DUAL_IO_EN (SPI_CNTRL2[13]) and DUAL_IO_DIR (SPI_CNTRL2[12]) are set as 1, the MOSI0 is the even bit data output and the MISO0 will be set as the odd bit data output. If the DUAL_IO_EN (SPI_CNTRL2[13]) is set as 1 and DUAL_IO_DIR (SPI_CNTRL2[12]) is set as 0, both the MISO0 and MOSI0 will be set as data input ports.

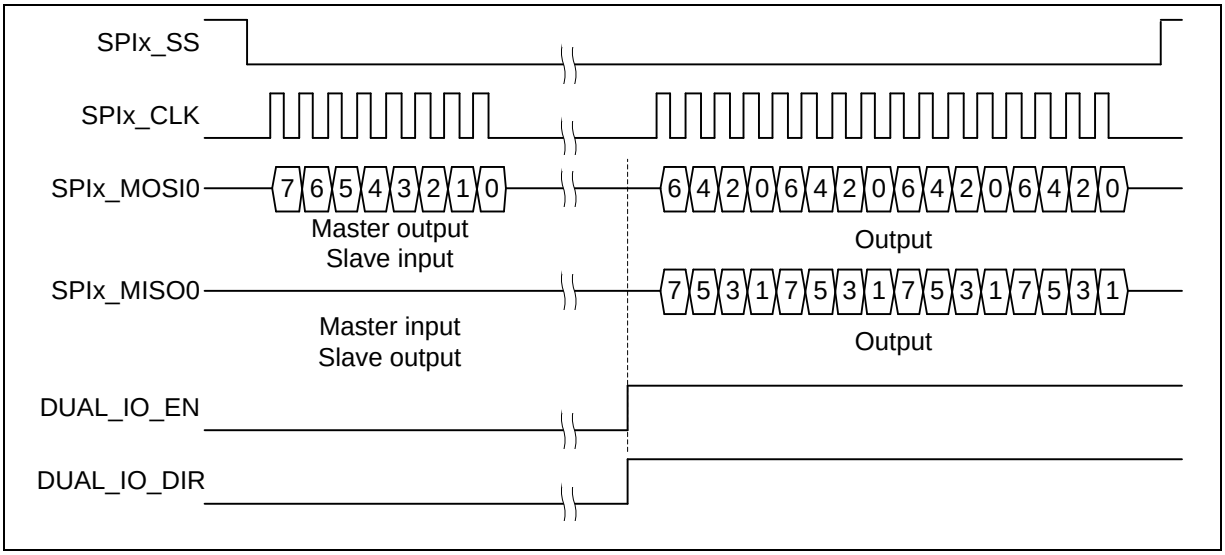


Figure 6-98 Bit Sequence of Dual Output Mode

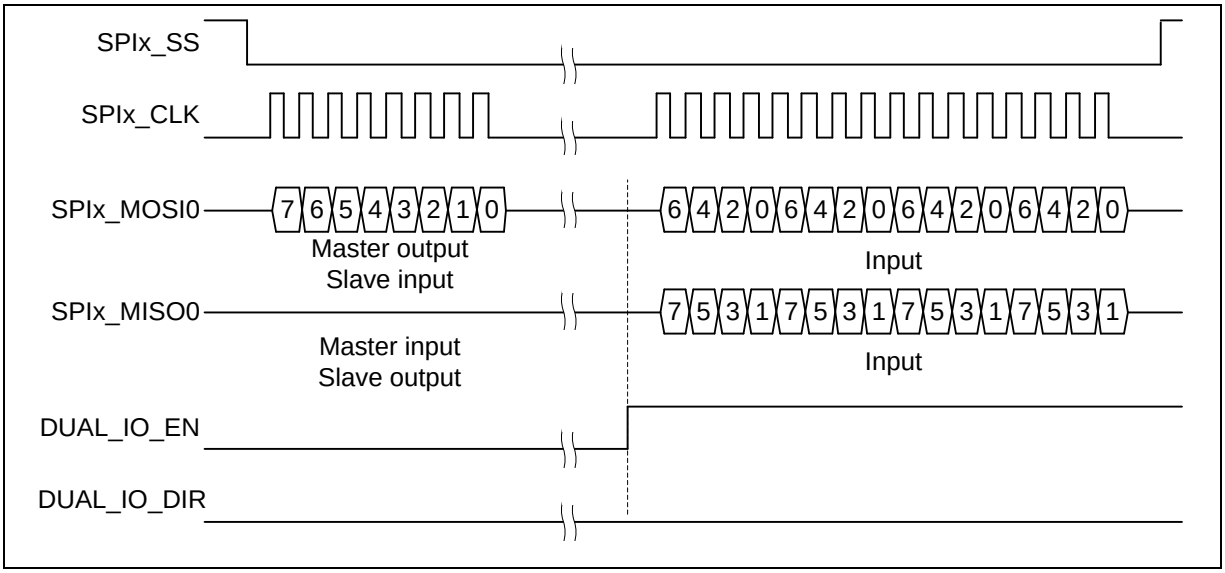


Figure 6-99 Bit Sequence of Dual Input Mode

6.14.5.11 *FIFO Mode*

The SPI controller supports FIFO mode when FIFO (SPI_CNTRL[21]) is set to 1. The SPI controllers equip with eight 32-bit wide transmit and receive FIFO buffers.

The transmit FIFO buffer is an 8-layer depth, 32-bit wide, first-in, first-out register buffer. 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 8-layer transmit FIFO buffer is full, the TX_FULL bit will be set to 1. When the SPI transmission logic unit draws out the last datum of the transmit FIFO buffer, so that the 8-layer transmit FIFO buffer is empty, the TX_EMPTY bit will be set to 1. Notice that the TX_EMPTY flag is set to 1 while the last transaction is still in progress. In Master mode, the software should check both the GO_BUSY bit and TX_EMPTY bit to make sure whether the SPI is in idle or not.

The received FIFO buffer is also an 8-layer depth, 32-bit wide, first-in, first-out register buffer. The

receive control logic will store the received data to this buffer. The 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.

The transmitting and receiving threshold can be configured by setting the TX_THRESHOLD (SPI_FIFO_CTL[30:28]) and RX_THRESHOLD (SPI_FIFO_CTL[26:24]). 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.

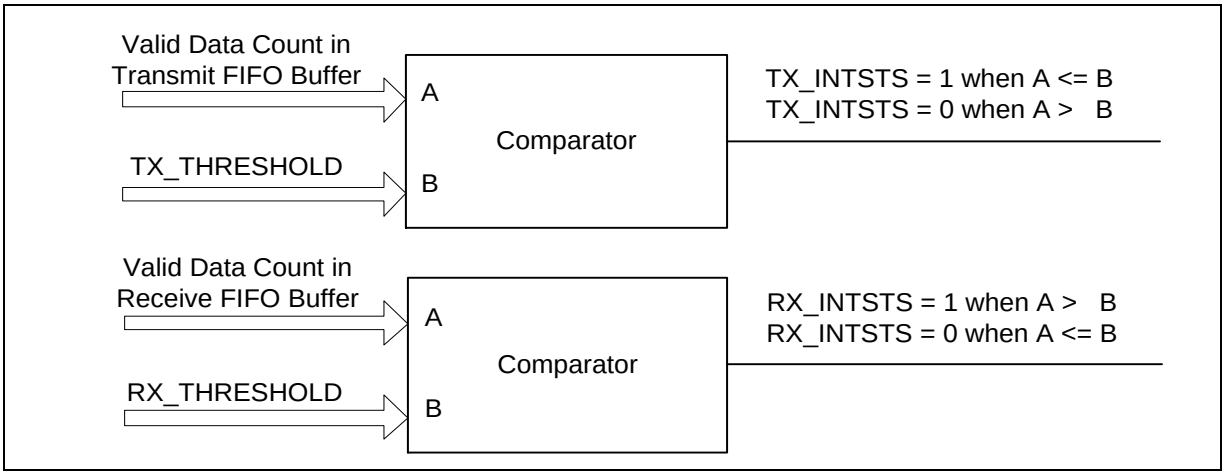


Figure 6-100 FIFO Threshold Comparator

In FIFO mode, the software can write 8 data to the SPI transmit FIFO buffer in advance. When the SPI controller operates with FIFO mode, GO_BUSY (SPI_CNTRL[0]) 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.

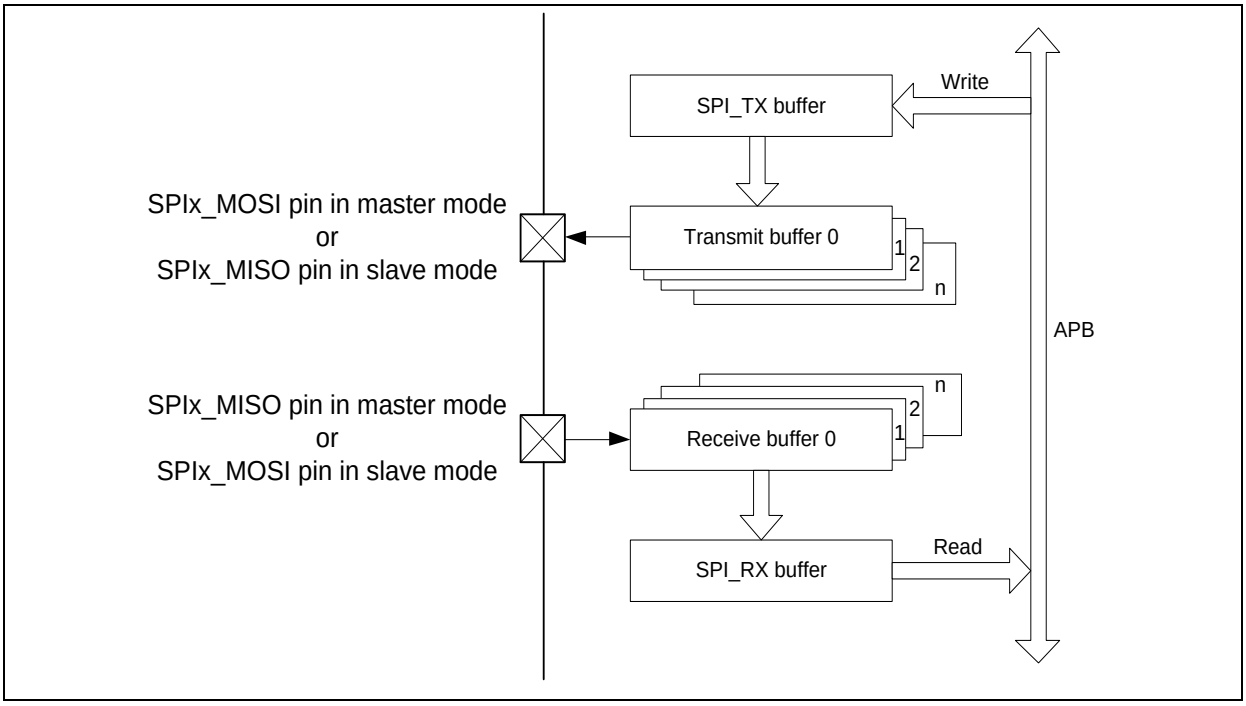


Figure 6-101 FIFO mode Block Diagram

In Master mode, the first datum is written to the SPI_TX register, the TX_EMPTY flag (SPI_STATUS[16]) will be cleared to 0. The transmission will start after 1 APB clock cycle and 6 peripheral clock cycles. User can write the next data into SPI_TX register immediately. The SPI controller will insert a suspend interval between two successive transactions. The period of suspend interval is decided by the setting of SP_CYCLE (SPI_CNRTL[15:12]). If the SP_CYCLE (SPI_CNTRL[15:12]) equals 0, SPI controller can perform continuous transfer. User can write data into SPI_TX register as long as the TX_FULL (SPI_STATUS[27]) is 0.

In the Example 1 of the Figure 6-102, it indicates the updated condition of TX_EMPTY (SPI_STATUS[26]) and the relationship among the FIFO buffer, shift register and the skew buffer. The TX_EMPTY (SPI_STATUS[16]) is set to 0 when the Data 0 is written into the FIFO buffer. The Data 0 will be loaded into the shift register by core logical and the TX_EMPTY (SPI_STATUS[26]) will be to 1. The Data 0 in shift register will be shifted into skew buffer by bit for transmission until the transfer is done.

In the Example 2, it indicates the updated condition of TX_FULL (SPI_STATUS[27]) when there are 8 data in the FIFO buffer and the next data of Data 9 is not written into the FIFO buffer when the TX_FULL = 1.

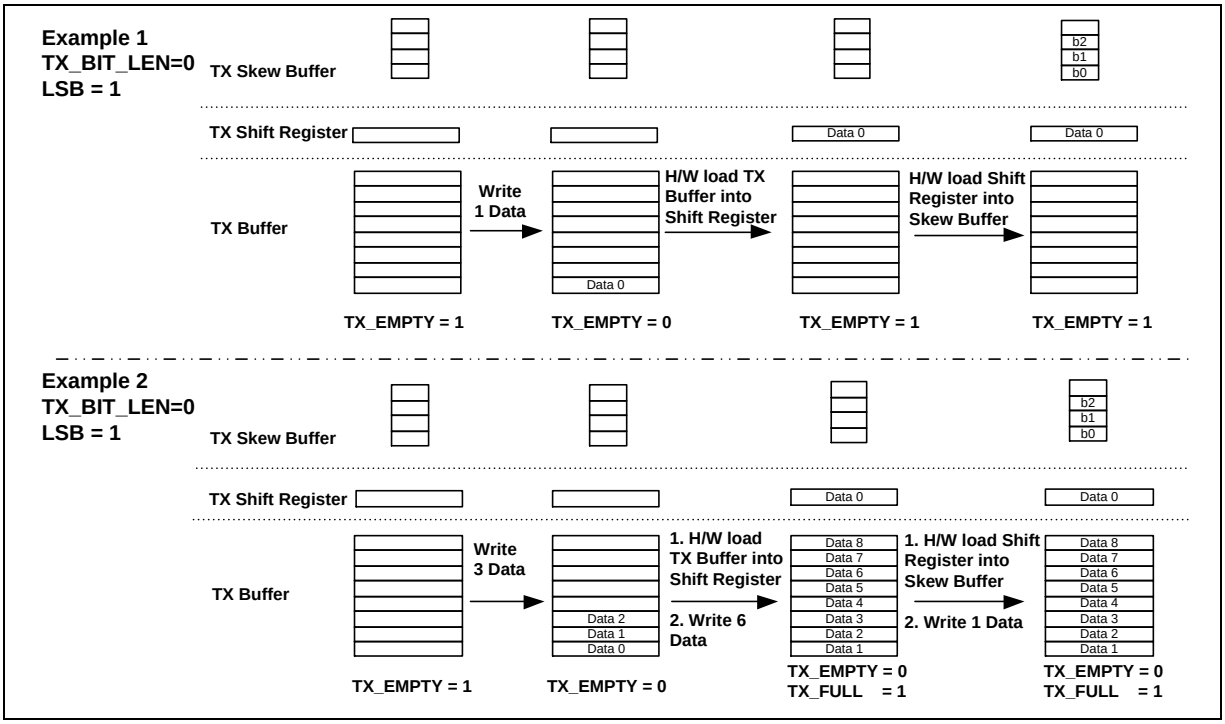


Figure 6-102 Transmit FIFO Buffer Example

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the SPI_TX0 register does not be updated after all data transfer are done, the transfer will stop.

In Master mode, during reception operation, the serial data are received from SPIx_MISO pin and stored to receive FIFO buffer.

The receive data (Data 0's b0, b1… b31) is stored into skew buffer first according the serial clock (SPICLKx) and then is shifted into the shift register bit by bit. The core logic will load the data in shift register into FIFO buffer when the receive data bit reach the value of TX_BIT_LEN (SPI_CNTRL[7:3]). The RX_EMPTY (SPI_STATUS[24]) will be cleared to 0 while the receive FIFO buffer contains unread data (see the Example 1 of Receive FIFO Buffer Example). The received data can be read by software from SPI_RX0 register as long as the RX_EMPTY (SPI_STATUS[24]) is 0. If the receive FIFO buffer contains 8 unread data, the RX_FULL (SPI_STATUS[25]) will be set to 1 (see the

Example 2 of Receive FIFO Buffer Example).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.

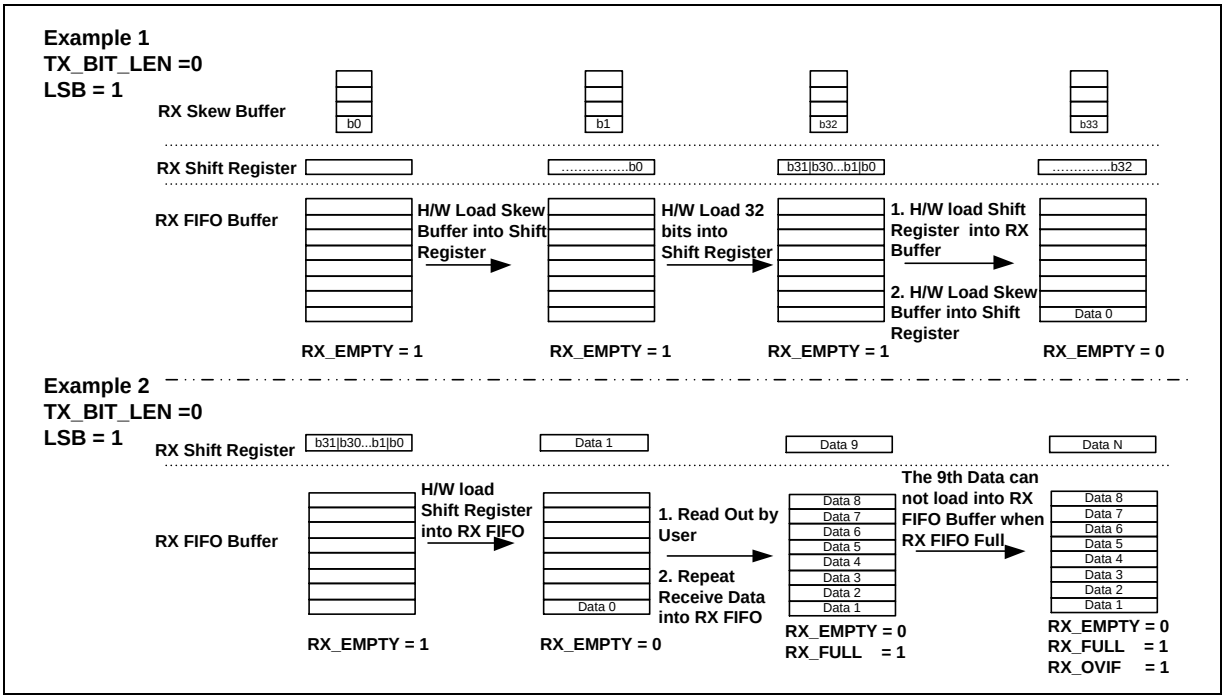


Figure 6-103 Receive FIFO Buffer Example

In Slave mode, during 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.

In 2-bit Transfer mode, the transmit data is loaded into shift register after 2 datum have been written into the TX FIFO buffer. It uses 2 shift registers and 2 4-level skew buffers concurrently. For the detailed timing of 2-bit Transfer mode, please refer to the section of 2-bit Transfer mode.

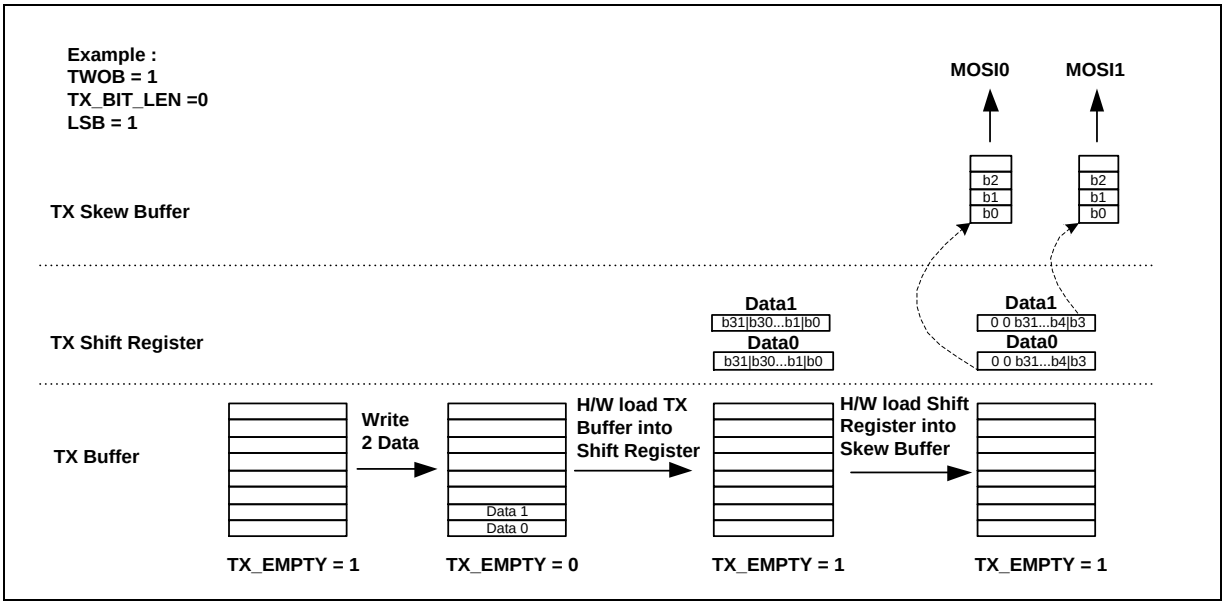


Figure 6-104 Two-Bit Transfer Mode FIFO Buffer Example

In Slave mode, during receiving operation, the serial data is received from SPIx_MOSI0/1 pin and stored to SPI_RX0 register. The reception mechanism is similar to Master mode reception operation. If the receive FIFO buffer contains 8 unread data, the RX_FULL (SPI_STATUS[25]) will be set to 1 and the RX_OVERRUN (SPI_STATUS[2]) will be set 1 if there is more serial data is received from SPIx_MOSI and follow-up data will be dropped (refer to the Receive FIFO Buffer Example figure). If the receive bit count mismatch with the TX_BIT_LEN (SPI_CNTRL[7:3]) when the slave selection line goes to inactive state, the LTRIG_FLAG (SPI_SSR[5]) will be set to 0.

A receive time-out function is built-in in this controller. When the receive FIFO is not empty and no read operation in receive FIFO over 64 SPI clock period in Master mode or over 576 SPI peripheral clock period in Slave mode, the receive time-out occurs and the TIMEOUT (SPI_STATUS[20]) be set to 1. When the receive FIFO is read by user, the time-out status will be cleared automatically.

6.14.5.12 *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 clock signal. The SPI controller will issue an interrupt if the SSTA_INTEN (SPI_CNTRL2[10]) is set to 1. If the count of the received bits is less than the setting of TX_BIT_LEN (SPI_CNTRL[7:3]) and there is no more serial clock input over the expected time period which is defined by the user, the user can set the SLV_ABORT (SPI_CNTRL2[9]) bit to abort the current transfer. The unit transfer interrupt flag, IF, will be set to 1 if the software set the SLV_ABORT bit.

Receive FIFO time-out interrupt

In FIFO mode, there is 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 peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode, it will send a time-out interrupt to the system if the time-out interrupt enable bit, TIMEOUT_INTEN (SPI_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 (SPI_FIFO_CTL[30:28]), 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, TX_INTEN (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 (SPI_FIFO_CTL[26:24]), 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, RX_INTEN (SPI_FIFO_CTL[2]), is set to 1.

6.14.6 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 serial clock (SPICLK) idle state can be configured as high state or low state by setting the CLKP bit (SPI_CNTRL[11]). It also provides the bit length of a transaction word in TX_BIT_LEN (SPI_CNTRL[7:3]), and transmit/receive data from MSB or LSB first in LSB bit (SPI_CNTRL[10]). User can also select which edge of serial clock to transmit/receive data in TX_NEG/RX_NEG (SPI_CNTRL[2:1]). Four SPI timing diagrams for Master/Slave operations and the related settings are shown below.

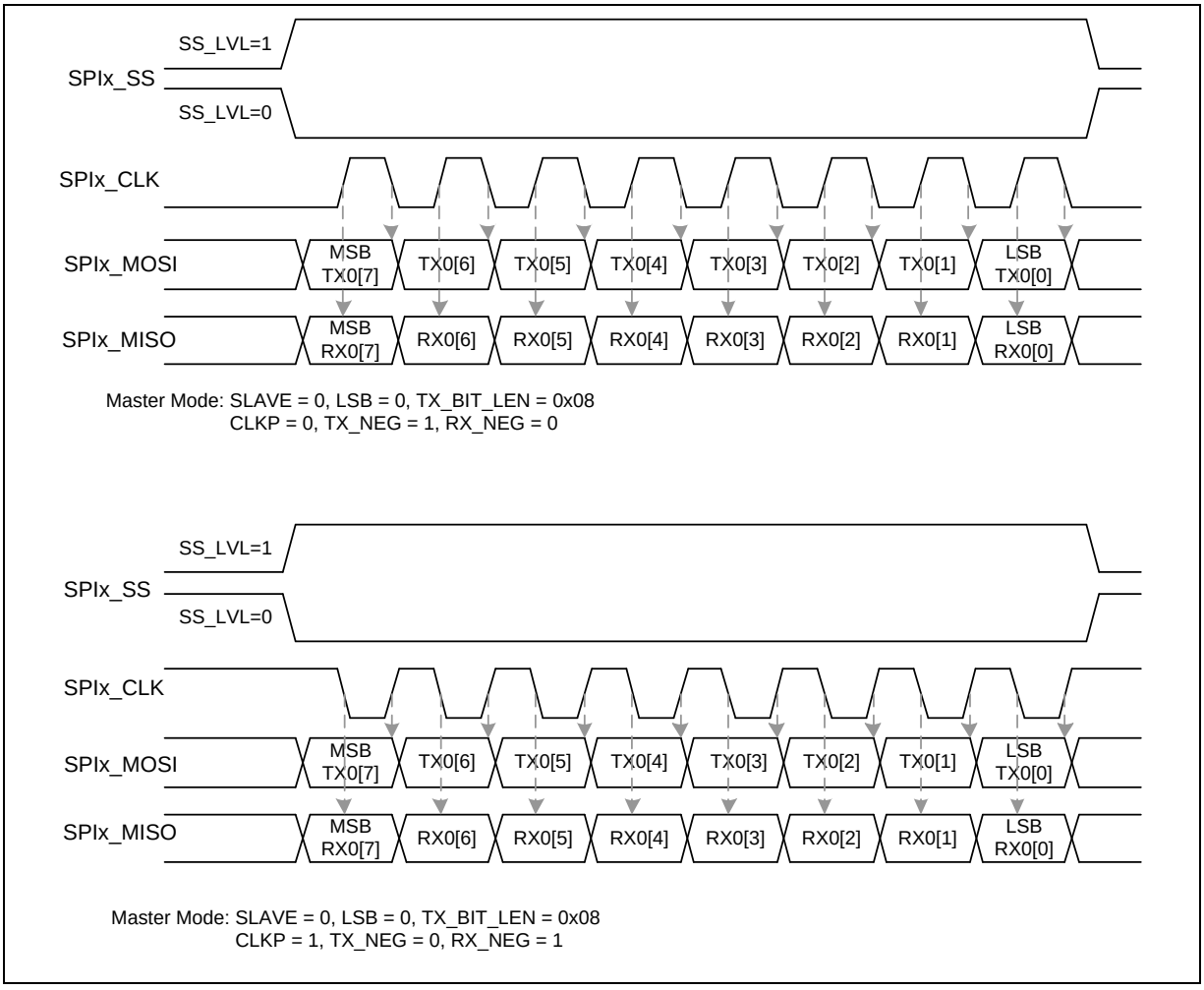


Figure 6-105 SPI Timing in Master Mode

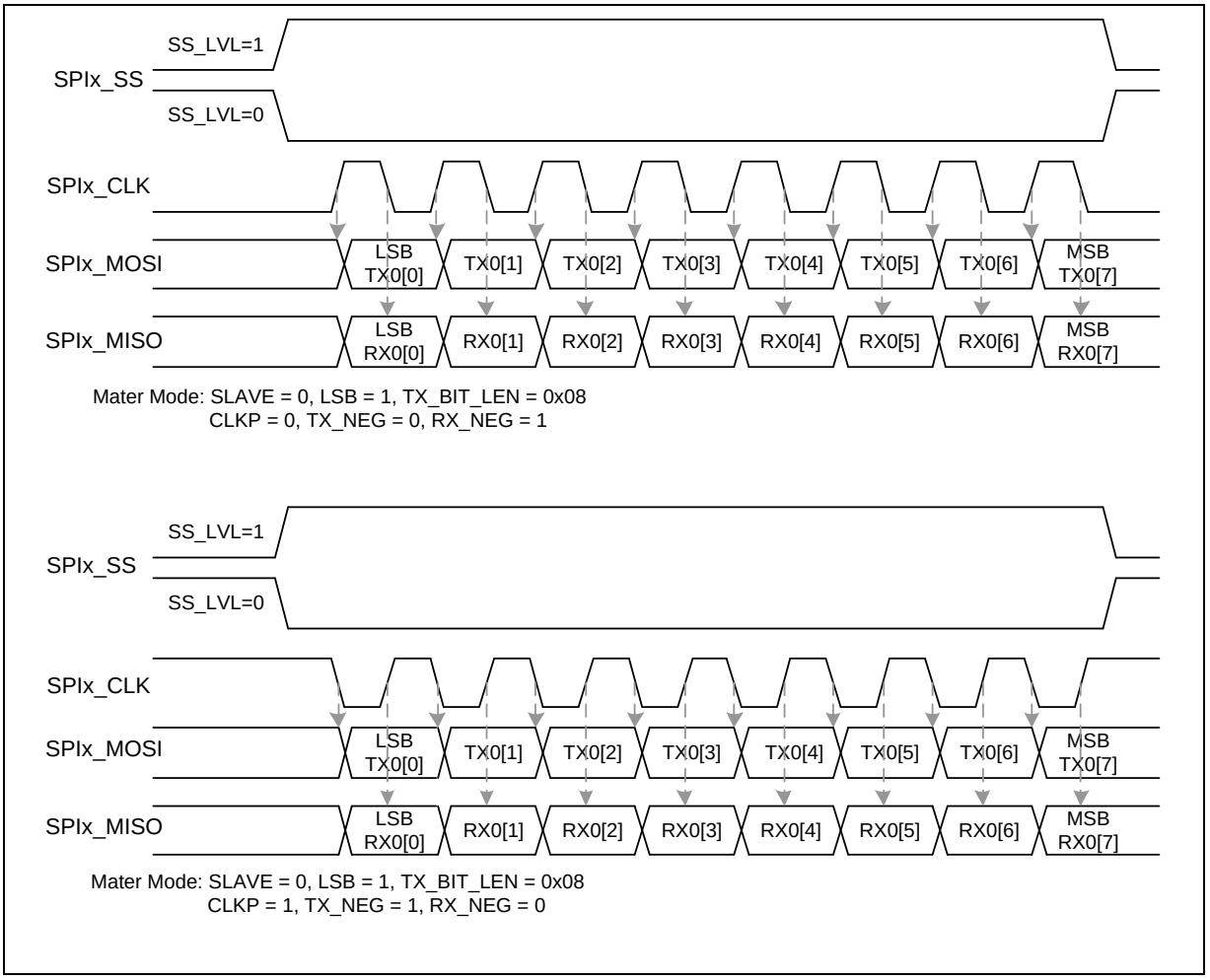


Figure 6-106 SPI Timing in Master Mode (Alternate Phase of SPI bus clock)

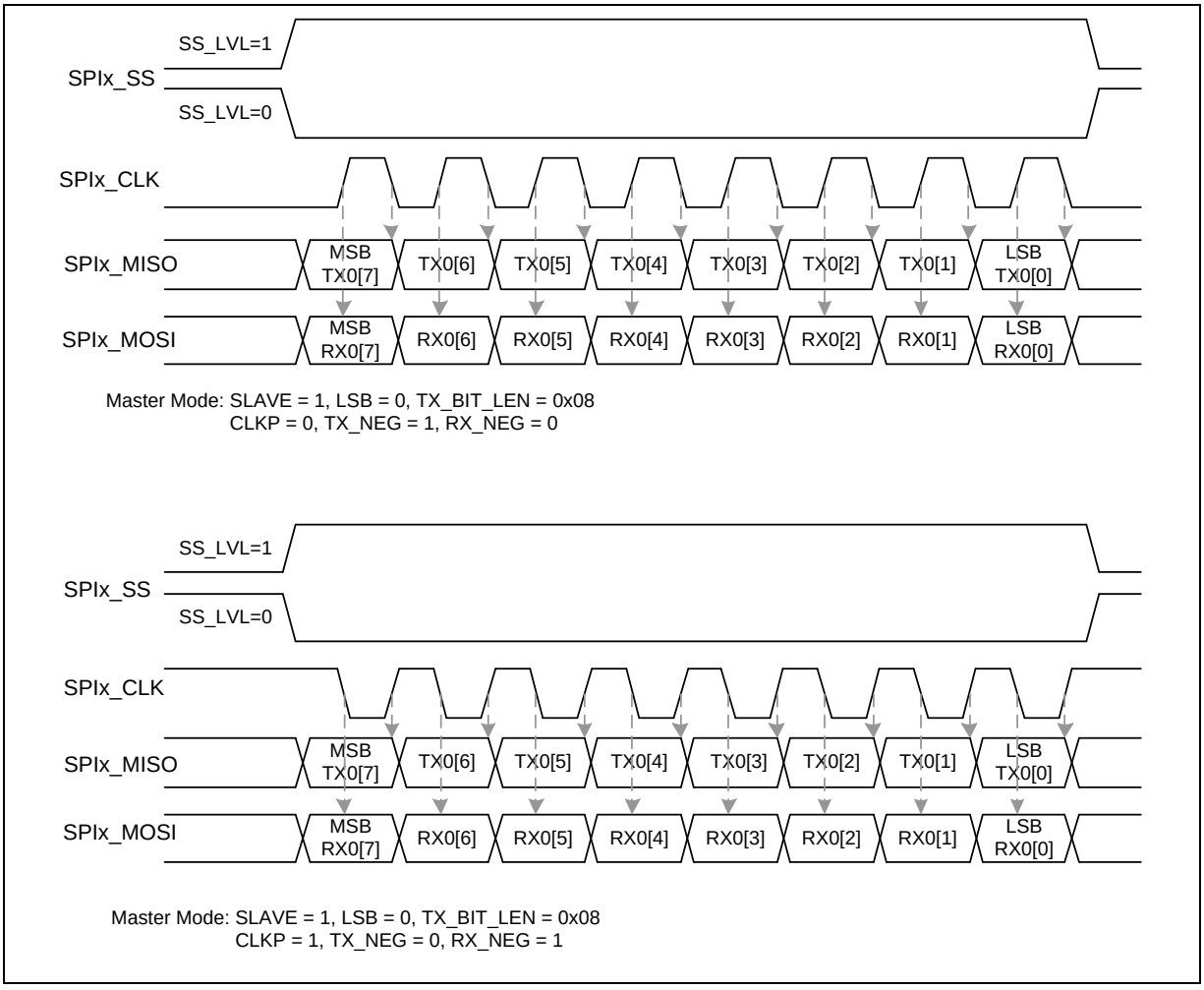


Figure 6-107 SPI Timing in Slave Mode

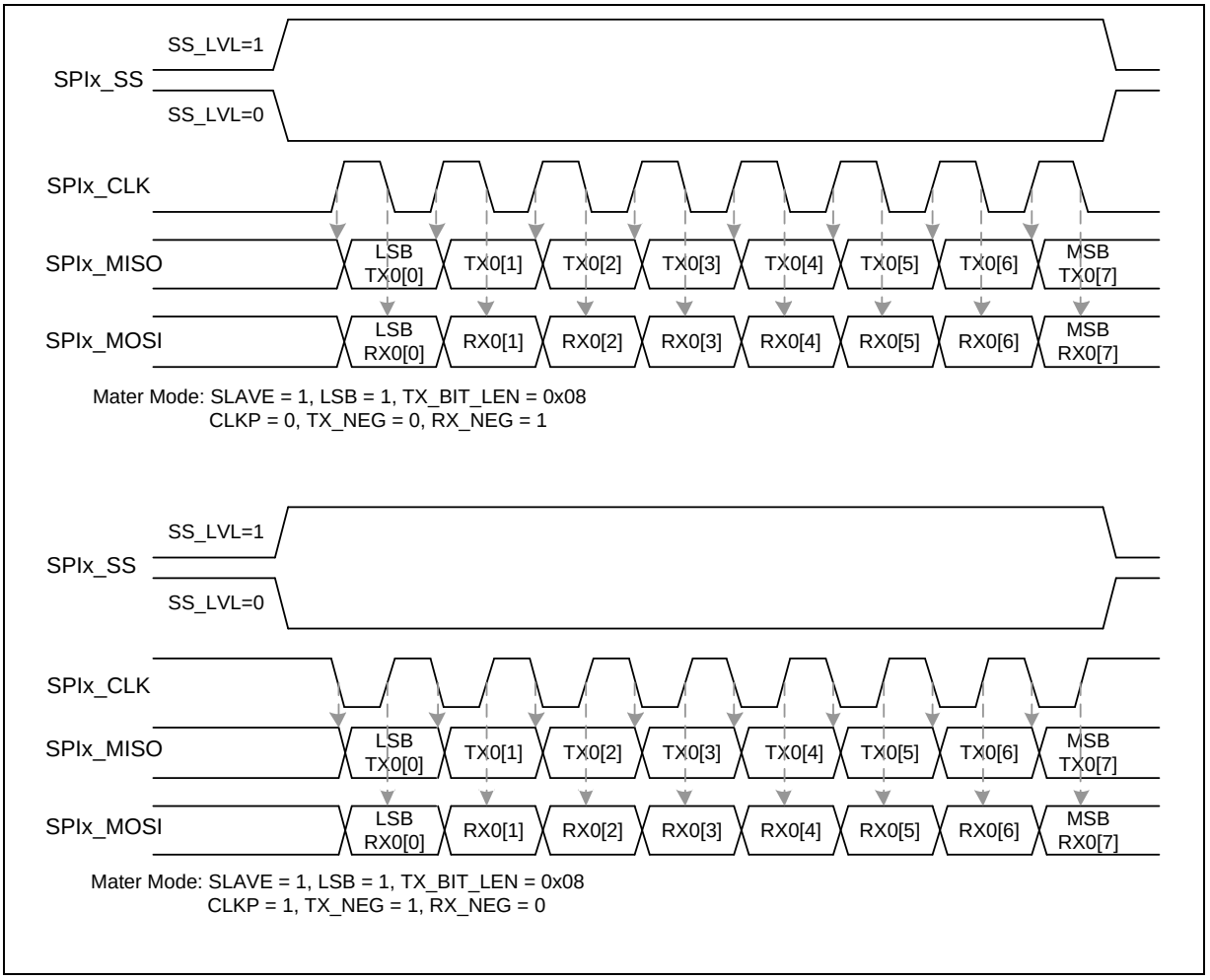


Figure 6-108 SPI Timing in Slave Mode (Alternate Phase of SPI bus clock)

6.14.7 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 SPI bus clock
- Data bit is driven on negative edge of SPI bus clock
- Data is transferred from MSB first
- SPI bus clock idle state is low-level
- Only one byte will be transmitted/received in a transaction
- Use the first SPI slave select pin to connect with an off-chip Slave device. Slave select signal is active low

The operation flow is as follows.

1. Set the DIVIDER (SPI_DIVIDER [7:0]) register to determine the output frequency of serial clock.
2. Write the related settings into the SPI_CNTRL register to control this SPI Master actions

(1). Set this SPI controller as Master device. SLAVE (SPI_CNTRL[18]) = 0.

(2). Set the SPI bus clock idle state as low-level. CLKP (SPI_CNTRL[11]) = 0.

(3). Transmit data on falling edge of SPI bus clock. TX_NEG (SPI_CNTRL[2]) = 1.

(4). Capture data on rising edge of SPI bus clock. RX_NEG (SPI_CNTRL[1]) = 0.

(5). Set the bit length of a transaction as 8 bits. TX_BIT_LEN (SPI_CNTRL[7:3]) = 0x08.

(6). Set transfer sequence as MSB first. LSB (SPI_CNTRL[10]) = 0.

3. Write the SPI_SSR register a proper value for the related settings of Master mode:

   (1). Disable the automatic slave selection function. AUTOSS (SPI_SSR[3]) = 0.

   (2). Select low level trigger output of slave select signal. Set SS_LVL (SPI_SSR[2]) = 0 and SS_LTRIG (SPI_SSR[4]) = 1.

   (3). Set SSR (SPI_SSR[0]) to 1 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, software does not need to update the SPI_TX0 register.

6. Set the GO_BUSY (SPI_CNTRL[0]) to 1 to start the data transfer with the SPI interface.

7. Waiting for SPI interrupt if the interrupt function is enabled 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 SSR to 0 to inactivate the off-chip Slave device.

Example 2, The SPI controller is set as a Slave device and connects with an off-chip Master device. The off-chip Master device communicates with the on-chip SPI Slave controller through the SPI interface with the following specifications:

- Data bit is latched on positive edge of SPI bus clock
- Data bit is driven on negative edge of SPI bus clock
- Data is transferred from LSB first
- SPI bus clock idle state is high-level
- Only one byte will be transmitted/received in a transaction
- Slave select signal is high level active

The operation flow is as follows.

1. Write the SPI_SSR register a proper value for the related settings of Slave mode. Select high level and level trigger for the input of slave select signal by setting SS_LVL (SPI_SSR[2]) to 1 and setting SS_LTRIG (SPI_SSR[4]) to 1.

2. Write the related settings in the SPI_CNTRL register to control this SPI Slave actions

   (1). Set this SPI controller as Slave device. SLAVE (SPI_CNTRL[18]) = 1.

   (2). Select high-level as the SPI bus clock idle state. CLKP (SPI_CNTRL[11]) = 1.

   (3). Transmit data on falling edge of SPI bus clock. TX_NEG (SPI_CNTRL[2]) = 1.

(4). Capture data on rising edge of SPI bus clock. RX_NEG (SPI_CNTRL[1]) = 0.

(5). Set the bit length of a transaction as 8 bits. TX_BIT_LEN (SPI_CNTRL[7:3]) = 0x08.

(6). Set transfer sequence as LSB first. LSB (SPI_CNTRL[10]) = 1.

3. If this SPI Slave attempts to transmit (be read) one byte data to the off-chip Master device, write the byte data that will be transmitted into the SPI_TX0 register
4. If this SPI Slave just only attempts to receive (be written) one byte data from the off-chip Master device and does not care what data will be transmitted, software does not need to update the SPI_TX0 register.
5. Set GO_BUSY (SPI_CNTRL[0]) to 1 to wait for the slave select trigger input and serial clock input from the off-chip Master device to start the data transfer with the SPI interface.
6. Waiting for SPI interrupt if the interrupt function is enabled, or just polling the GO_BUSY bit till it is cleared to 0 by hardware automatically.
7. Read out the received one byte data from SPI_RX0[7:0].
8. Go to 3. to continue another data transfer or clear GO_BUSY bit to stop data transfer.

6.14.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI0_BA = 0x4003_0000 **SPI1_BA = 0x4003_4000** **SPI2_BA = 0x4013_0000**				
SPI_CNTRL	SPIx_BA+0x00	R/W	Control and Status Register	0x0500_3004
SPI_DIVIDER	SPIx_BA+0x04	R/W	Clock Divider Register	0x0000_0000
SPI_SSR	SPIx_BA+0x08	R/W	Slave Select Register	0x0000_0000
SPI_RX0	SPIx_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPI_RX1	SPIx_BA+0x14	R	Data Receive Register 1	0x0000_0000
SPI_TX0	SPIx_BA+0x20	W	Data Transmit Register 0	0x0000_0000
SPI_TX1	SPIx_BA+0x24	W	Data Transmit Register 1	0x0000_0000
SPI_VARCLK	SPIx_BA+0x34	R/W	Variable Clock Pattern Register	0x007F_FF87
SPI_DMA	SPIx_BA+0x38	R/W	SPI DMA Control Register	0x0000_0000
SPI_CNTRL2	SPIx_BA+0x3C	R/W	Control and Status Register 2	0x0000_1000
SPI_FIFO_CTL	SPIx_BA+0x40	R/W	FIFO Control Register	0x4400_0000
SPI_STATUS	SPIx_BA+0x44	R/W	SPI Status Register	0x0500_0000

6.14.9 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_3004

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**TX_FULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** A mutual mirror bit of SPI_STATUS[27]. 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.
[26]	**TX_EMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** A mutual mirror bit of SPI_STAUTS[26]. 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[25]	**RX_FULL**	**Receive FIFO Buffer Full Indicator (Read Only)** A mutual mirror bit of SPI_STATUS[25]. 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[24]	**RX_EMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** A mutual mirror bit of SPI_STATUS[24]. 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[23]	**VARCLK_EN**	**Variable Clock Enable Bit (Master Only)** 0 = Serial clock output frequency is fixed and decided only by the value of DIVIDER. 1 = Serial clock output frequency is variable. The output frequency is decided by the value of VARCLK, DIVIDER1, and DIVIDER2. **Note:** When this VARCLK_EN bit is set to 1, the setting of TX_BIT_LEN must be programmed as 0x10 (16-bit mode)
[22]	**TWOB**	**Two-bit Transfer Mode Enable Bit** 0 = Two-bit transfer mode Disabled. 1 = Two-bit transfer mode Enabled.

		Note: When TWOB is enabled, the serial transmitted 2-bit data are from SPI_TX1/0, and the received 2-bit data input are put in SPI_RX1/0.
[21]	**FIFO**	**FIFO Mode Enable Bit** 0 = FIFO mode Disabled. 1 = FIFO mode Enabled. **Notes:** 1. Before enabling FIFO mode, the other related settings should be set in advance. 2. In Master mode, if the FIFO mode is enabled, the GO_BUSY bit will be set to 1 automatically after writing data to the transmit FIFO buffer; the GO_BUSY bit will be cleared to 0 automatically when the SPI controller is in idle. If all data stored at transmit FIFO buffer are sent out, the TX_EMPTY bit will be set to 1 and the GO_BUSY bit will be cleared to 0.
[19]	**REORDER**	**Byte Reorder Function Enable Bit** 0 = Byte reorder functions Disabled. 1 = Byte reorder function Enabled. A byte suspend interval will be inserted among each byte. The period of the byte suspend interval depends on the setting of SP_CYCLE. **Notes:** 1. Byte reorder function is only available if TX_BIT_LEN is defined as 16, 24, and 32 bits. 2. In Slave mode with level-trigger configuration, the slave select pin must be kept at active state during the byte suspend interval. 3. The byte reorder function is not supported when the variable serial clock function or the dual I/O mode is enabled.
[18]	**SLAVE**	**Slave Mode Enable Bit** 0 = Master mode. 1 = Slave mode.
[17]	**IE**	**Transfer Interrupt Enable Bit** 0 = SPI transfer done Interrupt Disabled. 1 = SPI transfer done Interrupt Enabled.
[16]	**IF**	**Transfer Done 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 transmit/receive transaction in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation: (SP_CYCLE[3:0] + 0.5) * period of SPI bus clock cycle Example: SP_CYCLE = 0x0 … 0.5 SPI bus clock cycle. SP_CYCLE = 0x1 … 1.5 SPI bus clock cycle. …… SP_CYCLE = 0xE … 14.5 SPI bus clock cycle. SP_CYCLE = 0xF … 15.5 SPI bus clock cycle. If the variable clock function is enabled and the transmit FIFO buffer is not empty, the minimum period of suspend interval between the successive transactions is (6.5 + SP_CYCLE) * SPI bus clock cycle.
[11]	**CLKP**	**Clock Polarity** 0 = SPI bus clock idle low.

		1 = SPI bus clock idle high.
[10]	**LSB**	**LSB First** 0 = The MSB, which bit of transmit/receive register depends on the setting of TX_BIT_LEN, is transmitted/received first (which bit in SPI_TX0/1 and SPI_RX0/1 register depending on the TX_BIT_LEN field). 1 = The LSB, bit 0 of the SPI TX0/1 register is sent first to the SPI data output pin, and the first bit received from the SPI data input pin will be put in the LSB position of the RX register (bit 0 of SPI_RX0/1).
[9:8]	**Reserved**	Reserved.
[7:3]	**TX_BIT_LEN**	**Transmit Bit Length** This field specifies how many bits can be transmitted / received in one transaction. The minimum bit length is 8 bits and can up to 32 bits. TX_BIT_LEN = 0x08 … 8 bit. TX_BIT_LEN = 0x09 … 9 bits. …… TX_BIT_LEN = 0x1F … 31 bits. TX_BIT_LEN = 0x00 … 32 bits. TX_BIT_LEN = 0x01~0x07 … reserved.
[2]	**TX_NEG**	**Transmit on Negative Edge** 0 = Transmitted data output signal is changed on the rising edge of SPI bus clock. 1 = Transmitted data output signal is changed on the falling edge of SPI bus clock.
[1]	**RX_NEG**	**Receive on Negative Edge** 0 = Received data input signal is latched on the rising edge of SPI bus clock. 1 = Received data input signal is latched on the falling edge of SPI bus clock.
[0]	**GO_BUSY**	**SPI Transfer Control Bit and Busy Status** If the FIFO mode is disabled, during the data transfer, this bit keeps the value of 1. As the transfer is finished, this bit will be cleared automatically. Software can read this bit to check if the SPI is in busy status. In FIFO mode, this bit will be controlled by hardware. Software should not modify this bit. In Slave mode, this bit always returns 1 when software reads this register. In Master mode, this bit reflects the busy or idle status of SPI. 0 = Data transfer stopped 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. **Notes:** 1. When FIFO mode is disabled, all configurations should be set before writing 1 to this GO_BUSY bit. 2. When FIFO mode is disabled and the software uses TX or RX PDMA function to transfer data, this bit will be cleared after the PDMA block finishes the data transfer.

SPI Divider Register (SPI_DIVIDER)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**DIVIDER2**	**Clock Divider 2 (Master Only)** The value in this field is the 2nd frequency divider for generating the second clock of the variable clock function. The frequency is obtained according to the following equation: $f_{sclk} = \frac{f_{spi\_eclk}}{(DIVIDER2+1)*2}$ If the VARCLK_EN bit is cleared to 0, this setting is unmeaning.
[15:8]	**Reserved**	Reserved.
[7:0]	**DIVIDER**	**Clock Divider 1** The value in this field is the frequency divider for generating the SPI peripheral clock, fspi_eclk, and the SPI serial clock of SPI Master. The frequency is obtained according to the following equation: If BCn(SPI_CNTR2[31]) = 0. $f_{spi\_eclk} = \frac{f_{spi\_clk\_src}}{(DIVIDER+1)*2}$ else if BCn =1,. $f_{spi\_eclk} = \frac{f_{spi\_clock\_src}}{(DIVIDER+1)}$ where $f_{spi\_clock\_src}$ is the SPI peripheral clock source. It is defined in the CLK_SEL1 register.

SPI Slave Select Register (SPI_SSR)

Register	Offset	R/W	Description	Reset Value
SPI_SSR	SPIx_BA+0x08	R/W	Slave Select Register	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**LTRIG_FLAG**	**Level Trigger Accomplish Flag (Read Only)** In Slave mode, this bit indicates whether the received bit number meets the requirement or not after the current transaction done. 0 = The transferred bit length of one transaction does not meet the specified requirement. 1 = The transferred bit length meets the specified requirement which defined in TX_BIT_LEN. **Note:** This bit is READ only. As the software sets the GO_BUSY bit to 1, the LTRIG_FLAG will be cleared to 0 after 4 SPI peripheral clock periods plus 1 system clock period. In FIFO mode, this bit is unmeaning.
[4]	**SS_LTRIG**	**Slave Select Level Trigger Enable Bit (Slave Only)** 0 = Input slave select signal is edge-trigger. This is the default value. It depends on the SS_LVL bit to decide the signal is active at falling-edge or rising-edge. 1 = Slave select signal will be level-trigger, which depends on the SS_LVL bit to decide the signal is active low or active high.
[3]	**AUTOSS**	**Automatic Slave Selection Function Enable Bit (Master Only)** 0 = Automatic slave selection function Disabled. Slave selection signal will be asserted/de-asserted according to SS (SPI_SSR[0]). 1 = Automatic slave selection function Enabled.
[2]	**SS_LVL**	**Slave Select Active Level** This bit defines the active polarity of slave select signal (SPIx_SS0/1). 0 = The slave select signal SPIx_SS0/1 is active on low-level/falling-edge. 1 = The slave select signal SPIx_SS0/1 is active on high-level/rising-edge.
[1:0]	**SSR**	**Slave Select Control Bits (Master Only)** If AUTOSS bit is cleared to 0, 0 = set the SPIx_SS0/1 line to inactive state. 1 = set the SPIx_SS0/1 line to active state. If the AUTOSS bit is set to 1, 0 = Keep the SPIx_SS0/1 line at inactive state.

		1 = SPIx_SS0/1 line will be automatically driven to active state for the duration of data transfer, and will be driven to inactive state for the rest of the time. The active state of SPIx_SS is specified in SS_LVL (SPI_SSR[2]).

SPI Data Receive Register (SPI_RX)

Register	Offset	R/W	Description	Reset Value
SPI_RX0	SPIx_BA+0x10	R	Data Receive Register 0	0x0000_0000
SPI_RX1	SPIx_BA+0x14	R	Data Receive Register 1	0x0000_0000

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

Bits	Description	
[31:0]	**RX**	**Data Receive Register** The data receive register holds the datum received from SPI data input pin. If the FIFO mode is disabled, the software can access the last received data by reading this register. If the FIFO bit is set as 1 and the RX_EMPTY bit, SPI_CNTRL[24] or SPI_STATUS[24], is not set to 1, then the receive FIFO buffers can be accessed through software by reading this register. This is a read-only register.

SPI Data Transmit Register (SPI_TX)

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

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

Bits	Description	
[31:0]	**TX**	**Data Transmit Register** The Data Transmit Registers hold the data to be transmitted in the next transfer. The number of valid bits depend on the setting of 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. If TX_BIT_LEN is set to 0x00, the SPI controller will perform a 32-bit transfer. **Note:** when the SPI controller is configured as a Slave device and the FIFO mode is disabled, if the SPI controller attempts to transmit data to a Master, the software must update the transmit data register before setting the GO_BUSY bit to 1.

SPI Variable Clock Pattern Register (SPI_VARCLK)

Register	Offset	R/W	Description	Reset Value
SPI_VARCLK	SPIx_BA+0x34	R/W	Variable Clock Pattern Register	0x007F_FF87

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

Bits	Description	
[31:0]	**VARCLK**	**Variable Clock Pattern** This register defines the clock pattern of the SPI transfer. If the variable clock function is disabled, this setting is unmeaning. Refer to the “Variable Clock Function” paragraph for more detail description.

SPI DMA Control Register (SPI_DMA)

Register	Offset	R/W	Description	Reset Value
SPI_DMA	SPIx_BA+0x38	R/W	SPI DMA Control Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMA_RST**	**PDMA Reset** 0 = No effect. 1 = Reset the PDMA control logic in this SPI controller. This bit will be cleared to 0 automatically.
[1]	**RX_DMA_GO**	**Receive DMA Start** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[0]	**TX_DMA_GO**	**Transmit DMA Start** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled. **Note:** In SPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, RX PDMA function cannot be enabled prior to TX PDMA function. User can enable TX PDMA function firstly or enable both functions simultaneously.

SPI Control and Status Register 2 (SPI_CNTRL2)

Register	Offset	R/W	Description	Reset Value
SPI_CNTRL2	SPIx_BA+0x3C	R/W	The second Control and Status Register	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		**DUAL_IO_EN**	**DUAL_IO_DIR**	**SLV_START_INTSTS**	**SSTA_INTEN**	**SLV_ABORT**	**NOSLVSEL**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31]	**BCn**	**SPI Peripheral Clock Backward Compatible Option** 0 = Backward compatible clock configuration. 1 = The clock configuration is not backward compatible. Refer to the description of SPI_DIVIDER register for details.
[30:17]	**Reserved**	Reserved.
[16]	**SS_INT_OPT**	**Slave Select Inactive Interrupt Option** This setting is only available if the SPI controller is configured as level trigger Slave device. 0 = As the slave select signal goes to inactive level, the IF bit will NOT be set to 1. 1 = As the slave select signal goes to inactive level, the IF bit will be set to 1.
[15:14]	**Reserved**	Reserved.
[13]	**DUAL_IO_EN**	**Dual I/O Mode Enable Bit** 0 = Dual I/O Mode function Disabled. 1 = Dual I/O Mode function Enabled.
[12]	**DUAL_IO_DIR**	**Dual I/O Mode Direction Selection** 0 = Dual input mode. 1 = Dual output mode.
[11]	**SLV_START_INTSTS**	**Slave 3-wire Mode Start Interrupt Status** This bit indicates if a transaction has started in Slave 3-wire mode. It is a mutual mirror bit of SPI_STATUS[11]. 0 = Slave has not detected any SPI clock transition since the SSTA_INTEN bit was set to 1. 1 = A transaction has started in Slave 3-wire mode. It will be cleared automatically when a transaction is done or by writing 1 to this bit.
[10]	**SSTA_INTEN**	**Slave 3-wire Mode Start Interrupt Enable Bit** Used to enable interrupt when the transfer has started in Slave 3-wire mode. If there is no transfer done interrupt over the time period which is defined by user

		after the transfer start, the user can set the SLV_ABORT bit to force the transfer done. 0 = Transaction start interrupt Disabled. 1 = Transaction start interrupt Enabled. It will be cleared to 0 as the current transfer is done or the SLV_START_INTSTS bit is cleared.
[9]	**SLV_ABORT**	**Slave 3-wire Mode Abort Control Bit** In normal operation, there is an interrupt event when the received data meet the required bits which defined in TX_BIT_LEN. If the received bits are less than the requirement and there is no more SPI clock input over one transaction time in Slave 3-wire mode, user can set this bit to force the current transfer done and then user can get a transfer done interrupt event. 0 = No effect. 1 = Force the current transaction done. **Note:** This bit will be cleared to 0 automatically by hardware after it is set to 1 by software.
[8]	**NOSLVSEL**	**Slave 3-wire Mode Enable Bit (Slave Only)** In Slave 3-wire mode, the SPI controller can work on 3-wire interface including SPIx_CLK, SPIx_MISO, and SPIx_MOSI. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface. 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 (SPI_SSR[4]) will be set as 1 automatically.
[7:0]	**Reserved**	Reserved.

SPI FIFO Control Register (SPI_FIFO_CTL)

Register	Offset	R/W	Description	Reset Value
SPI_FIFO_CTL	SPIx_BA+0x40	R/W	SPI FIFO Control Register	0x4400_0000

31	30	29	28	27	26	25	24
Reserved	TX_THRESHOLD			Reserved	RX_THRESHOLD		
23	22	21	20	19	18	17	16
Reserved		TIMEOUT_INTEN	Reserved				
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RXOV_INTEN	Reserved		TX_INTEN	RX_INTEN	TX_CLR	RX_CLR

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TX_THRESHOLD**	**Transmit FIFO Threshold** If the valid data count of the transmit FIFO buffer is less than or equal to the TX_THRESHOLD setting, the TX_INTSTS bit will be set to 1, else the TX_INTSTS bit will be cleared to 0.
[27]	**Reserved**	Reserved.
[26:24]	**RX_THRESHOLD**	**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 Bit** 0 = Receive time-out interrupt Disabled. 1 = Receive time-out interrupt Enabled.
[20:7]	**Reserved**	Reserved.
[6]	**RXOV_INTEN**	**Receive FIFO Overrun Interrupt Enable Bit** 0 = Receive FIFO overrun interrupt Disabled. 1 = Receive FIFO overrun interrupt Enabled.
[5:4]	**Reserved**	Reserved.
[3]	**TX_INTEN**	**TX Threshold Interrupt Enable Bit** 0 = TX threshold interrupt Disabled. 1 = TX threshold interrupt Enabled.
[2]	**RX_INTEN**	**RX Threshold Interrupt Enable Bit** 0 = RX threshold interrupt Disabled. 1 = RX threshold interrupt Enabled.

[1]	**TX_CLR**	**Clear Transmit FIFO Buffer** 0 = No effect. 1 = Clear Transmit FIFO buffer. The TX_FULL flag will be cleared to 0 and the TX_EMPTY flag will be set to 1. This bit will be cleared to 0 by hardware after software sets it to 1.
[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.

SPI Status Register (SPI_STATUS)

Register	Offset	R/W	Description	Reset Value
SPI_STATUS	SPIx_BA+0x44	R/W	SPI Status Register	0x0500_0000

31	30	29	28	27	26	25	24
TX_FIFO_COUNT				TX_FULL	TX_EMPTY	RX_FULL	RX_EMPTY
23	22	21	20	19	18	17	16
Reserved			TIMEOUT	Reserved			IF
15	14	13	12	11	10	9	8
RX_FIFO_COUNT				SLV_START_INTSTS	Reserved		
7	6	5	4	3	2	1	0
Reserved			TX_INTSTS	Reserved	RX_OVERRUN	Reserved	RX_INTSTS

Bits	Description	
[31:28]	**TX_FIFO_COUNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27]	**TX_FULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** A mutual mirror bit of SPI_CNTRL[27]. 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.
[26]	**TX_EMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** A mutual mirror bit of SPI_CNTRL[26]. 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[25]	**RX_FULL**	**Receive FIFO Buffer Empty Indicator (Read Only)** A mutual mirror bit of SPI_CNTRL[25]. 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[24]	**RX_EMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** A mutual mirror bit of SPI_CNTRL[24]. 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[23:21]	**Reserved**	Reserved.
[20]	**TIMEOUT**	**Time-out Interrupt Flag** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and there is not be read over 64 SPI clock period in Master mode and over 576 SPI peripheral clock period in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically.

		Note: This bit will be cleared by writing 1 to itself.
[19:17]	**Reserved**	Reserved.
[16]	**IF**	**SPI Unit Transfer Interrupt Flag** A mutual mirror bit of SPI_CNTRL[16]. 0 = No transaction has been finished since this bit was cleared to 0. 1 = SPI controller has finished one unit transfer. **Note:** This bit will be cleared by writing 1 to itself.
[15:12]	**RX_FIFO_COUNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[11]	**SLV_START_INTSTS**	**Slave Start Interrupt Status** It is used to dedicate if a transaction has started in Slave 3-wire mode. It is a mutual mirror bit of SPI_CNTRL2[11]. 0 = Slave has not detected any SPI clock transition since the SSTA_INTEN bit was set to 1. The transfer is not started. 1 = A transaction has started in Slave 3-wire mode. It will be cleared as a transaction is done or by writing 1 to this bit.
[10:5]	**Reserved**	Reserved.
[4]	**TX_INTSTS**	**Transmit FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TX_THRESHOLD. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TX_THRESHOLD.
[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. 0 = Receive FIFO does not overrun. 1 = Receive FIFO overrun. **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.

6.15 I^{2}S Controller (I^{2}S)

6.15.1 Overview

The I^{2}S controller consists of IIS protocol to interface with external audio CODEC. Two 8 word depth FIFO buffers for read path and write path respectively and is capable of handling 8/16/24/32 bits word sizes. PDMA controller handles the data movement between FIFO and memory.

6.15.2 Features

- Operated as either Master or Slave
- Capable of handling 8, 16, 24 and 32 bits word
- Supports monaural and stereo audio data
- Supports four data format:
  - I^{2}S data format
  - MSB justified data format
  - PCM mode A
  - PCM mode B
- Provides two 8 word depth FIFO buffers, one for transmitting and the other for receiving
- Generates interrupt requests when buffer levels cross a programmable boundary
- Supports PDMA transfer

6.15.3 Block Diagram

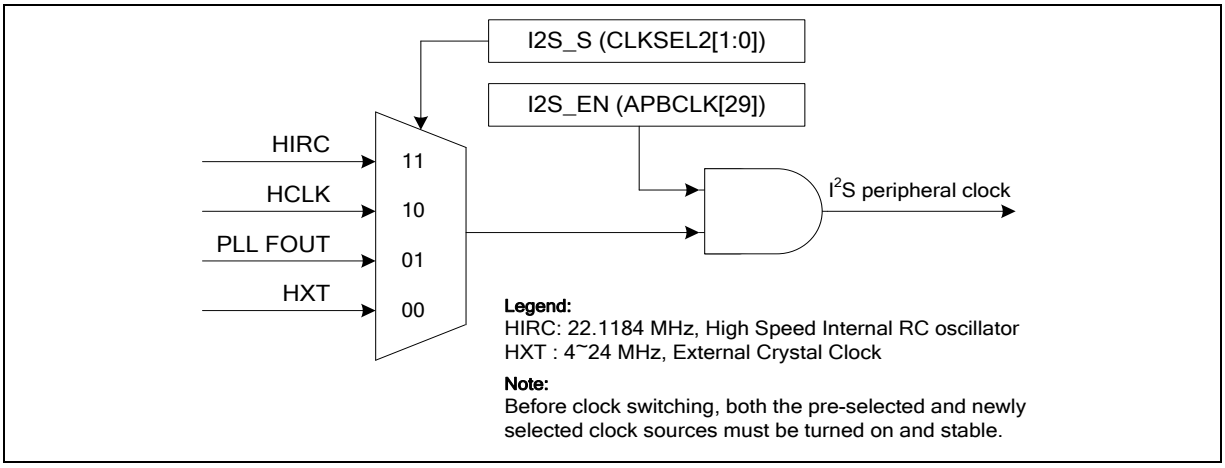


Figure 6-109 I^{2}S Clock Control Diagram

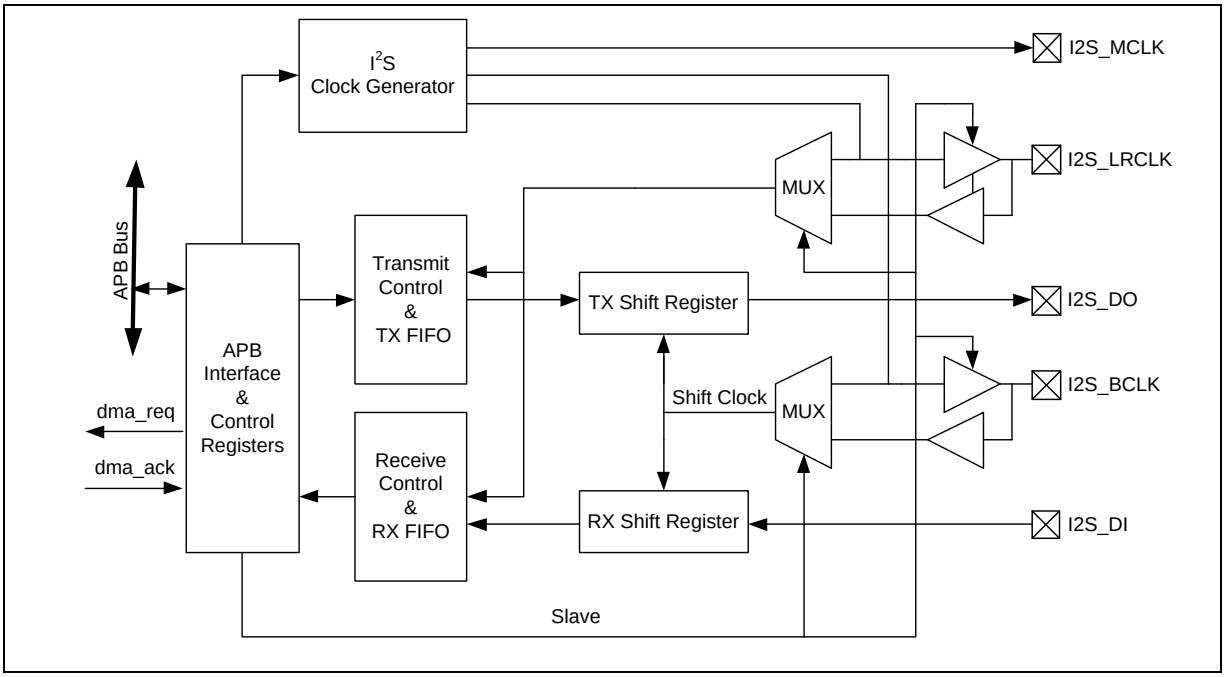


Figure 6-110 I^{2}S Controller Block Diagram

6.15.4 Basic Configuration

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

- The I^{2}S pins are configured in GPC_MFP and GPA_MFP registers. NUC123xxxAEx provides the alternative of configuring the I^{2}S pins in GPA_MFPH, GPC_MFPL and GPC_MFPH registers. (For NUC123xxxAEx, if GPA_MFPH, GPC_MFPL and GPC_MFPH are used as pin multi-function setting, the GPC_MFP and GPA_MFP will become invalid).
- Select the source of I2S_CLK in I2S_S (CLKSEL2[1:0]).
- Enable I^{2}S peripheral clock in I2S_EN (APBCLK[29]).
- Reset I^{2}S controller in I2S_RST (IPRSTC2[29]).

6.15.5 Functional Description

6.15.5.1 *Master/Slave Interface*

The I^{2}S function can operate as master or slave mode by setting SLAVE (I2S_CON[8]) to communicate with other I^{2}S slave or master device. The serial bus clock I2S_BCLK is permanently generated by the master device even through there is no transferring data bit at the moment. The word select signal I2S_LRCLK is also generated by the master device and it indicates the beginning of a new data word and the targeted audio channel. Both the I2S_LRCLK and the transmitting data change synchronously to the falling edges of I2S_BCLK.

In some applications, especially for Audio-ADC or Audio-DAC, a master clock signal, I2S_MCLK, is required with a fixed phase relation to the I2S_BCLK. The I2S_MCLK is enabled by MCLKEN (I2S_CON[15]). In Master mode, the I2S_MCLK, I2S_BCLK, I2S_LRCLK is output to device slave. And if in slave mode, the I2S_MCLK is output to device master, and I2S_BCLK or I2S_LRCLK is input from device master.

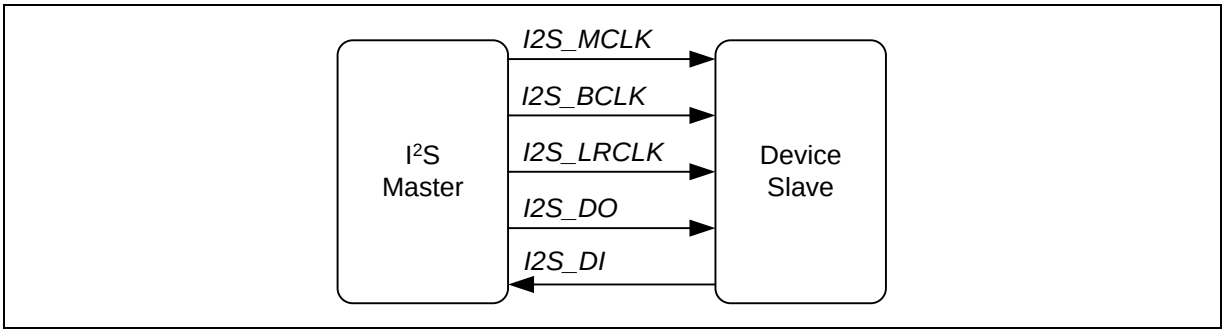


Figure 6-111 Master mode Interface Block Diagram

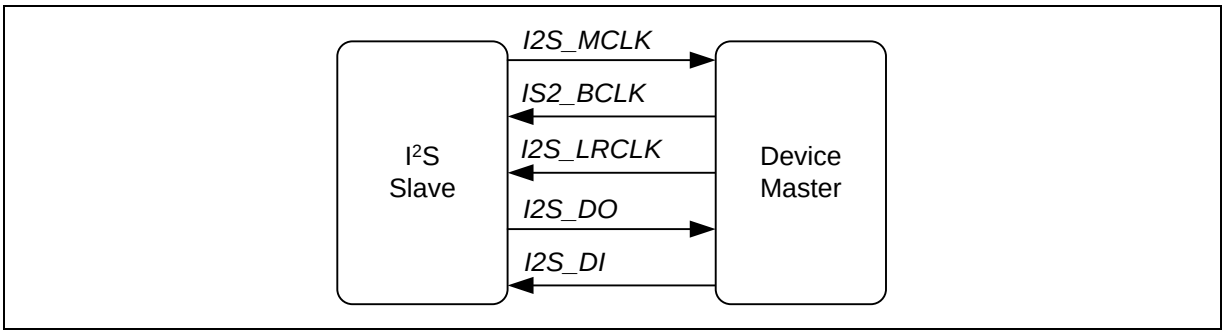


Figure 6-112 Slave mode Interface Block Diagram

6.15.5.2 *I^{2}S Operation*

The I^{2}S controller supports MSB justified and I^{2}S data format. The I2S_LRCLK signal indicates which audio channel is in transferring. The bit count of an audio channel is determined by WORDWIDTH (I2S_CON[5:4]). The transferring sequence is always started from the MSB (most significance bit) to the LSB (least significance bit).

As the figures shown below, transmitting data are read on rising edge of I2S_BCLK and sent out on falling edge of I2S_BCLK in I^{2}S protocol. In I^{2}S data format, the MSB is sent and latched at the second I2S_BCLK cycle of an audio channel. In MSB justified data format, the MSB is sent and latched at the first I2S_BCLK cycle of an audio channel. The MSB justified data format of I^{2}S protocol is selected by

FORMAT (I2S_CON[7]).

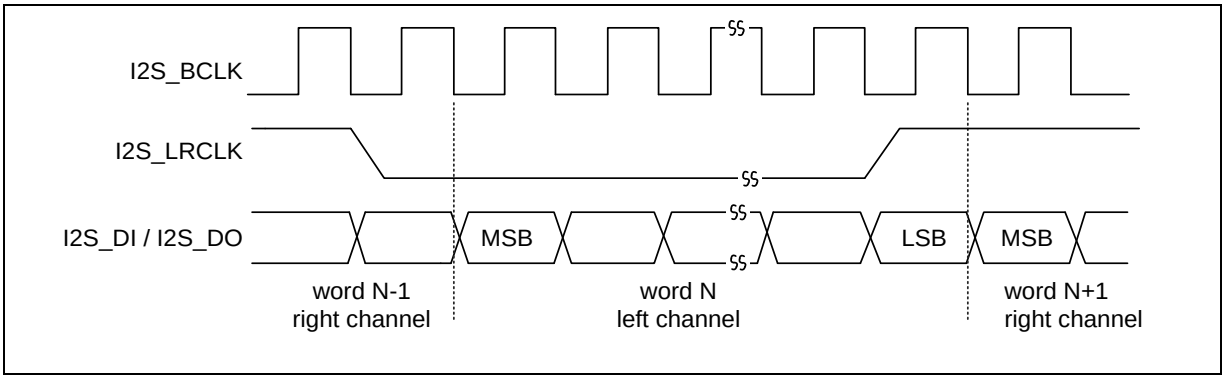


Figure 6-113 I^{2}S Bus Timing Diagram (FORMAT = 0)

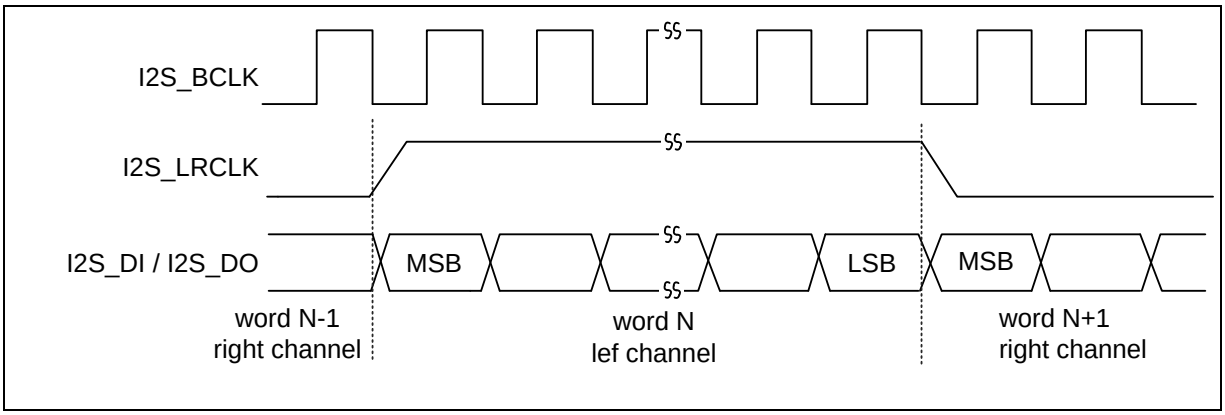


Figure 6-114 MSB Justified Timing Diagram (FORMAT = 1)

The I^{2}S controller also support PCM transmission which can be selected by PCM (I2S_CON[24]) and FORMAT (I2S_CON[7]) can be used to select PCM mode A and PCM mode B. The I2S_LRCLK in PCM protocol is used to indicate the start of a left/right audio frame.

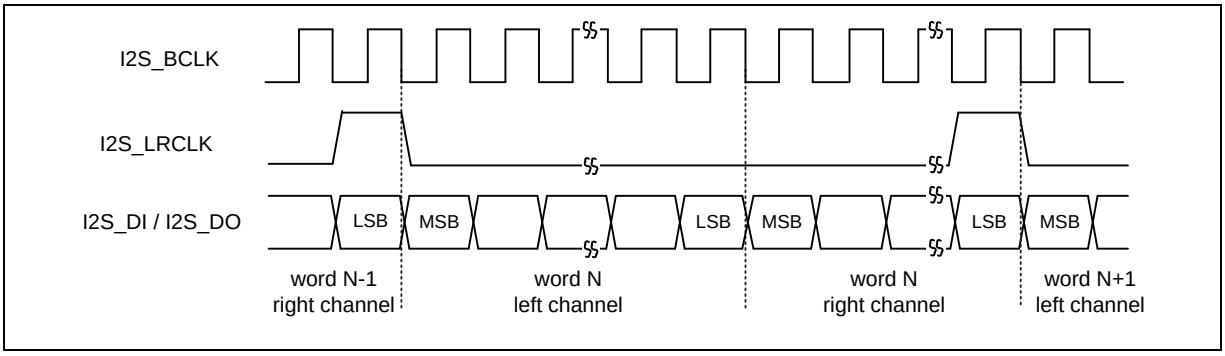


Figure 6-115 PCM A Audio Timing Diagram (FORMAT = 0)

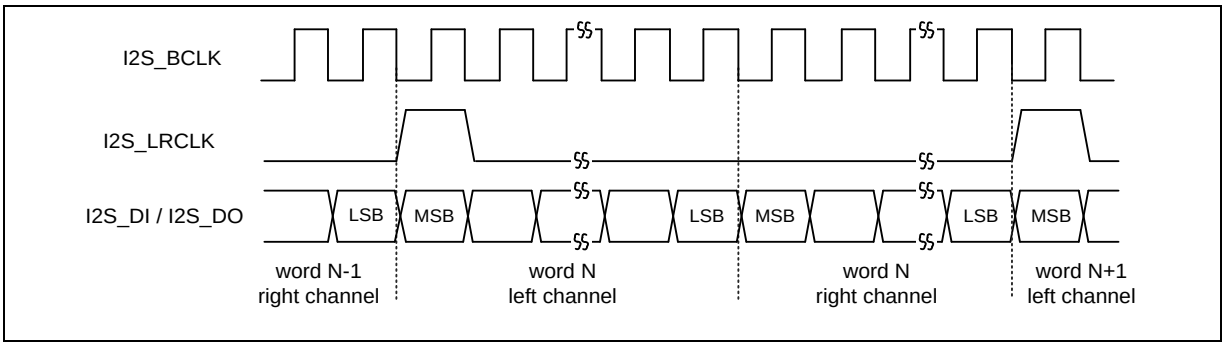


Figure 6-116 PCM B Audio Timing Diagram (FORMAT = 1)

6.15.5.3 *Zero Crossing*

When playing the audio by I^{2}S function, the output data comes from the memory by PDMA or by CPU. However, there may be some pop noise which induces the uncomfortable hearing if the playing sound volume is changed greatly by user. The zero-crossing event of audio data means the playing sound is relatively silent at the moment. Therefore, the zero-cross interrupt can be used for the indication of gain level adjustment to prevent the huge variance of sound volume.

If zero-cross detection of left/right channel is enabled by LCHZCEN (I2S_CON[17]) / RCHZCEN (I2S_CON[16]), the hardware will detect the next transferring data of left/right channel whether it is zero value or its sign bit (MSB) has been changed. If zero value or sign bit (MSB) changing of the transmitting audio data has been detected while zero-cross detection is enabled, the hardware will set the LZCF (I2S_STATUS [23]) / RZCF (I2S_STATUS [22]) for the left/right channel and then keep the output audio data silent (all data bit zero) automatically, until the corresponding zero-cross flag is cleared by software.

6.15.5.4 *PDMA Mode*

The I^{2}S function can use PDMA function to access the data without CPU's intervention. When transmitting data with PDMA function, if TX FIFO is not full, the I^{2}S will generate the PDMA request

signal and get a data from memory by PDMA automatically. When receiving data with PDMA function, if the RX FIFO is not empty, the I^{2}S will generate the PDMA request signal and move a received data to memory by PDMA automatically. User can use PDMA function to save the CPU resource consumption.

6.15.5.5 *I^{2}S Interrupt Sources*

The I^{2}S controller supports left channel zero-cross interrupt, right channel zero-cross interrupt, transmit FIFO threshold level interrupt, transmit FIFO overflow interrupt and transmit FIFO underflow interrupt in transmit operation. In receive operation, it supports receive FIFO threshold level interrupt, receive FIFO overflow interrupt and receive FIFO underflow interrupt. When I^{2}S interrupt occurs, user can check I2STXINT (I2S_STATUS[2]) and I2SRXINT (I2S_STATUS[1]) flags to recognize the interrupt sources.

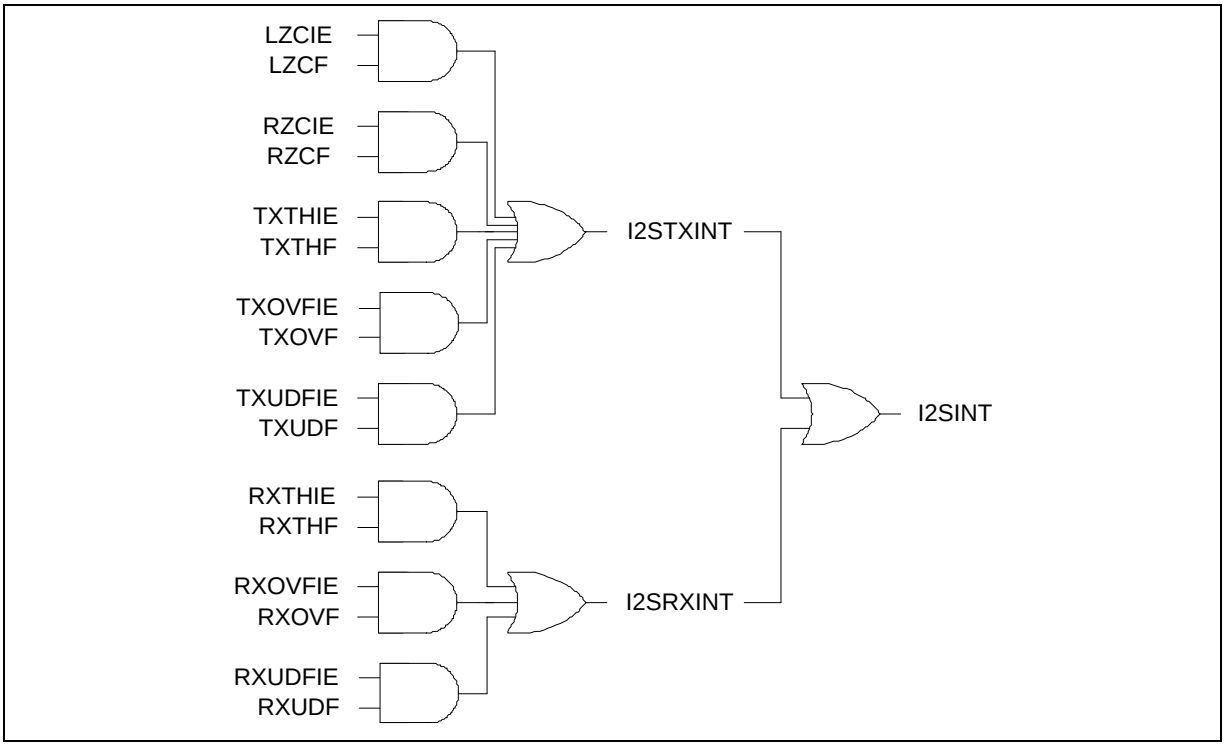


Figure 6-117 I^{2}S Interrupts

6.15.5.6 *FIFO Operation*

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

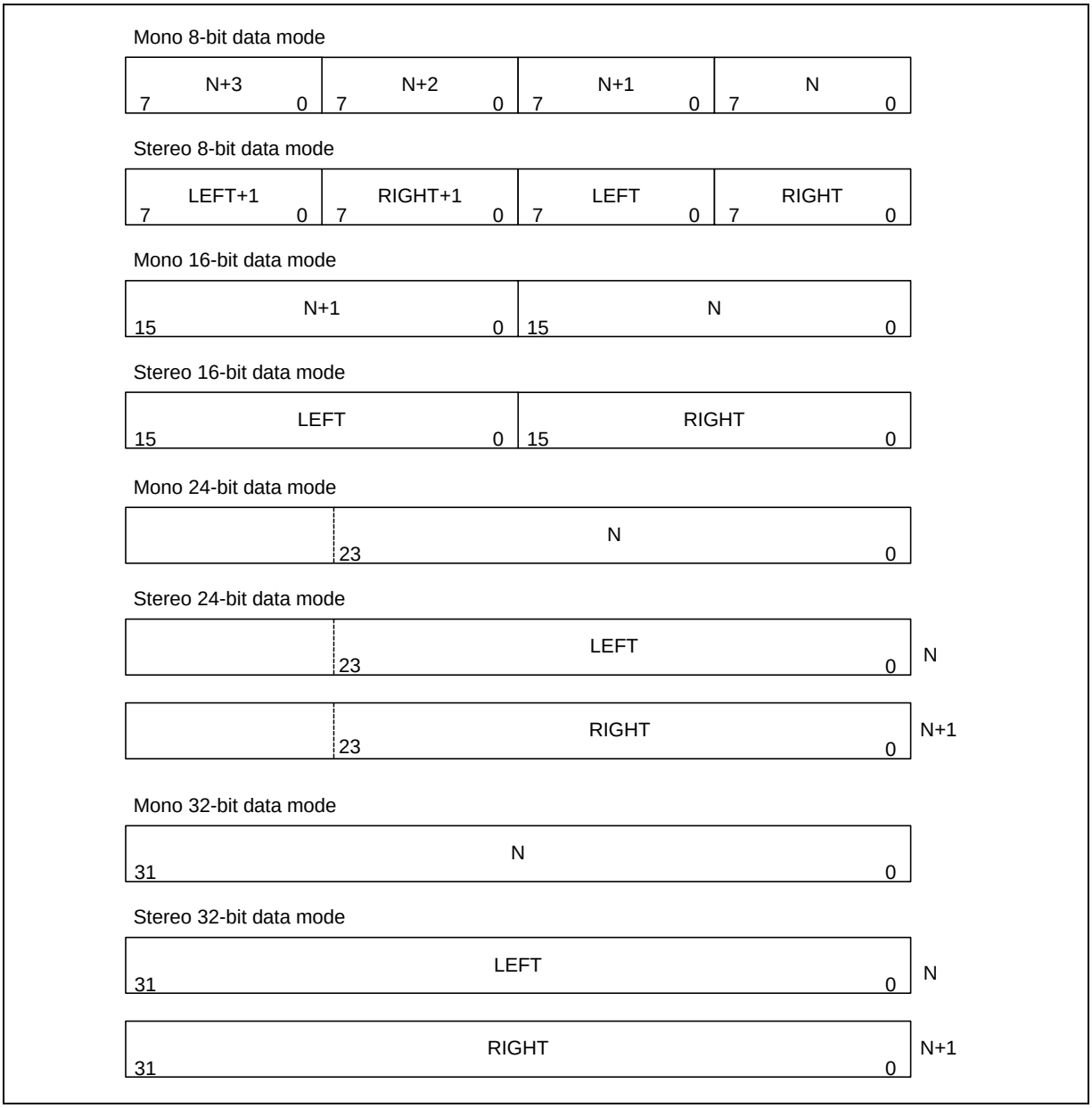


Figure 6-118 FIFO Contents for Various I^{2}S Modes

6.15.6 Register Map

R: Read only, **W**: Write only, **R/W**: Both read and write

Register	Offset	R/W	Description	Reset Value
I2S_BA = 0x401A_0000				
I2S_CON	I2S_BA+0x00	R/W	I^2S Control Register	0x0000_0000
I2S_CLKDIV	I2S_BA+0x04	R/W	I^2S Clock Divider Register	0x0000_0000
I2S_IE	I2S_BA+0x08	R/W	I^2S Interrupt Enable Register	0x0000_0000
I2S_STATUS	I2S_BA+0x0C	R/W	I^2S Status Register	0x0014_1000
I2S_TXFIFO	I2S_BA+0x10	R/W	I^2S Transmit FIFO Register	0x0000_0000
I2S_RXFIFO	I2S_BA+0x14	R/W	I^2S Receive FIFO Register	0x0000_0000

6.15.7 Register Description

I²S Control Register (I2S_CON)

Register	Offset	R/W	Description	Reset Value
I2S_CON	I2S_BA+0x00	R/W	I²S Control Register	0x0000_0000

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**PCM**	**PCM Interface Enable Bit** 0 = I²S Interface. 1 = PCM interface.
[23]	**RXLCH**	**Receive Left Channel Enable Bit** When monaural format is selected (MONO = 1), I²S will receive right channel data if RXLCH is set to 0, and receive left channel data if RXLCH is set to 1. 0 = Receives right channel data when monaural format is selected. 1 = Receives left channel data when monaural format is selected.
[22]	**Reserved**	Reserved.
[21]	**RXDMA**	**Receive DMA Enable Bit** When RX DMA is enabled, I²S requests DMA to transfer data from receive FIFO to SRAM if FIFO is not empty. 0 = RX DMA Disabled. 1 = RX DMA Enabled.
[20]	**TXDMA**	**Transmit DMA Enable Bit** When TX DMA is enabled, I²S request DMA to transfer data from SRAM to transmit FIFO if FIFO is not full. 0 = TX DMA Disabled. 1 = TX DMA Enabled.
[19]	**CLR_RXFIFO**	**Clear Receive FIFO** Write 1 to clear receive FIFO, internal pointer is reset to FIFO start point, and RX_LEVEL[3:0] returns to zero and receive FIFO becomes empty. **Note:** This bit is cleared by hardware automatically, and reading it returns zero.

[18]	**CLR_TXFIFO**	**Clear Transmit FIFO** Write 1 to clear transmit FIFO, internal pointer is reset to FIFO start point, and TX_LEVEL[3:0] returns to zero and transmit FIFO becomes empty but data in transmit FIFO is not changed. **Note:** This bit is cleared by hardware automatically, and reading it returns zero.
[17]	**LCHZCEN**	**Left Channel Zero Cross Detection Enable Bit** If this bit is set to 1, when left channel data sign bit is changed or all bits of the next shift data are zero, then LZCF flag (I2S_STATUS[23]) is set to 1. 0 = Left channel zero-cross detection Disabled. 1 = Left channel zero-cross detection Enabled.
[16]	**RCHZCEN**	**Right Channel Zero-cross Detection Enable Bit** If this bit is set to 1, when left channel data sign bit is changed or all bit of the next shift data are zero, then RZCF flag (I2S_STATUS[22]) is set to 1. 0 = Right channel zero-cross detection Disabled. 1 = Right channel zero-cross detection Enabled.
[15]	**MCLKEN**	**Master Clock Enable Bit** If MCLKEN is set to 1, I^2S controller will generate master clock on I2S_MCLK pin for external audio devices. 0 = Master clock Disabled. 1 = Master clock Enabled.
[14:12]	**RXTH**	**Receive FIFO Threshold Level Selection** When received data word(s) in buffer is equal to or higher than threshold level, the RXTHF flag is set. 000 = 1 word data in receive FIFO. 001 = 2 word data in receive FIFO. 010 = 3 word data in receive FIFO. 011 = 4 word data in receive FIFO. 100 = 5 word data in receive FIFO. 101 = 6 word data in receive FIFO. 110 = 7 word data in receive FIFO. 111 = 8 word data in receive FIFO.
[11:9]	**TXTH**	**Transmit FIFO Threshold Level Selection** If remain data word in transmit FIFO is equal to or less than threshold level, the TXTHF flag is set. 000 = 0 word data in transmit FIFO. 001 = 1 word data in transmit FIFO. 010 = 2 words data in transmit FIFO. 011 = 3 words data in transmit FIFO. 100 = 4 words data in transmit FIFO. 101 = 5 words data in transmit FIFO. 110 = 6 words data in transmit FIFO. 111 = 7 words data in transmit FIFO.
[8]	**SLAVE**	**Slave Mode Enable Bit** I^2S can be operated as Master or Slave mode. For Master mode, I2S_BCLK and I2S_LRCLK pins are output mode and send bit clock to audio CODEC chip. In Slave mode, I2S_BCLK and I2S_LRCLK pins are input mode and I2S_BCLK and I2S_LRCLK signals are received from outer Audio CODEC chip. 0 = Master mode. 1 = Slave mode.

[7]	**FORMAT**	**Data Format Selection** If PCM=0,. 0 = I^2S data format. 1 = MSB justified data format. If PCM=1,. 0 = PCM mode A. 1 = PCM mode B.
[6]	**MONO**	**Monaural Data Format Enable Bit** 0 = Data is stereo format. 1 = Data is monaural format.
[5:4]	**WORDWIDTH**	**Word Width Selection** 00 = Data is 8-bit. 01 = Data is 16-bit. 10 = Data is 24-bit. 11 = Data is 32-bit.
[3]	**MUTE**	**Transmit Mute Enable Bit** 0 = Transmit data is shifted from buffer. 1= Transmit channel zero.
[2]	**RXEN**	**Receive Enable Bit** 0 = Data receiving Disabled. 1 = Data receiving Enabled.
[1]	**TXEN**	**Transmit Enable Bit** 0 = Data transmission Disabled. 1 = Data transmission Enabled.
[0]	**I2SEN**	**I^2S Controller Enable Bit** 0 = I^2S controller Disabled. 1 = I^2S controller Enabled.

I²S Clock Divider (I2S_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2S_CLKDIV	I2S_BA+0x04	R/W	I²S Clock Divider Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**BCLK_DIV**	**Bit Clock Divider** If I²S operates in Master mode, I²S controller will generate bit clock on I2S_BCLK pin. The bit clock rate, F_BCLK, is determined by the following expression. F_BCLK = F_I2SCLK / (2x(BCLK_DIV + 1)), where F_I2SCLK is the frequency of I²S peripheral clock.
[7:3]	**Reserved**	Reserved.
[2:0]	**MCLK_DIV**	**Master Clock Divider** If MCLKEN is set to 1, I²S controller will generate master clock for external audio devices. The master clock rate, F_MCLK, is determined by the following expressions. If MCLK_DIV >= 1, F_MCLK = F_I2SCLK / (2x(MCLK_DIV)). If MCLK_DIV = 0, F_MCLK = F_I2SCLK. F_I2SCLK is the frequency of I²S peripheral clock. In general, the master clock rate is 256 times sampling clock rate.

I²S Interrupt Enable Register (I2S_IE)

Register	Offset	R/W	Description	Reset Value
I2S_IE	I2S_BA+0x08	R/W	I²S Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**LZCIE**	**Left Channel Zero-cross Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and left channel zero-cross. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[11]	**RZCIE**	**Right Channel Zero-cross Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[10]	**TXTHIE**	**Transmitted FIFO Threshold Level Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and data words in transmit FIFO are less than TXTH[2:0]. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[9]	**TXOVFIE**	**Transmitted FIFO Overflow Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and transmitted FIFO overflow flag is set to 1. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[8]	**TXUDFIE**	**Transmitted FIFO Underflow Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and transmitted FIFO underflow flag is set to 1. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[7:3]	**Reserved**	Reserved.
[2]	**RXTHIE**	**Received FIFO Threshold Level Interrupt Enable Bit** When data word in receive FIFO is equal to or higher then RXTH[2:0] and the RXTHF bit is set to 1. If RXTHIE bit is enabled, interrupt occurs. 0 = Interrupt Disabled. 1 = Interrupt Enabled.

[1]	**RXOVFIE**	**Receive FIFO Overflow Interrupt Enable Bit** 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[0]	**RXUDFIE**	**Receive FIFO Underflow Interrupt Enable Bit** If software reads the received FIFO when it is empty, RXUDF flag (I2S_STATUS[8]) is set to 1. 0 = Interrupt Disabled. 1 = Interrupt Enabled.

I²S Status Register (I2S_STATUS)

Register	Offset	R/W	Description	Reset Value
I2S_STATUS	I2S_BA+0x0C	R/W	I²S Status Register	0x0014_1000

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

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

[20]	**TXEMPTY**	**Transmit FIFO Empty (Read Only)** This bit reflect data word number in transmit FIFO is zero. 0 = Not empty. 1 = Empty.
[19]	**TX_FULL**	**Transmit FIFO Full (Read Only)** This bit reflect data word number in transmit FIFO is 8. 0 = Not full. 1 = Full.
[18]	**TXTHF**	**Transmit FIFO Threshold Flag (Read Only)** When the number of data word(s) in transmit FIFO is equal to or less than threshold value set in TXTH[2:0], the TXTHF bit becomes to 1. It keeps at 1 till TX_LEVEL[3:0] is larger than TXTH[2:0]. 0 = The number of data word(s) in FIFO is larger than threshold level. 1 = The number of data word(s) in FIFO is equal to or less than threshold level.
[17]	**TXOVF**	**Transmit FIFO Overflow Flag** Write data to transmit FIFO when it is full and this bit set to 1. 0 = No overflow. 1 = Overflow. **Note:** This bit can be cleared by software writing '1'.
[16]	**TXUDF**	**Transmit FIFO Underflow Flag** When transmit FIFO is empty and shift logic hardware read data from data FIFO causes this set to 1. 0 = No underflow. 1 = Underflow. **Note:** This bit can be cleared by software writing '1'.
[15:13]	**Reserved**	Reserved.
[12]	**RXEMPTY**	**Receive FIFO Empty (Read Only)** This bit reflects data words number in receive FIFO is zero. 0 = Not empty. 1 = Empty.
[11]	**RX_FULL**	**Receive FIFO Full (Read Only)** This bit reflect data words number in receive FIFO is 8. 0 = Not full. 1 = Full.
[10]	**RXTHF**	**Receive FIFO Threshold Flag (Read Only)** When the number of data word(s) in receive FIFO is equal to or larger than threshold value set in RXTH[2:0], the RXTHF bit becomes to 1. It keeps at 1 till RX_LEVEL[3:0] is less than RXTH[2:0]. 0 = The number of data word(s) in FIFO is less than threshold level. 1 = The number of data word(s) in FIFO is equal to or larger than threshold level.
[9]	**RXOVF**	**Receive FIFO Overflow Flag** When receive FIFO is full and hardware attempt to write data to receive FIFO, this bit will be set to 1, data in 1st buffer is overwrote. 0 = No overflow occurred. 1 = Overflow occurred. **Note:** This bit can be cleared by software writing '1'.

[8]	**RXUDF**	**Receive FIFO Underflow Flag** Underflow event will occur if read the empty receive FIFO. 0 = No underflow occurred. 1 = Underflow occurred. **Note:** This bit can be cleared by software writing '1'.
[7:4]	**Reserved**	Reserved.
[3]	**RIGHT**	**Right Channel (Read Only)** This bit indicates the current transmit data is belong to right channel. 0 = Left channel. 1 = Right channel.
[2]	**I2STXINT**	**I^2S Transmit Interrupt (Read Only)** 0 = No transmit interrupt. 1 = Transmit interrupt.
[1]	**I2SRXINT**	**I^2S Receive Interrupt (Read Only)** 0 = No receive interrupt. 1 = Receive interrupt.
[0]	**I2SINT**	**I^2S Interrupt Flag (Read Only)** This bit is wire-OR of I2STXINT and I2SRXINT bits. 0 = No I^2S interrupt. 1 = I^2S interrupt.

I^{2}S Transmit FIFO (I2S_TXFIFO)

Register	Offset	R/W	Description	Reset Value
I2S_TXFIFO	I2S_BA+0x10	R/W	I^{2}S Transmit FIFO	0x0000_0000

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

Bits	Description	
[31:0]	**TXFIFO**	**Transmit FIFO Buffer** I^{2}S contains 8 words (8x32 bit) data buffer for data transmission. Write data to this register to prepare data for transmission. The remaining word number is indicated by TX_LEVEL (I2S_STATUS[31:28]).

I²S Receive FIFO (I2S_RXFIFO)

Register	Offset	R/W	Description	Reset Value
I2S_RXFIFO	I2S_BA+0x14	R/W	I²S Receive FIFO	0x0000_0000

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

Bits	Description	
[31:0]	**RXFIFO**	**Receive FIFO Buffer** I²S contains 8 words (8x32 bit) data buffer for data receive. Read this register to get data in FIFO. The remaining data word number is indicated by RX_LEVEL[3:0] (I2S_STATUS[27:24]).

6.16 USB Device Controller (USB)

6.16.1 Overview

There is one set of USB 2.0 full-speed device controller and transceiver in this device. It is compliant with USB 2.0 full-speed device specification and supports Control/Bulk/Interrupt/ Isochronous transfer types.

In this device controller, there are two main interfaces: APB bus and USB bus which comes from the USB PHY transceiver. For the APB bus, the CPU can program control registers through it. There are 512 bytes internal SRAM as data buffer in this controller. For IN or OUT transfer, it is necessary to write data to SRAM or read data from SRAM through the APB interface or SIE. User needs to set the effective starting address of SRAM for each endpoint buffer through buffer segmentation register (BUFSEGx).

There are 8 endpoints in this controller. Each of the endpoint can be configured as IN or OUT endpoint. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. The block of ENDPOINT CONTROL is also used to manage the data sequential synchronization, endpoint state control, current start address, transaction status, and data buffer status for each endpoint.

There are four different interrupt events in this controller. They are the wake-up event, device plug-in or plug-out event, USB events, such as IN ACK, OUT ACK, and BUS events, such as suspend and resume, etc. Any event will cause an interrupt, and user just needs to check the related event flags in interrupt event status register (USB_INTSTS) to acknowledge what kind of interrupt occurring, and then check the related USB Endpoint Status Register (USB_EPSTS) to acknowledge what kind of event occurring in this endpoint.

A software-disable function is also supported for this USB controller. It is used to simulate the disconnection of this device from the host. If user enables DRVSE0 bit (USB_DRVSE0), the USB controller will force the output of USB_D+ and USB_D- to level low and its function is disabled. After disable the DRVSE0 bit, host will enumerate this USB device again.

For more information on the Universal Serial Bus, please refer to *Universal Serial Bus Specification Revision 1.1.*

6.16.2 Features

- Compliant with USB 2.0 Full-Speed specification
- Provides 1 interrupt vector with 4 different interrupt events (WAKEUP, FLDET, USB and BUS)
- Supports Control/Bulk/Interrupt/Isochronous transfer types
- Supports suspend function when no bus activity existing for 3 ms
- Provides 8 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 512 bytes buffer size
- Provides remote wake-up capability

6.16.3 Block Diagram

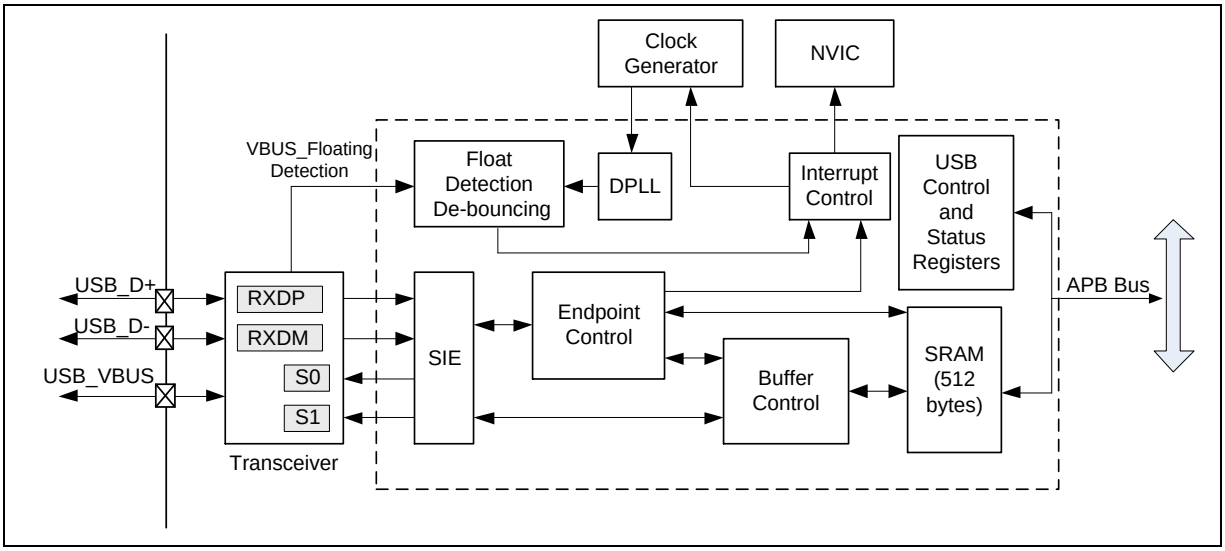


Figure 6-119 USB Block Diagram

6.16.4 Basic Configuration

The USBD clock source is derived from PLL. User has to set the PLL related configurations before USB device controller is enabled.

- Enable USBD peripheral clock in USBD_EN (CLKAPB[27]).
- USBD clock rate is generated by 4-bit pre-scaler USBD_N (CLKDIV[7:4]).
- Reset USBD controller in USBD_RST (IPRSTC2[27]).

6.16.5 Functional Description

6.16.5.1 *SIE (Serial Interface Engine)*

The SIE is the front-end of the device controller and handles most of the USB packet protocol. The SIE typically comprehends signaling up to the transaction level. The functions that it handles could include:

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

6.16.5.2 *Endpoint Control*

There are 8 endpoints in this controller. There are 2 different configuration addresses for endpoint 0~5 (see the register mapping table in section 6.16.6). Each of the endpoint can be configured as Control,

Bulk, Interrupt or Isochronous transfer type. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. It is also used to manage the data sequential synchronization, endpoint state control, current endpoint start address, current transaction status, and data buffer status in each endpoint.

6.16.5.3 *Digital Phase Lock Loop*

The bit rate of USB data is 12 MHz. The DPLL uses the 48 MHz which comes from the clock controller to lock the input data RXDP and RXDM. The 12 MHz bit rate clock is also converted from DPLL.

6.16.5.4 *Floating De-bounce*

A USB device may be plugged-in or plugged-out from the USB host. To monitor the state of a USB device when it is detached from the USB host, the device controller provides hardware de-bounce for USB floating detect interrupt to avoid bounce problems on USB plug-in or unplug. Floating detect interrupt appears about 10 ms later than USB plug-in or plug-out. User can acknowledge USB plug-in/plug-out by reading register “USB_FLDET”. The flag in “FLDET” represents the current state on the bus without de-bounce. If the FLDET is 1, it means the controller has plug-in the USB. If user polls this flag to check USB state, he/she must add software de-bounce if necessary.

6.16.5.5 *Interrupt*

This USB provides 1 interrupt vector with 4 interrupt events (WAKEUP, FLDET, USB and BUS). The WAKEUP event is used to wake-up the system clock when the Power-down mode is enabled. (The power mode function is defined in system Power-down control register, PWRCON). The FLDET event is used for USB plug-in or unplug. The USB event notifies user of some USB requests, like IN ACK, OUT ACK etc., and the BUS event notifies user of some bus events, like suspend, resume, etc. User must set related bits in the interrupt enable register (USB_INTEN) of USB Device Controller to enable USB interrupts.

Wake-up interrupt is only present when the chip entered Power-down mode and then wake-up event had happened. After the chip enters Power-down mode, any change on USB_D+ or USB_D- can wake-up this chip (provided that USB wake-up function is enabled). If this change is not intentional, no interrupt but wake-up interrupt will occur. After USB wake-up, this interrupt will occur when no other USB interrupt events are present for more than 20 ms. The Figure 6-120 is the control flow of wake-up interrupt.

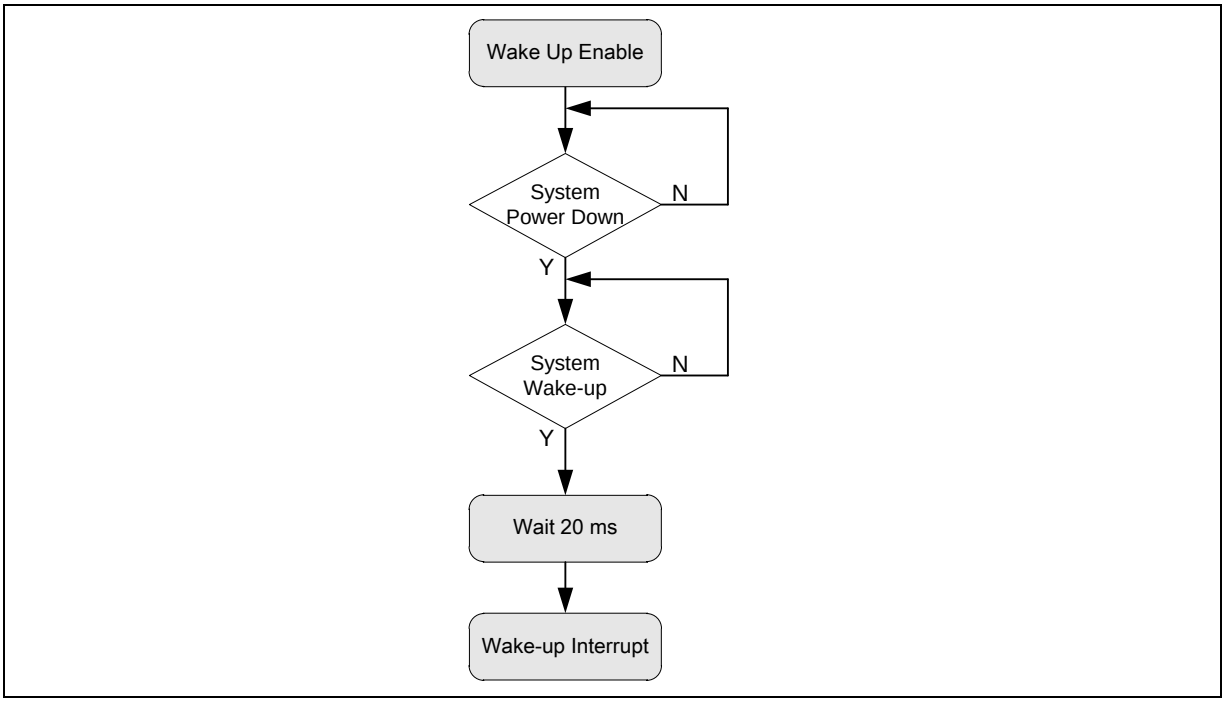


Figure 6-120 Wake-up Interrupt Operation Flow

USB interrupt is used to notify user of any USB event on the bus, and a user can read EPSTS (USB_EPSTS[31:8]) and EPEVT7~0 (USB_INTSTS[23:16]) to know what kind of request is to which endpoint and take necessary responses.

Same as USB interrupt, BUS interrupt notifies user of some bus events, such as USB reset, suspend, time-out and resume. User can read USB_ATTR to acknowledge bus events.

6.16.5.6 *Power Saving*

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

6.16.5.7 *Buffer Control*

There is 512 bytes SRAM in the controller and the 8 endpoints share this buffer. The user shall configure each endpoint's effective starting address in the buffer segmentation register before the USB function active. The BUFFER CONTROL block is used to control each endpoint's effective starting address and its SRAM size is defined in the MXPLD register.

Figure 6-121 depicts the starting address for each endpoint according the content of BUFSEG and MXPLD registers. If the BUFSEG0 is programmed as 0x08h and MXPLD0 is set as 0x40h, the SRAM size of endpoint 0 is start from USB_BA+0x108h and end in USB_BA+0x148h. (**Note:** the USB SRAM base is USB_BA+0x100h).

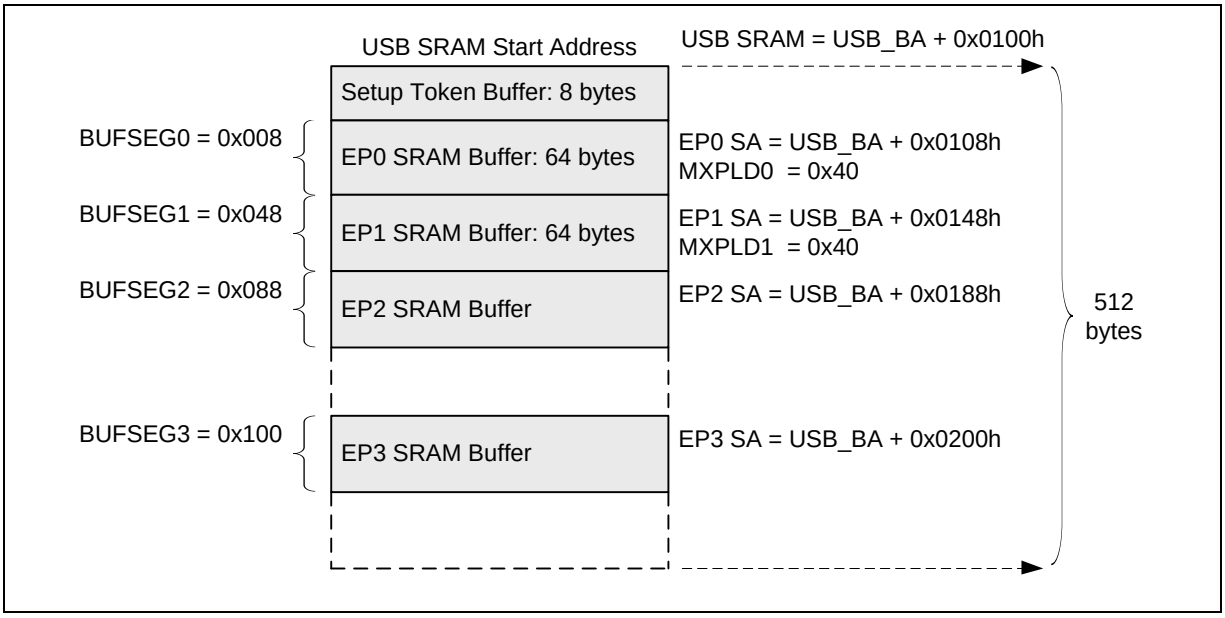


Figure 6-121 Endpoint SRAM Structure

6.16.5.8 *Handling Transactions with USB Device Peripheral*

User can use interrupt or poll USB_INTSTS to monitor the USB transactions. When transactions occur, USB_INTSTS will be set by hardware and send an interrupt request to CPU (if related interrupt enabled), or user can polling USB_INTSTS to get these events without interrupt. The following is the control flow with interrupt enabled.

When USB host has requested data from device controller, user needs to prepare related data into the specified endpoint buffer in advance. After buffering the required data, user needs to write the actual data length in the specified MAXPLD register. Once this register is written, the internal signal “In_Rdy” will be asserted and the buffering data will be transmitted immediately after receiving associated IN token from Host. Note that after transferring the specified data, the signal “In_Rdy” will de-assert automatically by hardware.

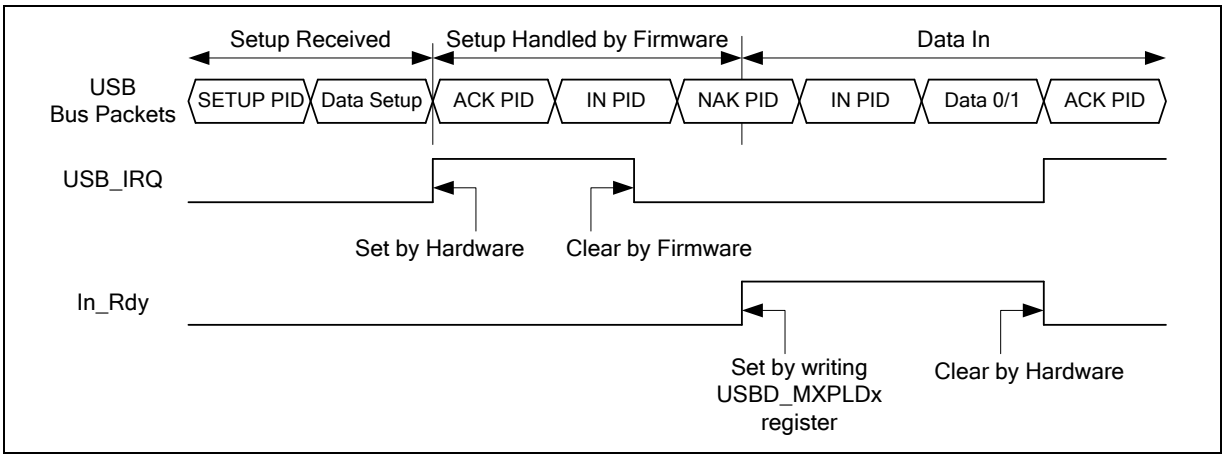


Figure 6-122 Setup Transaction Followed by Data in Transaction

Alternatively, when USB host wants to transmit data to the OUT endpoint in the device controller,

hardware will buffer these data to the specified endpoint buffer. After this transaction is completed, hardware will record the data length in related MAXPLD register and de-assert the signal “Out_Rdy”. This will avoid hardware accepting next transaction until user moves out current data in the related endpoint buffer. Once user has processed this transaction, the related register “MAXPLD” needs to be written by firmware to assert the signal “Out_Rdy” again to accept next transaction.

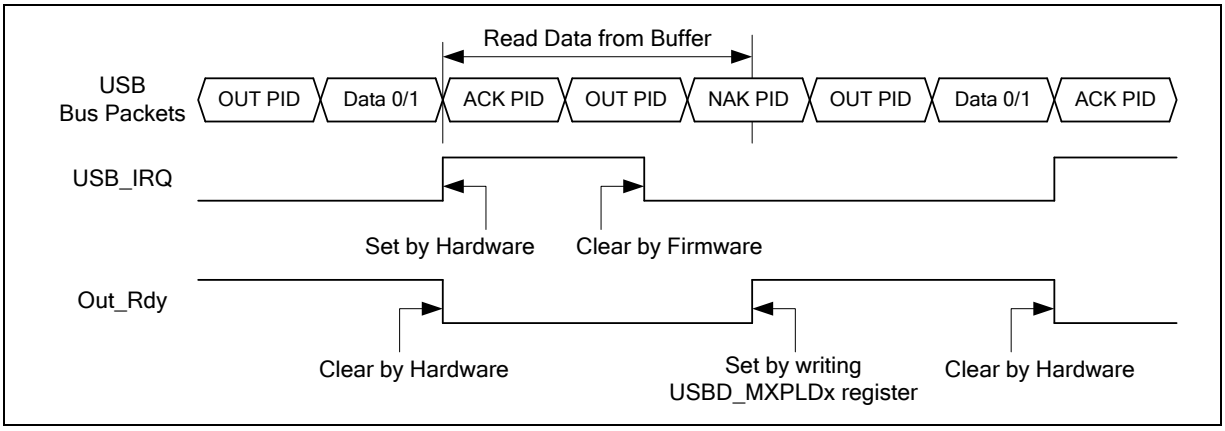


Figure 6-123 Data Out Transfer

6.16.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
USB_BA = 0x4006_0000				
USB_INTEN	USB_BA+0x000	R/W	USB Interrupt Enable Register	0x0000_0000
USB_INTSTS	USB_BA+0x004	R/W	USB Interrupt Event Status Register	0x0000_0000
USB_FADDR	USB_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000
USB_EPSTS	USB_BA+0x00C	R	USB Endpoint Status Register	0x0000_00x0
USB_ATTR	USB_BA+0x010	R/W	USB Bus Status and Attribution Register	0x0000_0040
USB_FLDET	USB_BA+0x014	R	USB Floating Detected Register	0x0000_0000
USB_BUFSEG	USB_BA+0x018	R/W	Setup Token Buffer Segmentation Register	0x0000_0000
USB_BUFSEG0	USB_BA+0x020	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USB_MXPLD0	USB_BA+0x024	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USB_CFG0	USB_BA+0x028	R/W	Endpoint 0 Configuration Register	0x0000_0000
USB_CFGP0	USB_BA+0x02C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG1	USB_BA+0x030	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USB_MXPLD1	USB_BA+0x034	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USB_CFG1	USB_BA+0x038	R/W	Endpoint 1 Configuration Register	0x0000_0000
USB_CFGP1	USB_BA+0x03C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG2	USB_BA+0x040	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USB_MXPLD2	USB_BA+0x044	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USB_CFG2	USB_BA+0x048	R/W	Endpoint 2 Configuration Register	0x0000_0000
USB_CFGP2	USB_BA+0x04C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG3	USB_BA+0x050	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USB_MXPLD3	USB_BA+0x054	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USB_CFG3	USB_BA+0x058	R/W	Endpoint 3 Configuration Register	0x0000_0000
USB_CFGP3	USB_BA+0x05C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG4	USB_BA+0x060	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USB_MXPLD4	USB_BA+0x064	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USB_CFG4	USB_BA+0x068	R/W	Endpoint 4 Configuration Register	0x0000_0000
USB_CFGP4	USB_BA+0x06C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

USB_BUFSEG5	USB_BA+0x070	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USB_MXPLD5	USB_BA+0x074	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USB_CFG5	USB_BA+0x078	R/W	Endpoint 5 Configuration Register	0x0000_0000
USB_CFGP5	USB_BA+0x07C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_DRVSE0	USB_BA+0x090	R/W	USB Drive SE0 Control Register	0x0000_0001
USB_PDMA	USB_BA+0x0A4	R/W	USB PDMA Control Register	0x0000_0000
USB_BUFSEG0	USB_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USB_MXPLD0	USB_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USB_CFG0	USB_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USB_CFGP0	USB_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG1	USB_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USB_MXPLD1	USB_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USB_CFG1	USB_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USB_CFGP1	USB_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG2	USB_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USB_MXPLD2	USB_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USB_CFG2	USB_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USB_CFGP2	USB_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG3	USB_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USB_MXPLD3	USB_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USB_CFG3	USB_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USB_CFGP3	USB_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG4	USB_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USB_MXPLD4	USB_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USB_CFG4	USB_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USB_CFGP4	USB_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG5	USB_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USB_MXPLD5	USB_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USB_CFG5	USB_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USB_CFGP5	USB_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG6	USB_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000

USB_MXPLD6	USB_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USB_CFG6	USB_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USB_CFGP6	USB_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_BUFSEG7	USB_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000
USB_MXPLD7	USB_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000
USB_CFG7	USB_BA+0x578	R/W	Endpoint 7 Configuration Register	0x0000_0000
USB_CFGP7	USB_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

Memory Type	**Address**	**Size**	**Description**
USB_BA = 0x4006_0000			
SRAM	USB_BA+0x100 ~ USB_BA+0x2FF	512 Bytes	The SRAM is used for the entire endpoints buffer. Refer to section 6.16.5.7 for the endpoint SRAM structure and its description.

6.16.7 Register Description

USB Interrupt Enable Register (USB_INTEN)

Register	Offset	R/W	Description	Reset Value
USB_INTEN	USB_BA+0x000	R/W	USB Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**INNAK_EN**	**Active NAK Function and Its Status in IN Token** 0 = When device responds NAK after receiving IN token NAK status wasn't updated into the endpoint status register (USBD_EPSTS), so that the USB interrupt event will not be asserted. 1 = IN NAK status will be updated to USBD_EPSTS register and the USB interrupt event will be asserted, when the device responds NAK after receiving IN token.
[14:9]	**Reserved**	Reserved.
[8]	**WAKEUP_EN**	**Wake-up Function Enable Bit** 0 = USB wake-up function Disabled. 1 = USB wake-up function Enabled.
[7:4]	**Reserved**	Reserved.
[3]	**WAKEUP_IE**	**USB Wake-up Interrupt Enable Bit** 0 = Wake-up Interrupt Disabled. 1 = Wake-up Interrupt Enabled.
[2]	**FLDET_IE**	**Floating Detected Interrupt Enable Bit** 0 = Floating detect Interrupt Disabled. 1 = Floating detect Interrupt Enabled.
[1]	**USB_IE**	**USB Event Interrupt Enable Bit** 0 = USB event interrupt Disabled. 1 = USB event interrupt Enabled.
[0]	**BUS_IE**	**Bus Event Interrupt Enable Bit** 0 = BUS event interrupt Disabled. 1 = BUS event interrupt Enabled.

USB Interrupt Event Status Register (USB_INTSTS)

This register is a USB Interrupt Event Status register and can be cleared by writing '1' to the corresponding bit.

Register	Offset	R/W	Description	Reset Value
USB_INTSTS	USB_BA+0x004	R/W	USB Interrupt Event Status Register	0x0000_0000

31	30	29	28	27	26	25	24
SETUP	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
EPEVT7	**EPEVT6**	**EPEVT5**	**EPEVT4**	**EPEVT3**	**EPEVT2**	**EPEVT1**	**EPEVT0**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**WAKEUP_STS**	**FLDET_STS**	**USB_STS**	**BUS_STS**

Bits	Description	
[31]	**SETUP**	**Setup Event Status** 0 = No Setup event. 1 = Setup event occurred, and cleared by writing 1 to USB_INTSTS[31].
[30:24]	**Reserved**	Reserved.
[23]	**EPEVT7**	**Endpoint 7's USB Event Status** 0 = No event occurred in endpoint 7. 1 = USB event occurred on Endpoint 7, check USB_EPSTS[31:29] to know which kind of USB event was occurred, cleared by writing 1 to USB_INTSTS[23] or USB_INTSTS[1].
[22]	**EPEVT6**	**Endpoint 6's USB Event Status** 0 = No event occurred on Endpoint 6. 1 = USB event occurred on Endpoint 6, check USB_EPSTS[28:26] to know which kind of USB event was occurred, cleared by writing 1 to USB_INTSTS[22] or USB_INTSTS[1].
[21]	**EPEVT5**	**Endpoint 5's USB Event Status** 0 = No event occurred on Endpoint 5. 1 = USB event occurred on Endpoint 5, check USB_EPSTS[25:23] to know which kind of USB event was occurred, cleared by writing 1 to USB_INTSTS[21] or USB_INTSTS[1].
[20]	**EPEVT4**	**Endpoint 4's USB Event Status** 0 = No event occurred on Endpoint 4. 1 = USB event occurred on Endpoint 4, check USB_EPSTS[22:20] to know which kind of USB event was occurred, cleared by writing 1 to USB_INTSTS[20] or USB_INTSTS[1].
[19]	**EPEVT3**	**Endpoint 3's USB Event Status** 0 = No event occurred on Endpoint 3. 1 = USB event occurred on Endpoint 3, check USB_EPSTS[19:17] to know which kind of USB event was occurred, cleared by writing 1 to USB_INTSTS[19] or USB_INTSTS[1].

[18]	**EPEVT2**	**Endpoint 2's USB Event Status** 0 = No event occurred on Endpoint 2. 1 = USB event occurred on Endpoint 2, check USB_EPSTS[16:14] to know which kind of USB event was occurred, cleared by writing 1 to USB_INTSTS[18] or USB_INTSTS[1].
[17]	**EPEVT1**	**Endpoint 1's USB Event Status** 0 = No event occurred on Endpoint 1. 1 = USB event occurred on Endpoint 1, check USB_EPSTS[13:11] to know which kind of USB event was occurred, cleared by writing 1 to USB_INTSTS[17] or USB_INTSTS[1].
[16]	**EPEVT0**	**Endpoint 0's USB Event Status** 0 = No event occurred on Endpoint 0. 1 = USB event occurred on Endpoint 0, check USB_EPSTS[10:8] to know which kind of USB event was occurred, cleared by writing 1 to USB_INTSTS[16] or USB_INTSTS[1].
[15:4]	**Reserved**	Reserved.
[3]	**WAKEUP_STS**	**Wake-up Interrupt Status** 0 = No Wake-up event occurred. 1 = Wake-up event occurred, cleared by writing 1 to USB_INTSTS[3].
[2]	**FLDET_STS**	**Floating Detected Interrupt Status** 0 = There is not attached/detached event in the USB. 1 = There is attached/detached event in the USB bus and it is cleared by writing 1 to USB_INTSTS[2].
[1]	**USB_STS**	**USB Event Interrupt Status** The USB event includes the Setup Token, IN Token, OUT ACK, ISO IN or ISO OUT events in the bus. 0 = No USB event occurred. 1 = USB event occurred, check EPSTS0~7[2:0] to know which kind of USB event was occurred, cleared by writing 1 to USB_INTSTS[1] or EPEVT0~7 and SETUP (USB_INTSTS[31]).
[0]	**BUS_STS**	**BUS Interrupt Status** The BUS event means that there is one of the suspense or the resume function in the bus. 0 = No BUS event occurred. 1 = Bus event occurred; check USB_ATTR[3:0] to know which kind of bus event was occurred, cleared by writing 1 to USB_INTSTS[0].

USB Device Function Address Register (USB_FADDR)

A seven-bit value uses as the address of a device on the USB BUS.

Register	Offset	R/W	Description	Reset Value
USB_FADDR	USB_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:0]	**FADDR**	The function address of the USB device.

USB Endpoint Status Register (USB_EPSTS)

Register	Offset	R/W	Description	Reset Value
USB_EPSTS	USB_BA+0x00C	R	USB Endpoint Status Register	0x0000_0000

31	30	29	28	27	26	25	24
EPSTS7			**EPSTS6**			**EPSTS5**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
EPSTS5	**EPSTS4**			**EPSTS3**			**EPSTS2**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
EPSTS2		**EPSTS1**			**EPSTS0**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OVERRUN	**Reserved**						

Bits	Description	
[31:29]	**EPSTS7**	**Endpoint 7 Bus Status** These bits are used to indicate the current status of this endpoint. 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 011 = Setup ACK. 110 = Out Packet Data1 ACK. 111 = Isochronous transfer end.
[28:26]	**EPSTS6**	**Endpoint 6 Bus Status** These bits are used to indicate the current status of this endpoint. 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 011 = Setup ACK. 110 = Out Packet Data1 ACK. 111 = Isochronous transfer end.
[25:23]	**EPSTS5**	**Endpoint 5 Bus Status** These bits are used to indicate the current status of this endpoint. 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 011 = Setup ACK. 110 = Out Packet Data1 ACK. 111 = Isochronous transfer end.

[22:20]	**EPSTS4**	**Endpoint 4 Bus Status** These bits are used to indicate the current status of this endpoint. 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 011 = Setup ACK. 110 = Out Packet Data1 ACK. 111 = Isochronous transfer end.
[19:17]	**EPSTS3**	**Endpoint 3 Bus Status** These bits are used to indicate the current status of this endpoint. 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 011 = Setup ACK. 110 = Out Packet Data1 ACK. 111 = Isochronous transfer end.
[16:14]	**EPSTS2**	**Endpoint 2 Bus Status** These bits are used to indicate the current status of this endpoint. 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 011 = Setup ACK. 110 = Out Packet Data1 ACK. 111 = Isochronous transfer end.
[13:11]	**EPSTS1**	**Endpoint 1 Bus Status** These bits are used to indicate the current status of this endpoint. 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 011 = Setup ACK. 110 = Out Packet Data1 ACK. 111 = Isochronous transfer end.
[10:8]	**EPSTS0**	**Endpoint 0 Bus Status** These bits are used to indicate the current status of this endpoint. 000 = In ACK. 001 = In NAK. 010 = Out Packet Data0 ACK. 011 = Setup ACK. 110 = Out Packet Data1 ACK. 111 = Isochronous transfer end.
[7]	**OVERRUN**	**Overrun** It indicates that the received data is over the maximum payload number or not. 0 = No overrun. 1 = Out data is more than the Max Payload in MXPLD register or the Setup data is more than 8 bytes.
[6:0]	**Reserved**	Reserved.

USB Bus Status and Attribution Register (USB_ATTR)

Register	Offset	R/W	Description	Reset Value
USB_ATTR	USB_BA+0x010	R/W	USB Bus Status and Attribution Register	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**BYTEM**	**PWRDN**	**DPPU_EN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USB_EN	**Reserved**	**RWAKEUP**	**PHY_EN**	**TIME-OUT**	**RESUME**	**SUSPEND**	**USBRST**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**BYTEM**	**CPU Access USB SRAM Size Mode Selection** 0 = Word Mode: The size of the transfer from CPU to USB SRAM can be Word only. 1 = Byte Mode: The size of the transfer from CPU to USB SRAM can be Byte only.
[9]	**PWRDN**	**Power-down PHY Transceiver, Low Active** 0 = Power-down related circuits of PHY transceiver. 1 = Turn-on related circuits of PHY transceiver.
[8]	**DPPU_EN**	**Pull-up Resistor on USB_D+ Enable Bit** 0 = the Pull-up resistor in USB_D+ bus Disabled. 1 = Pull-up resistor in USB_D+ bus Enabled.USB_D+.
[7]	**USB_EN**	**USB Controller Enable Bit** 0 = USB Controller Disabled. 1 = USB Controller Enabled.
[6]	**Reserved**	Reserved.
[5]	**RWAKEUP**	**Remote Wake-up** 0 = Release the USB bus from K state. 1 = Force USB bus to K (USB_D+ low and USB_D- high) state, used for remote wake-up.
[4]	**PHY_EN**	**PHY Transceiver Function Enable Bit** 0 = PHY transceiver function Disabled. 1 = PHY transceiver function Enabled.
[3]	**TIME-OUT**	**Time-out Status (Read Only)** 0 = No time-out. 1 = Bus no response ACK by Host more than 18 bits time in IN token.

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

Floating detection Register (USB_FLDET)

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

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**FLDET**	**Device Floating Detected** 0 = Controller isn't attached to the USB host. 1 = Controller is attached to the USB host.

Setup Token Buffer Segmentation Register (USB_BUFSEG)

For Setup token only.

Register	Offset	R/W	Description	Reset Value
USB_BUFSEG	USB_BA+0x018	R/W	Setup Token Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:3]	**BUFSEG**	**Buffer Segmentation** It Is Used to Indicate the Offset Address for the Setup Token with the USB SRAM Starting Address the Effective Starting Address Is: USB_SRAM address + {BUFSEG[8:3], 3'b000} Where the USB_SRAM address = USB_BA+0x100h. **Note:** It is used for Setup token only.
[2:0]	**Reserved**	Reserved.

Endpoint Buffer Segmentation Register (BUFSEGx) x = 0~7

Register	Offset	R/W	Description	Reset Value
USB_BUFSEG0	USB_BA+0x020	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG1	USB_BA+0x030	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG2	USB_BA+0x040	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG3	USB_BA+0x050	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG4	USB_BA+0x060	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG5	USB_BA+0x070	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG0	USB_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG1	USB_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG2	USB_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG3	USB_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG4	USB_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG5	USB_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG6	USB_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USB_BUFSEG7	USB_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:3]	**BUFSEG**	**Endpoint Buffer Segmentation** It Is Used to Indicate the Offset Address for Each Endpoint with the USB SRAM Starting Address the Effective Starting Address of the Endpoint Is: USB_SRAM address + {BUFSEG[8:3], 3'b000} Where the USB_SRAM address = USB_BA+0x100h. Refer to section 6.16.5.7 for the endpoint SRAM structure and its description.
[2:0]	**Reserved**	Reserved.

Endpoint Maximal Payload Register (USB_MXPLDx) x = 0~7

Register	Offset	R/W	Description	Reset Value
USB_MXPLD0	USB_BA+0x024	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USB_MXPLD1	USB_BA+0x034	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USB_MXPLD2	USB_BA+0x044	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USB_MXPLD3	USB_BA+0x054	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USB_MXPLD4	USB_BA+0x064	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USB_MXPLD5	USB_BA+0x074	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USB_MXPLD0	USB_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USB_MXPLD1	USB_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USB_MXPLD2	USB_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USB_MXPLD3	USB_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USB_MXPLD4	USB_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USB_MXPLD5	USB_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USB_MXPLD6	USB_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USB_MXPLD7	USB_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**MXPLD**	**Maximal Payload** It is used to define the data length which is transmitted to host (IN token) or the actual data length which is received from the host (OUT token). It is also used to indicate that the endpoint is ready to be transmitted out IN token or received in OUT token. (1) When the register is written by CPU, For IN token, the value of MXPLD is used to define the data length to be transmitted and indicate the data buffer is ready.

		For OUT token, it means that the controller is ready to receive data from the host and the value of MXPLD is the maximal data length comes from host. (2) When the register is read by CPU, For IN token, the value of MXPLD indicates the data length be transmitted to host For OUT token, the value of MXPLD indicates the actual data length received from host. **Note:** Once MXPLD is written, the data packets will be transmitted/received immediately after IN/OUT token arrived.

Endpoint Configuration Register (USB_CFGx) x = 0~7

Register	Offset	R/W	Description	Reset Value
USB_CFG0	USB_BA+0x028	R/W	Endpoint 0 Configuration Register	0x0000_0000
USB_CFG1	USB_BA+0x038	R/W	Endpoint 1 Configuration Register	0x0000_0000
USB_CFG2	USB_BA+0x048	R/W	Endpoint 2 Configuration Register	0x0000_0000
USB_CFG3	USB_BA+0x058	R/W	Endpoint 3 Configuration Register	0x0000_0000
USB_CFG4	USB_BA+0x068	R/W	Endpoint 4 Configuration Register	0x0000_0000
USB_CFG5	USB_BA+0x078	R/W	Endpoint 5 Configuration Register	0x0000_0000
USB_CFG0	USB_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USB_CFG1	USB_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USB_CFG2	USB_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USB_CFG3	USB_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USB_CFG4	USB_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USB_CFG5	USB_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USB_CFG6	USB_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USB_CFG7	USB_BA+0x578	R/W	Endpoint 7 Configuration Register	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**CSTALL**	**Clear STALL Response** 0 = Device Disabled to clear the STALL handshake in the setup stage. 1 = Clear the device to respond STALL handshake in the setup stage.
[8]	**Reserved**	Reserved.

[7]	**DSQ_SYNC**	**Data Sequence Synchronization** 0 = DATA0 PID. 1 = DATA1 PID. It is used to specify the DATA0 or DATA1 PID in the following IN token transaction. Hardware will toggle automatically in IN token base on the bit.
[6:5]	**STATE**	**Endpoint STATE** 00 = Endpoint Disabled. 01 = Out endpoint. 10 = IN endpoint. 11 = Undefined.
[4]	**ISOCH**	**Isochronous Endpoint** This bit is used to set the endpoint as Isochronous endpoint, no handshaking. 0 = No Isochronous endpoint. 1 = Isochronous endpoint.
[3:0]	**EP_NUM**	**Endpoint Number** These bits are used to define the endpoint number of the current endpoint.

Endpoint Set Stall and Clear In/Out Ready Control Register (USB_CFGPx) x = 0~7

Register	Offset	R/W	Description	Reset Value
USB_CFGP0	USB_BA+0x02C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP1	USB_BA+0x03C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP2	USB_BA+0x04C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP3	USB_BA+0x05C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP4	USB_BA+0x06C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP5	USB_BA+0x07C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP0	USB_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP1	USB_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP2	USB_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP3	USB_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP4	USB_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP5	USB_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP6	USB_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USB_CFGP7	USB_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**SSTALL**	**Set STALL** 0 = Device Disabled to respond STALL. 1 = Set the device to respond STALL automatically.
[0]	**CLRRDY**	**Clear Ready** When the MXPLD register is set by user, it means that the endpoint is ready to transmit or receive data. If the user wants to turn off this transaction before the transaction start, user can set this bit to 1 to turn it off and it is automatically cleared

		to 0. For IN token: 0 = No effect. 1 = Clear the IN token had ready to transmit the data to USB host. For OUT token: 0 = No effect. 1 = Clear the OUT token had ready to receive the data from USB host. This bit is written 1 only and is always 0 when it was read back.

USB Drive SE0 Register (USB_DRVSE0)

Register	Offset	R/W	Description	Reset Value
USB_DRVSE0	USB_BA+0x090	R/W	Force USB PHY to drive SE0	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**DRVSE0**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**DRVSE0**	**Drive Single Ended Zero in USB Bus** The Single Ended Zero (SE0) is when both lines (USB_D+ and USB_D-) are being pulled low. 0 = None. 1 = Force USB PHY transceiver to drive SE0.

USB PDMA Control Register (USB_PDMA)

Register	Offset	R/W	Description	Reset Value
USB_PDMA	USB_BA+0x0A4	R/W	USB PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**PDMA_EN**	**PDMA Function Enable Bit** 0 = The PDMA function is not active. 1 = The PDMA function in USB is active. This bit will be automatically cleared after PDMA transfer done.
[0]	**PDMA_RW**	**PDMA_RW** 0 = The PDMA will move data from memory to USB buffer. 1 = The PDMA will move data from USB buffer to memory.

6.17 Analog-to-Digital Converter (ADC)

6.17.1 Overview

NuMicro® NUC123 Series contains one 10-bit successive approximation analog-to-digital converters (SAR A/D converter) with 8 input channels. The A/D converter supports three operation modes: single, single-cycle scan and continuous scan mode. The A/D converters can be started by software, PWM center-aligned trigger and external STADC pin.

6.17.2 Features

- Conversion range : 0 to AV_{DD}
- 10-bit resolution and 8-bit accuracy is guaranteed
- Up to 8 single-end analog input channels
- Maximum ADC clock frequency as 6 MHz (NUC123xxxANx Only)
- Maximum ADC clock frequency as 3 MHz (NUC123xxxAEx Only)
- Up to 166 kSPS (Samples Per Second) conversion rate (NUC123xxxANx Only)
- Up to 200 kSPS (Samples Per Second) conversion rate (NUC123xxxAEx Only)
- Three operating modes
  - Single mode: A/D conversion is performed one time on a specified channel
  - Single-cycle Scan mode: A/D conversion is performed one cycle on all specified channels with the sequence from the lowest numbered channel to the highest numbered channel
  - Continuous Scan mode: A/D converter continuously performs Single-cycle scan mode until software stops A/D conversion
- A/D conversion started by:
  - Software writes 1 to ADST bit
  - External pin STADC (PB.8)
  - PWM center-aligned trigger
- Supports 8 data registers to stored conversion result with valid and overrun indicators
- Supports 2 sets of digital comparators to monitor conversion result of specified channel and to generate an interrupt when conversion result matches comparison condition
- Channel 7 supports 2 input sources: external analog voltage and internal band-gap voltage
- Supports PDMA transfer

6.17.3 Block Diagram

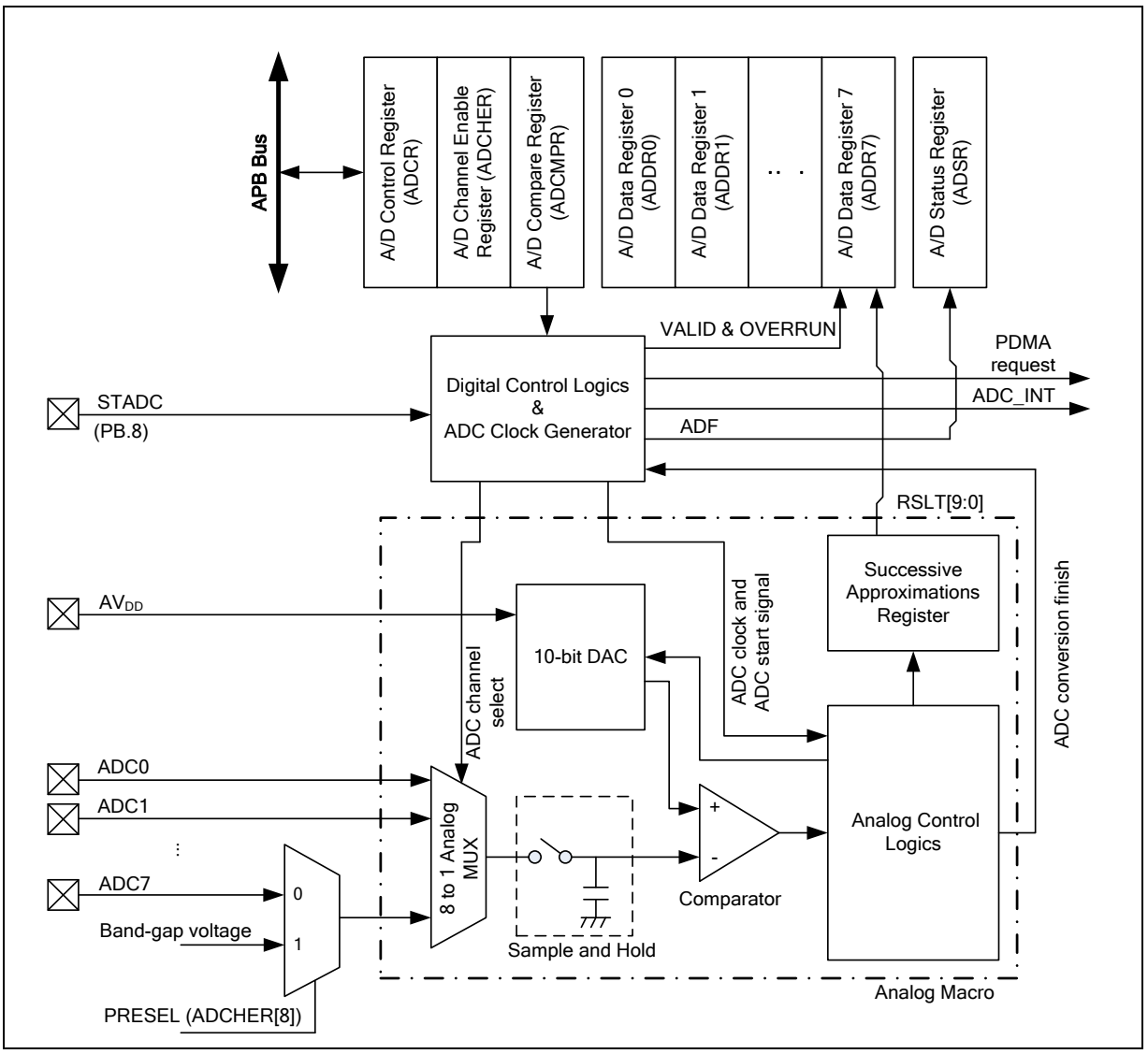


Figure 6-124 ADC Controller Block Diagram

6.17.4 Basic Configuration

The ADC pins are configured in GPD_MFP and GPF_MFP registers. NUC123xxxAEx provides the alternative of configuring the ADC pins in GPD_MFPL and GPF_MFPL registers. (For NUC123xxxAEx, if GPD_MFPL and GPF_MFPL are used as pin multi-function setting, the GPD_MFP and GPF_MFP will become invalid).

The ADC controller clock source is enabled by ADC_EN (APBCLK[28]) bit. If the ADC pins are configured to ADC analog input by setting the corresponding multiple function pin control registers, software should disable the digital input path of these ADC analog input pins by setting the corresponding GPIO digital input path control registers (GPIOD_OFFD and GPIOF_OFFD).

6.17.5 Functional Description

The A/D converter is operated by successive approximation with 10-bit resolution. The ADC has three

operation modes: single mode, single-cycle scan mode and continuous scan mode. When changing the operating mode or analog input channel, in order to prevent incorrect operation, software must clear ADST (ADCR[11]) bit to 0.

6.17.5.1 *ADC Engine Clock Generator*

The maximum sampling rate is up to 200 kSPS. The ADC peripheral clock has four clock sources selected by ADC_S (CLKSEL1[3:2]), the ADC clock frequency is divided by an 8-bit prescaler with the formula:

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

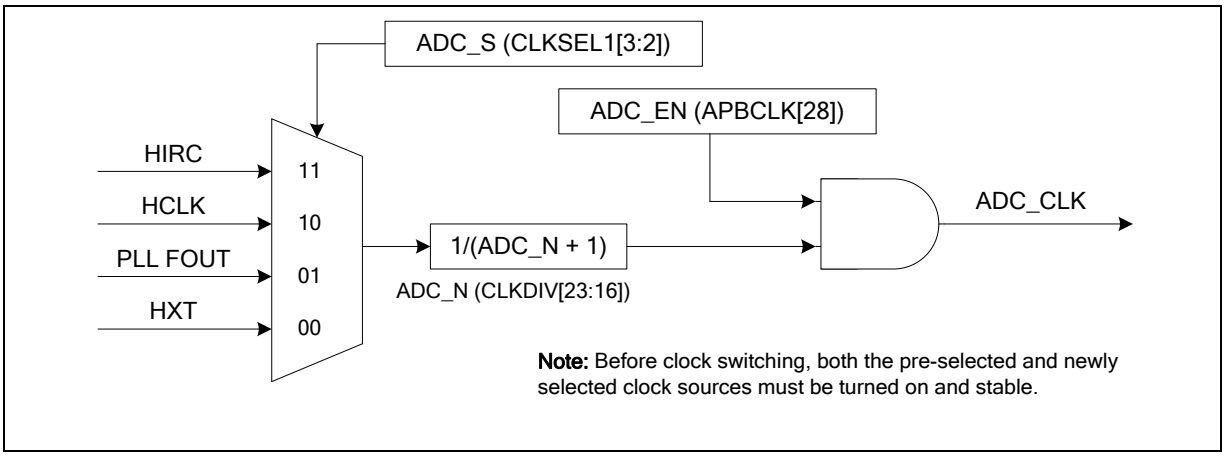


Figure 6-125 ADC Clock Control

6.17.5.2 *Single Mode*

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

1. A/D conversion will be started when the ADST (ADCR[11]) is set to 1 by software or external trigger input or by PWM trigger.
2. When A/D conversion is finished, the result is stored in the A/D data register corresponding to the channel.
3. The ADF (ADSR[0]) will be set to 1. If the ADIE (ADCR[1]) is set to 1, the ADC interrupt will be asserted.
4. The ADST (ADCR[11]) bit remains 1 during A/D conversion. When A/D conversion ends, the ADST (ADCR[11]) bit is automatically cleared to 0 and the A/D converter enters idle state.

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

Note: when ADST is cleared to 0 by hardware automatically in single mode, user needs to wait one ADC_CLK cycle for the next A/D conversion.

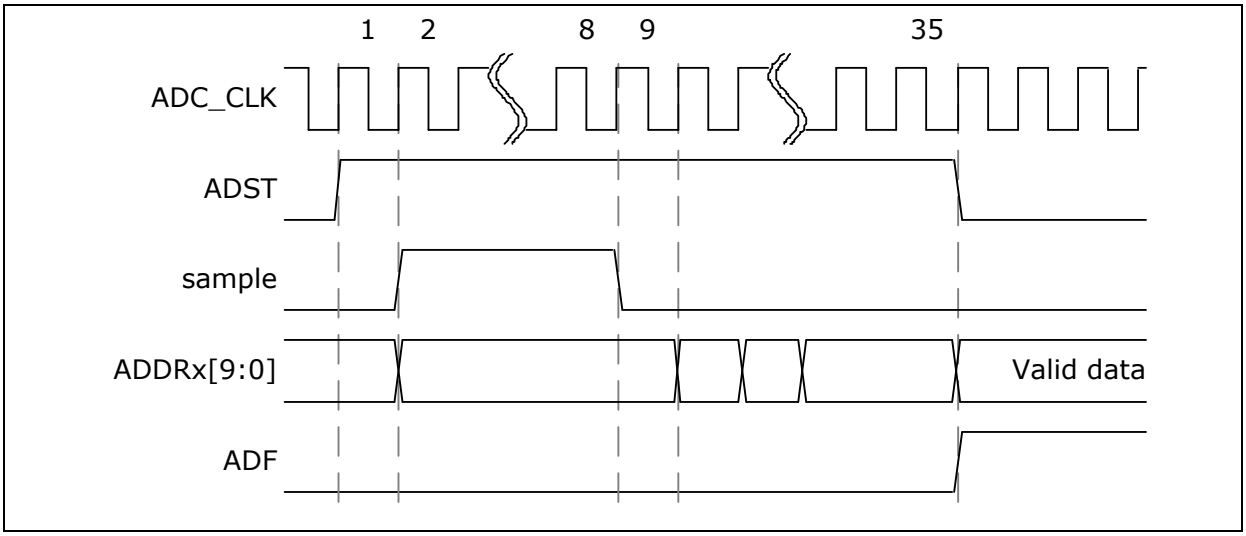


Figure 6-126 Single Mode Conversion Timing Diagram (for NUC123xxxANx)

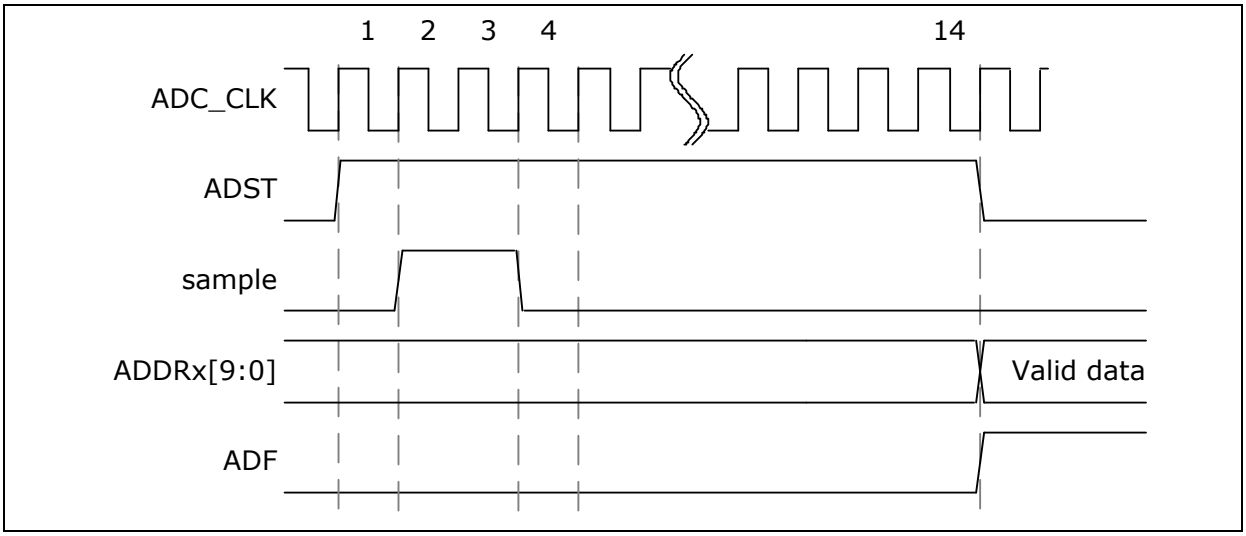


Figure 6-127 Single Mode Conversion Timing Diagram (for NUC123xxxAEx)

6.17.5.3 *Single-Cycle Scan Mode*

In Single-cycle Scan mode, A/D conversion will sample and convert the specified channels once in the sequence from the lowest number enabled channel to the highest number enabled channel.

1. When the ADST (ADCR[11]) is set to 1 by software or external trigger input or by PWM trigger, A/D conversion starts on the channel with the lowest number.
2. When A/D conversion for each enabled channel is completed, the result is sequentially transferred to the A/D data register corresponding to each channel.
3. When the conversions of all the enabled channels are completed, the ADF (ADSR[0]) is set to 1. If the ADC interrupt function is enabled, the ADC interrupt occurs.

After A/D conversion ends, the ADST (ADCR[11]) bit is automatically cleared to 0 and the A/D converter enters idle state. If ADST (ADCR[11]) is cleared to 0 before all enabled ADC channels conversion done, ADC controller will finish current conversion and save the result to the ADDRx of the

current conversion channel. In this case, ADF (ADSR[0]) bit will not be set.

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

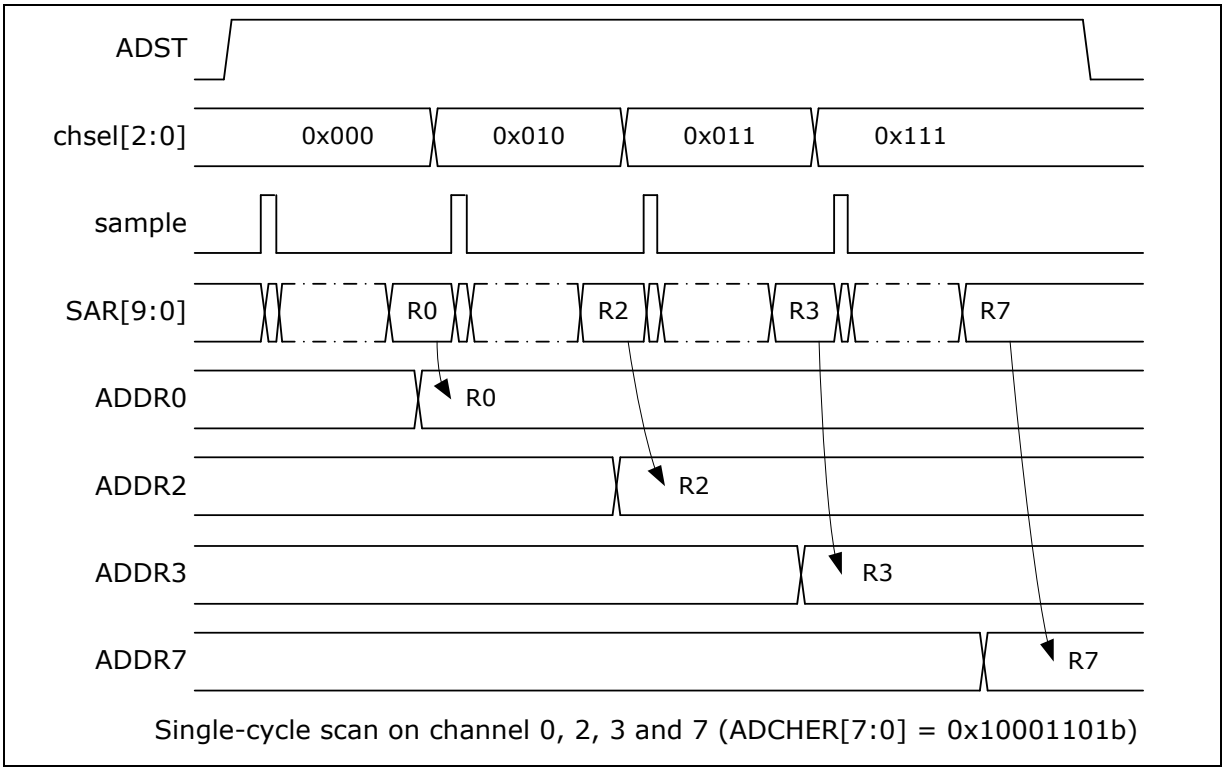


Figure 6-128 Single-Cycle Scan on Enabled Channels Timing Diagram

6.17.5.4 *Continuous Scan Mode*

In continuous scan mode, A/D conversion is performed sequentially on the specified channels that enabled by CHEN (ADCHER[7:0]) register (maximum 8 channels for ADC). The operations are as follows:

1. When the ADST (ADCR[11]) is set to 1 by software or external trigger input or by PWM trigger, A/D conversion starts on the channel with the lowest number.
2. When A/D conversion for each enabled channel is completed, the result of each enabled channel is stored in the A/D data register corresponding to each enabled channel.
3. When A/D converter completes the conversions of all enabled channels sequentially, the ADF (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 lowest number will start again if software has not cleared the ADST (ADCR[11]) bit.
4. As long as the ADST (ADCR[11]) bit remains at 1, the step 2 ~ 3 will be repeated. When ADST (ADCR[11]) is cleared to 0, ADC controller will stop conversion. In this case, ADF (ADSR[0]) bit will not be set.

An example timing diagram for continuous scan on enabled channels (0, 2, 3 and 7) is shown below:

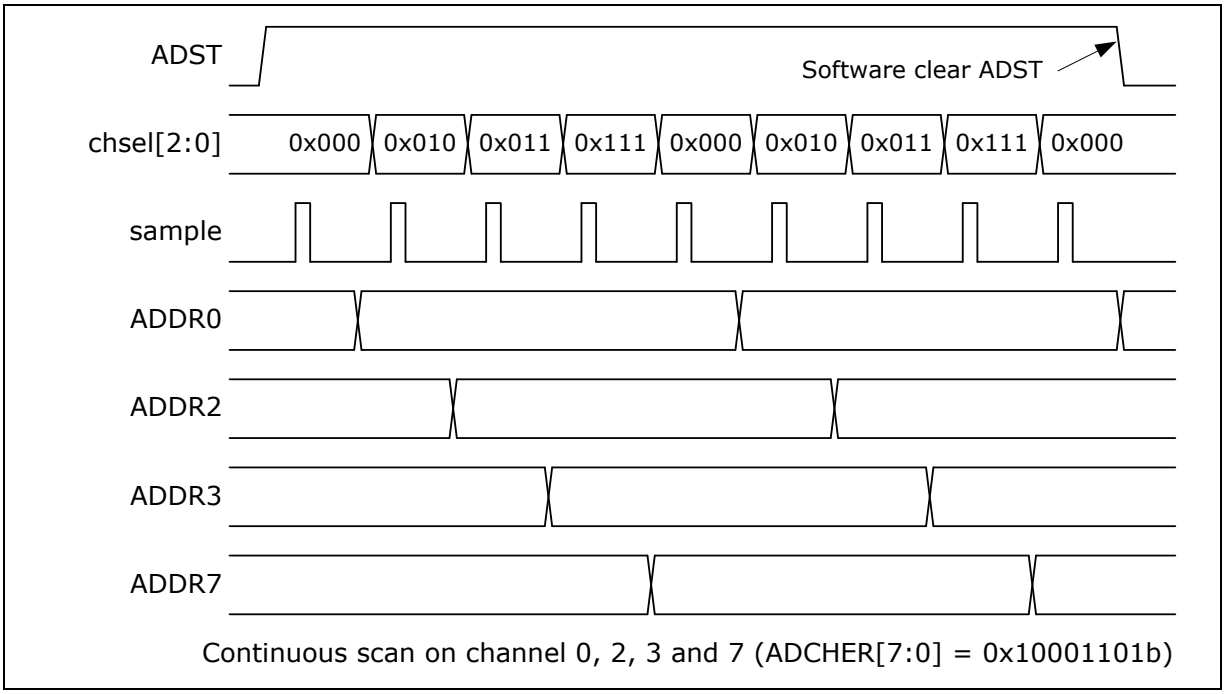


Figure 6-129 Continuous Scan on Enabled Channels Timing Diagram

6.17.5.5 *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 (ADCR[8]) is set to high to enable ADC external trigger function, setting the TRGS[1:0] bits to 00b is to select external trigger input from the STADC pin. Software can set TRGCOND[1:0] to select trigger condition is falling/rising edge or low/high level. If level trigger condition is selected, the STADC pin must be kept at defined state at least 8 PCLKs. The ADST (ADCR[11]) bit will be set to 1 at the 9th PCLK and start to conversion. Conversion is continuous if external trigger input is kept at active state in level trigger mode. It is stopped only when external condition trigger condition disappears. If edge trigger condition is selected, the high and low state must be kept at least 4 PLCKs. Pulse that is shorter than this specification will be ignored.

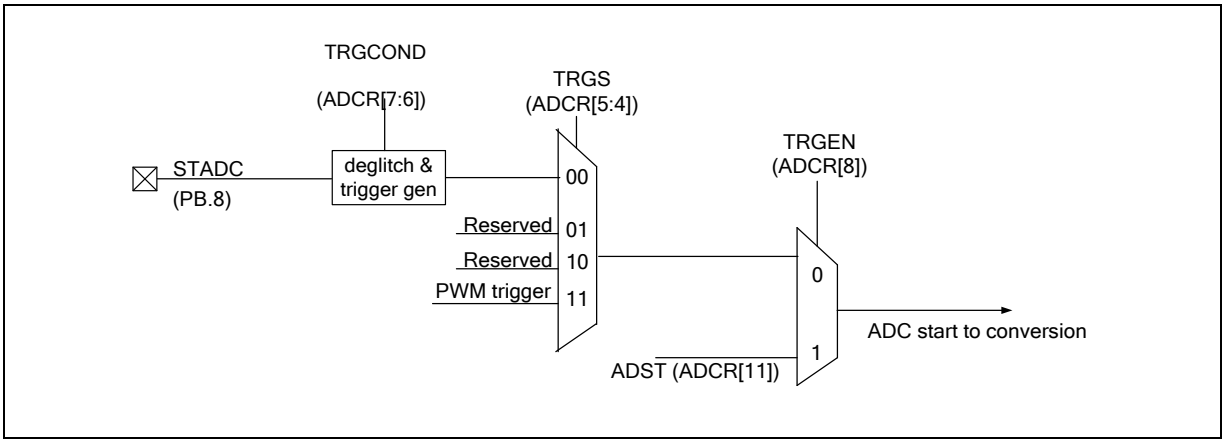


Figure 6-130 ADC Start Conversion Conditions

6.17.5.6 *Internal Reference Voltage*

The band-gap voltage reference (VBG) is an internal fixed reference voltage regardless of power supply variations. The VBG output is internally connected to ADC channel 7 source multiplexer and Analog Comparators's (ACMP) negative input side.

For battery power detection application, user can use the VBG as ADC input channel such that user can convert the A/D conversion result to calculate AVDD with following formula.

AVDD = ((2 ^ N) / R) * VBG

N: ADC resolution

R: A/D conversion result

VBG: Band-gap voltage

The block diagram is shown as Figure 6-131.

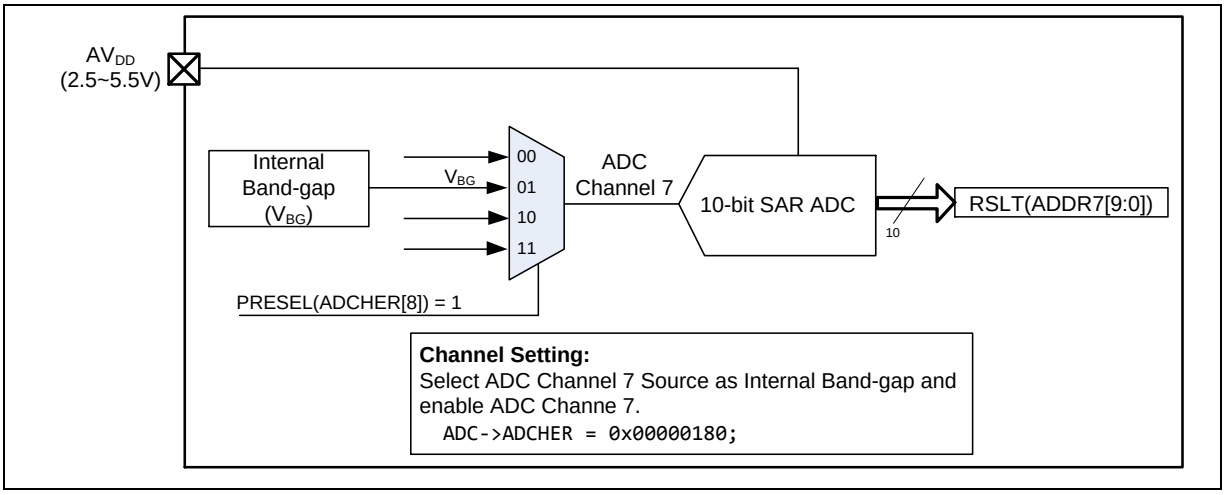


Figure 6-131 VBG for Measuring AVDD Application Block Diagram

For example, the VBG typical value is 1.26 V, the ADC is 10-bit resolution, select VBG as ADC channel 7 input source, and enable ADC channel 7. Then trigger ADC to converse.

If the A/D conversion result is 235:

N = 10

R = 235

VBG = 1.26 V

AVDD = ((2 ^ 10) / 235) * 1.26 = (1024 / 235) * 1.26 = 5.49 V

If the A/D conversion result is 512:

AVDD = ((2 ^ 10) / 512) * 1.26 = (1024 / 512) * 1.26= 2.52 V

6.17.5.7 *Conversion Result Monitor by Compare Function*

ADC controller provides two compare registers, ADCMPR0 and ADCMPR1, to monitor maximum two specified channels. Software can select which channel to be monitored by set CMPCH (ADCMPRx[5:0]). CMPCOND (ADCMPRx[2]) is used to determine the compare condition. If

CMPCOND (ADCMPRx[2]) bit is cleared to 0, the internal match counter will increase one when the conversion result is less than the value specified in CMPD[9:0]; 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[9:0]. When the conversion of the channel specified by CMPCH is completed, the comparing action will be triggered one time automatically. When the compare result meets the setting, compare match counter will increase 1, otherwise, the compare match counter will be cleared to 0. When counter value reach the setting of (CMPMATCNT+1) then CMPFx (ADSR[2:1]) bit will be set to 1. If CMPIE (ADCR[1]) is set, an ADC interrupt request is generated. Software can use it to monitor the external analog input pin voltage transition in scan mode without imposing a load on software. Detail logic diagram is shown below.

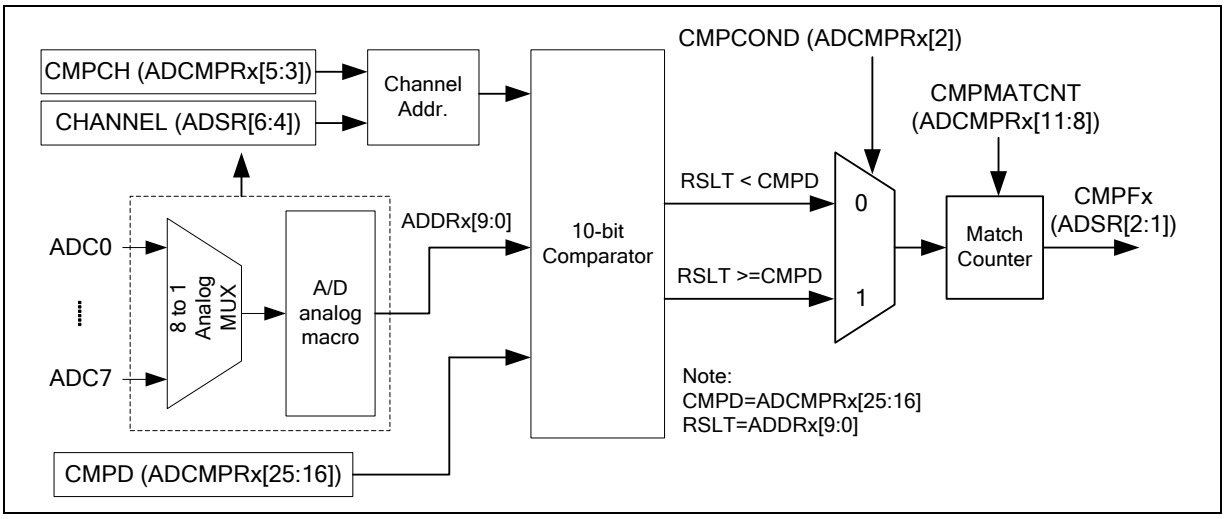


Figure 6-132 A/D Conversion Result Monitor Logics Diagram

6.17.5.8 *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 (ADSR[0]), will be set to 1. The CMPF0 (ADSR[1]) and CMPF1 (ADSR[2]) are the compare flags of compare function. When the conversion result meets the settings of ADCMPR0/1, the corresponding flag will be set to 1. When one of the flags, ADF (ADSR[0]), CMPF0 (ADSR[1]) and CMPF1 (ADSR[2]), is set to 1 and the corresponding interrupt enable bit, ADIE (ADCR[1]) and CMPIE (ADCMPRx[1]) is set to 1, the ADC interrupt will be asserted. Software can clear the flag to revoke the interrupt request.

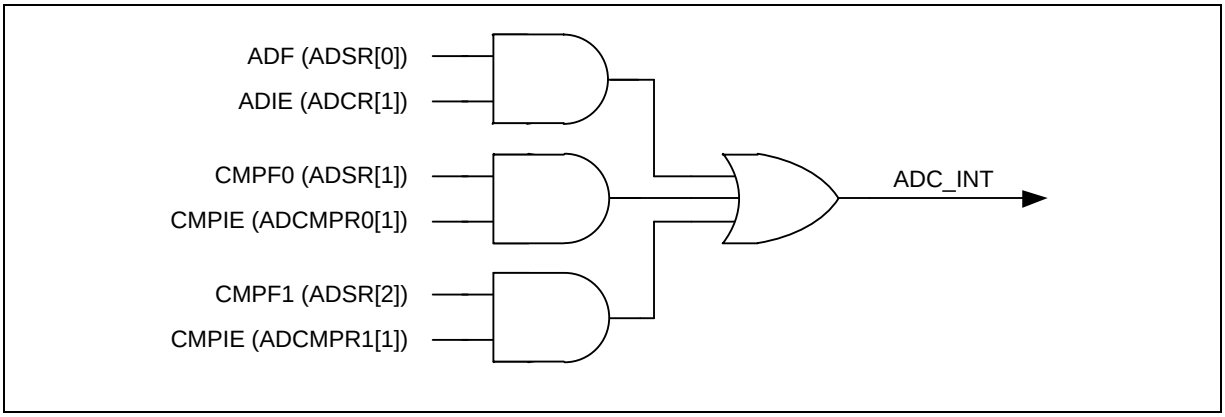


Figure 6-133 A/D Controller Interrupt

6.17.5.9 *Peripheral DMA Request*

When A/D conversion is finished, the conversion result will be loaded into ADDR register and VALID bit will be set to 1. If the PTEN bit of ADCR is set, ADC controller will generate a request to PDMA. User can use PDMA to transfer the conversion results to a user-specified memory space without CPU's intervention. The source address of PDMA operation is fixed at ADPDMA, no matter what channels was selected. When PDMA is transferring the conversion result, ADC will continue converting the next selected channel if the operation mode of ADC is single scan mode or continuous scan mode. User can monitor current PDMA transfer data through reading ADPDMA register. If ADC completes the conversion of a selected channel and the last conversion result of the same channel has not been transferred by PDMA, OVERRUN bit of the corresponding channel will be set and the last ADC conversion result will be overwrote by the new ADC conversion result. PDMA will transfer the latest data of selected channels to the user-specified destination address.

6.17.5.10 *Conversion Result*

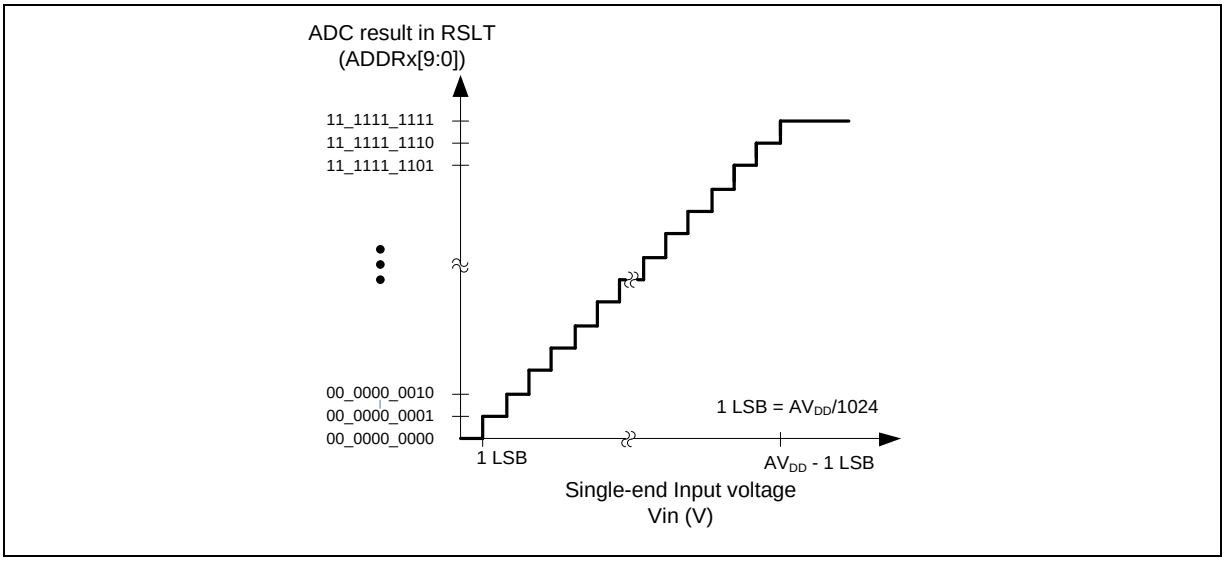


Figure 6-134 ADC Single-end Input Conversion Voltage and Conversion Result Mapping Diagram

6.17.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
ADC_BA = 0x400E_0000				
ADDR0	ADC_BA+0x00	R	A/D Data Register 0	0x0000_0000
ADDR1	ADC_BA+0x04	R	A/D Data Register 1	0x0000_0000
ADDR2	ADC_BA+0x08	R	A/D Data Register 2	0x0000_0000
ADDR3	ADC_BA+0x0C	R	A/D Data Register 3	0x0000_0000
ADDR4	ADC_BA+0x10	R	A/D Data Register 4	0x0000_0000
ADDR5	ADC_BA+0x14	R	A/D Data Register 5	0x0000_0000
ADDR6	ADC_BA+0x18	R	A/D Data Register 6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	A/D Data Register 7	0x0000_0000
ADCR	ADC_BA+0x20	R/W	A/D Control Register	0x0000_0000
ADCHER	ADC_BA+0x24	R/W	A/D Channel Enable Register	0x0000_0000
ADCMPR0	ADC_BA+0x28	R/W	A/D Compare Register 0	0x0000_0000
ADCMPR1	ADC_BA+0x2C	R/W	A/D Compare Register 1	0x0000_0000
ADSR	ADC_BA+0x30	R/W	A/D Status Register	0x0000_0000
ADPDMA	ADC_BA+0x40	R	ADC PDMA Current Transfer Data	0x0000_0000

6.17.7 Register Description

A/D Data Registers (ADDR0 ~ ADDR7)

Register	Offset	R/W	Description	Reset Value
ADDR0	ADC_BA+0x00	R	A/D Data Register 0	0x0000_0000
ADDR1	ADC_BA+0x04	R	A/D Data Register 1	0x0000_0000
ADDR2	ADC_BA+0x08	R	A/D Data Register 2	0x0000_0000
ADDR3	ADC_BA+0x0C	R	A/D Data Register 3	0x0000_0000
ADDR4	ADC_BA+0x10	R	A/D Data Register 4	0x0000_0000
ADDR5	ADC_BA+0x14	R	A/D Data Register 5	0x0000_0000
ADDR6	ADC_BA+0x18	R	A/D Data Register 6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	A/D Data Register 7	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**Valid Flag (Read Only)** This bit is set to 1 when the corresponding channel analog input conversion is completed and cleared by hardware after ADDR register is read. 0 = Data in RSLT[9:0] is not valid. 1 = Data in RSLT[9:0] is valid. **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)** If converted data in RSLT (ADDR[9:0]) has not been read before new conversion result is loaded to this register, OVERRUN (ADDR[16]) is set to 1 and previous conversion result is gone. It is cleared by hardware after ADDR register is read. 0 = Data in RSLT[9:0] is recent conversion result. 1 = Data in RSLT[9:0] is overwritten.
[15:10]	**Reserved**	Reserved.

[9:0]	**RSLT**	**A/D Conversion Result** This field contains 10 bits conversion result of ADC.

A/D Control Register (ADCR)

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

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

Bits	Description	
[30:12]	**Reserved**	Reserved.
[11]	**ADST**	**A/D Conversion Start** ADST bit can be set to 1 from three sources: software, STADC pin and PWM output. ADST will be cleared to 0 by hardware automatically at the ends of single mode and single-cycle scan mode. In continuous scan mode, A/D conversion is continuously performed until software writes 0 to this bit. 0 = Conversion stopped and A/D converter entering Idle state. 1 = Conversion started. **Note:** when ADST is cleared to 0 by hardware automatically in single mode, user needs to wait one ADC_CLK cycle for the next A/D conversion.
[9]	**PTEN**	**PDMA Transfer Enable Bit** When A/D conversion is completed, the converted data is loaded into ADDR 0~7, software can enable this bit to generate a PDMA data transfer request. When PTEN = 1, software must set ADIE = 0 to disable interrupt. 0 = PDMA data transfer Disabled. 1 = PDMA data transfer Enabled.
[8]	**TRGEN**	**External Trigger Enable Bit** Enable or disable triggering of A/D conversion by external STADC pin or by PWM trigger. 0= External trigger Disabled. 1= External trigger Enabled. **Note:** ADC external trigger function is only supported in Single-cycle Scan mode. If hardware trigger is enabled, the ADST bit can be set to 1 by the selected hardware trigger source.

[7:6]	**TRGCOND**	**External Trigger Condition** These two bits decide external pin STADC trigger event is level or edge. The signal must be kept at stable state at least 8 PCLKs for level trigger and 4 PCLKs at high and low state for edge trigger. 00 = Low level. 01 = High level. 10 = Falling edge. 11 = Rising edge.
[5:4]	**TRGS**	**Hardware Trigger Source** 00 = A/D conversion is started by external STADC pin. 11 = A/D conversion is started by PWM center-aligned trigger. Others = Reserved. **Note:** TRGEN (ADCR[8]) and ADST (ADCR[11]) shall be cleared to 0 before changing TRGS.
[3:2]	**ADMD**	**A/D Converter Operation Mode** 00 = Single conversion. 01 = Reserved. 10 = Single-cycle scan. 11 = Continuous scan. **Note:** A/D conversion shall be stopped before changing the operation mode.
[1]	**ADIE**	**A/D Interrupt Enable Bit** 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 Bit** Before starting A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit for saving power consumption. 0 = A/D converter Disabled. 1 = A/D converter Enabled.

A/D Channel Enable Register (ADCHER)

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

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**PRESEL**	**Analog Input Channel 7 Selection** 0 = External analog input. 1 = Internal band-gap voltage. **Note:** When software selects the band-gap voltage as the analog input source of ADC channel 7, ADC clock rate needs to be limited to lower than 300 kHz.
[7:0]	**CHEN**	**Analog Input Channel Enable Bit** Set CHEN[7:0] to enable the corresponding analog input channel 7 ~ 0. 0 = Channel Disabled. 1 = Channel Enabled.

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

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

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

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CMPD**	**Comparison Data** The 10-bit data is used to compare with conversion result of specified channel.
[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 (ADCMPRx[2]), the internal match counter will increase 1, otherwise, the compare match counter will be cleared to 0. When the internal counter reaches the value to (CMPMATCNT +1), CMPFx (ADSR[2:1]) 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 10-bit A/D conversion result is less than the 10-bit CMPD (ADCMPRx[25:16]), the internal match counter will increase one. 1 = Set the compare condition as that when a 10-bit A/D conversion result is greater or equal to the 10-bit CMPD (ADCMPRx[25:16]), the internal match counter will increase one. **Note:** When the internal counter reaches the value to (CMPMATCNT +1), the CMPFx bit will be set.

[1]	**CMPIE**	**Compare Interrupt Enable Bit** If the compare function is enabled and the compare condition matches the setting of CMPCOND (ADCMPRx[2]) and CMPMATCNT (ADCMPRx[11:9]), CMPFx (ADSR[2:1]) will be set. In the meanwhile, if CMPIE (ADCMPRx[1]) is set to 1, a compare interrupt request is generated. 0 = Compare function interrupt Disabled. 1 = Compare function interrupt Enabled.
[0]	**CMPEN**	**Compare Enable Bit** Set this bit to 1 to enable ADC controller to compare CMPD (ADCMPRx[25:16]) with specified channel conversion result when converted data is loaded into ADDRx 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							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VALID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CHANNEL**			**BUSY**	**CMPF1**	**CMPF0**	**ADF**

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**OVERRUN**	**Overrun Flag (Read Only)** OVERRUN[7:0] is a mirror of the OVERRUN bits in ADDR7[16] ~ ADDR0[16].
[15:8]	**VALID**	**Data Valid Flag (Read Only)** VALID[7:0] is a mirror of the VALID bits in ADDR7[17] ~ ADDR0[17].
[7]	**Reserved**	Reserved.
[6:4]	**CHANNEL**	**Current Conversion Channel (Read Only)** This field reflects current conversion channel when BUSY (ADSR[3]) is 1. When BUSY (ADSR[3]) is 0, it shows number of the next converted channel.
[3]	**BUSY**	**BUSY/IDLE (Read Only)** 0 = A/D converter is in Idle state. 1 = A/D converter is busy at conversion. This bit is a mirror of ADST bit in ADCR.
[2]	**CMPF1**	**Compare Flag** When the selected channel A/D conversion result meets setting condition in ADCMPR1 then this bit is set to 1. It is cleared by writing 1. 0 = Conversion result in ADDR does not meet ADCMPR1 setting. 1 = Conversion result in ADDR meets ADCMPR1 setting.
[1]	**CMPF0**	**Compare Flag** When the selected channel A/D conversion result meets setting condition in ADCMPR0 then this bit is set to 1. And it is cleared by writing 1. 0 = Conversion result in ADDR does not meet ADCMPR0 setting. 1 = Conversion result in ADDR meets ADCMPR0 setting.

[0]	**ADF**	**A/D Conversion End Flag** A status flag that indicates the end of A/D conversion. ADF is set to 1 at these two conditions: 1. When A/D conversion ends in Single mode. 2. When A/D conversion ends on all specified channels in Scan mode. This flag can be cleared by writing 1 to itself.

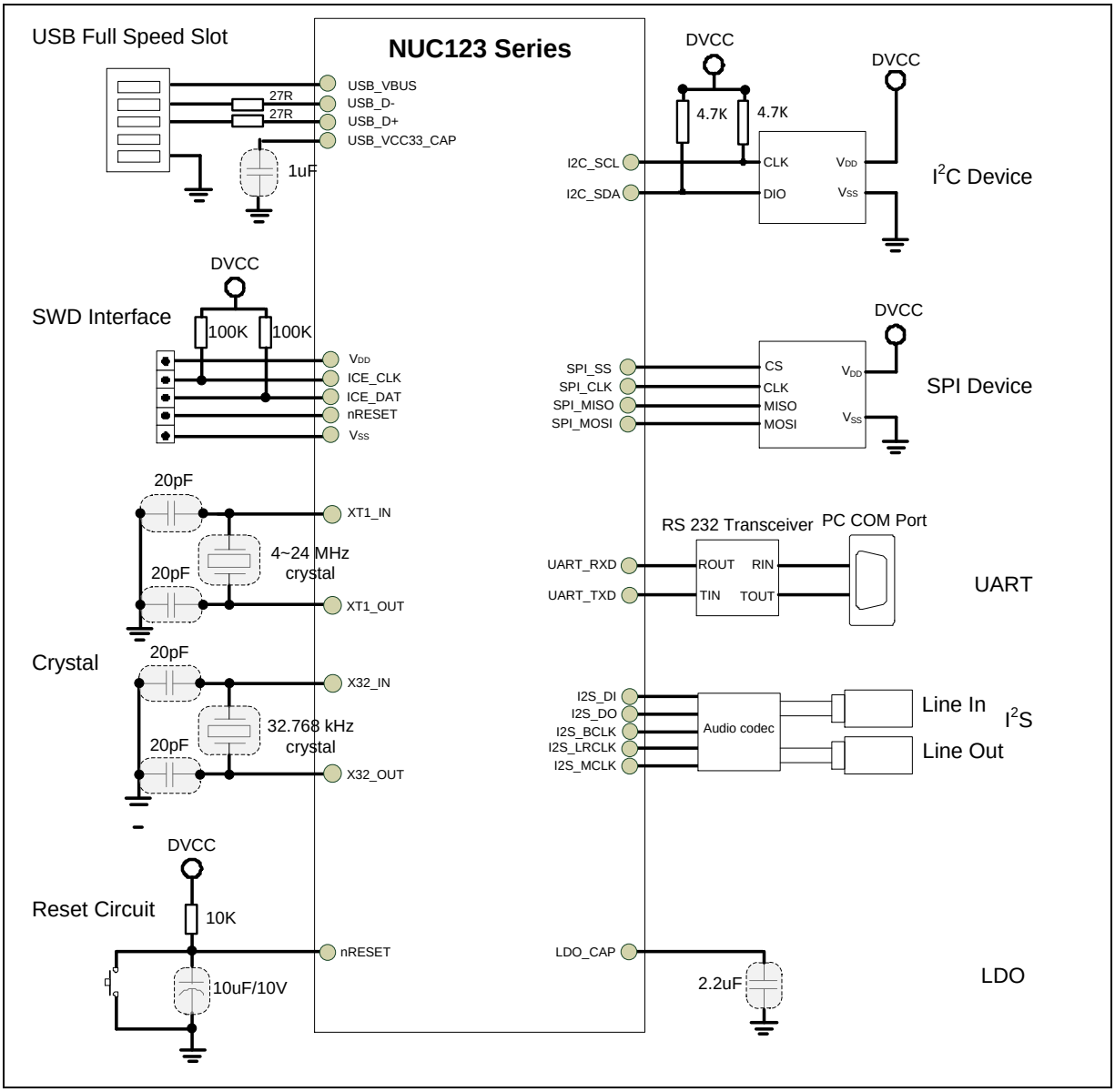
A/D PDMA Current Transfer Data Register (ADPDMA)

Register	Offset	R/W	Description	Reset Value
ADPDMA	ADC_BA+0x40	R	A/D PDMA current transfer data Register	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**AD_PDMA**	**ADC PDMA Current Transfer Data Register (Read Only)** When transferring A/D conversion result with PDMA, software can read this register to monitor current PDMA transfer data.

7 PERIPHERAL APPLICATION SCHEME



Note 1: It is recommended to use 100 kΩ pull-up resistor on both ICE_DAT and ICE_CLK pin.

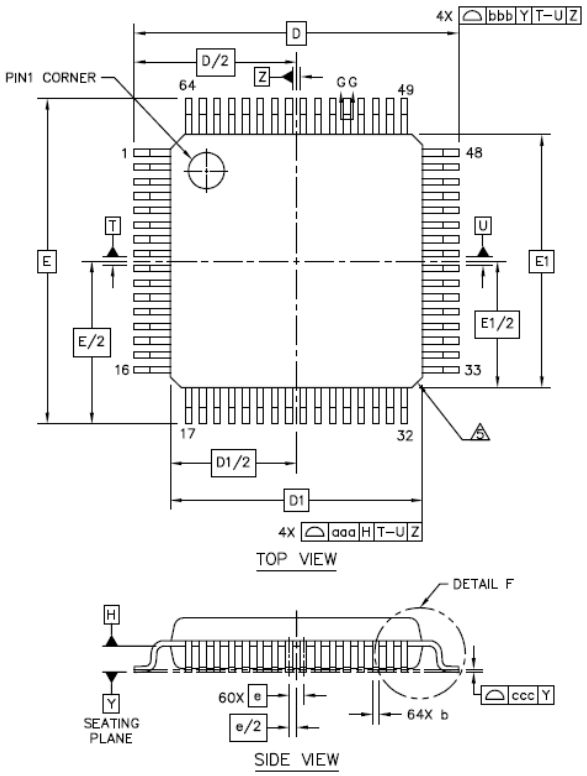
Note 2: It is recommended to use 10 kΩ pull-up resistor and 10 μF capacitor on nRESET pin.

8 ELECTRICAL CHARACTERISTICS

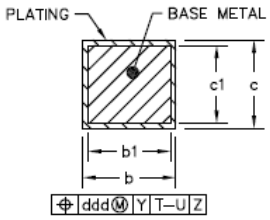
For information on the NUC123 series electrical characteristics, please refer to NuMicro® NUC123 Series Datasheet.

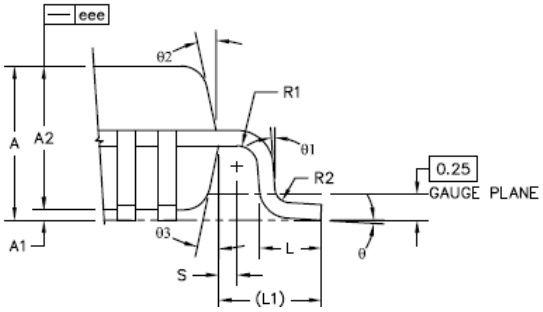
9 PACKAGE DIMENSIONS

9.1 64L LQFP (7x7x1.4 mm footprint 2.0 mm)

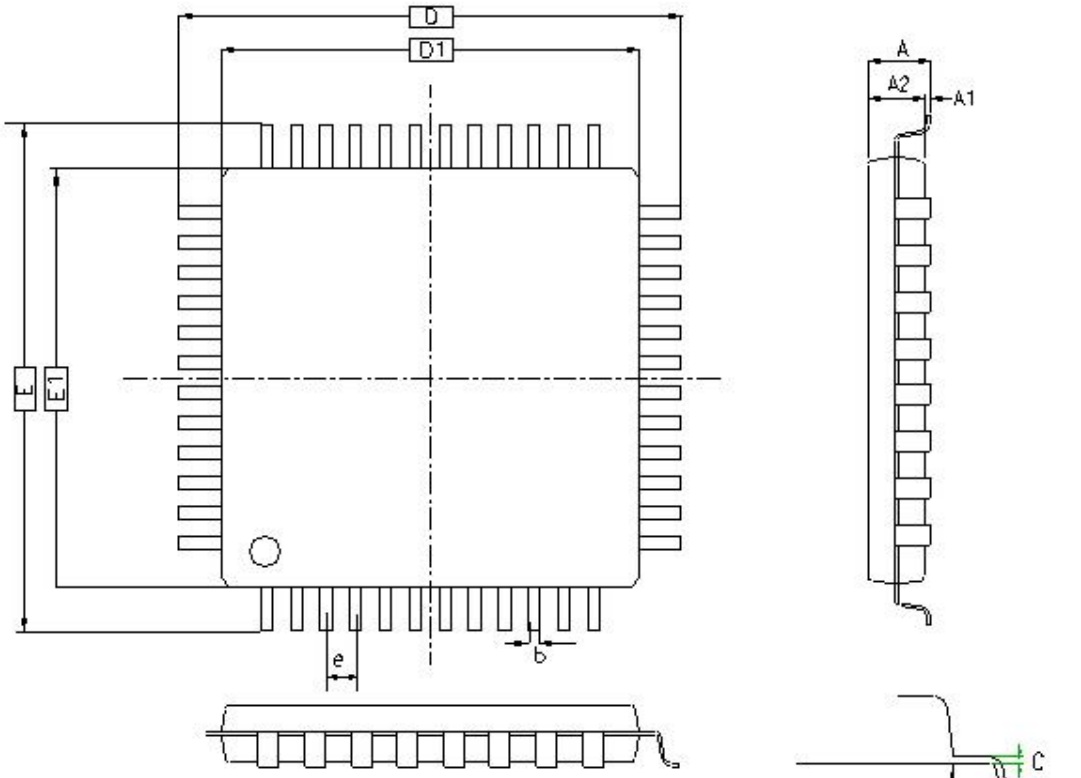


		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	---	---	1.6
STAND OFF		A1	0.05	---	0.15
MOLD THICKNESS		A2	1.35	1.4	1.45
LEAD WIDTH(PLATING)		b	0.13	0.18	0.23
LEAD WIDTH		b1	0.13	0.16	0.19
L/F THICKNESS(PLATING)		c	0.09	---	0.2
L/F THICKNESS		c1	0.09	---	0.16
BODY SIZE	X	D	9 BSC		
	Y	E	9 BSC		
	X	D1	7 BSC		
	Y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





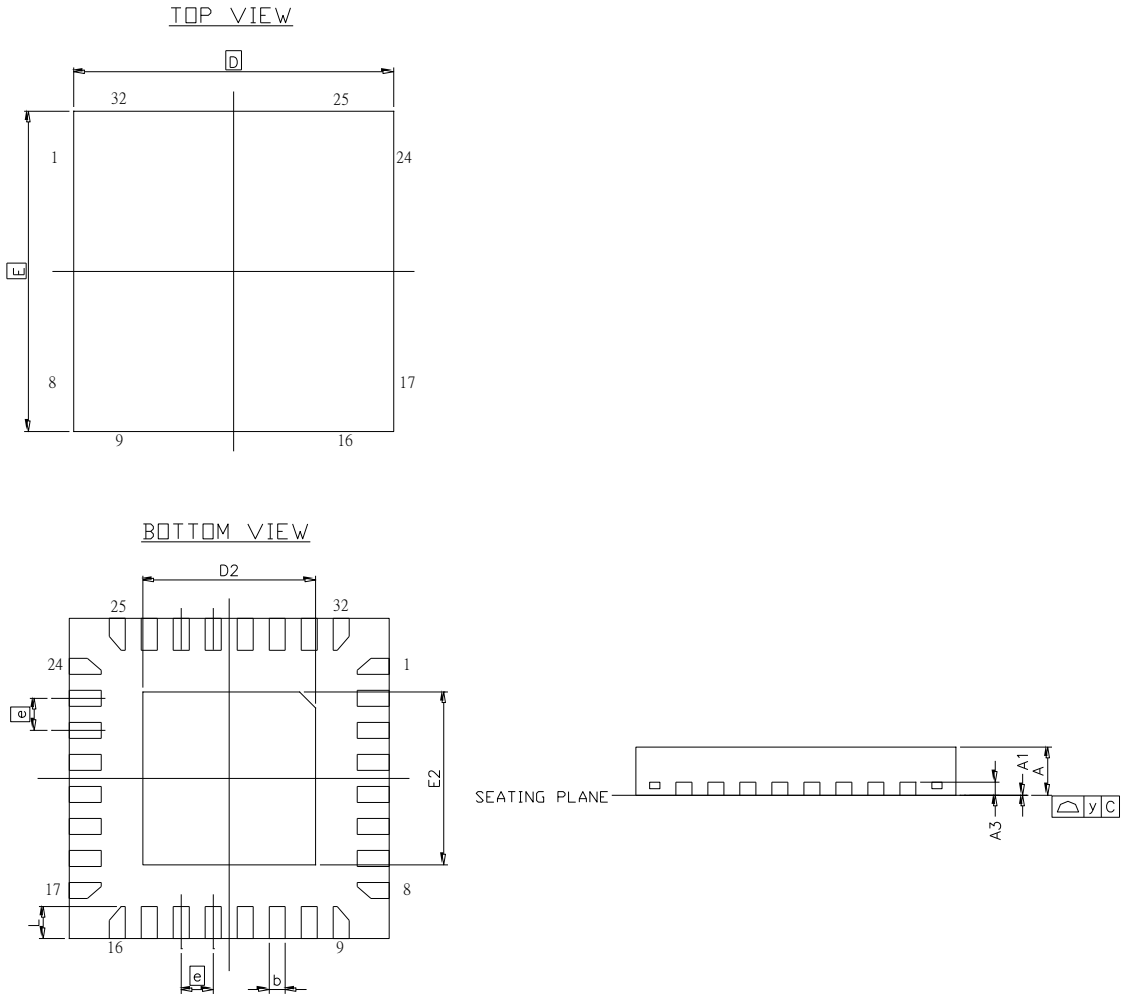
9.2 48L LQFP (7x7x1.4 mm footprint 2.0 mm)



COTROL DIMENSIONS ARE IN MILLIMETERS.

SYMBOL	MILLIMETER			INCH		
	MIN	NOM.	MAX.	MIN	NOM.	MAX.
A	—	—	1.60	—	—	0.063
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
D1	6.90	7.00	7.10	0.272	0.276	0.280
E1	6.90	7.00	7.10	0.272	0.276	0.280
e	0.35	0.50	0.65	0.014	0.020	0.260
D	8.9	9.00	9.10	0.350	0.354	0.358
E	8.9	9.00	9.10	0.350	0.354	0.358
L	0.45	0.60	0.75	0.018	0.024	0.030
L1	—	1.00	—	—	0.039	—
C	0.09	—	0.20	0.0035	—	0.0079
θ	0°	—	7°	0°	—	7°
b	0.17	0.22	0.27	0.007	0.0087	0.011

9.3 33L QFN (5x5x0.8 mm)



SYMBOL	DIMENSION (MM)			DIMENSION (INCH)		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	0.70	0.75	0.80	0.0275	0.0295	0.0315
A1	0	0.02	0.05	0	0.001	0.002
A3	0.20 REF			0.008 REF		
b	0.18	0.25	0.30	0.007	0.010	0.012
D	5.00 BSC			0.197 BSC		
E	5.00 BSC			0.197 BSC		
e	0.50 BSC			0.0197 BSC		
L	0.30	0.40	0.50	0.012	0.016	0.020
y	0.10			0.0039		
D2	3.20	3.40	3.60	0.124	0.138	0.142
E2	3.20	3.40	3.60	0.124	0.138	0.142

10 REVISION HISTORY

Date	Revision	Description
2012.04.01	1.00	Initial version.
2015.05.29	2.00	1. Merged NUC123xxxANx & NUC123xxxAEx into this document.
2015.11.04	2.01	1. Removed ADC function pins of NUC123 QFN33 package type in section 4.3.1.3, 4.3.2.3 and 4.4.1.
2016.03.25	2.02	1. Updated ADC function pins of NUC123 QFN33 package type in section 4.3.1.3, 4.3.2.3 and 4.4.1.
2016.05.25	2.03	1. Added Register Protection description in section 6.2.7 2. Added Internal Reference Voltage description in section 6.17.5.6
2017.07.13	2.04	1. Updated BOD_VL (BODCR[2:1]) description in section 6.2.6. 2. Updated CBOV1-0 (Config0[22:21]) description in section 6.4.4.6.
2020.05.08	2.05	1. Added peripheral application scheme in chapter 7. 2. Added notes about the hardware reference design for ICE_DAT, ICE_CLK and nRESET pins in section 4.4 and chapter 7.

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.