ARM Cortex®-M0

32-bit Microcontroller

NuMicro® Family
NUC121/125 Series
Technical Reference Manual

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

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

All data and specifications are subject to change without notice.

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

www.nuvoton.com

TABLE OF CONTENTS

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

List of Figures

List of Tables

1 GENERAL DESCRIPTION

The NuMicro® NUC121/125 series is a 32-bit Cortex®-M0 microcontroller with USB 2.0 Full-speed device, a 12-bit ADC and 4 sets of 6-channel BPWM. The NUC121/125 series provides the high 50 MHz operating speed, 8 Kbytes SRAM, 8 USB endpoints and 24 channels of BPWM, which make it powerful in USB communication and data processing. The NUC121/125 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 NUC121/125 series runs up to 50 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 NUC121/125 series is equipped with 32 Kbytes Flash memory, 8 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℃ ~ +105℃. It is also equipped with plenty of peripheral devices, such as 8-channel 12-bit ADC, USCI, UART, SPI, I^{2}C, I^{2}S, USB 2.0 FS device, 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 NUC121/125 series to reduce component count, board space and system cost.

Additionally, the NUC121/125 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. Also all series support SPROM. Moreover, the NUC125 support Voltage Adjustable Interface with individual I/O (1.8V-5.5V) for saving additional cost on adjusting the interface voltage difference of peripheral components.

1.1 Key Features Support Table

***** USCI can be set to UART, I^{2}C or SPI

Product Line	USBD	USCI	UART	I^{2}C	SPI/ I^{2}S	Timer	BPWM	ADC
NUC121	1	1	1	2	1	4	24	12
NUC125	1	1	1	2	1	4	23	11

Table 1.1-1 Key Features Support Table

The NuMicro® NUC121/125 series is suitable for a wide range of applications such as:

- USB Keyboard / Mouse
- Gaming - Joystick
- Industrial Automation
- Home Automation
- VR peripheral application
- USB audio
- Alarm system

2 FEATURES

2.1 NuMicro® NUC121/125 Features

- Core
  - ARM® Cortex®-M0 core running up to 50 MHz
  - One 24-bit system timer
  - Supports Low Power Sleep mode
  - Single-cycle 32-bit hardware multiplier
  - Supports programmable 4 level priorities of Nested Vectored Interrupt Controller (NVIC)
  - Supports programmable mask-able interrupts
  - Supports Serial Wire Debug(SWD) with 2 watch-points/4 breakpoints
- Built-in LDO for wide operating voltage ranged from 2.5V to 5.5V
- Flash Memory
  - Supports 32 KB application ROM (APROM)
  - Supports 4.5 KB Flash for loader (LDROM)
  - Supports 512 bytes Security Protection Rom (SPROM)
  - Supports 12 bytes User Configuration block to control system initiation
  - Supports Data Flash with configurable memory size
  - Supports 512 bytes page erase for all embedded flash
  - Supports In-System-Programming (ISP), In-Application-Programming (IAP) update embedded flash memory
  - Supports CRC-32 checksum calculation function
  - Supports flash all one verification function
  - Hardware external read protection of whole flash memory by Security Lock Bit
  - Supports 2-wired ICP update through SWD/ICE interface
- SRAM Memory
  - 8 KB embedded SRAM
  - Supports byte-, half-word- and word-access
  - Supports PDMA mode
- PDMA (Peripheral DMA)
  - Supports 5 independent configurable channels for automatic data transfer between memories and peripherals
  - Supports single and burst transfer type
  - Supports Normal and Scatter-Gather Transfer modes
  - Supports two types of priorities modes: Fixed-priority and Round-robin modes
  - Supports byte-, half-word- and word-access
  - Supports incrementing mode for the source and destination address for each channel
  - Supports time-out function for channel 0 and channel 1
  - Supports software and SPI/I2S, UART, USCI, USB, ADC, PWM and TIMER request
- Clock Control
  - Built-in 48 MHz internal high speed RC oscillator (HIRC) for USB device operation (Frequency variation < 2% at -40℃ ~ +105℃)
    - ◆ Dynamically calibrating the HIRC OSC to 48 MHz ±0.25% from -40℃ to 105℃ by external 32.768K crystal oscillator (LXT) or internal USB synchronous mode
  - Built-in 10 kHz internal low speed RC oscillator for Watchdog Timer and Wake-up operation
  - Supports one interface to connect external crystal oscillator for high speed or low speed application

    - ◆ Built-in 4~24 MHz external high speed crystal oscillator (HXT) for precise timing operation
    - ◆ Built-in 32.768 kHz external low speed crystal oscillator (LXT) for low-power system operation
  - Supports one PLL up to 100 MHz for high performance system operation, sourced from HIRC and HXT
  - Supports clock on-the-fly switch
  - Supports clock failure detection for high/low speed external crystal oscillator
  - Supports auto clock switch once clock failure detected
  - Supports exception (NMI) generated once a clock failure detected
  - Supports divided clock output

- GPIO
  - Four I/O modes
  - TTL/Schmitt trigger input selectable
  - I/O pin configured as interrupt source with edge/level trigger setting
  - Supports high driver and high sink current I/O (up to 20 mA at 5V)
  - Supports software selectable slew rate control
  - Supports up to 52/38/22 GPIOs for LQFP64/48 and QFN33 respectively

- 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 to count the event from external pin
  - Supports input capture function to capture or reset counter value
  - Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated
  - Support Timer0 ~ Timer3 time-out interrupt signal or capture interrupt signal to trigger BPWM, PWM, ADC and PDMA function
  - Supports Inter-Timer trigger mode

- Watchdog Timer
  - Supports multiple clock sources from LIRC (default selection), HCLK/2048 and LXT
  - Supports 8 selections of time-out period (1.6ms ~ 26.0sec for LIRC)
  - Supports wake up from Power-down or Idle mode
  - Supports Interrupt or reset selectable on watchdog time-out

- Window Watchdog Timer
  - Supports multiple clock sources from HCLK/2048 (default selection) and LIRC
  - Supports Window set by 6-bit counter with 11-bit prescale
  - Supports Interrupt

- BPWM/Capture
  - Supports maximum clock frequency up to 100MHz
  - Supports up to two BPWM modules, each module provides one 16-bit counter and 6 output channels
  - Supports independent mode for BPWM output/Capture input channel
  - Supports 12-bit pre-scalar from 1 to 4096
  - Supports 16-bit resolution BPWM counter
    - ◆ Up, down and up/down counter operation type
  - Supports mask function and tri-state enable for each BPWM pin
  - Supports interrupt on the following events:
    - ◆ BPWM counter match zero, period value or compared value
  - Supports trigger ADC on the following events:
    - ◆ BPWM counter match zero, period value or compared value

- – Supports capture mode with 16-bit resolution for each BPWM pin
- – Supports rising edges, falling edges or both edges capture condition
- – Supports input rising edges, falling edges or both edges capture interrupt
- – Supports rising edges, falling edges or both edges capture with counter reload option

- ● PWM/Capture
  - – Supports maximum clock frequency up to 100MHz
  - – Supports up to two PWM modules, each module provides three 16-bit counter and 6 output channels
  - – Supports independent mode for PWM output/Capture input channel
  - – Supports complementary mode for 3 complementary paired PWM output channel
    - ◆ Dead-time insertion with 12-bit resolution
    - ◆ Two compared values during one period
  - – Supports 12-bit pre-scalar from 1 to 4096
  - – Supports 16-bit resolution PWM counter
    - ◆ Up, down and up/down counter operation type
  - – Supports mask function and tri-state enable for each PWM pin
  - – Supports brake function
    - ◆ Brake source from pin and system safety events (clock failed, Brown-out detection and CPU lockup)
    - ◆ Noise filter for brake source from pin
    - ◆ Edge detect brake source to control brake state until brake interrupt cleared
    - ◆ Level detect brake source to auto recover function after brake condition removed
  - – Supports interrupt on the following events:
    - ◆ PWM counter match zero, period value or compared value
    - ◆ Brake condition happened
  - – Supports trigger ADC on the following events:
    - ◆ PWM counter match zero, period value or compared value
  - – Supports capture mode with 16-bit resolution for each PWM pin
  - – Supports rising edges, falling edges or both edges capture condition
  - – Supports input rising edges, falling edges or both edges capture interrupt
  - – Supports rising edges, falling edges or both edges capture with counter reload option
  - – Supports PDMA for capture mode

- ● USCI
  - – UART Mode
    - ◆ Supports one transmit buffer and two receive buffer for data payload
    - ◆ Supports hardware auto flow control function
    - ◆ Supports programmable baud-rate generator
    - ◆ Support 9-Bit Data Transfer (Support 9-Bit RS-485)
    - ◆ Baud rate detection possible by built-in capture event of baud rate generator
    - ◆ Supports Wake-up function (Data and nCTS Wakeup Only)
    - ◆ Supports PDMA transfer
  - – SPI Mode
    - ◆ Supports Master or Slave mode operation (the maximum frequency -- Master = f_{PCLK} / 2, Slave = f_{PCLK} / 5)
    - ◆ Supports one transmit buffer and two receive buffers for data payload
    - ◆ Configurable bit length of a transfer word from 4 to 16-bit
    - ◆ Supports MSB first or LSB first transfer sequence
    - ◆ Supports Word Suspend function
    - ◆ Supports 3-wire, no slave select signal, bi-direction interface
    - ◆ Supports wake-up function by slave select signal in Slave mode
    - ◆ Supports one data channel half-duplex transfer
    - ◆ Supports PDMA transfer
  - – I^{2}C Mode
    - ◆ Full master and slave device capability

    - Supports of 7-bit addressing, as well as 10-bit addressing
    - Communication in standard mode (100 kBit/s) or in fast mode (up to 400 kBit/s)
    - Supports multi-master bus
    - Supports one transmit buffer and two receive buffer for data payload
    - Supports 10-bit bus time-out capability
    - Supports bus monitor mode.
    - Supports Power down wake-up by data toggle or address match
    - Supports setup/hold time programmable
    - Supports multiple address recognition (two slave address with mask option)

- UART
  - Supports one set of UART
  - Supports maximum clock frequency up to 10 Mbps
  - Full-duplex asynchronous communications
  - Separates receive and transmit 16/16 bytes entry FIFO for data payloads
  - Supports hardware auto-flow control (RX, TX, CTS and RTS)
  - Programmable receiver buffer trigger level
  - Supports programmable baud rate generator for each channel individually
  - Supports 8-bit receiver buffer time-out detection function
  - Programmable transmitting data delay time between the last stop and the next start bit by setting DLY (UART_TOUT [15:8])
  - Supports Auto-Baud Rate measurement and baud rate compensation function
  - Supports break error, frame error, parity error and receive/transmit buffer overflow detection function
  - Fully programmable serial-interface characteristics
    - Programmable number of data bit, 5-, 6-, 7-, 8- bit character
    - Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
    - Programmable stop bit, 1, 1.5, or 2 stop bit generation
  - Supports IrDA SIR function mode
    - Supports for 3/16 bit duration for normal mode
  - Supports LIN function mode
    - Supports LIN master/slave mode
    - Supports programmable break generation function for transmitter
    - Supports break detection function for receiver
  - Supports RS-485 mode
    - Supports RS-485 9-bit mode
    - Supports hardware or software enables to program nRTS pin to control RS-485 transmission direction
  - Supports nCTS, incoming data, Received Data FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function
  - Supports PDMA transfer

- SPI / I^{2}S
  - SPI
    - Supports one set of SPI controller
    - Supports Master or Slave mode operation
    - Configurable bit length of a transfer word from 8 to 32-bit
    - Provides separate 4-/8-level depth transmit and receive FIFO buffers
    - Supports MSB first or LSB first transfer sequence
    - Supports Byte Reorder function
    - Supports PDMA transfer
  - I^{2}S
    - Supports Master or Slave mode operation
    - Capable of handling 8-, 16-, 24- and 32-bit word sizes in I^{2}S mode
    - Provides separate 4-level depth transmit and receive FIFO buffers in I^{2}S mode

  - Supports monaural and stereo audio data in I^{2}S mode
  - Supports PCM mode A, PCM mode B, I^{2}S and MSB justified data format in I^{2}S mode
  - Supports PDMA transfer

- I^{2}C
  - Supports up to two sets of I^{2}C devices
  - Supports speed up to 1Mbps
  - Supports Master/Slave mode
  - Supports bidirectional data transfer between masters and slaves
  - Supports multi-master bus bus (no central master)
  - Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
  - Serial clock synchronization allows devices with different bit rates to communicate via one serial bus
  - Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer
  - Supports 14-bit time-out counter requesting the I^{2}C interrupt if the I^{2}C bus hangs up and timer-out counter overflows
  - Programmable clocks allow versatile rate control
  - Supports multiple address recognition, four slave address with mask option
  - Supports two-level buffer function
  - Supports setup/hold time programmable
  - Supports wake-up function

- USB 2.0 FS Device Controller
  - Compliant with USB 2.0 Full-Speed specification
  - Provides 1 interrupt vector with 4 different interrupt events (NEVWK, VBUSDET, USB and BUS)
  - Supports Control/Bulk/Interrupt/Isochronous transfer type
  - Supports suspend function when no bus activity existing for 3 ms
  - Supports 8 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 768 bytes buffer size
  - Provides remote wake-up capability
  - Supports USB 2.0 Link Power Management (LPM)
  - Supports Crystal-less function
  - Supports PDMA transfer

- ADC
  - Supports 12-bit SAR ADC
  - 12-bit resolution and 10-bit accuracy is guaranteed
  - Analog input voltage range: 0~ AV_{DD}
  - Up to 12 single-end analog input channels or 6 differential analog input channels
  - Maximum ADC peripheral clock frequency is 16 MHz
  - Conversion rate up to 800K SPS at 5V
  - Configurable ADC internal sampling time
  - Supports single, burst, single-cycle scan, and continuous scan modes on enabled channels
  - Supports individual conversion result register with valid and overrun indicators for each channel
  - Supports digital comparator to monitor conversion result and user can select whether to generate an interrupt when conversion result matches the compare register setting
  - An A/D conversion can be triggered by:
    - Software enable
    - External pin (STADC)
    - Timer 0~3 overflow pulse trigger

    - ◆ PWM triggers with optional start delay period
  - Supports 2 internal channels for
    - ◆ Band-gap VBG input
    - ◆ Temperature sensor input
  - Supports PDMA transfer

- Supports 96-bit Unique ID (UID)
- Supports 128-bit Unique Customer ID (UCID)
- One built-in temperature sensor with 1℃ resolution
- Brown-out detector
  - With 4 levels: 4.3 V/ 3.7V/ 2.7V/ 2.2V
  - Supports Brown-out Interrupt and Reset option
- Low Voltage Reset
  - Threshold voltage levels: 2.0 V
- Operating Temperature: -40℃~105℃
- Packages
  - All Green package (RoHS)
  - LQFP 64-pin (7mm x 7mm)
  - LQFP 48-pin (7mm x 7mm)
  - QFN 33-pin (5mm x 5 mm)

3 ABBREVIATIONS

3.1 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
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	48 MHz Internal High Speed RC Oscillator
HXT	4~24 MHz External High Speed Crystal Oscillator
IAP	In Application Programming
ICP	In Circuit Programming
ISP	In System Programming
LDO	Low Dropout Regulator
LIN	Local Interconnect Network
LIRC	10 kHz internal low speed RC oscillator (LIRC)
MPU	Memory Protection Unit
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PDMA	Peripheral Direct Memory Access
PLL	Phase-Locked Loop
PWM	Pulse Width Modulation
QEI	Quadrature Encoder Interface
SD	Secure Digital
SPI	Serial Peripheral Interface

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

Table 3.1-1 List of Abbreviations

4 PARTS INFORMATION LIST AND PIN CONFIGURATION

4.1 NuMicro® NUC121/125 Selection Guide

4.1.1 NuMicro® NUC121/125 Naming Rule

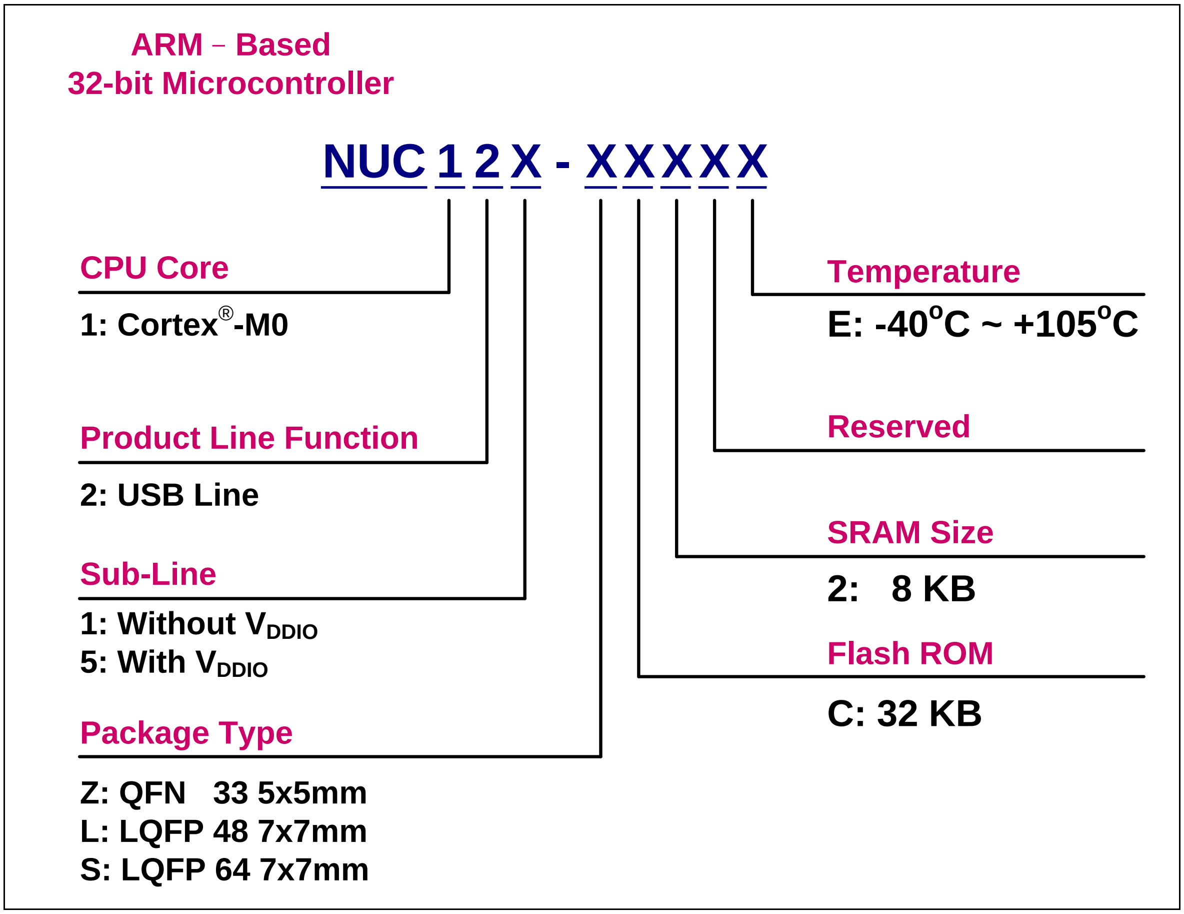


Figure 4.1-1 NuMicro® NUC121/125 Selection Code

4.1.2 NuMicro® NUC121 USB Series Selection Guide

* USCI can be set to UART, I^{2}C or SPI

Part Number	Flash (KB)	SRAM (KB)	ISP Loader ROM (KB)	I/O	Timer/PWM	Connectivity					PWM	ADC (12-Bit)	PDMA	ICP/ISP/IAP	1.8V Power Pin	Package
						USCI*	UART	SPI/I^{2}S	I^{2}C	USBD						
NUC121ZC2AE	32	8	4.5	22	4	1	1	1	2	1	17	4-ch	5-ch	√	-	QFN 33
NUC121LC2AE	32	8	4.5	38	4	1	1	1	2	1	24	10-ch	5-ch	√	-	LQFP 48
NUC121SC2AE	32	8	4.5	52	4	1	1	1	2	1	24	12-ch	5-ch	√	-	LQFP 64

Table 4.1-1 NuMicro® NUC121 USB Series Selection Guide

4.1.3 NuMicro® NUC125 USB Series Selection Guide

* USCI can be set to UART, I^{2}C or SPI

Part Number	Flash (KB)	SRAM (KB)	ISP Loader ROM (KB)	I/O	Timer/PWM	Connectivity					PWM	ADC (12-Bit)	PDMA	ICP/ISP/IAP	1.8V Power Pin	Package
						USCI*	UART	SPI/I^{2}S	I^{2}C	USBD						
NUC125ZC2AE	32	8	4.5	22	4	1	1	1	2	1	17	4-ch	5-ch	√	√	QFN 33
NUC125LC2AE	32	8	4.5	37	4	1	1	1	2	1	23	9-ch	5-ch	√	√	LQFP 48
NUC125SC2AE	32	8	4.5	51	4	1	1	1	2	1	23	11-ch	5-ch	√	√	LQFP 64

Table 4.1-2 NuMicro® NUC125 USB Series Selection Guide

4.2 Pin Configuration

4.2.1 NuMicro® NUC121 QFN 33-Pin Diagram

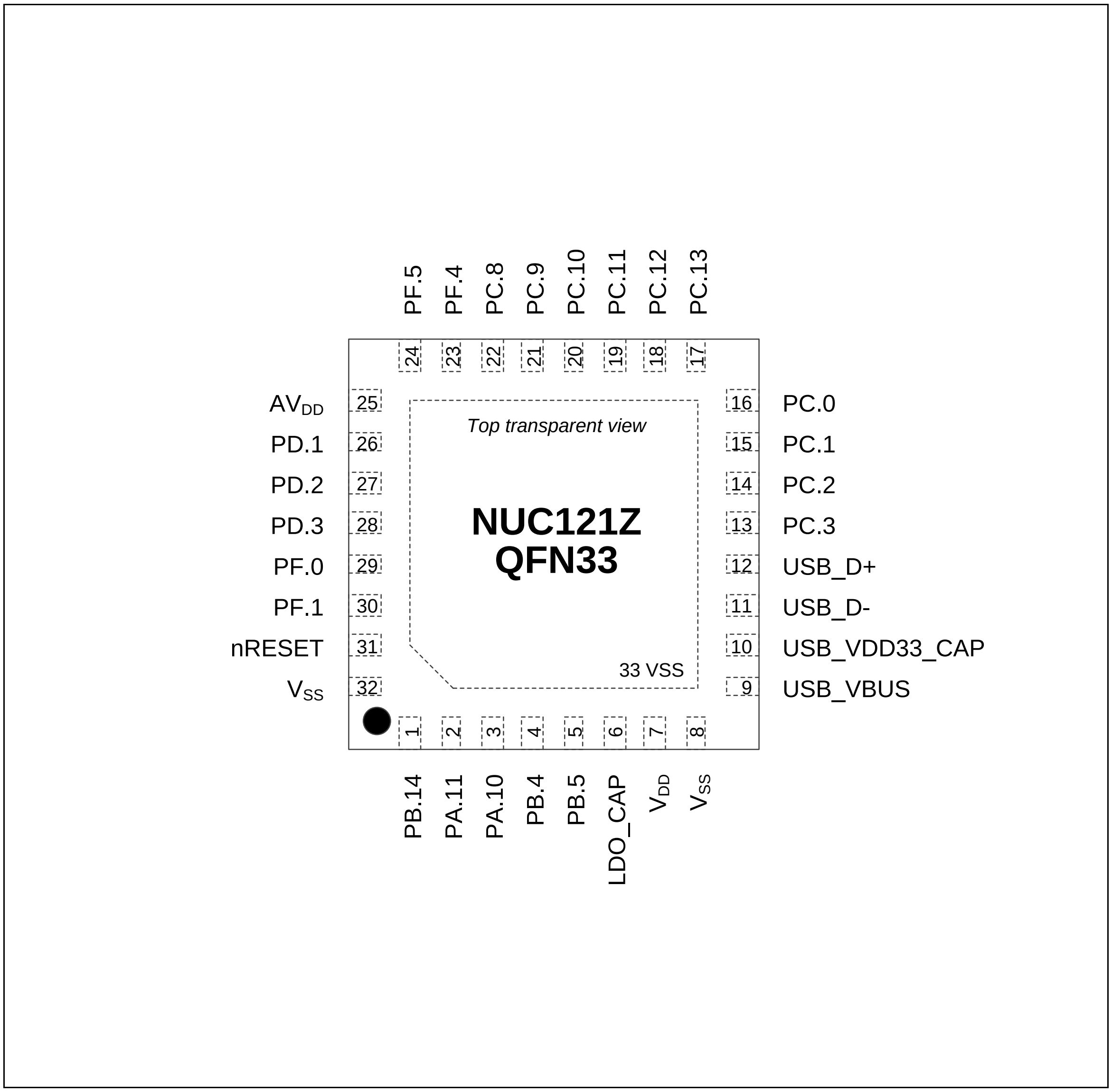


Figure 4.2-1 NuMicro® NUC121 QFN 33-Pin Diagram

4.2.2 NuMicro® NUC121 QFN 33-Pin Function Diagram

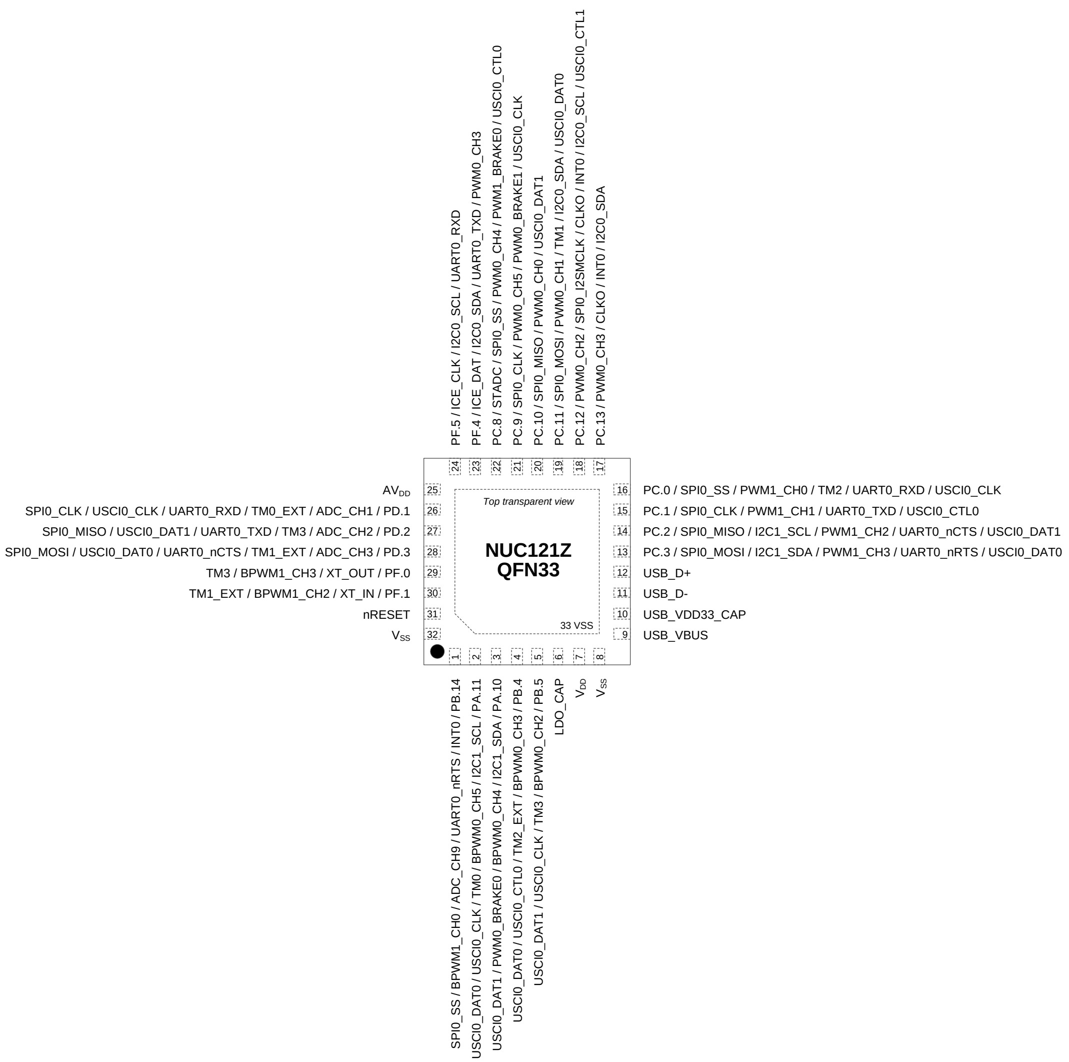

PF.5 / ICE_CLK / I2C0_SCL / UART0_RXD
PF.4 / ICE_DAT / I2C0_SDA / UART0_TXD / PWM0_CH3
PC.8 / STADC / SPI0_SS / PWM0_CH4 / PWM1_BRAKE0 / USCI0_CTL0
PC.9 / SPI0_CLK / PWM0_CH5 / PWM0_BRAKE1 / USCI0_CLK
PC.10 / SPI0_MISO / PWM0_CH0 / USCI0_DAT1
PC.11 / SPI0_MOSI / PWM0_CH1 / TM1 / I2C0_SDA / USCI0_DAT0
PC.12 / PWM0_CH2 / SPI0_I2SMCLK / CLKO / INT0 / I2C0_SCL / USCI0_CTL1
PC.13 / PWM0_CH3 / CLKO / INT0 / I2C0_SDA
24
23
22
21
20
19
18
17
AV$_{DD}$
SPI0_CLK / USCI0_CLK / UART0_RXD / TM0_EXT / ADC_CH1 / PD.1
SPI0_MISO / USCI0_DAT1 / UART0_TXD / TM3 / ADC_CH2 / PD.2
SPI0_MOSI / USCI0_DAT0 / UART0_nCTS / TM1_EXT / ADC_CH3 / PD.3
TM3 / BPWM1_CH3 / XT_OUT / PF.0
TM1_EXT / BPWM1_CH2 / XT_IN / PF.1
nRESET
V$_{SS}$
25
26
27
28
29
30
31
32
Top transparent view
NUC121Z
QFN33
33 VSS
16
15
14
13
12
11
10
9
PC.0 / SPI0_SS / PWM1_CH0 / TM2 / UART0_RXD / USCI0_CLK
PC.1 / SPI0_CLK / PWM1_CH1 / UART0_TXD / USCI0_CTL0
PC.2 / SPI0_MISO / I2C1_SCL / PWM1_CH2 / UART0_nCTS / USCI0_DAT1
PC.3 / SPI0_MOSI / I2C1_SDA / PWM1_CH3 / UART0_nRTS / USCI0_DAT0
USB_D+
USB_D-
USB_VDD33_CAP
USB_VBUS
1
2
3
4
5
6
7
8
SPI0_SS / BPWM1_CH0 / ADC_CH9 / UART0_nRTS / INT0 / PB.14
USCI0_DAT0 / USCI0_CLK / TM0 / BPWM0_CH5 / I2C1_SCL / PA.11
USCI0_DAT1 / PWM0_BRAKE0 / BPWM0_CH4 / I2C1_SDA / PA.10
USCI0_DAT0 / USCI0_CTL0 / TM2_EXT / BPWM0_CH3 / PB.4
USCI0_DAT1 / USCI0_CLK / TM3 / BPWM0_CH2 / PB.5
LDO_CAP
V$_{DD}$
V$_{SS}$

Figure 4.2-2 NuMicro® NUC121 QFN 33-Pin Function Diagram

4.2.3 NuMicro® NUC121 LQFP 48-Pin Diagram

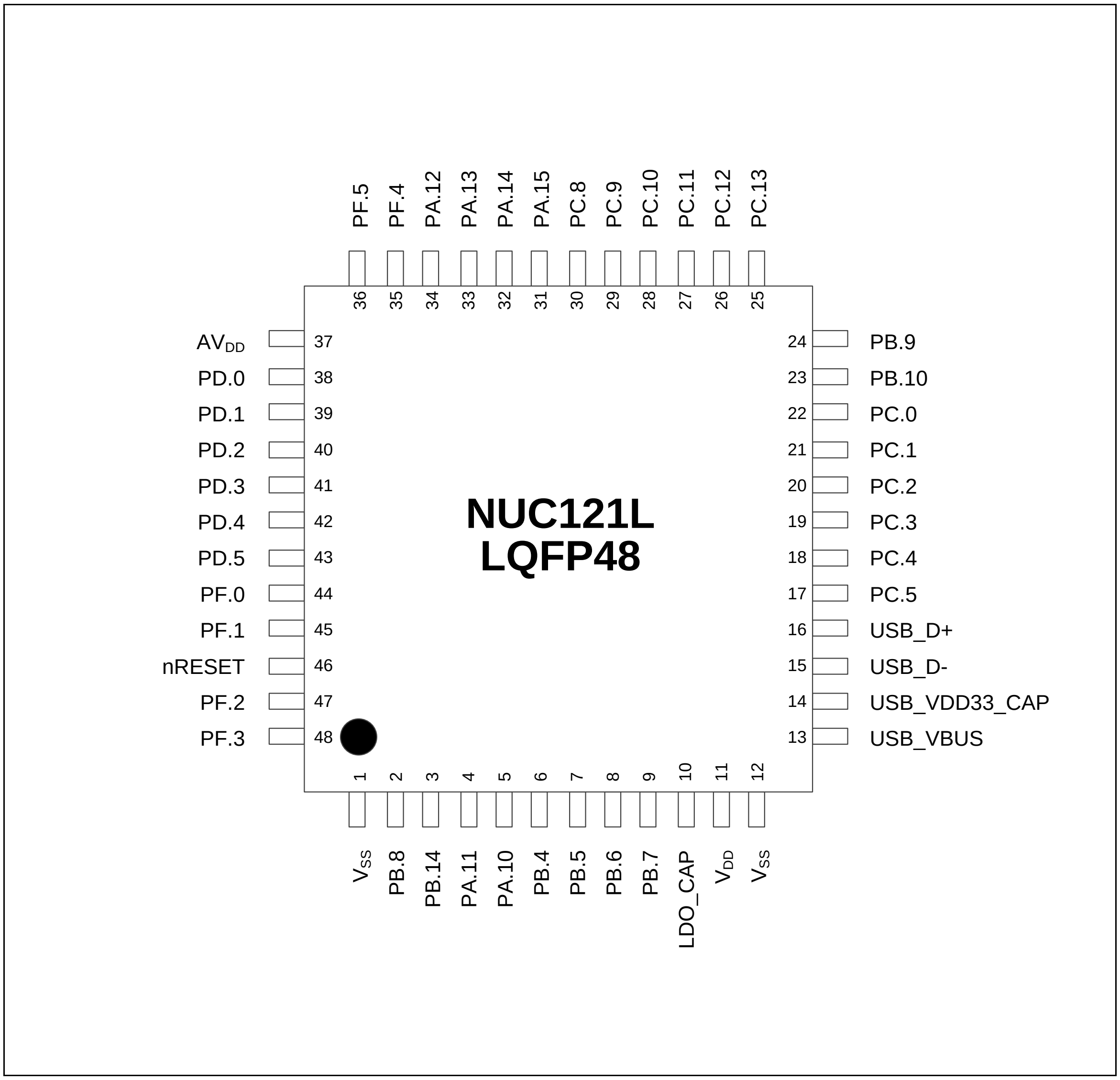


Figure 4.2-3 NuMicro® NUC121 LQFP 48-Pin Diagram

4.2.4 NuMicro® NUC121 LQFP 48-Pin Function Diagram

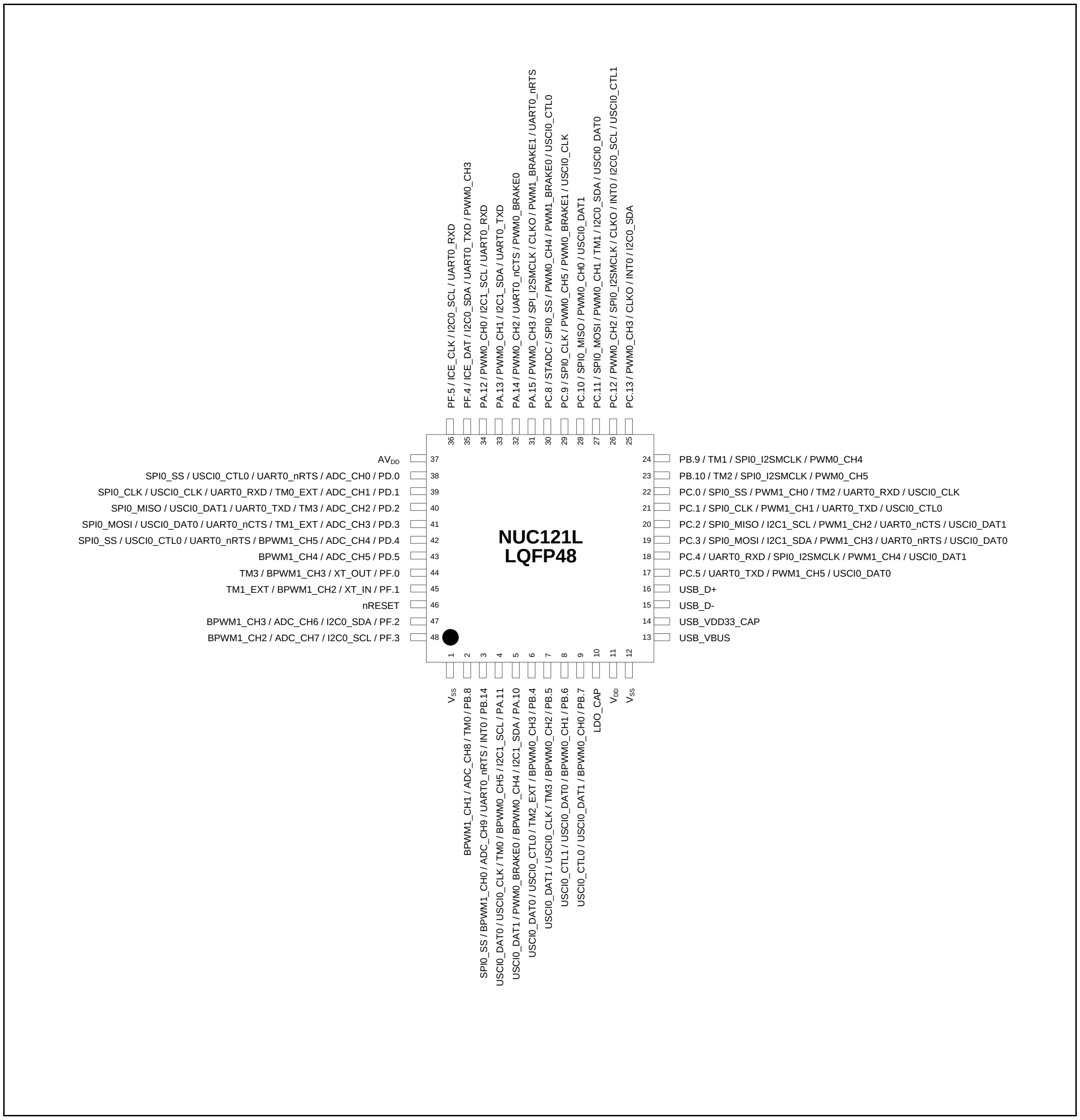

NUC121L
LQFP48
PF.5 / ICE_CLK / I2C0_SCL / UART0_RXD
36
PF.4 / ICE_DAT / I2C0_SDA / UART0_TXD / PWM0_CH3
35
PA.12 / PWM0_CH0 / I2C1_SCL / UART0_RXD
34
PA.13 / PWM0_CH1 / I2C1_SDA / UART0_TXD
33
PA.14 / PWM0_CH2 / UART0_nCTS / PWM0_BRAKE0
32
PA.15 / PWM0_CH3 / SPI_I2SMCLK / CLKO / PWM1_BRAKE1 / UART0_nRTS
31
PC.8 / STADC / SPI0_SS / PWM0_CH4 / PWM1_BRAKE0 / USCI0_CTL0
30
PC.9 / SPI0_CLK / PWM0_CH5 / PWM0_BRAKE1 / USCI0_CLK
29
PC.10 / SPI0_MISO / PWM0_CH0 / USCI0_DAT1
28
PC.11 / SPI0_MOSI / PWM0_CH1 / TM1 / I2C0_SDA / USCI0_DAT0
27
PC.12 / PWM0_CH2 / SPI0_I2SMCLK / CLKO / INT0 / I2C0_SCL / USCI0_CTL1
26
PC.13 / PWM0_CH3 / CLKO / INT0 / I2C0_SDA
25
AV_{DD}
37
SPI0_SS / USCI0_CTL0 / UART0_nRTS / ADC_CH0 / PD.0
38
SPI0_CLK / USCI0_CLK / UART0_RXD / TM0_EXT / ADC_CH1 / PD.1
39
SPI0_MISO / USCI0_DAT1 / UART0_TXD / TM3 / ADC_CH2 / PD.2
40
SPI0_MOSI / USCI0_DAT0 / UART0_nCTS / TM1_EXT / ADC_CH3 / PD.3
41
SPI0_SS / USCI0_CTL0 / UART0_nRTS / BPWM1_CH5 / ADC_CH4 / PD.4
42
BPWM1_CH4 / ADC_CH5 / PD.5
43
TM3 / BPWM1_CH3 / XT_OUT / PF.0
44
TM1_EXT / BPWM1_CH2 / XT_IN / PF.1
45
nRESET
46
BPWM1_CH3 / ADC_CH6 / I2C0_SDA / PF.2
47
BPWM1_CH2 / ADC_CH7 / I2C0_SCL / PF.3
48
24
PB.9 / TM1 / SPI0_I2SMCLK / PWM0_CH4
23
PB.10 / TM2 / SPI0_I2SMCLK / PWM0_CH5
22
PC.0 / SPI0_SS / PWM1_CH0 / TM2 / UART0_RXD / USCI0_CLK
21
PC.1 / SPI0_CLK / PWM1_CH1 / UART0_TXD / USCI0_CTL0
20
PC.2 / SPI0_MISO / I2C1_SCL / PWM1_CH2 / UART0_nCTS / USCI0_DAT1
19
PC.3 / SPI0_MOSI / I2C1_SDA / PWM1_CH3 / UART0_nRTS / USCI0_DAT0
18
PC.4 / UART0_RXD / SPI0_I2SMCLK / PWM1_CH4 / USCI0_DAT1
17
PC.5 / UART0_TXD / PWM1_CH5 / USCI0_DAT0
16
USB_D+
15
USB_D-
14
USB_VDD33_CAP
13
USB_VBUS
1
V_{SS}
2
BPWM1_CH1 / ADC_CH8 / TM0 / PB.8
3
SPI0_SS / BPWM1_CH0 / ADC_CH9 / UART0_nRTS / INT0 / PB.14
4
USCI0_DAT0 / USCI0_CLK / TM0 / BPWM0_CH5 / I2C1_SCL / PA.11
5
USCI0_DAT1 / PWM0_BRAKE0 / BPWM0_CH4 / I2C1_SDA / PA.10
6
USCI0_DAT0 / USCI0_CTL0 / TM2_EXT / BPWM0_CH3 / PB.4
7
USCI0_DAT1 / USCI0_CLK / TM3 / BPWM0_CH2 / PB.5
8
USCI0_CTL1 / USCI0_DAT0 / BPWM0_CH1 / PB.6
9
USCI0_CTL0 / USCI0_DAT1 / BPWM0_CH0 / PB.7
10
LDO_CAP
11
V_{DD}
12
V_{SS}

Figure 4.2-4 NuMicro® NUC121 LQFP 48-Pin Function Diagram

4.2.5 NuMicro® NUC121 LQFP 64-Pin Diagram

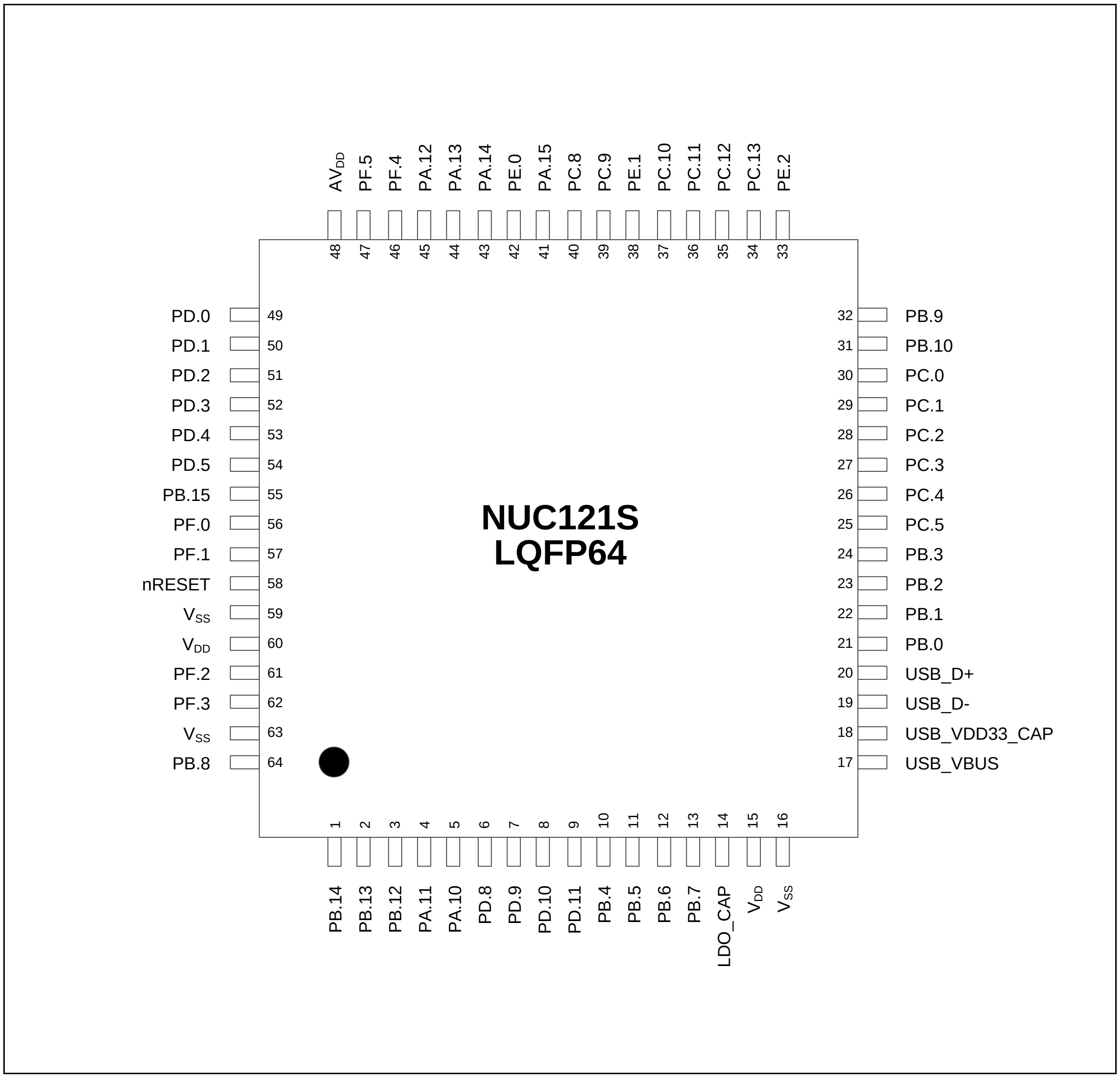


Figure 4.2-5 NuMicro® NUC121 LQFP 64-Pin Diagram

4.2.6 NuMicro® NUC121 LQFP 64-Pin Function Diagram

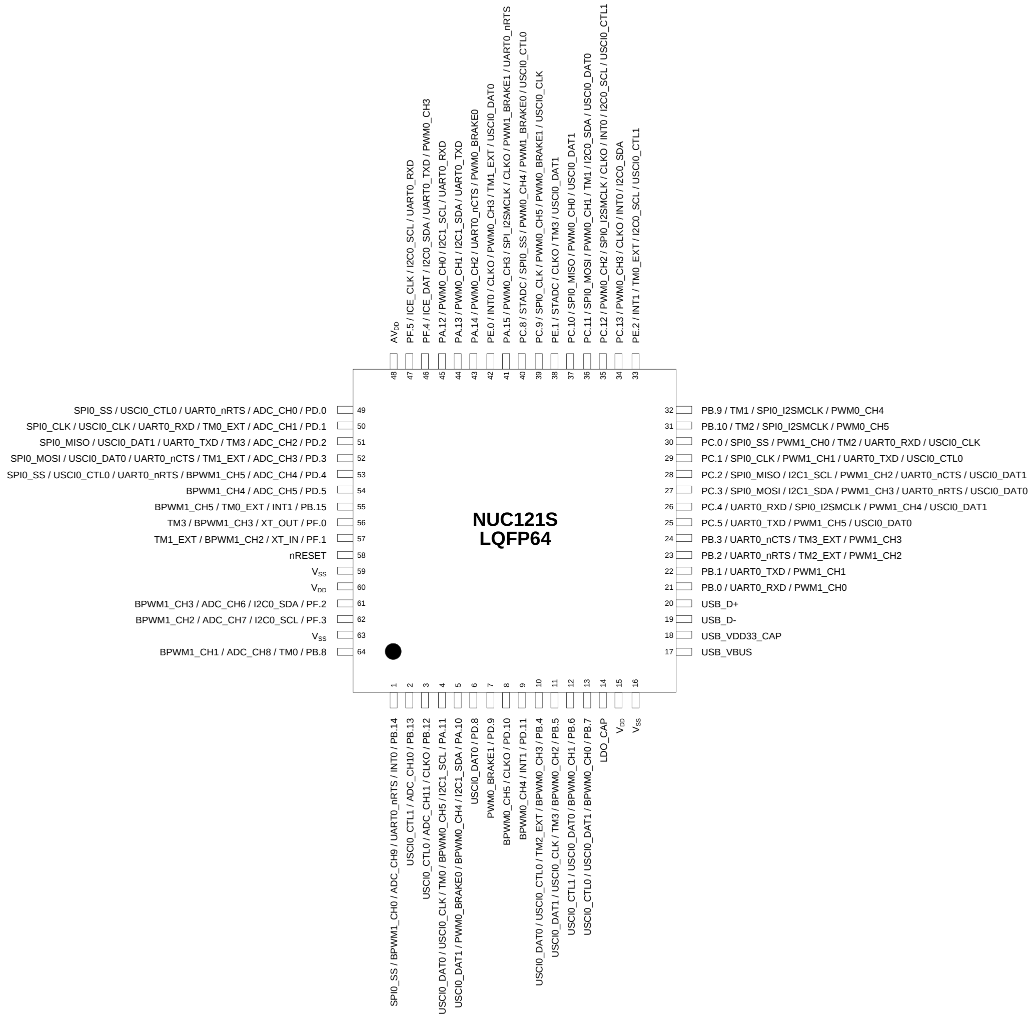

NUC121S
LQFP64
48 AV_{DD}
47 PF.5 / ICE_CLK / I2C0_SCL / UART0_RXD
46 PF.4 / ICE_DAT / I2C0_SDA / UART0_TXD / PWM0_CH3
45 PA.12 / PWM0_CH0 / I2C1_SCL / UART0_RXD
44 PA.13 / PWM0_CH1 / I2C1_SDA / UART0_TXD
43 PA.14 / PWM0_CH2 / UART0_nCTS / PWM0_BRAKE0
42 PE.0 / INT0 / CLKO / PWM0_CH3 / TM1_EXT / USCI0_DAT0
41 PA.15 / PWM0_CH3 / SPI_I2SMCLK / CLKO / PWM1_BRAKE1 / UART0_nRTS
40 PC.8 / STADC / SPI0_SS / PWM0_CH4 / PWM1_BRAKE0 / USCI0_CTL0
39 PC.9 / SPI0_CLK / PWM0_CH5 / PWM0_BRAKE1 / USCI0_CLK
38 PE.1 / STADC / CLKO / TM3 / USCI0_DAT1
37 PC.10 / SPI0_MISO / PWM0_CH0 / USCI0_DAT1
36 PC.11 / SPI0_MOSI / PWM0_CH1 / TM1 / I2C0_SDA / USCI0_DAT0
35 PC.12 / PWM0_CH2 / SPI0_I2SMCLK / CLKO / INT0 / I2C0_SCL / USCI0_CTL1
34 PC.13 / PWM0_CH3 / CLKO / INT0 / I2C0_SDA
33 PE.2 / INT1 / TM0_EXT / I2C0_SCL / USCI0_CTL1
SPI0_SS / USCI0_CTL0 / UART0_nRTS / ADC_CH0 / PD.0 49
SPI0_CLK / USCI0_CLK / UART0_RXD / TM0_EXT / ADC_CH1 / PD.1 50
SPI0_MISO / USCI0_DAT1 / UART0_TXD / TM3 / ADC_CH2 / PD.2 51
SPI0_MOSI / USCI0_DAT0 / UART0_nCTS / TM1_EXT / ADC_CH3 / PD.3 52
SPI0_SS / USCI0_CTL0 / UART0_nRTS / BPWM1_CH5 / ADC_CH4 / PD.4 53
BPWM1_CH4 / ADC_CH5 / PD.5 54
BPWM1_CH5 / TM0_EXT / INT1 / PB.15 55
TM3 / BPWM1_CH3 / XT_OUT / PF.0 56
TM1_EXT / BPWM1_CH2 / XT_IN / PF.1 57
nRESET 58
V_{SS} 59
V_{DD} 60
BPWM1_CH3 / ADC_CH6 / I2C0_SDA / PF.2 61
BPWM1_CH2 / ADC_CH7 / I2C0_SCL / PF.3 62
V_{SS} 63
BPWM1_CH1 / ADC_CH8 / TM0 / PB.8 64
32 PB.9 / TM1 / SPI0_I2SMCLK / PWM0_CH4
31 PB.10 / TM2 / SPI0_I2SMCLK / PWM0_CH5
30 PC.0 / SPI0_SS / PWM1_CH0 / TM2 / UART0_RXD / USCI0_CLK
29 PC.1 / SPI0_CLK / PWM1_CH1 / UART0_TXD / USCI0_CTL0
28 PC.2 / SPI0_MISO / I2C1_SCL / PWM1_CH2 / UART0_nCTS / USCI0_DAT1
27 PC.3 / SPI0_MOSI / I2C1_SDA / PWM1_CH3 / UART0_nRTS / USCI0_DAT0
26 PC.4 / UART0_RXD / SPI0_I2SMCLK / PWM1_CH4 / USCI0_DAT1
25 PC.5 / UART0_TXD / PWM1_CH5 / USCI0_DAT0
24 PB.3 / UART0_nCTS / TM3_EXT / PWM1_CH3
23 PB.2 / UART0_nRTS / TM2_EXT / PWM1_CH2
22 PB.1 / UART0_TXD / PWM1_CH1
21 PB.0 / UART0_RXD / PWM1_CH0
20 USB_D+
19 USB_D-
18 USB_VDD33_CAP
17 USB_VBUS
1 SPI0_SS / BPWM1_CH0 / ADC_CH9 / UART0_nRTS / INT0 / PB.14
2 USCI0_CTL1 / ADC_CH10 / PB.13
3 USCI0_CTL0 / ADC_CH11 / CLKO / PB.12
4 USCI0_DAT0 / USCI0_CLK / TM0 / BPWM0_CH5 / I2C1_SCL / PA.11
5 USCI0_DAT1 / PWM0_BRAKE0 / BPWM0_CH4 / I2C1_SDA / PA.10
6 USCI0_DAT0 / PD.8
7 PWM0_BRAKE1 / PD.9
8 BPWM0_CH5 / CLKO / PD.10
9 BPWM0_CH4 / INT1 / PD.11
10 USCI0_DAT0 / USCI0_CTL0 / TM2_EXT / BPWM0_CH3 / PB.4
11 USCI0_DAT1 / USCI0_CLK / TM3 / BPWM0_CH2 / PB.5
12 USCI0_CTL1 / USCI0_DAT0 / BPWM0_CH1 / PB.6
13 USCI0_CTL0 / USCI0_DAT1 / BPWM0_CH0 / PB.7
14 LDO_CAP
15 V_{DD}
16 V_{SS}

Figure 4.2-6 NuMicro® NUC121 LQFP 64-Pin Function Diagram

4.2.7 NuMicro® NUC125 QFN 33-Pin Diagram

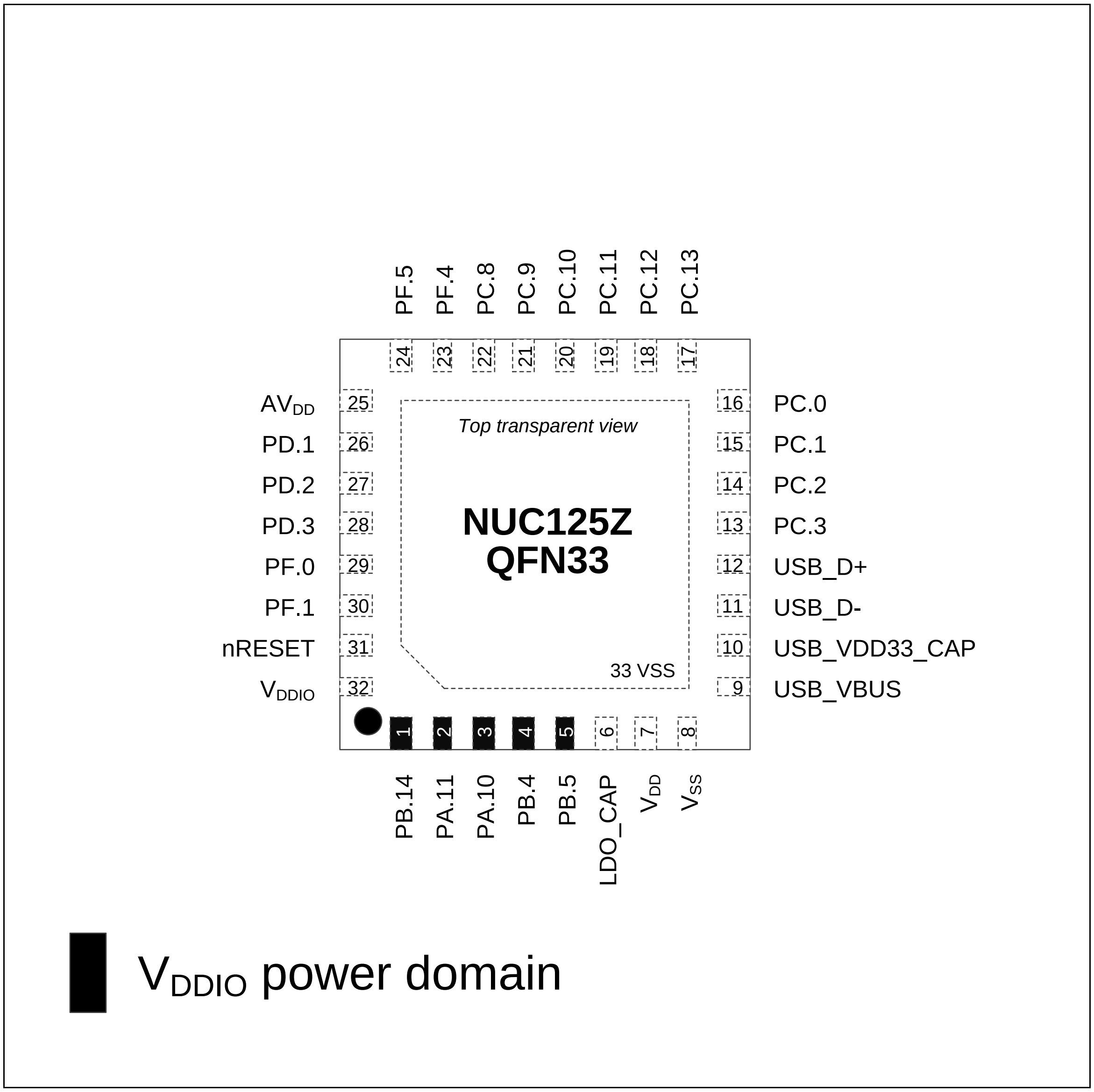


Figure 4.2-7 NuMicro® NUC125 QFN 33-Pin Diagram

4.2.8 NuMicro® NUC125 QFN 33-Pin Function Diagram

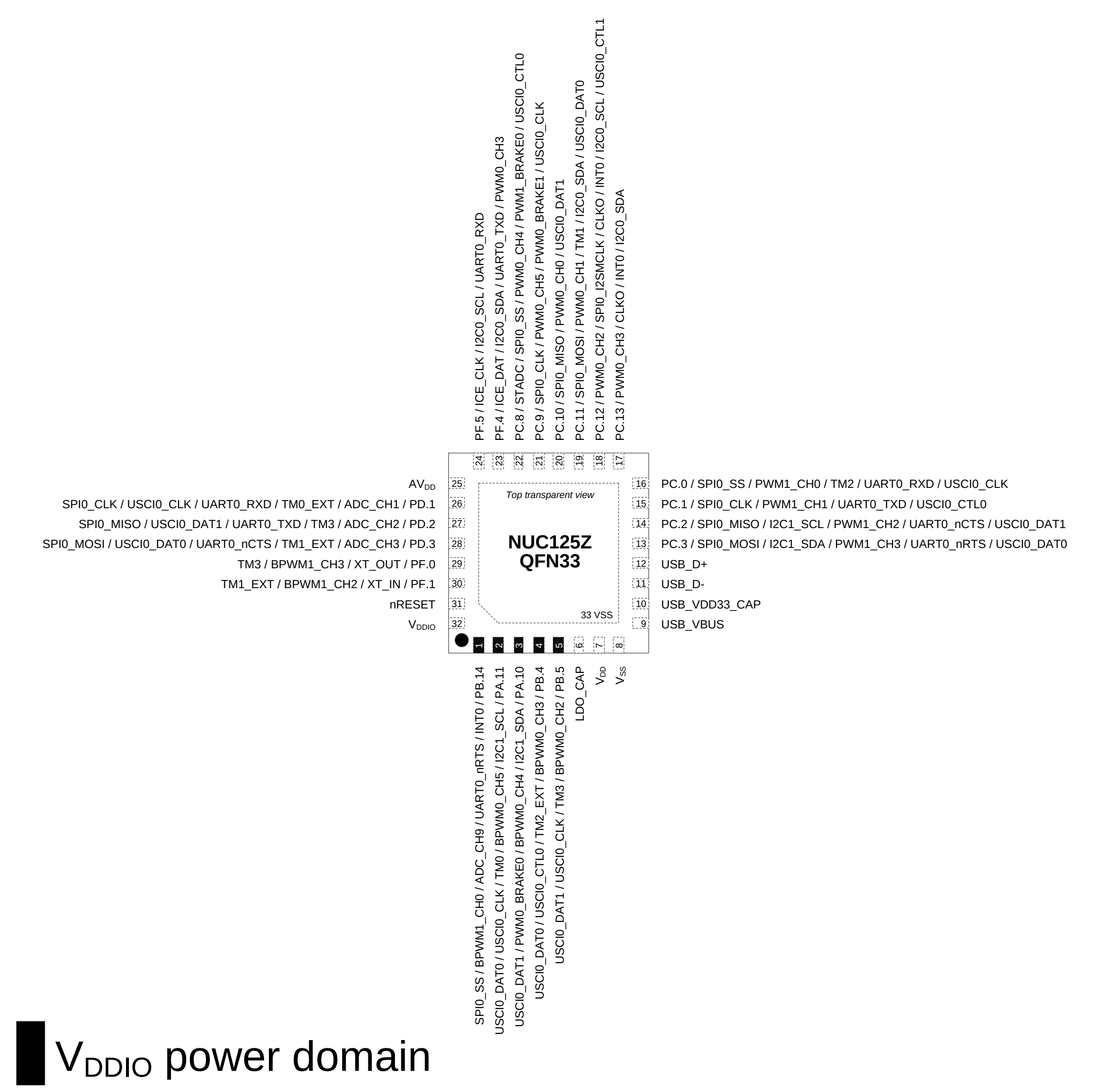

PF.5 / ICE_CLK / I2C0_SCL / UART0_RXD
PF.4 / ICE_DAT / I2C0_SDA / UART0_TXD / PWM0_CH3
PC.8 / STADC / SPI0_SS / PWM0_CH4 / PWM1_BRAKE0 / USCI0_CTL0
PC.9 / SPI0_CLK / PWM0_CH5 / PWM0_BRAKE1 / USCI0_CLK
PC.10 / SPI0_MISO / PWM0_CH0 / USCI0_DAT1
PC.11 / SPI0_MOSI / PWM0_CH1 / TM1 / I2C0_SDA / USCI0_DAT0
PC.12 / PWM0_CH2 / SPI0_I2SMCLK / CLKO / INT0 / I2C0_SCL / USCI0_CTL1
PC.13 / PWM0_CH3 / CLKO / INT0 / I2C0_SDA
24
23
22
21
20
19
18
17
AV_{DD} 25
SPI0_CLK / USCI0_CLK / UART0_RXD / TM0_EXT / ADC_CH1 / PD.1 26
SPI0_MISO / USCI0_DAT1 / UART0_TXD / TM3 / ADC_CH2 / PD.2 27
SPI0_MOSI / USCI0_DAT0 / UART0_nCTS / TM1_EXT / ADC_CH3 / PD.3 28
TM3 / BPWM1_CH3 / XT_OUT / PF.0 29
TM1_EXT / BPWM1_CH2 / XT_IN / PF.1 30
nRESET 31
V_{DDIO} 32
Top transparent view
NUC125Z
QFN33
33 VSS
16 PC.0 / SPI0_SS / PWM1_CH0 / TM2 / UART0_RXD / USCI0_CLK
15 PC.1 / SPI0_CLK / PWM1_CH1 / UART0_TXD / USCI0_CTL0
14 PC.2 / SPI0_MISO / I2C1_SCL / PWM1_CH2 / UART0_nCTS / USCI0_DAT1
13 PC.3 / SPI0_MOSI / I2C1_SDA / PWM1_CH3 / UART0_nRTS / USCI0_DAT0
12 USB_D+
11 USB_D-
10 USB_VDD33_CAP
9 USB_VBUS
1
2
3
4
5
6
7
8
SPI0_SS / BPWM1_CH0 / ADC_CH9 / UART0_nRTS / INT0 / PB.14
USCI0_DAT0 / USCI0_CLK / TM0 / BPWM0_CH5 / I2C1_SCL / PA.11
USCI0_DAT1 / PWM0_BRAKE0 / BPWM0_CH4 / I2C1_SDA / PA.10
USCI0_DAT0 / USCI0_CTL0 / TM2_EXT / BPWM0_CH3 / PB.4
USCI0_DAT1 / USCI0_CLK / TM3 / BPWM0_CH2 / PB.5
LDO_CAP
V_{DD}
V_{SS}
V_{DDIO} power domain

Figure 4.2-8 NuMicro® NUC125 QFN 33-Pin Function Diagram

4.2.9 NuMicro® NUC125 LQFP 48-Pin Diagram

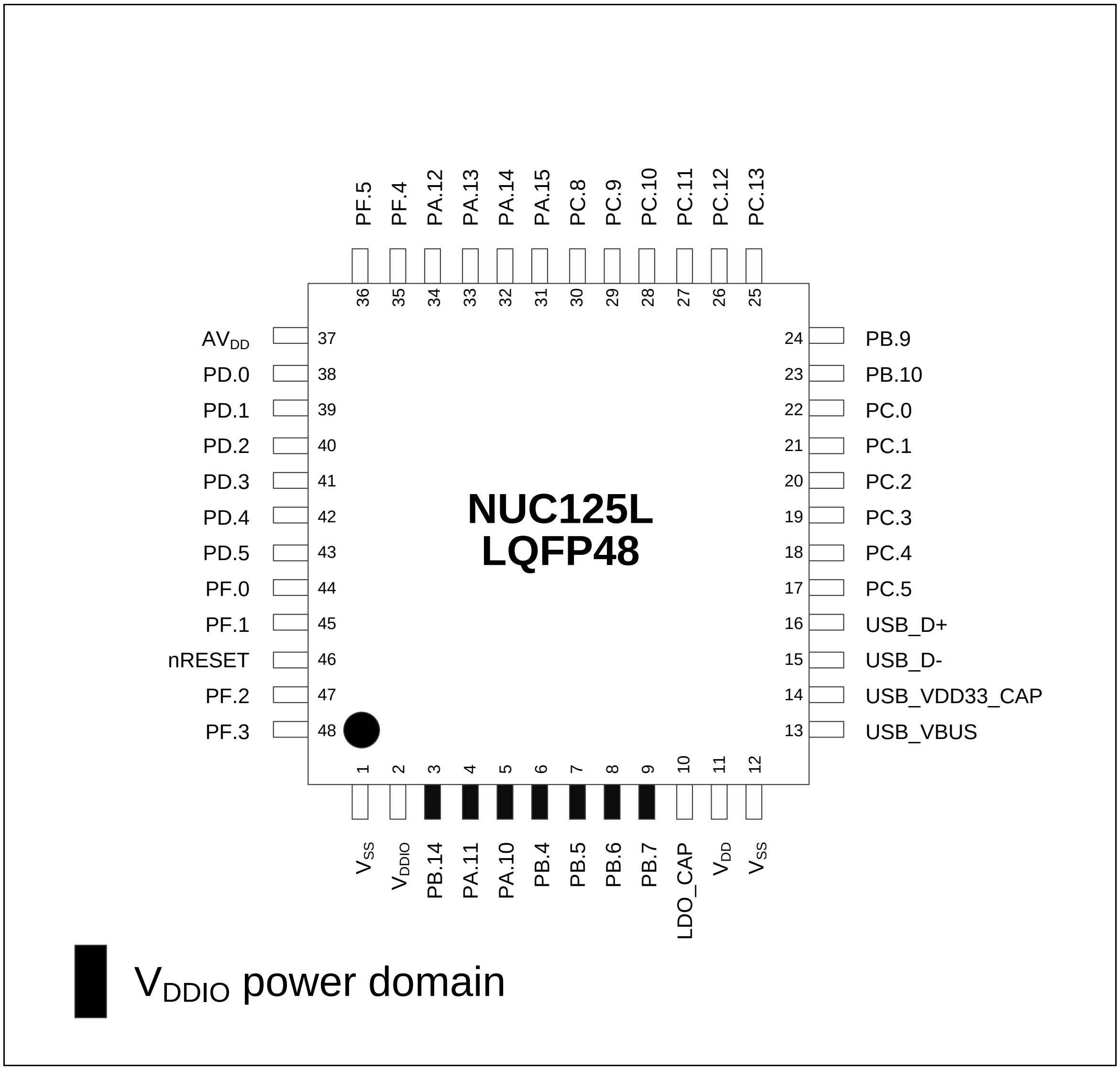


Figure 4.2-9 NuMicro® NUC125 LQFP 48-Pin Diagram

4.2.10 NuMicro® NUC125 LQFP 48-Pin Function Diagram

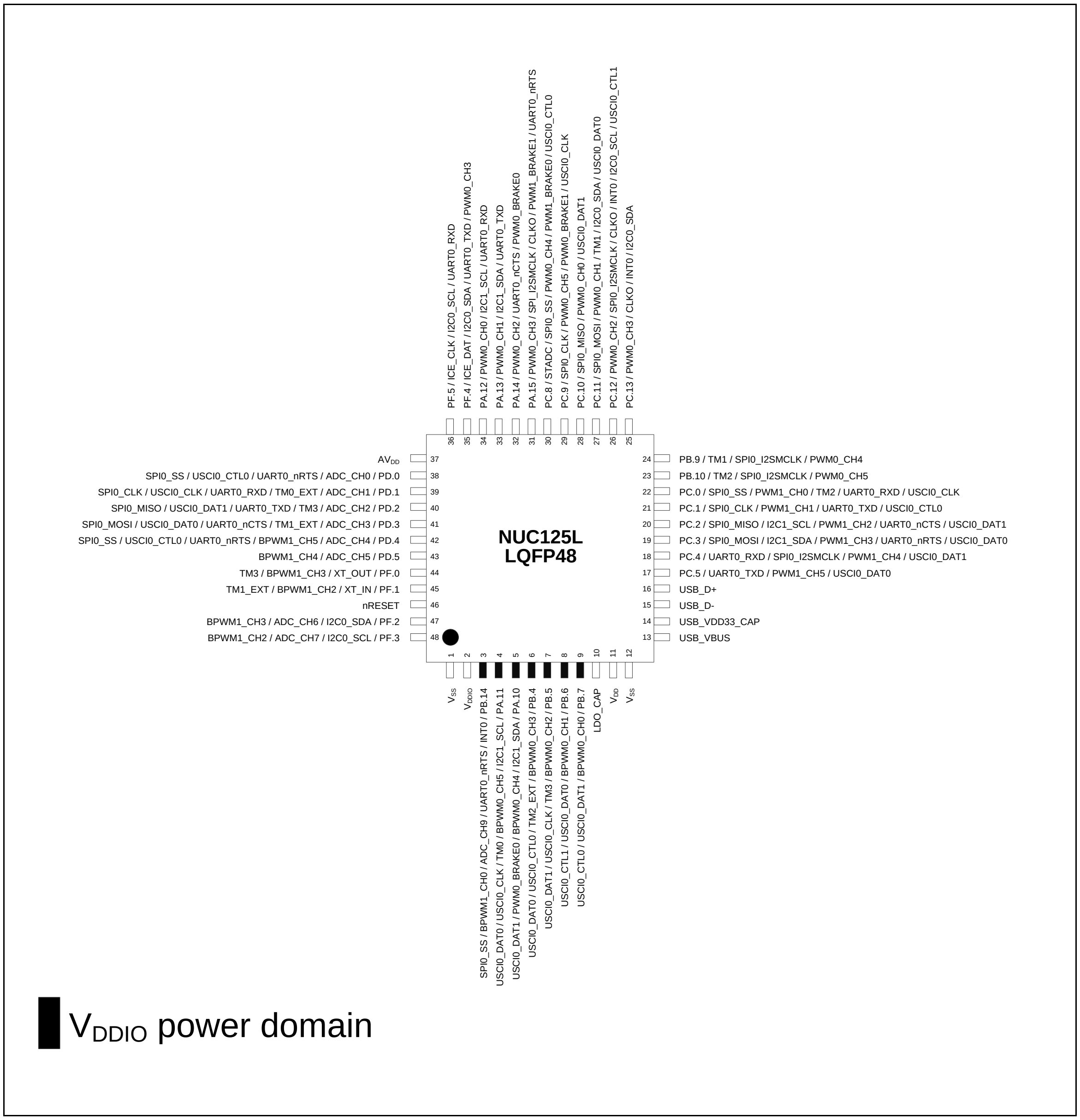

PF.5 / ICE_CLK / I2C0_SCL / UART0_RXD
PF.4 / ICE_DAT / I2C0_SDA / UART0_TXD / PWM0_CH3
PA.12 / PWM0_CH0 / I2C1_SCL / UART0_RXD
PA.13 / PWM0_CH1 / I2C1_SDA / UART0_TXD
PA.14 / PWM0_CH2 / UART0_nCTS / PWM0_BRAKE0
PA.15 / PWM0_CH3 / SPI_I2SMCLK / CLKO / PWM1_BRAKE1 / UART0_nRTS
PC.8 / STADC / SPI0_SS / PWM0_CH4 / PWM1_BRAKE0 / USCI0_CTL0
PC.9 / SPI0_CLK / PWM0_CH5 / PWM0_BRAKE1 / USCI0_CLK
PC.10 / SPI0_MISO / PWM0_CH0 / USCI0_DAT1
PC.11 / SPI0_MOSI / PWM0_CH1 / TM1 / I2C0_SDA / USCI0_DAT0
PC.12 / PWM0_CH2 / SPI0_I2SMCLK / CLKO / INT0 / I2C0_SCL / USCI0_CTL1
PC.13 / PWM0_CH3 / CLKO / INT0 / I2C0_SDA
36
35
34
33
32
31
30
29
28
27
26
25
AV$_{DD}$
37
SPI0_SS / USCI0_CTL0 / UART0_nRTS / ADC_CH0 / PD.0
38
SPI0_CLK / USCI0_CLK / UART0_RXD / TM0_EXT / ADC_CH1 / PD.1
39
SPI0_MISO / USCI0_DAT1 / UART0_TXD / TM3 / ADC_CH2 / PD.2
40
SPI0_MOSI / USCI0_DAT0 / UART0_nCTS / TM1_EXT / ADC_CH3 / PD.3
41
SPI0_SS / USCI0_CTL0 / UART0_nRTS / BPWM1_CH5 / ADC_CH4 / PD.4
42
BPWM1_CH4 / ADC_CH5 / PD.5
43
TM3 / BPWM1_CH3 / XT_OUT / PF.0
44
TM1_EXT / BPWM1_CH2 / XT_IN / PF.1
45
nRESET
46
BPWM1_CH3 / ADC_CH6 / I2C0_SDA / PF.2
47
BPWM1_CH2 / ADC_CH7 / I2C0_SCL / PF.3
48
NUC125L
LQFP48
24
PB.9 / TM1 / SPI0_I2SMCLK / PWM0_CH4
23
PB.10 / TM2 / SPI0_I2SMCLK / PWM0_CH5
22
PC.0 / SPI0_SS / PWM1_CH0 / TM2 / UART0_RXD / USCI0_CLK
21
PC.1 / SPI0_CLK / PWM1_CH1 / UART0_TXD / USCI0_CTL0
20
PC.2 / SPI0_MISO / I2C1_SCL / PWM1_CH2 / UART0_nCTS / USCI0_DAT1
19
PC.3 / SPI0_MOSI / I2C1_SDA / PWM1_CH3 / UART0_nRTS / USCI0_DAT0
18
PC.4 / UART0_RXD / SPI0_I2SMCLK / PWM1_CH4 / USCI0_DAT1
17
PC.5 / UART0_TXD / PWM1_CH5 / USCI0_DAT0
16
USB_D+
15
USB_D-
14
USB_VDD33_CAP
13
USB_VBUS
1
2
3
4
5
6
7
8
9
10
11
12
V$_{SS}$
V$_{DDIO}$
SPI0_SS / BPWM1_CH0 / ADC_CH9 / UART0_nRTS / INT0 / PB.14
USCI0_DAT0 / USCI0_CLK / TM0 / BPWM0_CH5 / I2C1_SCL / PA.11
USCI0_DAT1 / PWM0_BRAKE0 / BPWM0_CH4 / I2C1_SDA / PA.10
USCI0_DAT0 / USCI0_CTL0 / TM2_EXT / BPWM0_CH3 / PB.4
USCI0_DAT1 / USCI0_CLK / TM3 / BPWM0_CH2 / PB.5
USCI0_CTL1 / USCI0_DAT0 / BPWM0_CH1 / PB.6
USCI0_CTL0 / USCI0_DAT1 / BPWM0_CH0 / PB.7
LDO_CAP
V$_{DD}$
V$_{SS}$
V$_{DDIO}$ power domain

Figure 4.2-10 NuMicro® NUC125 LQFP 48-Pin Function Diagram

4.2.11 NuMicro® NUC125 LQFP 64-Pin Diagram

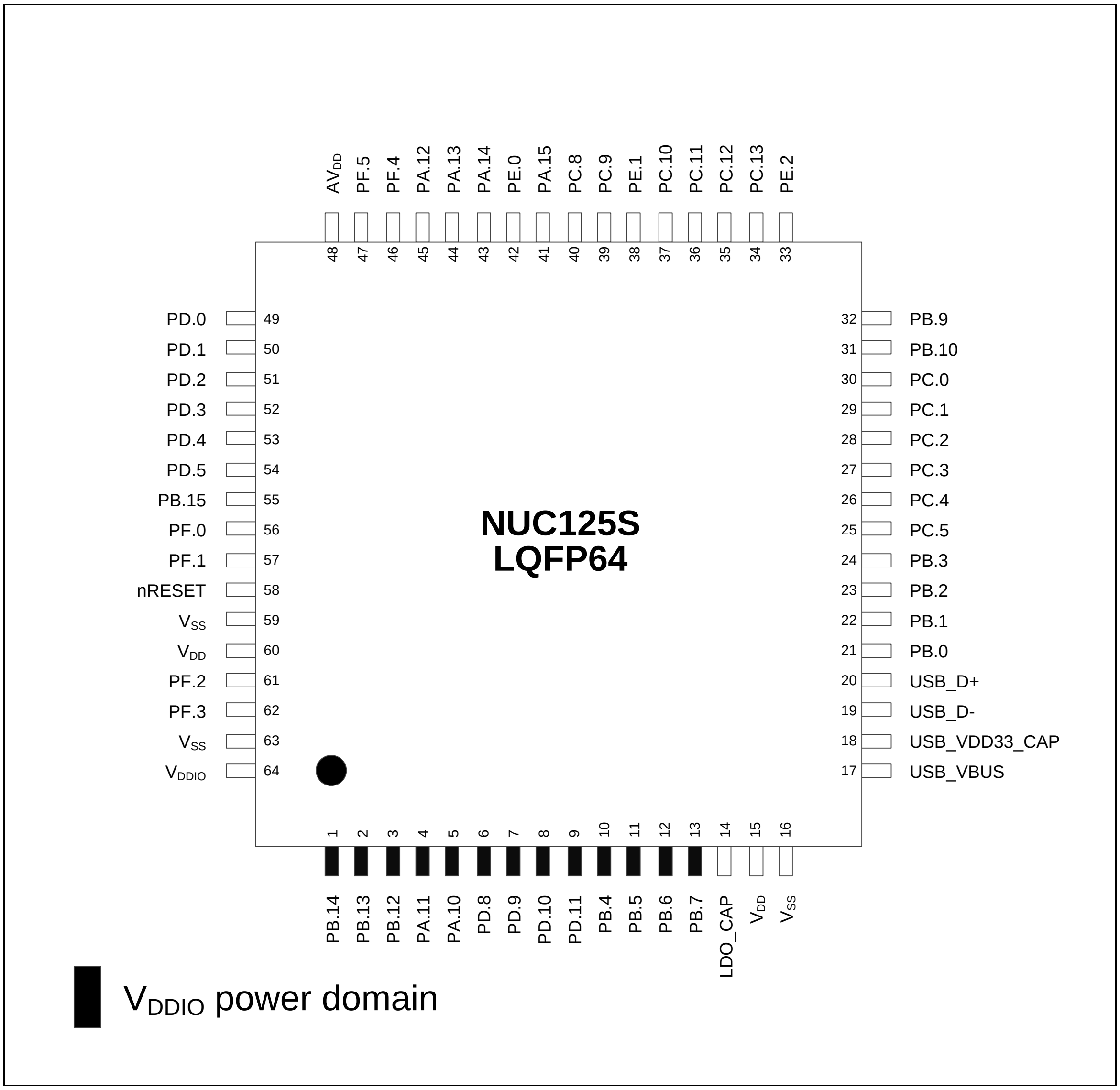


Figure 4.2-11 NuMicro® NUC125 LQFP 64-Pin Diagram

4.2.12 NuMicro® NUC125 LQFP 64-Pin Function Diagram

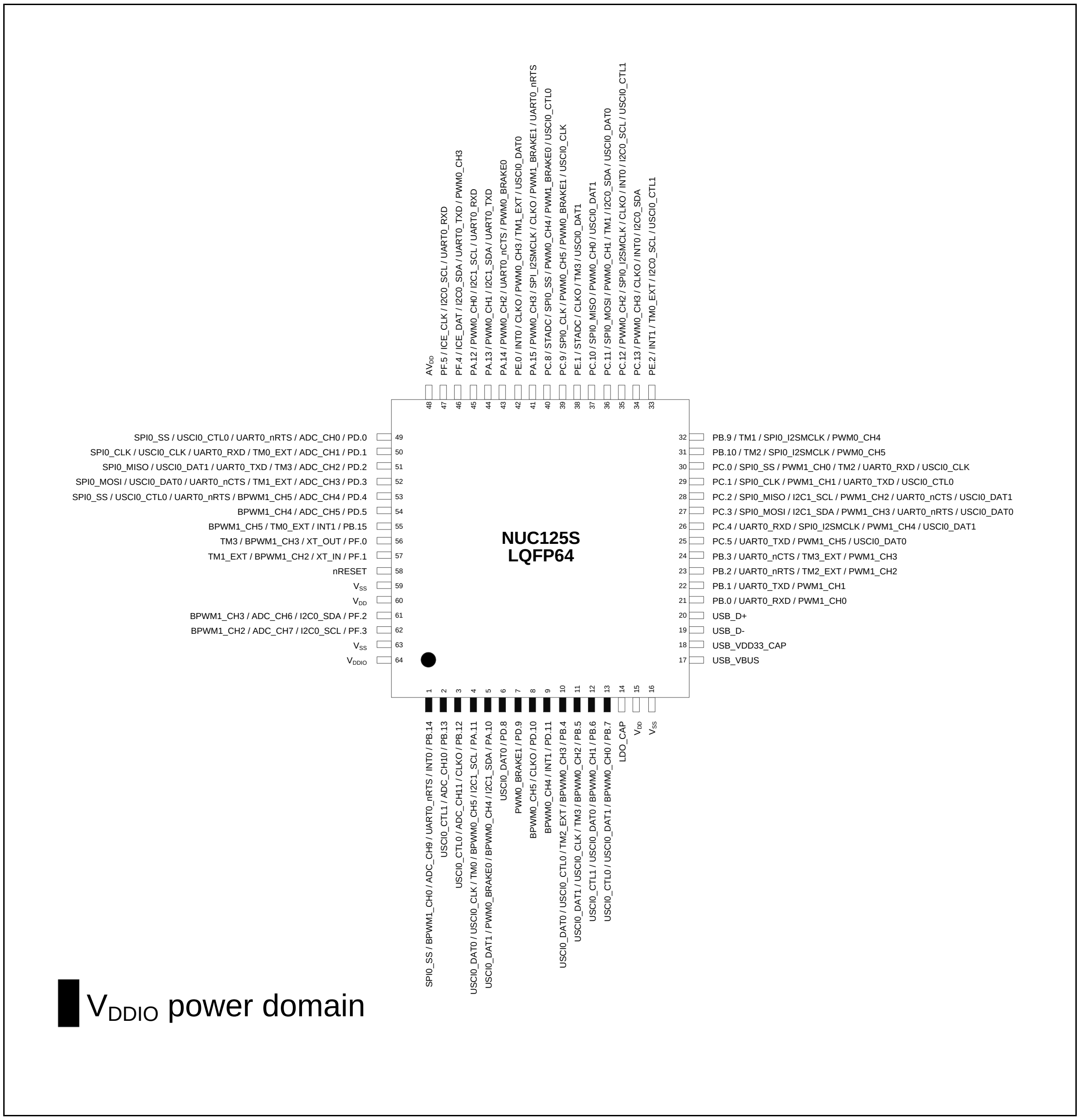

NUC125S
LQFP64
48 AV$_{DD}$
47 PF.5 / ICE_CLK / I2C0_SCL / UART0_RXD
46 PF.4 / ICE_DAT / I2C0_SDA / UART0_TXD / PWM0_CH3
45 PA.12 / PWM0_CH0 / I2C1_SCL / UART0_RXD
44 PA.13 / PWM0_CH1 / I2C1_SDA / UART0_TXD
43 PA.14 / PWM0_CH2 / UART0_nCTS / PWM0_BRAKE0
42 PE.0 / INT0 / CLKO / PWM0_CH3 / TM1_EXT / USCI0_DAT0
41 PA.15 / PWM0_CH3 / SPI_I2SMCLK / CLKO / PWM1_BRAKE1 / UART0_nRTS
40 PC.8 / STADC / SPI0_SS / PWM0_CH4 / PWM1_BRAKE0 / USCI0_CTL0
39 PC.9 / SPI0_CLK / PWM0_CH5 / PWM0_BRAKE1 / USCI0_CLK
38 PE.1 / STADC / CLKO / TM3 / USCI0_DAT1
37 PC.10 / SPI0_MISO / PWM0_CH0 / USCI0_DAT1
36 PC.11 / SPI0_MOSI / PWM0_CH1 / TM1 / I2C0_SDA / USCI0_DAT0
35 PC.12 / PWM0_CH2 / SPI0_I2SMCLK / CLKO / INT0 / I2C0_SCL / USCI0_CTL1
34 PC.13 / PWM0_CH3 / CLKO / INT0 / I2C0_SDA
33 PE.2 / INT1 / TM0_EXT / I2C0_SCL / USCI0_CTL1
SPI0_SS / USCI0_CTL0 / UART0_nRTS / ADC_CH0 / PD.0 49
SPI0_CLK / USCI0_CLK / UART0_RXD / TM0_EXT / ADC_CH1 / PD.1 50
SPI0_MISO / USCI0_DAT1 / UART0_TXD / TM3 / ADC_CH2 / PD.2 51
SPI0_MOSI / USCI0_DAT0 / UART0_nCTS / TM1_EXT / ADC_CH3 / PD.3 52
SPI0_SS / USCI0_CTL0 / UART0_nRTS / BPWM1_CH5 / ADC_CH4 / PD.4 53
BPWM1_CH4 / ADC_CH5 / PD.5 54
BPWM1_CH5 / TM0_EXT / INT1 / PB.15 55
TM3 / BPWM1_CH3 / XT_OUT / PF.0 56
TM1_EXT / BPWM1_CH2 / XT_IN / PF.1 57
nRESET 58
V$_{SS}$ 59
V$_{DD}$ 60
BPWM1_CH3 / ADC_CH6 / I2C0_SDA / PF.2 61
BPWM1_CH2 / ADC_CH7 / I2C0_SCL / PF.3 62
V$_{SS}$ 63
V$_{DDIO}$ 64
32 PB.9 / TM1 / SPI0_I2SMCLK / PWM0_CH4
31 PB.10 / TM2 / SPI0_I2SMCLK / PWM0_CH5
30 PC.0 / SPI0_SS / PWM1_CH0 / TM2 / UART0_RXD / USCI0_CLK
29 PC.1 / SPI0_CLK / PWM1_CH1 / UART0_TXD / USCI0_CTL0
28 PC.2 / SPI0_MISO / I2C1_SCL / PWM1_CH2 / UART0_nCTS / USCI0_DAT1
27 PC.3 / SPI0_MOSI / I2C1_SDA / PWM1_CH3 / UART0_nRTS / USCI0_DAT0
26 PC.4 / UART0_RXD / SPI0_I2SMCLK / PWM1_CH4 / USCI0_DAT1
25 PC.5 / UART0_TXD / PWM1_CH5 / USCI0_DAT0
24 PB.3 / UART0_nCTS / TM3_EXT / PWM1_CH3
23 PB.2 / UART0_nRTS / TM2_EXT / PWM1_CH2
22 PB.1 / UART0_TXD / PWM1_CH1
21 PB.0 / UART0_RXD / PWM1_CH0
20 USB_D+
19 USB_D-
18 USB_VDD33_CAP
17 USB_VBUS
1 SPI0_SS / BPWM1_CH0 / ADC_CH9 / UART0_nRTS / INT0 / PB.14
2 USCI0_CTL1 / ADC_CH10 / PB.13
3 USCI0_CTL0 / ADC_CH11 / CLKO / PB.12
4 USCI0_DAT0 / USCI0_CLK / TM0 / BPWM0_CH5 / I2C1_SCL / PA.11
5 USCI0_DAT1 / PWM0_BRAKE0 / BPWM0_CH4 / I2C1_SDA / PA.10
6 USCI0_DAT0 / PD.8
7 PWM0_BRAKE1 / PD.9
8 BPWM0_CH5 / CLKO / PD.10
9 BPWM0_CH4 / INT1 / PD.11
10 USCI0_DAT0 / USCI0_CTL0 / TM2_EXT / BPWM0_CH3 / PB.4
11 USCI0_DAT1 / USCI0_CLK / TM3 / BPWM0_CH2 / PB.5
12 USCI0_CTL1 / USCI0_DAT0 / BPWM0_CH1 / PB.6
13 USCI0_CTL0 / USCI0_DAT1 / BPWM0_CH0 / PB.7
14 LDO_CAP
15 V$_{DD}$
16 V$_{SS}$
V$_{DDIO}$ power domain

Figure 4.2-12 NuMicro® NUC125 LQFP 64-Pin Function Diagram

4.3 Pin Description

4.3.1 NUC121 USB Series QFN33 Pin Description

MFP* = Multi-function pin. (Refer to section SYS_GPx_MFPL and SYS_GPx_MFPH)

PA.10 MFP5 means SYS_GPA_MFPH[11:8]=0x5.

PC.0 MFP0 means SYS_GPC_MFPL[3:0]=0x0.

Pin No.	Pin Name	Type	MFP*	Description
1	PB.14	I/O	MFP0	General purpose digital I/O pin.
	INT0	I	MFP1	External interrupt0 input pin.
	UART0_nRTS	O	MFP2	Request to Send output pin for UART0.
	ADC_CH9	A	MFP3	ADC channel 9 analog input.
	BPWM1_CH0	I/O	MFP4	BPWM1 channel 0 output/capture input.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
2	PA.11	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SCL	I/O	MFP1	I^{2}C1 clock pin.
	BPWM0_CH5	I/O	MFP4	BPWM0 channel 5 output/capture input.
	TM0	I/O	MFP5	Timer0event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
3	PA.10	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SDA	I/O	MFP1	I^{2}C1 data input/output pin.
	BPWM0_CH4	I/O	MFP4	BPWM0 channel 4 output/capture input.
	PWM0_BRAKE0	I	MFP5	Brake input pin 0 of PWM0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
4	PB.4	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH3	I/O	MFP4	BPWM0 channel 3 output/capture input.
	TM2_EXT	I	MFP5	Timer2 external counter input
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
5	PB.5	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH2	I/O	MFP4	BPWM0 channel 2 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
6	LDO_CAP	A	MFP0	LDO output pin.

Pin No.	Pin Name	Type	MFP*	Description
7	V_{DD}	A	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital function.
8	V_{SS}	A	MFP0	Ground pin for digital circuit.
9	USB_VBUS	A	MFP0	Power supply from USB host or HUB.
10	USB_VDD33_CAP	A	MFP0	Internal power regulator output 3.3V decoupling pin.
11	USB_D-	I	MFP0	USB differential signal D-.
12	USB_D+	I	MFP0	USB differential signal D+.
13	PC.3	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP1	SPI0 MOSI (Master Out, Slave In) pin.
	I2C1_SDA	I/O	MFP3	I^{2}C1 data input/output pin.
	PWM1_CH3	I/O	MFP4	PWM1 channel3 output/capture input.
	UART0_nRTS	O	MFP6	Request to Send output pin for UART0.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
14	PC.2	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP1	SPI0 MISO (Master In, Slave Out) pin.
	I2C1_SCL	I/O	MFP3	I^{2}C1 clock pin.
	PWM1_CH2	I/O	MFP4	PWM1 channel2 output/capture input.
	UART0_nCTS	I	MFP6	Clear to Send input pin for UART0.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
15	PC.1	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP1	SPI0 serial clock pin.
	PWM1_CH1	I/O	MFP4	PWM1 channel1 output/capture input.
	UART0_TXD	O	MFP6	Data transmitter output pin for UART0.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
16	PC.0	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS	I/O	MFP1	SPI0 slave select pin.
	PWM1_CH0	I/O	MFP4	PWM1 channel0 output/capture input.
	TM2	I/O	MFP5	Timer2 event counter input / toggle output
	UART0_RXD	I	MFP6	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin.
17	PC.13	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP2	PWM0 channel3 output/capture input.
	CLKO	O	MFP3	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.

Pin No.	Pin Name	Type	MFP*	Description
18	PC.12	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP2	PWM0 channel2 output/capture input.
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	CLKO	O	MFP4	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SCL	I/O	MFP6	I^{2}C0 clock pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
19	PC.11	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP3	SPI0 MOSI (Master Out, Slave In) pin.
	PWM0_CH1	I/O	MFP4	PWM0 channel1 output/capture input.
	TM1	I/O	MFP5	Timer1 event counter input / toggle output
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
20	PC.10	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP3	SPI0 MISO (Master In, Slave Out) pin.
	PWM0_CH0	I/O	MFP4	PWM0 channel0 output/capture input.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
21	PC.9	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP3	SPI0 serial clock pin.
	PWM0_CH5	I/O	MFP4	PWM0 channel5 output/capture input.
	PWM0_BRAKE1	I	MFP5	Brake input pin 1 of PWM0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin
22	PC.8	I/O	MFP0	General purpose digital I/O pin.
	STADC	I	MFP2	ADC external trigger input.
	SPI0_SS	I/O	MFP3	SPI0 slave select pin.
	PWM0_CH4	I/O	MFP4	PWM0 channel4 output/capture input.
	PWM1_BRAKE0	I	MFP5	Brake input pin 0 of PWM1.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
23	PF.4	I/O	MFP0	General purpose digital I/O pin.
	ICE_DAT	I/O	MFP1	Serial wired debugger data pin
	I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
	UART0_TXD	O	MFP3	Data transmitter output pin for UART0.
	PWM0_CH3	I/O	MFP4	PWM0 channel3 output/capture input.
24	PF.5	I/O	MFP0	General purpose digital I/O pin.

Pin No.	Pin Name	Type	MFP*	Description
	ICE_CLK	I	MFP1	Serial wired debugger clock pin
	I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.
	UART0_RXD	I	MFP3	Data receiver input pin for UART0.
25	AV$_{DD}$	A	MFP0	Power supply for internal analog circuit.
26	PD.1	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH1	A	MFP3	ADC channel 1 analog input.
	TM0_EXT	I	MFP4	Timer0 external counter input
	UART0_RXD	I	MFP5	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	SPI0_CLK	I/O	MFP7	SPI0 serial clock pin.
27	PD.2	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH2	A	MFP3	ADC channel 2 analog input.
	TM3	I/O	MFP4	Timer3 event counter input / toggle output
	UART0_TXD	O	MFP5	Data transmitter output pin for UART0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
	SPI0_MISO	I/O	MFP7	SPI0 MISO (Master In, Slave Out) pin.
28	PD.3	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH3	A	MFP3	ADC channel 3 analog input.
	TM1_EXT	I	MFP4	Timer1 external counter input
	UART0_nCTS	I	MFP5	Clear to Send input pin for UART0.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
	SPI0_MOSI	I/O	MFP7	SPI0 MOSI (Master Out, Slave In) pin.
29	PF.0	I/O	MFP0	General purpose digital I/O pin.
	XT_OUT	O	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal output pin.
	BPWM1_CH3	I/O	MFP4	BPWM1 channel 3 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
30	PF.1	I/O	MFP0	General purpose digital I/O pin.
	XT_IN	I	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal input pin.
	BPWM1_CH2	I/O	MFP4	BPWM1 channel 2 output/capture input.
	TM1_EXT	I	MFP5	Timer1 external counter input
31	nRESET	I	MFP0	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state.
32	V$_{SS}$	A	MFP0	Ground pin for digital circuit.
33	V$_{SS}$	A	MFP0	Ground pin for digital circuit.

Table 4.3-1 NUC121 USB Series QFN33 Pin Description

4.3.2 NUC121 USB Series LQFP48 Pin Description

MFP* = Multi-function pin. (Refer to section SYS_GPx_MFPL and SYS_GPx_MFPH)

PA.10 MFP5 means SYS_GPA_MFPH[11:8]=0x5.

PC.0 MFP0 means SYS_GPC_MFPL[3:0]=0x0..

Pin No.	Pin Name	Type	MFP*	Description
1	V_{SS}	A	MFP0	Ground pin for digital circuit.
2	PB.8	I/O	MFP0	General purpose digital I/O pin.
	TM0	I/O	MFP1	Timer0event counter input / toggle output
	ADC_CH8	A	MFP3	ADC channel 8 analog input.
	BPWM1_CH1	I/O	MFP4	BPWM1 channel 1 output/capture input.
3	PB.14	I/O	MFP0	General purpose digital I/O pin.
	INT0	I	MFP1	External interrupt0 input pin.
	UART0_nRTS	O	MFP2	Request to Send output pin for UART0.
	ADC_CH9	A	MFP3	ADC channel 9 analog input.
	BPWM1_CH0	I/O	MFP4	BPWM1 channel 0 output/capture input.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
4	PA.11	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SCL	I/O	MFP1	I^{2}C1 clock pin.
	BPWM0_CH5	I/O	MFP4	BPWM0 channel 5 output/capture input.
	TM0	I/O	MFP5	Timer0event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
5	PA.10	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SDA	I/O	MFP1	I^{2}C1 data input/output pin.
	BPWM0_CH4	I/O	MFP4	BPWM0 channel 4 output/capture input.
	PWM0_BRAKE0	I	MFP5	Brake input pin 0 of PWM0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
6	PB.4	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH3	I/O	MFP4	BPWM0 channel 3 output/capture input.
	TM2_EXT	I	MFP5	Timer2 external counter input
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
7	PB.5	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH2	I/O	MFP4	BPWM0 channel 2 output/capture input.

Pin No.	Pin Name	Type	MFP*	Description
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
8	PB.6	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH1	I/O	MFP4	BPWM0 channel 1 output/capture input.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
9	PB.7	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH0	I/O	MFP4	BPWM0 channel 0 output/capture input.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin.
10	LDO_CAP	A	MFP0	LDO output pin.
11	V_{DD}	A	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital function.
12	V_{SS}	A	MFP0	Ground pin for digital circuit.
13	USB_VBUS	A	MFP0	Power supply from USB host or HUB.
14	USB_VDD33_CAP	A	MFP0	Internal power regulator output 3.3V decoupling pin.
15	USB_D-	I	MFP0	USB differential signal D-.
16	USB_D+	I	MFP0	USB differential signal D+.
17	PC.5	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP2	Data transmitter output pin for UART0.
	PWM1_CH5	I/O	MFP4	PWM1 channel5 output/capture input.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
18	PC.4	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP2	Data receiver input pin for UART0.
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM1_CH4	I/O	MFP4	PWM1 channel4 output/capture input.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
19	PC.3	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP1	SPI0 MOSI (Master Out, Slave In) pin.
	I2C1_SDA	I/O	MFP3	I^{2}C1 data input/output pin.
	PWM1_CH3	I/O	MFP4	PWM1 channel3 output/capture input.
	UART0_nRTS	O	MFP6	Request to Send output pin for UART0.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
20	PC.2	I/O	MFP0	General purpose digital I/O pin.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

Pin No.	Pin Name	Type	MFP*	Description
	SPI0_MISO	I/O	MFP1	SPI0 MISO (Master In, Slave Out) pin.
	I2C1_SCL	I/O	MFP3	I^{2}C1 clock pin.
	PWM1_CH2	I/O	MFP4	PWM1 channel2 output/capture input.
	UART0_nCTS	I	MFP6	Clear to Send input pin for UART0.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
21	PC.1	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP1	SPI0 serial clock pin.
	PWM1_CH1	I/O	MFP4	PWM1 channel1 output/capture input.
	UART0_TXD	O	MFP6	Data transmitter output pin for UART0.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
22	PC.0	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS	I/O	MFP1	SPI0 slave select pin.
	PWM1_CH0	I/O	MFP4	PWM1 channel0 output/capture input.
	TM2	I/O	MFP5	Timer2 event counter input / toggle output
	UART0_RXD	I	MFP6	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin.
23	PB.10	I/O	MFP0	General purpose digital I/O pin.
	TM2	I/O	MFP1	Timer2 event counter input / toggle output
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM0_CH5	I/O	MFP4	PWM0 channel5 output/capture input.
24	PB.9	I/O	MFP0	General purpose digital I/O pin.
	TM1	I/O	MFP1	Timer1 event counter input / toggle output
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM0_CH4	I/O	MFP4	PWM0 channel4 output/capture input.
25	PC.13	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP2	PWM0 channel3 output/capture input.
	CLKO	O	MFP3	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
26	PC.12	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP2	PWM0 channel2 output/capture input.
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	CLKO	O	MFP4	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.

Pin No.	Pin Name	Type	MFP*	Description
	I2C0_SCL	I/O	MFP6	I^{2}C0 clock pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
27	PC.11	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP3	SPI0 MOSI (Master Out, Slave In) pin.
	PWM0_CH1	I/O	MFP4	PWM0 channel1 output/capture input.
	TM1	I/O	MFP5	Timer1 event counter input / toggle output
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
28	PC.10	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP3	SPI0 MISO (Master In, Slave Out) pin.
	PWM0_CH0	I/O	MFP4	PWM0 channel0 output/capture input.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
29	PC.9	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP3	SPI0 serial clock pin.
	PWM0_CH5	I/O	MFP4	PWM0 channel5 output/capture input.
	PWM0_BRAKE1	I	MFP5	Brake input pin 1 of PWM0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin
30	PC.8	I/O	MFP0	General purpose digital I/O pin.
	STADC	I	MFP2	ADC external trigger input.
	SPI0_SS	I/O	MFP3	SPI0 slave select pin.
	PWM0_CH4	I/O	MFP4	PWM0 channel4 output/capture input.
	PWM1_BRAKE0	I	MFP5	Brake input pin 0 of PWM1.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
31	PA.15	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP1	PWM0 channel3 output/capture input.
	SPI_I2SMCLK	O	MFP2	I2S0 master clock output pin.
	CLKO	O	MFP3	Clock Out
	PWM1_BRAKE1	I	MFP4	Brake input pin 1 of PWM1.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
32	PA.14	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP1	PWM0 channel2 output/capture input.
	UART0_nCTS	I	MFP3	Clear to Send input pin for UART0.
	PWM0_BRAKE0	I	MFP4	Brake input pin 0 of PWM0.
33	PA.13	I/O	MFP0	General purpose digital I/O pin.

Pin No.	Pin Name	Type	MFP*	Description
	PWM0_CH1	I/O	MFP1	PWM0 channel1 output/capture input.
	I2C1_SDA	I/O	MFP2	I^{2}C1 data input/output pin.
	UART0_TXD	O	MFP3	Data transmitter output pin for UART0.
34	PA.12	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH0	I/O	MFP1	PWM0 channel0 output/capture input.
	I2C1_SCL	I/O	MFP2	I^{2}C1 clock pin.
	UART0_RXD	I	MFP3	Data receiver input pin for UART0.
35	PF.4	I/O	MFP0	General purpose digital I/O pin.
	ICE_DAT	I/O	MFP1	Serial wired debugger data pin
	I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
	UART0_TXD	O	MFP3	Data transmitter output pin for UART0.
	PWM0_CH3	I/O	MFP4	PWM0 channel3 output/capture input.
36	PF.5	I/O	MFP0	General purpose digital I/O pin.
	ICE_CLK	I	MFP1	Serial wired debugger clock pin
	I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.
	UART0_RXD	I	MFP3	Data receiver input pin for UART0.
37	AV$_{DD}$	A	MFP0	Power supply for internal analog circuit.
38	PD.0	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH0	A	MFP3	ADC channel 0 analog input.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
39	PD.1	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH1	A	MFP3	ADC channel 1 analog input.
	TM0_EXT	I	MFP4	Timer0 external counter input
	UART0_RXD	I	MFP5	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	SPI0_CLK	I/O	MFP7	SPI0 serial clock pin.
40	PD.2	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH2	A	MFP3	ADC channel 2 analog input.
	TM3	I/O	MFP4	Timer3 event counter input / toggle output
	UART0_TXD	O	MFP5	Data transmitter output pin for UART0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
	SPI0_MISO	I/O	MFP7	SPI0 MISO (Master In, Slave Out) pin.

Pin No.	Pin Name	Type	MFP*	Description
41	PD.3	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH3	A	MFP3	ADC channel 3 analog input.
	TM1_EXT	I	MFP4	Timer1 external counter input
	UART0_nCTS	I	MFP5	Clear to Send input pin for UART0.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
	SPI0_MOSI	I/O	MFP7	SPI0 MOSI (Master Out, Slave In) pin.
42	PD.4	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH4	A	MFP2	ADC channel 4 analog input.
	BPWM1_CH5	I/O	MFP4	BPWM1 channel 5 output/capture input.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
43	PD.5	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH5	A	MFP2	ADC channel 5 analog input.
	BPWM1_CH4	I/O	MFP4	BPWM1 channel 4 output/capture input.
44	PF.0	I/O	MFP0	General purpose digital I/O pin.
	XT_OUT	O	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal output pin.
	BPWM1_CH3	I/O	MFP4	BPWM1 channel 3 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
45	PF.1	I/O	MFP0	General purpose digital I/O pin.
	XT_IN	I	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal input pin.
	BPWM1_CH2	I/O	MFP4	BPWM1 channel 2 output/capture input.
	TM1_EXT	I	MFP5	Timer1 external counter input
46	nRESET	I	MFP0	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state.
47	PF.2	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
	ADC_CH6	A	MFP3	ADC channel 6 analog input.
	BPWM1_CH3	I/O	MFP4	BPWM1 channel 3 output/capture input.
48	PF.3	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.
	ADC_CH7	A	MFP3	ADC channel 7 analog input.
	BPWM1_CH2	I/O	MFP4	BPWM1 channel 2 output/capture input.

Table 4.3-2 NUC121 USB Series LQFP48 Pin Description

4.3.3 NUC121 USB Series LQFP64 Pin Description

MFP* = Multi-function pin. (Refer to section SYS_GPx_MFPL and SYS_GPx_MFPH)

PA.10 MFP5 means SYS_GPA_MFPH[11:8]=0x5.

PC.0 MFP0 means SYS_GPC_MFPL[3:0]=0x0.

Pin No.	Pin Name	Type	MFP*	Description
1	PB.14	I/O	MFP0	General purpose digital I/O pin.
	INT0	I	MFP1	External interrupt0 input pin.
	UART0_nRTS	O	MFP2	Request to Send output pin for UART0.
	ADC_CH9	A	MFP3	ADC channel 9 analog input.
	BPWM1_CH0	I/O	MFP4	BPWM1 channel 0 output/capture input.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
2	PB.13	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH10	A	MFP3	ADC channel 10 analog input.
	USCI0_CTL1	I/O	MFP6	USCI0 CTL1 pin.
3	PB.12	I/O	MFP0	General purpose digital I/O pin.
	CLKO	O	MFP2	Clock Out
	ADC_CH11	A	MFP3	ADC channel 11 analog input.
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
4	PA.11	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SCL	I/O	MFP1	I^{2}C1 clock pin.
	BPWM0_CH5	I/O	MFP4	BPWM0 channel 5 output/capture input.
	TM0	I/O	MFP5	Timer0event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
5	PA.10	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SDA	I/O	MFP1	I^{2}C1 data input/output pin.
	BPWM0_CH4	I/O	MFP4	BPWM0 channel 4 output/capture input.
	PWM0_BRAKE0	I	MFP5	Brake input pin 0 of PWM0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
6	PD.8	I/O	MFP0	General purpose digital I/O pin.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
7	PD.9	I/O	MFP0	General purpose digital I/O pin.
	PWM0_BRAKE1	I	MFP5	Brake input pin 1 of PWM0.
8	PD.10	I/O	MFP0	General purpose digital I/O pin.
	CLKO	O	MFP1	Clock Out

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

Pin No.	Pin Name	Type	MFP*	Description
	BPWM0_CH5	I/O	MFP4	BPWM0 channel 5 output/capture input.
9	PD.11	I/O	MFP0	General purpose digital I/O pin.
	INT1	I	MFP1	External interrupt1 input pin.
	BPWM0_CH4	I/O	MFP4	BPWM0 channel 4 output/capture input.
10	PB.4	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH3	I/O	MFP4	BPWM0 channel 3 output/capture input.
	TM2_EXT	I	MFP5	Timer2 external counter input
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
11	PB.5	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH2	I/O	MFP4	BPWM0 channel 2 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
12	PB.6	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH1	I/O	MFP4	BPWM0 channel 1 output/capture input.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
13	PB.7	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH0	I/O	MFP4	BPWM0 channel 0 output/capture input.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin.
14	LDO_CAP	A	MFP0	LDO output pin.
15	V_{DD}	A	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital function.
16	V_{SS}	A	MFP0	Ground pin for digital circuit.
17	USB_VBUS	A	MFP0	Power supply from USB host or HUB.
18	USB_VDD33_CAP	A	MFP0	Internal power regulator output 3.3V decoupling pin.
19	USB_D-	I	MFP0	USB differential signal D-.
20	USB_D+	I	MFP0	USB differential signal D+.
21	PB.0	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP1	Data receiver input pin for UART0.
	PWM1_CH0	I/O	MFP4	PWM1 channel0 output/capture input.
22	PB.1	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP1	Data transmitter output pin for UART0.

Pin No.	Pin Name	Type	MFP*	Description
	PWM1_CH1	I/O	MFP4	PWM1 channel1 output/capture input.
23	PB.2	I/O	MFP0	General purpose digital I/O pin.
	UART0_nRTS	O	MFP1	Request to Send output pin for UART0.
	TM2_EXT	I	MFP2	Timer2 external counter input
	PWM1_CH2	I/O	MFP4	PWM1 channel2 output/capture input.
24	PB.3	I/O	MFP0	General purpose digital I/O pin.
	UART0_nCTS	I	MFP1	Clear to Send input pin for UART0.
	TM3_EXT	I	MFP2	Timer3 external counter input
	PWM1_CH3	I/O	MFP4	PWM1 channel3 output/capture input.
25	PC.5	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP2	Data transmitter output pin for UART0.
	PWM1_CH5	I/O	MFP4	PWM1 channel5 output/capture input.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
26	PC.4	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP2	Data receiver input pin for UART0.
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM1_CH4	I/O	MFP4	PWM1 channel4 output/capture input.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
27	PC.3	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP1	SPI0 MOSI (Master Out, Slave In) pin.
	I2C1_SDA	I/O	MFP3	I^{2}C1 data input/output pin.
	PWM1_CH3	I/O	MFP4	PWM1 channel3 output/capture input.
	UART0_nRTS	O	MFP6	Request to Send output pin for UART0.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
28	PC.2	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP1	SPI0 MISO (Master In, Slave Out) pin.
	I2C1_SCL	I/O	MFP3	I^{2}C1 clock pin.
	PWM1_CH2	I/O	MFP4	PWM1 channel2 output/capture input.
	UART0_nCTS	I	MFP6	Clear to Send input pin for UART0.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
29	PC.1	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP1	SPI0 serial clock pin.
	PWM1_CH1	I/O	MFP4	PWM1 channel1 output/capture input.
	UART0_TXD	O	MFP6	Data transmitter output pin for UART0.

Pin No.	Pin Name	Type	MFP*	Description
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
30	PC.0	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS	I/O	MFP1	SPI0 slave select pin.
	PWM1_CH0	I/O	MFP4	PWM1 channel0 output/capture input.
	TM2	I/O	MFP5	Timer2 event counter input / toggle output
	UART0_RXD	I	MFP6	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin.
31	PB.10	I/O	MFP0	General purpose digital I/O pin.
	TM2	I/O	MFP1	Timer2 event counter input / toggle output
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM0_CH5	I/O	MFP4	PWM0 channel5 output/capture input.
32	PB.9	I/O	MFP0	General purpose digital I/O pin.
	TM1	I/O	MFP1	Timer1 event counter input / toggle output
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM0_CH4	I/O	MFP4	PWM0 channel4 output/capture input.
33	PE.2	I/O	MFP0	General purpose digital I/O pin.
	INT1	I	MFP1	External interrupt1 input pin.
	TM0_EXT	I	MFP5	Timer0 external counter input
	I2C0_SCL	I/O	MFP6	I^{2}C0 clock pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
34	PC.13	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP2	PWM0 channel3 output/capture input.
	CLKO	O	MFP3	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
35	PC.12	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP2	PWM0 channel2 output/capture input.
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	CLKO	O	MFP4	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SCL	I/O	MFP6	I^{2}C0 clock pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
36	PC.11	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP3	SPI0 MOSI (Master Out, Slave In) pin.

Pin No.	Pin Name	Type	MFP*	Description
	PWM0_CH1	I/O	MFP4	PWM0 channel1 output/capture input.
	TM1	I/O	MFP5	Timer1 event counter input / toggle output
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
37	PC.10	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP3	SPI0 MISO (Master In, Slave Out) pin.
	PWM0_CH0	I/O	MFP4	PWM0 channel0 output/capture input.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
38	PE.1	I/O	MFP0	General purpose digital I/O pin.
	STADC	I	MFP2	ADC external trigger input.
	CLKO	O	MFP3	Clock Out
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
39	PC.9	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP3	SPI0 serial clock pin.
	PWM0_CH5	I/O	MFP4	PWM0 channel5 output/capture input.
	PWM0_BRAKE1	I	MFP5	Brake input pin 1 of PWM0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin
40	PC.8	I/O	MFP0	General purpose digital I/O pin.
	STADC	I	MFP2	ADC external trigger input.
	SPI0_SS	I/O	MFP3	SPI0 slave select pin.
	PWM0_CH4	I/O	MFP4	PWM0 channel4 output/capture input.
	PWM1_BRAKE0	I	MFP5	Brake input pin 0 of PWM1.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
41	PA.15	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP1	PWM0 channel3 output/capture input.
	SPI_I2SMCLK	O	MFP2	I2S0 master clock output pin.
	CLKO	O	MFP3	Clock Out
	PWM1_BRAKE1	I	MFP4	Brake input pin 1 of PWM1.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
42	PE.0	I/O	MFP0	General purpose digital I/O pin.
	INT0	I	MFP1	External interrupt0 input pin.
	CLKO	O	MFP3	Clock Out
	PWM0_CH3	I/O	MFP4	PWM0 channel3 output/capture input.

Pin No.	Pin Name	Type	MFP*	Description
	TM1_EXT	I	MFP5	Timer1 external counter input
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
43	PA.14	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP1	PWM0 channel2 output/capture input.
	UART0_nCTS	I	MFP3	Clear to Send input pin for UART0.
	PWM0_BRAKE0	I	MFP4	Brake input pin 0 of PWM0.
44	PA.13	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH1	I/O	MFP1	PWM0 channel1 output/capture input.
	I2C1_SDA	I/O	MFP2	I^{2}C1 data input/output pin.
	UART0_TXD	O	MFP3	Data transmitter output pin for UART0.
45	PA.12	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH0	I/O	MFP1	PWM0 channel0 output/capture input.
	I2C1_SCL	I/O	MFP2	I^{2}C1 clock pin.
	UART0_RXD	I	MFP3	Data receiver input pin for UART0.
46	PF.4	I/O	MFP0	General purpose digital I/O pin.
	ICE_DAT	I/O	MFP1	Serial wired debugger data pin
	I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
	UART0_TXD	O	MFP3	Data transmitter output pin for UART0.
	PWM0_CH3	I/O	MFP4	PWM0 channel3 output/capture input.
47	PF.5	I/O	MFP0	General purpose digital I/O pin.
	ICE_CLK	I	MFP1	Serial wired debugger clock pin
	I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.
	UART0_RXD	I	MFP3	Data receiver input pin for UART0.
48	AV$_{DD}$	A	MFP0	Power supply for internal analog circuit.
49	PD.0	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH0	A	MFP3	ADC channel 0 analog input.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
50	PD.1	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH1	A	MFP3	ADC channel 1 analog input.
	TM0_EXT	I	MFP4	Timer0 external counter input
	UART0_RXD	I	MFP5	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.

Pin No.	Pin Name	Type	MFP*	Description
	SPI0_CLK	I/O	MFP7	SPI0 serial clock pin.
51	PD.2	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH2	A	MFP3	ADC channel 2 analog input.
	TM3	I/O	MFP4	Timer3 event counter input / toggle output
	UART0_TXD	O	MFP5	Data transmitter output pin for UART0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
	SPI0_MISO	I/O	MFP7	SPI0 MISO (Master In, Slave Out) pin.
52	PD.3	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH3	A	MFP3	ADC channel 3 analog input.
	TM1_EXT	I	MFP4	Timer1 external counter input
	UART0_nCTS	I	MFP5	Clear to Send input pin for UART0.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
	SPI0_MOSI	I/O	MFP7	SPI0 MOSI (Master Out, Slave In) pin.
53	PD.4	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH4	A	MFP2	ADC channel 4 analog input.
	BPWM1_CH5	I/O	MFP4	BPWM1 channel 5 output/capture input.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
54	PD.5	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH5	A	MFP2	ADC channel 5 analog input.
	BPWM1_CH4	I/O	MFP4	BPWM1 channel 4 output/capture input.
55	PB.15	I/O	MFP0	General purpose digital I/O pin.
	INT1	I	MFP1	External interrupt1 input pin.
	TM0_EXT	I	MFP2	Timer0 external counter input
	BPWM1_CH5	I/O	MFP4	BPWM1 channel 5 output/capture input.
56	PF.0	I/O	MFP0	General purpose digital I/O pin.
	XT_OUT	O	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal output pin.
	BPWM1_CH3	I/O	MFP4	BPWM1 channel 3 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
57	PF.1	I/O	MFP0	General purpose digital I/O pin.
	XT_IN	I	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal input pin.
	BPWM1_CH2	I/O	MFP4	BPWM1 channel 2 output/capture input.

Pin No.	Pin Name	Type	MFP*	Description
	TM1_EXT	I	MFP5	Timer1 external counter input
58	nRESET	I	MFP0	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state.
59	V_{SS}	A	MFP0	Ground pin for digital circuit.
60	V_{DD}	A	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital function.
61	PF.2	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
	ADC_CH6	A	MFP3	ADC channel 6 analog input.
	BPWM1_CH3	I/O	MFP4	BPWM1 channel 3 output/capture input.
62	PF.3	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.
	ADC_CH7	A	MFP3	ADC channel 7 analog input.
	BPWM1_CH2	I/O	MFP4	BPWM1 channel 2 output/capture input.
63	V_{SS}	A	MFP0	Ground pin for digital circuit.
64	PB.8	I/O	MFP0	General purpose digital I/O pin.
	TM0	I/O	MFP1	Timer0event counter input / toggle output
	ADC_CH8	A	MFP3	ADC channel 8 analog input.
	BPWM1_CH1	I/O	MFP4	BPWM1 channel 1 output/capture input.

Table 4.3-3 NUC121 USB Series LQFP64 Pin Description

4.3.4 NUC125 USB Series QFN33 Pin Description

MFP* = Multi-function pin. (Refer to section SYS_GPx_MFPL and SYS_GPx_MFPH)

PA.10 MFP5 means SYS_GPA_MFPH[11:8]=0x5.

PC.0 MFP0 means SYS_GPC_MFPL[3:0]=0x0.

Pin No.	Pin Name	Type	MFP*	Description
1	PB.14	I/O	MFP0	General purpose digital I/O pin.
	INT0	I	MFP1	External interrupt0 input pin.
	UART0_nRTS	O	MFP2	Request to Send output pin for UART0.
	ADC_CH9	A	MFP3	ADC channel 9 analog input.
	BPWM1_CH0	I/O	MFP4	BPWM1 channel 0 output/capture input.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
2	PA.11	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SCL	I/O	MFP1	I^{2}C1 clock pin.
	BPWM0_CH5	I/O	MFP4	BPWM0 channel 5 output/capture input.
	TM0	I/O	MFP5	Timer0event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
3	PA.10	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SDA	I/O	MFP1	I^{2}C1 data input/output pin.
	BPWM0_CH4	I/O	MFP4	BPWM0 channel 4 output/capture input.
	PWM0_BRAKE0	I	MFP5	Brake input pin 0 of PWM0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
4	PB.4	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH3	I/O	MFP4	BPWM0 channel 3 output/capture input.
	TM2_EXT	I	MFP5	Timer2 external counter input
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
5	PB.5	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH2	I/O	MFP4	BPWM0 channel 2 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
6	LDO_CAP	A	MFP0	LDO output pin.
7	V_{DD}	A	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital function.
8	V_{SS}	A	MFP0	Ground pin for digital circuit.

Pin No.	Pin Name	Type	MFP*	Description
9	USB_VBUS	A	MFP0	Power supply from USB host or HUB.
10	USB_VDD33_CAP	A	MFP0	Internal power regulator output 3.3V decoupling pin.
11	USB_D-	I	MFP0	USB differential signal D-.
12	USB_D+	I	MFP0	USB differential signal D+.
13	PC.3	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP1	SPI0 MOSI (Master Out, Slave In) pin.
	I2C1_SDA	I/O	MFP3	I^{2}C1 data input/output pin.
	PWM1_CH3	I/O	MFP4	PWM1 channel3 output/capture input.
	UART0_nRTS	O	MFP6	Request to Send output pin for UART0.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
14	PC.2	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP1	SPI0 MISO (Master In, Slave Out) pin.
	I2C1_SCL	I/O	MFP3	I^{2}C1 clock pin.
	PWM1_CH2	I/O	MFP4	PWM1 channel2 output/capture input.
	UART0_nCTS	I	MFP6	Clear to Send input pin for UART0.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
15	PC.1	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP1	SPI0 serial clock pin.
	PWM1_CH1	I/O	MFP4	PWM1 channel1 output/capture input.
	UART0_TXD	O	MFP6	Data transmitter output pin for UART0.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
16	PC.0	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS	I/O	MFP1	SPI0 slave select pin.
	PWM1_CH0	I/O	MFP4	PWM1 channel0 output/capture input.
	TM2	I/O	MFP5	Timer2 event counter input / toggle output
	UART0_RXD	I	MFP6	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin.
17	PC.13	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP2	PWM0 channel3 output/capture input.
	CLKO	O	MFP3	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
18	PC.12	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP2	PWM0 channel2 output/capture input.

Pin No.	Pin Name	Type	MFP*	Description
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	CLKO	O	MFP4	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SCL	I/O	MFP6	I^{2}C0 clock pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
19	PC.11	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP3	SPI0 MOSI (Master Out, Slave In) pin.
	PWM0_CH1	I/O	MFP4	PWM0 channel1 output/capture input.
	TM1	I/O	MFP5	Timer1 event counter input / toggle output
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
20	PC.10	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP3	SPI0 MISO (Master In, Slave Out) pin.
	PWM0_CH0	I/O	MFP4	PWM0 channel0 output/capture input.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
21	PC.9	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP3	SPI0 serial clock pin.
	PWM0_CH5	I/O	MFP4	PWM0 channel5 output/capture input.
	PWM0_BRAKE1	I	MFP5	Brake input pin 1 of PWM0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin
22	PC.8	I/O	MFP0	General purpose digital I/O pin.
	STADC	I	MFP2	ADC external trigger input.
	SPI0_SS	I/O	MFP3	SPI0 slave select pin.
	PWM0_CH4	I/O	MFP4	PWM0 channel4 output/capture input.
	PWM1_BRAKE0	I	MFP5	Brake input pin 0 of PWM1.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
23	PF.4	I/O	MFP0	General purpose digital I/O pin.
	ICE_DAT	I/O	MFP1	Serial wired debugger data pin
	I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
	UART0_TXD	O	MFP3	Data transmitter output pin for UART0.
	PWM0_CH3	I/O	MFP4	PWM0 channel3 output/capture input.
24	PF.5	I/O	MFP0	General purpose digital I/O pin.
	ICE_CLK	I	MFP1	Serial wired debugger clock pin
	I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.

Pin No.	Pin Name	Type	MFP*	Description
	UART0_RXD	I	MFP3	Data receiver input pin for UART0.
25	AV_{DD}	A	MFP0	Power supply for internal analog circuit.
26	PD.1	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH1	A	MFP3	ADC channel 1 analog input.
	TM0_EXT	I	MFP4	Timer0 external counter input
	UART0_RXD	I	MFP5	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	SPI0_CLK	I/O	MFP7	SPI0 serial clock pin.
27	PD.2	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH2	A	MFP3	ADC channel 2 analog input.
	TM3	I/O	MFP4	Timer3 event counter input / toggle output
	UART0_TXD	O	MFP5	Data transmitter output pin for UART0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
	SPI0_MISO	I/O	MFP7	SPI0 MISO (Master In, Slave Out) pin.
28	PD.3	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH3	A	MFP3	ADC channel 3 analog input.
	TM1_EXT	I	MFP4	Timer1 external counter input
	UART0_nCTS	I	MFP5	Clear to Send input pin for UART0.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
	SPI0_MOSI	I/O	MFP7	SPI0 MOSI (Master Out, Slave In) pin.
29	PF.0	I/O	MFP0	General purpose digital I/O pin.
	XT_OUT	O	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal output pin.
	BPWM1_CH3	I/O	MFP4	BPWM1 channel 3 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
30	PF.1	I/O	MFP0	General purpose digital I/O pin.
	XT_IN	I	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal input pin.
	BPWM1_CH2	I/O	MFP4	BPWM1 channel 2 output/capture input.
	TM1_EXT	I	MFP5	Timer1 external counter input
31	nRESET	I	MFP0	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state.
32	V_{DDIO}	A	MFP0	Power supply for PB.14, PA.11, PA.10, PB.4 and PB.5.
33	V_{SS}	A	MFP0	Ground pin for digital circuit.

Table 4.3-4 NUC121 USB Series QFN33 Pin Description

4.3.5 NUC125 USB Series LQFP48 Pin Description

MFP* = Multi-function pin. (Refer to section SYS_GPx_MFPL and SYS_GPx_MFPH)

PA.10 MFP5 means SYS_GPA_MFPH[11:8]=0x5.

PC.0 MFP0 means SYS_GPC_MFPL[3:0]=0x0..

Pin No.	Pin Name	Type	MFP*	Description
1	V_{SS}	A	MFP0	Ground pin for digital circuit.
2	V_{DDIO}	A	MFP0	Power supply for PB.14, PA.11, PA.10, PB.4, PB.5, PB.6 and PB.7.
3	PB.14	I/O	MFP0	General purpose digital I/O pin.
	INT0	I	MFP1	External interrupt0 input pin.
	UART0_nRTS	O	MFP2	Request to Send output pin for UART0.
	ADC_CH9	A	MFP3	ADC channel 9 analog input.
	BPWM1_CH0	I/O	MFP4	BPWM1 channel 0 output/capture input.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
4	PA.11	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SCL	I/O	MFP1	I^2C1 clock pin.
	BPWM0_CH5	I/O	MFP4	BPWM0 channel 5 output/capture input.
	TM0	I/O	MFP5	Timer0event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
5	PA.10	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SDA	I/O	MFP1	I^2C1 data input/output pin.
	BPWM0_CH4	I/O	MFP4	BPWM0 channel 4 output/capture input.
	PWM0_BRAKE0	I	MFP5	Brake input pin 0 of PWM0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
6	PB.4	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH3	I/O	MFP4	BPWM0 channel 3 output/capture input.
	TM2_EXT	I	MFP5	Timer2 external counter input
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
7	PB.5	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH2	I/O	MFP4	BPWM0 channel 2 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.

Pin No.	Pin Name	Type	MFP*	Description
8	PB.6	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH1	I/O	MFP4	BPWM0 channel 1 output/capture input.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
9	PB.7	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH0	I/O	MFP4	BPWM0 channel 0 output/capture input.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin.
10	LDO_CAP	A	MFP0	LDO output pin.
11	V_{DD}	A	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital function.
12	V_{SS}	A	MFP0	Ground pin for digital circuit.
13	USB_VBUS	A	MFP0	Power supply from USB host or HUB.
14	USB_VDD33_CAP	A	MFP0	Internal power regulator output 3.3V decoupling pin.
15	USB_D-	I	MFP0	USB differential signal D-.
16	USB_D+	I	MFP0	USB differential signal D+.
17	PC.5	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP2	Data transmitter output pin for UART0.
	PWM1_CH5	I/O	MFP4	PWM1 channel5 output/capture input.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
18	PC.4	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP2	Data receiver input pin for UART0.
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM1_CH4	I/O	MFP4	PWM1 channel4 output/capture input.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
19	PC.3	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP1	SPI0 MOSI (Master Out, Slave In) pin.
	I2C1_SDA	I/O	MFP3	I^{2}C1 data input/output pin.
	PWM1_CH3	I/O	MFP4	PWM1 channel3 output/capture input.
	UART0_nRTS	O	MFP6	Request to Send output pin for UART0.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
20	PC.2	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP1	SPI0 MISO (Master In, Slave Out) pin.
	I2C1_SCL	I/O	MFP3	I^{2}C1 clock pin.
	PWM1_CH2	I/O	MFP4	PWM1 channel2 output/capture input.

Pin No.	Pin Name	Type	MFP*	Description
	UART0_nCTS	I	MFP6	Clear to Send input pin for UART0.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
21	PC.1	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP1	SPI0 serial clock pin.
	PWM1_CH1	I/O	MFP4	PWM1 channel1 output/capture input.
	UART0_TXD	O	MFP6	Data transmitter output pin for UART0.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
22	PC.0	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS	I/O	MFP1	SPI0 slave select pin.
	PWM1_CH0	I/O	MFP4	PWM1 channel0 output/capture input.
	TM2	I/O	MFP5	Timer2 event counter input / toggle output
	UART0_RXD	I	MFP6	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin.
23	PB.10	I/O	MFP0	General purpose digital I/O pin.
	TM2	I/O	MFP1	Timer2 event counter input / toggle output
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM0_CH5	I/O	MFP4	PWM0 channel5 output/capture input.
24	PB.9	I/O	MFP0	General purpose digital I/O pin.
	TM1	I/O	MFP1	Timer1 event counter input / toggle output
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM0_CH4	I/O	MFP4	PWM0 channel4 output/capture input.
25	PC.13	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP2	PWM0 channel3 output/capture input.
	CLKO	O	MFP3	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SDA	I/O	MFP6	I^2C0 data input/output pin.
26	PC.12	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP2	PWM0 channel2 output/capture input.
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	CLKO	O	MFP4	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SCL	I/O	MFP6	I^2C0 clock pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
27	PC.11	I/O	MFP0	General purpose digital I/O pin.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

Pin No.	Pin Name	Type	MFP*	Description
	SPI0_MOSI	I/O	MFP3	SPI0 MOSI (Master Out, Slave In) pin.
	PWM0_CH1	I/O	MFP4	PWM0 channel1 output/capture input.
	TM1	I/O	MFP5	Timer1 event counter input / toggle output
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
28	PC.10	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP3	SPI0 MISO (Master In, Slave Out) pin.
	PWM0_CH0	I/O	MFP4	PWM0 channel0 output/capture input.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
29	PC.9	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP3	SPI0 serial clock pin.
	PWM0_CH5	I/O	MFP4	PWM0 channel5 output/capture input.
	PWM0_BRAKE1	I	MFP5	Brake input pin 1 of PWM0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin
30	PC.8	I/O	MFP0	General purpose digital I/O pin.
	STADC	I	MFP2	ADC external trigger input.
	SPI0_SS	I/O	MFP3	SPI0 slave select pin.
	PWM0_CH4	I/O	MFP4	PWM0 channel4 output/capture input.
	PWM1_BRAKE0	I	MFP5	Brake input pin 0 of PWM1.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
31	PA.15	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP1	PWM0 channel3 output/capture input.
	SPI_I2SMCLK	O	MFP2	I2S0 master clock output pin.
	CLKO	O	MFP3	Clock Out
	PWM1_BRAKE1	I	MFP4	Brake input pin 1 of PWM1.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
32	PA.14	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP1	PWM0 channel2 output/capture input.
	UART0_nCTS	I	MFP3	Clear to Send input pin for UART0.
	PWM0_BRAKE0	I	MFP4	Brake input pin 0 of PWM0.
33	PA.13	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH1	I/O	MFP1	PWM0 channel1 output/capture input.
	I2C1_SDA	I/O	MFP2	I^{2}C1 data input/output pin.
	UART0_TXD	O	MFP3	Data transmitter output pin for UART0.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

Pin No.	Pin Name	Type	MFP*	Description
34	PA.12	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH0	I/O	MFP1	PWM0 channel0 output/capture input.
	I2C1_SCL	I/O	MFP2	I^{2}C1 clock pin.
	UART0_RXD	I	MFP3	Data receiver input pin for UART0.
35	PF.4	I/O	MFP0	General purpose digital I/O pin.
	ICE_DAT	I/O	MFP1	Serial wired debugger data pin
	I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
	UART0_TXD	O	MFP3	Data transmitter output pin for UART0.
	PWM0_CH3	I/O	MFP4	PWM0 channel3 output/capture input.
36	PF.5	I/O	MFP0	General purpose digital I/O pin.
	ICE_CLK	I	MFP1	Serial wired debugger clock pin
	I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.
	UART0_RXD	I	MFP3	Data receiver input pin for UART0.
37	AV$_{DD}$	A	MFP0	Power supply for internal analog circuit.
38	PD.0	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH0	A	MFP3	ADC channel 0 analog input.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
39	PD.1	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH1	A	MFP3	ADC channel 1 analog input.
	TM0_EXT	I	MFP4	Timer0 external counter input
	UART0_RXD	I	MFP5	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	SPI0_CLK	I/O	MFP7	SPI0 serial clock pin.
40	PD.2	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH2	A	MFP3	ADC channel 2 analog input.
	TM3	I/O	MFP4	Timer3 event counter input / toggle output
	UART0_TXD	O	MFP5	Data transmitter output pin for UART0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
	SPI0_MISO	I/O	MFP7	SPI0 MISO (Master In, Slave Out) pin.
41	PD.3	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH3	A	MFP3	ADC channel 3 analog input.
	TM1_EXT	I	MFP4	Timer1 external counter input

Pin No.	Pin Name	Type	MFP*	Description
	UART0_nCTS	I	MFP5	Clear to Send input pin for UART0.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
	SPI0_MOSI	I/O	MFP7	SPI0 MOSI (Master Out, Slave In) pin.
42	PD.4	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH4	A	MFP2	ADC channel 4 analog input.
	BPWM1_CH5	I/O	MFP4	BPWM1 channel 5 output/capture input.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
43	PD.5	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH5	A	MFP2	ADC channel 5 analog input.
	BPWM1_CH4	I/O	MFP4	BPWM1 channel 4 output/capture input.
44	PF.0	I/O	MFP0	General purpose digital I/O pin.
	XT_OUT	O	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal output pin.
	BPWM1_CH3	I/O	MFP4	BPWM1 channel 3 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
45	PF.1	I/O	MFP0	General purpose digital I/O pin.
	XT_IN	I	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal input pin.
	BPWM1_CH2	I/O	MFP4	BPWM1 channel 2 output/capture input.
	TM1_EXT	I	MFP5	Timer1 external counter input
46	nRESET	I	MFP0	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state.
47	PF.2	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
	ADC_CH6	A	MFP3	ADC channel 6 analog input.
	BPWM1_CH3	I/O	MFP4	BPWM1 channel 3 output/capture input.
48	PF.3	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.
	ADC_CH7	A	MFP3	ADC channel 7 analog input.
	BPWM1_CH2	I/O	MFP4	BPWM1 channel 2 output/capture input.

Table 4.3-5 NUC125 USB Series LQFP48 Pin Description

4.3.6 NUC125 USB Series LQFP64 Pin Description

MFP* = Multi-function pin. (Refer to section SYS_GPx_MFPL and SYS_GPx_MFPH)

PA.10 MFP5 means SYS_GPA_MFPH[11:8]=0x5.

PC.0 MFP0 means SYS_GPC_MFPL[3:0]=0x0.

Pin No.	Pin Name	Type	MFP*	Description
1	PB.14	I/O	MFP0	General purpose digital I/O pin.
	INT0	I	MFP1	External interrupt0 input pin.
	UART0_nRTS	O	MFP2	Request to Send output pin for UART0.
	ADC_CH9	A	MFP3	ADC channel 9 analog input.
	BPWM1_CH0	I/O	MFP4	BPWM1 channel 0 output/capture input.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
2	PB.13	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH10	A	MFP3	ADC channel 10 analog input.
	USCI0_CTL1	I/O	MFP6	USCI0 CTL1 pin.
3	PB.12	I/O	MFP0	General purpose digital I/O pin.
	CLKO	O	MFP2	Clock Out
	ADC_CH11	A	MFP3	ADC channel 11 analog input.
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
4	PA.11	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SCL	I/O	MFP1	I^{2}C1 clock pin.
	BPWM0_CH5	I/O	MFP4	BPWM0 channel 5 output/capture input.
	TM0	I/O	MFP5	Timer0event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
5	PA.10	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SDA	I/O	MFP1	I^{2}C1 data input/output pin.
	BPWM0_CH4	I/O	MFP4	BPWM0 channel 4 output/capture input.
	PWM0_BRAKE0	I	MFP5	Brake input pin 0 of PWM0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
6	PD.8	I/O	MFP0	General purpose digital I/O pin.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
7	PD.9	I/O	MFP0	General purpose digital I/O pin.
	PWM0_BRAKE1	I	MFP5	Brake input pin 1 of PWM0.
8	PD.10	I/O	MFP0	General purpose digital I/O pin.
	CLKO	O	MFP1	Clock Out

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

Pin No.	Pin Name	Type	MFP*	Description
	BPWM0_CH5	I/O	MFP4	BPWM0 channel 5 output/capture input.
9	PD.11	I/O	MFP0	General purpose digital I/O pin.
	INT1	I	MFP1	External interrupt1 input pin.
	BPWM0_CH4	I/O	MFP4	BPWM0 channel 4 output/capture input.
10	PB.4	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH3	I/O	MFP4	BPWM0 channel 3 output/capture input.
	TM2_EXT	I	MFP5	Timer2 external counter input
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
11	PB.5	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH2	I/O	MFP4	BPWM0 channel 2 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
12	PB.6	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH1	I/O	MFP4	BPWM0 channel 1 output/capture input.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
13	PB.7	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH0	I/O	MFP4	BPWM0 channel 0 output/capture input.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin.
14	LDO_CAP	A	MFP0	LDO output pin.
15	V_{DD}	A	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital function.
16	V_{SS}	A	MFP0	Ground pin for digital circuit.
17	USB_VBUS	A	MFP0	Power supply from USB host or HUB.
18	USB_VDD33_CAP	A	MFP0	Internal power regulator output 3.3V decoupling pin.
19	USB_D-	I	MFP0	USB differential signal D-.
20	USB_D+	I	MFP0	USB differential signal D+.
21	PB.0	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP1	Data receiver input pin for UART0.
	PWM1_CH0	I/O	MFP4	PWM1 channel0 output/capture input.
22	PB.1	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP1	Data transmitter output pin for UART0.

Pin No.	Pin Name	Type	MFP*	Description
	PWM1_CH1	I/O	MFP4	PWM1 channel1 output/capture input.
23	PB.2	I/O	MFP0	General purpose digital I/O pin.
	UART0_nRTS	O	MFP1	Request to Send output pin for UART0.
	TM2_EXT	I	MFP2	Timer2 external counter input
	PWM1_CH2	I/O	MFP4	PWM1 channel2 output/capture input.
24	PB.3	I/O	MFP0	General purpose digital I/O pin.
	UART0_nCTS	I	MFP1	Clear to Send input pin for UART0.
	TM3_EXT	I	MFP2	Timer3 external counter input
	PWM1_CH3	I/O	MFP4	PWM1 channel3 output/capture input.
25	PC.5	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP2	Data transmitter output pin for UART0.
	PWM1_CH5	I/O	MFP4	PWM1 channel5 output/capture input.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
26	PC.4	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP2	Data receiver input pin for UART0.
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM1_CH4	I/O	MFP4	PWM1 channel4 output/capture input.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
27	PC.3	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP1	SPI0 MOSI (Master Out, Slave In) pin.
	I2C1_SDA	I/O	MFP3	I^{2}C1 data input/output pin.
	PWM1_CH3	I/O	MFP4	PWM1 channel3 output/capture input.
	UART0_nRTS	O	MFP6	Request to Send output pin for UART0.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
28	PC.2	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP1	SPI0 MISO (Master In, Slave Out) pin.
	I2C1_SCL	I/O	MFP3	I^{2}C1 clock pin.
	PWM1_CH2	I/O	MFP4	PWM1 channel2 output/capture input.
	UART0_nCTS	I	MFP6	Clear to Send input pin for UART0.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
29	PC.1	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP1	SPI0 serial clock pin.
	PWM1_CH1	I/O	MFP4	PWM1 channel1 output/capture input.
	UART0_TXD	O	MFP6	Data transmitter output pin for UART0.

Pin No.	Pin Name	Type	MFP*	Description
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
30	PC.0	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS	I/O	MFP1	SPI0 slave select pin.
	PWM1_CH0	I/O	MFP4	PWM1 channel0 output/capture input.
	TM2	I/O	MFP5	Timer2 event counter input / toggle output
	UART0_RXD	I	MFP6	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin.
31	PB.10	I/O	MFP0	General purpose digital I/O pin.
	TM2	I/O	MFP1	Timer2 event counter input / toggle output
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM0_CH5	I/O	MFP4	PWM0 channel5 output/capture input.
32	PB.9	I/O	MFP0	General purpose digital I/O pin.
	TM1	I/O	MFP1	Timer1 event counter input / toggle output
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	PWM0_CH4	I/O	MFP4	PWM0 channel4 output/capture input.
33	PE.2	I/O	MFP0	General purpose digital I/O pin.
	INT1	I	MFP1	External interrupt1 input pin.
	TM0_EXT	I	MFP5	Timer0 external counter input
	I2C0_SCL	I/O	MFP6	I^2C0 clock pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
34	PC.13	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP2	PWM0 channel3 output/capture input.
	CLKO	O	MFP3	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SDA	I/O	MFP6	I^2C0 data input/output pin.
35	PC.12	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP2	PWM0 channel2 output/capture input.
	SPI0_I2SMCLK	O	MFP3	I2S0 master clock output pin.
	CLKO	O	MFP4	Clock Out
	INT0	I	MFP5	External interrupt0 input pin.
	I2C0_SCL	I/O	MFP6	I^2C0 clock pin.
	USCI0_CTL1	I/O	MFP7	USCI0 CTL1 pin.
36	PC.11	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP3	SPI0 MOSI (Master Out, Slave In) pin.

Pin No.	Pin Name	Type	MFP*	Description
	PWM0_CH1	I/O	MFP4	PWM0 channel1 output/capture input.
	TM1	I/O	MFP5	Timer1 event counter input / toggle output
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
37	PC.10	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP3	SPI0 MISO (Master In, Slave Out) pin.
	PWM0_CH0	I/O	MFP4	PWM0 channel0 output/capture input.
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
38	PE.1	I/O	MFP0	General purpose digital I/O pin.
	STADC	I	MFP2	ADC external trigger input.
	CLKO	O	MFP3	Clock Out
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
	USCI0_DAT1	I/O	MFP7	USCI0 DAT1 pin.
39	PC.9	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP3	SPI0 serial clock pin.
	PWM0_CH5	I/O	MFP4	PWM0 channel5 output/capture input.
	PWM0_BRAKE1	I	MFP5	Brake input pin 1 of PWM0.
	USCI0_CLK	I/O	MFP7	USCI0 clock pin
40	PC.8	I/O	MFP0	General purpose digital I/O pin.
	STADC	I	MFP2	ADC external trigger input.
	SPI0_SS	I/O	MFP3	SPI0 slave select pin.
	PWM0_CH4	I/O	MFP4	PWM0 channel4 output/capture input.
	PWM1_BRAKE0	I	MFP5	Brake input pin 0 of PWM1.
	USCI0_CTL0	I/O	MFP7	USCI0 CTL0 pin
41	PA.15	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH3	I/O	MFP1	PWM0 channel3 output/capture input.
	SPI_I2SMCLK	O	MFP2	I2S0 master clock output pin.
	CLKO	O	MFP3	Clock Out
	PWM1_BRAKE1	I	MFP4	Brake input pin 1 of PWM1.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
42	PE.0	I/O	MFP0	General purpose digital I/O pin.
	INT0	I	MFP1	External interrupt0 input pin.
	CLKO	O	MFP3	Clock Out
	PWM0_CH3	I/O	MFP4	PWM0 channel3 output/capture input.

Pin No.	Pin Name	Type	MFP*	Description
	TM1_EXT	I	MFP5	Timer1 external counter input
	USCI0_DAT0	I/O	MFP7	USCI0 DAT0 pin.
43	PA.14	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH2	I/O	MFP1	PWM0 channel2 output/capture input.
	UART0_nCTS	I	MFP3	Clear to Send input pin for UART0.
	PWM0_BRAKE0	I	MFP4	Brake input pin 0 of PWM0.
44	PA.13	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH1	I/O	MFP1	PWM0 channel1 output/capture input.
	I2C1_SDA	I/O	MFP2	I^{2}C1 data input/output pin.
	UART0_TXD	O	MFP3	Data transmitter output pin for UART0.
45	PA.12	I/O	MFP0	General purpose digital I/O pin.
	PWM0_CH0	I/O	MFP1	PWM0 channel0 output/capture input.
	I2C1_SCL	I/O	MFP2	I^{2}C1 clock pin.
	UART0_RXD	I	MFP3	Data receiver input pin for UART0.
46	PF.4	I/O	MFP0	General purpose digital I/O pin.
	ICE_DAT	I/O	MFP1	Serial wired debugger data pin
	I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
	UART0_TXD	O	MFP3	Data transmitter output pin for UART0.
	PWM0_CH3	I/O	MFP4	PWM0 channel3 output/capture input.
47	PF.5	I/O	MFP0	General purpose digital I/O pin.
	ICE_CLK	I	MFP1	Serial wired debugger clock pin
	I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.
	UART0_RXD	I	MFP3	Data receiver input pin for UART0.
48	AV$_{DD}$	A	MFP0	Power supply for internal analog circuit.
49	PD.0	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH0	A	MFP3	ADC channel 0 analog input.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
50	PD.1	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH1	A	MFP3	ADC channel 1 analog input.
	TM0_EXT	I	MFP4	Timer0 external counter input
	UART0_RXD	I	MFP5	Data receiver input pin for UART0.
	USCI0_CLK	I/O	MFP6	USCI0 clock pin.

Pin No.	Pin Name	Type	MFP*	Description
	SPI0_CLK	I/O	MFP7	SPI0 serial clock pin.
51	PD.2	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH2	A	MFP3	ADC channel 2 analog input.
	TM3	I/O	MFP4	Timer3 event counter input / toggle output
	UART0_TXD	O	MFP5	Data transmitter output pin for UART0.
	USCI0_DAT1	I/O	MFP6	USCI0 DAT1 pin.
	SPI0_MISO	I/O	MFP7	SPI0 MISO (Master In, Slave Out) pin.
52	PD.3	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH3	A	MFP3	ADC channel 3 analog input.
	TM1_EXT	I	MFP4	Timer1 external counter input
	UART0_nCTS	I	MFP5	Clear to Send input pin for UART0.
	USCI0_DAT0	I/O	MFP6	USCI0 DAT0 pin.
	SPI0_MOSI	I/O	MFP7	SPI0 MOSI (Master Out, Slave In) pin.
53	PD.4	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH4	A	MFP2	ADC channel 4 analog input.
	BPWM1_CH5	I/O	MFP4	BPWM1 channel 5 output/capture input.
	UART0_nRTS	O	MFP5	Request to Send output pin for UART0.
	USCI0_CTL0	I/O	MFP6	USCI0 CTL0 pin.
	SPI0_SS	I/O	MFP7	SPI0 slave select pin.
54	PD.5	I/O	MFP0	General purpose digital I/O pin.
	ADC_CH5	A	MFP2	ADC channel 5 analog input.
	BPWM1_CH4	I/O	MFP4	BPWM1 channel 4 output/capture input.
55	PB.15	I/O	MFP0	General purpose digital I/O pin.
	INT1	I	MFP1	External interrupt1 input pin.
	TM0_EXT	I	MFP2	Timer0 external counter input
	BPWM1_CH5	I/O	MFP4	BPWM1 channel 5 output/capture input.
56	PF.0	I/O	MFP0	General purpose digital I/O pin.
	XT_OUT	O	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal output pin.
	BPWM1_CH3	I/O	MFP4	BPWM1 channel 3 output/capture input.
	TM3	I/O	MFP5	Timer3 event counter input / toggle output
57	PF.1	I/O	MFP0	General purpose digital I/O pin.
	XT_IN	I	MFP1	External 4~24 MHz (high speed) or 32.768 kHz (low speed) crystal input pin.
	BPWM1_CH2	I/O	MFP4	BPWM1 channel 2 output/capture input.

Pin No.	Pin Name	Type	MFP*	Description
	TM1_EXT	I	MFP5	Timer1 external counter input
58	nRESET	I	MFP0	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state.
59	V_{SS}	A	MFP0	Ground pin for digital circuit.
60	V_{DD}	A	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital function.
61	PF.2	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SDA	I/O	MFP2	I^2C0 data input/output pin.
	ADC_CH6	A	MFP3	ADC channel 6 analog input.
	BPWM1_CH3	I/O	MFP4	BPWM1 channel 3 output/capture input.
62	PF.3	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SCL	I/O	MFP2	I^2C0 clock pin.
	ADC_CH7	A	MFP3	ADC channel 7 analog input.
	BPWM1_CH2	I/O	MFP4	BPWM1 channel 2 output/capture input.
63	V_{SS}	A	MFP0	Ground pin for digital circuit.
64	V_{DDIO}	A	MFP0	Power supply for PB.14, PB.13, PB.12, PA.11, PA.10, PD.8, PD.9, PD.10, PD.11, PB.4, PB.5, PB.6 and PB.7.

Table 4.3-6 NUC125 USB Series LQFP64 Pin Description

4.3.7 GPIO Multi-function Pin Summary

MFP* = Multi-function pin. (Refer to section SYS_GPx_MFPL and SYS_GPx_MFPH)

PA.10 MFP5 means SYS_GPA_MFPH[11:8]=0x5.

PC.0 MFP0 means SYS_GPC_MFPL[3:0]=0x0.

Group	Pin Name	GPIO	MFP*	Type	Description
ADC0	ADC0_CH0	PD.0	MFP3	A	ADC0 analog input.
	ADC0_CH1	PD.1	MFP3	A	ADC1 analog input.
	ADC0_CH2	PD.2	MFP3	A	ADC2 analog input.
	ADC0_CH3	PD.3	MFP3	A	ADC3 analog input.
	ADC0_CH4	PD.4	MFP2	A	ADC4 analog input.
	ADC0_CH5	PD.5	MFP2	A	ADC5 analog input.
	ADC0_CH6	PF.2	MFP3	A	ADC6 analog input.
	ADC0_CH7	PF.3	MFP3	A	ADC7 analog input.
	ADC0_CH8	PB.8	MFP3	A	ADC8 analog input.
	ADC0_CH9	PB.14	MFP3	A	ADC9 analog input.
	ADC0_CH10	PB.13	MFP3	A	ADC10 analog input.
	ADC0_CH11	PB.12	MFP3	A	ADC11 analog input.
	STADC	PC.8	MFP2	I	ADC external trigger input.
	STADC	PE.1	MFP2	I	ADC external trigger input
BPWM0	BPWM0_CH0	PB.7	MFP4	I/O	BPWM0 output/capture input.
	BPWM0_CH1	PB.6	MFP4	I/O	BPWM0 output/capture input.
	BPWM0_CH2	PB.5	MFP4	I/O	BPWM0 output/capture input.
	BPWM0_CH3	PB.4	MFP4	I/O	BPWM0 output/capture input.
	BPWM0_CH4	PA.10	MFP4	I/O	BPWM0 output/capture input.
	BPWM0_CH4	PD.11	MFP4	I/O	BPWM0 output/capture input.
	BPWM0_CH5	PA.11	MFP4	I/O	BPWM0 output/capture input.
	BPWM0_CH5	PD.10	MFP4	I/O	BPWM0 output/capture input.
BPWM1	BPWM1_CH0	PB.14	MFP4	I/O	BPWM1 output/capture input.
	BPWM1_CH1	PB.8	MFP4	I/O	BPWM1 output/capture input.
	BPWM1_CH2	PF.1	MFP4	I/O	BPWM1 output/capture input.
	BPWM1_CH2	PF.3	MFP4	I/O	BPWM1 output/capture input.
	BPWM1_CH3	PF.0	MFP4	I/O	BPWM1 output/capture input.
	BPWM1_CH3	PF.2	MFP4	I/O	BPWM1 output/capture input.
	BPWM1_CH4	PD.5	MFP4	I/O	BPWM1 output/capture input.
	BPWM1_CH5	PB.15	MFP4	I/O	BPWM1 output/capture input.

Group	Pin Name	GPIO	MFP*	Type	Description
	BPWM1_CH5	PD.4	MFP4	I/O	BPWM1 output/capture input.
CLKO	CLKO	PA.15	MFP3	O	Clock Out.
	CLKO	PB.12	MFP2	O	Clock Out.
	CLKO	PC.12	MFP4	O	Clock Out.
	CLKO	PC.13	MFP3	O	Clock Out.
	CLKO	PD.10	MFP1	O	Clock Out.
	CLKO	PE.0	MFP3	O	Clock Out.
	CLKO	PE.1	MFP3	O	Clock Out.
I2C0	I2C0_SCL	PC.12	MFP6	I/O	I2C0 clock pin.
	I2C0_SCL	PE.2	MFP6	I/O	I2C0 clock pin.
	I2C0_SCL	PF.3	MFP2	I/O	I2C0 clock pin.
	I2C0_SCL	PF.5	MFP2	I/O	I2C0 clock pin.
	I2C0_SDA	PC.11	MFP6	I/O	I2C0 data input/output pin.
	I2C0_SDA	PC.13	MFP6	I/O	I2C0 data input/output pin.
	I2C0_SDA	PF.2	MFP2	I/O	I2C0 data input/output pin.
	I2C0_SDA	PF.4	MFP2	I/O	I2C0 data input/output pin.
I2C1	I2C1_SCL	PA.11	MFP1	I/O	I2C1 clock pin.
	I2C1_SCL	PA.12	MFP2	I/O	I2C1 clock pin.
	I2C1_SCL	PC.2	MFP3	I/O	I2C1 clock pin.
	I2C1_SDA	PA.10	MFP1	I/O	I2C1 data input/output pin.
	I2C1_SDA	PA.13	MFP2	I/O	I2C1 data input/output pin.
	I2C1_SDA	PC.3	MFP3	I/O	I2C1 data input/output pin.
ICE	ICE_CLK	PF.5	MFP1	I	Serial wired debugger clock pin.
	ICE_DAT	PF.4	MFP1	I/O	Serial wired debugger data pin.
INT0	INT0	PB.14	MFP1	I	External interrupt0 input pin.
	INT0	PC.12	MFP5	I	External interrupt0 input pin.
	INT0	PC.13	MFP5	I	External interrupt0 input pin.
	INT0	PE.0	MFP1	I	External interrupt0 input pin.
INT1	INT1	PB.15	MFP1	I	External interrupt1 input pin.
	INT1	PD.11	MFP1	I	External interrupt1 input pin.
	INT1	PE.2	MFP1	I	External interrupt1 input pin.
PWM0	PWM0_BRAKE0	PA.10	MFP5	I	PWM0 brake input 0.
	PWM0_BRAKE0	PA.14	MFP4	I	PWM0 brake input 0.
	PWM0_BRAKE1	PC.9	MFP5	I	PWM0 brake input 1.

Group	Pin Name	GPIO	MFP*	Type	Description
	PWM0_BRAKE1	PD.9	MFP5	I	PWM0 brake input 1.
	PWM0_CH0	PA.12	MFP1	I/O	PWM0 output/capture input.
	PWM0_CH0	PC.10	MFP4	I/O	PWM0 output/capture input.
	PWM0_CH1	PA.13	MFP1	I/O	PWM0 output/capture input.
	PWM0_CH1	PC.11	MFP4	I/O	PWM0 output/capture input.
	PWM0_CH2	PA.14	MFP1	I/O	PWM0 output/capture input.
	PWM0_CH2	PC.12	MFP2	I/O	PWM0 output/capture input.
	PWM0_CH3	PA.15	MFP1	I/O	PWM0 output/capture input.
	PWM0_CH3	PC.13	MFP2	I/O	PWM0 output/capture input.
	PWM0_CH3	PE.0	MFP4	I/O	PWM0 output/capture input.
	PWM0_CH3	PF.4	MFP4	I/O	PWM0 output/capture input.
	PWM0_CH4	PB.9	MFP4	I/O	PWM0 output/capture input.
	PWM0_CH4	PC.8	MFP4	I/O	PWM0 output/capture input.
	PWM0_CH5	PB.10	MFP4	I/O	PWM0 output/capture input.
	PWM0_CH5	PC.9	MFP4	I/O	PWM0 output/capture input.
PWM1	PWM1_BRAKE0	PC.8	MFP4	I	PWM1 brake input 0.
	PWM1_BRAKE1	PA.15	MFP4	I	PWM1 brake input 1.
	PWM1_CH0	PB.0	MFP4	I/O	PWM1 output/capture input.
	PWM1_CH0	PC.0	MFP4	I/O	PWM1 output/capture input.
	PWM1_CH1	PB.1	MFP4	I/O	PWM1 output/capture input.
	PWM1_CH1	PC.1	MFP4	I/O	PWM1 output/capture input.
	PWM1_CH2	PB.2	MFP4	I/O	PWM1 output/capture input.
	PWM1_CH2	PC.2	MFP4	I/O	PWM1 output/capture input.
	PWM1_CH3	PB.3	MFP4	I/O	PWM1 output/capture input.
	PWM1_CH3	PC.3	MFP4	I/O	PWM1 output/capture input.
	PWM1_CH4	PC.4	MFP4	I/O	PWM1 output/capture input.
	PWM1_CH5	PC.5	MFP4	I/O	PWM1 output/capture input.
SPI0	SPI0_CLK	PC.1	MFP1	I/O	SPI0 serial clock pin.
	SPI0_CLK	PC.9	MFP3	I/O	SPI0 serial clock pin.
	SPI0_CLK	PD.1	MFP7	I/O	SPI0 serial clock pin.
	SPI0_MISO0	PC.2	MFP1	I/O	SPI0 1st MISO (Master In, Slave Out) pin.
	SPI0_MISO0	PC.10	MFP3	I/O	SPI0 1st MISO (Master In, Slave Out) pin.
	SPI0_MISO0	PD.2	MFP7	I/O	SPI0 1st MISO (Master In, Slave Out) pin.
	SPI0_MOSI0	PC.3	MFP1	I/O	SPI0 1st MOSI (Master Out, Slave In) pin.

Group	Pin Name	GPIO	MFP*	Type	Description
	SPI0_MOSI0	PC.11	MFP3	I/O	SPI0 1st MOSI (Master Out, Slave In) pin.
	SPI0_MOSI0	PD.3	MFP7	I/O	SPI0 1st MOSI (Master Out, Slave In) pin.
	SPI0_SS	PB.14	MFP7	I/O	SPI0 slave select pin.
	SPI0_SS	PC.0	MFP1	I/O	SPI0 slave select pin.
	SPI0_SS	PC.8	MFP3	I/O	SPI0 slave select pin.
	SPI0_SS	PD.0	MFP7	I/O	SPI0 slave select pin.
	SPI0_SS	PD.4	MFP7	I/O	SPI0 slave select pin.
	SPI0_I2SMCLK	PA.15	MFP2	O	I2S0 master clock output pin.
	SPI0_I2SMCLK	PB.9	MFP3	O	I2S0 master clock output pin.
	SPI0_I2SMCLK	PB.10	MFP3	O	I2S0 master clock output pin.
	SPI0_I2SMCLK	PC.4	MFP3	O	I2S0 master clock output pin.
	SPI0_I2SMCLK	PC.12	MFP3	O	I2S0 master clock output pin.
TM0	TM0	PA.11	MFP5	I/O	Timer0 event counter input / toggle output.
	TM0	PB.8	MFP1	I/O	Timer0 event counter input / toggle output.
	TM0_EXT	PB.15	MFP2	I	Timer0 external counter input.
	TM0_EXT	PD.1	MFP4	I	Timer0 external counter input.
	TM0_EXT	PE.2	MFP5	I	Timer0 external counter input.
TM1	TM1	PB.9	MFP5	I/O	Timer1 event counter input / toggle output.
	TM1	PC.11	MFP1	I/O	Timer1 event counter input / toggle output.
	TM1_EXT	PD.3	MFP4	I	Timer1 external counter input.
	TM1_EXT	PF.1	MFP5	I	Timer1 external counter input.
	TM1_EXT	PE.3	MFP5	I	Timer1 external counter input.
TM2	TM2	PB.10	MFP1	I/O	Timer2 event counter input / toggle output.
	TM2	PC.0	MFP5	I/O	Timer2 event counter input / toggle output.
	TM2_EXT	PB.2	MFP2	I	Timer2 external counter input.
	TM2_EXT	PB.4	MFP5	I	Timer2 external counter input.
TM3	TM3	PB.5	MFP5	I/O	Timer3 event counter input / toggle output.
	TM3	PD.2	MFP4	I/O	Timer3 event counter input / toggle output.
	TM3	PE.1	MFP5	I/O	Timer3 event counter input / toggle output.
	TM3	PF.0	MFP5	I/O	Timer3 event counter input / toggle output.
	TM3_EXT	PB.3	MFP2	I	Timer3 external counter input.
UART0	UART0_RXD	PA12	MFP3	I	Data receiver input pin for UART0.
	UART0_RXD	PB.0	MFP1	I	Data receiver input pin for UART0.
	UART0_RXD	PC.0	MFP6	I	Data receiver input pin for UART0.

Group	Pin Name	GPIO	MFP*	Type	Description
	UART0_RXD	PC.4	MFP2	I	Data receiver input pin for UART0.
	UART0_RXD	PD.1	MFP5	I	Data receiver input pin for UART0.
	UART0_RXD	PF.5	MFP3	I	Data receiver input pin for UART0.
	UART0_TXD	PA.13	MFP3	O	Data transmitter output pin for UART0.
	UART0_TXD	PB.1	MFP1	O	Data transmitter output pin for UART0.
	UART0_TXD	PC.1	MFP6	O	Data transmitter output pin for UART0.
	UART0_TXD	PC.5	MFP2	O	Data transmitter output pin for UART0.
	UART0_TXD	PD.2	MFP5	O	Data transmitter output pin for UART0.
	UART0_TXD	PF.4	MFP3	O	Data transmitter output pin for UART0.
	UART0_nCTS	PA.14	MFP3	I	Clear to Send input pin for UART0.
	UART0_nCTS	PB.3	MFP1	I	Clear to Send input pin for UART0.
	UART0_nCTS	PC.2	MFP6	I	Clear to Send input pin for UART0.
	UART0_nCTS	PD.3	MFP5	I	Clear to Send input pin for UART0.
	UART0_nRTS	PA.15	MFP5	O	Request to Send output pin for UART0.
	UART0_nRTS	PB.2	MFP1	O	Request to Send output pin for UART0.
	UART0_nRTS	PB.14	MFP2	O	Request to Send output pin for UART0.
	UART0_nRTS	PC.3	MFP6	O	Request to Send output pin for UART0.
	UART0_nRTS	PD.0	MFP5	O	Request to Send output pin for UART0.
	UART0_nRTS	PD.4	MFP5	O	Request to Send output pin for UART0.
USCI0	USCI0_CLK	PC.0	MFP7	I/O	USCI0 clock pin.
	USCI0_CLK	PC.9	MFP7	I/O	USCI0 clock pin.
	USCI0_CLK	PD.1	MFP6	I/O	USCI0 clock pin.
	USCI0_CTL0	PB.4	MFP6	I/O	USCI0 CTL0 pin.
	USCI0_CTL0	PB.7	MFP7	I/O	USCI0 CTL0 pin.
	USCI0_CTL0	PB.12	MFP6	I/O	USCI0 CTL0 pin.
	USCI0_CTL0	PC.1	MFP7	I/O	USCI0 CTL0 pin.
	USCI0_CTL0	PC.8	MFP7	I/O	USCI0 CTL0 pin.
	USCI0_CTL0	PD.0	MFP6	I/O	USCI0 CTL0 pin.
	USCI0_CTL0	PD.4	MFP6	I/O	USCI0 CTL0 pin.
	USCI0_CTL1	PB6	MFP7	I/O	USCI0 CTL1 pin.
	USCI0_CTL1	PB.13	MFP6	I/O	USCI0 CTL1 pin.
	USCI0_CTL1	PC.12	MFP7	I/O	USCI0 CTL1 pin.
	USCI0_CTL1	PE.2	MFP7	I/O	USCI0 CTL1 pin.
	USCI0_DAT0	PA.11	MFP7	I/O	USCI0 DAT0 pin.

Group	Pin Name	GPIO	MFP*	Type	Description
	USCI0_DAT0	PB4	MFP7	I/O	USCI0 DAT0 pin.
	USCI0_DAT0	PB.6	MFP6	I/O	USCI0 DAT0 pin.
	USCI0_DAT0	PC.3	MFP7	I/O	USCI0 DAT0 pin.
	USCI0_DAT0	PC.5	MFP7	I/O	USCI0 DAT0 pin.
	USCI0_DAT0	PC.11	MFP7	I/O	USCI0 DAT0 pin.
	USCI0_DAT0	PD.3	MFP6	I/O	USCI0 DAT0 pin.
	USCI0_DAT0	PD.8	MFP6	I/O	USCI0 DAT0 pin.
	USCI0_DAT0	PE.0	MFP7	I/O	USCI0 DAT0 pin.
	USCI0_DAT1	PA.10	MFP6	I/O	USCI0 DAT1 pin.
	USCI0_DAT1	PB.5	MFP7	I/O	USCI0 DAT1 pin.
	USCI0_DAT1	PB.7	MFP6	I/O	USCI0 DAT1 pin.
	USCI0_DAT1	PC.2	MFP7	I/O	USCI0 DAT1 pin.
	USCI0_DAT1	PC.4	MFP7	I/O	USCI0 DAT1 pin.
	USCI0_DAT1	PC.10	MFP7	I/O	USCI0 DAT1 pin.
	USCI0_DAT1	PD.2	MFP6	I/O	USCI0 DAT1 pin.
	USCI0_DAT1	PE.1	MFP7	I/O	USCI0 DAT1 pin.
X32	X32_IN	PF.1	MFP1	I	External 32.768 kHZ (low speed) crystal input pin.
	X32_OUT	PF.0	MFP1	O	External 32.768 kHZ (low speed) crystal output pin.
XT1	XT1_IN	PF.1	MFP1	I	External 4~24 MHz (high speed) crystal input pin.
	XT1_OUT	PF.0	MFP1	O	External 4~24 MHz (high speed) crystal output pin.

Table 4.3-7 NUC121 GPIO Multi-function Table

5 BLOCK DIAGRAM

5.1 NuMicro® NUC121/125 Block Diagram

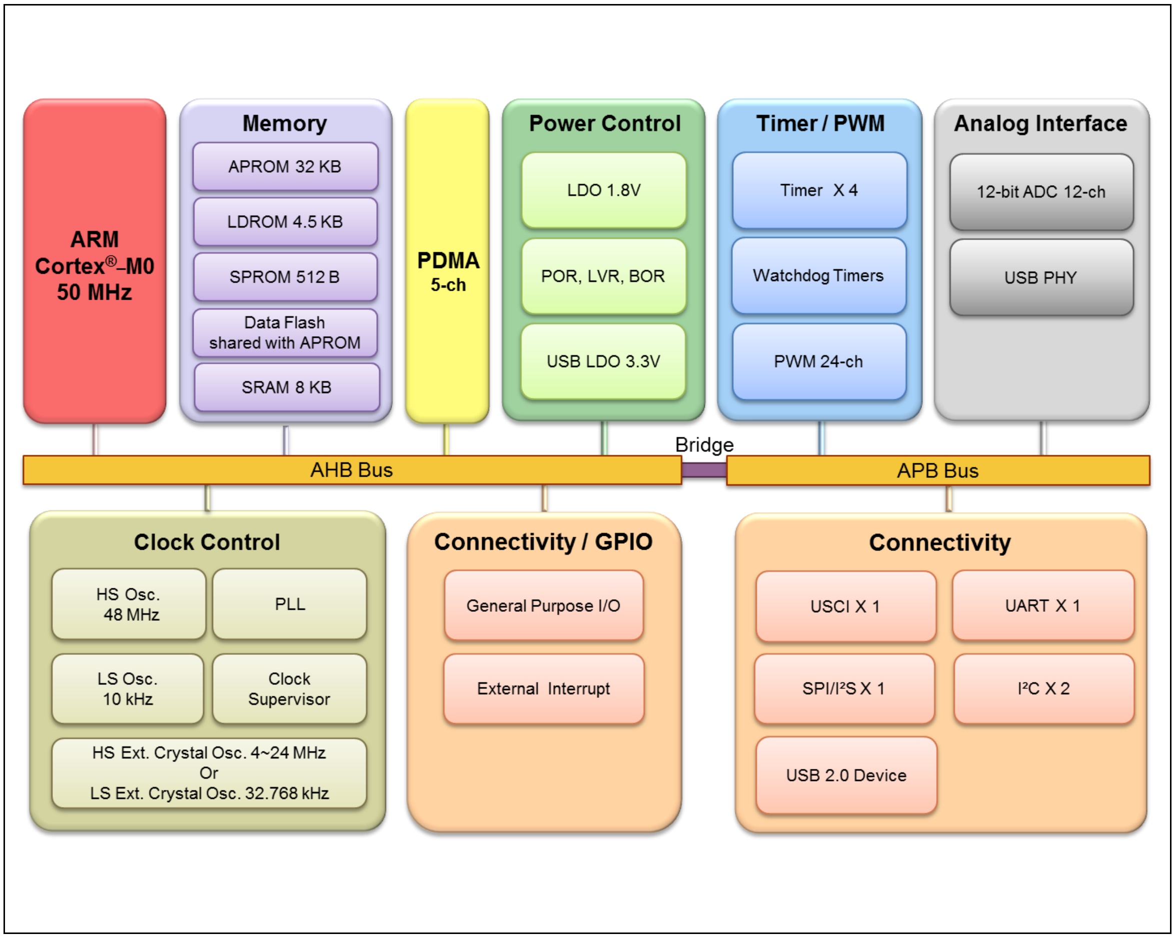


Figure 5.1-1 NuMicro® NUC121/125 Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 ARM® Cortex®-M0 Core

The Cortex®-M0 processor, a configurable, multistage, 32-bit RISC processor, has three AMBA AHB-Lite interfaces for best parallel performance and includes an NVIC component. The processor with 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. The Cortex®-M0 is a processor with the same capability as the Cortex®-M0 processor and includes floating point arithmetic functionality. The NuMicro® NUC121/125 series is embedded with Cortex®-M0 processor. Throughout this document, the name Cortex®-M0 refers to both Cortex®-M0 and Cortex®-M0 processors. Figure 6.1-1 shows the functional controller of the processor.

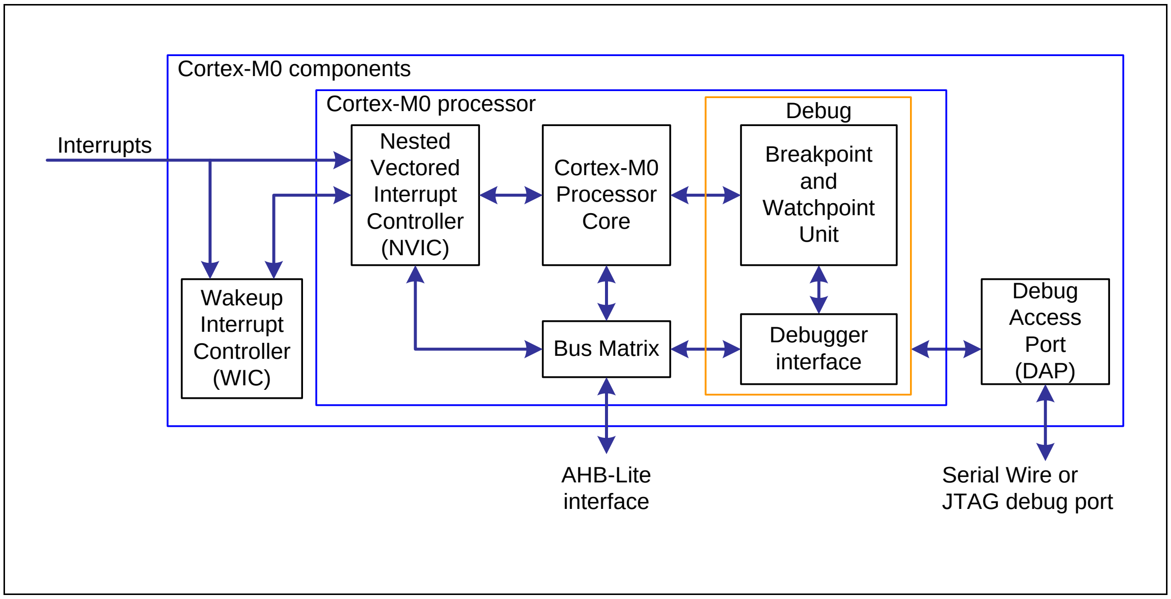


Figure 6.1-1 Cortex®-M0 Block Diagram

The implemented device provides:

- A low gate count processor:
- ARMv6-M Thumb® instruction set
- Thumb-2 technology
- ARMv6-M compliant 24-bit SysTick timer
- A 32-bit hardware multiplier
- System interface supported with little-endian data accesses
- Ability to have deterministic, fixed-latency, interrupt handling
- Load/store-multiples and multicycle-multiplies that can be abandoned and restarted to facilitate rapid interrupt handling
- C Application Binary Interface compliant exception model. This is the ARMv6-M, C Application Binary Interface (C-ABI) compliant exception model that enables

the use of pure C functions as interrupt handlers

  - ◆ Low Power Sleep mode entry using the Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or return from interrupt sleep-on-exit feature

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

6.2 System Manager

6.2.1 Overview

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

- System Reset
- Power Modes and Wake-up Sources
- System Power Distribution
- SRAM Memory Orginization
- System Control Register for Part Number ID, Chip Reset and Multi-function Pin Control
- System Timer (SysTick)
- Nested Vectored Interrupt Controller (NVIC)
- System Control register

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 SYS_RSTSTS 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)
  - CPU Lockup Reset
- Software Reset Sources
  - CHIP Reset will reset whole chip by writing 1 to CHIPRST (SYS_IPRST0[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 (SYS_IPRST0[1])

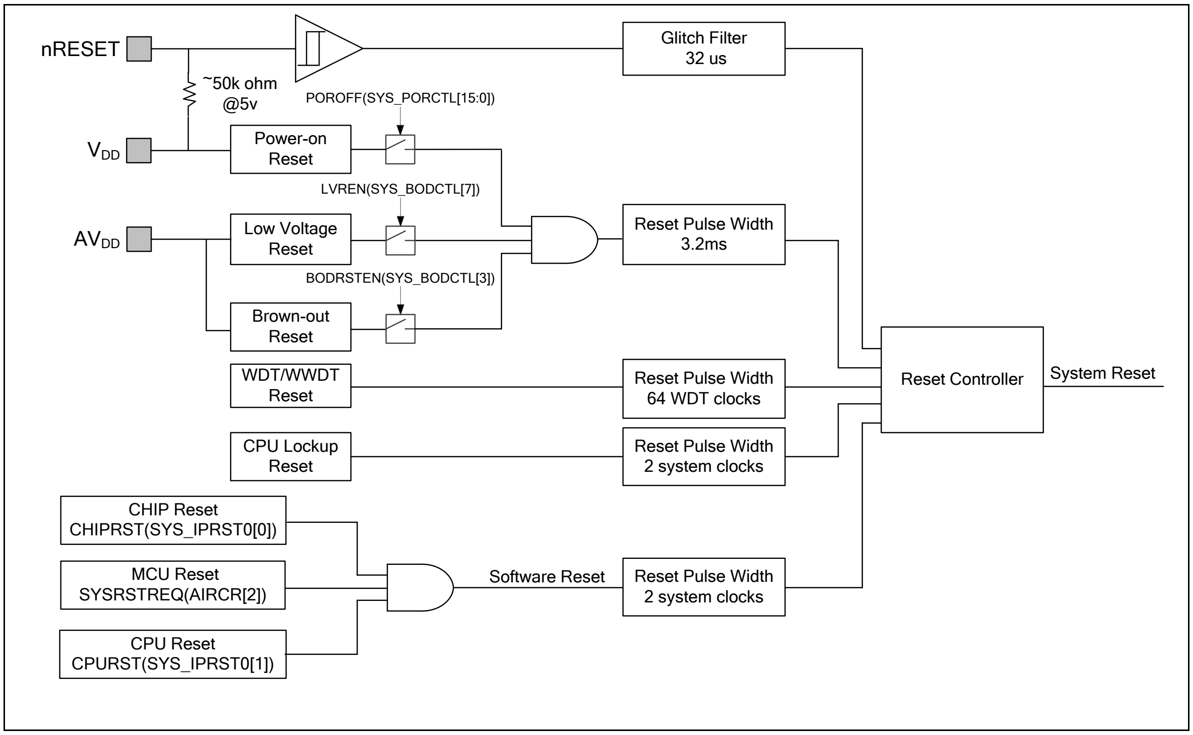


Figure 6.2-1 System Reset Sources

There are a total of 9 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 6.2-1.

Reset Sources **Register**	**POR**	**NRESET**	**WDT**	**LVR**	**BOD**	**Lockup**	**CHIP**	**MCU**	**CPU**
SYS_RSTSTS	0x001	Bit 1 = 1	Bit 2 = 1	Bit 3 = 1	Bit 4 = 1	Bit 8 = 1	Bit 0 = 1	Bit 5 = 1	Bit 7 = 1
CHIPRST **(SYS_IPRST0[0])**	0x0	-	-	-	-	-	-	-	-
BODEN **(SYS_BODCTL[0])**	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-
BODVL **(SYS_BODCTL[2:1])**									
BODRSTEN **(SYS_BODCTL[3])**									
HXTEN **(CLK_PWRCTL[0])**	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	
LXTEN **(CLK_PWRCTL[1])**	0x0	-	-	-	-	-	-	-	-
WDTCKEN **(CLK_APBCLK0[0])**	0x1	-	0x1	-	-	-	0x1	-	-
HCLKSEL **(CLK_CLKSEL0[2:0])**	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-
WDTSEL **(CLK_CLKSEL1[1:0])**	0x3	0x3	-	-	-	-	-	-	-
HXTSTB **(CLK_STATUS[0])**	0x0	-	-	-	-	-	-	-	-
LXTSTB **(CLK_STATUS[1])**	0x0	-	-	-	-	-	-	-	-
PLLSTB **(CLK_STATUS[2])**	0x0	-	-	-	-	-	-	-	-
HIRCSTB **(CLK_STATUS[4])**	0x0	-	-	-	-	-	-	-	-
CLKSFAIL **(CLK_STATUS[7])**	0x0	0x0	-	-	-	-	-	-	-
RSTEN **(WDT_CTL[1])**	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
WDTEN **(WDT_CTL[7])**									
WDT_CTL **except bit 1 and bit 7.**	0x0700	0x0700	0x0700	0x0700	0x0700	-	0x0700	-	-

WDT_ALTCTL	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_RLDCNT	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_CTL	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	-	0x3F0800	-	-
WWDT_STATUS	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_CNT	0x3F	0x3F	0x3F	0x3F	0x3F	-	0x3F	-	-
BS **(FMC_ISPCTL[1])**	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
BL **(FMC_ISPCTL[16])**									
FMC_DFBA	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	-	Reload from CONFIG1	-	-
CBS **(FMC_ISPSTS[2:1])**	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
VECMAP **(FMC_ISPSTS[23:9])**	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	-	Reload base on CONFIG0	-	-
Other Peripheral Registers	Reset Value								-
FMC Registers	Reset Value								
Note: '-' means that the value of register keeps original setting.									

Table 6.2-1 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 32 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 32 us (glitch filter). The PINRF(SYS_RSTSTS[1]) will be set to 1 if the previous reset source is nRESET reset. Figure 6.2-2 shows the nRESET reset waveform.

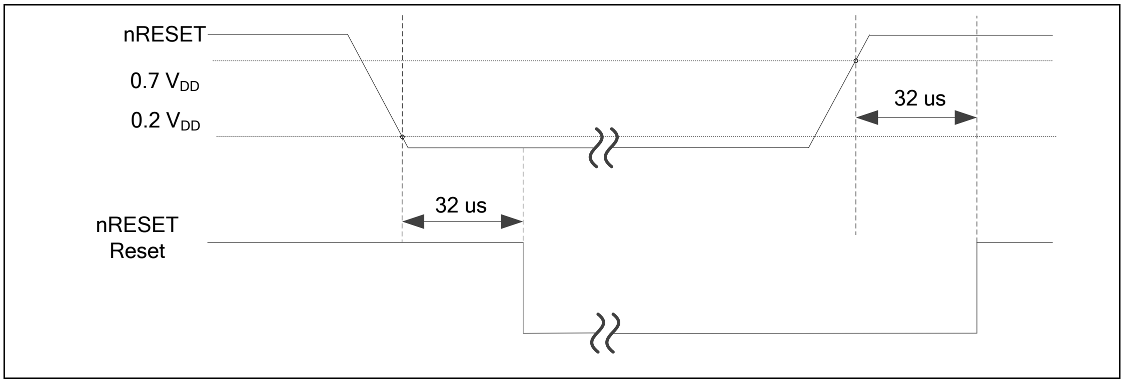


Figure 6.2-2 nRESET Reset Waveform

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 PORF(SYS_RSTSTS[0]) will be set to 1 to indicate there is a POR reset event. The PORF(SYS_RSTSTS[0]) bit can be cleared by writing 1 to it. Figure 6.2-3 shows the power-on reset waveform.

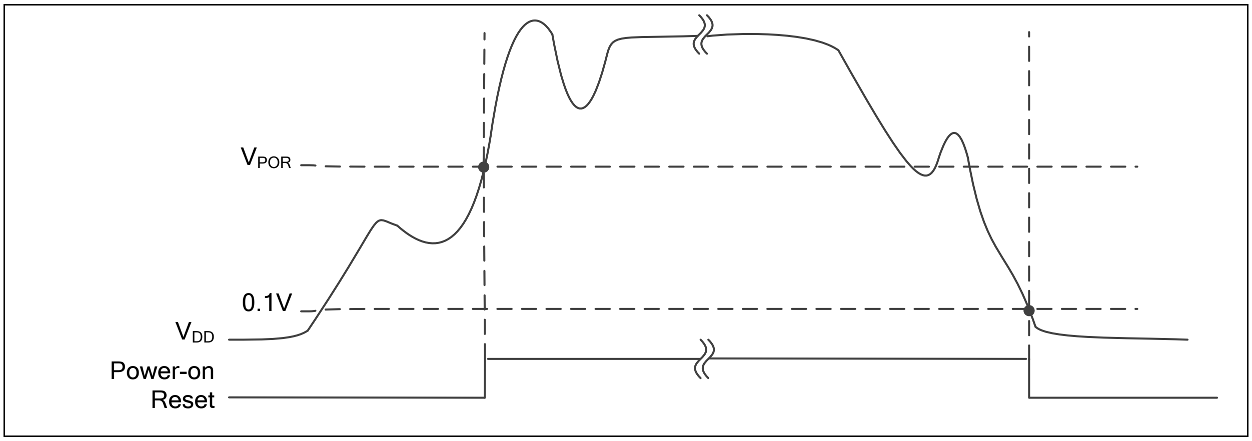


Figure 6.2-3 Power-on Reset (POR) Waveform

6.2.2.3 Low Voltage Reset (LVR)

If the Low Voltage Reset function is enabled by setting the Low Voltage Reset Enable Bit LVREN (SYS_BODCTL[7]) to 1, after 200us 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 set by LVRDGSEL (SYS_BODCTL[14:12]), 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 set by LVRDGSEL (SYS_BODCTL[14:12]). The LVRF(SYS_RSTSTS[3]) will be set to 1 if the previous reset source is LVR reset. The default setting of Low Voltage Reset is enabled without De-glitch function. Figure 6.2-4 shows the Low Voltage Reset waveform.

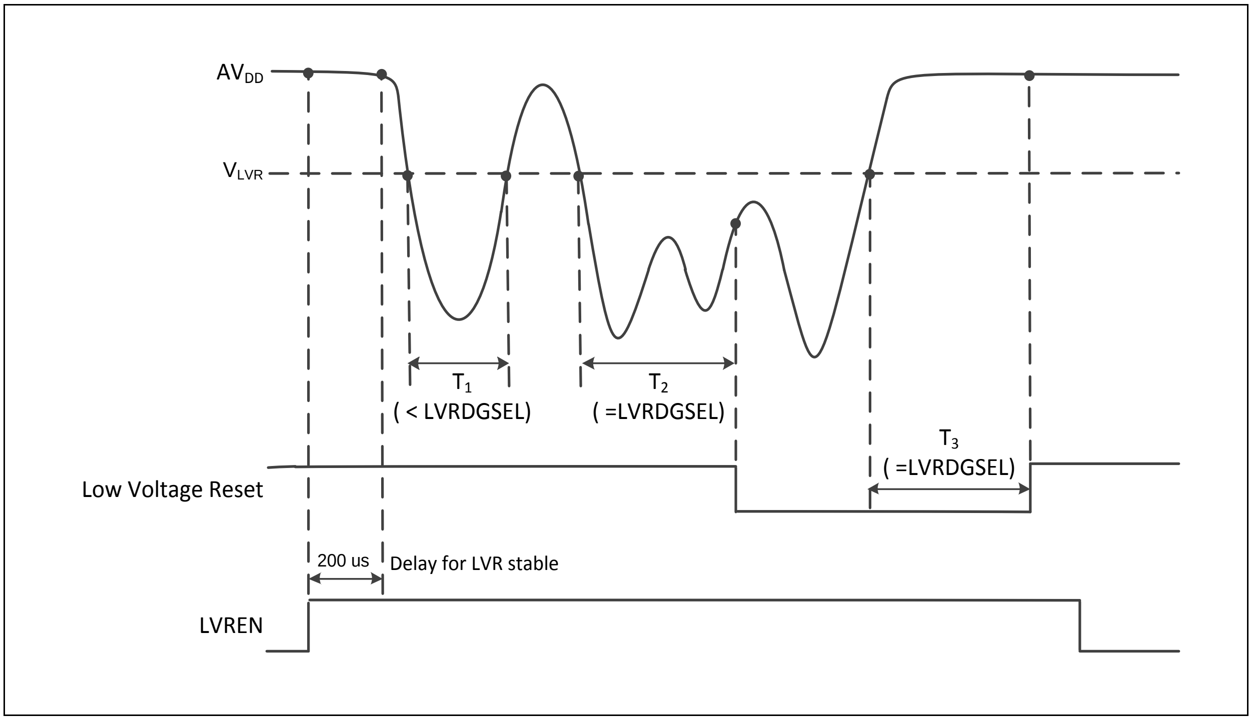


Figure 6.2-4 Low Voltage Reset (LVR) Waveform

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

If the Brown-out Detector (BOD) function is enabled by setting the Brown-out Detector Enable Bit BODEN (SYS_BODCTL[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 set by BODDGSEL (SYS_BODCTL[10:8]), 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 set by BODDGSEL (SYS_BODCTL[10:8]). The default value of BODEN, BODVL and BODRSTEN(SYS_BODCTL[3]) is set by flash controller user configuration register CBODEN (CONFIG0 [23]), CBOV (CONFIG0 [22:21]) and CBORST(CONFIG0[20]) respectively. User can determine the initial BOD setting by setting the CONFIG0 register. Figure 6.2-5 shows the Brown-Out Detector waveform.

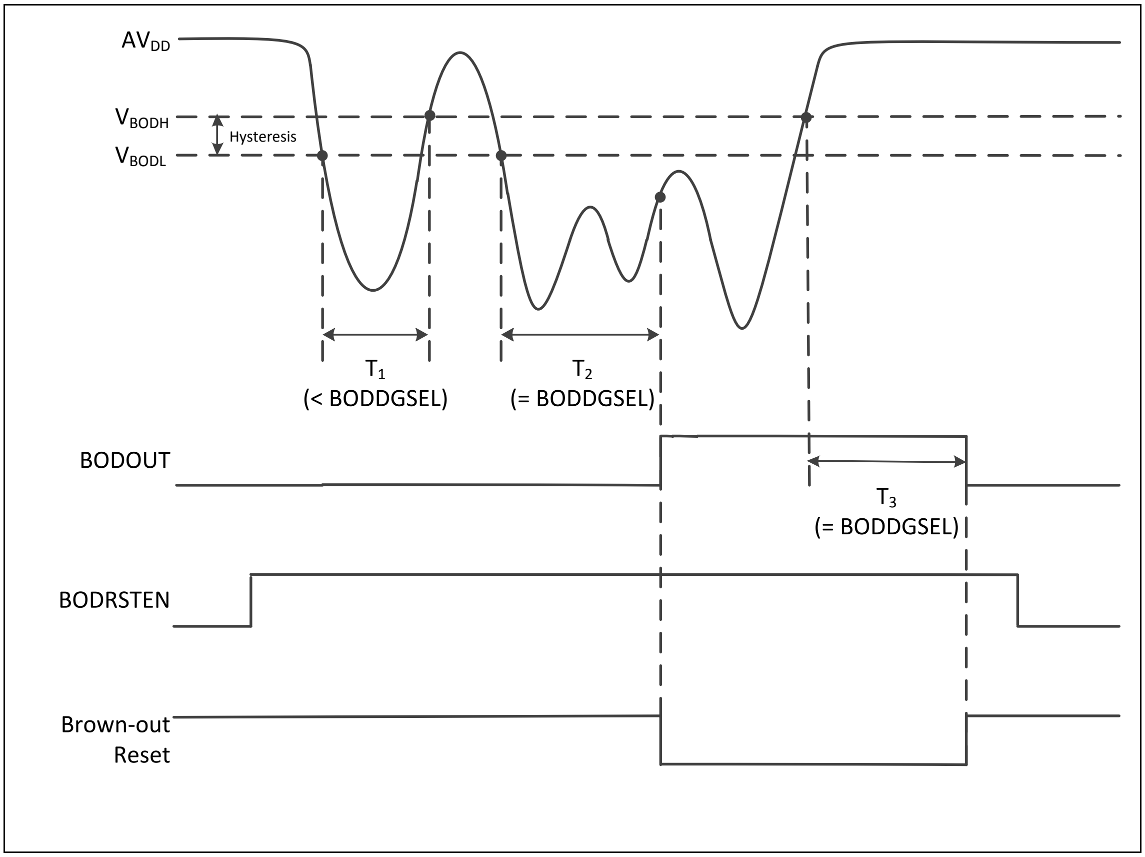


Figure 6.2-5 Brown-out Detector (BOD) Waveform

6.2.2.5 Watchdog Timer Reset (WDT)

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

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

6.2.2.6 CPU Lockup Reset

CPU enters lockup status after CPU produces hardfault at hardfault handler and chip gives immediate indication of seriously errant kernel software. This is the result of the CPU being locked because of an unrecoverable exception following the activation of the processor’s built in system state protection hardware. When chip enters debug mode, the CPU lockup reset will be ignored.

6.2.2.7 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 CPURST(SYS_IPRST0[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(FMC_ISPCTL[1]) bit is automatically reloaded from CONFIG0 setting. User can set the CHIPRST(SYS_IPRST0[1]) to 1 to assert the CHIP Reset signal.

The MCU Reset is similar with CHIP Reset. The difference is that BS(FMC_ISPCTL[1]) will not be reloaded from CONFIG0 setting and keep its original software setting for booting from APROM or LDROM. User can set the 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 6.2-2 lists the available clocks for each power mode.

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

Table 6.2-2 Power Mode Difference Table

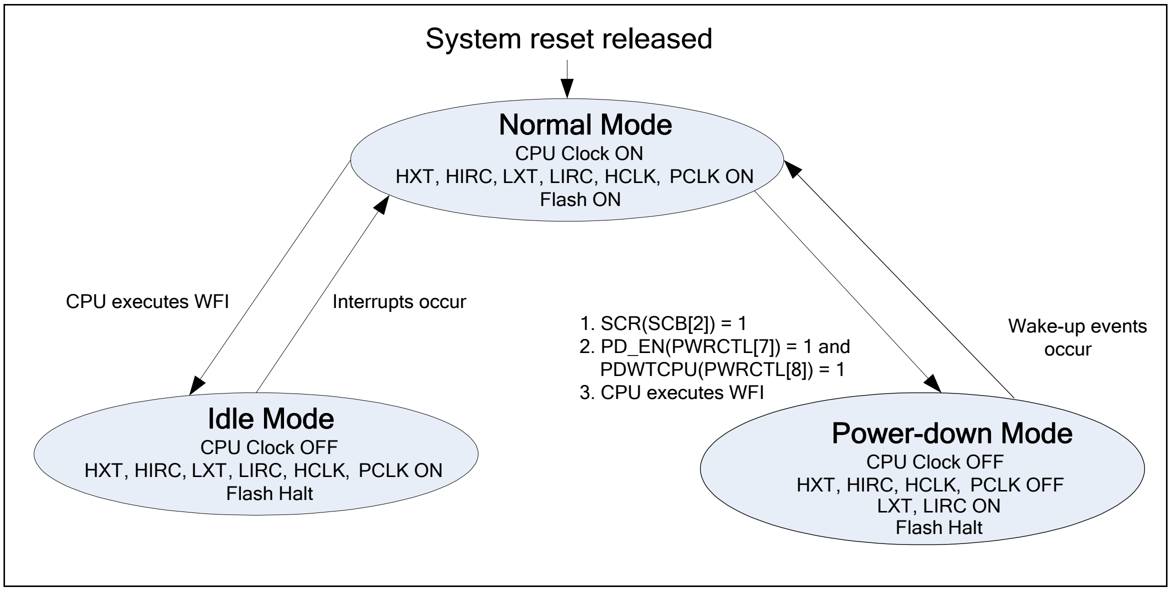


Figure 6.2-6 Power Mode State Machine

1. LXT (32768 Hz XTL) ON or OFF depends on SW setting in run mode.
2. LIRC (10 kHz OSC) ON or OFF depends on S/W setting in run mode.
3. If TIMER clock source is selected as LIRC/LXT and LIRC/LXT is on.
4. If WDT clock source is selected as LIRC and LIRC is on.
5. If UART clock source is selected as LXT and LXT is on.

	Normal Mode	Idle Mode	Power-Down Mode
HXT (4~24 MHz XTL)	ON	ON	Halt
HIRC (48 MHz OSC)	ON	ON	Halt
LXT (32768 Hz XTL)	ON	ON	ON/OFF[1]
LIRC (10 kHz OSC)	ON	ON	ON/OFF[2]
PLL	ON	ON	Halt
LDO	ON	ON	ON
CPU	ON	Halt	Halt
HCLK/PCLK	ON	ON	Halt
SRAM retention	ON	ON	ON
FLASH	ON	ON	Halt
GPIO	ON	ON	Halt
PDMA	ON	ON	Halt
TIMER	ON	ON	ON/OFF[3]
BPWM	ON	ON	Halt
PWM	ON	ON	Halt
WDT	ON	ON	ON/OFF[4]
WWDT	ON	ON	Halt
USCI	ON	ON	Halt
UART	ON	ON	ON/OFF[5]
I^2C	ON	ON	Halt
SPI/I^2S	ON	ON	Halt
USBD	ON	ON	Halt
ADC	ON	ON	Halt

Table 6.2-3 Clocks in Power Modes

Wake-up Sources in Power-down Mode:

WDT, I²C, Timer, UART, BOD, GPIO, EINT, USCI and USBD

After chip enters power down, the following wake-up sources can wake chip up to normal mode. Table 6.2-4 lists the condition about how to enter Power-down mode again for each peripheral.

*User needs to wait this condition before setting PD_EN(PWRCTL[6]) 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 SYS_BODCTL[BODIF].
GPIO	GPIO Interrupt	After software write 1 to clear the INTSRC[n] bit.
TIMER	Timer Interrupt	After software writes 1 to clear TWKF (TIMERx_INTSTS[1]) and TIF (TIMERx_INTSTS[0]).
WDT	WDT Interrupt	After software writes 1 to clear WKF (WDT_CTL[5]) (Write Protect).
UART	RX Data wake-up	After software writes 1 to clear DATWKIF (UARTx_INTSTS[17]).
	nCTS wake-up	After software writes 1 to clear CTSWKIF (UARTx_INTSTS[16]).
I^2C	Falling edge in the I2C_SDA or I2C_CLK	After software writes 1 to clear WKIF(I2C_WKSTS[0]).
USCI		
USBD	Remote Wake-up	After software writes 1 to clear BUSIF (USBD_INTSTS[0]).

Table 6.2-4 Condition of Entering Power-down Mode Again

6.2.4 System Power Distribution

In this chip, power distribution is divided into four 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.
- A dedicated power from V_{DDIO} supplies the power for PA.10, PA.11, PB.4 ~ PB.7, PB.12 ~ PB.14 and PD.8 ~ 11 of NUC125.

The outputs of internal voltage regulators, LDO and VDD33, 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 6.2-7 shows the power distribution of the NuMicro® NUC121 and NUC125.

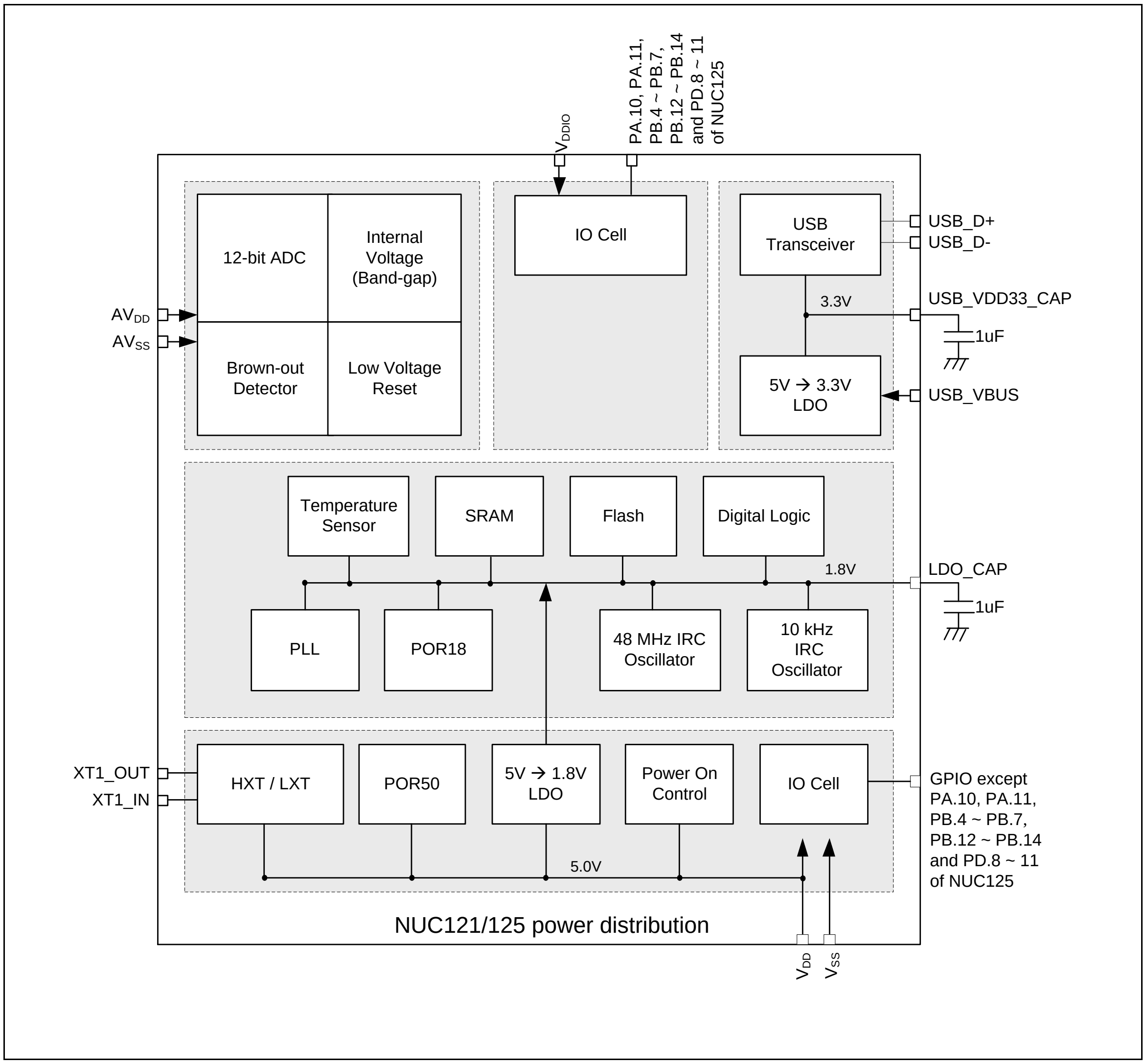


Figure 6.2-7 NuMicro® NUC121/125 Power Distribution Diagram

6.2.5 System Memory Map

The NUC121/125 series provides 4G-byte addressing space. The memory locations assigned to each on-chip controllers are shown in Table 6.2-5. The detailed register definition, memory space, and programming will be described in the following sections for each on-chip peripheral. The NUC121/125 series only supports little-endian data format.

Address Space	Token	Controllers
Flash and SRAM Memory Space		
0x0000_0000 – 0x0000_7FFF	FLASH_BA	FLASH Memory Space (32 KB)
0x2000_0000 – 0x2000_1FFF	SRAM_BA	SRAM Memory Space (8 KB)
AHB Controllers Space (0x5000_0000 – 0x501F_FFFF)		
0x5000_0000 – 0x5000_01FF	SYS_BA	System 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
Peripheral Controllers Space (0x4000_0000 – 0x401F_FFFF)		
0x4000_4000 – 0x4000_7FFF	WDT_BA	Watchdog Timer Control Registers
0x4001_0000 – 0x4001_3FFF	TMR01_BA	Timer0/Timer1 Control Registers
0x4002_0000 – 0x4002_3FFF	I2C0_BA	I^{2}C0 Interface Control Registers
0x4003_0000 – 0x4003_3FFF	SPI0_BA	SPI0 with master/slave function Control Registers
0x4004_0000 – 0x4004_3FFF	PWM0_BA	PWM0 Control Registers
0x4004_4000 – 0x4004_7FFF	PWM2_BA	PWM2 Control Registers
0x4005_0000 – 0x4005_3FFF	UART0_BA	UART0 Control Registers
0x4006_0000 – 0x4006_3FFF	USBD_BA	USB 2.0 FS device Controller Registers
0x4007_0000 – 0x4007_3FFF	USCI0_BA	USCI0 Control Registers
0x400E_0000 – 0x400E_FFFF	ADC_BA	Analog-Digital-Converter (ADC) Control Registers
0x4011_0000 – 0x4011_3FFF	TMR23_BA	Timer2/Timer3 Control Registers
0x4012_0000 – 0x4012_3FFF	I2C1_BA	I^{2}C1 Interface Control Registers
0x4014_0000 – 0x4014_3FFF	PWM1_BA	PWM1 Control Registers
0x4014_4000 – 0x4014_7FFF	PWM3_BA	PWM3 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 6.2-5 Address Space Assignments for On-Chip Controllers

6.2.6 SRAM Memory Orginization

The NUC121/125 series supports embedded SRAM with total 8 Kbytes size in one bank.

- Supports total 8 Kbytes SRAM
- Supports byte / half word / word write
- Supports oversize response error

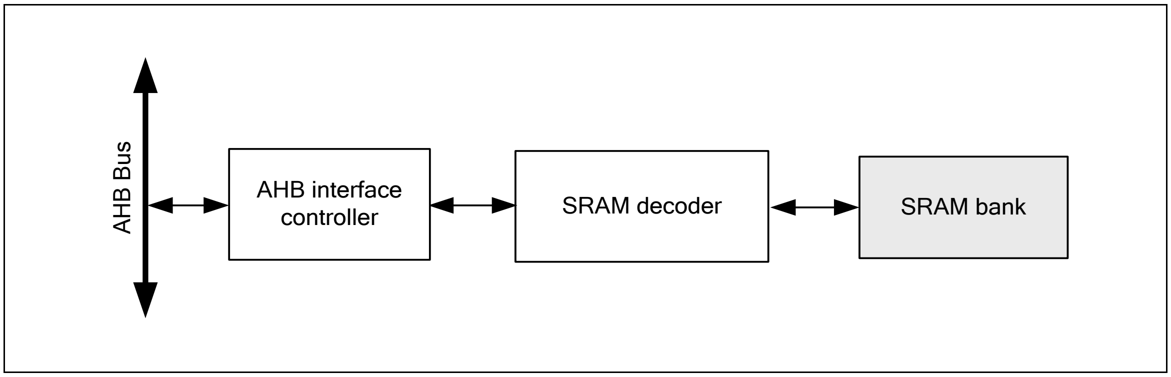


Figure 6.2-8 SRAM Block Diagram

Figure 6.2-9 shows the SRAM organization of NUC121/125 series. There are one SRAM bank in NUC121/125 series and addressed to 8 Kbytes. The address space is from 0x2000_0000 to 0x2000_1FFF. The address between 0x2000_2000 to 0x3FFF_FFFF is illegal memory space and chip will enter hardfault if CPU accesses these illegal memory addresses.

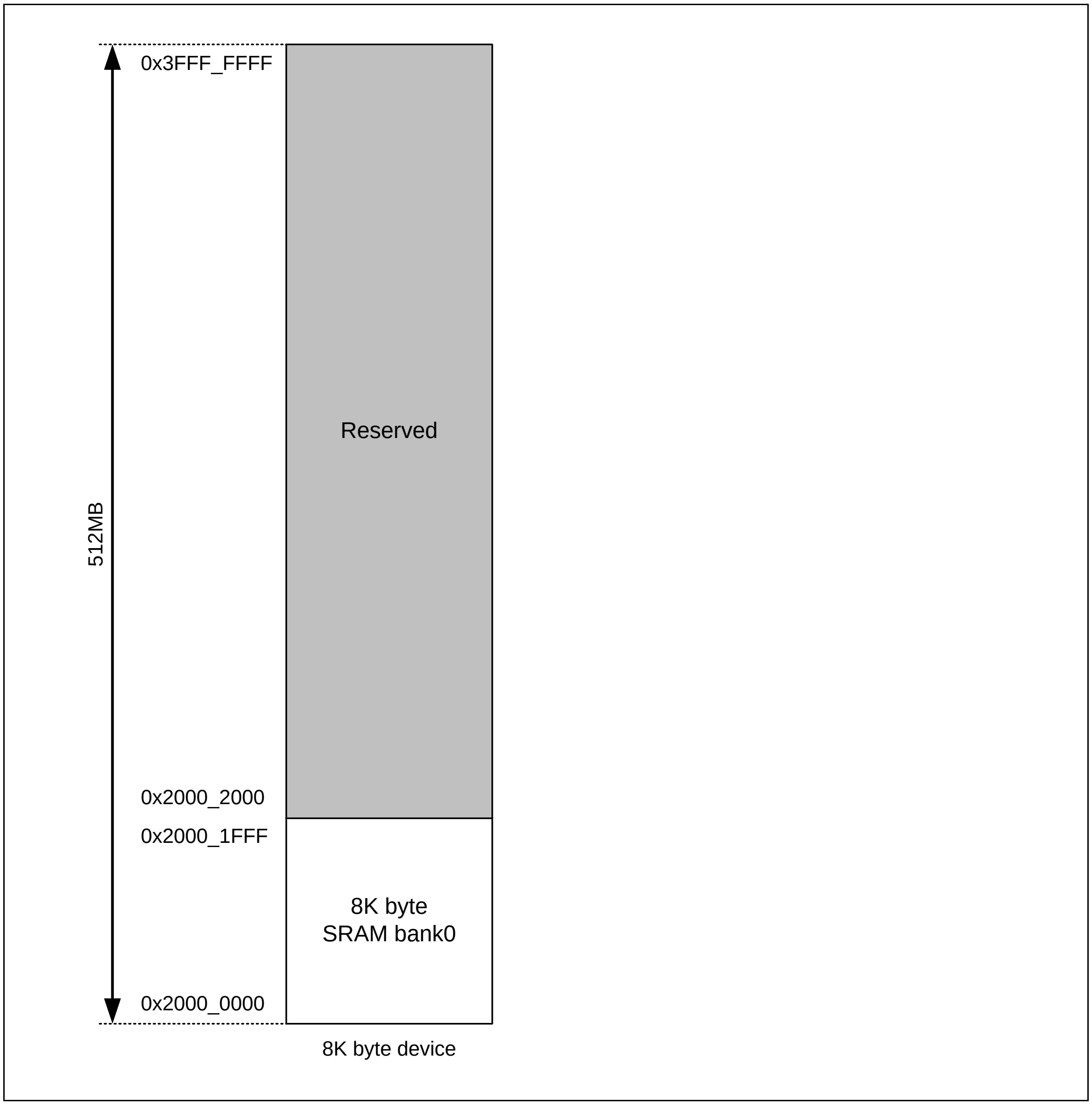


Figure 6.2-9 SRAM Memory Organization

6.2.7 Register Lock

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 SYS_REGLCTL 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 register protection is disabled, user can read SYS_REGLCTL[0] to check register protection status. 1 is protection disable and 0 is protection enable. Once user wants to enable register protection, SYS_REGLCTL can be wroten any data to enable register protection.

6.2.8 Auto Trim

This chip supports auto-trim function: the HIRC trim (48 MHz RC oscillator), according to the accurate LXT (32.768 kHz crystal oscillator) or internal USB synchronous mode, automatically gets accurate HIRC output frequency, 0.25 % deviation within all temperature ranges.

For instance, the system needs an accurate 48 MHz clock for USB application. In such case, if neither using PLL as the system clock source nor soldering 32.768 kHz crystal in system, user has to set REFCKSEL (SYS_IRCTCTL[10] reference clock selection) to “1”, set FREQSEL (SYS_IRCTCTL[1:0] trim frequency selection) to “10”, and the auto-trim function will be enabled. The interrupt status bit FREQLOCK (SYS_IRCTISTS[0] HIRC frequency lock status) “1” indicates the HIRC output frequency is accurate within 0.25% deviation.

6.2.9 UART0_TXD and USCI0_DAT0 Modulation with BPWM2

This chip supports UART0_TXD to modulate with BPWM2 channel. User can set MODPWMSEL (SYS_MODCTL[7:4]) to choice which BPWM2 channel to modulate with UART0_TXD and set MODEN (SYS_MODCTL[0]) to enable modulation function. User can set TXDINV (UART_LINE[8]) to inverse UART0_TXD before moulating with BPWM2.

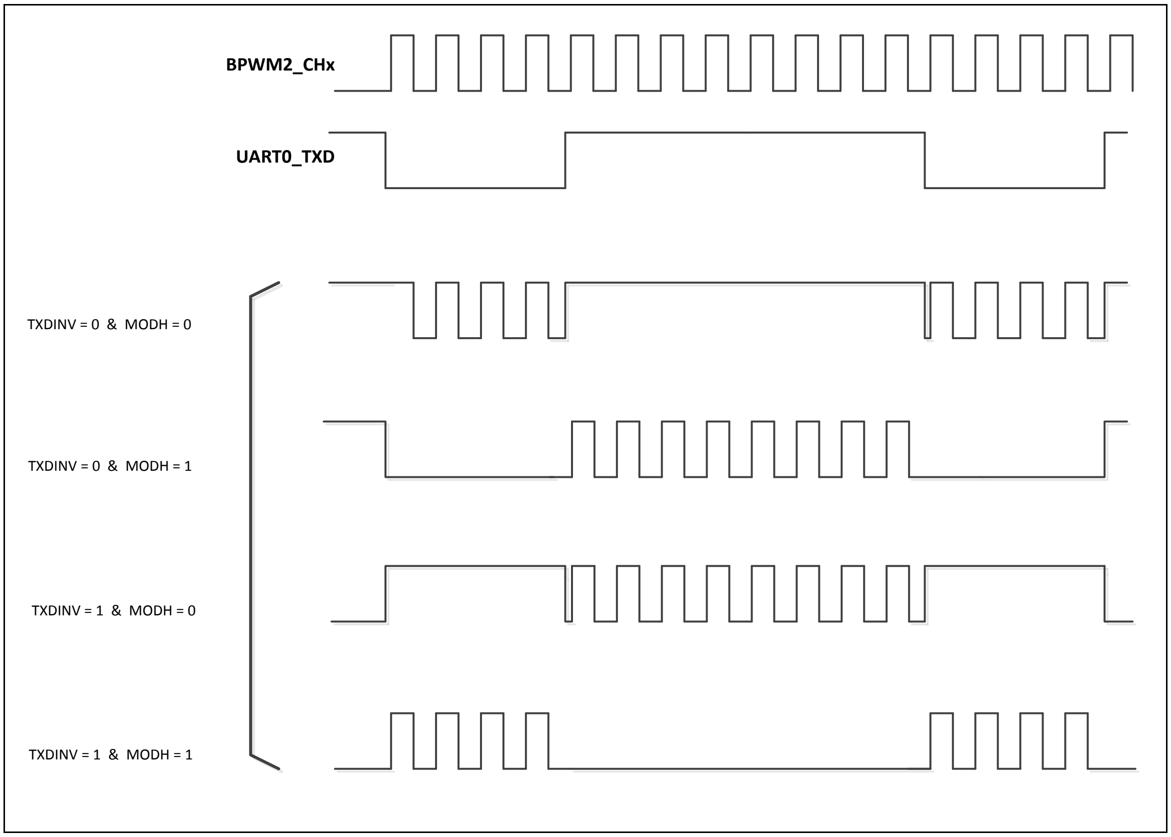


Figure 6.2-10 UART0_TXD Modulated with PWM Channel

In addition, this chip also supports USCI0_DAT0 to modulate with BPWM2 channel. User can set MODPWMSEL (SYS_MODCTL[7:4]) to choice which BPWM2 channel to modulate with USCI0_DAT0 and set MODEN (SYS_MODCTL[0]) to enable modulation function. User can set DATOINV (UUART_LINECTL[5]) to inverse USCI0_DAT0 before moulating with BPWM.

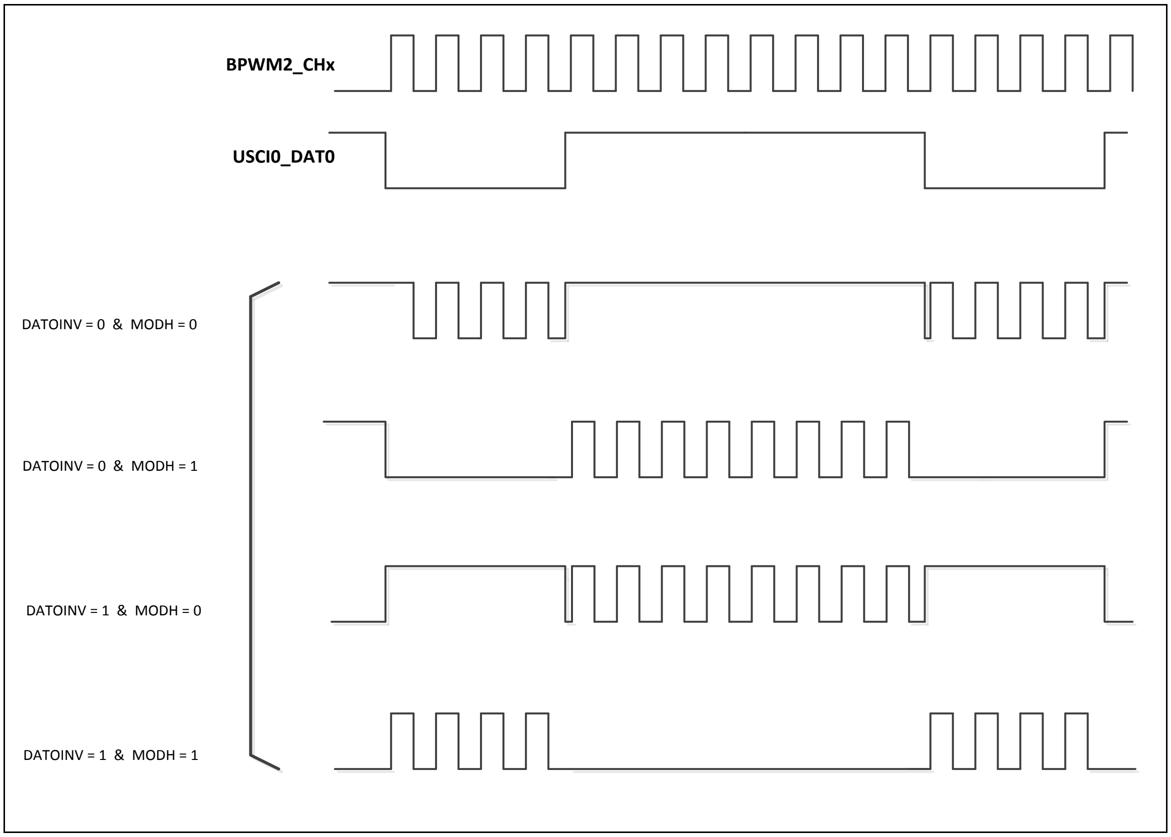


Figure 6.2-11 USCI0_DAT0 Modulated with PWM Channel

6.2.10 Register Map

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

Register	Offset	R/W	Description	Reset Value
SYS Base Address: **SYS_BA = 0x5000_0000**				
SYS_PDID	SYS_BA+0x00	R	Part Device Identification Number Register	0x0001_2XXX [1]
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Status Register	0x0000_004B
SYS_IPRST0	SYS_BA+0x08	R/W	Peripheral Reset Control Register 0	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	Peripheral Reset Control Register 1	0x0000_0000
SYS_IPRST2	SYS_BA+0x10	R/W	Peripheral Reset Control Register 2	0x0000_0000
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-Out Detector Control Register	0x0000_038X
SYS_IVSCTL	SYS_BA+0x1C	R/W	Internal Voltage Source Control Register	0x0000_0000
SYS_PORCTL	SYS_BA+0x24	R/W	Power-On-Reset Controller Register	0x0000_00XX
SYS_GPA_MFPH	SYS_BA+0x34	R/W	GPIOA High Byte Multiple Function Control Register	0x0000_0000
SYS_GPB_MFPL	SYS_BA+0x38	R/W	GPIOB Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPB_MFPH	SYS_BA+0x3C	R/W	GPIOB High Byte Multiple Function Control Register	0x0000_0000
SYS_GPC_MFPL	SYS_BA+0x40	R/W	GPIOC Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPC_MFPH	SYS_BA+0x44	R/W	GPIOC High Byte Multiple Function Control Register	0x0000_0000
SYS_GPD_MFPL	SYS_BA+0x48	R/W	GPIOD Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPD_MFPH	SYS_BA+0x4C	R/W	GPIOD High Byte Multiple Function Control Register	0x0000_0000
SYS_GPE_MFPL	SYS_BA+0x50	R/W	GPIOE Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPF_MFPL	SYS_BA+0x58	R/W	GPIOF Low Byte Multiple Function Control Register	0x0011_0011
SYS_MODCTL	SYS_BA+0xC0	R/W	Modulation Control Register	0x0000_0000
SYS_IRCTCTL	SYS_BA+0x80	R/W	HIRC Trim Control Register	0x0000_0000
SYS_IRCTIEN	SYS_BA+0x84	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000
SYS_IRCTISTS	SYS_BA+0x88	R/W	HIRC Trim Interrupt Status Register	0x0000_0000
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock Control Register	0x0000_0000
SYS_TSOFFSET	SYS_BA+0x114	R	Temperature sensor offset Register	0x0000_0XXX

6.2.11 Register Description

Part Device Identification Number Register (SYS_PDID)

Register	Offset	R/W	Description	Reset Value
SYS_PDID	SYS_BA+0x00	R	Part Device Identification Number Register	0x0001_2XXX [1]

[1] Every 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 (Read Only)** This register reflects device part number code. Software can read this register to identify which device is used.

NUC121 USB Series		NUC125 USB Series	
Part Number	**PDID**	**Part Number**	**PDID**
NUC121ZC2AE	0x00012145	NUC125ZC2AE	0x00012545
NUC121LC2AE	0x00012125	NUC125LC2AE	0x00012525
NUC121SC2AE	0x00012105	NUC125SC2AE	0x00012505

Table 6.2-6 Part Number and PDID Mapping Table

System Reset Status Register (SYS_RSTSTS)

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

Register	Offset	R/W	Description	Reset Value
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Status Register	0x0000_004B

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							CPULKRF
7	6	5	4	3	2	1	0
CPURF	Reserved	MCURF	BODRF	LVRF	WDTRF	PINRF	PORF

Bits	Description	
[31:9]	Reserved	Reserved.
[8]	CPULKRF	**the CPULK Reset Flag Is Set by Hardware If Cortex-M0 Lockup Happened** 0 = No reset from CPU lockup happened. 1 = The Cortex-M0 lockup happened and chip is reset. **Note:** This bit can be cleared by software writing ‘1’.
[7]	CPURF	**CPU Reset Flag** The CPU reset flag is set by hardware if software writes CPURST (SYS_IPRST0[1]) 1 to reset Cortex®-M0 Core and Flash Memory Controller (FMC). 0 = No reset from CPU. 1 = The Cortex®-M0 Core and FMC are reset by software setting CPURST to 1. **Note:** This bit can be cleared by software writing ‘1’.
[6]	Reserved	Reserved.
[5]	MCURF	**MCU Reset Flag** The MCU reset flag is set by the “Reset Signal” from the Cortex®-M0 Core to indicate the previous reset source. 0 = No reset from Cortex®-M0. 1 = The Cortex®-M0 had issued the reset signal to reset the system by writing 1 to the bit SYSRESETREQ(AIRCR[2], Application Interrupt and Reset Control Register, address = 0xE000ED0C) in system control registers of Cortex®-M0 core. **Note:** This bit can be cleared by software writing ‘1’.
[4]	BODRF	**BOD Reset Flag** The BOD reset 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’.

Bits	Description	
[3]	**LVRF**	**LVR Reset Flag** The LVR reset 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 = LVR controller had issued the reset signal to reset the system. **Note:** This bit can be cleared by software writing ‘1’.
[2]	**WDTRF**	**WDT Reset Flag** The WDT reset 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 RSTF(WDT_CTL[2]) bit is set if the system has been reset by WDT time-out reset. Window Watchdog Timer register WWDTRF(WWDT_STATUS[1]) bit is set if the system has been reset by WWDT time-out reset.
[1]	**PINRF**	**NRESET Pin Reset Flag** The nRESET pin 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 = Pin nRESET had issued the reset signal to reset the system. **Note:** This bit can be cleared by software writing ‘1’.
[0]	**PORF**	**POR Reset Flag** The POR reset flag is set by the “Reset Signal” from the Power-on Reset (POR) Controller or bit CHIPRST (SYS_IPRST0[0]) to indicate the previous reset source. 0 = No reset from POR or CHIPRST. 1 = Power-on Reset (POR) or CHIPRST had issued the reset signal to reset the system. **Note:** This bit can be cleared by software writing ‘1’.

Peripheral Reset Control Register 0 (SYS_IPRST0)

Register	Offset	R/W	Description	Reset Value
SYS_IPRST0	SYS_BA+0x08	R/W	Peripheral Reset Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**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					**PDMARST**	**CPURST**	**CHIPRST**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Controller Reset (Write Protect)** Setting this bit to 1 will generate a reset signal to the PDMA. User needs 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 SYS_REGLCTL register.
[1]	**CPURST**	**Processor Core One-shot Reset (Write Protect)** Setting this bit will only reset the processor core and Flash Memory Controller(FMC), and this bit will automatically return to 0 after the 2 clock cycles. 0 = Processor core normal operation. 1 = Processor core one-shot reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**CHIPRST**	**Chip One-shot Reset (Write Protect)** Setting this bit will reset the whole chip, including Processor core and all peripherals, and this bit will automatically return to 0 after the 2 clock cycles. The CHIPRST is same as the POR reset, all the chip controllers is reset and the chip setting from flash are also reload. About the difference between CHIPRST 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 SYS_REGLCTL register.

Peripheral Reset Control Register 1 (SYS_IPRST1)

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

Register	Offset	R/W	Description	Reset Value
SYS_IPRST1	SYS_BA+0x0C	R/W	Peripheral Reset Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			ADCRST	USBDRST	Reserved		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PWM1RST	PWM0RST	BPWM1RST	BPWM0RST	Reserved			UART0RST
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			SPI0RST	Reserved		I2C1RST	I2C0RST
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		TMR3RST	TMR2RST	TMR1RST	TMR0RST	GPIORST	Reserved

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**ADCRST**	**ADC Controller Reset** 0 = ADC controller normal operation. 1 = ADC controller reset.
[27]	**USBDRST**	**USB Device Controller Reset** 0 = USB device controller normal operation. 1 = USB device controller reset.
[26:24]	**Reserved**	Reserved.
[23]	**PWM1RST**	**PWM1 Controller Reset** 0 = PWM1 controller normal operation. 1 = PWM1 controller reset.
[22]	**PWM0RST**	**PWM0 Controller Reset** 0 = PWM0 controller normal operation. 1 = PWM0 controller reset.
[21]	**BPWM1RST**	**BPWM1 Controller Reset** 0 = BPWM1 controller normal operation. 1 = BPWM1 controller reset.
[20]	**BPWM0RST**	**BPWM0 Controller Reset** 0 = BPWM0 controller normal operation. 1 = BPWM0 controller reset.
[19:17]	**Reserved**	Reserved.
[16]	**UART0RST**	**UART0 Controller Reset** 0 = UART0 controller normal operation.

		1 = UART0 controller reset.
[15:13]	**Reserved**	Reserved.
[12]	**SPI0RST**	**SPI0 Controller Reset** 0 = SPI0 controller normal operation. 1 = SPI0 controller reset.
[11:10]	**Reserved**	Reserved.
[9]	**I2C1RST**	**I2C1 Controller Reset** 0 = I2C1 controller normal operation. 1 = I2C1 controller reset.
[8]	**I2C0RST**	**I2C0 Controller Reset** 0 = I2C0 controller normal operation. 1 = I2C0 controller reset.
[7:6]	**Reserved**	Reserved.
[5]	**TMR3RST**	**Timer3 Controller Reset** 0 = Timer3 controller normal operation. 1 = Timer3 controller reset.
[4]	**TMR2RST**	**Timer2 Controller Reset** 0 = Timer2 controller normal operation. 1 = Timer2 controller reset.
[3]	**TMR1RST**	**Timer1 Controller Reset** 0 = Timer1 controller normal operation. 1 = Timer1 controller reset.
[2]	**TMR0RST**	**Timer0 Controller Reset** 0 = Timer0 controller normal operation. 1 = Timer0 controller reset.
[1]	**GPIORST**	**GPIO Controller Reset** 0 = GPIO controller normal operation. 1 = GPIO controller reset.
[0]	**Reserved**	Reserved.

Peripheral Reset Control Register 2 (SYS_IPRST2)

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

Register	Offset	R/W	Description	Reset Value
SYS_IPRST2	SYS_BA+0x10	R/W	Peripheral Reset Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**8**	**11**	**10**	**9**	**8**
Reserved							**USCI0RST**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**USCI0RST**	**USCI0 Controller Reset** 0 = USCI0 controller normal operation. 1 = USCI0 controller reset.
[7:0]	**Reserved**	Reserved.

Brown-out Detector Control Register (SYS_BODCTL)

Partial of the SYS_BODCTL control registers values are initiated by the flash configuration and partial bits are write-protected bit.

Register	Offset	R/W	Description	Reset Value
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-Out Detector Control Register	0x0000_038X

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**LVRDGSEL**			**Reserved**	**BODDGSEL**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LVREN	**BODOUT**	**BODLPM**	**BODIF**	**BODRSTEN**	**BODVL**		**BODEN**

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**LVRDGSEL**	**LVR Output De-glitch Time Select (Write Protect)** 000 = Without de-glitch function. 001 = 4 system clock (HCLK). 010 = 8 system clock (HCLK). 011 = 16 system clock (HCLK). 100 = 32 system clock (HCLK). 101 = 64 system clock (HCLK). 110 = 128 system clock (HCLK). 111 = 256 system clock (HCLK). **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[11]	**Reserved**	Reserved.
[10:8]	**BODDGSEL**	**Brown-out Detector Output De-glitch Time Select (Write Protect)** 000 = BOD output is sampled by LIRC clock. 001 = 4 system clock (HCLK). 010 = 8 system clock (HCLK). 011 = 16 system clock (HCLK). 100 = 32 system clock (HCLK). 101 = 64 system clock (HCLK). 110 = 128 system clock (HCLK). 111 = 256 system clock (HCLK). **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Bits	Description	
[7]	**LVREN**	**Low Voltage Reset Enable Bit (Write Protect)** The LVR function resets 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. **Note1:** After enabling the bit, the LVR function will be active with 200us delay for LVR output stable (default). **Note2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6]	**BODOUT**	**Brown-out Detector Output Status** 0 = Brown-out Detector output status is 0. It means the detected voltage is higher than BODVL setting or BODEN is 0. 1 = Brown-out Detector output status is 1. It means the detected voltage is lower than BODVL setting. If the BODEN is 0, BOD function disabled, this bit always responds 0.
[5]	**BODLPM**	**Brown-out Detector Low Power Mode (Write Protect)** 0 = BOD operate in normal mode (default). 1 = BOD Low Power mode Enabled. **Note1:** The BOD consumes about 100uA in normal mode, the low power mode can reduce the current to about 1/10 but slow the BOD response. **Note2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[4]	**BODIF**	**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 BODVL setting. 1 = When Brown-out Detector detects the V_{DD} is dropped down through the voltage of BODVL setting or the V_{DD} is raised up through the voltage of BODVL setting, this bit is set to 1 and the brown-out interrupt is requested if brown-out interrupt is enabled. **Note:** This bit can be cleared by software writing ‘1’.
[3]	**BODRSTEN**	**Brown-out Reset Enable Bit (Write Protect)** The default value is set by flash controller user configuration register CBORST(CONFIG0[20]) bit. 0 = Brown-out “INTERRUPT” function Enabled. 1 = Brown-out “RESET” function Enabled. **Note1:** While the Brown-out Detector function is enabled (BODEN high) and BOD reset function is enabled (BODRSTEN high), BOD will assert a signal to reset chip when the detected voltage is lower than the threshold (BODOUT high). While the BOD function is enabled (BODEN high) and BOD interrupt function is enabled (BODRSTEN low), BOD will assert an interrupt if BODOUT is high. BOD interrupt will keep till to the BODEN set to 0. BOD interrupt can be blocked by disabling the NVIC BOD interrupt or disabling BOD function (set BODEN low). **Note2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2:1]	**BODVL**	**Brown-out Detector Threshold Voltage Selection (Write Protect)** The default value is set by flash controller user configuration register CBOV (CONFIG0 [22:21]). 00 = Brown-Out Detector threshold voltage is 2.2V. 01 = Brown-Out Detector threshold voltage is 2.7V. 10 = Brown-Out Detector threshold voltage is 3.7V. 11 = Brown-Out Detector threshold voltage is 4.5V. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Bits	Description	
[0]	**BODEN**	**Brown-out Detector Enable Bit (Write Protect)** The default value is set by flash controller user configuration register CBODEN (CONFIG0 [23]). 0 = Brown-out Detector function Disabled. 1 = Brown-out Detector function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Internal Voltage Source Control Register (SYS_IVSCTL)

Register	Offset	R/W	Description	Reset Value
SYS_IVSCTL	SYS_BA+0x1C	R/W	Internal Voltage Source 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							**VTEMPEN**

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

Power-on Reset Controller Register (SYS_PORCTL)

Register	Offset	R/W	Description	Reset Value
SYS_PORCTL	SYS_BA+0x24	R/W	Power-On-Reset Controller 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**
POROFF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POROFF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**POROFF**	**Power-on Reset Enable Bit (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:** These bits are write protected. Refer to the SYS_REGLCTL register.

GPIOA High Byte Multiple Function Control Register (SYS_GPA_MFPH)

Please refer to 4.3.7 GPIO Multi-function Pin Summary.

If input pin is selected to analog signal, GPIO Px_DINOFF register does not work and system will disable Px.n digital input path automatically to avoid input current leakage.

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFPH	SYS_BA+0x34	R/W	GPIOA High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PA15MFP				**PA14MFP**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PA13MFP				**PA12MFP**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PA11MFP				**PA10MFP**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**Reserved**			

Bits	Description	
[31:28]	**PA15MFP**	**PA.15 Multi-function Pin Selection**
[27:24]	**PA14MFP**	**PA.14 Multi-function Pin Selection**
[23:20]	**PA13MFP**	**PA.13 Multi-function Pin Selection**
[19:16]	**PA12MFP**	**PA.12 Multi-function Pin Selection**
[15:12]	**PA11MFP**	**PA.11 Multi-function Pin Selection**
[11:8]	**PA10MFP**	**PA.10 Multi-function Pin Selection**
[7:4]	**Reserved**	Reserved.
[3:0]	**Reserved**	Reserved.

GPIOB Low Byte Multiple Function Control Register (SYS_GPB_MFPL)

Please refer to 4.3.7 GPIO Multi-function Pin Summary.

If input pin is selected to analog signal, GPIO Px_DINOFF register does not work and system will disable Px.n digital input path automatically to avoid input current leakage.

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFPL	SYS_BA+0x38	R/W	GPIOB Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PB7MFP				PB6MFP			
23	22	21	20	19	18	17	16
PB5MFP				PB4MFP			
15	14	13	12	11	10	9	8
PB3MFP				PB2MFP			
7	6	5	4	3	2	1	0
PB1MFP				PB0MFP			

Bits	Description	
[31:28]	**PB7MFP**	**PB.7 Multi-function Pin Selection**
[27:24]	**PB6MFP**	**PB.6 Multi-function Pin Selection**
[23:20]	**PB5MFP**	**PB.5 Multi-function Pin Selection**
[19:16]	**PB4MFP**	**PB.4 Multi-function Pin Selection**
[15:12]	**PB3MFP**	**PB.3 Multi-function Pin Selection**
[11:8]	**PB2MFP**	**PB.2 Multi-function Pin Selection**
[7:4]	**PB1MFP**	**PB.1 Multi-function Pin Selection**
[3:0]	**PB0MFP**	**PB.0 Multi-function Pin Selection**

GPIOB High Byte Multiple Function Control Register (SYS_GPB_MFPH)

Please refer to 4.3.7 GPIO Multi-function Pin Summary.

If input pin is selected to analog signal, GPIO Px_DINOFF register does not work and system will disable Px.n digital input path automatically to avoid input current leakage.

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFPH	SYS_BA+0x3C	R/W	GPIOB High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PB15MFP				PB14MFP			
23	22	21	20	19	18	17	16
PB13MFP				PB12MFP			
15	14	13	12	11	10	9	8
Reserved				PB10MFP			
7	6	5	4	3	2	1	0
PB9MFP				PB8MFP			

Bits	Description	
[31:28]	**PB15MFP**	**PB.15 Multi-function Pin Selection**
[27:24]	**PB14MFP**	**PB.14 Multi-function Pin Selection**
[23:20]	**PB13MFP**	**PB.13 Multi-function Pin Selection**
[19:16]	**PB12MFP**	**PB.12 Multi-function Pin Selection**
[15:12]	**Reserved**	Reserved.
[11:8]	**PB10MFP**	**PB.10 Multi-function Pin Selection**
[7:4]	**PB9MFP**	**PB.9 Multi-function Pin Selection**
[3:0]	**PB8MFP**	**PB.8 Multi-function Pin Selection**

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

GPIOC Low Byte Multiple Function Control Register (SYS_GPC_MFPL)

Please refer to 4.3.7 GPIO Multi-function Pin Summary.

If input pin is selected to analog signal, GPIO Px_DINOFF register does not work and system will disable Px.n digital input path automatically to avoid input current leakage.

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFPL	SYS_BA+0x40	R/W	GPIOC Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PC5MFP				**PC4MFP**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PC3MFP				**PC2MFP**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PC1MFP				**PC0MFP**			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**Reserved**	Reserved.
[23:20]	**PC5MFP**	**PC.5 Multi-function Pin Selection**
[19:16]	**PC4MFP**	**PC.4 Multi-function Pin Selection**
[15:12]	**PC3MFP**	**PC.3 Multi-function Pin Selection**
[11:8]	**PC2MFP**	**PC.2 Multi-function Pin Selection**
[7:4]	**PC1MFP**	**PC.1 Multi-function Pin Selection**.
[3:0]	**PC0MFP**	**PC.0 Multi-function Pin Selection**

GPIOC High Byte Multiple Function Control Register (SYS_GPC_MFPH)

Please refer to 4.3.7 GPIO Multi-function Pin Summary.

If input pin is selected to analog signal, GPIO Px_DINOFF register does not work and system will disable Px.n digital input path automatically to avoid input current leakage.

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFPH	SYS_BA+0x44	R/W	GPIOC High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PC13MFP				**PC12MFP**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PC11MFP				**PC10MFP**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PC9MFP				**PC8MFP**			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**Reserved**	Reserved.
[23:20]	**PC13MFP**	**PC.13 Multi-function Pin Selection**
[19:16]	**PC12MFP**	**PC.12 Multi-function Pin Selection**
[15:12]	**PC11MFP**	**PC.11 Multi-function Pin Selection**
[11:8]	**PC10MFP**	**PC.10 Multi-function Pin Selection**
[7:4]	**PC9MFP**	**PC.9 Multi-function Pin Selection**
[3:0]	**PC8MFP**	**PC.8 Multi-function Pin Selection**

GPIOD Low Byte Multiple Function Control Register (SYS_GPD_MFPL)

Please refer to 4.3.7 GPIO Multi-function Pin Summary.

If input pin is selected to analog signal, GPIO Px_DINOFF register does not work and system will disable Px.n digital input path automatically to avoid input current leakage.

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFPL	SYS_BA+0x48	R/W	GPIOD Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				Reserved			
23	22	21	20	19	18	17	16
PD5MFP				PD4MFP			
15	14	13	12	11	10	9	8
PD3MFP				PD2MFP			
7	6	5	4	3	2	1	0
PD1MFP				PD0MFP			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**Reserved**	Reserved.
[23:20]	**PD5MFP**	**PD.5 Multi-function Pin Selection**
[19:16]	**PD4MFP**	**PD.4 Multi-function Pin Selection**
[15:12]	**PD3MFP**	**PD.3 Multi-function Pin Selection**
[11:8]	**PD2MFP**	**PD.2 Multi-function Pin Selection**
[7:4]	**PD1MFP**	**PD.1 Multi-function Pin Selection**
[3:0]	**PD0MFP**	**PD.0 Multi-function Pin Selection**

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

GPIOD High Byte Multiple Function Control Register (SYS_GPD_MFPH)

Please refer to 4.3.7 GPIO Multi-function Pin Summary.

If input pin is selected to analog signal, GPIO Px_DINOFF register does not work and system will disable Px.n digital input path automatically to avoid input current leakage.

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFPH	SYS_BA+0x4C	R/W	GPIOD High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**Reserved**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PD11MFP				**PD10MFP**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD9MFP				**PD8MFP**			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**Reserved**	Reserved.
[23:20]	**Reserved**	Reserved.
[19:16]	**Reserved**	Reserved.
[15:12]	**PD11MFP**	**PD.11 Multi-function Pin Selection**
[11:8]	**PD10MFP**	**PD.10 Multi-function Pin Selection**
[7:4]	**PD9MFP**	**PD.9 Multi-function Pin Selection**
[3:0]	**PD8MFP**	**PD.8 Multi-function Pin Selection**

GPIOE Low Byte Multiple Function Control Register (SYS_GPE_MFPL)

Please refer to 4.3.7 GPIO Multi-function Pin Summary.

If input pin is selected to analog signal, GPIO Px_DINOFF register does not work and system will disable Px.n digital input path automatically to avoid input current leakage.

Register	Offset	R/W	Description	Reset Value
SYS_GPE_MFPL	SYS_BA+0x50	R/W	GPIOE Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**Reserved**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**PE2MFP**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PE1MFP				**PE0MFP**			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**Reserved**	Reserved.
[23:20]	**Reserved**	Reserved.
[19:16]	**Reserved**	Reserved.
[15:12]	**Reserved**	Reserved.
[11:8]	**PE2MFP**	**PE.2 Multi-function Pin Selection**
[7:4]	**PE1MFP**	**PE.1 Multi-function Pin Selection**
[3:0]	**PE0MFP**	**PE.0 Multi-function Pin Selection**

GPIOF Low Byte Multiple Function Control Register (SYS_GPF_MFPL)

Please refer to 4.3.7 GPIO Multi-function Pin Summary.

If input pin is selected to analog signal, GPIO Px_DINOFF register does not work and system will disable Px.n digital input path automatically to avoid input current leakage.

Register	Offset	R/W	Description	Reset Value
SYS_GPF_MFPL	SYS_BA+0x58	R/W	GPIOF Low Byte Multiple Function Control Register	0x0011_0011

31	30	29	28	27	26	25	24
Reserved				**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PF5MFP				**PF4MFP**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PF3MFP				**PF2MFP**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PF1MFP				**PF0MFP**			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**Reserved**	Reserved.
[23:20]	**PF5MFP**	**PF.5 Multi-function Pin Selection**
[19:16]	**PF4MFP**	**PF.4 Multi-function Pin Selection**
[15:12]	**PF3MFP**	**PF.3 Multi-function Pin Selection**
[11:8]	**PF2MFP**	**PF.2 Multi-function Pin Selection**
[7:4]	**PF1MFP**	**PF.1 Multi-function Pin Selection**
[3:0]	**PF0MFP**	**PF.0 Multi-function Pin Selection**

Modulation Control Register (SYS_MODCTL)

Register	Offset	R/W	Description	Reset Value
SYS_MODCTL	SYS_BA+0xC0	R/W	Modulation 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**
MODPWMSEL				Reserved		MODH	MODEN

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:4]	**MODPWMSEL**	**BPWM2 Channel Select for Modulation** Select the BPWM2 channel to modulate with the UART0_TXD or USCI0_DAT0. 0000: BPWM2 Channel 0 modulate with UART0_TXD. 0001: BPWM2 Channel 1 modulate with UART0_TXD. 0010: BPWM2 Channel 2 modulate with UART0_TXD. 0011: BPWM2 Channel 3 modulete with UART0_TXD. 0100: BPWM2 Channel 4 modulete with UART0_TXD. 0101: BPWM2 Channel 5 modulete with UART0_TXD. 0110: Reserved. 0111: Reserved. 1000: BPWM2 Channel 0 modulate with USCI0_DAT0. 1001: BPWM2 Channel 1 modulate with USCI0_DAT0. 1010: BPWM2 Channel 2 modulate with USCI0_DAT0. 1011: BPWM2 Channel 3 modulete with USCI0_DAT0. 1100: BPWM2 Channel 4 modulete with USCI0_DAT0. 1101: BPWM2 Channel 5 modulete with USCI0_DAT0. 1110: Reserved. 1111: Reserved. **Note:** This bis is valid while MODEN (SYS_MODCTL[0]) is set to 1.
[3:2]	**Reserved**	Reserved.
[1]	**MODH**	**Modulation at Data High** Select modulation pulse(BPWM2) at UART0_TXD high or low 0: Modulation pulse at UART0_TXD or USCI0_DAT0 low. 1: Modulation pulse at UART0_TXD or USCI0_DAT0 high.
[0]	**MODEN**	**Modulation Function Enable Bit** This bit enables modulation funcion by modulating with BPWM2 channel output and

		USCI0(USCI0_DAT0) or UART0(UART0_TXD) output. 0 = Modulation Function Disabled. 1 = Modulation Function Enabled.

HIRC Trim Control Register (SYS_IRCTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTCTL	SYS_BA+0x80	R/W	HIRC Trim Control Register	0x0000_0000

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**REFCKSEL**	**Reference Clock Selection** 0 = HIRC trim reference clock is from external 32.768 kHz crystal oscillator. 1 = HIRC trim reference clock is from internal USB synchronous mode.
[9]	**Reserved**	Reserved.
[8]	**CESTOPEN**	**Clock Error Stop Enable Bit** 0 = The trim operation is keep going if clock is inaccuracy. 1 = The trim operation is stopped if clock is inaccuracy. **Note1:** CESTOPEN is set to 0 and FREQLOCK is set to 1, CLKERRIF will be set to 1 once and auto-trim function will stop trimming HIRC until LXT or internal USB synchronous mode works normally. **Note2:** After FREQLOCK is set to 1, CLKERRIF will be set to 1 only once and RC_TRIM will hang on and stop trimming RC until a reference clock is recovered when CESTOPEN is 0.
[7:6]	**RETRYCNT**	**Trim Value Update Limitation Count** This field defines that how many times the auto trim circuit will try to update the HIRC trim value before the frequency of HIRC locked. Once the HIRC locked, the internal trim value update counter will be reset. If the trim value update counter reached this limitation value and frequency of HIRC still doesn’t lock, the auto trim operation will be disabled and FREQSEL(SYS_IRCTCTL1[1:0]) will be cleared to 00. 00 = Trim retry count limitation is 64 loops. 01 = Trim retry count limitation is 128 loops. 10 = Trim retry count limitation is 256 loops. 11 = Trim retry count limitation is 512 loops.

[5:4]	**LOOPSEL**	**Trim Calculation Loop Selection** This field defines that trim value calculation is based on how many clocks of reference clock . 00 = Trim value calculation is based on average difference in 4 clocks of reference clock. 01 = Trim value calculation is based on average difference in 8 clocks of reference clock. 10 = Trim value calculation is based on average difference in 16 clocks of reference clock. 11 = Trim value calculation is based on average difference in 32 clocks of reference clock. **Note:** For example, if LOOPSEL is set as 00, auto trim circuit will calculate trim value based on the average frequency difference in 4 clocks of reference clock.
[3:2]	**Reserved**	Reserved.
[1:0]	**FREQSEL**	**Trim Frequency Selection** This field indicates the target frequency of internal high speed RC oscillator (HIRC) auto trim. During auto trim operation, if clock error detected with CESTOPEN(SYS_IRCTCTL1[8]) is set to 1 or trim retry limitation count reached, this field will be cleared to 00 automatically. 00 = Disable HIRC auto trim function. 01 = Reserved. 10 = Enable HIRC auto trim function and trim HIRC to 48 MHz. 11 = Reserved.

HIRC Trim Interrupt Enable Register (SYS_IRCTIEN)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTIEN	SYS_BA+0x84	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CLKEIEN**	**HIRC Clock Error Interrupt Enable Bit** This bit controls if CPU would get an interrupt while clock is inaccuracy during auto trim operation. If this bit is set to1, and CLKERRIF(SYS_IRCTSTS[2]) is set during auto trim operation, an interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = Disable CLKERRIF(SYS_IRCTSTS[2]) status to trigger an interrupt to CPU. 1 = Enable CLKERRIF(SYS_IRCTSTS[2]) status to trigger an interrupt to CPU.
[1]	**TFAILIEN**	**HIRC Trim Failure Interrupt Enable Bit** This bit controls if an interrupt will be triggered while HIRC trim value update limitation count reached and HIRC frequency still not locked on target frequency set by FREQSEL(SYS_IRCTCTL1[1:0]). If this bit is high and TFAILIF(SYS_IRCTSTS[1]) is set during auto trim operation, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. 0 = Disable TFAILIF(SYS_IRCTSTS[1]) status to trigger an interrupt to CPU. 1 = Enable TFAILIF(SYS_IRCTSTS[1]) status to trigger an interrupt to CPU.
[0]	**Reserved**	Reserved.

HIRC Trim Interrupt Status Register (SYS_IRCTISTS)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTISTS	SYS_BA+0x88	R/W	HIRC Trim Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CLKERRIF**	**HIRC Clock Error Interrupt Status** When the frequency of internal USB synchronous mode or 48 MHz internal high speed RC oscillator (HIRC) is shift larger to unreasonable value, this bit will be set and to be an indicate that clock frequency is inaccuracy Once this bit is set to 1, the auto trim operation stopped and FREQSEL(SYS_IRCTCL[1:0]) will be cleared to 00 by hardware automatically if CESTOPEN(SYS_IRCTCTL[8]) is set to 1. If this bit is set and CLKEIEN(SYS_IRCTIEN1[2]) is high, an interrupt will be triggered to notify the clock frequency is inaccuracy. Write 1 to clear this to 0. 0 = HIRC Clock frequency is accuracy. 1 = HIRC Clock frequency is inaccuracy.
[1]	**TFAILIF**	**HIRC Trim Failure Interrupt Status** This bit indicates that HIRC trim value update limitation count reached and the HIRC clock frequency still doesn't be locked. Once this bit is set, the auto trim operation stopped and FREQSEL(SYS_iRCTCTL[1:0]) will be cleared to 00 by hardware automatically. If this bit is set and TFAILIEN(SYS_IRCTIEN1[1]) is high, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. Write 1 to clear this to 0. 0 = HIRC trim value update limitation count does not reach. 1 = HIRC trim value update limitation count reached and frequency still not locked.
[0]	**FREQLOCK**	**HIRC Frequency Lock Status** This bit indicates the HIRC frequency is locked. This is a status bit and doesn't trigger any interrupt. Write 1 to clear this to 0. This bit will be set automatically, if the frequecy is lock and the RC_TRIM is enabled. 0 = The internal high-speed RC oscillator 1 frequency doesn't lock at 48 MHz yet. 1 = The internal high-speed RC oscillator 1 frequency locked at 48 MHz.

Register Lock Control Register (SYS_REGLCTL)

Register	Offset	R/W	Description	Reset Value
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock 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**
REGLCTL							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**REGLCTL**	**Register Lock Control Code (Write)** Some registers have write-protection function. Writing these registers have to disable the protected function by writing the sequence value “59h”, “16h”, “88h” to this field. After this sequence is completed, the REGLCTL bit will be set to 1 and write-protection registers can be normal write. **Register Lock Control Disable Index (Read)** 0 = Write-protection Enabled for writing protected registers. Any write to the protected register is ignored. 1 = Write-protection Disabled for writing protected registers. The Protected registers are: **SYS_IPRST0**: address 0x5000_0008 **SYS_BODCTL**: address 0x5000_0018 **SYS_PORCTL**: address 0x5000_0024 **SYS_TSOFFSET:** address 0x5000_0114 **CLK_PWRCTL**: address 0x5000_0200 (bit[6] is not protected for power-down wake-up interrupt clear) **CLK_APBCLK0 [0]**: address 0x5000_0208 (bit[0] is watchdog clock enable) **CLK_CLKSEL0**: address 0x5000_0210 (for HCLK and CPU STCLK clock source select) **CLK_CLKSEL1 [1:0]**: address 0x5000_0214 (for watchdog clock source select) **CLK_CLKSEL3 [8]**: address 0x5000_0234 (for USBD clock source select) **CLK_CLKDSTS:** address 0x5000_0274 **FMC_ISPCTL**: address 0x4000_C000 (Flash ISP Control register) **FMC_ISPTRG**: address 0x4000_C010 (ISP Trigger Control register) **FMC_ISPSTS**: address 0x4000_C040 **WDT_CTL**: address 0x4000_4000 **WDT_ALTCTL**: address 0x4000_4004 **FMC_FTCTL**: address 0x4000_5018 **FMC_ICPCMD**: address 0x4000_501C **EADC_TEST**: address 0x4004_3200

		AHBMCTL: address 0x40000400 **CLK_PLLCTL:** address 0x40000240 **PWM_CTL0:** address 0x4005_8000 **PWM_CTL0:** address 0x4005_9000 **PWM_DTCTL0_1:** address 0x4005_8070 **PWM_DTCTL0_1:** address 0x4005_9070 **PWM_DTCTL2_3:** address 0x4005_8074 **PWM_DTCTL2_3:** address 0x4005_9074 **PWM_DTCTL4_5:** address 0x4005_8078 **PWM_DTCTL4_5:** address 0x4005_9078 **PWM_BRKCTL0_1:** address 0x4005_80C8 **PWM_BRKCTL0_1:** address 0x4005_90C8 **PWM_BRKCTL2_3:** address 0x4005_80CC **PWM_BRKCTL2_3:** address 0x4005_90CC **PWM_BRKCTL4_5:** address 0x4005_80D0 **PWM_BRKCTL4_5:** address 0x4005_90D0 **PWM_INTEN1:** address 0x4005_80E4 **PWM_INTEN1:** address 0x4005_90E4 **PWM_INTSTS1:** address 0x4005_80EC **PWM_INTSTS1:** address 0x4005_90EC **PWM_SELFTEST:** address 0x4005_8300 **PWM_SELFTEST:** address 0x4005_9300

Temperature sensor offset Register (SYS_TSOFFSET)

Register	Offset	R/W	Description	Reset Value
SYS_TSOFFSET	SYS_BA+0x114	R	Temperature sensor offset Register	0x0000_0XXX

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**VTEMP**	**Temperature Sensor Offset Value** This field reflects temperature sensor output voltage offset at 25°C. Based on this value, with the value of temperature sensor output obtained from ADC conversion result and temperature sensor characteristics, it's easily to get temperature result by roughly calculation.

6.2.12 System Timer (SysTick)

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

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

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

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

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

6.2.12.1 System Timer Control Register Map

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

Register	Offset	R/W	Description	Reset Value
SYST Base Address: **SCS_BA = 0xE000_E000**				
SYST_CTRL	SCS_BA+0x10	R/W	SysTick Control and Status Register	0x0000_0000
SYST_LOAD	SCS_BA+0x14	R/W	SysTick Reload Value Register	0xXXXX_XXXX
SYST_VAL	SCS_BA+0x18	R/W	SysTick Current value Register	0xXXXX_XXXX

6.2.12.2 System Timer Control Register Description

SysTick Control and Status Register (SYST_CTRL)

Register	Offset	R/W	Description	Reset Value
SYST_CTRL	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 the (optional) external reference clock. 1 = Core clock used for SysTick.
[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 zero has occurred. 1 = Counting down to 0 will cause the SysTick exception to be pended. Clearing the SysTick current value register by a register write in software will not cause SysTick to be pended.
[0]	**ENABLE**	**System Tick Counter Enabled** 0 = Counter Disabled. 1 = Counter will operate in a multi-shot manner.

SysTick Reload Value Register (SYST_LOAD)

Register	Offset	R/W	Description	Reset Value
SYST_LOAD	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_VAL)

Register	Offset	R/W	Description	Reset Value
SYST_VAL	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. Unsupported bits RAZ (see SysTick Reload Value register).

6.2.13 Nested Vectored Interrupt Controller (NVIC)

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

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

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

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

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

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

6.2.13.1 Exception Model and System Interrupt Map

Table 6.2-4 lists the exception model supported by NUC121/125 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 Type	Vector Number	Vector Address	Priority
Reset	1	0x00000004	-3
NMI	2	0x00000008	-2
Hard Fault	3	0x0000000C	-1
Reserved	4 ~ 10		Reserved
SVCall	11	0x0000002C	Configurable
Reserved	12 ~ 13		Reserved
PendSV	14	0x00000038	Configurable
SysTick	15	0x0000003C	Configurable

Interrupt (IRQ0 ~ IRQ)	16 ~ 47	0x00000000 + (Vector Number)*4	Configurable

Table 6.2-7 Exception Model

Vector Number	**Interrupt Number (Bit In Interrupt Registers)**	**Interrupt Name**	**Interrupt Description**
0 ~ 15	-	-	System exceptions
16	0	BOD_INT	Brown-Out low voltage detected interrupt
17	1	WDT_INT	Window Watchdog Timer interrupt
18	2	EINT024	External interrupt from PB.14, PC.13, PC.12, PE.0 pin
19	3	EINT135	External interrupt from PD.11, PE.2, PB.15 pin
20	4	GPAB_INT	External signal interrupt from GPA/GPB
21	5	GPCDEF_INT	External interrupt from GPC/GPD/GPE/GPF
22	6	PWM0_INT	PWM0 interrupt
23	7	PWM1_INT	PWM1 interrupt
24	8	TMR0_INT	Timer 0 interrupt
25	9	TMR1_INT	Timer 1 interrupt
26	10	TMR2_INT	Timer 2 interrupt
27	11	TMR3_INT	Timer 3 interrupt
28	12	UART0_INT	UART0 interrupt
29	13	-	-
30	14	SPI0_INT	SPI0 interrupt
31	15	-	-
32	16	-	-
33	17	-	-
34	18	I2C0_INT	I^{2}C0 interrupt
35	19	I2C1_INT	I^{2}C1 interrupt
36	20	BPWM0_INT	BPWM0 interrupt
37	21	BPWM1_INT	BPWM1 interrupt
38	22	USCI_INT	USCI0 interrupt
39	23	USBD_INT	USB Device interrupt
40	24	-	-
41	25	PWM_BRAKE_INT	PWM Brake interrupt
42	26	PDMA_INT	PDMA interrupt

43	27	-	-
44	28	PWRWU_INT	Clock controller interrupt for chip wake-up from Power-down state
45	29	ADC_INT	ADC interrupt
46	30	CLKDIRC_INT	Clock fail detect and IRC TRIM interrupt
47	31	-	-

Table 6.2-8 Interrupt Number Table

6.2.13.2 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.

6.2.13.3 NVIC Control Registers

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

Register	Offset	R/W	Description	Reset Value
NVIC 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.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

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 Priority Register (NVIC_IPR0)

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

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

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

IRQ4 ~ IRQ7 Priority Register (NVIC_IPR1)

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

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

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

IRQ8 ~ IRQ11 Priority Register (NVIC_IPR2)

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

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

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

IRQ12 ~ IRQ15 Priority Register (NVIC_IPR3)

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

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

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

IRQ16 ~ IRQ19 Priority Register (NVIC_IPR4)

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

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

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

IRQ20 ~ IRQ23 Priority Register (NVIC_IPR5)

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

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

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

IRQ24 ~ IRQ27 Priority Register (NVIC_IPR6)

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

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

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

IRQ28 ~ IRQ31 Priority Register (NVIC_IPR7)

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

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

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

6.2.13.4 Interrupt Source Register Map

Besides the interrupt control registers associated with the NVIC, the NUC121/125 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 (EINT0/2/4) Interrupt Source Identity	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1/3/5) Interrupt Source Identity	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/GPB) Interrupt Source Identity	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (GPC/GPD/GPE/GPF) Interrupt Source Identity	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWM0) Interrupt Source Identity	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (PWM1) Interrupt Source Identity	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) Interrupt Source Identity	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) Interrupt Source Identity	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) Interrupt Source Identity	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) Interrupt Source Identity	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0) Interrupt Source Identity	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	Reserved	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) Interrupt Source Identity	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	Reserved	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	Reserved	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	Reserved	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I^{2}C0) Interrupt Source Identity	0xXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I^{2}C1) Interrupt Source Identity	0xXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	R	IRQ20 (BPWM0) Interrupt Source Identity	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (BPWM1) Interrupt Source Identity	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (USCI0) Interrupt Source Identity	0xXXXX_XXXX

IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USBD) Interrupt Source Identity	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	R	Reserved	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (PWM_BRAKE) Interrupt Source Identity	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (PDMA) Interrupt Source Identity	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	Reserved	0xXXXX_XXXX
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) Interrupt Source Identity	0xXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) Interrupt Source Identity	0xXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	R	IRQ30 (IRC/CLKD) Interrupt Source Identity	0xXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	R	Reserved	0xXXXX_XXXX
NMI_SEL	INT_BA+0x80	R/W	NMI Source Interrupt Select Control Register	0x0000_0000
MCU_IRQ	INT_BA+0x84	R/W	MCU Interrupt Request Source Register	0x0000_0000
MCU_IRQCR	INT_BA+0x88	R/W	MCU Interrupt Request Control Register	0x0000_0000

6.2.13.5 Interrupt Source Register Description

Interrupt Source Identity Register (IRQn_SRC)

Register	Offset	R/W	Description	Reset Value
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) Interrupt Source Identity	0xXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) Interrupt Source Identity	0xXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0/2/4) Interrupt Source Identity	0xXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1/3/5) Interrupt Source Identity	0xXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/PB) Interrupt Source Identity	0xXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (GPC/GPD/GPE/GPF) Interrupt Source Identity	0xXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWM0) Interrupt Source Identity	0xXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (PWM1) Interrupt Source Identity	0xXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) Interrupt Source Identity	0xXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) Interrupt Source Identity	0xXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) Interrupt Source Identity	0xXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) Interrupt Source Identity	0xXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0) Interrupt Source Identity	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	Reserved	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) Interrupt Source Identity	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	Reserved	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	Reserved	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	Reserved	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 (BPWM0) Interrupt Source Identity	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (BPWM1) Interrupt Source Identity	0xXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (USCI0) Interrupt Source Identity	0xXXXX_XXXX
IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USBD) Interrupt Source Identity	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	R	Reserved	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (PWM_BRAKE) Interrupt Source Identity	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (PDMA) Interrupt Source Identity	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	Reserved	0xXXXX_XXXX

IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) Interrupt Source Identity	0xXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) Interrupt Source Identity	0xXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	R	IRQ30 (IRC/CLKD) Interrupt Source Identity	0xXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	R	Reserved	0xXXXX_XXXX

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

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**INT_SRC**	**Interrupt Source** Define the interrupt sources for interrupt event.

Bits	Address	INT-Num	Description
[2:0]	INT_BA+0x00	0	Bit2: 0 Bit1: 0 Bit0: BOD_INT
[2:0]	INT_BA+0x04	1	Bit2: 0 Bit1: WWDT_INT Bit0: WDT_INT
[2:0]	INT_BA+0x08	2	Bit2: EINT4 – external interrupt 4 from PE.0 Bit1: EINT2 – external interrupt 2 from PC.0 Bit0: EINT0 – external interrupt 0 from PA.0/PD.2/PE.4
[2:0]	INT_BA+0x0C	3	Bit2: EINT5 – external interrupt 5 from PF.0 Bit1: EINT3 – external interrupt 3 from PD.0 Bit0: EINT1 – external interrupt 1 from PB.0/PD.3/PE.5
[2:0]	INT_BA+0x10	4	Bit2: 0 Bit1: PB_INT Bit0: PA_INT

[3:0]	INT_BA+0x14	5	Bit3: PF_INT Bit2: PE_INT Bit1: PD_INT Bit0: PC_INT
[3:0]	INT_BA+0x18	6	Bit2: 0 Bit1: 0 Bit0: PWM0_INT
[2:0]	INT_BA+0x1C	7	Bit2: 0 Bit1: 0 Bit0: PWM1_INT
[2:0]	INT_BA+0x20	8	Bit2: 0 Bit1: 0 Bit0: TMR0_INT
[2:0]	INT_BA+0x24	9	Bit2: 0 Bit1: 0 Bit0: TMR1_INT
[2:0]	INT_BA+0x28	10	Bit2: 0 Bit1: 0 Bit0: TMR2_INT
[2:0]	INT_BA+0x2C	11	Bit2: 0 Bit1: 0 Bit0: TMR3_INT
[2:0]	INT_BA+0x30	12	Bit2: 0 Bit1: 0 Bit0: UART0_INT
[2:0]	INT_BA+0x34	13	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x38	14	Bit2: 0 Bit1: 0 Bit0: SPI0_INT
[2:0]	INT_BA+0x3C	15	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x40	16	Reserved
[2:0]	INT_BA+0x44	17	Reserved
[2:0]	INT_BA+0x48	18	Bit2: 0 Bit1: 0 Bit0: I2C0_INT
[2:0]	INT_BA+0x4C	19	Bit2: 0 Bit1: 0 Bit0: I2C1_INT

[2:0]	INT_BA+0x50	20	Bit2: 0 Bit1: 0 Bit0: BPWM0_INT
[2:0]	INT_BA+0x54	21	Bit2: 0 Bit1: 0 Bit0: BPWM1_INT
[3:0]	INT_BA+0x58	22	Bit2: 0 Bit1: 0 Bit0: USCI0_INT
[2:0]	INT_BA+0x5C	23	Bit2: 0 Bit1: 0 Bit0: USBD_INT
[2:0]	INT_BA+0x60	24	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x64	25	Bit2: 0 Bit1: PWM1_BRAKE_INT Bit0: PWM0_BRAKE_INT
[2:0]	INT_BA+0x68	26	Bit2: 0 Bit1: 0 Bit0: PDMA_INT
[2:0]	INT_BA+0x6C	27	Reserved
[2:0]	INT_BA+0x70	28	Bit2: 0 Bit1: 0 Bit0: PWRWU_INT
[2:0]	INT_BA+0x74	29	Bit2: 0 Bit1: 0 Bit0: ADC_INT0
[2:0]	INT_BA+0x78	30	Bit2: SRAMF_INT Bit1: CLKD_INT Bit0: IRC_INT
[2:0]	INT_BA+0x7C	31	Reserved

Table 6.2-9 Interrupt Source Identity Detail Description

NMI Source Interrupt Select Control Register (NMI_SEL)

Register	Offset	R/W	Description	Reset Value
NMI_SEL	INT_BA+0x80	R/W	NMI Source Interrupt Select Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[8]	**NMI_EN**	**NMI Interrupt Enable Bit (Write Protect)** 0 = NMI interrupt Disabled. 1 = NMI interrupt Enabled. **Note:** This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register SYS_REGLCTL at address GCR_BA+0x100.
[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.

MCU Interrupt Request Source Register (MCU_IRQ)

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

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

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

6.2.14 System Control

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

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

6.2.14.1 System 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**				
SCS_CPUID	SCS_BA+0xD00	R	CPUID Register	0x410C_C200
SCS_ICSR	SCS_BA+0xD04	R/W	Interrupt Control and State Register	0x0000_0000
SCS_AIRCR	SCS_BA+0xD0C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000
SCS_SCR	SCS_BA+0xD10	R/W	System Control Register	0x0000_0000
SCS_SHPR2	SCS_BA+0xD1C	R/W	System Handler Priority Register 2	0x0000_0000
SCS_SHPR3	SCS_BA+0xD20	R/W	System Handler Priority Register 3	0x0000_0000

6.2.14.2 System Control Register Description

CPUID Register (CPUID)

Register	Offset	R/W	Description	Reset Value
SCS_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
SCS_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 not pending. 1 = NMI exception pending. Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it detects a write of 1 to this bit. Entering the handler then clears this bit to 0. This means a read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler.
[30:29]	**Reserved**	Reserved.
[28]	**PENDSVSET**	**PendSV Set-pending Bit** Write 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. 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 Operation: 0 = No effect. 1 = Removes the pending state from the SysTick exception. This is a write only bit. When you want to clear PENDST bit, you must “write 0 to PENDSTSET and write 1 to PENDSTCLR” at the same time.
[24]	**Reserved**	Reserved.
[23]	**ISRPREEMPT**	**If Set, a Pending Exception Will Be Serviced on Exit From the Debug Halt State** This bit is read only.
[22]	**ISRPENDING**	**Interrupt Pending Flag, Excluding NMI and Faults:** 0 = Interrupt not pending. 1 = Interrupt pending. This bit is read only.
[21:18]	**Reserved**	Reserved.
[17:12]	**VECTPENDING**	**Indicates the Exception Number of the Highest Priority Pending Enabled Exception:** 0 = No pending exceptions. Non-zero = Exception number of the highest priority pending enabled exception.
[11:6]	**Reserved**	Reserved.
[5:0]	**VECTACTIVE**	**Contains the Active Exception Number** 0 = Thread mode. Non-zero = Exception number of the currently active exception.

Application Interrupt and Reset Control Register (AIRCR)

Register	Offset	R/W	Description	Reset Value
SCS_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
SCS_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
SCS_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.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

System Handler Priority Register 3 (SHPR3)

Register	Offset	R/W	Description	Reset Value
SCS_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 clocks for the whole chip, including system clocks and all peripheral clocks. The clock controller also implements the power control function with the individually clock ON/OFF control, clock source selection and a clock divider. The chip will not enter Power-down mode until CPU sets the Power-down enable bit PDEN(CLK_PWRCTL[7]) and Cortex®-M0 core executes the WFI instruction. After that, chip enters Power-down mode and wait for wake-up interrupt source triggered to leave Power-down mode. In Power-down mode, the clock controller turns off the 4~24 MHz external high speed crystal (HXT) and 48 MHz internal high speed RC oscillator (HIRC) to reduce the overall system power consumption. Figure 6.3-1 shows the clock generator and the overview of the clock source control.

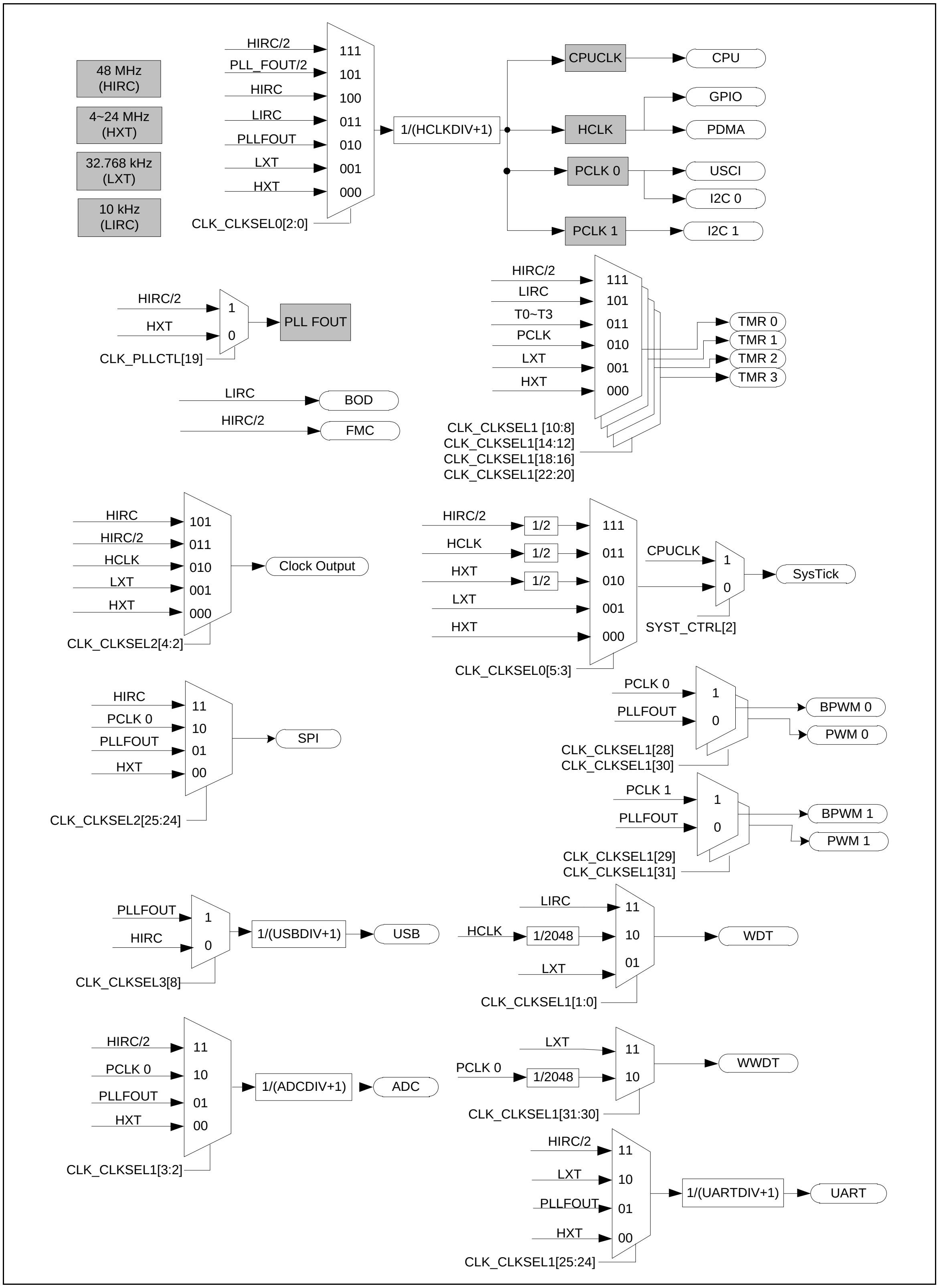


Figure 6.3-1 Clock Generator Global View Diagram

6.3.2 Clock Generator

The clock generator consists of 5 clock sources, which are listed below:

- 32.768 kHz external low-speed crystal oscillator (LXT)
- 4~24 MHz external high speed crystal oscillator (HXT)
- Programmable PLL output clock frequency (PLLFOUT), PLL source can be selected from external 4~24 MHz external high speed crystal (HXT) or 24 MHz (Internal high speed oscillator, HIRC/2)
- 48 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 (HIRCSTB(CLK_STATUS[4]), LIRCSTB(CLK_STATUS[3]), PLLSTB(CLK_STATUS[2]), HXTSTB(CLK_STATUS[0]) and LXTSTB(CLK_STATUS[1]) are set to 1 after stable counter value reach a define value as shown in the following table.

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 (LIRCEN(CLK_PWRCTL[3]), HIRCEN(CLK_PWRCTL[2]),XTLEN(CLK_PWRCTL[1:0]) and PD(CLK_PLLCTL[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	Clock Stable Time
HXT	4096 HXT clocks	341.33 uS for 12 Mhz

PLL	It’s based on the value of STBSEL (CLK_PLLCTL[23])
	STBSEL = 0, stable count is 6144 PLL clocks.
	STBSEL = 1, stable count is 12288 PLL clocks.(Default)
		STBSEL = 0
		122.88 uS for 50 Mhz
		STBSEL = 1:
		245.76 uS for 50 Mhz

HIRC	512 HIRC clocks	10.667 uS for 48Mhz
LIRC	1 LIRC clock	100 uS for 10 kHz
LXT	8192 LXT clock	250 mS for 32 KHz

Table 6.3-1 Clock Stable Count Value Table

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

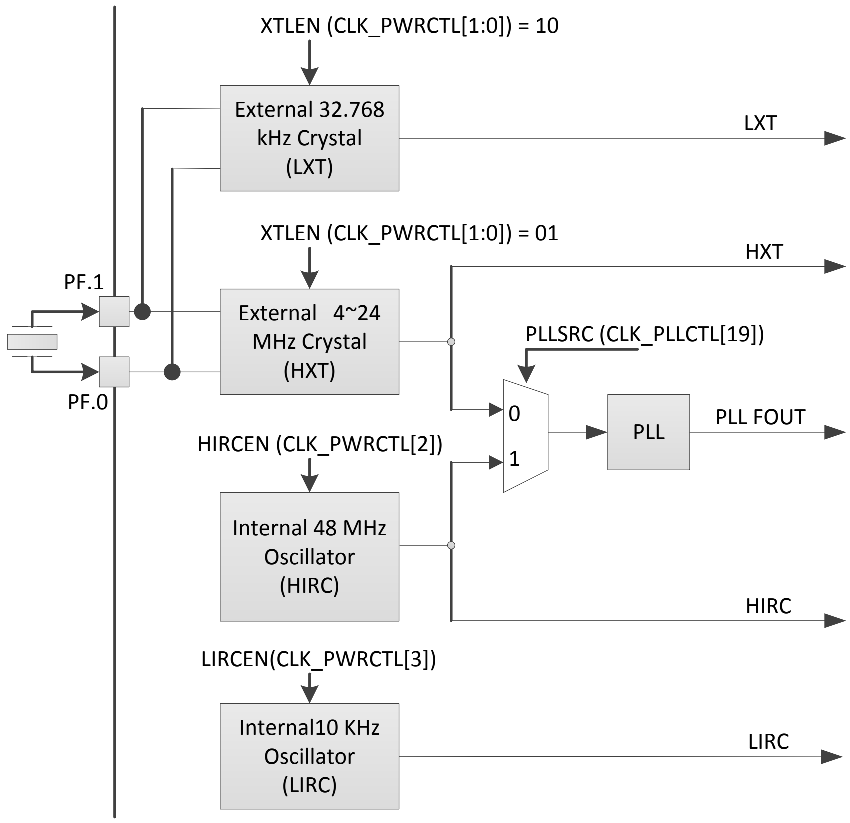

XTLEN (CLK_PWRCTL[1:0]) = 10
External 32.768 kHz Crystal (LXT)
LXT
XTLEN (CLK_PWRCTL[1:0]) = 01
HXT
PF.1
External 4~24 MHz Crystal (HXT)
PF.0
PLLSRC (CLK_PLLCTL[19])
0
1
PLL
PLL FOUT
HIRCEN (CLK_PWRCTL[2])
Internal 48 MHz Oscillator (HIRC)
HIRC
LIRCEN(CLK_PWRCTL[3])
Internal10 KHz Oscillator (LIRC)
LIRC

Figure 6.3-2 Clock Generator Block Diagram

6.3.3 System Clock and SysTick Clock

The system clock has 5 clock sources, which were generated from clock generator block. The clock source switch depends on the register HCLKSEL (CLK_CLKSEL0 [2:0]). The block diagram is shown in Figure 6.3-3

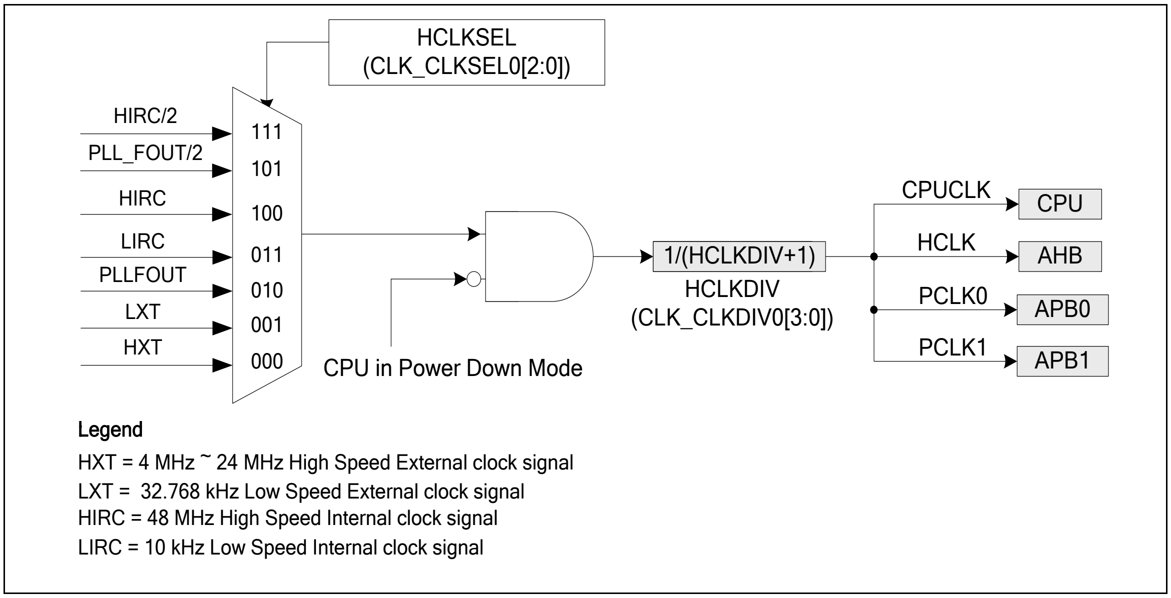

HCLKSEL
(CLK_CLKSEL0[2:0])
HIRC/2
PLL_FOUT/2
HIRC
LIRC
PLLFOUT
LXT
HXT
111
101
100
011
010
001
000
CPU in Power Down Mode
1/(HCLKDIV+1)
HCLKDIV
(CLK_CLKDIV0[3:0])
CPUCLK
HCLK
PCLK0
PCLK1
CPU
AHB
APB0
APB1
Legend
HXT = 4 MHz ~ 24 MHz High Speed External clock signal
LXT = 32.768 kHz Low Speed External clock signal
HIRC = 48 MHz High Speed Internal clock signal
LIRC = 10 kHz Low Speed Internal clock signal

Figure 6.3-3 System Clock Block Diagram

There are two clock fail detectors to observe HXT and LXT clock source if stop and they have individual enable and interrupt control. When HXT fail detector is enabled, the HIRC clock is enabled automatically. When LXT detector is enabled, the LIRC clock is enabled automatically.

When HXT clock fail detector is enabled, the system clock will auto switch to HIRC/2 (24 MHz) if HXT clock stop being detected on the following condition: system clock source comes from HXT or system clock source comes from PLL with HXT as the input of PLL. If HXT clock stop condition is detected, the HXTFIF (CLK_CLKDSTS[0]) is set to 1 and chip will enter interrupt if HXTFIE (CLK_CLKDCTL[5]) is set to 1. User can trying to recover HXT by disable HXT and enable HXT again to check if the clock stable bit is set to 1 or not. If HXT clock stable bit is set to 1, it means HXT is recover to oscillate after re-enable action and user can switch system clock to HXT again.

The HXT clock stop detect and system clock switch to HIRC/2 (24 MHz) procedure is shown in Figure 6.3-4

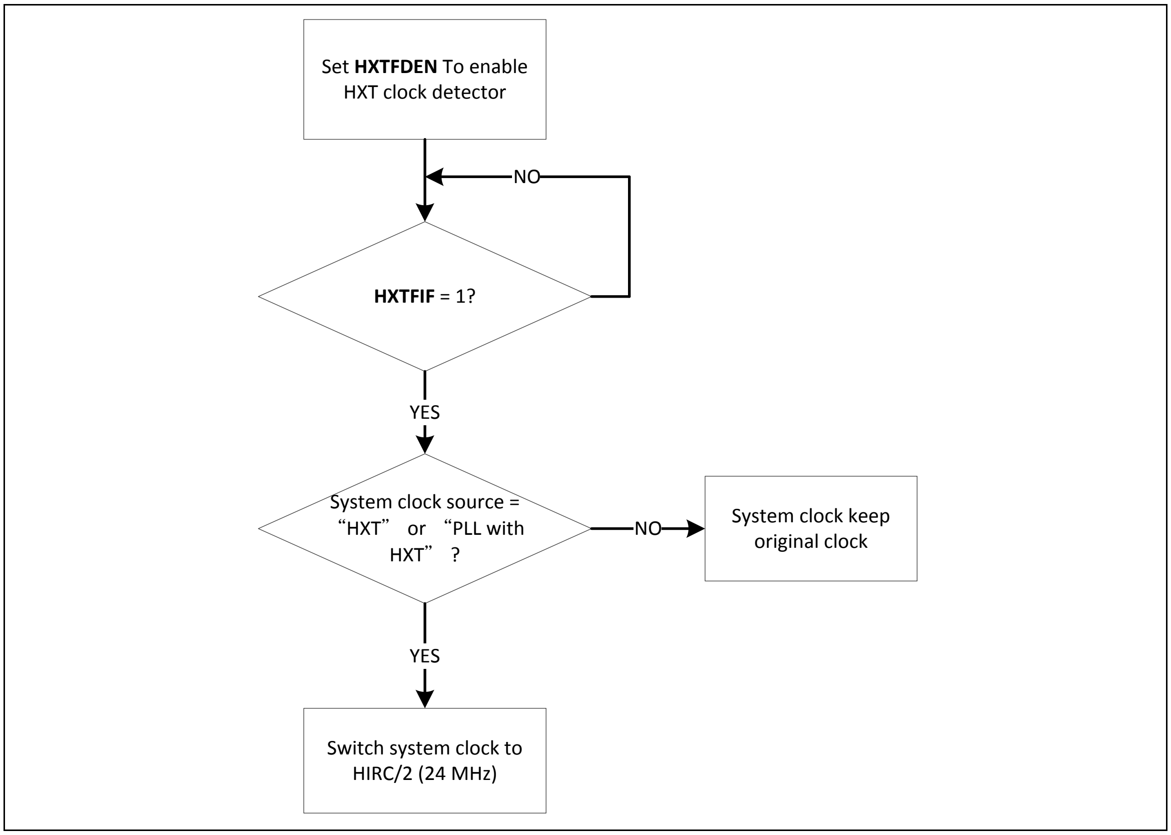

Set **HXTFDEN** To enable HXT clock detector
NO
HXTFIF = 1?
YES
System clock source = "HXT" or "PLL with HXT" ?
NO
System clock keep original clock
YES
Switch system clock to HIRC/2 (24 MHz)

Figure 6.3-4 HXT Stop Protect Procedure

Except HXT fail(stop) detector, HXT also has a frequency detector to observe HXT clock frequncy if normally and it also has individual enable(HXTFQDEN=CLK_CLKDCTL[16]) and interrupt control (HXTFQIEN=CLK_CLKDCTL[17]). When HXT frequency detector is enabled, the HIRC clock is enabled automatically. Otherwise, before HXT frequency detector is enabled, we need set the frequency detector upper boundary(UPERBD=CLK_CDUPB[9:0]) and lower boundary (LOWERBD=CLK_CDLOWB[9:0]).

If HXT clock frequency abnormally condition is detected, the HXTFQIF(CLK_CLKDSTS[8]) is set to 1 and chip will enter interrupt if HXTFQIEN (CLK_CLKDCTL[17]) is set to 1. Different with HXT fail(stop) detector, when HXT clock frequency abnormally condition is detected, the system clock will NOT auto switch to HIRC/2 (24 MHz) even though system clock source comes from HXT or system clock source comes from PLL with HXT as the input of PLL. The HXT frequency detector just reminds user HXT clock frequency abnormally through to observe HXTFQIF (CLK_CLKDSTS[8]).

The clock source of SysTick in Cortex®-M0 core can use CPU clock or external clock (SYST_CTRL[2]). If using external clock, the SysTick clock (STCLK) has 5 clock sources. The clock source switch depends on the setting of the register STCLKSEL (CLK_CLKSEL0[5:3]). The block diagram is shown in Figure 6.3-5

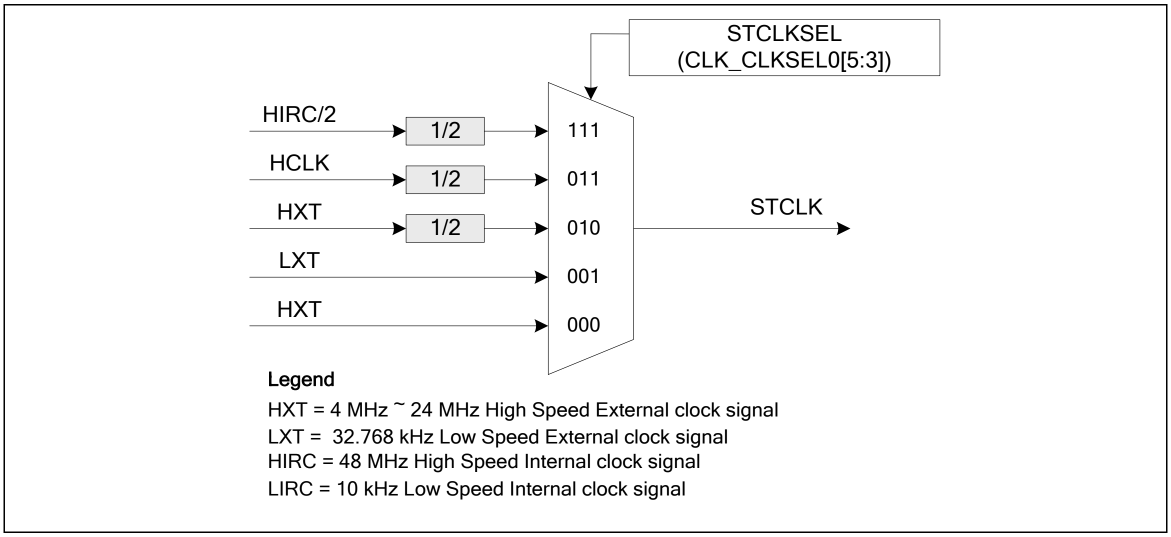

STCLKSEL
(CLK_CLKSEL0[5:3])
HIRC/2
HCLK
HXT
LXT
HXT
1/2
1/2
1/2
111
011
010
001
000
STCLK
Legend
HXT = 4 MHz ~ 24 MHz High Speed External clock signal
LXT = 32.768 kHz Low Speed External clock signal
HIRC = 48 MHz High Speed Internal clock signal
LIRC = 10 kHz Low Speed Internal clock signal

Figure 6.3-5 SysTick Clock Control Block Diagram

6.3.4 Peripherals Clock

The peripherals clock had different clock source switch setting, which depends on the different peripheral. Please refer to the CLK_CLKSEL1 and CLK_CLKSEL2 register description in 6.3.8 Register Description.

6.3.5 Power-down Mode Clock

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

For theses clocks, which still keep active, are listed below:

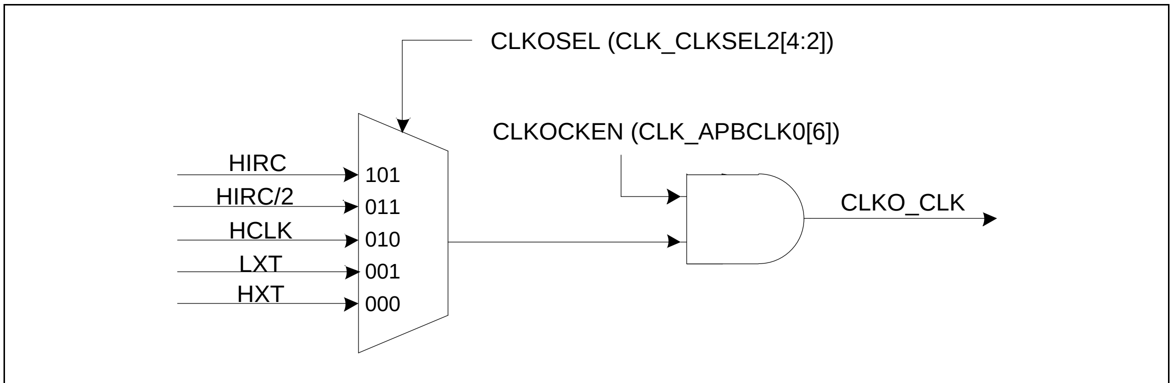
- Clock Generator
  - 10 kHz internal low-speed RC oscillator (LIRC) clock
  - 32.768 kHz external low-speed crystal oscillator (LXT) clock
- Peripherals Clock (When the modules adopt LXT or LIRC as clock source)

6.3.6 Clock 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 F_{in} 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 FREQSEL (CLK_CLKOCTL[3:0]).

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



Legend

HXT = 4 MHz ~ 24 MHz High Speed External clock signal

LXT = 32.768 kHz Low Speed External clock signal

HIRC = 48 MHz High Speed Internal clock signal

LIRC = 10 kHz Low Speed Internal clock signal

Figure 6.3-6 Clock Source of Clock Output

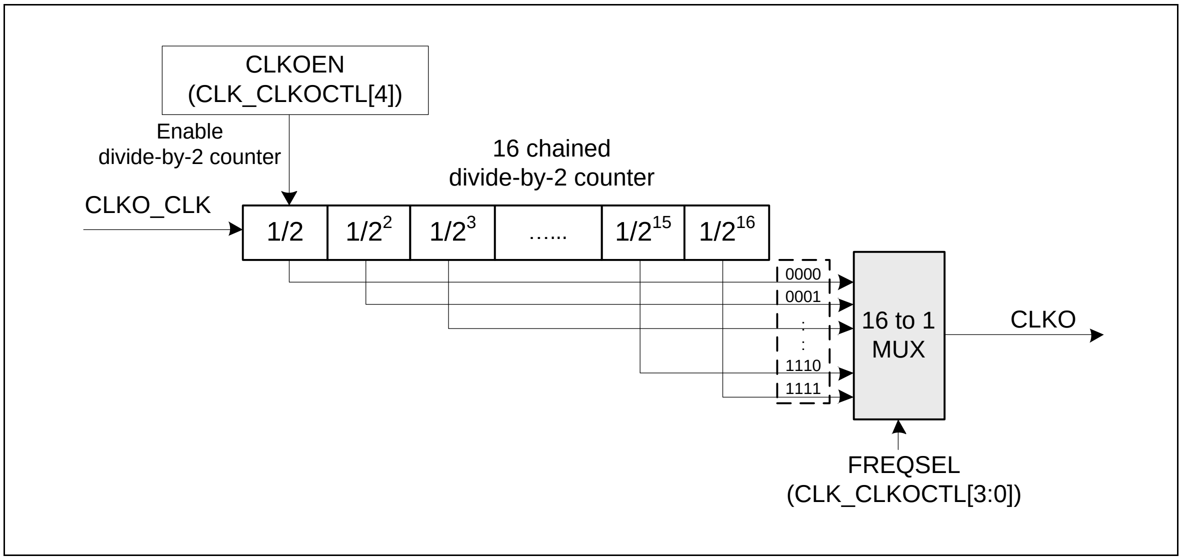


Figure 6.3-7 Clock Output Block Diagram

6.3.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
CLK Base Address: **CLK_BA = 0x5000_0200**				
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power-down Control Register	0x0000_1C1C
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x003F_8004
CLK_APBCLK0	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register 0	0x0000_0001
CLK_STATUS	CLK_BA+0x0C	R	Clock Status Monitor Register	0x0000_00XX
CLK_CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0000_003X
CLK_CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0xF377_770F
CLK_CLKDIV0	CLK_BA+0x18	R/W	Clock Divider Number Register 0	0x0000_0000
CLK_CLKSEL2	CLK_BA+0x1C	R/W	Clock Source Select Control Register 2	0x0002_0008
CLK_PLLCTL	CLK_BA+0x20	R/W	PLL Control Register	0x008D_C01E
CLK_CLKOCTL	CLK_BA+0x24	R/W	Clock Output Control Register	0x0000_0000
CLK_APBCLK1	CLK_BA+0x30	R/W	APB Devices Clock Enable Control Register 1	0x0000_0000
CLK_CLKSEL3	CLK_BA+0x34	R/W	Clock Source Select Control Register 3	0x0000_0000
CLK_CLKDCTL	CLK_BA+0x70	R/W	Clock Fail Detector Control Register	0x0000_0000
CLK_CLKDSTS	CLK_BA+0x74	R/W	Clock Fail Detector Status Register	0x0000_0000
CLK_CDUPB	CLK_BA+0x78	R/W	Clock Frequency Detector Upper Boundary Register	0x0000_0000
CLK_CDLOWB	CLK_BA+0x7c	R/W	Clock Frequency Detector Low Boundary Register	0x0000_0000

6.3.8 Register Description

System Power-down Control Register (CLK_PWRCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power-down Control Register	0x0000_1C1C

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:10]	**HXTGAIN**	**HXT Gain Control Bit (Write Protect)** This is a protected register. Please refer to open lock sequence to program it. Gain control is used to enlarge the gain of crystal to make sure crystal work normally. If gain control is enabled, crystal will consume more power than gain control off. 00 = HXT frequency is lower than from 8 MHz. 01 = HXT frequency is from 8 MHz to 12 MHz. 10 = HXT frequency is from 12 MHz to 16 MHz. 11 = HXT frequency is higher than 16 MHz. **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register.
[9:8]	**Reserved**	Reserved.
[7]	**PDEN**	**System Power-down Enable (Write Protect)** When this bit is set to 1, Power-down mode is enabled and chip keeps active till the CPU sleep mode is also active and then the chip enters Power-down mode. When chip wakes up from Power-down mode, this bit is auto cleared. Users need to set this bit again for next Power-down. In Power-down mode, HXT and the HIRC will be disabled in this mode, but LXT and LIRC are not controlled by Power-down mode. In Power-down mode, the PLL and system clock are disabled, and ignored the clock source selection. The clocks of peripheral are not controlled by Power-down mode, if the peripheral clock source is from LXT or LIRC. 0 = Chip operating normally or chip in idle mode because of WFI command. 1 = Chip enters Power-down mode instant or wait CPU sleep command WFI. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

[6]	**PDWKIF**	**Power-down Mode Wake-up Interrupt Status** Set by "Power-down wake-up event", it indicates that resume from Power-down mode" The flag is set if the EINT0~5, GPIO, USBD, UART0, WDT, BOD, TMR0~3 or I^{2}C0~1 wake-up occurred. **Note1:** Write 1 to clear the bit to 0. **Note2:** This bit works only if PDWKIEN (CLK_PWRCTL[5]) set to 1.
[5]	**PDWKIEN**	**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 PDWKIF and PDWKIEN are high. **Note2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[4]	**PDWKDLY**	**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 4~24 MHz external high speed crystal oscillator (HXT), and 512 clock cycles(selected by HIRCSTBS) when chip work at 48 MHz internal high speed RC oscillator (HIRC). 0 = Clock cycles delay Disabled. 1 = Clock cycles delay Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**LIRCEN**	**LIRC Enable Bit (Write Protect)** 0 = 10 kHz internal low speed RC oscillator (LIRC) Disabled. 1 = 10 kHz internal low speed RC oscillator (LIRC) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**HIRCEN**	**HIRC Enable Bit (Write Protect)** 0 = 48 MHz internal high speed RC oscillator (HIRC) Disabled. 1 = 48 MHz internal high speed RC oscillator (HIRC) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1:0]	**XTLEN**	**External HXT or LXT Crystal Oscillator Enable Bit (Write Protect)** The default clock source is from HIRC. These two bits are default set to "00" and the PF.0 and PF.1 pins are GPIO. 00 = HXT & LXT Disabled (default). 01 = HXT Enabled. 10 = LXT Enabled. 11 = Reserved. **Note 1:** These bits are write protected. Refer to the SYS_REGLCTL register. **Note 2:** These bits are over-written by CFOSC (CONFIG0[26]) after reset.

Register/Instruction Mode	**SLEEPDEEP (SCR[2])**	**PDEN (CLK_PWRCTL [7])**	**CPU Run WFI Instruction**	**Clock Disable**
Normal operation	0	0	NO	All clocks are disabled by control register.
Idle mode **(CPU enter Sleep mode)**	0	0	YES	Only CPU clock is disabled.

Power-down mode **(CPU enters Deep Sleep mode)**	1	1	YES	Most clocks are disabled except LIRC/LXT, and only WDT/Timer peripheral clocks still enable if their clock sources are selected as LIRC/LXT.

Table 6.3-2 Power-down Mode Control Table

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

AHB Devices Clock Enable Control Register (CLK_AHBCLK)

The bits in this register are used to enable/disable clock for system clock, AHB bus devices clock.

Register	Offset	R/W	Description	Reset Value
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x003F_8004

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

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21]	**GPIOFCKEN**	**General Purpose I/O PF Group Clock Enable Bit** 0 = GPIO PF group clock Disabled. 1 = GPIO PF group clock Enabled.
[20]	**GPIOECKEN**	**General Purpose I/O PE Group Clock Enable Bit** 0 = GPIO PE group clock Disabled. 1 = GPIO PE group clock Enabled.
[19]	**GPIODCKEN**	**General Purpose I/O PD Group Clock Enable Bit** 0 = GPIO PD group clock Disabled. 1 = GPIO PD group clock Enabled.
[18]	**GPIOCCKEN**	**General Purpose I/O PC Group Clock Enable Bit** 0 = GPIO PC group clock Disabled. 1 = GPIO PC group clock Enabled.
[17]	**GPIOBCKEN**	**General Purpose I/O PB Group Clock Enable Bit** 0 = GPIO PB group clock Disabled. 1 = GPIO PB group clock Enabled.
[16]	**GPIOACKEN**	**General Purpose I/O PA Group Clock Enable Bit** 0 = GPIO PA group clock Disabled. 1 = GPIO PA group clock Enabled.
[15:3]	**Reserved**	Reserved.
[2]	**ISPCKEN**	**Flash ISP Controller Clock Enable Bit** 0 = Flash ISP peripheral clock Disabled. 1 = Flash ISP peripheral clock Enabled.

[1]	**PDMACKEN**	**PDMA Controller Clock Enable Bit** 0 = PDMA peripheral clock Disabled. 1 = PDMA peripheral clock Enabled.
[0]	**Reserved**	Reserved.

APB Devices Clock Enable Control Register (CLK_APBCLK0)

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

Register	Offset	R/W	Description	Reset Value
CLK_APBCLK0	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register 0	0x0000_0001

31	30	29	28	27	26	25	24
Reserved			ADCCKEN	USBDCKEN	Reserved		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PWM1CKEN	PWM0CKEN	BPWM1CKEN	BPWM0CKEN	Reserved			UART0CKEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			SPI0CKEN	Reserved		I2C1CKEN	I2C0CKEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	CLKOCKEN	TMR3CKEN	TMR2CKEN	TMR1CKEN	TMR0CKEN	Reserved	WDTCKEN

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**ADCCKEN**	**Analog-digital-converter (ADC) Clock Enable Bit** 0 = ADC clock Disabled. 1 = ADC clock Enabled.
[27]	**USBDCKEN**	**USB Device Clock Enable Bit** 0 = USB Device clock Disabled. 1 = USB Device clock Enabled.
[26:24]	**Reserved**	Reserved.
[23]	**PWM1CKEN**	**PWM1 Clock Enable Bit** 0 = PWM1 clock Disabled. 1 = PWM1 clock Enabled.
[22]	**PWM0CKEN**	**PWM0 Clock Enable Bit** 0 = PWM0 clock Disabled. 1 = PWM0 clock Enabled.
[21]	**BPWM1CKEN**	**BPWM1 Clock Enable Bit** 0 = BPWM1 clock Disabled. 1 = BPWM1 clock Enabled.
[20]	**BPWM0CKEN**	**BPWM0 Clock Enable Bit** 0 = BPWM0 clock Disabled. 1 = BPWM0 clock Enabled.
[19:17]	**Reserved**	Reserved.

[16]	**UART0CKEN**	**UART0 Clock Enable Bit** 0 = UART0 clock Disabled. 1 = UART0 clock Enabled.
[15:13]	**Reserved**	Reserved.
[12]	**SPI0CKEN**	**SPI0 Clock Enable Bit** 0 = SPI0 Clock Disabled. 1 = SPI0 Clock Enabled.
[11:10]	**Reserved**	Reserved.
[9]	**I2C1CKEN**	**I2C1 Clock Enable Bit** 0 = I2C1 Clock Disabled. 1 = I2C1 Clock Enabled.
[8]	**I2C0CKEN**	**I2C0 Clock Enable Bit** 0 = I2C0 Clock Disabled. 1 = I2C0 Clock Enabled.
[7]	**Reserved**	Reserved.
[6]	**CLKOCKEN**	**CLKO Clock Enable Bit** 0 = CLKO Clock Disabled. 1 = CLKO Clock Enabled.
[5]	**TMR3CKEN**	**Timer3 Clock Enable Bit** 0 = Timer3 Clock Disabled. 1 = Timer3 Clock Enabled.
[4]	**TMR2CKEN**	**Timer2 Clock Enable Bit** 0 = Timer2 Clock Disabled. 1 = Timer2 Clock Enabled.
[3]	**TMR1CKEN**	**Timer1 Clock Enable Bit** 0 = Timer1 Clock Disabled. 1 = Timer1 Clock Enabled.
[2]	**TMR0CKEN**	**Timer0 Clock Enable Bit** 0 = Timer0 Clock Disabled. 1 = Timer0 Clock Enabled.
[1]	**Reserved**	Reserved.
[0]	**WDTCKEN**	**Watchdog Timer Clock Enable Bit (Write Protect)** 0 = Watchdog Timer Clock Disabled. 1 = Watchdog Timer Clock Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Clock Status Monitor Register (CLK_STATUS)

The bits in this register are used to monitor if the chip clock source is stable or not, and whether the clock switch is failed.

Register	Offset	R/W	Description	Reset Value
CLK_STATUS	CLK_BA+0x0C	R	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**
CLKSFAIL	**Reserved**		**HIRCSTB**	**LIRCSTB**	**PLLSTB**	**LXTSTB**	**HXTSTB**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**CLKSFAIL**	**Clock Switching Fail Flag (Read Only)** 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. 0 = Clock switching success. 1 = Clock switching failure. **Note 1:** Clock switch will finish automatically when target clcok is stable even if this bit already set to 1.
[6:5]	**Reserved**	Reserved.
[4]	**HIRCSTB**	**HIRC Clock Source Stable Flag (Read Only)** 0 = 48 MHz internal high speed RC oscillator (HIRC) clock is not stable or disabled. 1 = 48 MHz internal high speed RC oscillator (HIRC) clock is stabe and enabled.
[3]	**LIRCSTB**	**LIRC Clock Source Stable Flag (Read Only)** 0 = 10 kHz internal low speed RC oscillator (LIRC) clock is not stable or disabled. 1 = 10 kHz internal low speed RC oscillator (LIRC) clock is stable and enabled.
[2]	**PLLSTB**	**Internal PLL Clock Source Stable Flag (Read Only)** 0 = Internal PLL clock is not stable or disabled. 1 = Internal PLL clock is stable and enabled.
[1]	**LXTSTB**	**LXT Clock Source Stable Flag (Read Only)** 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock is not stable or disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) clock is stabled and enabled.
[0]	**HXTSTB**	**HXT Clock Source Stable Flag (Read Only)** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock is not stable or disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT)clock is stable and enabled.

Clock Source Select Control Register 0 (CLK_CLKSEL0)

Register	Offset	R/W	Description	Reset Value
CLK_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**
PCLK1SEL	**PCLK0SEL**	**STCLKSEL**			**HCLKSEL**		

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**PCLK1SEL**	**PCLK1 Clock Source Selection (Write Protect)** 0 = APB1 BUS clock source from HCLK clock. 1 = APB1 BUS clock source from HCLK/2. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6]	**PCLK0SEL**	**PCLK0 Clock Source Selection (Write Protect)** 0 = APB0 BUS clock source from HCLK clock. 1 = APB0 BUS clock source from HCLK/2. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5:3]	**STCLKSEL**	**Cortex®-M0 SysTick Clock Source Selection (Write Protect)** If SYST_CTRL[2]=0, SysTick uses listed clock source below. 000 = Clock source from HXT clock. 001 = Clock source from LXT clock. 010 = Clock source from HXT/2. 011 = Clock source from HCLK/2. 111 = Clock source from HIRC/4 (12 MHz). **Note:** if SysTick clock source is not from HCLK (i.e. SYST_CTRL[2] = 0), SysTick clock source must less than or equal to HCLK/2. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

[2:0]	**HCLKSEL**	**HCLK Clock Source Selection (Write Protect)** Before clock switching, the related clock sources (both pre-select and new-select) must be turned on. The 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 HXT clock. 001 = Clock source from LXT clock. 010 = Clock source from PLL clock. 011 = Clock source from LIRC clock. 100 = Clock source from HIRC clock. 101 = Clock source from PLL/2. 111 = Clock source from HIRC/2(24 MHz). Other = Reserved. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Clock Source Select Control Register 1 (CLK_CLKSEL1)

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

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0xF377_770F

31	30	29	28	27	26	25	24
PWM1SEL	PWM0SEL	BPWM1SEL	BPWM0SEL	Reserved		UARTSEL	
23	22	21	20	19	18	17	16
Reserved	TMR3SEL			Reserved	TMR2SEL		
15	14	13	12	11	10	9	8
Reserved	TMR1SEL			Reserved	TMR0SEL		
7	6	5	4	3	2	1	0
Reserved				ADCSEL		WDTSEL	

Bits	Description	
[31]	**PWM1SEL**	**PWM1 Clock Source Selection** The peripheral clock source of PWM1 is defined by PWM1SEL. 0 = Clock source from PLL clock. 1 = Clock source from PCLK1 clock.
[30]	**PWM0SEL**	**PWM0 Clock Source Selection** The peripheral clock source of PWM1 is defined by PWM0SEL. 0 = Clock source from PLL clock. 1 = Clock source from PCLK0 clock.
[29]	**BPWM1SEL**	**BPWM1 Clock Source Selection** The peripheral clock source of BPWM1 is defined by BPWM1SEL. 0 = Clock source from PLL clock. 1 = Clock source from PCLK1 clock.
[28]	**BPWM0SEL**	**BPWM0 Clock Source Selection** The peripheral clock source of BPWM0 is defined by BPWM0SEL. 0 = Clock source from PLL clock. 1 = Clock source from PCLK0 clock.
[27:26]	**Reserved**	Reserved.
[25:24]	**UARTSEL**	**UART Clock Source Selection** 00 = Clock source from HXT clock. 01 = Clock source from PLL clock. 10 = Clock source from LXT clock. 11 = Clock source from HIRC/2 clock.
[23]	**Reserved**	Reserved.

[22:20]	**TMR3SEL**	**TIMER3 Clock Source Selection** 000 = Clock source from HXT clock. 001 = Clock source from LXT clock. 010 = Clock source from PCLK1 clock. 011 = Clock source from external clock T3 pin. 101 = Clock source from LIRC clock. 111 = Clock source from HIRC/2 (24 MHz). Others = Reserved.
[19]	**Reserved**	Reserved.
[18:16]	**TMR2SEL**	**TIMER2 Clock Source Selection** 000 = Clock source from HXT clock. 001 = Clock source from LXT clock. 010 = Clock source from PCLK1 clock. 011 = Clock source from external clock T2 pin. 101 = Clock source from LIRC clock. 111 = Clock source from HIRC/2 (24 MHz). Others = Reserved.
[15]	**Reserved**	Reserved.
[14:12]	**TMR1SEL**	**TIMER1 Clock Source Selection** 000 = Clock source from HXT. 001 = Clock source from LXT. 010 = Clock source from PCLK0. 011 = Clock source from external clock T1 pin. 101 = Clock source from LIRC. 111 = Clock source from HIRC/2 (24 MHz). Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**TMR0SEL**	**TIMER0 Clock Source Selection** 000 = Clock source from HXT clock. 001 = Clock source from LXT clock. 010 = Clock source from PCLK0 clock. 011 = Clock source from external clock T0 pin. 101 = Clock source from LIRC clock. 111 = Clock source from HIRC/2 (24 MHz). Others = Reserved.
[7:4]	**Reserved**	Reserved.
[3:2]	**ADCSEL**	**ADC Peripheral Clock Source Selection** 00 = Clock source is from HXT clock. 01 = Clock source is from PLL clock. 10 = Clock source is from PCLK0 clock. 11 = Clock source is from HIRC/2 (24 MHz).

[1:0]	**WDTSEL**	**Watchdog Timer Clock Source Selection (Write Protect)** 00 = Reserved. 01 = Clock source from LXT clock. 10 = Clock source from HCLK/2048. 11 = Clock source from LIRC clock. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.

Clock Divider Number Register 0 (CLK_CLKDIV0)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV0	CLK_BA+0x18	R/W	Clock Divider Number Register 0	0x0000_0000

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

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

Clock Source Select Control Register 2 (CLK_CLKSEL2)

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

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL 2	CLK_BA+0x1C	R/W	Clock Source Select Control Register 2	0x0002_0008

31	30	29	28	27	26	25	24
Reserved						SPI0SEL	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						WWDTSEL	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			CLKOSEL			Reserved	

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:24]	**SPI0SEL**	**SPI0 Clock Source Selection** 00 = Clock source from HXT clock. 01 = Clock source from PLL clock. 10 = Clock source from PCLK0 clock. 11 = Clock source from HIRC clock.
[23:18]	**Reserved**	Reserved.
[17:16]	**WWDTSEL**	**Window Watchdog Timer Clock Source Selection** 10 = Clock source from HCLK/2048 clock. 11 = Clock source from LIRC clock. Others = Reserved.
[15:5]	**Reserved**	Reserved.
[4:2]	**CLKOSEL**	**Clock Divider Clock Source Selection** 000 = Clock source from HXT clock. 001 = Clock source from LXT clock. 010 = Clock source from HCLK clock. 011 = Clock source from HIRC/2 clock . 101 = Clock source from HIRC clock. Others = Reserved.
[1:0]	**Reserved**	Reserved.

PLL Control Register (CLK_PLLCTL)

The PLL reference clock input is from the 4~24 MHz external high speed crystal oscillator (HXT) clock input or from the 24 MHz internal high speed RC oscillator (HIRC/2). This register is used to control the PLL output frequency and PLL operation mode.

Register	Offset	R/W	Description	Reset Value
CLK_PLLCTL	CLK_BA+0x20	R/W	PLL Control Register	0x008D_C01E

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**STBSEL**	**PLL Stable Counter Selection** 0 = PLL stable time is 6144 PLL source clock (suitable for source clock is equal to or less than 12MHz). 1 = PLL stable time is 12288 PLL source clock (suitable for source clock is larger than 12MHz).
[22:20]	**Reserved**	Reserved.
[19]	**PLLSRC**	**PLL Source Clock Selection** 0 = PLL source clock from external 4~24 MHz high-speed crystal (HXT). 1 = PLL source clock from internal 24 MHz high-speed oscillator (HIRC/2).
[18]	**OE**	**PLL OE (FOUT Enable) Pin Control** 0 = PLL FOUT Enabled. 1 = PLL FOUT is fixed low.
[17]	**BP**	**PLL Bypass Control** 0 = PLL is in normal mode (default). 1 = PLL clock output is same as PLL input clock FIN.
[16]	**PD**	**Power-down Mode** If set the PDEN bit to 1 in CLK_PWRCTL register, the PLL will enter Power-down mode, too. 0 = PLL is in normal mode. 1 = PLL is in Power-down mode (default).
[15:14]	**OUTDIV**	**PLL Output Divider Control** Refer to the formulas below the table.

[13:9]	**INDIV**	**PLL Input Divider Control** Refer to the formulas below the table.
[8:0]	**FBDIV**	**PLL Feedback Divider Control** Refer to the formulas below the table.

Output Clock Frequency Setting

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

Constraint:

1. $4MHz < FIN < 24MHz$
2. $800KHz < \frac{FIN}{2*NR} < 8MHz$
3. $200MHz < FCO = FIN * \frac{NF}{NR} < 500MHz,$
   $FCO > 250MHz \;\; is \;\; preferred$
4. $FOUT \leq 100MHz$

Symbol	Description
FOUT	Output Clock Frequency
FIN	Input (Reference) Clock Frequency
NR	Input Divider (INDIV + 2)
NF	Feedback Divider (FBDIV + 2)

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

Table 6.3-3 PLL Output Clock Frequency Setting Description

Clock Output Control Register (CLK_CLKOCTL)

Register	Offset	R/W	Description	Reset Value
CLK_CLKOCTL	CLK_BA+0x24	R/W	Clock Output 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		DIV1EN	CLKOEN	FREQSEL			

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**DIV1EN**	**Clock Output Divide One Enable Bit** 0 = Clock Output will output clock with source frequency divided by FREQSEL. 1 = Clock Output will output clock with source frequency.
[4]	**CLKOEN**	**Clock Output Enable Bit** 0 = Clock Output function Disabled. 1 = Clock Output function Enabled.
[3:0]	**FREQSEL**	**Clock Output Frequency Selection** 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 FREQSEL[3:0].

APB Devices Clock Enable Control Register 1 (CLK_APBCLK1)

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

Register	Offset	R/W	Description	Reset Value
CLK_APBCLK1	CLK_BA+0x30	R/W	APB Devices Clock Enable Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							USCI0CKEN
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**USCI0CKEN**	**USCI0 Clock Enable Bit** 0 = USCI0 clock Disabled. 1 = USCI0 clock Enabled.
[7:0]	**Reserved**	Reserved.

Clock Source Select Control Register 3 (CLK_CLKSEL3)

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

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL3	CLK_BA+0x34	R/W	Clock Source Select Control Register 3	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**USBDSEL**	**USBD Clock Source Selection (Write Protect)** 0 = Clock source from HIRC. 1 = Clock source from PLL clock. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7:0]	**Reserved**	Reserved.

Clock Fail Detector Control Register (CLK_CLKDCTL)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDCTL	CLK_BA+0x70	R/W	Clock Fail Detector Control Register	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**HXTFQIEN**	**HXT Clock Frequency Monitor Interrupt Enable Bit** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock frequency monitor fail interrupt Disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock frequency monitor fail interrupt Enabled.
[16]	**HXTFQDEN**	**HXT Clock Frequency Monitor Enable Bit** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock frequency monitor Disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock frequency monitor Enabled.
[15:14]	**Reserved**	Reserved.
[13]	**LXTFIEN**	**LXT Clock Fail Interrupt Enable Bit** 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock Fail interrupt Disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) clock Fail interrupt Enabled.
[12]	**LXTFDEN**	**LXT Clock Fail Detector Enable Bit** 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock Fail detector Disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) clock Fail detector Enabled.
[11:6]	**Reserved**	Reserved.
[5]	**HXTFIEN**	**HXT Clock Fail Interrupt Enable Bit** 0 = 4~24 MHz external high speed crystal oscillator (HXT)clock Fail interrupt Disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT)clock Fail interrupt Enabled.
[4]	**HXTFDEN**	**HXT Clock Fail Detector Enable Bit** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock Fail detector Disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock Fail detector Enabled.
[3:0]	**Reserved**	Reserved.

Clock Fail Detector Status Register (CLK_CLKDSTS)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDSTS	CLK_BA+0x74	R/W	Clock Fail Detector 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							HXTFQIF
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						LXTFIF	HXTFIF

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**HXTFQIF**	**HXT Clock Frequency Monitor Interrupt Flag** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock normal. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock frequency abnormal. **Note:** Write 1 to clear the bit to 0.
[7:2]	**Reserved**	Reserved.
[1]	**LXTFIF**	**LXT Clock Fail Interrupt Flag** 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock normal. 1 = 32.768 kHz external low speed crystal oscillator (LXT) stop. **Note:** Write 1 to clear the bit to 0.
[0]	**HXTFIF**	**HXT Clock Fail Interrupt Flag** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock normal. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock stop. **Note:** Write 1 to clear the bit to 0.

Clock Frequency Monitor High Window Register (CLK_CDUPB)

Register	Offset	R/W	Description	Reset Value
CLK_CDUPB	CLK_BA+0x78	R/W	Clock Frequency Detector Upper Boundary Register	0x0000_0000

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

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>[31:10]</td><td><b>Reserved</b></td><td>Reserved.</td></tr>
<tr><td>[9:0]</td><td><b>UPERBD</b></td><td><b>HXT Clock Frequency Detector Upper Boundary</b>
The bits define the high value of frequency monitor window.
When HXT frequency monitor value higher than this register, the HXT frequency detect fail interrupt flag will set to 1.</td></tr>
</table>

Clock Frequency Monitor Low Window Register (CLK_CDLOWB)

Register	Offset	R/W	Description	Reset Value
CLK_CDLOWB	CLK_BA+0x7c	R/W	Clock Frequency Detector Low Boundary 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						LOWERBD	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LOWERBD							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**LOWERBD**	**HXT Clock Frequency Detector Low Boundary** The bits define the low value of frequency monitor window. When HXT frequency monitor value lower than this register, the HXT frequency detect fail interrupt flag will set to 1.

6.4 Flash Memory Controller (FMC)

6.4.1 Overview

The NUC121/125 series is equipped with 32 K-bytes on-chip embedded flash for application and Data Flash to store some application dependent data. A User Configuration block provides for system initialization. A 4.5 K-bytes loader ROM (LDROM) is used for In-System-Programming (ISP) function. A 512 bytes security protection ROM (SPROM) can conceal user program. This chip also supports In-Application-Programming (IAP) function, user switches the code executing without the chip reset after the embedded flash updated.

6.4.2 Features

- Supports 32 K-bytes application ROM (APROM).
- Supports 4.5 K-bytes loader ROM (LDROM).
- Supports configurable Data Flash size to share with APROM.
- Supports 512 bytes security protection ROM (SPROM) to conceal user program.
- Supports 12 bytes User Configuration block to control system initialization.
- Supports 512 bytes page erase for all embedded flash.
- Supports CRC-32 checksum calculation function (must be 512 bytes page alignment).
- Supports APROM, LDROM and embedded SRAM remap to system vector memory.
- Supports In-System-Programming (ISP) / In-Application-Programming (IAP) to update embedded flash memory.

6.4.3 Block Diagram

The flash memory controller (FMC) consists of AHB slave interface, flash control registers, flash initialization controller, flash operation control and embedded flash memory. Figure 6.4-1 shows the block diagram of flash memory controller.

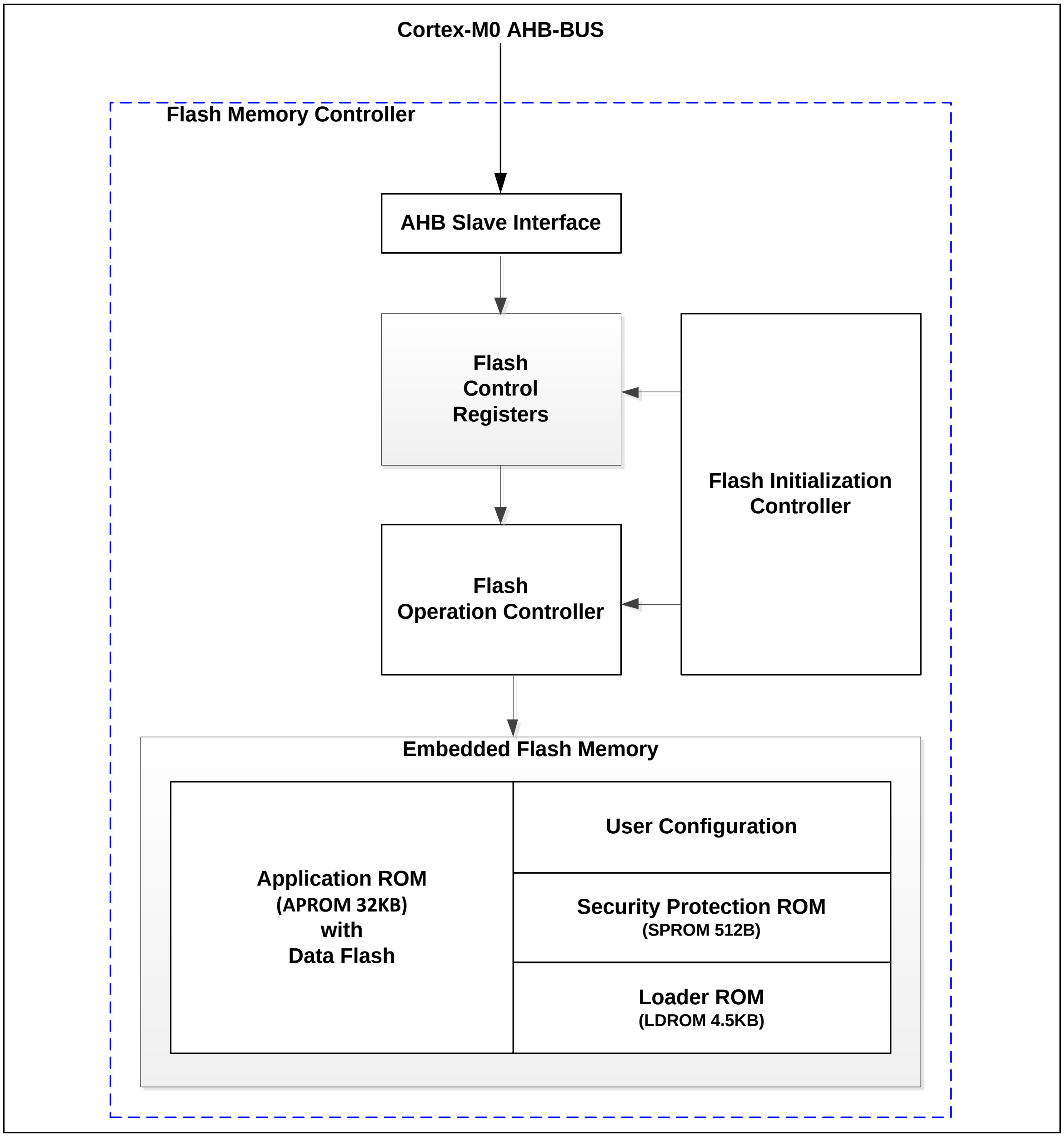


Figure 6.4-1 Flash Memory Control Block Diagram

AHB Slave Interface

There is a single AHB slave interface in flash memory controller for Cortex®-M0 to perform the instruction, data fetch and access ISP control registers.

Flash Control Registers

All of ISP control and status registers are in the flash control registers. The detail registers description is in the Register Description section

Flash Initialization Controller

When chip is powered on or active from reset, the flash initialization controller will start to access flash automatically and check the flash stability, and also reload User Configuration content to the flash control registers for system initialization.

Flash Operation Controller

The flash operations, such as checksum, flash erase, flash program, and flash read operation, have specific control timing for embedded flash memory. The flash operation controller generates those control timing by request from the flash control registers and the flash initialization controller.

Embedded Flash Memory

The embedded flash memory is the main memory for user application code and parameters. It is consists of the user configuration block, 4.5KB LDROM, 512B SPROM and 32KB APROM with Data Flash. The page erase flash size is 512B, and program bit width is 32 bits.

6.4.4 Functional Description

FMC functions include the memory organization, boot selection, IAP, ISP, the embedded flash programming, and checksum calculation. The flash memory map and system memory map are also introduced in the memory organization.

6.4.4.1 Memory Organization

The FMC memory consists of the embedded flash memory. The embedded flash memory is programmable, and includes APROM, LDROM, SPROM, Data Flash and the User Configuration block. The address map includes flash memory map and four types of system address map: LDROM with IAP, LDROM without IAP, APROM with IAP, and APROM without IAP functions.

6.4.4.2 LDROM, APROM and Data Flash

LDROM is designed for a loader to implement In-System-Programming (ISP) function by user. LDROM is a 4.5KB embedded flash memory, the flash address range is from 0x0010_0000 to 0x0010_1200. APROM is main memory for user applications. APROM size is 32KB. Data Flash is used to store application parameters (not instruction). Data Flash is shared with APROM and size is configurable. The base address of Data Flash is determined by DFBA (CONFIG1[19:0]). All of embedded flash memory is 512B page erased.

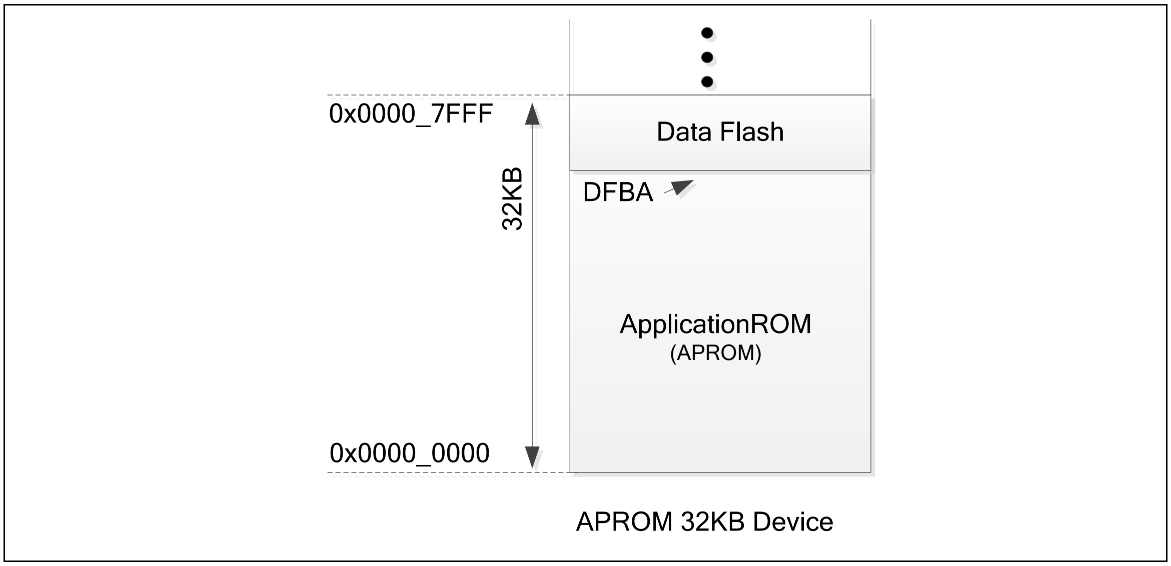


Figure 6.4-2 Data Flash is Shared with APROM

6.4.4.3 User Configuration Block

User Configuration block is internal programmable configuration area for boot options, such as flash security lock, boot select, brown-out voltage level, and Data Flash base address. It works like a fuse for power on setting. It is loaded from flash memory to its corresponding control registers during chip power on. User can set these bits according to different application requests. User Configuration block can be updated by ISP function and located at 0x0030_0000 with three 32 bits words (CONFIG0, CONFIG1 and CONFIG2). Any change on User Configuration block will take effect after system reboot

CONFIG0 (Address = 0x0030_0000)

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

Bits	Description	
[31]	**CWDTEN[2]**	**Watchdog Timer Hardware Enable Control Bits** 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 or active from reset pin. The clock source of watchdog timer is force at LIRC and LIRC can't be disabled. **CWDTEN[2:0]** is CONFIG0[31][4][3], 011 = 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. 111 = WDT hardware enable function is inactive. Others = WDT hardware enable function is active. WDT clock is always on.
[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[2:0] is set to 011
[29:28]	**Reserved**	Reserved.
[27]	**CFGXT1**	**PF[4:3] Multi-function Selection** 0 = PF[4:3] pins are configured as GPIO pins. 1 =PF[4:3] pins are configured as external 4~24 MHz external high speed crystal oscillator (HXT) pins.
[26]	**CFOSC**	**CPU Clock Source Selection After Reset** The value of CFOSC will be loaded to HCLK (CLK_CLKSEL0[2:0]) in system clock controller after any reset occurs. HCLK[2:0] = 111 if CFOSC = 1, HCLK[2:0] = 000 if CFGSC=0. 0 = 4~24 MHz external high speed crystal oscillator (HXT). 1 = 24 MHz, the internal high speed RC oscillator divided by 2 (HIRC/2).
[25:24]	**Reserved**	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]	**CBOV**	**Brown-out Voltage Selection** 00 = Brown-out voltage is 2.2V. 01 = Brown-out voltage is 2.7V. 10 = Brown-out voltage is 3.7V. 11 = Brown-out voltage is 4.5V.
[20]	**CBORST**	**Brown-out Reset Enable Bit** 0 = Brown-out reset Enabled after powered on. 1 = Brown-out reset Disabled after powered on.
[19:13]	**Reserved**	Reserved.
[12]	**ICELOCK**	**ICE Lock Bit** This bit only used to disable ICE function. User may use it with LOCK (CONFIG0[1]) bit to increase security level. 0 = ICE can't access Cortex®-M0 debug port. 1 = ICE can access Cortex®-M0 debug port.
[11]	**Reserved**	Reserved.
[10]	**CIOINI**	**I/O Initial State Selection** 0 = All GPIO set as input tri-state mode after powered on. 1 = All GPIO set as Quasi-bidirectional mode after chip powered on.
[9:8]	**Reserved**	Reserved.
[7:6]	**CBS**	**Chip Booting 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 (FMC_ISPCTL[1]) is only be used to control boot switching when CBS[0] = 1. VECMAP (FMC_ISPSTS[29:9]) is only be used to remap 0x0~0x1ff when CBS[0] = 0.
[5]	**MERASE**	**ISP Flash Mass Erase Enable Bit** 0 = ISP Flash Mass Erase command is Enabled. 1 = ISP Flash Mass Erase command is Disabled.
[4:3]	**CWDTEN[1:0]**	**Watchdog Timer Hardware Enable Bit** When watchdog timer hardware enable function is enabled, the watchdog enable bit WDTEN (WDT_CTL[7]) and watchdog reset enable bit RSTEN (WDT_CTL[1]) is set to 1 automatically after power on. The clock source of watchdog timer is force at LIRC and LIRC can't be disable. **CWDTEN[2:0]** is CONFIG0[31][4][3], 011 = WDT hardware enable function is active. WDT clock is always on except chip enter Power-down mode. When chip enter Power-down mode, WDT clock is always on if CWDTPDEN is 0 or WDT clock is controlled by LIRCEN (CLK_PWRCTL[3]) if CWDTPDEN is 1. Please refer to bit field description of CWDTPDEN. 111 = WDT hardware enable function is inactive. Others = WDT hardware enable function is active. WDT clock is always on.
[2]	**Reserved**	Reserved.

[1]	**LOCK**	**Security Lock Control** This bit is used to enable security lock function. If security lock function enabled, only the software in APROM, LDROM or SPROM could read correct data. Any other way, e.g. ICE, will get 0xffffffff when read APROM, LDROM or SPROM. 0 = Flash memory content is locked. 1 = Flash memory content is not locked.
[0]	**DFEN**	**Data Flash Enable Bit** The Data Flash is shared with APROM, and the base address of Data Flash is decided by DFBA (CONFIG1[19:0]) when DFEN is 0. 0 = Data Flash Enabled. 1 = Data Flash Disabled.

CONFIG1 (Address = 0x0030_0004)

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:0]	**DFBA**	**Data Flash Base Address** This register works only when DFEN (CONFIG0[0])set to 0. If DFEN (CONFIG0[0]) 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.

CONFIG2 (Address = 0x0030_0008)

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**ALOCK**	**Advanced Security Lock Control** 0x5A = Flash memory content is unlocked if LOCK (CONFIG0[1]) is 1, one of LOCK and ALOCK is locked then flash memory content is locked, only both unlocked, flash memory content unlocked. The others = Flash memory content is locked.

6.4.4.4 Security Protection Memory (SPROM)

The security protection memory (SPROM) is a special flash memory area for security applications. It supports independent lock machnism which is different to sercurity function of LOCK (CONFIG0[1]). In other words, user can lock SPROM without lock whole flash memory. This make it is possible for user to only protect their security key or key function in SPROM without lock all flash memory. Therefore, end customer still can develop or modify the application in APROM or LDROM when SPROM is locked.The SPROM includes 512 bytes at location address 0x20_0000 ~ 0x20_01FF and doesn’t support “MERASE command”. In Security and Debug mode, SPROM is only erased by “page erase command” with a key data, 0x0055AA03, in ISPDAT. Figure 6.4-3 shows that the last byte of SPROM (address: 0x0020_01FF) is used to identify the SPROM code is Non-secured, Debug or Security mode.

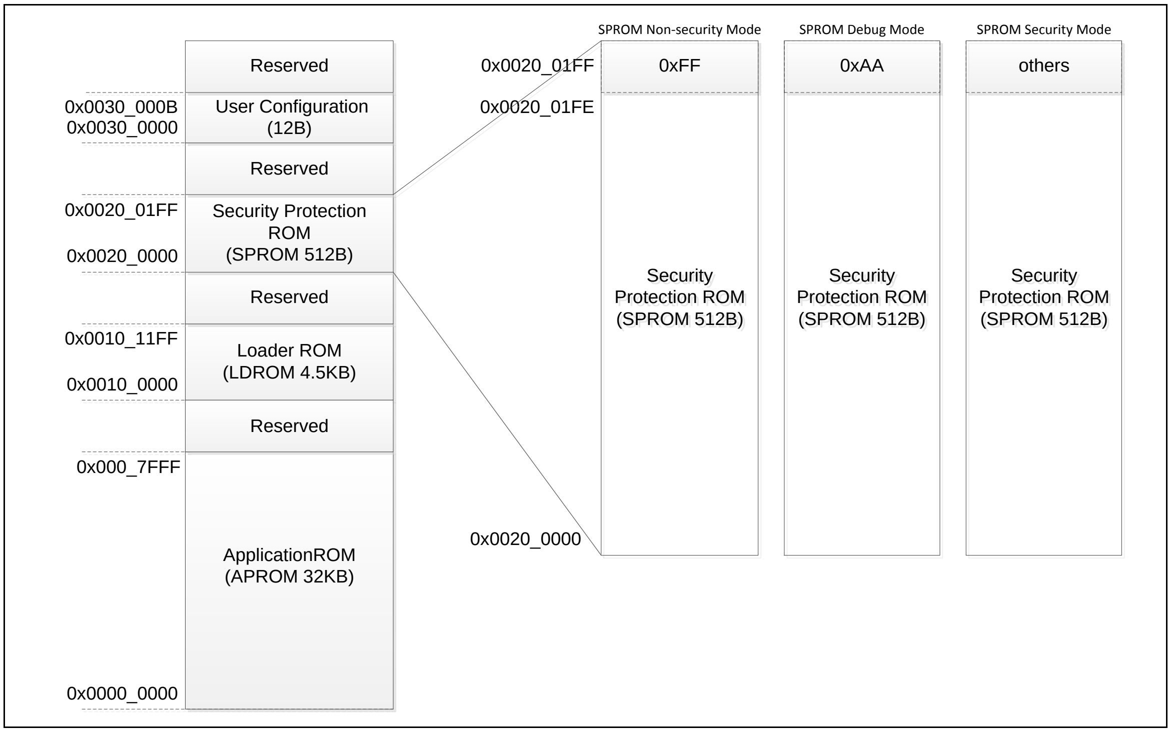


Figure 6.4-3 SPROM Security Mode

(1) SPROM non-secured mode (the last byte is 0xFF). The access behavior of SPROM is the same with APROM and LDROM. All area can be read by CPU or ISP command, and can be erased and programmed by ISP command.

(2) SPROM debug secured mode (the last byte is 0xAA). In order to debug easily, FMC controller accepts to execute program of SPROM when Cortex® -M0 ICE (In-Circuit-Emulator) port is connected. Other behaviors of SPROM are the same with SPROM secured mode.

(3) SPROM secured mode (the last byte is neither 0xFF nor 0xAA). In order to conceal SPROM code in secured mode, CPU only can perform instruction fetch and get data from SPROM when CPU is run at SPROM area. Otherwise, CPU will get all zero (0x0000_0000) for data access. In order to protect SPROM, the CPU instruction fetch will also get zero value when Cortex® -M0 ICE (In-Circuit-Emulator) port is connected in secured code. At this mode, SPROM doesn’t support ISP program and read flash command. It only supports page erase command with 0x0055AA03 data.

The SCODE (FMC_ISPSTS[31]) is SPROM secured flag to indicate that SPROM keeps secured mode, debug secured mode or not. It is set to 1 at flash initialization if the last byte of SPROM isn’t 0xFF, and can be cleared after the SPROM page erase operation complete. In order to easily test SPROM secured mode at normal run, it also can be set to 1 by user if the last byte of SPROM is 0xFF.

6.4.4.5 Flash Memory Map

In NUC121/125 series, the flash memory map is different from system memory map. The system memory map is used by CPU fetch code or data from FMC memory. The flash memory map is used for ISP function to read, program or erase FMC memory. Figure 6.4-4 shows the flash memory map.

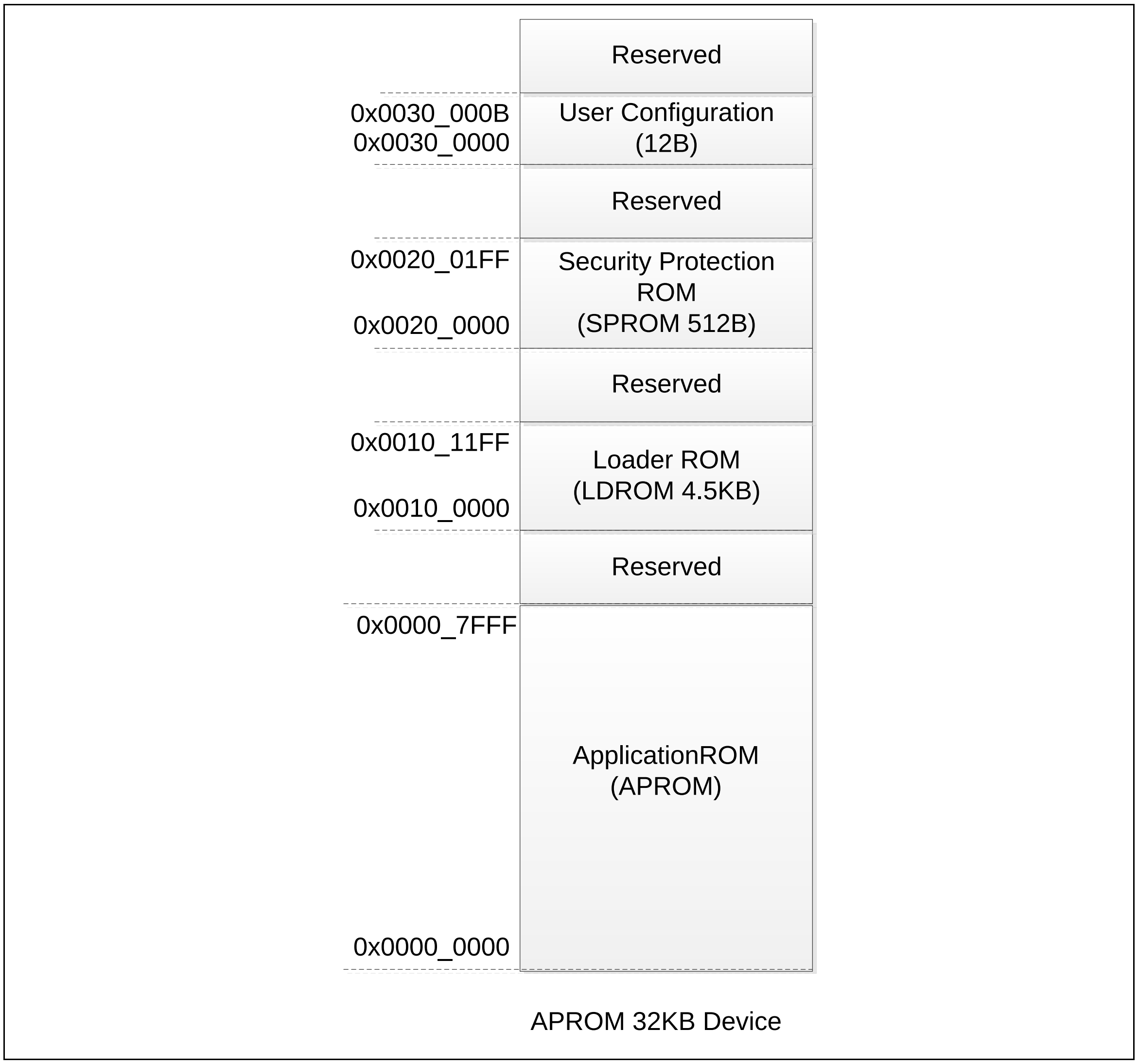


Figure 6.4-4 Flash Memory Map

6.4.4.6 System Memory Map with IAP mode

The system memory map is used by CPU to fetch code or data from FMC memory. In IAP mode, CPU can read and execute the code from APROM and LDROM. It also supports to call the funcitons in LDROM from APROM or call the funciton in APROM from LDROM. That is why it called IAP mode.

SPROM (0x0020_0000~0x0020_01FF) and LDROM (0x0010_0000~0x0010_07FF) address map both can be access as in the flash memory map. The Data Flash is shared with APROM and the Data Flash base address is defined by CONFIG1. The content of CONFIG1 is loaded into DFBA (Data Flash Base Address Register) at the flash initialization. The DFBA~0x0000_7FFF is the Data Flash region for Cortex®-M0 data access, and 0x0000_0200~(DFBA-1) is APROM region for Cortex®-M0 instruction access.

The address from 0x0000_0000 to 0x0000_01FF is called system memory vector. APROM and LDROM can map to the system memory vector for CPU start up. There are two kinds of system memory map with IAP mode when chip booting: (1) LDROM with IAP, and (2) APROM with IAP.

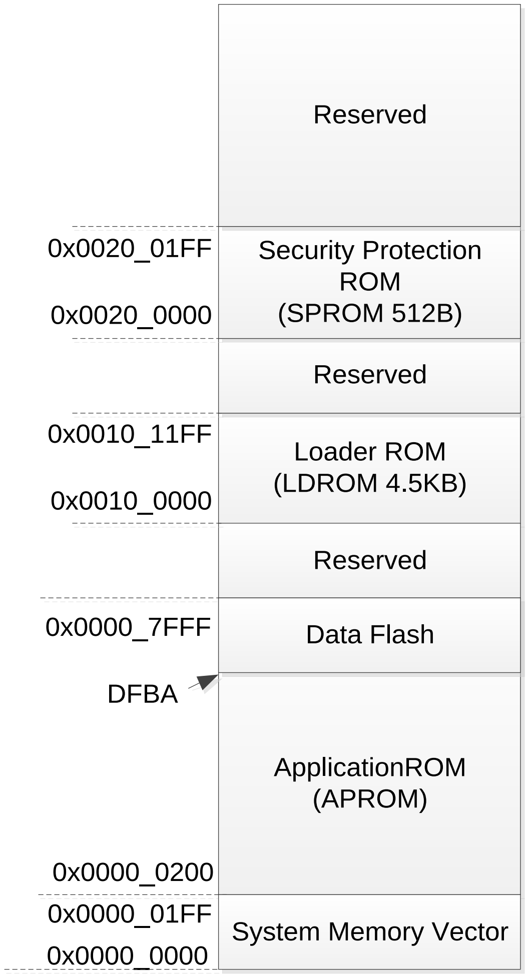


Figure 6.4-5 System Memory Map with IAP Mode

In LDROM with IAP mode, the default value of {VECMAP[11:0], 9'b0} is 0x100000 and first page of LDROM (0x0010_0000 ~ 0x0010_01FF) is mapping to the system memory vector for Cortex®-M0 instruction or data access.

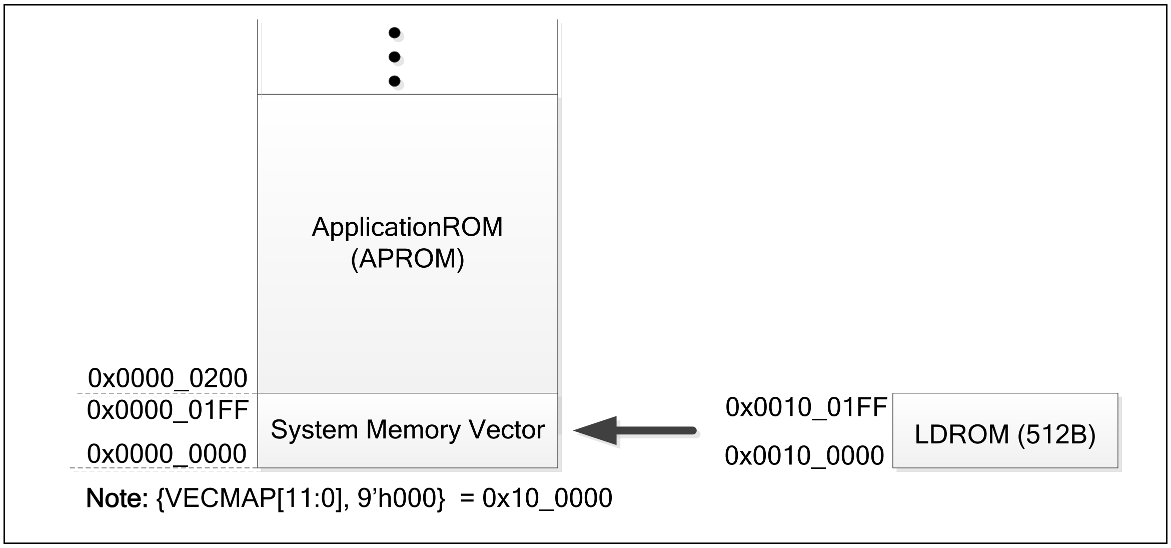


Figure 6.4-6 LDROM with IAP Mode

In APROM with IAP mode, the default value of {VECMAP[11:0], 9'b0} is 0x000000 and first page of APROM (0x0000_0000~0x0000_01FF) is mapping to the system memory vector for Cortex®-M0 instruction or data access.

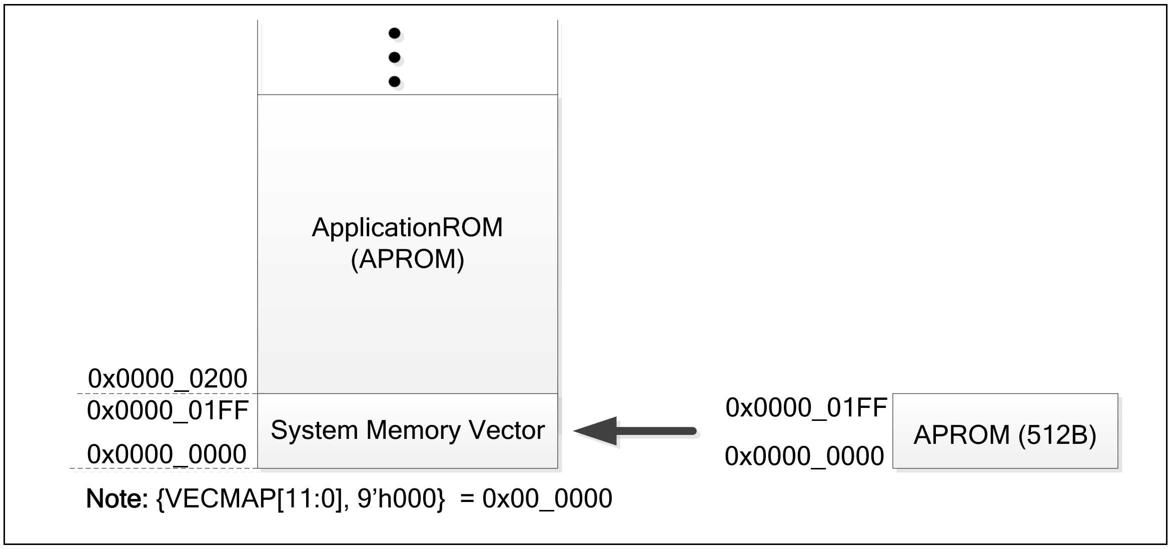


Figure 6.4-7 APROM with IAP Mode

In system memory map with IAP mode, the embedded SRAM, APROM and LDROM can remap to the system memory vector (align to 512 bytes) when CPU running. User can write the target remap address to FMC_ISPADDR register and then trigger ISP procedure with the "Vector Page Remap" command (0x2E). In VECMAP (FMC_ISPSTS[29:9]), shows the finial system memory vector mapping address. Please note that the vector mapping function only valid with IAP mode and it can't support SPROM.

6.4.4.7 System Memory Map without IAP mode

In system memory map without IAP mode, CPU still can access SPROM(0x0020_0000~0x0020_01FF), but the system memory vector mapping is not supported. There are two kinds of system memory map without IAP mode when chip booting: (1) LDROM without IAP, (2) APROM without IAP. In LDROM without IAP mode, LDROM base is mapping to 0x0000_0000. CPU program cannot run to access APROM. In APROM without IAP mode, APROM base is mapping to 0x0000_0000. CPU program cannot run to access LDROM. The Data Flash is shared with APROM and the Data Flash base address is defined by CONFIG1. The content of CONFIG1 is loaded into DFBA (Data Flash Base Address Register) at the flash initialization. The DFBA~0x0000_7FFF is the Data Flash region for Cortex®-M0 data access, and 0x0000_0000~(DFBA-1) is APROM region for Cortex®-M0 instruction access.

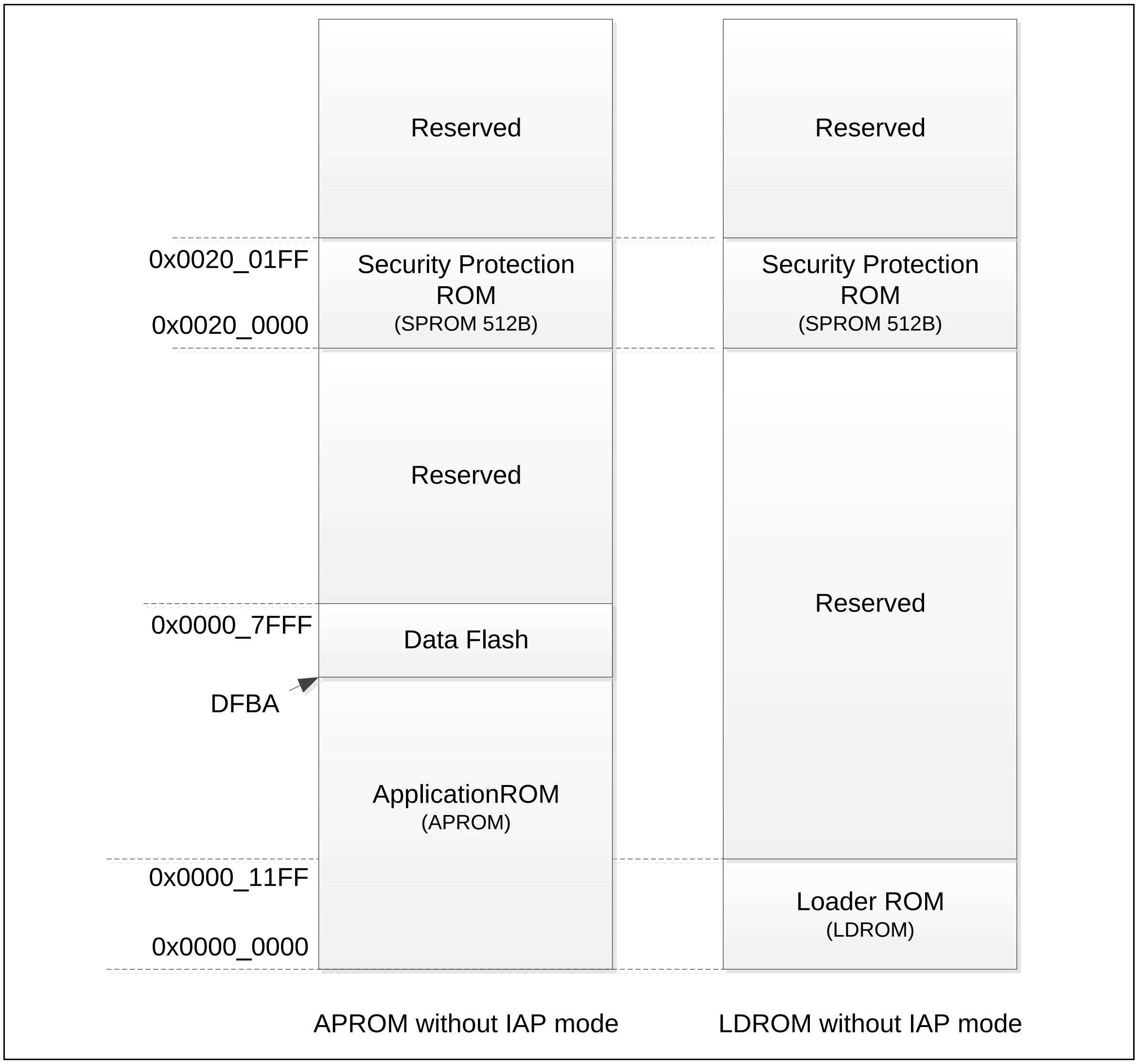


Figure 6.4-8 System Memory Map without IAP Mode

6.4.4.8 Boot Selection

The NUC121/125 series provides four booting sources, they are LDROM with IAP, LDROM without IAP, APROM with IAP, and APROM without IAP. The booting source and system memory map are setting by CBS (CONFIG0[7:6]).

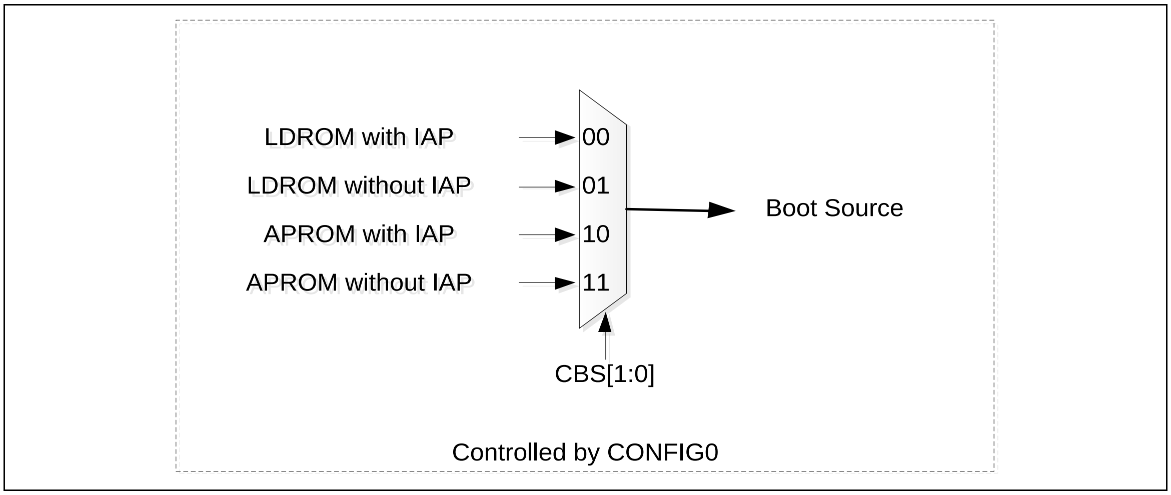


Figure 6.4-9 Boot Source Selection

CBS[1:0]	Boot Selection/System Memory Map	Vector Mapping Supporting
00	**LDROM with IAP**	**Yes**
01	**LDROM without IAP**	**No**
10	**APROM with IAP**	**Yes**
11	**APROM without IAP**	**No**

Table 6.4-1 Vector Mapping Supporting

6.4.4.9 In-Application-Programming (IAP)

The NUC121/125 series provides In-Application-Programming (IAP) function for user to switch the code executing between APROM, LDROM and SPROM. User can enable the IAP function by booting chip and setting the chip boot selection bits in CBS (CONFIG0[7:6]) as 10 or 00.

When chip boots with IAP function enabled, any executable code (align to 512 bytes) is allowed to map to the system memory vector(0x0000_0000~0x0000_01FF) any time. User can change the remap address to FMC_ISPADDR and then trigger ISP procedure with the “Vector Page Remap” command.

6.4.4.10 In-System-Programming (ISP)

The NUC121/125 series supports In-System-Programming (ISP) function allowing the embedded flash memory to be reprogrammed under software control. ISP is performed without removing the microcontroller from the system through the firmware and on-chip connectivity interface, such as UART, I^2C, and SPI.

The NUC121/125 series ISP provides the following functions for embedded flash memory.

- Supports flash page erase function
- Supports flash data program function
- Supports flash data read function
- Supports company ID read function

- Supports device ID read function
- Supports unique ID read function
- Supports memory checksum calculation function
- Supports system memory vector remap function

ISP Commands

ISP Command	FMC_ISPCMD	FMC_ISPADDR	FMC_ISPDAT
FLASH Page Erase	0x22	Valid address of flash memory origination. It must be 512 bytes alignment.	N/A
SPROM Page Erase	0x22	0x0020_0000	0x0055_AA03
FLASH 32-bit Program	0x21	Valid address of flash memory origination	FMC_ISPDAT :Programming Data
FLASH Read	0x00	Valid address of flash memory origination	FMC_ISPDAT: Return Data
Read Company ID	0x0B	0x0000_0000	FMC_ISPDAT: 0x0000_00DA
Read Checksum	0x0D	Keep address of "Run Checksum Calculation"	FMC_ISPDAT: Return Checksum
Run Checksum Calculation	0x2D	Valid start address of memory origination It must be 512 bytes alignment	FMC_ISPDAT: Size It must be 512 bytes alignment
Read Unique ID	0x04	0x0000_0000	FMC_ISPDAT: Unique ID Word 0
		0x0000_0004	FMC_ISPDAT: Unique ID Word 1
		0x0000_0008	FMC_ISPDAT: Unique ID Word 2
Vector Remap	0x2E	Valid address in APROM, LDROM or SRAM with 512 bytes alignment	N/A

Table 6.4-2 ISP Command List

ISP Procedure

The FMC controller provides embedded flash memory read, erase and program operation. Several control bits of FMC control register are write-protected, thus it is necessary to unlock before setting.

After unlocking the protected register bits, user needs to set the FMC_ISPCTL control register to decide to update LDROM, APROM, SPROM or user configuration block, and then set ISPEN (FMC_ISPCTL[0]) to enable ISP function.

Once the FMC_ISPCTL register is set properly, user can set FMC_ISPCMD (refer above ISP command list) for specific operation. Set FMC_ISPADDR for target flash memory based on flash memory origination. FMC_ISPDAT can be used to set the data to program or used to return the read data according to FMC_ISPCMD.

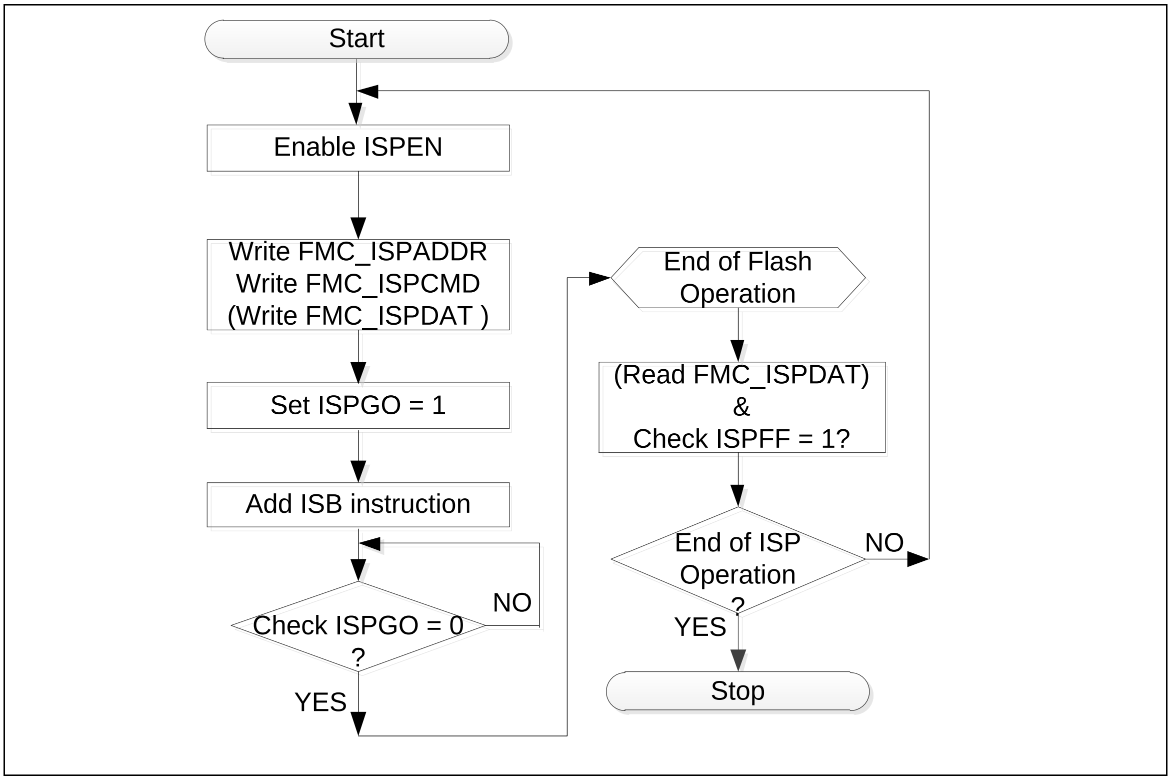


Figure 6.4-6.4-10 ISP Procedure Example

Finally, set ISPGO (FMC_ISPTRG[0]) register to perform the relative ISP function. The ISPGO(FMC_ISPTRG[0]) bit is self-cleared when ISP function has been done. To make sure ISP function has been finished before CPU goes ahead, ISB (Instruction Synchronization Barrier) instruction is used right after ISPGO(FMC_ISPTRG[0]) setting.

Several error conditions will be checked after ISP is completed. If an error condition occurs, ISP operation is not started and the ISP fail flag will be set instead. ISPFF(FMC_ISPSTS[6]) flag can only be cleared by software. User must check the ISPFF(FMC_ISPSTS[6]) bit and clear it after each ISP operation if it is set to 1.

When the ISPGO(FMC_ISPTRG[0]) bit is set, FMC start to process ISP command, CPU will be halt to wait ISP done if CPU trying to access flash memory. For example, if any interrupt request occurs, CPU will not service it till ISP operation is finished. User could move their code and exception handlers to SRAM to avoid this situation. The peripheral still keeps working as usual when ISP processing. When ISP operation is finished, the ISPGO bit will be cleared by hardware automatically. User can check whether ISP operation is finished or not by the ISPGO(FMC_ISPTRG[0]) bit. User should add ISB (Instruction Synchronization Barrier) instruction next to the instruction in which ISPGO (FMC_ISPTRG[0]) bit is set 1 to ensure correct execution of the instructions following ISP operation.

6.4.4.11 CRC32 Checksum Calculation

The NUC121/125 series supports the Cyclic Redundancy Check (CRC-32) checksum calculation function to help user quickly check the memory content includes APROM, LDROM and SPROM. The CRC-32 polynomial is below

$$\text{CRC-32: } X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X + 1$$

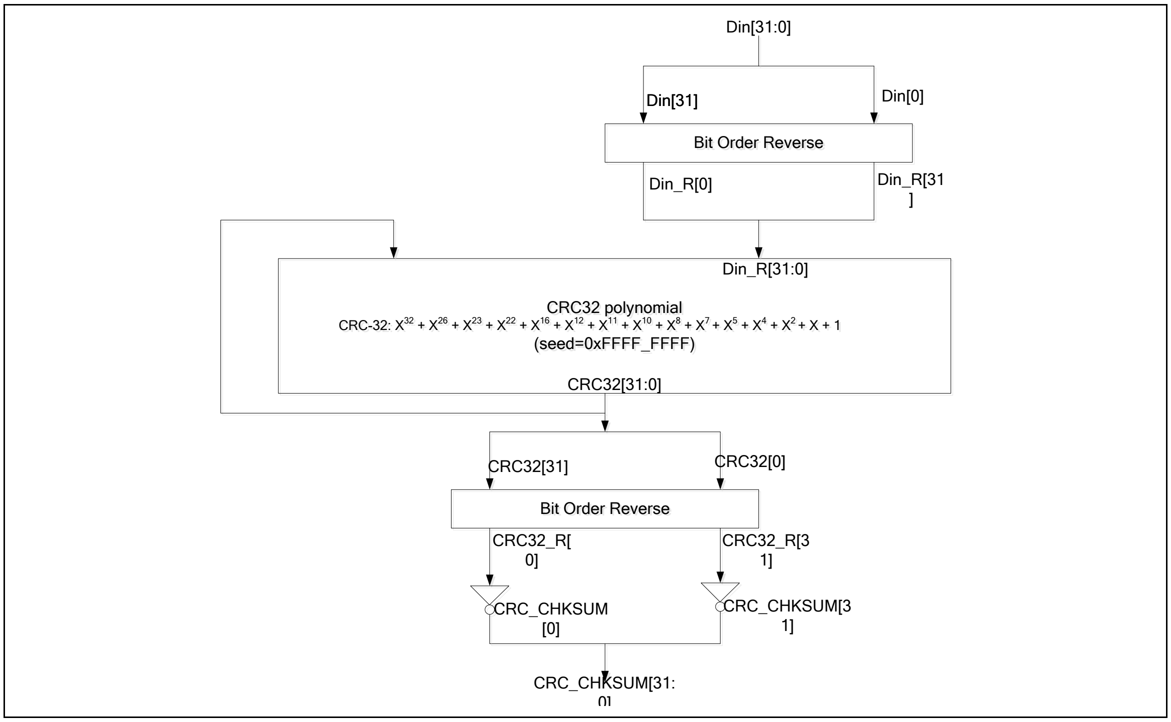


Figure 6.4-6.4-11 CRC-32 Checksum Calculation

Three steps complete this CRC-32 checksum calculation.

(Step 1) perform ISP “Run Memory Checksum” operation: user has to set the memory starting address (FMC_ISPADDR) and size (FMC_ISPDAT) to calculate. Both address and size have to be 512 bytes alignment, the size should be multiples of 512 bytes and the starting address needs to be located in APROM, LDROM and SPROM’s memory space.

(Step 2) perform ISP “Read Memory Checksum” operation: the FMC_ISPADDR should be kept as the same as step 1.

(Step 3) read FMC_ISPDAT to get checksum: The checksum is read from FMC_ISPDAT. If the checksum is 0x0000_0000, it must be one of two conditions (1) Checksum calculation is in-progress, (2) Address and size is over device limitation.

When SPROM is set to security mode, CPU and ISP read command cannot read the SPROM content directly but user can use this checksum function to verify SPROM content correction.

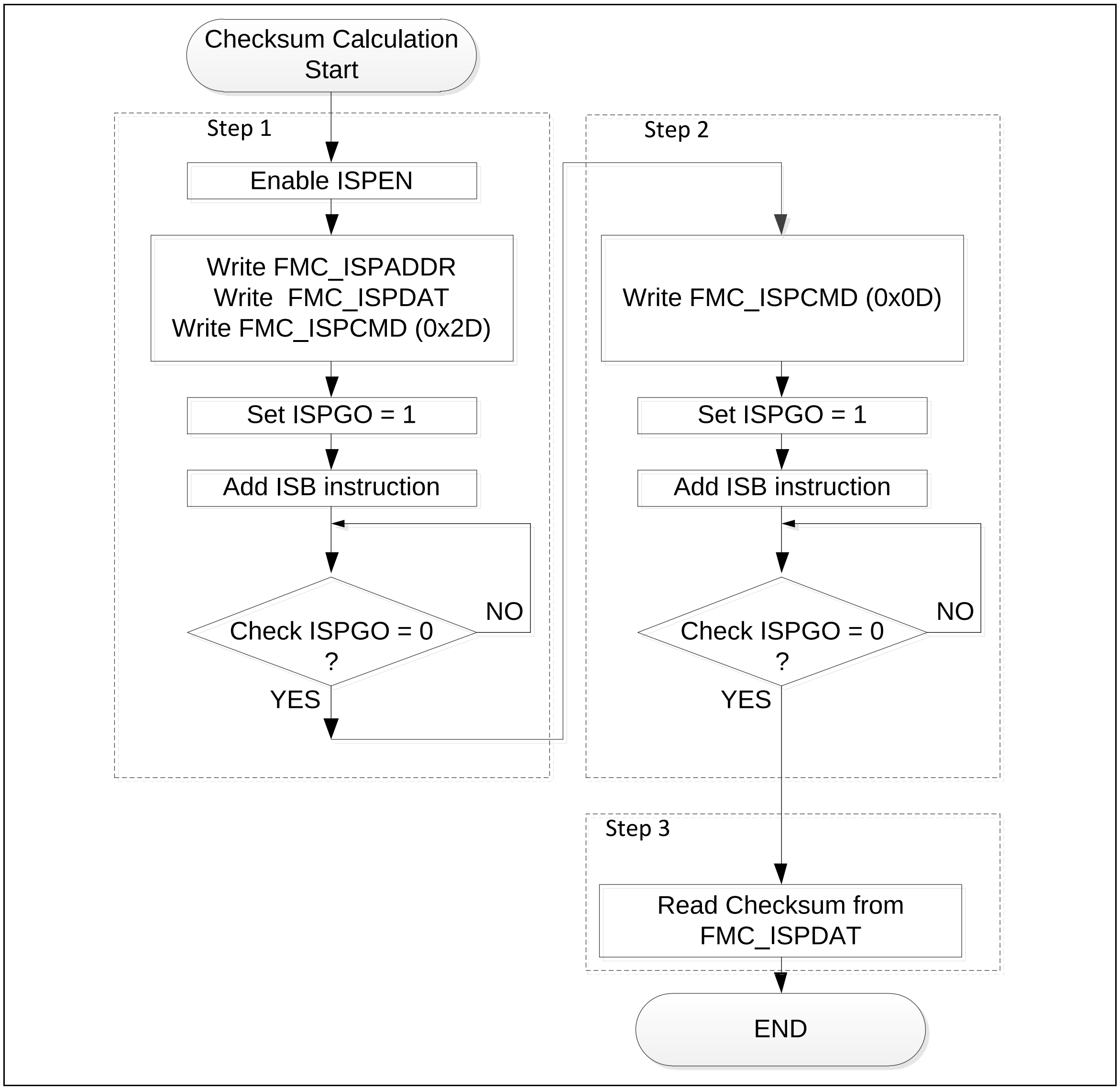


Figure 6.4-12 CRC-32 Checksum Calculation Flow

6.4.5 Flash Control Register Map

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

Register	Offset	R/W	Description	Reset Value
FMC Base Address: **FMC_BA = 0x5000_C000**				
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control Register	0x0000_000X
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000
FMC_DFBA	FMC_BA+0x14	R	Data Flash Base Address	0xXXXX_XXXX
FMC_FTCTL	FMC_BA+0x18	R/W	Flash Access Time Control Register	0x0000_0000
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP Status Register	0xX0X0_000X

6.4.6 Flash Control Register Description

ISP Control Register (FMC_ISPCTL)

Register	Offset	R/W	Description	Reset Value
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control Register	0x0000_000X

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**ISPFF**	**LDUEN**	**CFGUEN**	**APUEN**	**SPUEN**	**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: This bit needs to be cleared by writing 1 to it. (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) SPROM is erased/programmed if SPUEN is set to 0 (5) SPROM is programmed at SPROM secured mode. (6) Page Erase command at LOCK mode with ICE connection (7) Erase or Program command at brown-out detected (8) Destination address is illegal, such as over an available range. (9) Invalid ISP commands **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5]	**LDUEN**	**LDROM Update Enable (Write Protect)** LDROM update enable bit. 0 = LDROM cannot be updated. 1 = LDROM can be updated. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[4]	**CFGUEN**	**CONFIG Update Enable (Write Protect)** 0 = CONFIG cannot be updated. 1 = CONFIG can be updated. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

[3]	**APUEN**	**APROM Update Enable (Write Protect)** 0 = APROM cannot be updated when the chip runs in APROM. 1 = APROM can be updated when the chip runs in APROM. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**SPUEN**	**SPROM Update Enable (Write Protect)** 0 = SPROM cannot be updated. 1 = SPROM can be updated. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1]	**BS**	**Boot Select (Write Protect)** Set/clear this bit to select next booting from LDROM/APROM, respectively. This bit also functions as chip booting status flag, which can be used to check where chip booted from. This bit is initiated with the inversed value of CBS[1] (CONFIG0[7]) after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened 0 = Booting from APROM. 1 = Booting from LDROM. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**ISPEN**	**ISP Enable (Write Protect)** ISP function enable bit. Set this bit to enable ISP function. 0 = ISP function Disabled. 1 = ISP function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

ISP Address (FMC_ISPADDR)

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

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

Bits	Description	
[31:0]	**ISPADDR**	**ISP Address** The NUC121/125 series is equipped with embedded flash. ISPADDR[1:0] must be kept 00 for ISP 32-bit operation. and ISPADDR[8:0] must be kept all 0 for Vector Page Re-map Command For CRC32 Checksum Calculation command, this field is the flash starting address for checksum calculation, 512 bytes alignment is necessary for checksum calculation.

ISP Data Register (FMC_ISPDAT)

Register	Offset	R/W	Description	Reset Value
FMC_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. For Run CRC32 Checksum Calculation command, ISPDAT is the memory size (byte) and 512 bytes alignment. For ISP Read Checksum command, ISPDAT is the checksum result. If ISPDAT = 0x0000_0000, it means that (1) the checksum calculation is in progress, or (2) the memory range for checksum calculation is incorrect.

ISP Command (FMC_ISPCMD)

Register	Offset	R/W	Description	Reset Value
FMC_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	
[31:7]	**Reserved**	Reserved.
[6:0]	**CMD**	**ISP CMD** ISP command table is shown below: 0x00= FLASH Read. 0x04= Read Unique ID. 0x0B= Read Company ID. 0x0C= Read Device ID. 0x0D= Read CRC32 Checksum. 0x21= FLASH 32-bit Program. 0x22= FLASH Page Erase. 0x2D= Run CRC32 Checksum Calculation. 0x2E= Vector Remap. The other commands are invalid.

ISP Trigger Control Register (FMC_ISPTRG)

Register	Offset	R/W	Description	Reset Value
FMC_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:** This bit is write protected. Refer to the SYS_REGLCTL register.

Data Flash Base Address Register (FMC_DFBA)

Register	Offset	R/W	Description	Reset Value
FMC_DFBA	FMC_BA+0x14	R	Data Flash Base Address	0xXXXX_XXXX

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

Bits	Description	
[31:0]	**DFBA**	**Data Flash Base Address** This register indicates Data Flash start address. It is a read only register. The Data Flash is shared with APROM. the content of this register is loaded from CONFIG1 This register is valid when DFEN (CONFIG0[0]) =0 .

Flash Access Time Control Register (FMC_FTCTL)

Register	Offset	R/W	Description	Reset Value
FMC_FTCTL	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	**FOM**			**Reserved**			

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:4]	**FOM**	**Frequency Optimization Mode (Write Protect)** The NuMicro N121/125 series supports adjustable flash access timing to optimize the flash access cycles in different working frequency. 0x1 = Frequency ≤ 24MHz. Others = Frequency ≤ 50MHz. **Note 1:** x denotes the bit is don't care. **Note 2:** These bits are write protected. Refer to the SYS_REGLCTL register.
[6:0]	**Reserved**	Reserved.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

ISP Status Register (FMC_ISPSTS)

Register	Offset	R/W	Description	Reset Value
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP Status Register	0xX0X0_000X

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

Bits	Description	
[31]	**SCODE**	**Security Code Active Flag** This bit is set to 1 by hardware when detecting SPROM secured code is active at flash initialization, or software writes 1 to this bit to make secured code active; this bit is only cleared by SPROM page erase operation. 0 = SPROM secured code is inactive. 1 = SPROM secured code is active.
[30]	**Reserved**	Reserved.
[29:9]	**VECMAP**	**Vector Page Mapping Address (Read Only)** All access to 0x0000_0000~0x0000_01FF is remapped to the flash memory or SRAM address {VECMAP[20:0], 9'b0} ~ {VECMAP[20:0], 9'h1FF}, except SPROM. VECMAP [20:19] = 00 system vector address is mapped to flash memory. VECMAP [20:19] = 10 system vector address is mapped to SRAM memory. VECMAP [18:12] should be 0. **Note:** Vector mapping is page alignment (512 byte), and thus VECMAP starts from bit10.
[8:7]	**Reserved**	Reserved.
[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is the mirror of ISPFF (FMC_ISPCTL[6]), it needs to be cleared by writing 1 to FMC_ISPCTL[6] or FMC_ISPSTS[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) SPROM is erased/programmed if SPUEN is set to 0 (5) SPROM is programmed at SPROM secured mode. (6) Page Erase command at LOCK mode with ICE connection (7) Erase or Program command at brown-out detected (8) Destination address is illegal, such as over an available range. (9) Invalid ISP commands **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

[5:3]	**Reserved**	Reserved.
[2:1]	**CBS**	**Boot Selection of CONFIG (Read Only)** This bit is initiated with the CBS (CONFIG0[7:6]) after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened. 00 = LDROM with IAP mode. 01 = LDROM without IAP mode. 10 = APROM with IAP mode. 11 = APROM without IAP mode.
[0]	**ISPBUSY**	**ISP BUSY (Read Only)** 0 = ISP operation is finished. 1 = ISP operation is busy.

6.5 General Purpose I/O (GPIO)

6.5.1 Overview

The NUC121/125 series has up to 52 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 52 pins are arranged in 6 ports named as PA, PB, PC, PD, PE and PF.

PA has 6 pins on port (PA.10 ~ PA.15).

PB has 15 pins on port (PB.0 ~ PB.15, exclude PB.11).

PC has 12 pins on port (PC.0 ~ PC.13, exclude PC.6, PC.7).

PD has 10 pins on port (PD.0 ~ PD.11, exclude PD.6, PD.7).

PE has 3 pins on port (PE.0 ~ PE.2).

PF has 6 pins on port (PF.0 ~ PF.5).

Each of the 52 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, Push-pull output, Open-drain output or Quasi-bidirectional mode. After the chip is reset, the I/O mode of all pins are depending on CIOIN (CONFIG0[10]). 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-bidirectional mode
  - Push-Pull Output mode
  - Open-Drain Output mode
  - Input mode
- TTL/Schmitt trigger input selectable
- I/O pin can be configured as interrupt source with edge/level setting
- Supports High Slew Rate I/O mode
- Supports High Drive Strength mode for Port C
- Configurable default I/O mode of all pins after reset by CIOINI (CONFIG0[10]) setting
  - CIOIN = 0, all GPIO pins in input mode after chip reset
  - CIOIN = 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

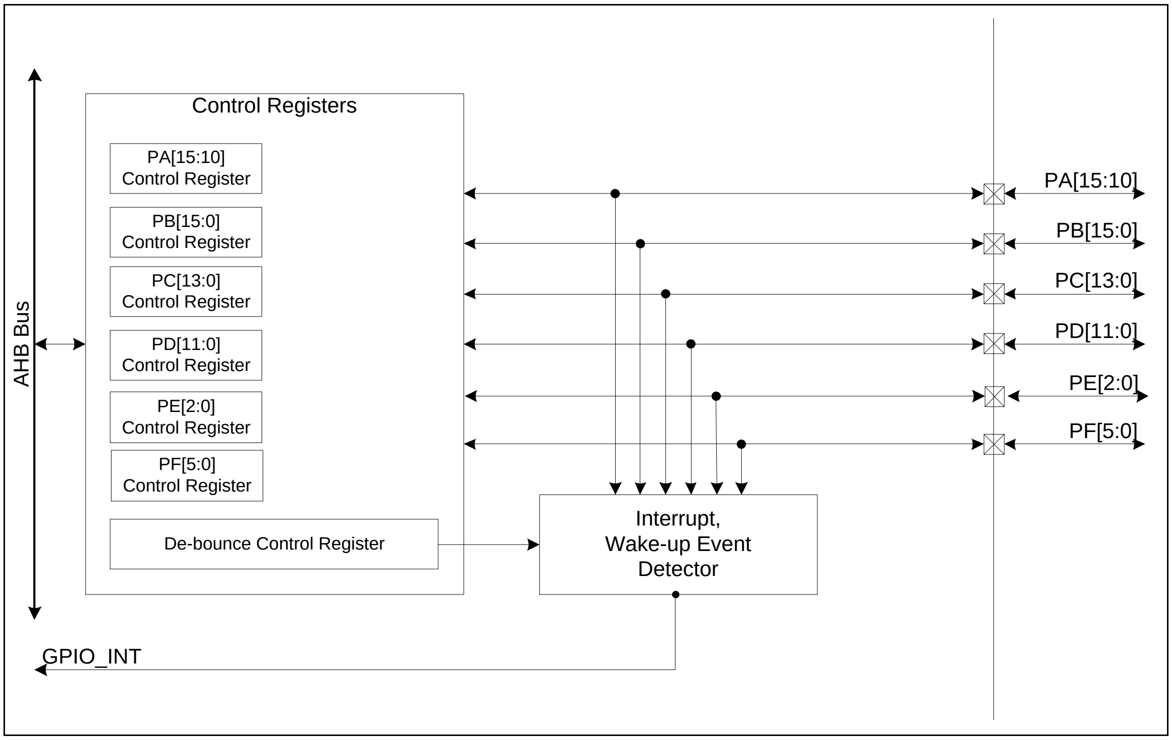
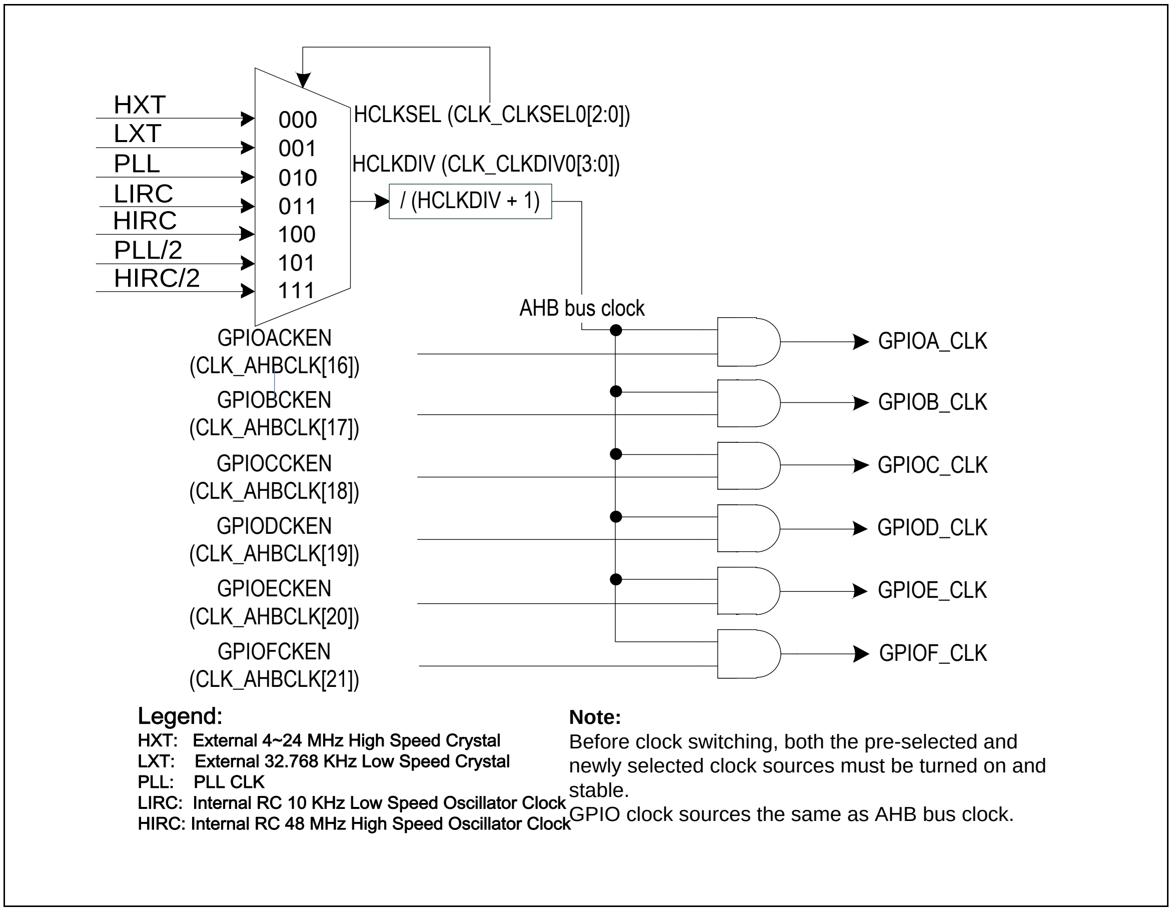

AHB Bus
Control Registers
PA[15:10] Control Register
PB[15:0] Control Register
PC[13:0] Control Register
PD[11:0] Control Register
PE[2:0] Control Register
PF[5:0] Control Register
De-bounce Control Register
Interrupt, Wake-up Event Detector
GPIO_INT
PA[15:10]
PB[15:0]
PC[13:0]
PD[11:0]
PE[2:0]
PF[5:0]

Figure 6.5-1 GPIO Controller Block Diagram

Note: The PB.11, PC.6/PC.7, PD.6/PD.7 pin is ignored.



Legend:

HXT:	External 4~24 MHz High Speed Crystal
LXT:	External 32.768 KHz Low Speed Crystal
PLL:	PLL CLK
LIRC:	Internal RC 10 KHz Low Speed Oscillator Clock
HIRC:	Internal RC 48 MHz High Speed Oscillator Clock

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

Figure 6.5-2 GPIO Clock Control Diagram

6.5.4 Basic Configuration

The basic configurations of GPIO are as follows:

- The GPIO pin functions are configured in SYS_PA_MFPH, SYS_PB_MFPL, SYS_PB_MFPH, SYS_PC_MFPL, SYS_PC_MFPH, SYS_PD_MFPL, SYS_PD_MFPH, SYS_PE_MFPL, and SYS_PF_MFPL registers.
- Enable GPIO clock on CLK_AHBCLK[21:16] register.

6.5.5 Functional Description

6.5.5.1 Input Mode

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

6.5.5.2 Push-pull Output Mode

Set MODEn (Px_MODE[2n+1:2n]) to 01 as Px.n pin is in Push-pull Output mode and the I/O pin supports digital output function with source/sink current capability. The bit value in the corresponding DOUT (Px_DOUT[n]) is driven on the pin.

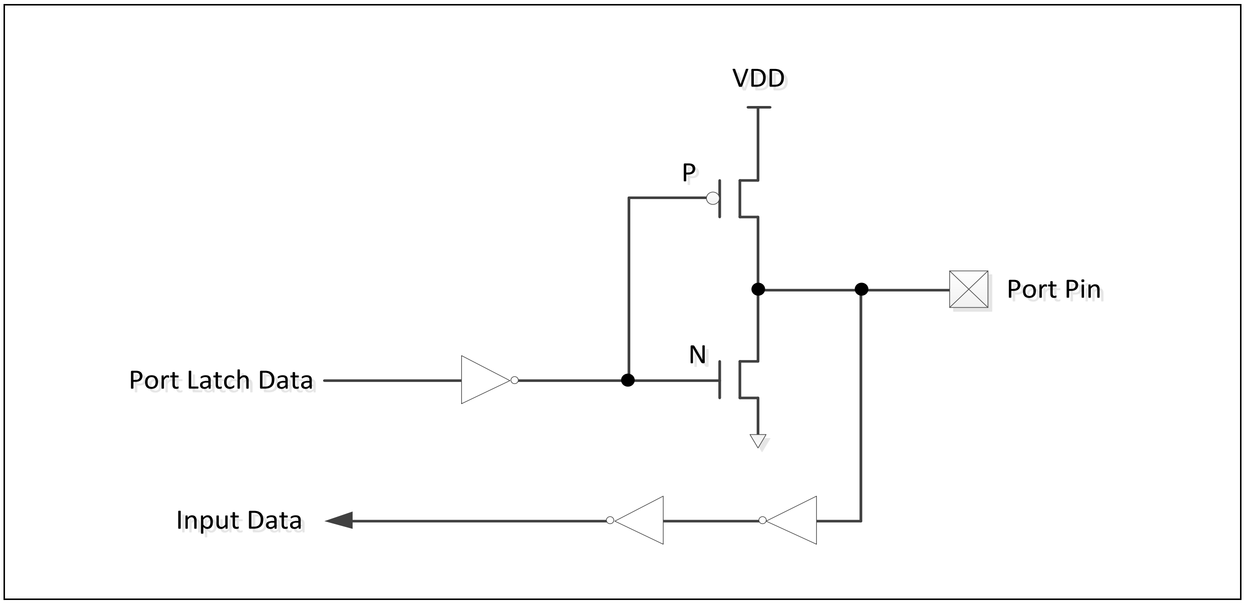


Figure 6.5-2 Push-Pull Output

6.5.5.3 Open-drain Output Mode

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

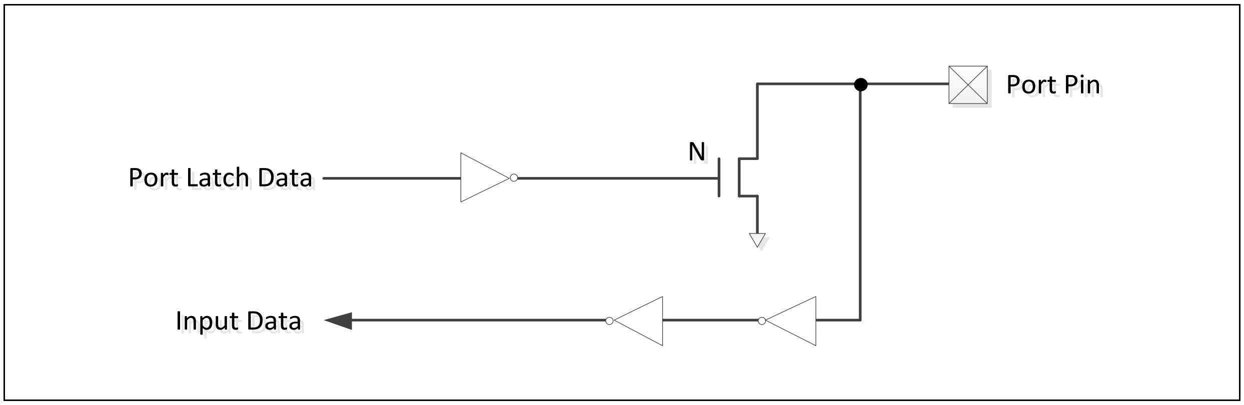


Figure 6.5-3 Open-Drain Output

6.5.5.4 Quasi-bidirectional Mode

Set MODEn (Px_MODE[2n+1:2n]) to 11 as the Px.n pin is in Quasi-bidirectional mode and the I/O pin supports digital output and input function at the same time but the source current is only up to hundreds uA. Before the digital input function is performed the corresponding DOUT (Px_DOUT[n]) bit must be set to 1. The quasi-bidirectional output is common on the 80C51 and most of its derivatives. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 0, the pin drive a low output on the pin. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 1, the pin will check the pin value. If pin value is high, no action takes. If pin state is low, the pin will drive strong high with 2 clock cycles on the pin and then disable the strong output drive. Meanwhile, the pin status is controlled by internal pull-up resistor. Note that the source current capability in quasi-bidirectional mode is only about 200 uA to 30 uA for V_{DD} is from 5.0 V to 2.5 V.

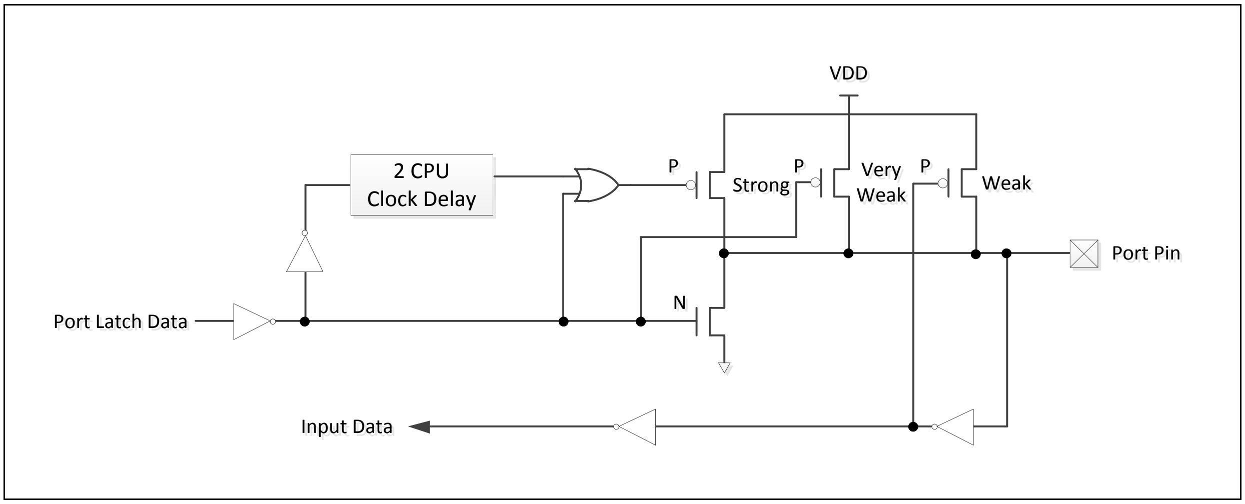


Figure 6.5-4 Quasi-Bidirectional I/O Mode

6.5.5.5 GPIO Interrupt and Wake-up Function

Each GPIO pin can be set as chip interrupt source by setting correlative RHIEN (Px_INTEN[n+16])/ FLIEN (Px_INTEN[n]) bit and TYPE (Px_INTTYPE[n]). Interrupt source flag register is INTSRC (Px_INTSRC[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 function is enable by setting DBEN (Px_DBEN[n]). The de-bounce clock source and sampling cycle period can be set through DBCLKSRC (GPIO_DBCTL[4]) and DBCLKSEL (GPIO_DBCTL[3:0])register.

ICLKON (GPIO_DBCTL[5]) is recommended setting 0 for reducing power consumption.

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.

Even the multifuction pin is selected to other IP, GPIO still can detect the related input pin to trigger interrupt flags or wake-up function.

6.5.5.6 GPIO Digital Input Disable Function

User can disable GPIO digital input path by setting DINOFF (Px_DINOFF[n]) to avoid input current leakage. When GPIO digital input path is disabled, the digital input pin value PIN (PxPIN[n]) is tied to low. By the way, the GPIO digital input path is force disabled by hardware and DINOFF control is useless when I/O function configure as ADC/ACMP/ext. XTL.

6.5.5.7 GPIO Data Ouput Write Mask Function

This function protect the corresponding DOUT (Px_DOUT[n]) bit. When the DATMSK (Px_DATMSK[n]) bit is set to 1, the corresponding DOUT (Px_DOUT[n]) bit is protected. If the write signal is masked, writing data to the protect bit is ignored. This function only protects the corresponding DOUT (Px_DOUT[n]) bit, and will not protect the corresponding PDIO (Pxn_PDIO[0]) bit.

6.5.5.8 GPIO Pads Related Function

In NUC121/125 series, GPIO pads support Input Schmitt Trigger Enable Function and High Slew Rate Control Function. Some pins supports High Drive Strength Control Function.

Schmitt trigger Input Schmitt Trigger Enable Function can be setting by SMTEN (Px_SMTEN[n]). High Slew Rate Control Function can increase the switching speed limit between 0 and 1. This function can be setting by HSREN (Px_SLEWCTL[n]).

PC.0 ~ PC.5 support High Drive Strength Control function. The pad with this function can drive more strength voltage than other pads.This function can be setting by HDRVEN (Px_DRVCTL[n]).

6.5.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
GPIO Base Address: **GPIO_BA = 0x5000_4000**				
PA_MODE	GPIO_BA+0x000	R/W	PA I/O Mode Control	0xXXX0_0000
PA_DINOFF	GPIO_BA+0x004	R/W	PA Digital Input Path Disable Control	0x0000_0000
PA_DOUT	GPIO_BA+0x008	R/W	PA Data Output Value	0x0000_FC00
PA_DATMSK	GPIO_BA+0x00C	R/W	PA Data Output Write Mask	0x0000_0000
PA_PIN	GPIO_BA+0x010	R	PA Pin Value	0x0000_XXFF
PA_DBEN	GPIO_BA+0x014	R/W	PA De-Bounce Enable Control	0x0000_0000
PA_INTTYPE	GPIO_BA+0x018	R/W	PA Interrupt Mode Trigger Type Control	0x0000_0000
PA_INTEN	GPIO_BA+0x01C	R/W	PA Interrupt Enable Control	0x0000_0000
PA_INTSRC	GPIO_BA+0x020	R/W	PA Interrupt Source Flag	0x0000_XX00
PA_SMTEN	GPIO_BA+0x024	R/W	PA Input Schmitt Trigger Enable	0x0000_0000
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA High Slew Rate Control	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0xXXXX_XXXX
PB_DINOFF	GPIO_BA+0x044	R/W	PB Digital Input Path Disable Control	0x0000_0000
PB_DOUT	GPIO_BA+0x048	R/W	PB Data Output Value	0x0000_F7FF
PB_DATMSK	GPIO_BA+0x04C	R/W	PB Data Output Write Mask	0x0000_0000
PB_PIN	GPIO_BA+0x050	R	PB Pin Value	0x0000_XXXX
PB_DBEN	GPIO_BA+0x054	R/W	PB De-Bounce Enable Control	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB Interrupt Mode Trigger Type Control	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB Interrupt Enable Control	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	PB Interrupt Source Flag	0x0000_XXXX
PB_SMTEN	GPIO_BA+0x064	R/W	PB Input Schmitt Trigger Enable	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB High Slew Rate Control	0x0000_0000
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0x0XXX_0XXX
PC_DINOFF	GPIO_BA+0x084	R/W	PC Digital Input Path Disable Control	0x0000_0000
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_3F3F
PC_DATMSK	GPIO_BA+0x08C	R/W	PC Data Output Write Mask	0x0000_0000

PC_PIN	GPIO_BA+0x090	R	PC Pin Value	0x0000_XXXX
PC_DBEN	GPIO_BA+0x094	R/W	PC De-Bounce Enable Control	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC Interrupt Mode Trigger Type Control	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC Interrupt Enable Control	0x0000_0000
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC Interrupt Source Flag	0x0000_XXXX
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC Input Schmitt Trigger Enable	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC High Slew Rate Control	0x0000_0000
PC_DRVCTL	GPIO_BA+0xAC	R/W	PC High Drive Strength Control	0x0000_0000
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0x00XX_0XXX
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD Digital Input Path Disable Control	0x0000_0000
PD_DOUT	GPIO_BA+0x0C8	R/W	PD Data Output Value	0x0000_0F3F
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD Data Output Write Mask	0x0000_0000
PD_PIN	GPIO_BA+0x0D0	R	PD Pin Value	0x0000_FXXX
PD_DBEN	GPIO_BA+0x0D4	R/W	PD De-Bounce Enable Control	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD Interrupt Mode Trigger Type Control	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD Interrupt Enable Control	0x0000_0000
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD Interrupt Source Flag	0x0000_0XXX
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD Input Schmitt Trigger Enable	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD High Slew Rate Control	0x0000_0000
PE_MODE	GPIO_BA+0x100	R/W	PE I/O Mode Control	0x0000_00XX
PE_DINOFF	GPIO_BA+0x104	R/W	PE Digital Input Path Disable Control	0x0000_0000
PE_DOUT	GPIO_BA+0x108	R/W	PE Data Output Value	0x0000_0007
PE_DATMSK	GPIO_BA+0x10C	R/W	PE Data Output Write Mask	0x0000_0000
PE_PIN	GPIO_BA+0x110	R	PE Pin Value	0x0000_FFFX
PE_DBEN	GPIO_BA+0x114	R/W	PE De-Bounce Enable Control	0x0000_0000
PE_INTTYPE	GPIO_BA+0x118	R/W	PE Interrupt Mode Trigger Type Control	0x0000_0000
PE_INTEN	GPIO_BA+0x11C	R/W	PE Interrupt Enable Control	0x0000_0000
PE_INTSRC	GPIO_BA+0x120	R/W	PE Interrupt Source Flag	0x0000_000X
PE_SMTEN	GPIO_BA+0x124	R/W	PE Input Schmitt Trigger Enable	0x0000_0000
PE_SLEWCTL	GPIO_BA+0x128	R/W	PE High Slew Rate Control	0x0000_0000
PF_MODE	GPIO_BA+0x140	R/W	PF I/O Mode Control	0x0000_0XXX

PF_DINOFF	GPIO_BA+0x144	R/W	PF Digital Input Path Disable Control	0x0000_0000
PF_DOUT	GPIO_BA+0x148	R/W	PF Data Output Value	0x0000_003F
PF_DATMSK	GPIO_BA+0x14C	R/W	PF Data Output Write Mask	0x0000_0000
PF_PIN	GPIO_BA+0x150	R	PF Pin Value	0x0000_00XX
PF_DBEN	GPIO_BA+0x154	R/W	PF De-Bounce Enable Control	0x0000_0000
PF_INTTYPE	GPIO_BA+0x158	R/W	PF Interrupt Mode Trigger Type Control	0x0000_0000
PF_INTEN	GPIO_BA+0x15C	R/W	PF Interrupt Enable Control	0x0000_0000
PF_INTSRC	GPIO_BA+0x160	R/W	PF Interrupt Source Flag	0x0000_00XX
PF_SMTEN	GPIO_BA+0x164	R/W	PF Input Schmitt Trigger Enable	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF High Slew Rate Control	0x0000_0000
GPIO_DBCTL	GPIO_BA+0x180	R/W	Interrupt De-bounce Control	0x0000_0020
PAn_PDIO **n=0,1..15**	GPIO_BA+0x200+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output	0x0000_000X
PBn_PDIO **n=0,1..15**	GPIO_BA+0x240+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output	0x0000_000X
PCn_PDIO **n=0,1..15**	GPIO_BA+0x280+(0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output	0x0000_000X
PDn_PDIO **n=0,1..15**	GPIO_BA+0x2C0+(0x04 * n)	R/W	GPIO PD.n Pin Data Input/Output	0x0000_000X
PEn_PDIO **n=0,1,2**	GPIO_BA+0x300+(0x04 * n)	R/W	GPIO PE.n Pin Data Input/Output	0x0000_000X
PFn_PDIO **n=0,1..7**	GPIO_BA+0x340+(0x04 * n)	R/W	GPIO PF.n Pin Data Input/Output	0x0000_000X

6.5.7 Register Description

Port A-F I/O Mode Control (Px_MODE)

Register	Offset	R/W	Description	Reset Value
PA_MODE	GPIO_BA+0x000	R/W	PA I/O Mode Control	0xXXX0_0000
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0xXXXX_XXXX
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0x0XXX_0XXX
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0x00XX_0XXX
PE_MODE	GPIO_BA+0x100	R/W	PE I/O Mode Control	0x0000_00XX
PF_MODE	GPIO_BA+0x140	R/W	PF I/O Mode Control	0x0000_0XXX

31	30	29	28	27	26	25	24
MODE15		MODE14		MODE13		MODE12	
23	22	21	20	19	18	17	16
MODE11		MODE10		MODE9		MODE8	
15	14	13	12	11	10	9	8
MODE7		MODE6		MODE5		MODE4	
7	6	5	4	3	2	1	0
MODE3		MODE2		MODE1		MODE0	

Bits	Description	
[2n+1:2n] n=0,1..15	**MODEn**	**Port A-f I/O Pin[n] Mode Control** Determine each I/O mode of Px.n pins. 00 = Px.n is in Input mode. 01 = Px.n is in Push-pull Output mode. 10 = Px.n is in Open-drain Output mode. 11 = Px.n is in Quasi-bidirectional mode. **Note1:** The initial value of this field is defined by CIOINI (CONFIG0 [10]). If CIOINI is set to 1, all existed pins will be quasi-bidirectional mode after chip powered on. For exampel, PA default value is 0xFFF00000. If CIOINI is set to 0, the default value is 0x0000_0000 and all pins will be input mode after chip powered on. **Note2:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note3:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

Port A-F Digital Input Path Disable Control (Px_DINOFF)

Register	Offset	R/W	Description	Reset Value
PA_DINOFF	GPIO_BA+0x004	R/W	PA Digital Input Path Disable Control	0x0000_0000
PB_DINOFF	GPIO_BA+0x044	R/W	PB Digital Input Path Disable Control	0x0000_0000
PC_DINOFF	GPIO_BA+0x084	R/W	PC Digital Input Path Disable Control	0x0000_0000
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD Digital Input Path Disable Control	0x0000_0000
PE_DINOFF	GPIO_BA+0x104	R/W	PE Digital Input Path Disable Control	0x0000_0000
PF_DINOFF	GPIO_BA+0x144	R/W	PF Digital Input Path Disable Control	0x0000_0000

31	30	29	28	27	26	25	24
DINOFF							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DINOFF							
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	**DINOFF**	**Port A-f Pin[n] Digital Input Path Disable Control** Each of these bits is used to control if the digital input path of corresponding Px.n pin is disabled. User can disable GPIO digital input path by setting DINOFF (Px_DINOFF[n]) to avoid input current leakage. When GPIO digital input path is disabled, the digital input pin value PIN (PxPIN[n]) is tied to low. By the way, the GPIO digital input path is force disabled by hardware and DINOFF control is useless when I/O function configure as ADC/ACMP/ext. XTL.. 0 = Px.n digital input path Enabled. 1 = Px.n digital input path Disabled (digital input tied to low). **Note1:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note2:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.
[15:0]	**Reserved**	Reserved.

Port A-F Data Output Value (Px_DOUT)

Register	Offset	R/W	Description	Reset Value
PA_DOUT	GPIO_BA+0x008	R/W	PA Data Output Value	0x0000_FC00
PB_DOUT	GPIO_BA+0x048	R/W	PB Data Output Value	0x0000_F7FF
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_3F3F
PD_DOUT	GPIO_BA+0x0C8	R/W	PD Data Output Value	0x0000_0F3F
PE_DOUT	GPIO_BA+0x108	R/W	PE Data Output Value	0x0000_0007
PF_DOUT	GPIO_BA+0x148	R/W	PF Data Output Value	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**
DOUT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DOUT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DOUT**	**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. **Note1:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note2:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

Port A-F Data Output Write Mask (Px_DATMSK)

Register	Offset	R/W	Description	Reset Value
PA_DATMSK	GPIO_BA+0x00C	R/W	PA Data Output Write Mask	0x0000_0000
PB_DATMSK	GPIO_BA+0x04C	R/W	PB Data Output Write Mask	0x0000_0000
PC_DATMSK	GPIO_BA+0x08C	R/W	PC Data Output Write Mask	0x0000_0000
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD Data Output Write Mask	0x0000_0000
PE_DATMSK	GPIO_BA+0x10C	R/W	PE Data Output Write Mask	0x0000_0000
PF_DATMSK	GPIO_BA+0x14C	R/W	PF 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
DATMSK							
7	6	5	4	3	2	1	0
DATMSK							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..15	**DATMSK**	**Port A-f Pin[n] Data Output Write Mask** These bits are used to protect the corresponding DOUT (Px_DOUT[n]) bit. When the DATMSK (Px_DATMSK[n]) bit is set to 1, the corresponding DOUT (Px_DOUT[n]) bit is protected. If the write signal is masked, writing data to the protect bit is ignored. 0 = Corresponding DOUT (Px_DOUT[n]) bit can be updated. 1 = Corresponding DOUT (Px_DOUT[n]) bit protected. **Note1:** This function only protects the corresponding DOUT (Px_DOUT[n]) bit, and will not protect the corresponding PDIO (Pxn_PDIO[0]) bit. **Note2:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note3:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

Port A-F Pin Value (Px_PIN)

Register	Offset	R/W	Description	Reset Value
PA_PIN	GPIO_BA+0x010	R	PA Pin Value	0x0000_XXFF
PB_PIN	GPIO_BA+0x050	R	PB Pin Value	0x0000_XXXX
PC_PIN	GPIO_BA+0x090	R	PC Pin Value	0x0000_XXXX
PD_PIN	GPIO_BA+0x0D0	R	PD Pin Value	0x0000_FXXX
PE_PIN	GPIO_BA+0x110	R	PE Pin Value	0x0000_FFFX
PF_PIN	GPIO_BA+0x150	R	PF Pin Value	0x0000_00XX

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**PIN**	**Port A-fPin[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. **Note1:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note2:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

Port A-F De-Bounce Enable Control (Px_DBEN)

Register	Offset	R/W	Description	Reset Value
PA_DBEN	GPIO_BA+0x014	R/W	PA De-Bounce Enable Control	0x0000_0000
PB_DBEN	GPIO_BA+0x054	R/W	PB De-Bounce Enable Control	0x0000_0000
PC_DBEN	GPIO_BA+0x094	R/W	PC De-Bounce Enable Control	0x0000_0000
PD_DBEN	GPIO_BA+0x0D4	R/W	PD De-Bounce Enable Control	0x0000_0000
PE_DBEN	GPIO_BA+0x114	R/W	PE De-Bounce Enable Control	0x0000_0000
PF_DBEN	GPIO_BA+0x154	R/W	PF De-Bounce Enable Control	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DBEN**	**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 (GPIO_DBCTL [4]), one de-bounce sample cycle period is controlled by DBCLKSEL (GPIO_DBCTL [3:0]). 0 = Px.n de-bounce function Disabled. 1 = Px.n de-bounce function Enabled. The de-bounce function is valid only for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored. **Note1:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note2:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

Port A-F Interrupt Type Control (Px_INTTYPE)

Register	Offset	R/W	Description	Reset Value
PA_INTTYPE	GPIO_BA+0x018	R/W	PA Interrupt Mode Trigger Type Control	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB Interrupt Mode Trigger Type Control	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC Interrupt Mode Trigger Type Control	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD Interrupt Mode Trigger Type Control	0x0000_0000
PE_INTTYPE	GPIO_BA+0x118	R/W	PE Interrupt Mode Trigger Type Control	0x0000_0000
PF_INTTYPE	GPIO_BA+0x158	R/W	PF Interrupt Mode Trigger Type 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**
TYPE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TYPE							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**TYPE**	**Port A-f Pin[n] Edge or Level Detection Interrupt Trigger Type Control** TYPE (Px_INTTYPE[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 RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[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. **Note1:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note2:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

Port A-F Interrupt Enable Control (Px_INTEN)

Register	Offset	R/W	Description	Reset Value
PA_INTEN	GPIO_BA+0x01C	R/W	PA Interrupt Enable Control	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB Interrupt Enable Control	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC Interrupt Enable Control	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD Interrupt Enable Control	0x0000_0000
PE_INTEN	GPIO_BA+0x11C	R/W	PE Interrupt Enable Control	0x0000_0000
PF_INTEN	GPIO_BA+0x15C	R/W	PF Interrupt Enable Control	0x0000_0000

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

Bits	Description	
[n+16] n=0,1..15	**RHIEN**	**Port A-f Pin[n] Rising Edge or High Level Interrupt Trigger Type Enable Bit** The RHIEN (Px_INTEN[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 RHIEN (Px_INTEN[n+16]) bit to 1 : If the interrupt is level trigger (TYPE (Px_INTTYPE[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 (TYPE (Px_INTTYPE[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. **Note1:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note2:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.
[n] n=0,1..15	**FLIEN**	**Port A-f Pin[n] Falling Edge or Low Level Interrupt Trigger Type Enable Bit** The FLIEN (Px_INTEN[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 FLIEN (Px_INTEN[n]) bit to 1 : If the interrupt is level trigger (TYPE (Px_INTTYPE[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(TYPE (Px_INTTYPE[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. **Note1:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note2:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

Port A-F Interrupt Source Flag (Px_INTSRC)

Register	Offset	R/W	Description	Reset Value
PA_INTSRC	GPIO_BA+0x020	R/W	PA Interrupt Source Flag	0x0000_XX00
PB_INTSRC	GPIO_BA+0x060	R/W	PB Interrupt Source Flag	0x0000_XXXX
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC Interrupt Source Flag	0x0000_XXXX
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD Interrupt Source Flag	0x0000_0XXX
PE_INTSRC	GPIO_BA+0x120	R/W	PE Interrupt Source Flag	0x0000_000X
PF_INTSRC	GPIO_BA+0x160	R/W	PF Interrupt Source Flag	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**
INTSRC							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTSRC							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**INTSRC**	**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. **Note1:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note2:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

Port A-F Input Schmitt Trigger Enable (Px_SMTEN)

Register	Offset	R/W	Description	Reset Value
PA_SMTEN	GPIO_BA+0x024	R/W	PA Input Schmitt Trigger Enable	0x0000_0000
PB_SMTEN	GPIO_BA+0x064	R/W	PB Input Schmitt Trigger Enable	0x0000_0000
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC Input Schmitt Trigger Enable	0x0000_0000
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD Input Schmitt Trigger Enable	0x0000_0000
PE_SMTEN	GPIO_BA+0x124	R/W	PE Input Schmitt Trigger Enable	0x0000_0000
PF_SMTEN	GPIO_BA+0x164	R/W	PF Input Schmitt Trigger 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**
SMTEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMTEN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**SMTEN**	**Port A-f Pin[n] Input Schmitt Trigger Enable Bit** 0 = Px.n input schmitt trigger function Disabled. 1 = Px.n input schmitt trigger function Enabled. **Note1:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note2:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

Port A-F High Slew Rate Control (Px_SLEWCTL)

Register	Offset	R/W	Description	Reset Value
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA High Slew Rate Control	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB High Slew Rate Control	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC High Slew Rate Control	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD High Slew Rate Control	0x0000_0000
PE_SLEWCTL	GPIO_BA+0x128	R/W	PE High Slew Rate Control	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF High Slew Rate 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**
HSREN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HSREN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**HSREN**	**Port A-f Pin[n] High Slew Rate Control** 0 = Px.n output with basic slew rate. 1 = Px.n output with higher slew rate. **Note1:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E. Max. n=5 for port F. **Note2:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

Port C High Drive Strength Control (Px_DRVCTL)

Register	Offset	R/W	Description	Reset Value
PC_DRVCTL	GPIO_BA+0xAC	R/W	PC High Drive Strength Control	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**HDRVEN**	**Port C Pin[n] Driving Strength Control** 0 = Px.n output with basic driving strength. 1 = Px.n output with high driving strength. **Note:** n=0,1..5 for port C.

Interrupt De-bounce Control (GPIO_DBCTL)

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

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**ICLKON**	**Interrupt Clock on Mode** 0 = Edge detection circuit is active only if I/O pin corresponding RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[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 (Pxn_PDIO)

Register	Offset	R/W	Description	Reset Value
PAn_PDIO **n=10,11..15**	GPIO_BA+0x200+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output	0x0000_000X
PBn_PDIO **n=0,1..15**	GPIO_BA+0x240+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output	0x0000_000X
PCn_PDIO **n=0,1..13**	GPIO_BA+0x280+(0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output	0x0000_000X
PDn_PDIO **n=0,1..11**	GPIO_BA+0x2C0+(0x04 * n)	R/W	GPIO PD.n Pin Data Input/Output	0x0000_000X
PEn_PDIO **n=0,1..13**	GPIO_BA+0x300+(0x04 * n)	R/W	GPIO PE.n Pin Data Input/Output	0x0000_000X
PFn_PDIO **n=0,1..5**	GPIO_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							**PDIO**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PDIO**	**GPIO Px.N Pin Data Input/Output** Writing this bit can control one GPIO pin output value. 0 = Corresponding GPIO pin set to low. 1 = Corresponding GPIO pin set to high. Read this register to get GPIO pin status. For example, writing PA0_PDIO will reflect the written value to bit DOUT (Px_DOUT[0]), reading PA0_PDIO will return the value of PIN (PA_PIN[0]). **Note1:** The writing operation will not be affected by register DATMSK (Px_DATMSK[n]). **Note2:** n = 10~15 for port A. Max. n=15 for port B. Max. n=13 for port C. Max. n=11 for port D. Max. n=2 for port E.

		Max. n=5 for port F. **Note3:** The PB.11, PC.6/PC.7, PD.6/D.7 pin is ignored.

6.6 PDMA Controller (PDMA)

6.6.1 Overview

The peripheral direct memory access (PDMA) controller is used to provide high-speed data transfer. The PDMA controller can transfer data from one address to another without CPU intervention. This has the benefit of reducing the workload of CPU and keeps CPU resources free for other applications. The PDMA controller has a total of 5 channels and each channel can perform transfer between memory and peripherals or between memory and memory. The PDMA supports time-out function for channel 0 and channel 1.

6.6.2 Features

- Supports 5 independently configurable channels
- Supports selectable 2 level of priority (fixed priority or round-robin priority)
- Supports transfer data width of 8, 16, and 32 bits
- Supports source and destination address increment size can be byte, half-word, word or no increment
- Supports software and SPI, UART, I^{2}S, I^{2}C, USCI, ADC, PWM and TIMER request
- Supports Scatter-Gather mode to perform sophisticated transfer through the use of the descriptor link list table
- Supports single and burst transfer type
- Supports time-out function for channel 0 and channel 1

6.6.3 Block Diagram

The PDMA controller block diagram is shown as follows.

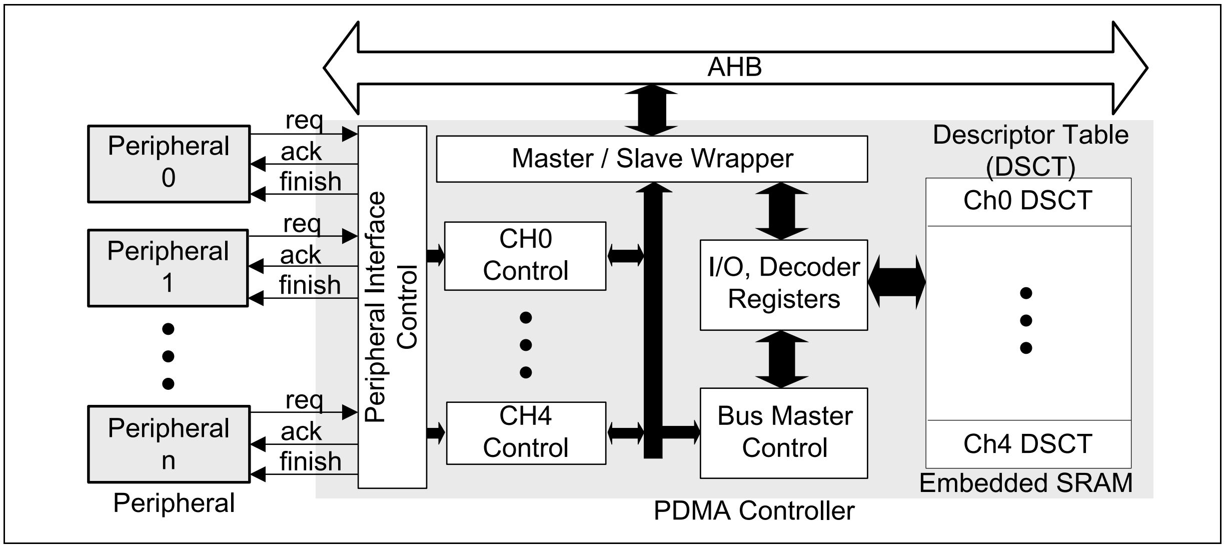


Figure 6.6-1 PDMA Controller Block Diagram

6.6.4 Basic Configuration

The peripheral direct memory access (PDMA) controller peripheral clock is enabled in

PDMACKEN(CLK_AHBCLK[1]).

6.6.5 Functional Description

The PDMA controller transfers data from one address to another without CPU intervention. The PDMA controller supports 5 independent channels and serves only one channel at one time, as the result, PDMA controller supports two level channel priorities: fixed and round-robin priority, PDMA controller serves channel in order from highest to lowest priority channel. The PDMA controller supports two operation modes: Basic mode and Scatter-gather mode. Basic mode is used to perform one descriptor table transfer. Scatter-gather mode has more entries for each PDMA channel, and thus the PDMA controller supports sophisticated transfer through the entries. The descriptor table entry data structure contains many transfer information including the transfer source address, transfer destination address, transfer count, burst size, transfer type and operation mode. Figure 6.6-2 shows the diagram of descriptor table (DSCT) data structure.

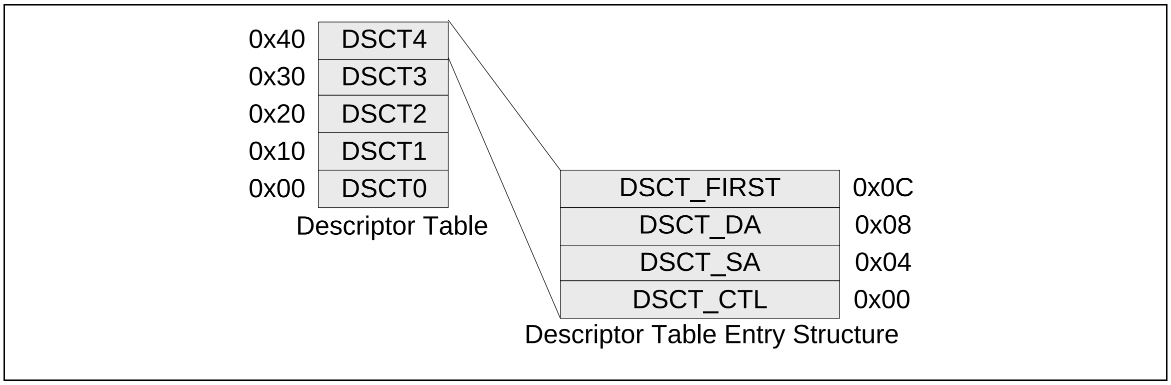


Figure 6.6-2 Descriptor Table Entry Structure

PDMA controller also supports single and burst transfer type and the request source can be from software or peripheral request, transfer between memory to memory using software request. A single transfer means that software or peripheral is ready to transfer one data (every data needs one request), and the burst transfer means that software or peripherals will transfer multiple data (multiple data only need one request).

PDMA time-out function is use to generate interrupt, while corresponding channel has been time-out.

6.6.5.1 Channel Priority

The PDMA controller supports two level channel priorities including fixed and round-robin priority. The fixed priority channel has higher priority than round-robin priority channel. If multiple channels are set as fixed or round-robin priority, the higher channel will have higher priority. A summary priority order is listed in Table 6.6-1. This table list all possible PDMA_PRISET and channel number combination and shows the their priority from hightest to lowest.

PDMA_PRISET	Channel Number	Priority Setting	Arbitration Priority In Descending Order
1	4	Channel4, Fixed Priority	Highest
1	3	Channel3, Fixed Priority	---
---	---	---	---
1	0	Channel0, Fixed Priority	---
0	4	Channel4, Round-Robin Priority	---
0	3	Channel3, Round-Robin Priority	---
---	---	---	---
0	0	Channel0, Round-Robin Priority	Lowest

Table 6.6-1 Channel Priority Table

6.6.5.2 PDMA Operation Mode

The PDMA controller supports two operation modes including Basic mode and Scatter-Gather mode.

Basic Mode

Basic mode is used to perform one descriptor table transfer mode that shown as Figure 6.6-3**.** This mode can be used to transfer data between memory and memory or peripherals and memory. PDMA controller operation mode can be set from OPMODE (PDMA_DSCTn_CTL[1:0], n denotes PDMA channel), PDMA will operate in idle mode by configured OPMODE bits to 1, the default setting is in idle state (OPMODE (PDMA_DSCTn_CTL[1:0]) = 00) and recommend user configure the descriptor table in idle state. If operation mode is not in idle state, user re-configure channel setting may make some operation error.

User must enable the transfer channel CHENn (PDMA_CHCTL[4:0]) and fill the transfer count TXCNT (PDMA_DSCTn_CTL[29:16]) register and select transfer width TXWIDTH (PDMA_DSCTn_CTL[13:12]), destination address increment size DAINC (PDMA_DSCTn_CTL[11:10]), source address increment size SAINC (PDMA_DSCTn_CTL[9:8]), burst size BURSIZE (PDMA_DSCTn_CTL[6:4]) and transfer type TXTYPE (PDMA_DSCTn_CTL[2]), then the PDMA controller will perform transfer operation in transfer state after receiving request signal. Finishing this task will generate an interrupt to CPU if each PDMA interrupt bit INTENn (PDMA_INTEN[4:0]) is enabled and the operation mode will be updated to idle state automatically as shown inFigure 6.6-4.

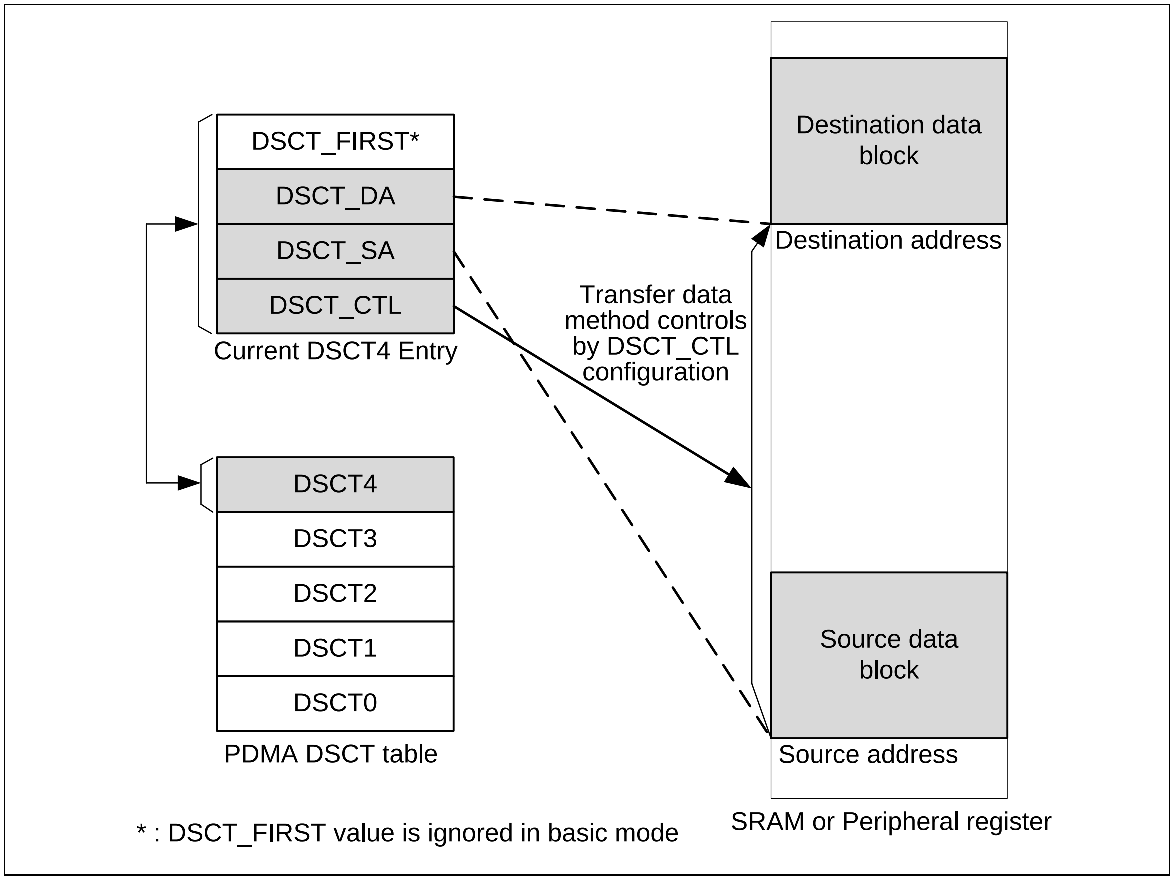


Figure 6.6-3 Descriptor Table Operation in Basic Mode

If software configures the operation mode to idle state, the PDMA controller will not perform any transfer and then clear this operation request. Finishing this task will also generate an interrupt to CPU if each PDMA interrupt bit is enabled.

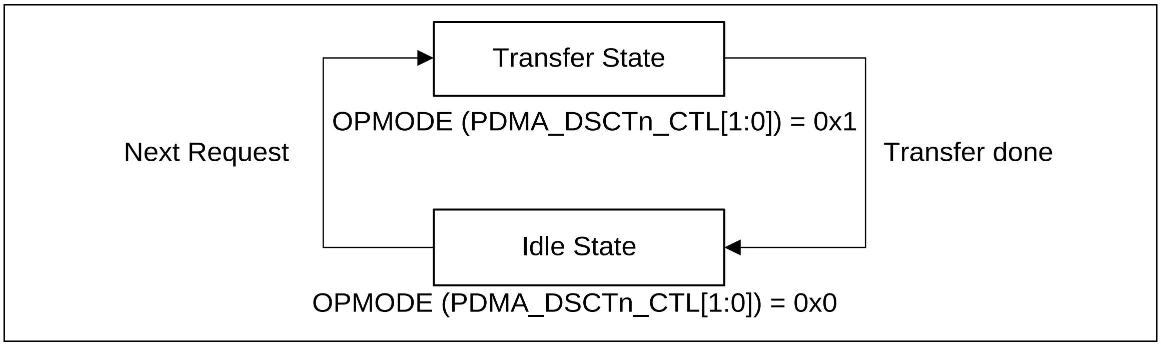

Transfer State
OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x1
Next Request
Transfer done
Idle State
OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x0

Figure 6.6-4 Basic Mode Finite State Machine

Scatter-Gather Mode

Scatter-Gather mode is a complex mode and can perform sophisticated transfer through the use of the description link list table as shown in Figure 6.6-5.

In Scatter-Gather mode, the first table entry is just used for jumping to the next table entry, the first task will not perform any operation transfer. Finishing each task will generate an interrupt to CPU if corresponding PDMA interrupt bit is enabled and TBINTDIS (PDMA_DSCTn_CTL[7]) bit is “0” (when finishing task and TBINTDIS bit is “0”, corresponding TDIFn (PDMA_TDSTS[4:0]) flag will be asserted and if this bit is “1” TDIFn will not be active)..

If channel 4 has been triggered as Figure 6.6-5, and the operation mode is in Scatter-Gather mode (OPMODE (PDMA_DSCT4_CTL[1:0]) = 0x2), the hardware will load the real PDMA information task from the address generated by adding PDMA_DSCT4_FIRST (link offset) and PDMA_SCATBA (base address) registers. As figure, base address is 0x2000_0000 (only MSB 16bits valid in PDMA_SCATBA), the first link offset is 0x0000_0100 (only LSB 16bits in PDMA_DSCTn_FIRST), then current DSCT entry PDMA_CURSCAT4 will update to address 0x2000_0100 and copy DSCT_CTL, DSCT_SA,DSCT_DA and DSCT_NEXT from SRAM to PDMA_DSCT4_CTL, PDMA_DSCT4_SA,PDMA_DSCT4_DA and PDMA_DSCT4_FIRST register for execution the task of this table.

Furthermore, the low half-word of next DSCT_NEXT in SRAM will load to the high half-word of PDMA_DSCT4_FIRST register. For example, if current table is the first DSCT4 entry, PDMA_DSCT4_FIRST will become 0x0200_0100, high half-word is next table offset.

After the task is finished, PDMA will update the OPMODE of current DSCT_CTL in SRAM to idle mode, then load the information from next table address 0x2000_0200, which is calculated by adding base address 0x2000_0000 and next offset 0x0200 from SRAM.

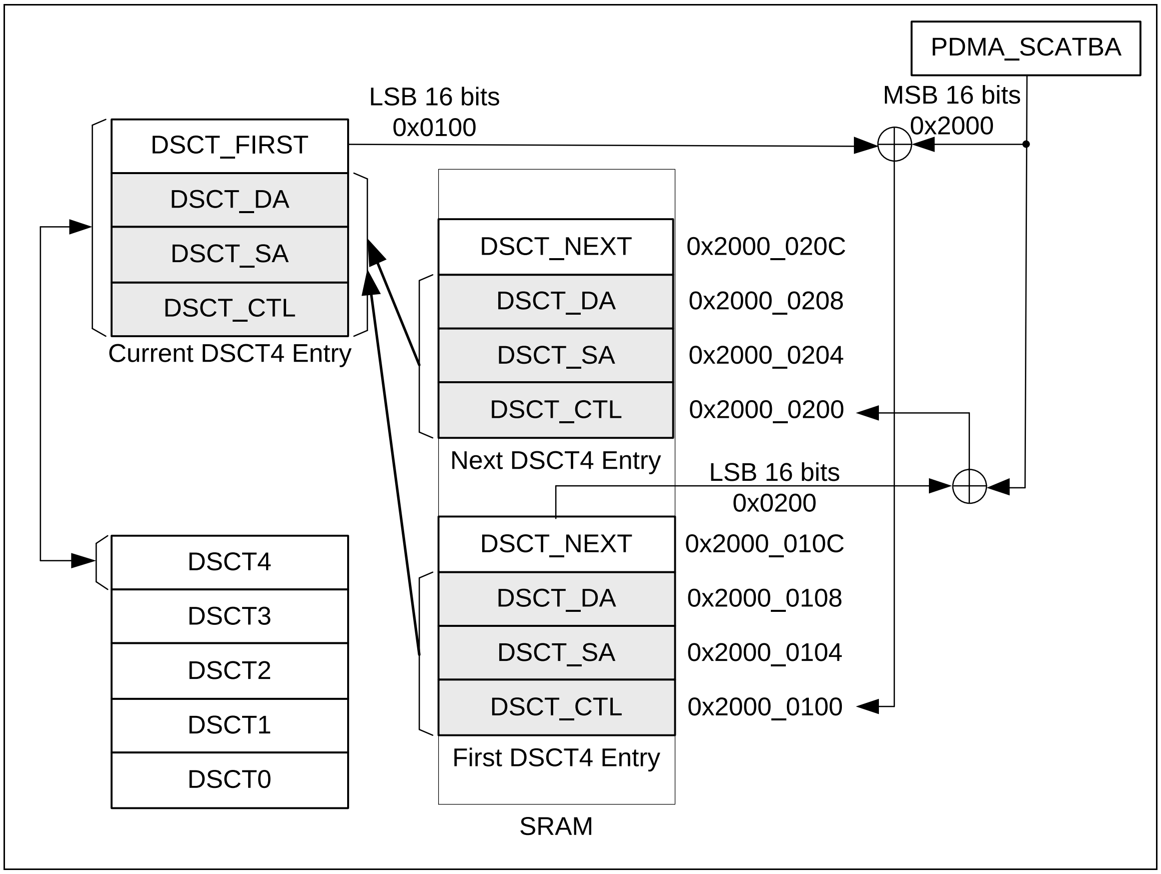


Figure 6.6-5 Descriptor Table Link List Structure

The above link list table operation is DSCT state in Scatter-Gather Mode as shown in Figure 6.6-6. When loading the information is finished, it will go to transfer state and start transfer by this information automatically. However, if the next PDMA information is also in the Scatter-Gather mode, the hardware will catch the next PDMA information block when the current task is finished. The Scatter-Gather mode stops until the PDMA controller operation mode switch to basic mode and transfer once or directly switch to idle state.

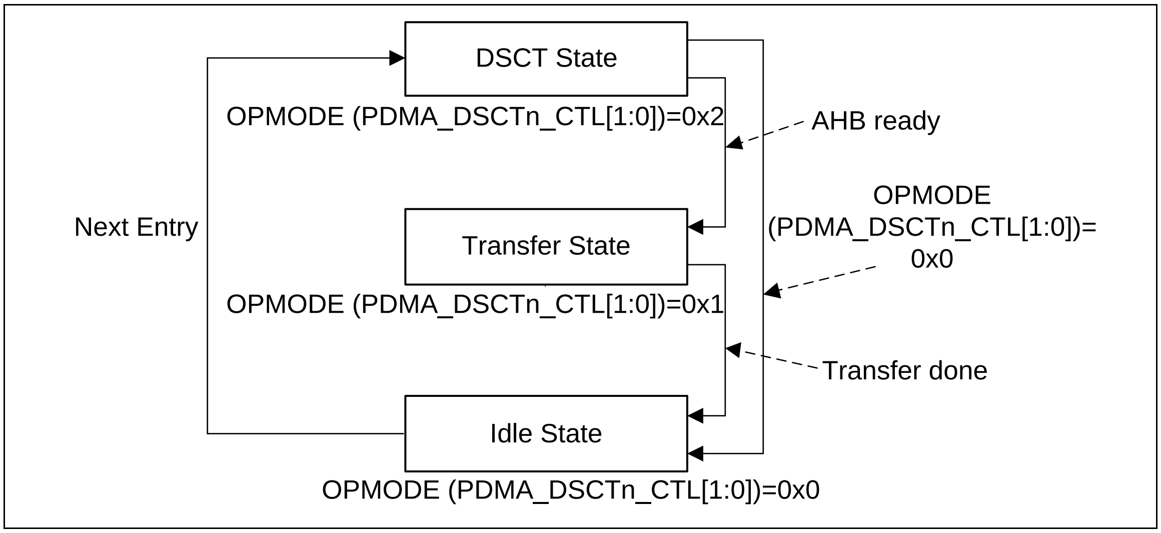


Figure 6.6-6 Scatter-Gather Mode Finite State Machine

Through Scatter-Gather mode, user can perform peripheral wrapper-around, multiple PDMA tasks or can be used for data transfer between varied locations in system memory instead of a set of contiguous locations.

The Ping-Pong audio buffer for I^{2}S can be implemented by the wrapper-around two link list tables. The audio buffer can be divide to two parts that assign to two link list tables for PDMA transfer, then PDMA controller can loop around two buffer for transferring audio data to I^{2}S TX buffer.

One PDMA task is controlled by one description link list table, so multiple description link list tables can make PDMA controller to perform multiple transfer tasks which the source/destination address can be different for each task, and it just uses one PDMA channel. Besides, the transfer block of multiple PDMA tasks can be scatter blocks in system memory.

6.6.5.3 Transfer Type

The PDMA controller supports two transfer types: single transfer type and burst transfer type, They are configured by setting TXTYPE (PDMA_DSCTn_CTL[2]).

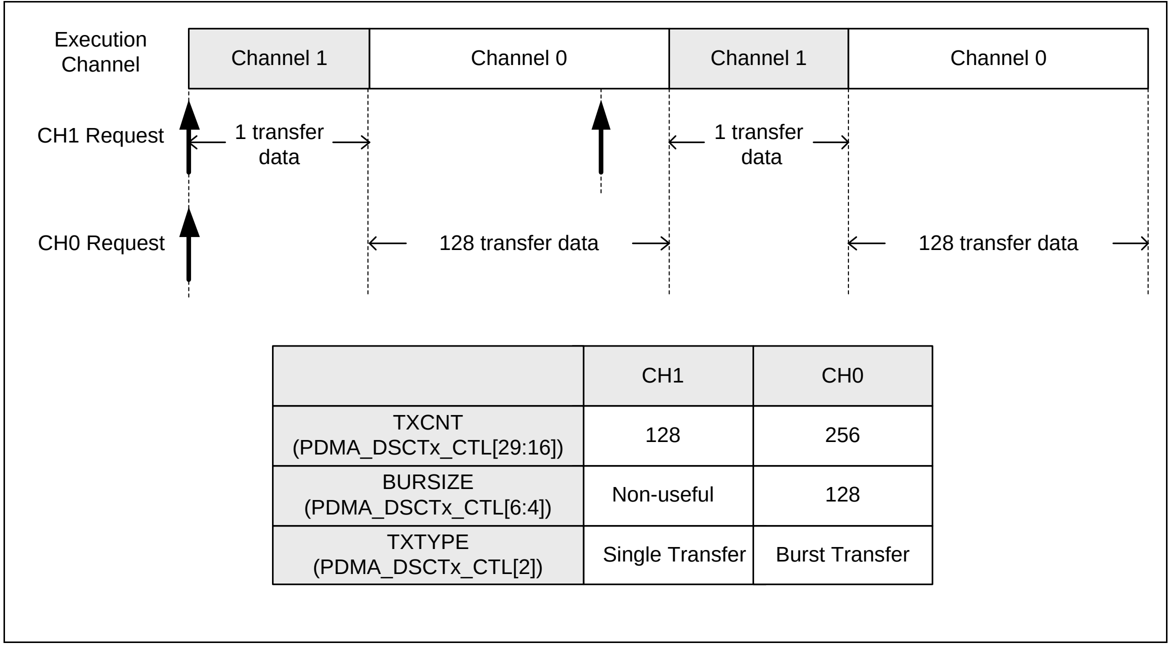
When PDMA controller operated in single transfer type, each transfer data needs one request signal which is coming from peripheral for one transfer. After data is transferred, TXCNT (PDMA_DSCTn_CTL[29:16]) will decrease 1. Transfer task will be finished after received TXCNT request signals, then TXCNT (PDMA_DSCTn_CTL[29:16]) bits decrease to 0. In this mode, the BURSIZE (PDMA_DSCTn_CTL[6:4]) is not useful to control the transfer size, because of the settings of BURSIZE bits will be ignored by hardware..

For the burst transfer type, PDMA controller transfers TXCNT (PDMA_DSCTn_CTL[29:16]) of data and need only one request signal from software corresponding channel request SWREQn (PDMA_SWREQ[4:0]). After transferred BURSIZE (PDMA_DSCTn_CTL[6:4]) of data, TXCNT (PDMA_DSCTn_CTL[29:16]) will decrease BURSIZE number. Transfer task will done until the transfer count TXCNT (PDMA_DSCTn_CTL[29:16]) decrease to 0. Note that burst transfer type can only be used for PDMA controller to do burst transfer between memory and memory. User must use single request type for memory-to-peripheral and peripheral-to-memory transfers.

Figure 6.6-7 shows an example about single and burst transfer type in basic mode.

In this example, channel 1 uses single transfer type and TXCNT (PDMA_DSCTn_CTL[29:16]) = 127, TXWIDTH(PDMA_DSCTn_CTL[13:12]) = 0(1 byte). Channel 0 uses burst transfer type, BURSIZE (PDMA_DSCTn_CTL[6:4]) = 0(128 transfers), TXCNT (PDMA_DSCTn_CTL[29:16]) = 255 and TXWIDTH(PDMA_DSCTn_CTL[13:12]) = 2(1 word). The operation sequence is described below:

1. Channel 0 and channel 1 get the trigger signal at the same time.
2. Channel 1 has higher priority than channel 0 by default; the PDMA controller will load the channel 1 descriptor table first and executing. But channel 1 is single transfer type and transfer width is one byte, and thus PDMA controller will only transfer 1 byte data.
3. Then, PDMA controller turns to the channel 0 and loads channel 0’s descriptor table. The channel 0 is burst transfer type, the burst size selected to 128 and transfer width is one word. Therefore, PDMA controller will transfer 128 words data.
4. When channel 0 transfers 128 words data, channel 1 gets another request signal, then after channel 0 finishes 128 words transferred, the PDMA controller will turn to channel 1 and transfer next 1 byte data.
5. After channel 1 transferred second 1 byte data, PDMA controller switches to low priority channel 0 to continuous next 128 words data transfer. If no channel 1 request receives, PDMA will start to transfer next 128 words data for channel 0.
6. PDMA controller will complete transfer when channel 0 finishes data transfer 256 words that received one request, and channel 1 finishes transferring 128 bytes after received 128 requests.



	CH1	CH0
TXCNT (PDMA_DSCTx_CTL[29:16])	128	256
BURSIZE (PDMA_DSCTx_CTL[6:4])	Non-useful	128
TXTYPE (PDMA_DSCTx_CTL[2])	Single Transfer	Burst Transfer

Figure 6.6-7 Example of Single Transfer Type and Burst Transfer Type in Basic Mode

6.6.5.4 Channel Time-out

Only channel 0 and channel 1 support time-out function. When the transfer channel is enabled and selected to the peripheral, corresponding channel time-out TOUTENn (PDMA_TOUTEN[n], n=0,1) is enabled, then channel’s corresponding time-out counter will start counting up from 0 while the channel has received trigger signal from the peripheral.

The time-out counter is based on output of HCLK prescaler, wich is setting by corresponding channel’s TOUTPSCn (PDMA_TOUTPSC[2+4n:4n], n=0,1). If time-out counter counts up from 0 to corresponding channel’s TOCn (PDMA_TOC0_1[16(n+1)-1):16n], n=0,1), the PDMA controller will generate interrupt signal when corresponding TOUTIENn (PDMA_TOUTIEN[n], n=0,1) is enabled. When time-out occurred, corresponding channel’s REQTOFn (PDMA_INTSTS[n+8], n=0,1) will be set to indicate channel time-out is happened.

Time-out counter reset to 0 while counter count to TOCn (PDMA_TOC0_1[16(n+1)-1):16n], n=0,1), received trigger signal or chip enter power-down mode.

Figure 6.6-8 shows an example about time-out counter operation. The operation sequence is described below:

1. The channel 0 time-out counter is not counting when time-out function is enabled by set TOUTEN0(PDMA_TOUTEN[0]) bit to 1.
2. Time-out counter is start counting from 0 to the value of TOC0(PDMA_TOC0_1[15:0]) bits when received first peripheral request.
3. Time-out counter is reset to 0 by received second peripheral request.
4. Channel 0 request time-out flag(REQTOF0(PDMA_INTSTS[8])) is set to high when time-out counter counts to 5. The counter will keep counting from 0 to 5, and user can clear REQTOF0 flag then polling REQTOF0 flag to check next time-out occurred.
5. Time-out counter is reset to 0 when time-out function is disabled.

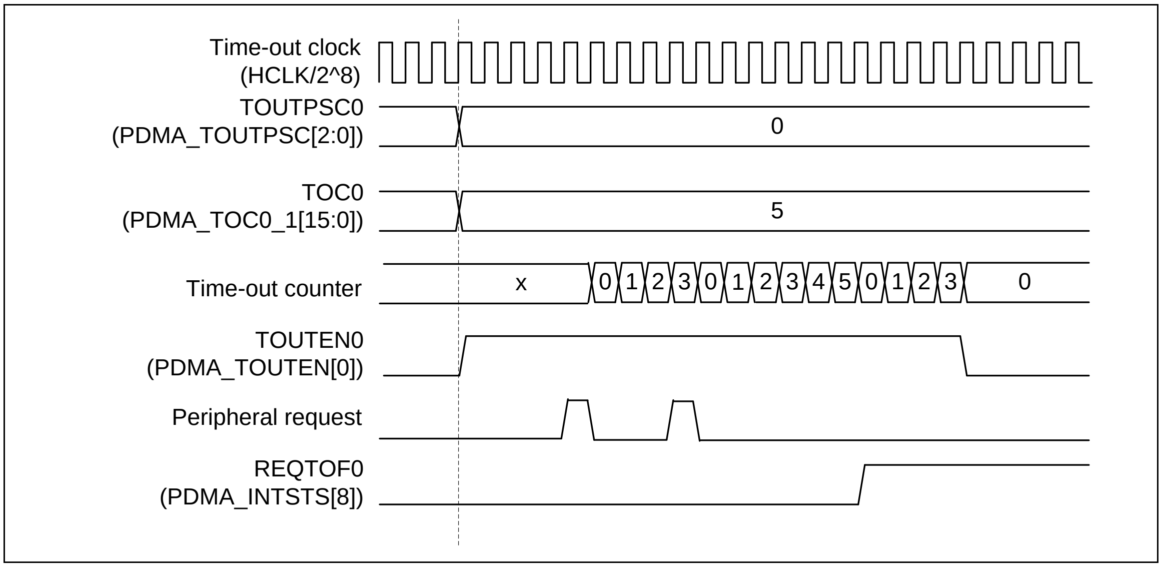


Figure 6.6-8 Example of PDMA Channel 0 Time-out Counter Operation

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 Base Address: **PDMA_BA = 0x5000_8000**				
PDMA_DSCT0_CTL	PDMA_BA + 0x000	R/W	Descriptor Table Control Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT0_SA	PDMA_BA + 0x004	R/W	Source Address Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT0_DA	PDMA_BA + 0x008	R/W	Destination Address Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT0_FIRST	PDMA_BA + 0x00C	R/W	First Scatter-Gather Descriptor Table Offset of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT1_CTL	PDMA_BA + 0x010	R/W	Descriptor Table Control Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT1_SA	PDMA_BA + 0x014	R/W	Source Address Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT1_DA	PDMA_BA + 0x018	R/W	Destination Address Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT1_FIRST	PDMA_BA + 0x01C	R/W	First Scatter-Gather Descriptor Table Offset of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT2_CTL	PDMA_BA + 0x020	R/W	Descriptor Table Control Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT2_SA	PDMA_BA + 0x024	R/W	Source Address Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT2_DA	PDMA_BA + 0x028	R/W	Destination Address Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT2_FIRST	PDMA_BA + 0x02C	R/W	First Scatter-Gather Descriptor Table Offset of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT3_CTL	PDMA_BA + 0x030	R/W	Descriptor Table Control Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT3_SA	PDMA_BA + 0x034	R/W	Source Address Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT3_DA	PDMA_BA + 0x038	R/W	Destination Address Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT3_FIRST	PDMA_BA + 0x03C	R/W	First Scatter-Gather Descriptor Table Offset of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT4_CTL	PDMA_BA + 0x040	R/W	Descriptor Table Control Register of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT4_SA	PDMA_BA + 0x044	R/W	Source Address Register of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT4_DA	PDMA_BA + 0x048	R/W	Destination Address Register of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT4_FIRST	PDMA_BA + 0x04C	R/W	First Scatter-Gather Descriptor Table Offset of PDMA Channel 4	0xXXXX_XXXX
PDMA_CURSCAT0	PDMA_BA + 0x050	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 0	0xXXXX_XXXX
PDMA_CURSCAT1	PDMA_BA + 0x054	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 1	0xXXXX_XXXX
PDMA_CURSCAT2	PDMA_BA + 0x058	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 2	0xXXXX_XXXX
PDMA_CURSCAT3	PDMA_BA + 0x05C	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 3	0xXXXX_XXXX
PDMA_CURSCAT4	PDMA_BA + 0x060	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 4	0xXXXX_XXXX

PDMA_CHCTL	PDMA_BA + 0x400	R/W	PDMA Channel Control Register	0x0000_0000
PDMA_PAUSE	PDMA_BA + 0x404	W	PDMA Transfer Pause Control Register	0x0000_0000
PDMA_SWREQ	PDMA_BA + 0x408	W	PDMA Software Request Register	0x0000_0000
PDMA_TRGSTS	PDMA_BA + 0x40C	R	PDMA Channel Request Status Register	0x0000_0000
PDMA_PRISET	PDMA_BA + 0x410	R/W	PDMA Fixed Priority Setting Register	0x0000_0000
PDMA_PRICLR	PDMA_BA + 0x414	W	PDMA Fixed Priority Clear Register	0x0000_0000
PDMA_INTEN	PDMA_BA + 0x418	R/W	PDMA Interrupt Enable Register	0x0000_0000
PDMA_INTSTS	PDMA_BA + 0x41C	R/W	PDMA Interrupt Status Register	0x0000_0000
PDMA_ABTSTS	PDMA_BA + 0x420	R/W	PDMA Channel Read/Write Target Abort Flag Register	0x0000_0000
PDMA_TDSTS	PDMA_BA + 0x424	R/W	PDMA Channel Transfer Done Flag Register	0x0000_0000
PDMA_SCATSTS	PDMA_BA + 0x428	R/W	PDMA Scatter-Gather Table Empty Status Register	0x0000_0000
PDMA_TACTSTS	PDMA_BA + 0x42C	R	PDMA Transfer Active Flag Register	0x0000_0000
PDMA_TOUTPSC	PDMA_BA + 0x430	R/W	PDMA Time-out Prescaler Register	0x0000_0000
PDMA_TOUTEN	PDMA_BA + 0x434	R/W	PDMA Time-out Enable Register	0x0000_0000
PDMA_TOUTIEN	PDMA_BA + 0x438	R/W	PDMA Time-out Interrupt Enable Register	0x0000_0000
PDMA_SCATBA	PDMA_BA + 0x43C	R/W	PDMA Scatter-Gather Descriptor Table Base Address Register	0x2000_0000
PDMA_TOC0_1	PDMA_BA + 0x440	R/W	PDMA Channel 0 and Channel 1 Time-out Counter Register	0x0000_0000
PDMA_RESET	PDMA_BA + 0x460	R/W	PDMA channel Reset Control Register	0x0000_0000
PDMA_REQSEL0_3	PDMA_BA + 0x480	R/W	PDMA Channel 0 to Channel 3 Request Source Select Register	0x0000_0000
PDMA_REQSEL4	PDMA_BA + 0x484	R/W	PDMA Channel 4 Request Source Select Register	0x0000_0000

6.6.7 Register Description

PDMA Descriptor Table Control Register (PDMA_DSCTn_CTL)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_CTL	PDMA_BA + 0x000	R/W	Descriptor Table Control Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT1_CTL	PDMA_BA + 0x010	R/W	Descriptor Table Control Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT2_CTL	PDMA_BA + 0x020	R/W	Descriptor Table Control Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT3_CTL	PDMA_BA + 0x030	R/W	Descriptor Table Control Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT4_CTL	PDMA_BA + 0x040	R/W	Descriptor Table Control Register of PDMA Channel 4	0xXXXX_XXXX

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

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:16]	**TXCNT**	**Transfer Count** The TXCNT represents the required number of PDMA transfer, the real transfer count is (TXCNT + 1); The maximum transfer count is 16384, every transfer may be byte, half-word or word that is dependent on TXWIDTH field. **Note:** When PDMA finish each transfer data, this field will be decrease immediately.
[15:14]	**Reserved**	Reserved.
[13:12]	**TXWIDTH**	**Transfer Width Selection** This field is used for transfer width. 00 = One byte (8 bit) is transferred for every operation. 01= One half-word (16 bit) is transferred for every operation. 10 = One word (32-bit) is transferred for every operation. 11 = Reserved. **Note:** The PDMA transfer source address (PDMA_DSCT_SA) and PDMA transfer destination address (PDMA_DSCT_DA) should be alignment under the TXWIDTH selection
[11:10]	**DAINC**	**Destination Address Increment** This field is used to set the destination address increment size.

Bits	Description	
		11 = No increment (fixed address). Others = Increment and size is depended on TXWIDTH selection.
[9:8]	**SAINC**	**Source Address Increment** This field is used to set the source address increment size. 11 = No increment (fixed address). Others = Increment and size is depended on TXWIDTH selection.
[7]	**TBINTDIS**	**Table Interrupt Disable Bit** This field can be used to decide whether to enable table interrupt or not. If the TBINTDIS bit is set when PDMA controller finishes transfer task, it will not generates interrupt. 0 = Table interrupt Enabled. 1 = Table interrupt Disabled. **Note:** If this bit set to '1', the TEMPTYF will not be set.
[6:4]	**BURSIZE**	**Burst Size** This field is used for peripheral to determine the burst size or used for determine the re-arbitration size. 000 = 128 Transfers. 001 = 64 Transfers. 010 = 32 Transfers. 011 = 16 Transfers. 100 = 8 Transfers. 101 = 4 Transfers. 110 = 2 Transfers. 111 = 1 Transfers. **Note:** This field is only useful in burst transfer type.
[3]	**Reserved**	Reserved.
[2]	**TXTYPE**	**Transfer Type** 0 = Burst transfer type. 1 = Single transfer type.
[1:0]	**OPMODE**	**PDMA Operation Mode Selection** 00 = Idle state: Channel is stopped or this table is complete, when PDMA finish channel table task, OPMODE will be cleared to idle state automatically. 01 = Basic mode: The descriptor table only has one task. When this task is finished, the PDMA_INTSTS[n] will be asserted. 10 = Scatter-Gather mode: When operating in this mode, user must give the first descriptor table address in PDMA_DSCT_FIRST register; PDMA controller will ignore this task, then load the next task to execute. 11 = Reserved. **Note:** Before filling transfer task in the Descriptor Table, user must check if the descriptor table is complete.

Note: The n in the descriptor table represents the PDMA channel.

Source Address Register (PDMA_DSCTn_SA)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_SA	PDMA_BA + 0x004	R/W	Source Address Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT1_SA	PDMA_BA + 0x014	R/W	Source Address Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT2_SA	PDMA_BA + 0x024	R/W	Source Address Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT3_SA	PDMA_BA + 0x034	R/W	Source Address Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT4_SA	PDMA_BA + 0x044	R/W	Source Address Register of PDMA Channel 4	0xXXXX_XXXX

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

Bits	Description	
[31:0]	**SA**	**PDMA Transfer Source Address Register** This field indicates a 32-bit source address of PDMA controller.

Destination Address Register (PDMA_DSCTn_DA)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_DA	PDMA_BA + 0x008	R/W	Destination Address Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT1_DA	PDMA_BA + 0x018	R/W	Destination Address Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT2_DA	PDMA_BA + 0x028	R/W	Destination Address Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT3_DA	PDMA_BA + 0x038	R/W	Destination Address Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT4_DA	PDMA_BA + 0x048	R/W	Destination Address Register of PDMA Channel 4	0xXXXX_XXXX

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

Bits	Description	
[31:0]	**DA**	**PDMA Transfer Destination Address Register** This field indicates a 32-bit destination address of PDMA controller.

First Scatter-Gather Descriptor Table Offset (PDMA_DSCTn_FIRST)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_FIRST	PDMA_BA + 0x00C	R/W	First Scatter-Gather Descriptor Table Offset of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT1_FIRST	PDMA_BA + 0x01C	R/W	First Scatter-Gather Descriptor Table Offset of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT2_FIRST	PDMA_BA + 0x02C	R/W	First Scatter-Gather Descriptor Table Offset of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT3_FIRST	PDMA_BA + 0x03C	R/W	First Scatter-Gather Descriptor Table Offset of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT4_FIRST	PDMA_BA + 0x04C	R/W	First Scatter-Gather Descriptor Table Offset of PDMA Channel 4	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**
FIRST							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FIRST							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FIRST**	**PDMA First Descriptor Table Offset** This field indicates the offset of the first descriptor table address in system memory. **Write Operation:** If the system memory based address is 0x2000_0000 (PDMA_SCATBA), and the first descriptor table is start from 0x2000_0100, then this field must fill in 0x0100. **Read Operation:** When operating in scatter-gather mode, the last two bits FIRST[1:0] will become reserved. **Note1:** The first descriptor table address must be word boundary. **Note2:** Before filled transfer task in the descriptor table, user must check if the descriptor table is complete.

Current Scatter-Gather Descriptor Table Address (PDMA_CURSCATn)

Register	Offset	R/W	Description	Reset Value
PDMA_CURSCAT0	PDMA_BA + 0x050	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 0	0xXXXX_XXXX
PDMA_CURSCAT1	PDMA_BA + 0x054	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 1	0xXXXX_XXXX
PDMA_CURSCAT2	PDMA_BA + 0x058	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 2	0xXXXX_XXXX
PDMA_CURSCAT3	PDMA_BA + 0x05C	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 3	0xXXXX_XXXX
PDMA_CURSCAT4	PDMA_BA + 0x060	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 4	0xXXXX_XXXX

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

Bits	Description	
[31:0]	**CURADDR**	**PDMA Current Description Address Register (Read Only)** This field indicates a 32-bit current external description address of PDMA controller. **Note:** This field is read only and only used for Scatter-Gather mode to indicate the current external description address.

Channel Control Register (PDMA_CHCTL)

Register	Offset	R/W	Description	Reset Value
PDMA_CHCTL	PDMA_BA + 0x400	R/W	PDMA Channel 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			**CHEN4**	**CHEN3**	**CHEN2**	**CHEN1**	**CHEN0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**CHENn**	**PDMA Channel N Enable Bit** Set this bit to 1 to enable PDMAn operation. Channel cannot be active if it is not set as enabled. 0 = PDMA channel [n] Disabled. 1 = PDMA channel [n] Enabled. **Note:** Set PDMA_PAUSE or PDMA_RESET register will also clear this bit.

PDMA Transfer Pause Control Register (PDMA_PAUSE)

Register	Offset	R/W	Description	Reset Value
PDMA_PAUSE	PDMA_BA + 0x404	W	PDMA Transfer Pause 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			**PAUSE4**	**PAUSE3**	**PAUSE2**	**PAUSE1**	**PAUSE0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**PAUSEn**	**PDMA Channel N Transfer Pause Control Register (Write Only)** User can set PAUSEn bit field to pause the PDMA transfer. 0 = No effect. 1 = Pause PDMA channel n transfer. When user sets PDMA_PAUSE bit, the operation will pause the on-going transfer, but not finish the rest of transfers, and then clear the channel enable bit (PDMA_CHCTL [CHEN]) and clear request active flag. If re-enable the paused channel agian, the remaining transfers will be processed.

PDMA Software Request Register (PDMA_SWREQ)

Register	Offset	R/W	Description	Reset Value
PDMA_SWREQ	PDMA_BA + 0x408	W	PDMA Software Request 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			**SWREQ4**	**SWREQ3**	**SWREQ2**	**SWREQ1**	**SWREQ0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**SWREQn**	**PDMA Channel N Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA channel n. 0 = No effect. 1 = Generate a software request. **Note1:** User can read PDMA_TRGSTS register to know which channel is on active. Active flag may be triggered by software request or peripheral request. **Note2:** If user does not enable corresponding PDMA channel, the software request will be ignored.

PDMA Channel Request Status Register (PDMA_TRGSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_TRGSTS	PDMA_BA + 0x40C	R	PDMA Channel Request 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			**REQSTS4**	**REQSTS3**	**REQSTS2**	**REQSTS1**	**REQSTS0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**REQSTSn**	**PDMA Channel N Request Status (Read Only)** This flag indicates whether channel[n] have a request or not, no matter request from software or peripheral. When PDMA controller finishes channel transfer, this bit will be cleared automatically. 0 = PDMA Channel n has no request. 1 = PDMA Channel n has a request. **Note:** If user pauses or resets each PDMA transfer by setting PDMA_PAUSE or PDMA_RESET register respectively, this bit will be cleared automatically after finishing current transfer.

PDMA Fixed Priority Setting Register (PDMA_PRISET)

Register	Offset	R/W	Description	Reset Value
PDMA_PRISET	PDMA_BA + 0x410	R/W	PDMA Fixed Priority Setting 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			**FPRISET4**	**FPRISET3**	**FPRISET2**	**FPRISET1**	**FPRISET0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**FPRISETn**	**PDMA Channel N Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel [n] to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel is round-robin priority. 1 = Corresponding PDMA channel is fixed priority. **Note:** This field only set to fixed priority, user should use PDMA_PRICLR register to clear fixed priority.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

PDMA Fixed Priority Clear Register (PDMA_PRICLR)

Register	Offset	R/W	Description	Reset Value
PDMA_PRICLR	PDMA_BA + 0x414	W	PDMA Fixed Priority Clear 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			**FPRICLR4**	**FPRICLR3**	**FPRICLR2**	**FPRICLR1**	**FPRICLR0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**FPRICLRn**	**PDMA Channel N Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel [n] fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.

PDMA Interrupt Enable Register (PDMA_INTEN)

Register	Offset	R/W	Description	Reset Value
PDMA_INTEN	PDMA_BA + 0x418	R/W	PDMA Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**INTEN4**	**INTEN3**	**INTEN2**	**INTEN1**	**INTEN0**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[n] n=0,1..4	**INTENn**	**PDMA Channel N Interrupt Enable Register** This field is used for enabling PDMA channel[n] interrupt. 0 = PDMA channel n interrupt Disabled. 1 = PDMA channel n interrupt Enabled.

PDMA Interrupt Status Register (PDMA_INTSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_INTSTS	PDMA_BA + 0x41C	R/W	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						**REQTOF1**	**REQTOF0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**TEIF**	**TDIF**	**ABTIF**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n+8] n=0,1	**REQTOFn**	**PDMA Channel N Request Time-out Flag** This flag indicates that PDMA controller has waited peripheral request for a period defined by PDMA_TOCn, user can write 1 to clear these bits. 0 = No request time-out. 1 = Peripheral request time-out.
[7:3]	**Reserved**	Reserved.
[2]	**TEIF**	**Table Empty Interrupt Flag (Read Only)** This bit indicates PDMA channel scatter-gather table is empty. User can read PDMA_SCATSTS register to indicate which channel scatter-gather table is empty. 0 = PDMA channel scatter-gather table is not empty. 1 = PDMA channel scatter-gather table is empty.
[1]	**TDIF**	**Transfer Done Interrupt Flag (Read Only)** This bit indicates that PDMA controller has finished transmission; User can read PDMA_TDSTS register to indicate which channel finished transfer. 0 = Not finished yet. 1 = PDMA channel has finished transmission.
[0]	**ABTIF**	**PDMA Read/Write Target Abort Interrupt Flag (Read-only)** This bit indicates that PDMA has target abort error; Software can read PDMA_ABTSTS register to find which channel has target abort error. 0 = No AHB bus ERROR response received. 1 = AHB bus ERROR response received.

PDMA Channel Read/Write Target Abort Flag Register (PDMA_ABTSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_ABTSTS	PDMA_BA + 0x420	R/W	PDMA Channel Read/Write Target Abort Flag 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			**ABTIF4**	**ABTIF3**	**ABTIF2**	**ABTIF1**	**ABTIF0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**ABTIFn**	**PDMA Channel N Read/Write Target Abort Interrupt Status Flag** This bit indicates which PDMA controller has target abort error or transfer source and destination address not alignment; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel n transfer. 1 = AHB bus ERROR response received when channel n transfer.

PDMA Channel Transfer Done Flag Register (PDMA_TDSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_TDSTS	PDMA_BA + 0x424	R/W	PDMA Channel Transfer Done Flag 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			**TDIF4**	**TDIF3**	**TDIF2**	**TDIF1**	**TDIF0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**TDIFn**	**PDMA Channel N Transfer Done Flag Register** This bit indicates whether PDMA controller channel transfer has been finished or not, user can write 1 to clear these bits. 0 = PDMA channel transfer has not finished. 1 = PDMA channel has finished transmission.

PDMA Scatter-Gather Table Empty Status Register (PDMA_SCATSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_SCATSTS	PDMA_BA + 0x428	R/W	PDMA Scatter-Gather Table Empty 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			TEMPTYF4	TEMPTYF3	TEMPTYF2	TEMPTYF1	TEMPTYF0

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:0]	**TEMPTYFn**	**Table Empty Flag Register** This bit indicates which PDMA channel table is empty when channel has a request, no matter request from software or peripheral, but operation mode of channel descriptor table is idle mode, or channel has finished current transfer and next table operation mode is idle mode for PDMA Scatter-Gather mode. User can write 1 to clear these bits. 0 = PDMA channel scatter-gather table is not empty. 1 = PDMA channel scatter-gather table is empty.

Note: The n represents the PDMA channel.

PDMA Transfer Active Flag Register (PDMA_TACTSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_TACTSTS	PDMA_BA + 0x42C	R	PDMA Transfer Active Flag 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			**TXACTF4**	**TXACTF3**	**TXACTF2**	**TXACTF1**	**TXACTF0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**TXACTFn**	**PDMA Channel N Transfer on Active Flag Register (Read Only)** This bit indicates which PDMA channel is in active. 0 = PDMA channel is not finished. 1 = PDMA channel is active.

PDMA Time-out Prescaler Register (PDMA_TOUTPSC)

Register	Offset	R/W	Description	Reset Value
PDMA_TOUTPSC	PDMA_BA + 0x430	R/W	PDMA Time-out Prescaler 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	**TOUTPSC1**			**Reserved**	**TOUTPSC0**		

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:4]	**TOUTPSC1**	**PDMA Channel 1 Time-out Clock Source Prescaler Bits** 000 = PDMA channel 1 time-out clock source is HCLK/2^{8}. 001 = PDMA channel 1 time-out clock source is HCLK/2^{9}. 010 = PDMA channel 1 time-out clock source is HCLK/2^{10}. 011 = PDMA channel 1 time-out clock source is HCLK/2^{11}. 100 = PDMA channel 1 time-out clock source is HCLK/2^{12}. 101 = PDMA channel 1 time-out clock source is HCLK/2^{13}. 110 = PDMA channel 1 time-out clock source is HCLK/2^{14}. 111 = PDMA channel 1 time-out clock source is HCLK/2^{15}.
[3]	**Reserved**	Reserved.
[2:0]	**TOUTPSC0**	**PDMA Channel 0 Time-out Clock Source Prescaler Bits** 000 = PDMA channel 0 time-out clock source is HCLK/2^{8}. 001 = PDMA channel 0 time-out clock source is HCLK/2^{9}. 010 = PDMA channel 0 time-out clock source is HCLK/2^{10}. 011 = PDMA channel 0 time-out clock source is HCLK/2^{11}. 100 = PDMA channel 0 time-out clock source is HCLK/2^{12}. 101 = PDMA channel 0 time-out clock source is HCLK/2^{13}. 110 = PDMA channel 0 time-out clock source is HCLK/2^{14}. 111 = PDMA channel 0 time-out clock source is HCLK/2^{15}.

PDMA Time-out Enable Register (PDMA_TOUTEN)

Register	Offset	R/W	Description	Reset Value
PDMA_TOUTEN	PDMA_BA + 0x434	R/W	PDMA Time-out 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						TOUTEN1	TOUTEN0

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TOUTEN1**	**PDMA Channel 1 Time-out Enable Bit** 0 = PDMA Channel 1 time-out function Disable. 1 = PDMA Channel 1 time-out function Enable.
[0]	**TOUTEN0**	**PDMA Channel 0 Time-out Enable Bit** 0 = PDMA Channel 0 time-out function Disable. 1 = PDMA Channel 0 time-out function Enable.

PDMA Time-out Interrupt Enable Register (PDMA_TOUTIEN)

Register	Offset	R/W	Description	Reset Value
PDMA_TOUTIEN	PDMA_BA + 0x438	R/W	PDMA Time-out Interrupt Enable Register	0x0000_0000

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="6">Reserved</td><td>TOUTIEN1</td><td>TOUTIEN0</td></tr>
</table>

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TOUTIEN1**	**PDMA Channel 1 Time-out Interrupt Enable Bit** 0 = PDMA Channel 1 time-out interrupt Disable. 1 = PDMA Channel 1 time-out interrupt Enable.
[0]	**TOUTIEN0**	**PDMA Channel 0 Time-out Interrupt Enable Bit** 0 = PDMA Channel 0 time-out interrupt Disable. 1 = PDMA Channel 0 time-out interrupt Enable.

PDMA Scatter-Gather Descriptor Table Base Address Register (PDMA_SCATBA)

Register	Offset	R/W	Description	Reset Value
PDMA_SCATBA	PDMA_BA + 0x43C	R/W	PDMA Scatter-Gather Descriptor Table Base Address Register	0x2000_0000

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

Bits	Description	
[31:16]	**SCATBA**	**PDMA Scatter-gather Descriptor Table Address Register** In Scatter-Gather mode, this is the base address for calculating the next link - list address. The next link address equation is Next Link Address = PDMA_SCATBA + PDMA_DSCT_FIRST. **Note:** Only useful in Scatter-Gather mode.
[15:0]	**Reserved**	Reserved.

PDMA Channel 0 and Channel 1 Time-out Counter Register (PDMA_TOC0_1)

Register	Offset	R/W	Description	Reset Value
PDMA_TOC0_1	PDMA_BA + 0x440	R/W	PDMA Channel 0 and Channel 1 Time-out Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
TOC1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TOC1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TOC0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TOC0							

Bits	Description	
[31:16]	**TOC1**	**Time-out Counter for Channel 1** This controls the period of time-out function for channel 1. The calculation unit is based on TOUTPSC1 (PDMA_TOUTPSC[6:4]) clock. The example of time-out period can refer TOC0 bit description.
[15:0]	**TOC0**	**Time-out Counter for Channel 0** This controls the period of time-out function for channel 0. The calculation unit is based on TOUTPSC0 (PDMA_TOUTPSC[2:0]) clock. Time-out period = (Period of time-out clock) * (16-bit TOCn),n = 0 ~ 1.

PDMA Channel Reset Control Register (PDMA_RESET)

Register	Offset	R/W	Description	Reset Value
PDMA_RESET	PDMA_BA + 0x460	R/W	PDMA channel Reset 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			RESET4	RESET3	RESET2	RESET1	RESET0

Bits	Description	
[31:5]	**Reserved**	Reserved.
[n] n=0,1..4	**RESETn**	**PDMA Channel N Reset Control Register** User can set this bit field to reset the PDMA channel. 0 = No effect. 1 = Reset PDMA channel n. When user sets PDMA_RESET bit, the operation will finish the on-going transfer and then clear the channel enable bit (PDMA_CHCTL [CHEN]) and clear request active flag. If re-enable channel after channel reset, PDMA will re-load the channel description table to execute PDMA task. **Note:** This bit will be cleared automatically after finishing reset process.

PDMA Channel 0 to Channel 3 Request Source Select Register (PDMA_REQSEL0_3)

Register	Offset	R/W	Description	Reset Value
PDMA_REQSEL0_3	PDMA_BA + 0x480	R/W	PDMA Channel 0 to Channel 3 Request Source Select Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		**REQSRC3**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**REQSRC2**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**REQSRC1**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**REQSRC0**					

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**REQSRC3**	**Channel 3 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 3. User can configure the peripheral setting by REQSRC3. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23:22]	**Reserved**	Reserved.
[21:16]	**REQSRC2**	**Channel 2 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 2. User can configure the peripheral setting by REQSRC2. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15:14]	**Reserved**	Reserved.
[13:8]	**REQSRC1**	**Channel 1 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 1. User can configure the peripheral setting by REQSRC1. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7:6]	**Reserved**	Reserved.
[5:0]	**REQSRC0**	**Channel 0 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 0. User can configure the peripheral by setting REQSRC0. 0 = Disable PDMA peripheral request. 1 = reserved. 4 = Channel connects to UART_TX. 5 = Channel connects to UART_RX. 6 = Reserved.

Bits	Description	
		7 = Reserved. 8 = Reserved. 9 = Reserved. 10 = Channel connects to USCI_TX. 11 = Channel connects to USCI_RX. 12 = Reserved. 13 = Reserved. 14 = Reserved. 15 = Reserved. 16 = Channel connects to SPI_TX. 17 = Channel connects to SPI_RX. 18 = Reserved. 19 = Reserved. 20 = Channel connects to ADC_RX. 21 = Channel connects to PWM0_P0_RX. 22 = Channel connects to PWM0_P1_RX. 23 = Channel connects to PWM0_P2_RX. 24 = Channel connects to PWM1_P0_RX. 25 = Channel connects to PWM1_P1_RX. 26 = Channel connects to PWM1_P2_RX. 27 = Reserved. 28 = Channel connects to I2C0_TX. 29 = Channel connects to I2C0_RX. 30 = Channel connects to I2C1_TX. 31 = Channel connects to I2C1_RX. 32 = Channel connects to TMR0. 33 = Channel connects to TMR1. 34 = Channel connects to TMR2. 35 = Channel connects to TMR3. Others = Reserved. **Note 1:** A peripheral can’t assign to two channels at the same time. **Note 2:** This field is useless when transfer between memory and memory.

PDMA Channel 4 Request Source Select Register (PDMA_REQSEL4)

Register	Offset	R/W	Description	Reset Value
PDMA_REQSEL4	PDMA_BA + 0x484	R/W	PDMA Channel 4 Request Source 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		REQSRC4					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**REQSRC4**	**Channel 4 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 4. User can configure the peripheral setting by REQSRC4. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.

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 CNT (TIMERx_CNT[23:0])
- Supports event counting function
- 24-bit capture value is readable through CAPDAT (TIMERx_CAP[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
- Support Timer0 ~ Timer3 time-out interrupt signal or capture interrupt signal to trigger PWM, BPWM, PDMA, ADC and DAC function

6.7.3 Block Diagram

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

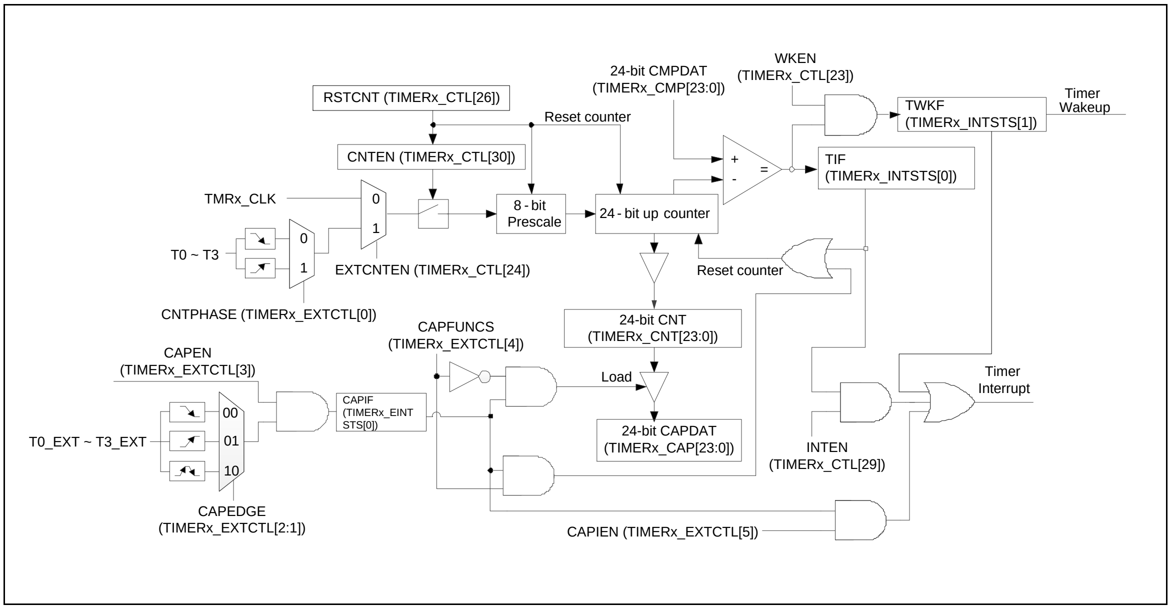


Figure 6.7-1 Timer Controller Block Diagram

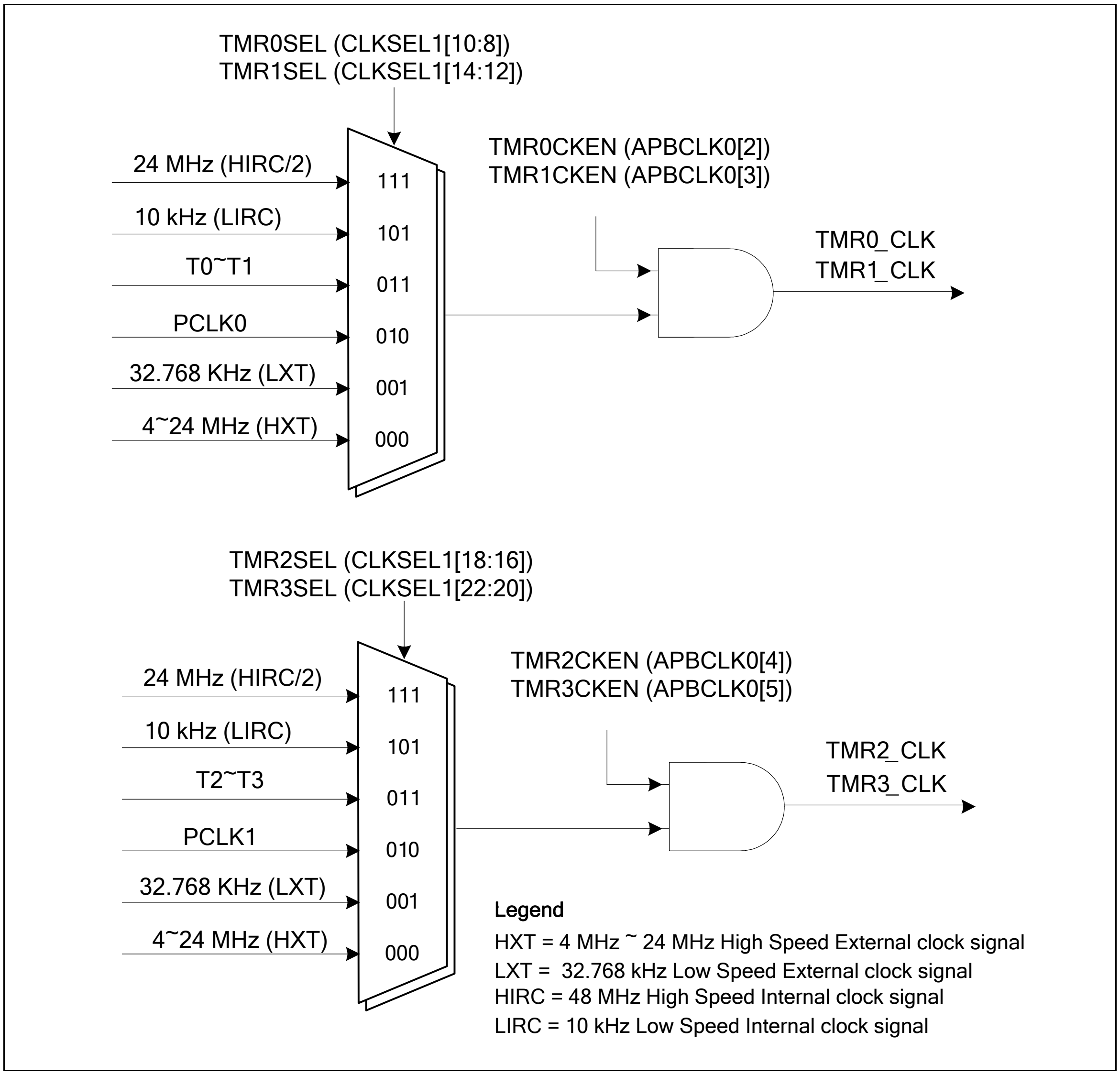


Figure 6.7-2 Clock Source of Timer Controller

6.7.4 Basic Configuration

The peripheral clock source of Tiimer0 ~ Timer3 can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]) and selected as different frequency in TMR0SEL (CLK_CLKSEL1[10:8]) for Timer0, TMR1SEL (CLK_CLKSEL1[14:12]) for Timer1, TMR2SEL (CLK_CLKSEL1[18:16]) for Timer2 and TMR3SEL (CLK_CLKSEL1[22:20]) for Timer3.

6.7.5 Functional Description

6.7.5.1 Timer Interrupt Flag

Timer controller supports two interrupt flags; one is TIF (TIMERx_INTSTS[0]) and it's set while timer counter value CNT (TIMERx_CNT[23:0]) matches the timer compared value CMPDAT (TIMERx_CMP[23:0]), the other is CAPIF (TIMERx_EINTSTS[0]) and it's set when the transition on the Tx_EXT pin associated CAPEDGE (TIMERx_EXTCTL[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 (TIMERx_CTL[28:27] is b’00) and CNTEN (TIMERx_CTL[30]) is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF (TIMERx_INTSTS[0]) will be set to 1, CNT value and CNTEN bit is cleared automatically by timer controller then timer counting operation stops. In the meantime, if the INTEN (TIMERx_CTL[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 (TIMERx_CTL[28:27] is b’01) and CNTEN (TIMERx_CTL[30]) is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF (TIMERx_INTSTS[0]) will be set to 1, CNT value will be cleared automatically by timer controller and timer counter operates counting again. In the meantime, if the INTEN (TIMERx_CTL[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 CMPDAT value periodically until the CNTEN bit is cleared by user.

6.7.5.5 Toggle-Output Mode

If timer controller is configured at toggle-output mode (TIMERx_CTL[28:27] is b’10) and CNTEN (TIMERx_CTL[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 T0 ~ T3 pin to output signal while specify TIF (TIMERx_INTSTS[0]) is set. Thus, the toggle-output signal on T0 ~ T3 pin is high and changing back and forth with 50% duty cycle. Also switch TGLPINSEL (TIMERx_CTL[22]) that toggle mode output to Tx/Tx_EXT.

6.7.5.6 Continuous Counting Mode

If timer controller is configured at continuous counting mode (TIMERx_CTL[28:27] is b’11) and CNTEN (TIMERx_CTL[30]) is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF (TIMERx_INTSTS[0]) will be set to 1 and CNT value keeps up counting. In the meantime, if the INTEN (TIMERx_CTL[29]) is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. User can change different CMPDAT value immediately without disabling timer counting and restarting timer counting in this mode.

For example, CMPDAT value is set as 80, first. The TIF will set to 1 when CNT value is equal to 80, timer counter is kept counting and CNT 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 CMPDAT value as 200 and clears TIF, the TIF will set to 1 again when CNT value reaches to 200. At last, user programs CMPDAT as 500 and clears TIF, the TIF will set to 1 again when CNT value reaches to 500.

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

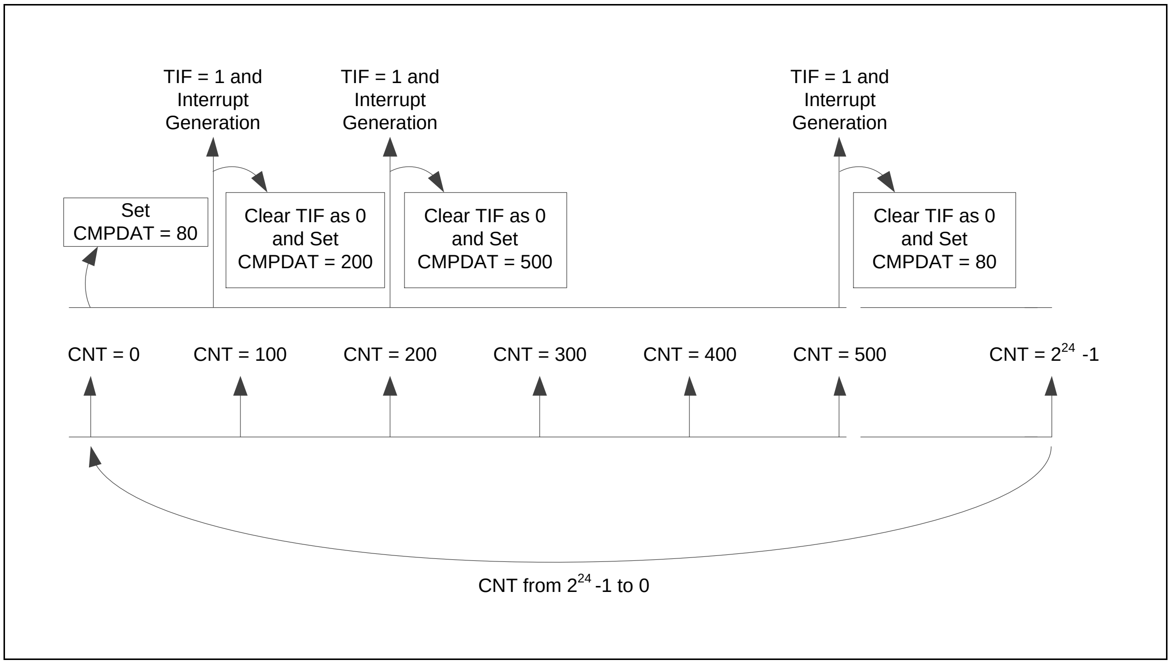


Figure 6.7-3 Continuous Counting Mode

6.7.5.7 Event Counting Mode

Timer controller also provides an application which can count the input event from Tx (x= 0~3) pin and the number of event will reflect to CNT (TIMERx_CNT[23:0]) value. It is also called as event counting function. In this function, EXTCNTEN (TIMERx_CTL[24]) should be set and the timer peripheral clock source should be set as PCLK.

User can enable or disable Tx pin de-bounce circuit by setting CNTDBEN (TIMERx_EXTCTL[7]). The input event frequency should be less than 1/3 PCLK if Tx pin de-bounce is disabled or less than 1/8 PCLK if Tx pin de-bounce is enabled to assure the returned CNT value is correct, and user can also select edge detection phase of Tx pin by setting CNTPHASE (TIMERx_EXTCTL[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 CNT (TIMERx_CNT[23:0]) for Tx pin. toggle-output mode is not supported since T0~T3 are used for input event

6.7.5.8 External Capture Mode

The event capture function is used to load CNT (TIMERx_CNT[23:0]) value to CAPDAT (TIMERx_CAP[23:0]) value while edge transition detected on Tx_EXT (x= 0~3) pin. In this mode, CAPFUNCS (TIMERx_EXTCTL[4]) should be set as 0 to select Tx_EXT transition which is using to trigger event capture function and the timer peripheral clock source should be set as PCLK.

User can enable or disable Tx_EXT pin de-bounce circuit by setting CAPDBEN (TIMERx_EXTCTL[6]). The transition frequency of Tx_EXT pin should be less than 1/3 PCLK if Tx_EXT pin de-bounce is disabled or less than 1/8 PCLK if Tx_EXT pin de-bounce is enabled to assure the capture function can be work normally, and user can also select edge transition detection of Tx_EXT pin by setting CAPEDGE (TIMERx_EXTCTL[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 Tx_EXT pin is detected.

Set CAPIEN (TIMERx_EXTCTL[5]) that generate CAPIF. Users must consider the Timer will keep register TIMERx_CAP unchanged and drop the new capture value, if the CPU does not clear the CAPIF status.

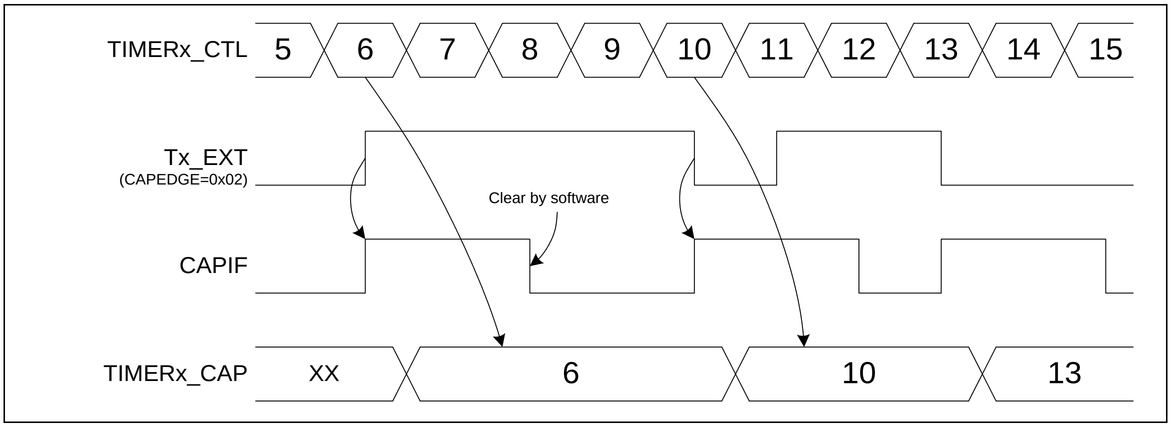


Figure 6.7-4 External Capture Mode

6.7.5.9 External Reset Counter Mode

Timer controller also provides reset counter function to reset CNT (TIMERx_CNT[23:0]) value while edge transition detected on Tx_EXT (x= 0~3). In this mode, most of the settings are the same as event capture mode except CAPFUNCS (TIMERx_EXTCTL[4]) should be set as 1 to select Tx_EXT transition which is using to trigger reset counter value.

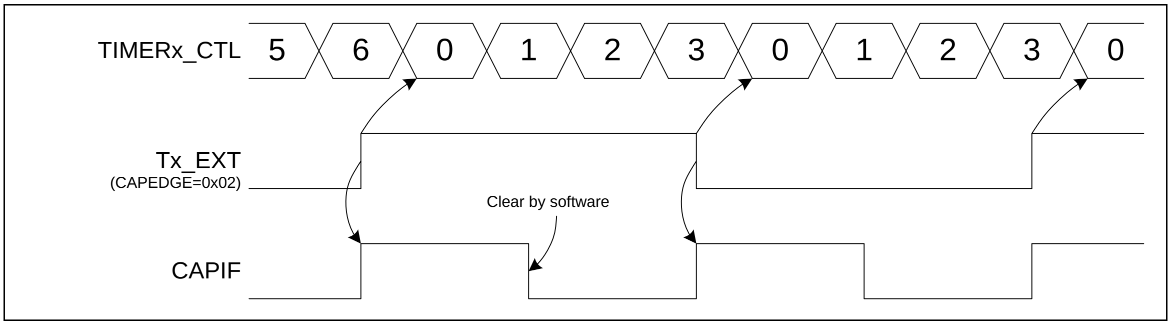


Figure 6.7-5 External Reset Counter Mode

6.7.5.10 *Timer Trigger Function*

Timer controller provides timer time-out interrupt or capture interrupt to trigger PWM, BPWM, PDMA, DAC and ADC. If TRGSSEL (TIMERx_CTL[18]) is 0, time-out interrupt signal is used to trigger PWM, BPWM, PDMA, ADC and DAC. If TRGSSEL (TIMERx_CTL[18]) is 1, capture interrupt signal is used to trigger PWM, BPWM, PDMA, ADC and DAC.

When the TRGPDMA (TIMERx_CTL[8]) is set, if the timer interrupt signal is generated, the timer controller will generate a request to PDMA.

When the TRGBPWM (TIMERx_CTL[9]) is set, if the timer interrupt signal is generated, the timer controller will generate a trigger pulse as BPWM external clock source.

When the TRGPWM (TIMERx_CTL[19]) is set, if the timer interrupt signal is generated, the timer controller will generate a trigger pulse as PWM external clock source.

When the TRGDAC (TIMERx_CTL[20]) is set, if the timer interrupt signal is generated, the timer controller will trigger DAC to start converter.

When the TRGADC (TIMERx_CTL[21]) is set, if the timer interrupt signal is generated, the timer controller will trigger ADC to start converter.

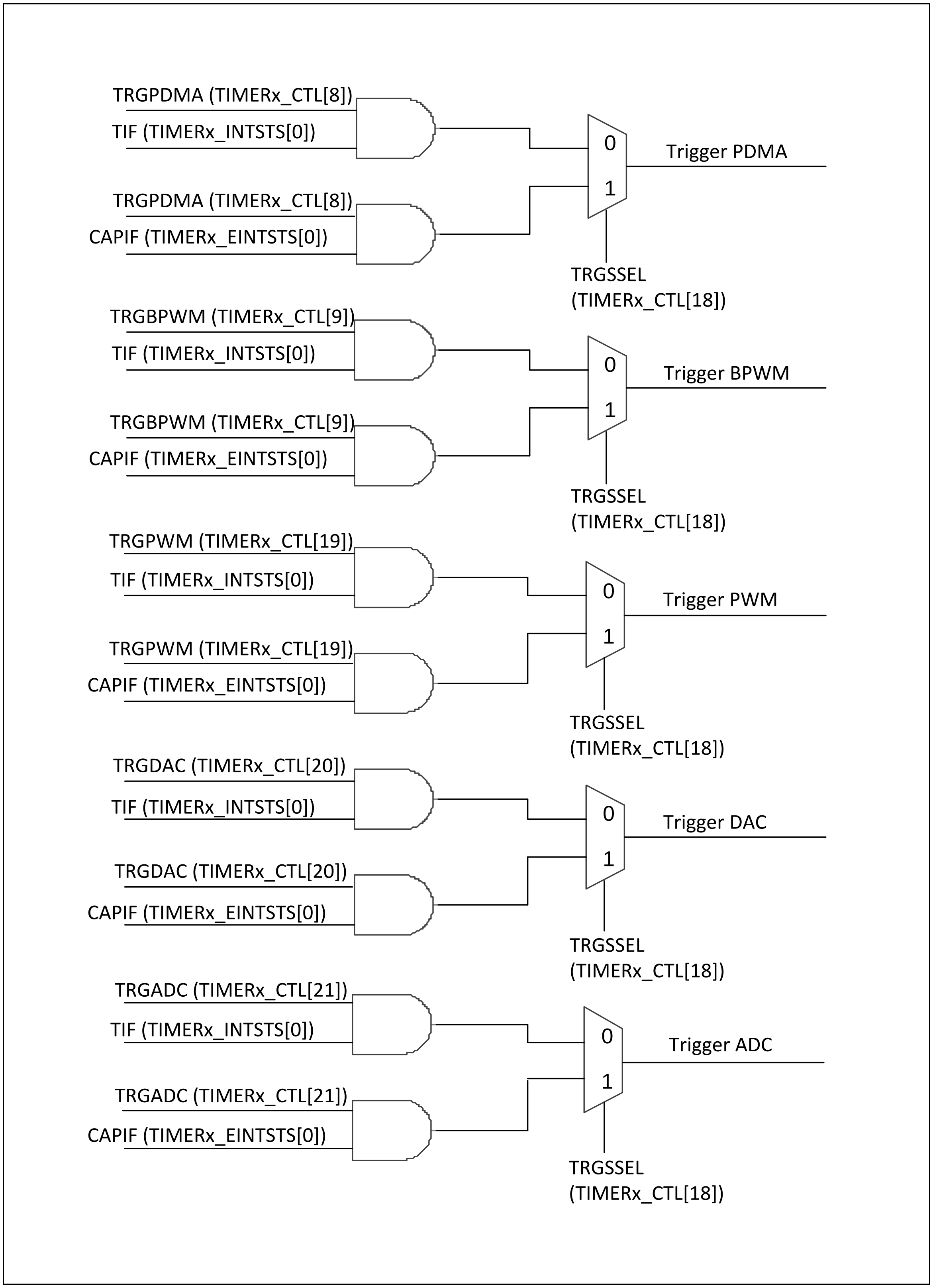


Figure 6.7-6 Internal Timer Trigger

6.7.5.11 other operation of Timer

User can set ICEDEBUG (TIMERx_CTL[31]) to 1, that TIMER counter will keep going no matter CPU is held by ICE or not. When user want to operate at different mode, user can set OPMODE (TIMERx_CTL[28:27]).

Set RSTCNT (TIMERx_CTL[26]) will reset the 24-bit up counter value CNT (TIMERx_CNT[23:0]). When user want to monitor TIMER counter, user can read ACTSTS (TIMERx_CTL[25]).

TIMER controller also provides wake-up function. Set WKEN (TIMERx_CTL[23]) and generate TIMER interrupt in power-down mode, the TWKF (TIMERx_INTSTS[1]) will be set.

Set PSC (TIMERx_CTL[7:0]) that TIMER input clock or event source is divided by (PSC+1) before it is fed to the timer up counter.

6.7.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
TIMER Base Address: **TMR01_BA = 0x4001_0000** **TMR23_BA = 0x4001_1000**				
TIMER0_CTL	TMR01_BA+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TIMER0_CMP	TMR01_BA+0x04	R/W	Timer0 Compare Register	0x0000_0000
TIMER0_INTSTS	TMR01_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TIMER0_CNT	TMR01_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TIMER0_CAP	TMR01_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TIMER0_EXTCTL	TMR01_BA+0x14	R/W	Timer0 External Control Register	0x0000_0000
TIMER0_EINTSTS	TMR01_BA+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TIMER1_CTL	TMR01_BA+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TIMER1_CMP	TMR01_BA+0x24	R/W	Timer1 Compare Register	0x0000_0000
TIMER1_INTSTS	TMR01_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER1_CNT	TMR01_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TIMER1_CAP	TMR01_BA+0x30	R	Timer1 Capture Data Register	0x0000_0000
TIMER1_EXTCTL	TMR01_BA+0x34	R/W	Timer1 External Control Register	0x0000_0000
TIMER1_EINTSTS	TMR01_BA+0x38	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2 Control and Status Register	0x0000_0005
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2 Compare Register	0x0000_0000
TIMER2_INTSTS	TMR23_BA+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TIMER2_CNT	TMR23_BA+0x0C	R	Timer2 Data Register	0x0000_0000
TIMER2_CAP	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TIMER2_EXTCTL	TMR23_BA+0x14	R/W	Timer2 External Control Register	0x0000_0000
TIMER2_EINTSTS	TMR23_BA+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TIMER3_CTL	TMR23_BA+0x20	R/W	Timer3 Control and Status Register	0x0000_0005
TIMER3_CMP	TMR23_BA+0x24	R/W	Timer3 Compare Register	0x0000_0000
TIMER3_INTSTS	TMR23_BA+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000
TIMER3_CNT	TMR23_BA+0x2C	R	Timer3 Data Register	0x0000_0000

TIMER3_CAP	TMR23_BA+0x30	R	Timer3 Capture Data Register	0x0000_0000
TIMER3_EXTCTL	TMR23_BA+0x34	R/W	Timer3 External Control Register	0x0000_0000
TIMER3_EINTSTS	TMR23_BA+0x38	R/W	Timer3 External Interrupt Status Register	0x0000_0000

6.7.7 Register Description

Timer Control Register (TIMERx_CTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_CTL	TMR01_BA+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TIMER1_CTL	TMR01_BA+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2 Control and Status Register	0x0000_0005
TIMER3_CTL	TMR23_BA+0x20	R/W	Timer3 Control and Status Register	0x0000_0005

31	30	29	28	27	26	25	24
ICEDEBUG	CNTEN	INTEN	OPMODE		RSTCNT	ACTSTS	EXTCNTEN
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
WKEN	TGLPINSEL	TRGADC	TRGDAC	TRGPWM	TRGSSEL	Reserved	Reserved
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						TRGBPWM	TRGPDMA
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSC							

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable (Write Protect)** 0 = ICE debug mode acknowledgement affects 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 SYS_REGLCTL register.
[30]	**CNTEN**	**Timer Counting Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. **Note1:** In stop status, and then set CNTEN 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 (TIMER_CTL[28:27] = 00) when the timer interrupt flag TIF (TIMERx_INTSTS[0]) is generated.
[29]	**INTEN**	**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]	**OPMODE**	**Timer Operating Mode Select** 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]	**RSTCNT**	**Timer Counter Reset Bit** Setting this bit will reset the 24-bit up counter value CNT (TIMERx_CNT[23:0]) and also force CNTEN (TIMERx_CTL[30]) to 0 if ACTSTS (TIMERx_CTL[25]) is 1. 0 = No effect. 1 = Reset internal 8-bit prescale counter, 24-bit up counter value and CNTEN bit.
[25]	**ACTSTS**	**Timer Active Status Bit (Read Only)** This bit indicates the 24-bit up counter status. 0 = 24-bit up counter is not active. 1 = 24-bit up counter is active.
[24]	**EXTCNTEN**	**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 PCLK as timer clock source.
[23]	**WKEN**	**Wake-up Function Enable Bit** If this bit is set to 1, while timer interrupt flag TIF (TIMERx_INTSTS[0]) is 1 and INTEN (TIMERx_CTL[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]	**TGLPINSEL**	**Toggle-output Pin Select** 0 = Toggle mode output to Tx (Timer Event Counter Pin). 1 = Toggle mode output to Tx_EXT (Timer External Capture Pin).
[21]	**TRGADC**	**Trigger ADC Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger ADC. 0 = Timer interrupt trigger ADC Disabled. 1 = Timer interrupt trigger ADC Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger ADC. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger ADC.
[20]	**TRGDAC**	**Trigger DAC Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger DAC. 0 = Timer interrupt trigger DAC Disabled. 1 = Timer interrupt trigger DAC Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger DAC. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger DAC.
[19]	**TRGPWM**	**Trigger PWM Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger PWM.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

		0 = Timer interrupt trigger PWM Disabled. 1 = Timer interrupt trigger PWM Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger PWM. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger PWM.
[18]	**TRGSSEL**	**Trigger Source Select Bit** This bit is used to select trigger source from Timer time-out interrupt signal or capture interrupt signal. 0 = Timer time-out interrupt signal is used to trigger PWM, BPWM, PDMA,ADC and DAC. 1 = Capture interrupt signal is used to trigger PWM, BPWM, PDMA, ADC and DAC.
[17:10]	**Reserved**	Reserved.
[9]	**TRGBPWM**	**Trigger BPWM Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger BPWM. 0 = Timer interrupt trigger BPWM Disabled. 1 = Timer interrupt trigger BPWM Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger BPWM. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger BPWM.
[8]	**TRGPDMA**	**Trigger PDMA Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger PDMA. 0 = Timer interrupt trigger PDMA Disabled. 1 = Timer interrupt trigger PDMA Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger PDMA. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger PDMA.
[7:0]	**PSC**	**Prescale Counter** Timer input clock or event source is divided by (PSC+1) before it is fed to the timer up counter. If this field is 0 (PSC = 0), then there is no scaling.

Timer Compare Register (TIMERx_CMP)

Register	Offset	R/W	Description	Reset Value
TIMER0_CMP	TMR01_BA+0x04	R/W	Timer0 Compare Register	0x0000_0000
TIMER1_CMP	TMR01_BA+0x24	R/W	Timer1 Compare Register	0x0000_0000
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2 Compare Register	0x0000_0000
TIMER3_CMP	TMR23_BA+0x24	R/W	Timer3 Compare Register	0x0000_0000

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

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

Timer Interrupt Status Register (TIMERx_INTSTS)

Register	Offset	R/W	Description	Reset Value
TIMER0_INTSTS	TMR01_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TIMER1_INTSTS	TMR01_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER2_INTSTS	TMR23_BA+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TIMER3_INTSTS	TMR23_BA+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31:2]	Reserved	Reserved.
[1]	TWKF	**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 is cleared by writing 1 to it.
[0]	TIF	**Timer Interrupt Flag** This bit indicates the interrupt flag status of Timer while 24-bit timer up counter CNT (TIMERx_CNT[23:0]) value reaches to CMPDAT (TIMERx_CMP[23:0]) value. 0 = No effect. 1 = CNT value matches the CMPDAT value. **Note:** This bit is cleared by writing 1 to it.

Timer Data Register (TIMERx_CNT)

Register	Offset	R/W	Description	Reset Value
TIMER0_CNT	TMR01_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TIMER1_CNT	TMR01_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TIMER2_CNT	TMR23_BA+0x0C	R	Timer2 Data Register	0x0000_0000
TIMER3_CNT	TMR23_BA+0x2C	R	Timer3 Data Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CNT**	**Timer Data Register** This field can be reflected the internal 24-bit timer counter value or external event input counter value from Tx (x=0~3) pin. If EXTCNTEN (TIMERx_CTL[24]) is 0, user can read CNT value to get current 24-bit counter value . If EXTCNTEN (TIMERx_CTL[24]) is 1, user can read CNT value to get current 24-bit event input counter value.

Timer Capture Data Register (TIMERx_CAP)

Register	Offset	R/W	Description	Reset Value
TIMER0_CAP	TMR01_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TIMER1_CAP	TMR01_BA+0x30	R	Timer1 Capture Data Register	0x0000_0000
TIMER2_CAP	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TIMER3_CAP	TMR23_BA+0x30	R	Timer3 Capture Data Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CAPDAT**	**Timer Capture Data Register** When CAPEN (TIMERx_EXTCTL[3]) bit is set, CAPFUNCS (TIMERx_EXTCTL[4]) bit is 0, and a transition on Tx_EXT pin is matched with the CAPEDGE (TIMERx_EXTCTL[2:1]) setting, CAPIF (TIMERx_EINTSTS[0]) will set to 1 and the current timer counter value CNT (TIMERx_CNT[23:0]) will be auto-loaded into this CAPDAT field.

Timer External Control Register (TIMERx_EXTCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_EXTCTL	TMR01_BA+0x14	R/W	Timer0 External Control Register	0x0000_0000
TIMER1_EXTCTL	TMR01_BA+0x34	R/W	Timer1 External Control Register	0x0000_0000
TIMER2_EXTCTL	TMR23_BA+0x14	R/W	Timer2 External Control Register	0x0000_0000
TIMER3_EXTCTL	TMR23_BA+0x34	R/W	Timer3 External Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**CNTDBEN**	**Timer Counter Pin De-bounce Enable Bit** 0 = Tx (x= 0~3) pin de-bounce Disabled. 1 = Tx (x= 0~3) pin de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of Tx pin is detected with de-bounce circuit.
[6]	**CAPDBEN**	**Timer External Capture Pin De-bounce Enable Bit** 0 = Tx_EXT (x= 0~3) pin de-bounce Disabled. 1 = Tx_EXT (x= 0~3) pin de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of Tx_EXT pin is detected with de-bounce circuit.
[5]	**CAPIEN**	**Timer External Capture Interrupt Enable Bit** 0 = Tx_EXT (x= 0~3) pin detection Interrupt Disabled. 1 = Tx_EXT (x= 0~3) pin detection Interrupt Enabled. **Note:** CAPIEN is used to enable timer external interrupt. If CAPIEN enabled, timer will rise an interrupt when CAPIF (TIMERx_EINTSTS[0]) is 1. For example, while CAPIEN = 1, CAPEN = 1, and CAPEDGE = 00, a 1 to 0 transition on the Tx_EXT pin will cause the CAPIF to be set then the interrupt signal is generated and sent to NVIC to inform CPU.
[4]	**CAPFUNCS**	**Capture Function Selection** 0 = External Capture Mode Enabled. 1 = External Reset Mode Enabled.

		Note1: When CAPFUNCS is 0, transition on Tx_EXT (x= 0~3) pin is using to save the 24-bit timer counter value. **Note2:** When CAPFUNCS is 1, transition on Tx_EXT (x= 0~3) pin is using to reset the 24-bit timer counter value.
[3]	**CAPEN**	**Timer External Capture Pin Enable** This bit enables the Tx_EXT pin. 0 =Tx_EXT (x= 0~3) pin Disabled. 1 =Tx_EXT (x= 0~3) pin Enabled.
[2:1]	**CAPEDGE**	**Timer External Capture Pin Edge Detect** 00 = A Falling edge on Tx_EXT (x= 0~3) pin will be detected. 01 = A Rising edge on Tx_EXT (x= 0~3) pin will be detected. 10 = Either Rising or Falling edge on Tx_EXT (x= 0~3) pin will be detected. 11 = Reserved.
[0]	**CNTPHASE**	**Timer External Count Phase** This bit indicates the detection phase of external counting pin Tx (x= 0~3). 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 (TIMERx_EINTSTS)

Register	Offset	R/W	Description	Reset Value
TIMER0_EINTSTS	TMR01_BA+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TIMER1_EINTSTS	TMR01_BA+0x38	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TIMER2_EINTSTS	TMR23_BA+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TIMER3_EINTSTS	TMR23_BA+0x38	R/W	Timer3 External Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CAPIF**	**Timer External Capture Interrupt Flag** This bit indicates the timer external capture interrupt flag status. 0 = Tx_EXT (x= 0~3) pin interrupt did not occur. 1 = Tx_EXT (x= 0~3) pin interrupt occurred. **Note1:** This bit is cleared by writing 1 to it. **Note2:** When CAPEN (TIMERx_EXTCTL[3]) bit is set, CAPFUNCS (TIMERx_EXTCTL[4]) bit is 0, and a transition on Tx_EXT (x= 0~3) pin is matched with the CAPEDGE (TIMERx_EXTCTL[2:1]) setting, this bit will set to 1 by hardware. **Note3:** There is a new incoming capture event detected before CPU clearing the CAPIF status. If the above condition occurred, the Timer will keep register TIMERx_CAP unchanged and drop the new capture value.

6.8 Basic PWM Generator and Capture Timer (BPWM)

6.8.1 Overview

The NUC121/125 series provides two BPWM generators: BPWM0 and BPWM1 as shown in Figure 6.8-1. Each BPWM supports 6 channels of BPWM output or input capture. There is a 12-bit prescaler to support flexible clock to the 16-bit BPWM counter with 16-bit comparator. The BPWM counter supports up, down and up-down counter types, all 6 channels share one counter. BPWM uses the comparator compared with counter to generate events. These events are used to generate BPWM pulse, interrupt and trigger signal for ADC to start conversion. For BPWM output control unit, it supports polarity output, independent pin mask and tri-state output enable.

The BPWM generator also supports input capture function to latch BPWM counter value to corresponding register when input channel has a rising transition, falling transition or both transition is happened.

6.8.2 Features

6.8.2.1 BPWM function features

- Supports maximum clock frequency up to100 MHz
- Supports up to two BPWM modules, each module provides 6 output channels
- Supports independent mode for BPWM output/Capture input channel
- Supports 12-bit pre-scalar from 1 to 4096
- Supports 16-bit resolution BPWM counter, each module provides 1 BPWM counter
  - ◆ Up, down and up/down counter operation type
- Supports mask function and tri-state enable for each BPWM pin
- Supports interrupt on the following events:
  - ◆ BPWM counter match zero, period value or compared value
- Supports trigger ADC on the following events:
  - ◆ BPWM counter match zero, period value or compared value

6.8.2.2 Capture Function Features

- Supports up to 12 capture input channels with 16-bit resolution
- Supports rising edge or falling edge or both edges capture condition
- Supports input rising/falling edge or both edges capture interrupt
- Supports rising/falling or both edges capture with counter reload option

6.8.3 Block Diagram

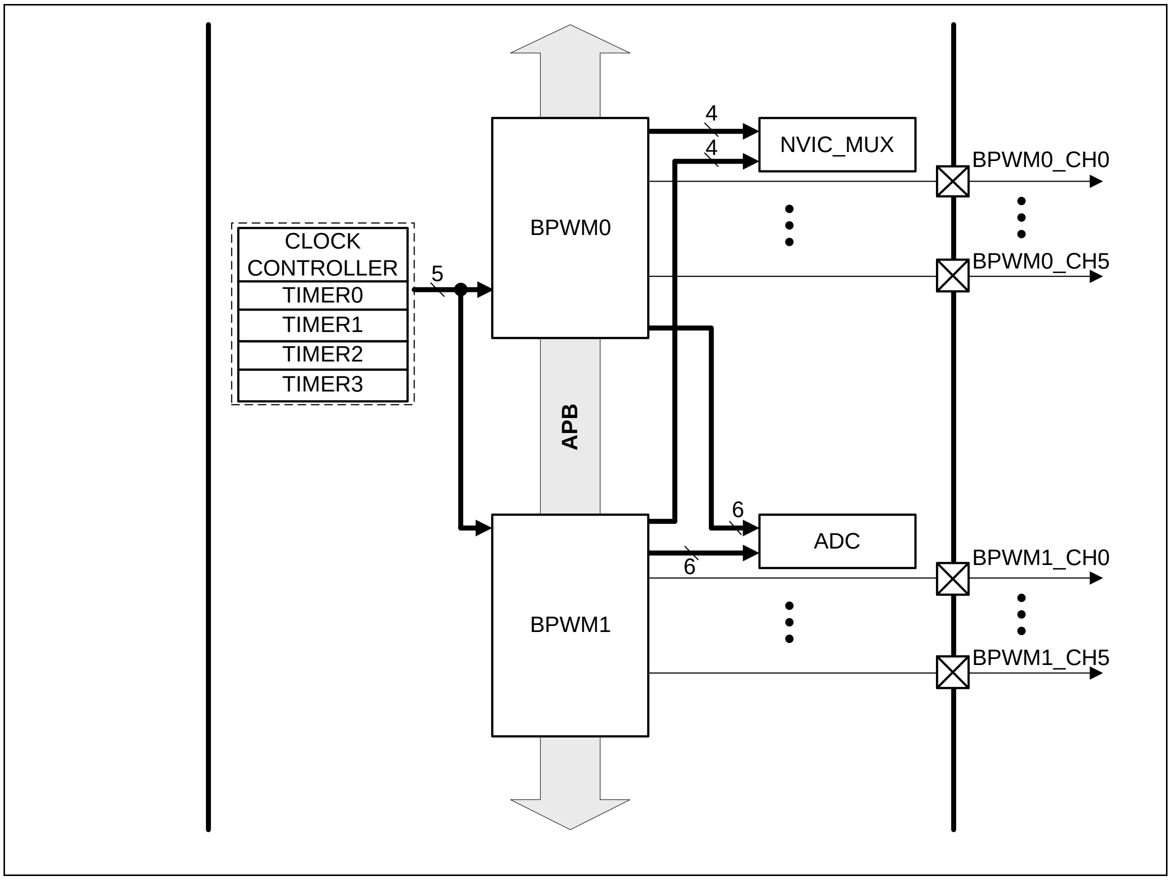


Figure 6.8-1 BPWM Generator Overview Block Diagram

BPWM system clock frequency can be set equal or double to HCLK frequency as Figure 6.8-2, the detail register setting, please refer to Table 6.8-1.

Each BPWM generator has only one clock source input and can be selected from system clock or four TIMER trigger BPWM outputs as Figure 6.8-3 by ECLKSRC0 (BPWM_CLKSRC[2:0]) for BPWM_CLK0.

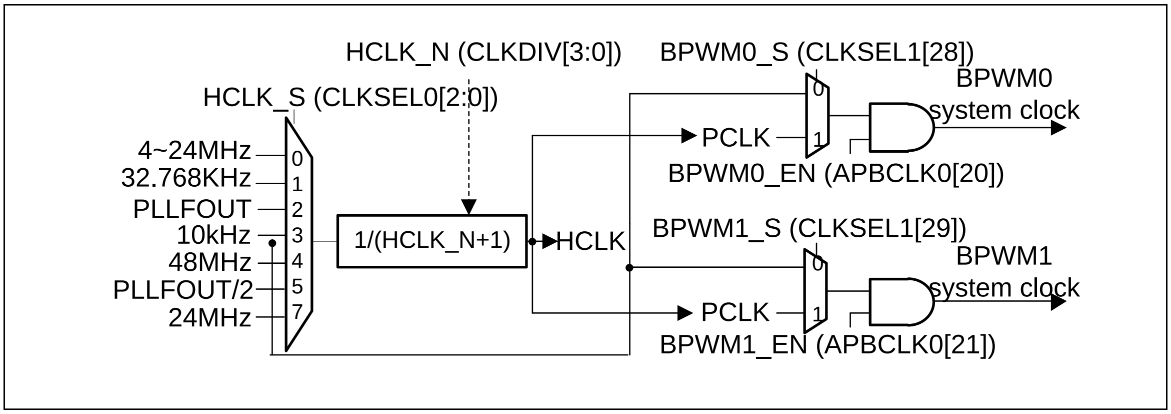


Figure 6.8-2 BPWM System Clock Source Control

BPWM System Clock/HCLK Frequency Ratio	HCLK_S (CLK_CLKSEL0[2:0])	HCLK_N (CLK_CLKDIV0[3:0])	BPWMn_S (CLK_CLKSEL3[X]), (N, X) Denotes (0, 18) Or (1, 19)
1/1	Don’t care	Don’t care	1
2/1	2	1	0

Table 6.8-1 BPWM System Clock Source Control Registers Setting Table

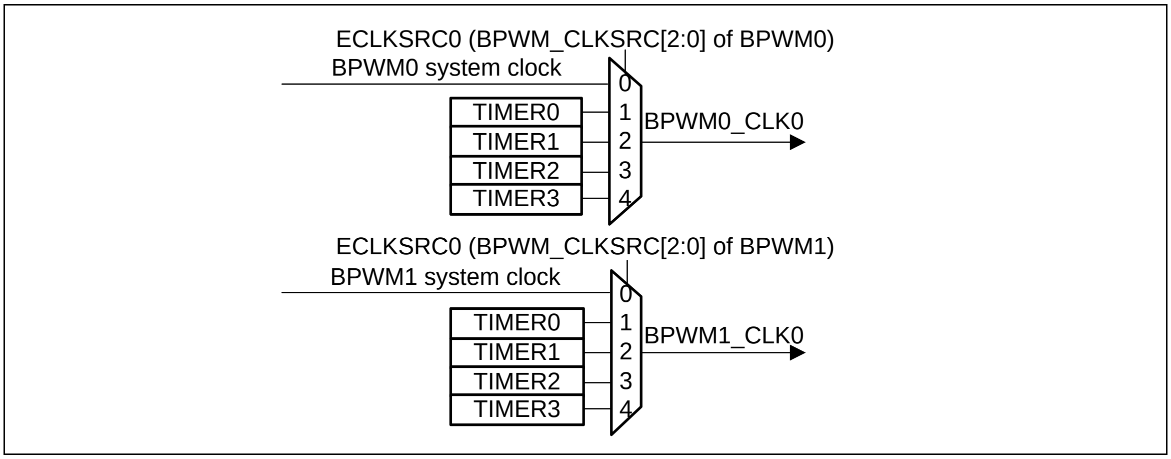


Figure 6.8-3 BPWM Clock Source Control

Figure 6.8-4 illustrates the architecture of BPWM Independent mode. All six channels share the same counter. When the counter counts to 0, PERIOD (BPWM_PERIODn[15:0]) or equal to comparator, events will be generated. These events are passed to the corresponding generators to generate BPWM pulse, interrupt signal and trigger signal for ADC to start conversion. Output control is used to changing BPWM pulse output state.

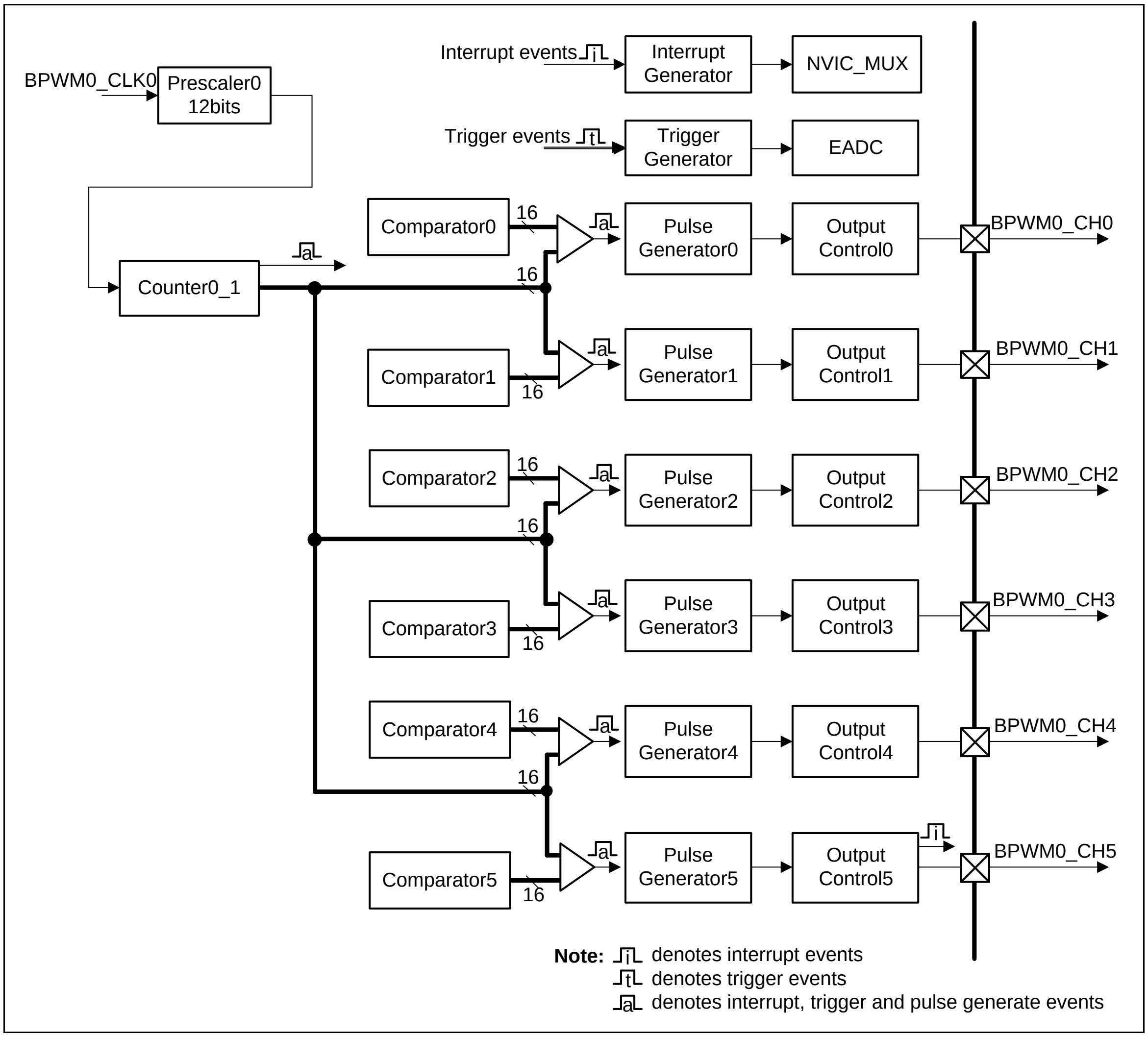


Figure 6.8-4 BPWM Independent Mode Architecture Diagram

6.8.4 Basic Configuration

The BPWM pin function is configured in GPB_MFP, GPC_MFP and GPD_MFP registers.

The BPWM clock can be enabled in CLK_APBCLK0[21:20]. The BPWM clock source is selected by CLK_CLKSEL1[29:28].

6.8.5 Functional Description

6.8.5.1 BPWM Prescaler

The BPWM prescaler is used to divide clock source, prescaler counting CLKPSC +1 times, and BPWM counter only count once. CLKPSC (Clock Pre-scale Register) is set by CLKPSC (BPWM_CLKPSCn[11:0], n denotes 0, 2, 4). Figure 6.8-5 shows an example of BPWM channel0 CLKPSC waveform.

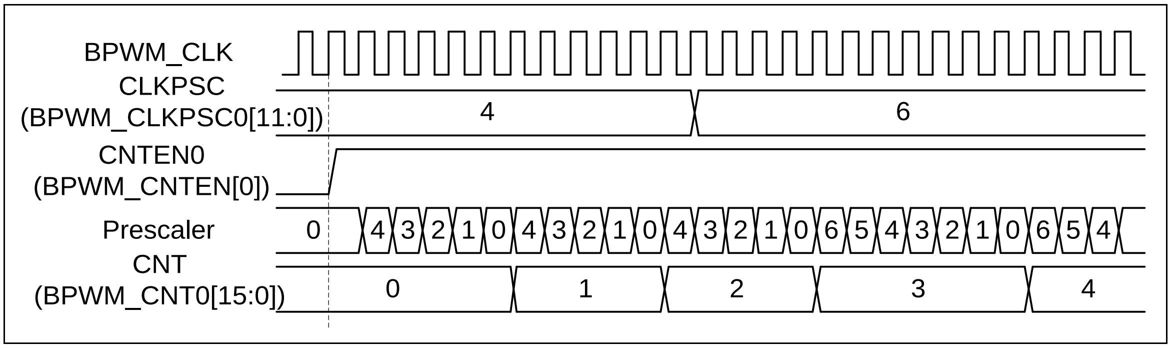


Figure 6.8-5 BPWM_CH0 CLKPSC waveform

6.8.5.2 BPWM Counter

BPWM supports 3 counter types operation: Up Counter, Down Counter and Up-Down Counter types.

6.8.5.3 Up Counter Type

In the up counter operation, the 16 bits BPWM counter is an up counter and starts up-counting from zero to PERIOD (BPWM_PERIODn[15:0], where n denotes channel number) to finish a BPWM period. The current counter value can be found by reading the CNT (BPWM_CNTn[15:0]). BPWM generates zero point event when counter counts to 0 and generates period point event when counting to PERIOD. Figure 6.8-6 shows an example of up counter, wherein BPWM period time = (PERIOD+1) * BPWM clock time.

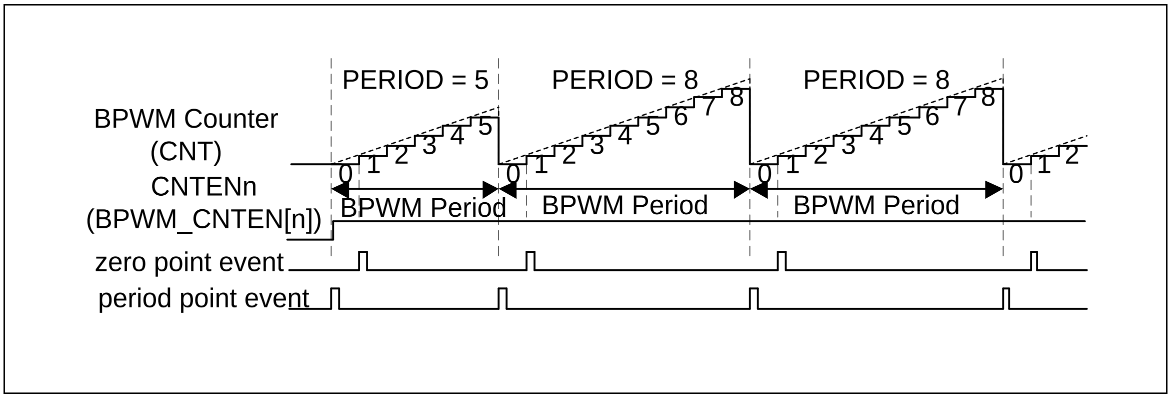


Figure 6.8-6 BPWM Up Counter Type

6.8.5.4 Down Counter Type

In the down counter operation, the 16 bits BPWM counter is a down counter and starts down-counting from PERIOD to zero to finish a BPWM period. The current counter value can be found by reading the CNT (BPWM_CNTn[15:0]). BPWM generates zero point event when counter counts to 0 and generates period point event when counting to PERIOD. Figure 6.8-7 shows an example of down counter, wherein BPWM period time = (PERIOD+1) * BPWM clock time.

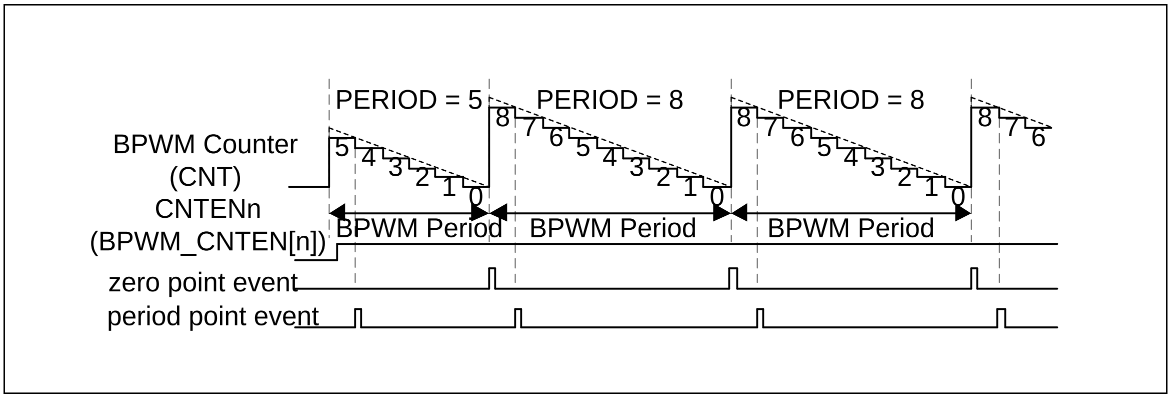


Figure 6.8-7 BPWM Down Counter Type

6.8.5.5 *Up-Down Counter Type*

In the up-down counter operation, the 16 bits BPWM counter is an up-down counter and starts counting-up from zero to PERIOD and then starts counting down to zero to finish a BPWM period. The current counter value can be found by reading the CNT (BPWM_CNTn[15:0]). BPWM generates zero point event when counter counts to 0 and generates center point event when counting to PERIOD. Figure 6.8-8 shows an example of up-down counter, wherein BPWM period time = (2xPERIOD) * BPWM clock time. The DIRF (BPWM_CNTn[16]) is counter direction indicator flag, where high is up counting, and low is down counting.

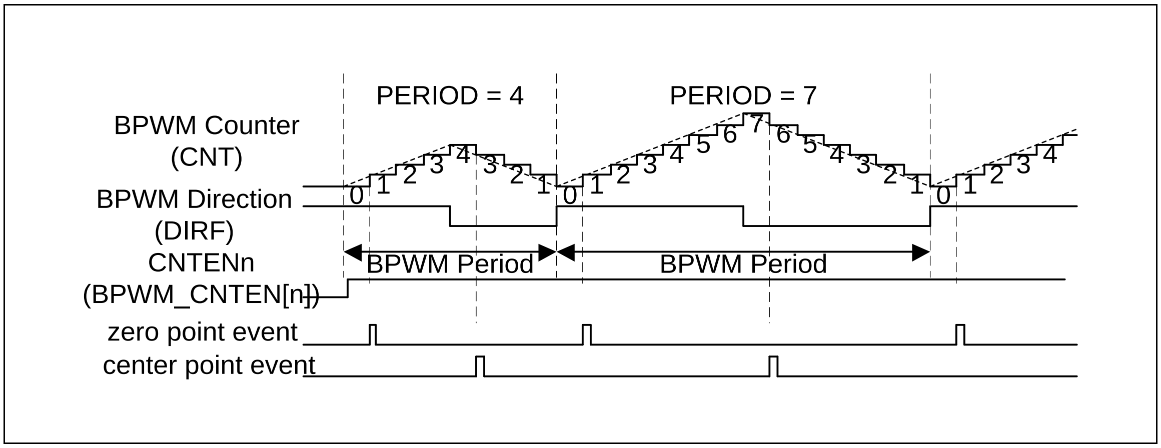


Figure 6.8-8 BPWM Up-Down Counter Type

6.8.5.6 *BPWM Comparator*

The CMPDAT (BPWM_CMPDATn[15:0]) is a basic comparator register of BPWM channel n; each channel only has one CMPDAT. The CMPDAT’s value is continuously compared to the counter value. When the counter is equal to compared register, BPWM generates an event and uses the event to generate BPWM pulse, interrupt or use to trigger ADC. In up-down counter type, two events will be generated in a BPWM period as shown in Figure 6.8-9.

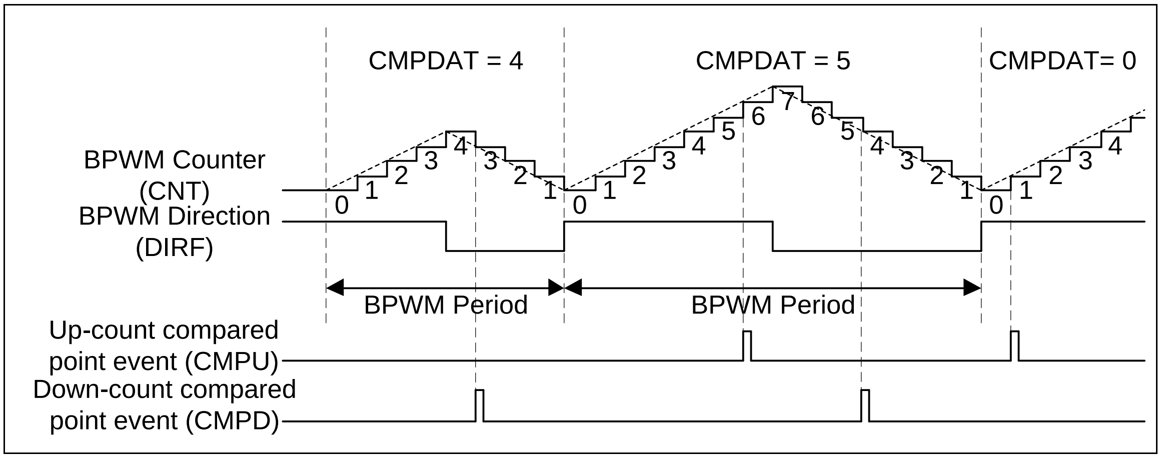


Figure 6.8-9 BPWM CMPDAT Events in Up-Down Counter Type

6.8.5.7 BPWM Double Buffering

The double buffering uses double buffers to separate software writing and hardware operation timing. After registers are modified through software, hardware will load register value to the buffer register according to the loading mode timing. The hardware action is based on the buffer value. This can prevent asynchronously operation problem due to software and hardware asynchronism.

The BPWM has double buffering function for PERIOD and CMPDAT. The concept of double buffering is used in loading modes, which are described in the following sections. For example, as shown in Figure 6.8-10, in period loading mode, writing PERIOD and CMPDAT through software, BPWM will load new values to their buffer PBUF (BPWM_PBUFn[15:0]) and CMPBUF (BPWM_CMPBUFn[15:0]) at start of the next period without affecting the current period counter operation. There are 3 loading modes for loading value to buffer: period loading mode, immediately loading mode and center loading mode.

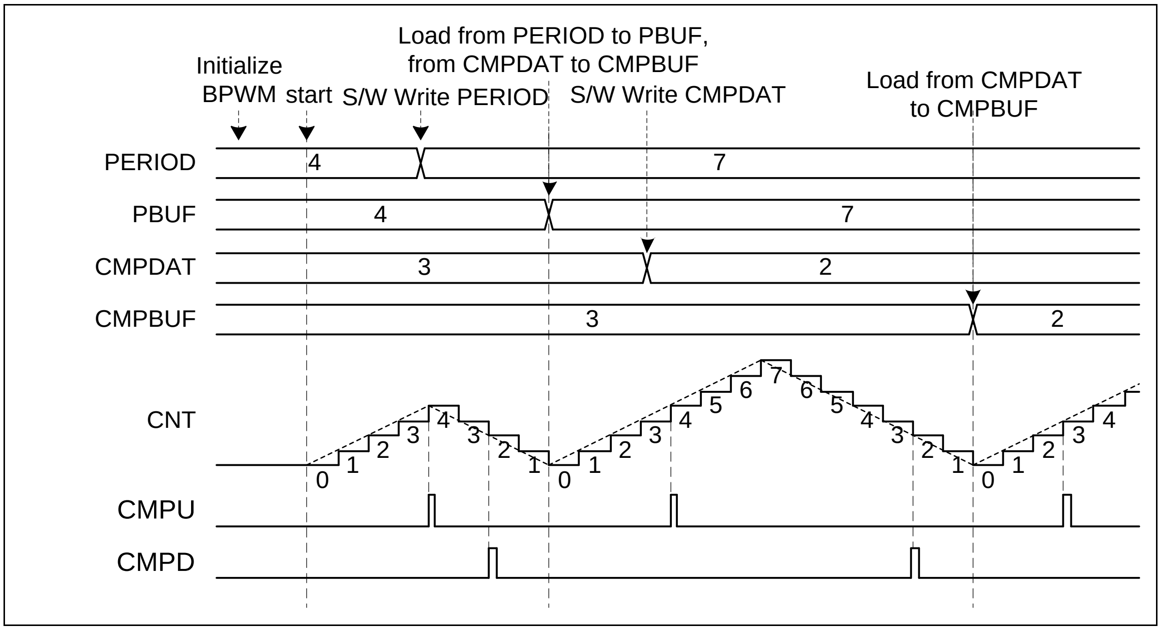


Figure 6.8-10 BPWM Double Buffering Illustration

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

6.8.5.8 Period Loading Mode

Period Loading mode is the default loading mode. It has lowest priority in loading modes. PERIOD and CMPDAT both will both load to their buffer while a period is completed. For example, after BPWM counter up counts from zero to PERIOD in up-counter operation or down counts from PERIOD to zero in the down-counter operation or up counts from zero to PERIOD and then down counts to zero in up-down counter operation.

Figure 6.8-11 shows period loading timing of up-count operation, where PERIOD DATA0 denotes the initial data of PERIOD, PERIOD DATA1 denotes the first updated PERIOD data by software and so on, CMPDAT also follows this rule. The following describes steps sequence of Figure 6.8-11. User can know the PERIOD and CMPDAT update condition, by watching PWM period and CMPU event.

1. Software writes CMPDAT DATA1 to CMPDAT at point 1.
2. Hardware loads CMPDAT DATA1 to CMPBUF at the end of PWM period at point 2.
3. Software writes PERIOD DATA1 to PERIOD at point 3.
4. Hardware loads PERIOD DATA1 to PBUF at the end of PWM period at point 4.
5. Software writes PERIOD DATA2 to PERIOD at point 5.
6. Hardware loads PERIOD DATA2 to PBUF at the end of PWM period at point 6.

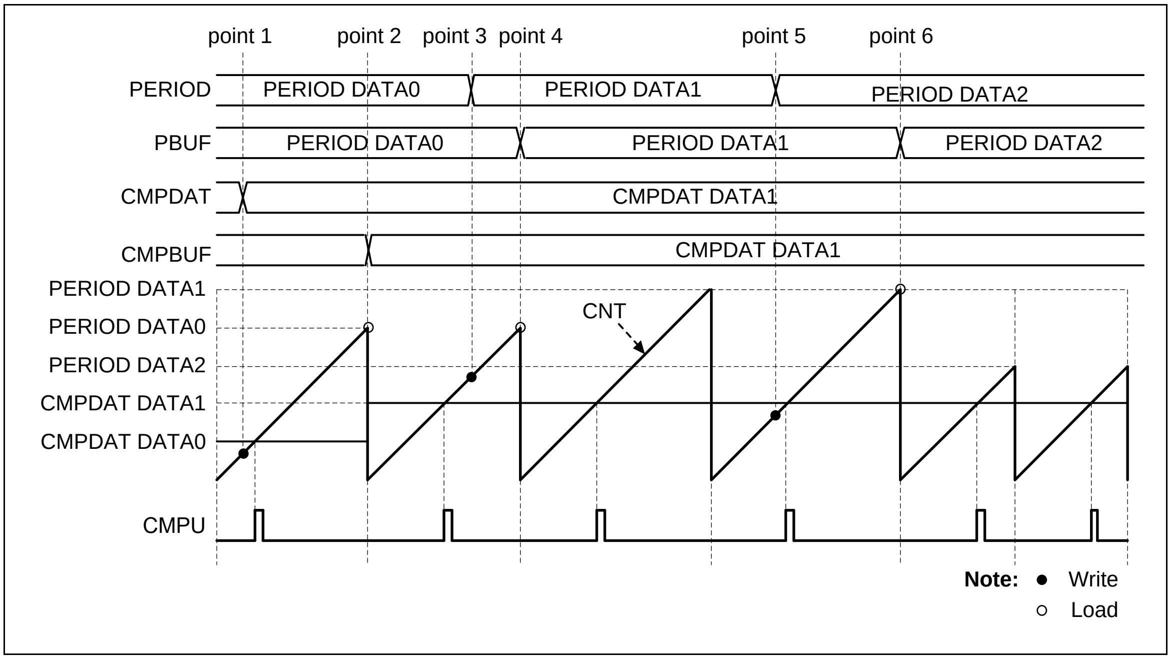


Figure 6.8-11 Period Loading Mode with Up-Counter Type

6.8.5.9 Immediately Loading Mode

If the IMMLDENn (BPWM_CTL0[21:16]) bit which corresponds to BPWM channel n is set to 1, software will load a value to buffer from PERIOD and CMPDAT immediately while software updates PERIOD or CMPDAT. If the updated PERIOD value is less than current counter value, counter will count wraparound. Immediately loading mode has the highest priority. If IMMLDENn has been set, other loading mode for channel n will become invalid. Figure 6.8-12 shows an example and its steps sequence is described below.

1. Software writes CMPDAT DATA1 and hardware immediately loading CMPDAT DATA1 to CMPBUF at point 1.

2. Software writes PERIOD DATA1 which is greater than current counter value at point 2; counter will continue counting until equal to PERIOD DATA1 to finish a period loading.
3. Software writes PERIOD DATA2 which is less than the current counter value at point 3; counter will continue counting to its maximum value 0xFFFF and count wraparound from 0 to PERIOD DATA2 to finish this period loading.

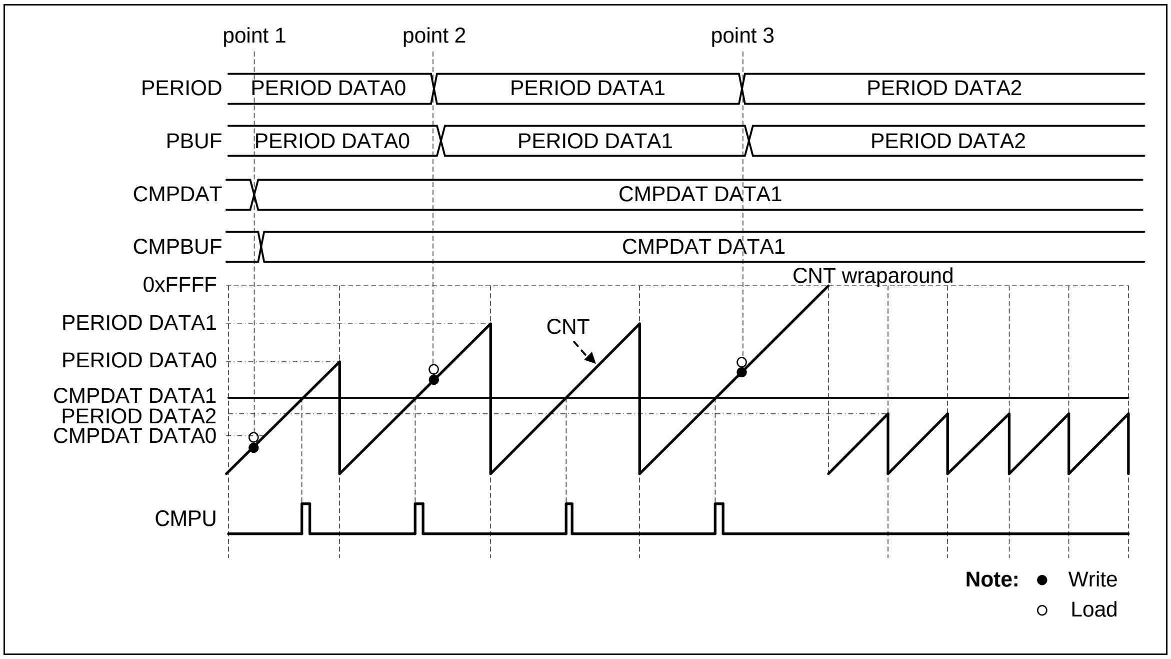


Figure 6.8-12 Immediately Loading Mode with Up-Counter Type

6.8.5.10 Center Loading Mode

If the CTRLDn (BPWM_CTL0[5:0]) bit which corresponds to BPWM channel n is set to 1 and in up-down counter type, CMPDAT will load to CMPBUFn in center of a period, that is, counter counts to PERIOD. PERIOD loading timing is the same as period loading mode. Figure 6.8-13 shows an example and its steps sequence is described below.

1. Software writes CMPDAT DATA1 at point 1.
2. Hardware loads CMPDAT DATA1 to CMPBUF at center of BPWM period at point 2.
3. Software writes PERIOD DATA1 at point 3.
4. Hardware loads PERIOD DATA1 to PBUF at the end of BPWM period at point 4.
5. Software writes CMPDAT DATA2 at point 5.
6. Hardware loads CMPDAT DATA2 to CMPBUF at center of BPWM period at point 6.
7. Software writes PERIOD DATA2 at point 7.
8. Hardware loads PERIOD DATA2 to PBUF at the end of BPWM period at point 8.

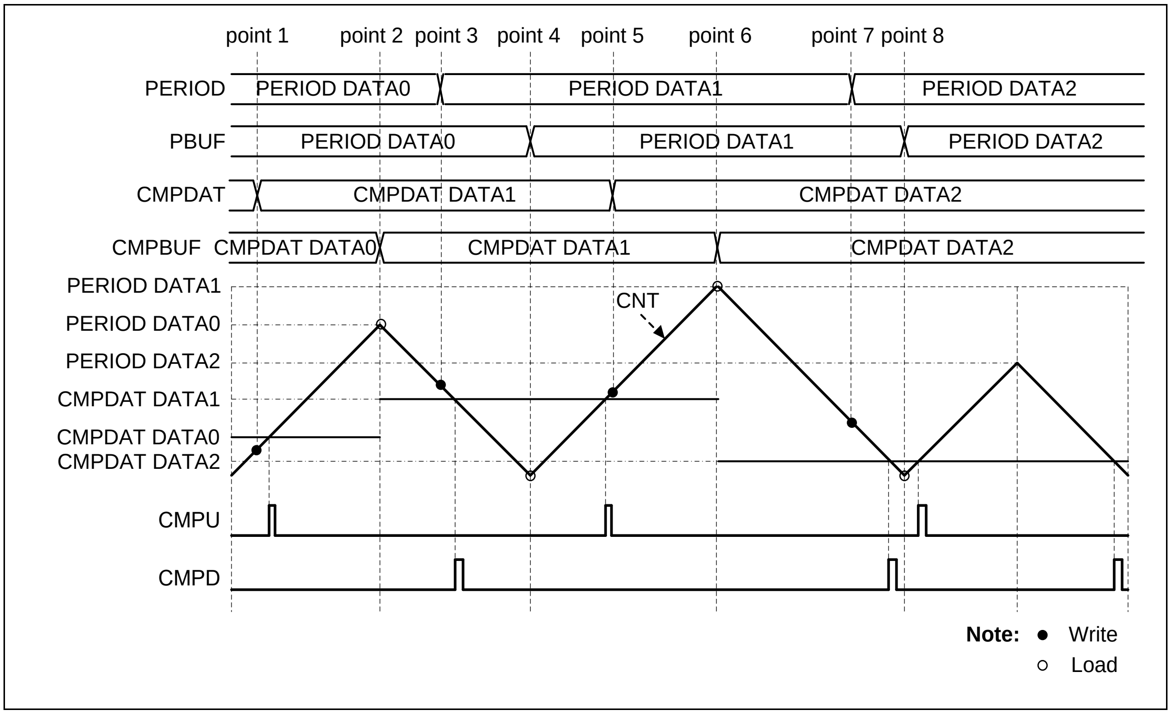


Figure 6.8-13 Center Loading Mode with Up-Down-Counter Type

6.8.5.11 BPWM Pulse Generator

BPWM pulse generator uses counter and comparator events to generate BPWM pulse. The events are: zero point, period point in up counter type and down counter type, center point in up-down counter type and counter equal to comparator point in three types. As to up-down counter type, there are two counter equal comparator points, one at up count another at down count.

Each event point can decide BPWM waveform to do nothing (X), set Low (L), set High (H) or toggle (T) by setting BPWM_WGCTL0 and BPWM_WGCTL1 registers. Using these points can easily generate asymmetric BPWM pulse or variant waveform as shown in Figure 6.8-14. In the figure, there is a comparator n to generate BPWM pulse. n denotes channel number 0 to 5. CMPU denotes CNT is equal to CMPDAT when counting up, CMPD denotes CNT is equal to CMPDAT when counting down.

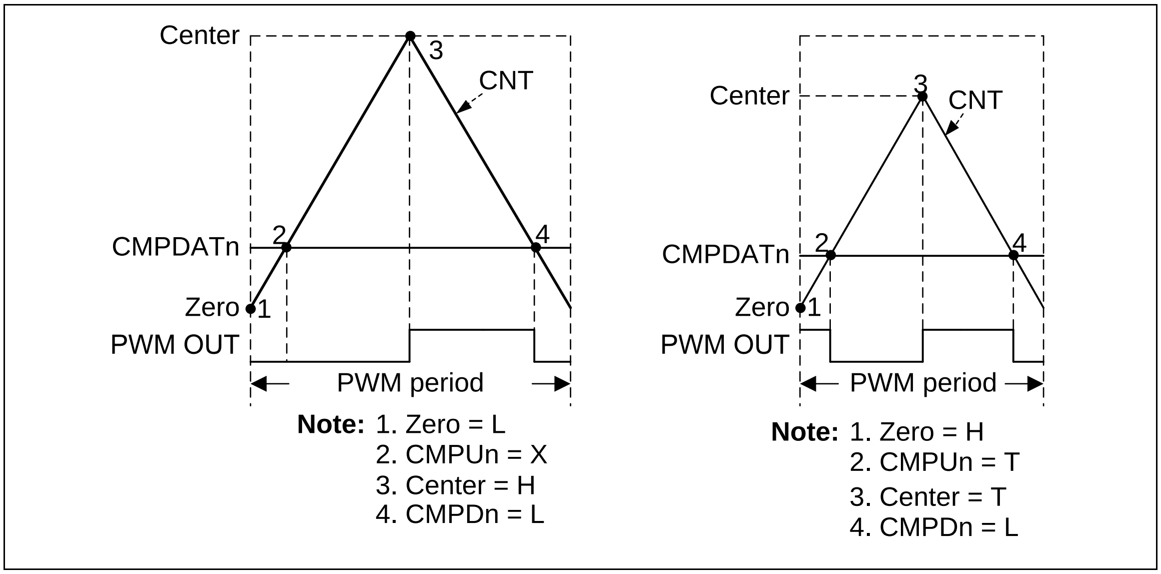


Figure 6.8-14 BPWM Pulse Generation

The generation events may be sometimes set to the same value, as the reason, events priority between different counter types are list below, up counter type (Table 6.8-2), down counter type (Table 6.8-3) and up-down counter type (Table 6.8-4).By using event priority, user can easily generate 0% to 100% duty pulse as shown in Figure 6.8-15.

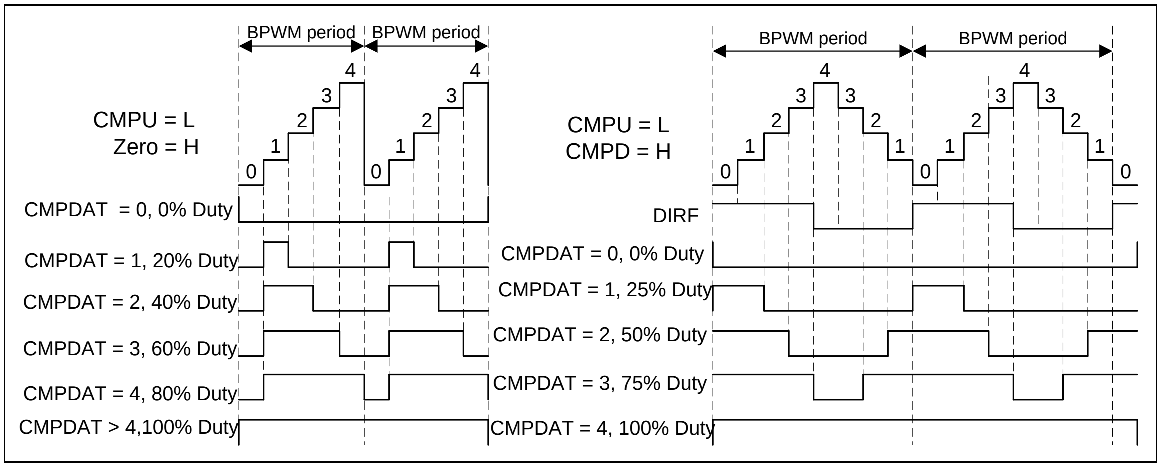


Figure 6.8-15 BPWM 0% to 100% Pulse Generation

Priority	Up Event
1 (Highest)	CNT = period (PERIOD)
2	CNT = CMPUn, n = 0~5
3 (Lowest)	CNT = zero

Table 6.8-2 BPWM Pulse Generation Event Priority for Up-Counter

Priority	Down Event
1 (Highest)	CNT = zero
2	CNT = CMPDn, n = 0~5
3 (Lowest)	CNT = period (PERIOD)

Table 6.8-3 BPWM Pulse Generation Event Priority for Down-Counter

Priority	Up Event	Down Event
1 (Highest)	CNT= CMPUn, n = 0~5	CNT = CMPDn, n = 0~5
2	CNT = zero	CNT = center (PERIOD)
3 (Lowest)	PERIOD = CMPDn, n = 0~5	CNT = CMPUn, n = 0~5

Table 6.8-4 BPWM Pulse Generation Event Priority for Up-Down-Counter

6.8.5.12 BPWM Output Control

After BPWM pulse generation, there are three steps to control the output of BPWM channels. There are Mask, Pin Polarity and Output Enable three steps as shown in Figure 6.8-16.

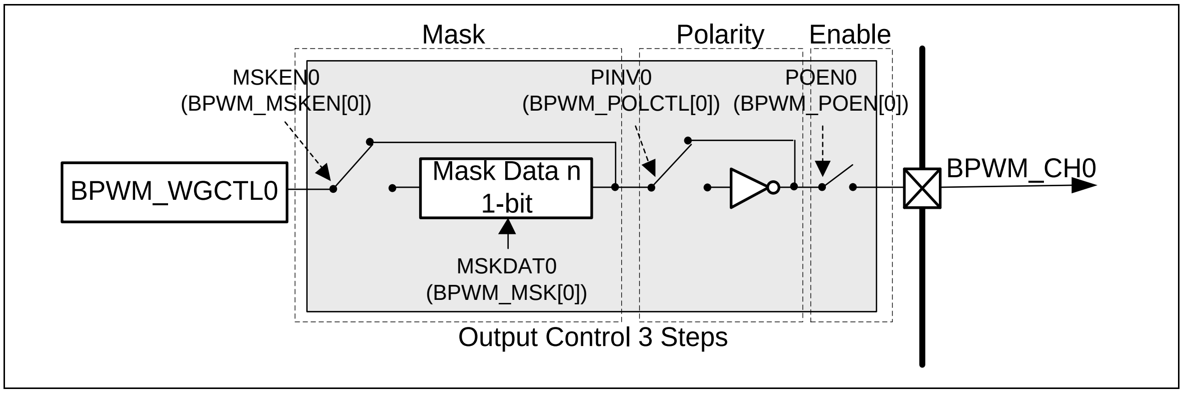


Figure 6.8-16 BPWM_CH0 Output Control 3 Steps

6.8.5.13 BPWM Mask Output Function

Each of the BPWM output channels can be manually overridden by using the appropriate bits in the BPWM Mask Enable Control Register (BPWM_MSKEN) and BPWM Masked Data Register (BPWM_MSK) to drive the BPWM channel outputs to specified logic states independent of the duty cycle comparison units. The BPWM mask bits are useful when controlling various types of Electrically Commutated Motor (ECM) like a BLDC motor. The BPWM_MSKEN register contains six bits, MSKENn(BPWM_MSKEN[5:0]) determine which BPWM channel output will be overridden, MSKENn(BPWM_MSKEN[5:0]) bits are active-high. The BPWM_MSK register contains six bits, MSKDATn(BPWM_MSK[5:0]), which determine the state of the BPWM channel output when the channel is masked via the MSKDAT bits. Figure 6.8-17 shows an example of how BPWM mask control can be used for the override feature.

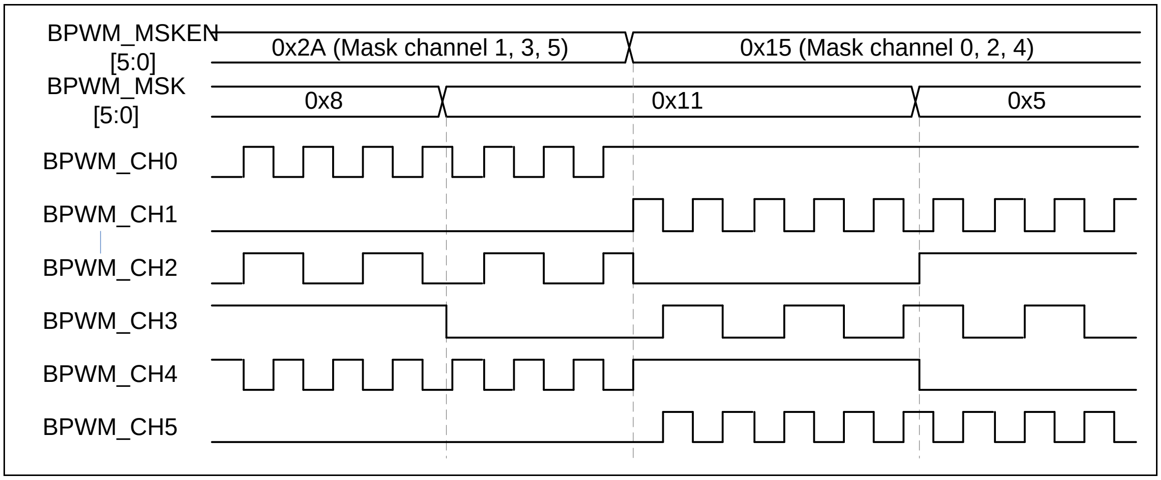


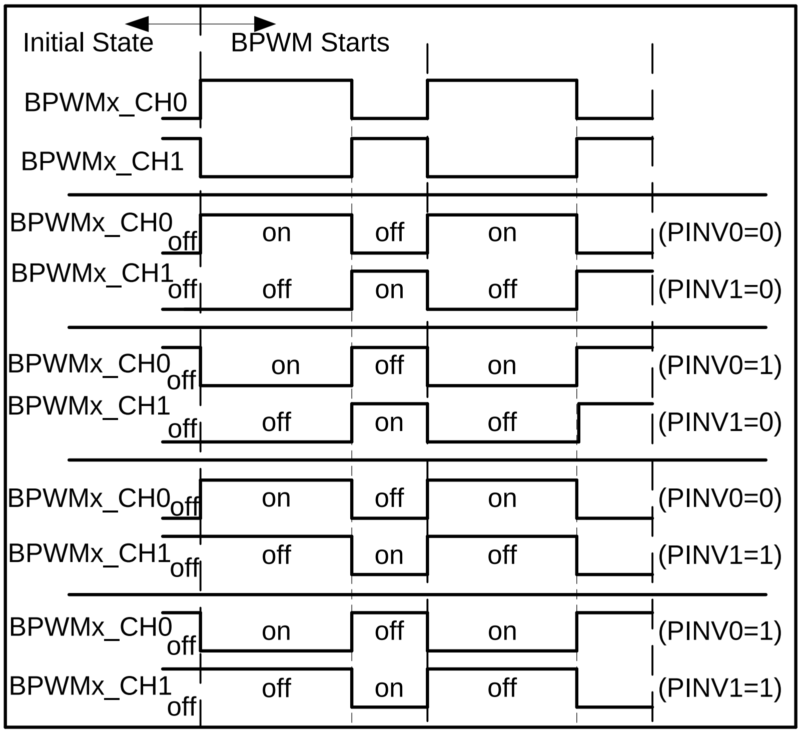
Figure 6.8-17 Illustration of Mask Control Waveform

6.8.5.14 Polarity Control

Each BPWM port from BPWM_CH0 to BPWM_CH5 has an independent polarity control module to configure the polarity of the active state of BPWM output. By default, the BPWM output is active high. This implies the BPWM OFF state is low and ON state is high. This definition is variable through setting BPWM Pin Polarity Inverse Control Register (BPWM_POLCTL), for each individual BPWM channel. Figure 6.8-18 shows the initial state before BPWM starts with different polarity settings.

6.8.5.15 Synchronous start function

The synchronous start function can be enabled when SSEN0 (BPWM_SSCTL[0]) is set. User can select synchronous source which is from BPWM0, BPWM1, PWM0, or PWM1 by SSRC (BPWM_SSCTL[9:8]). The selected BPWM or PWM channels (include channel0 to cheannel5 of each BPWM or PWM) will start counting at the same time once the synchronous start function is enabled and set CNTSEN (BPWM_SSTRG). It is noted that set CNTSEN (BPWM_SSTRG) will also set the counter enable bit (CNTENn, n denotes channel 0 to 5) to start counting.



Note: PINVx: Polarity inverse control bits; It controls the BPWM output initial state and polarity, x denotes 0 or 1.

Figure 6.8-18 Initial State and Polarity Control

6.8.5.16 BPWM Interrupt Generator

There are two independent interrupts for each BPWM as shown in Figure 6.8-19.

BPWM interrupt (BPWM_INT) comes from BPWM complementary pair events. The counter can generate the Zero point Interrupt Flag ZIF0 (BPWM_INTSTS0[0]) and the Period point Interrupt Flag PIF0 (BPWM_INTSTS0[8]). When BPWM channel n’s counter equals to the comparator value stored in BPWM_CMPDATn, the different interrupt flags will be triggered depending on the counting direction. If the matching occurs at up-count direction, the Up Interrupt Flag CMPUIFn (BPWM_INTSTS0[21:16]) is set and if matching at the opposite direction, the Down Interrupt Flag CMPDIFn (BPWM_INTSTS0[29:24]) is set. If the correspond interrupt enable bits are set, the trigger events will generates interrupt signals.

Another interrupt is the capture interrupt (CAP_INT). It shares the BPWM_INT vector in NVIC, CAP_INT can be generated when the CRLIFn (BPWM_CAPIF[5:0]) is triggered and the Capture Rising Interrupt Enable bit CAPRIENn (BPWM_CAPIEN[5:0]) is set to 1. Or in the falling edge condition, the CFLIFn (BPWM_CAPIF[13:8]) can be triggered when the Capture Falling Interrupt Enable bit CAPFIENn (BPWM_CAPIEN[13:8]) is set to 1.

Figure 6.8-19 demonstrates the architecture of the BPWM interrupts.

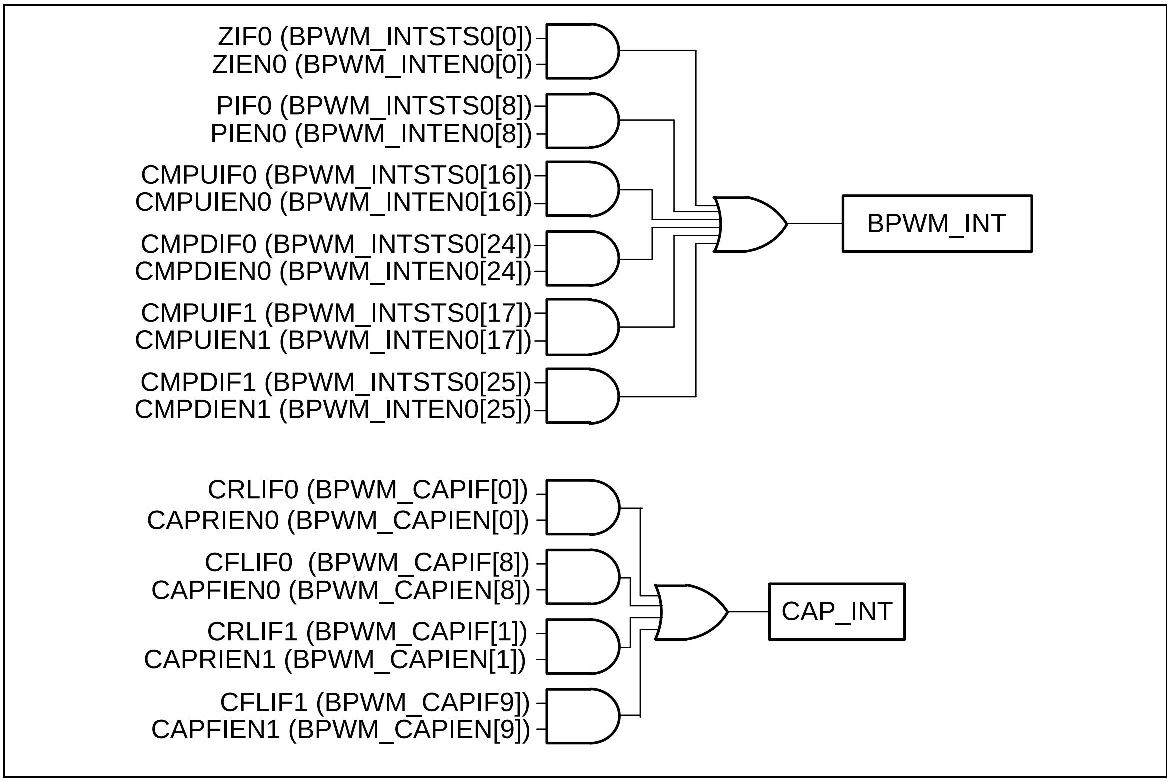

ZIF0 (BPWM_INTSTS0[0])
ZIEN0 (BPWM_INTEN0[0])
PIF0 (BPWM_INTSTS0[8])
PIEN0 (BPWM_INTEN0[8])
CMPUIF0 (BPWM_INTSTS0[16])
CMPUIEN0 (BPWM_INTEN0[16])
CMPDIF0 (BPWM_INTSTS0[24])
CMPDIEN0 (BPWM_INTEN0[24])
CMPUIF1 (BPWM_INTSTS0[17])
CMPUIEN1 (BPWM_INTEN0[17])
CMPDIF1 (BPWM_INTSTS0[25])
CMPDIEN1 (BPWM_INTEN0[25])
BPWM_INT
CRLIF0 (BPWM_CAPIF[0])
CAPRIEN0 (BPWM_CAPIEN[0])
CFLIF0 (BPWM_CAPIF[8])
CAPFIEN0 (BPWM_CAPIEN[8])
CRLIF1 (BPWM_CAPIF[1])
CAPRIEN1 (BPWM_CAPIEN[1])
CFLIF1 (BPWM_CAPIF9])
CAPFIEN1 (BPWM_CAPIEN[9])
CAP_INT

Figure 6.8-19 BPWM_CH0 and BPWM_CH1 Pair Interrupt Architecture Diagram

6.8.5.17 BPWM Trigger ADC Generator

BPWM can be one of the ADC conversion trigger source. Each BPWM pair channels share the same trigger source. Setting TRGSELn is to select the trigger sources, where TRGSELn is TRGSEL0, TRGSEL1, …, and TRGSEL5, which are located in BPWM_ADCTS0[3:0], BPWM_ADCTS0[11:8], BPWM_ADCTS0[19:16], BPWM_ADCTS0[27:24], BPWM_ADCTS1[3:0] and BPWM_ADCTS1[11:8], respectively. Setting TRGENn is to enable the trigger output to ADC, where TRGENn is TRGEN0, TRGEN1, …, TRGEN5, which are located in BPWM_ADCTS0[7], BPWM_ADCTS0[15], BPWM_ADCTS0[23], BPWM_ADCTS0[31], BPWM_ADCTS1[7] and BPWM_ADCTS1[15], respectively. The number n (n = 0, 1, .., 5) denotes BPWM channel number.

There are 7 BPWM events that can be selected as the trigger source for one pair of channels. Figure 6.8-20 is an example of BPWM_CH0 and BPWM_CH1. BPWM can trigger ADC to start conversion in different timings by setting PERIOD and CMPDAT. Figure 6.8-21 is the trigger ADC timing waveform in the up-down counter type.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

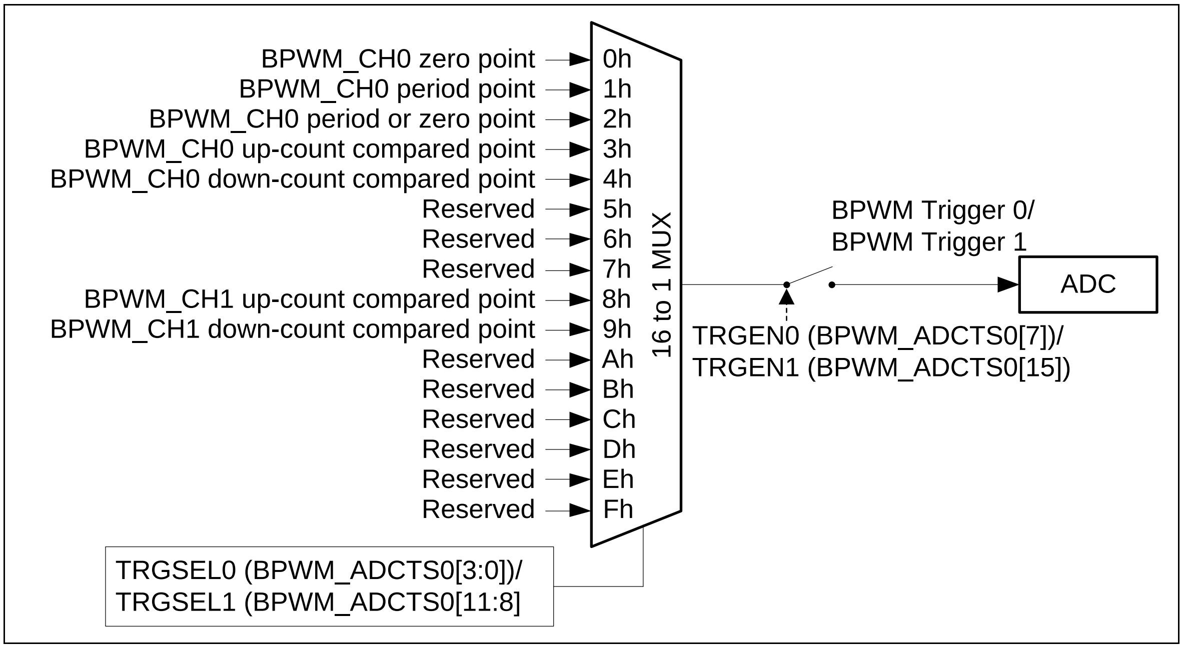


Figure 6.8-20 BPWM_CH0 and BPWM_CH1 Pair Trigger ADC Block Diagram

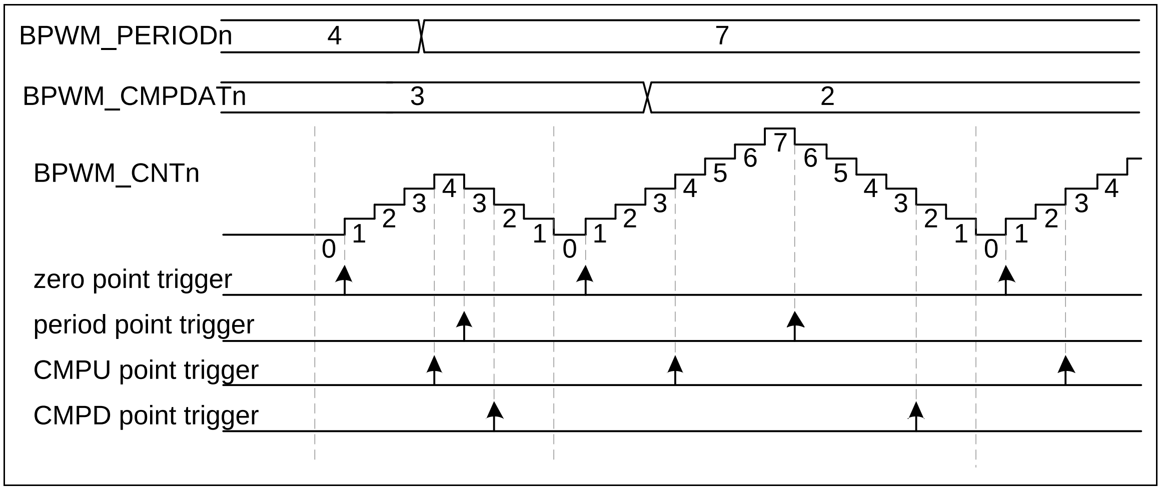


Figure 6.8-21 BPWM Trigger ADC in Up-Down Counter Type Timing Waveform

6.8.5.18 *Capture Operation*

The channels of the capture input and the BPWM output share the same pin and counter. The counter can operate in up or down counter type. The capture function will always latch the BPWM counter to the register RCAPDATn (BPWM_RCAPDATn[15:0]) or the register FCAPDATn (BPWM_FCAPDATn[15:0]) if the input channel has a rising transition or a falling transition, respectively. The capture function will also generate an interrupt CAP_INT (using BPWM_INT vector) if the rising or falling latch occurs and the corresponding channel n's rising or falling interrupt enable bits are set, where the CAPRIENn (BPWM_CAPIEN[5:0]) is for the rising edge and the CAPFIENn (BPWM_CAPIEN[13:8]) is for the falling edge. When rising or falling latch occurs, the corresponding BPWM counter may be reloaded with the value BPWM_PERIODn, depending on the setting of RCRLDENn or FCRLDENn (where RCRLDENn and FCRLDENn are located at BPWM_CAPCTL[21:16] and BPWM_CAPCTL[29:24], respectively). Note that the corresponding GPIO pins must be configured as the capture function by enable the CAPINENn

(BPWM_CAPINEN[5:0]) for the corresponding capture channel n. Figure 6.8-22 is the capture block diagram of channel 0.

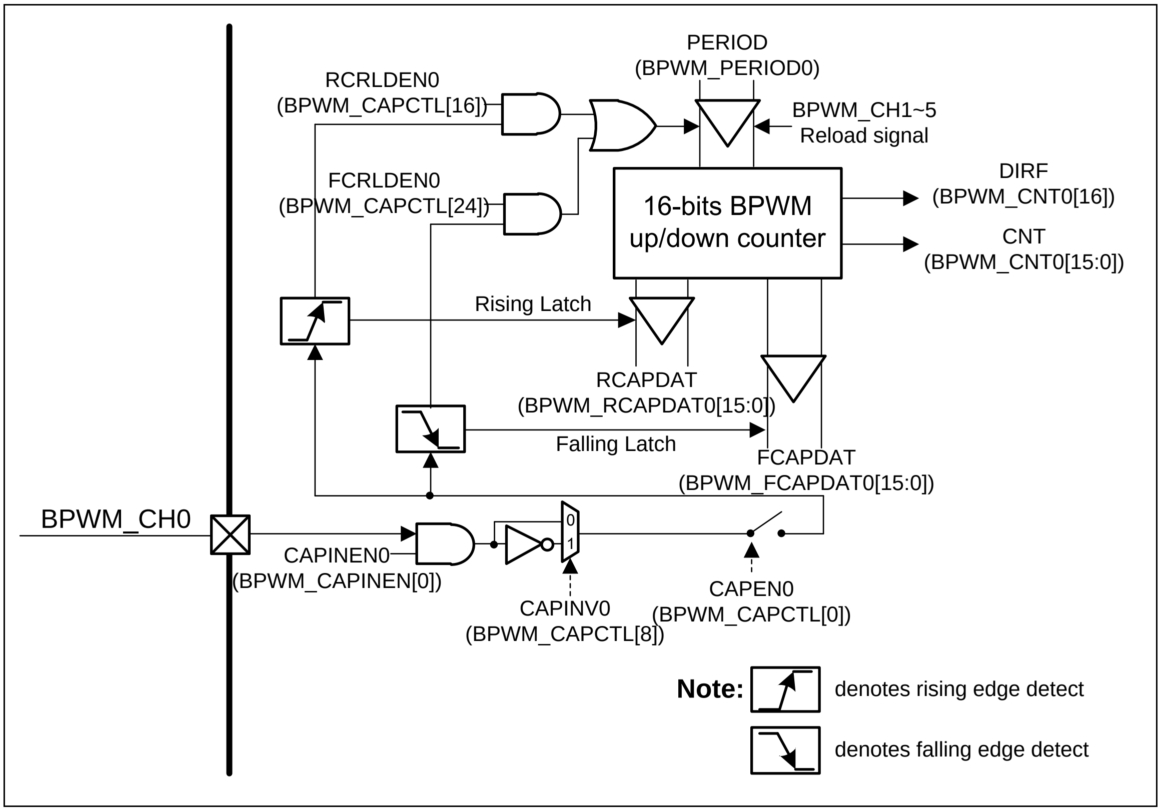


Figure 6.8-22 BPWM_CH0 Capture Block Diagram

Figure 6.8-23 illustrates the capture function timing. In this case, the capture counter is set as BPWM down counter type and the PERIOD is set to 8 so that the counter counts in the down direction, from 8 to 0. When detecting a falling edge at the capture input pin, the capture function latches counter value to the BPWM_FCAPDATn. When detecting the rising edge, it latches the counter value to the BPWM_RCAPDATn. In this timing diagram, when the falling edge is detected at the first time, the capture function will reload the counter value from the PERIOD setting because the FCRLDENn is enabled. But at the second time, the falling edge does not result in a reload because of the disabled FCRLDENn. In this example, the counter also reloads at the rising edge of the capture input because the RCRLDENn is enabled.

Moreover, if the case is setup as the up counter type, the counter will reload the value zero and count up to the value PERIOD. It is important that the counter is shared by all channels, and thus the counter reloads time also controlled by all channels' reload signals.

Figure 6.8-23 also illustrates the timing example for the interrupt and interrupt flag generation. When the rising edge at channel n is detected, the corresponding bit CRLIFn (BPWM_CAPIF[5:0]) is set by hardware. Similarly, a falling edge detection at channel n causes the corresponding bit CFLIFn (BPWM_CAPIF[13:8]) set by hardware. CRLIFn and CFLIFn can be cleared by software by writing '1'. If the CRLIFn is set and the CAPRIENn is enabled, the capture function generates an interrupt. If the CFLIFn is set and the CAPFIENn is enabled, the interrupt also happens.

A condition which is not shown in this figure is: if the rising latch happens again when the CRLIF is already set, the Overrun status CRIFOVn (BPWM_CAPSTS[5:0]) will be set to 1 by hardware to indicate the CRLIF overrunning. Also, if the falling latch happens again, the same hardware operation occurs for the interrupt flag CFLIF and the Overrun status CFIFOVn (BPWM_CAPSTS[13:8]).

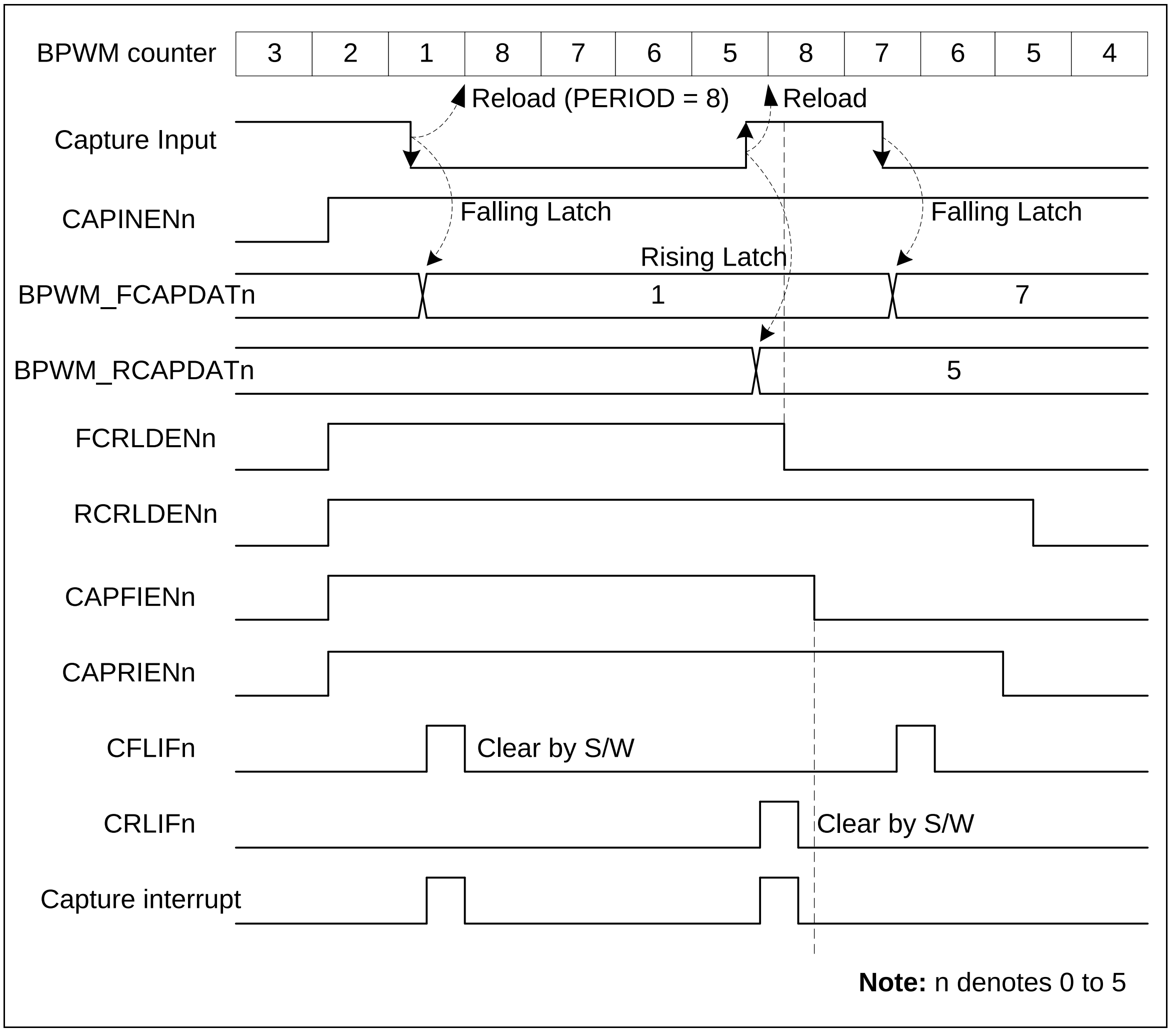


Figure 6.8-23 Capture Operation Waveform

The capture pulse width can be calculated according to the following formula:

For the negative pulse case, the channel low pulse width is calculated as (BPWM_PERIODn + 1 - BPWM_RCAPDATn). In Figure 6.8-23 case, low pulse width is 8+1-5 = 4.

For the positive pulse case, the channel high pulse width is calculated as (BPWM_PERIODn + 1 - BPWM_FCAPDATn). In Figure 6.8-23 case, high pulse width is 8+1-7 = 2.

6.8.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
BPWM Base Address: **BPWM0_BA = 0x4004_0000** **BPWM1_BA = 0x4014_0000**				
BPWM_CTL0 **x=0, 1**	BPWMx_BA+0x00	R/W	BPWM Control Register 0	0x0000_0000
BPWM_CTL1 **x=0, 1**	BPWMx_BA+0x04	R/W	BPWM Control Register 1	0x0000_0000
BPWM_CLKSRC **x=0, 1**	BPWMx_BA+0x10	R/W	BPWM Clock Source Register	0x0000_0000
BPWM_CLKPSC **x=0, 1**	BPWMx_BA+0x14	R/W	BPWM Clock Pre-scale Register	0x0000_0000
BPWM_CNTEN **x=0, 1**	BPWMx_BA+0x20	R/W	BPWM Counter Enable Register	0x0000_0000
BPWM_CNTCLR **x=0, 1**	BPWMx_BA+0x24	R/W	BPWM Clear Counter Register	0x0000_0000
BPWM_PERIOD **x=0, 1**	BPWMx_BA+0x30	R/W	BPWM Period Register	0x0000_0000
BPWM_CMPDAT0 **x=0, 1**	BPWMx_BA+0x50	R/W	BPWM Comparator Register 0	0x0000_0000
BPWM_CMPDAT1 **x=0, 1**	BPWMx_BA+0x54	R/W	BPWM Comparator Register 1	0x0000_0000
BPWM_CMPDAT2 **x=0, 1**	BPWMx_BA+0x58	R/W	BPWM Comparator Register 2	0x0000_0000
BPWM_CMPDAT3 **x=0, 1**	BPWMx_BA+0x5C	R/W	BPWM Comparator Register 3	0x0000_0000
BPWM_CMPDAT4 **x=0, 1**	BPWMx_BA+0x60	R/W	BPWM Comparator Register 4	0x0000_0000
BPWM_CMPDAT5 **x=0, 1**	BPWMx_BA+0x64	R/W	BPWM Comparator Register 5	0x0000_0000
BPWM_CNT **x=0, 1**	BPWMx_BA+0x90	R	BPWM Counter Register 0	0x0000_0000
BPWM_WGCTL0 **x=0, 1**	BPWMx_BA+0xB0	R/W	BPWM Generation Register 0	0x0000_0000
BPWM_WGCTL1 **x=0, 1**	BPWMx_BA+0xB4	R/W	BPWM Generation Register 1	0x0000_0000
BPWM_MSKEN **x=0, 1**	BPWMx_BA+0xB8	R/W	BPWM Mask Enable Register	0x0000_0000

BPWM_MSK **x=0, 1**	BPWMx_BA+0xBC	R/W	BPWM Mask Data Register	0x0000_0000
BPWM_POLCTL **x=0, 1**	BPWMx_BA+0xD4	R/W	BPWM Pin Polar Inverse Register	0x0000_0000
BPWM_POEN **x=0, 1**	BPWMx_BA+0xD8	R/W	BPWM Output Enable Register	0x0000_0000
BPWM_INTEN **x=0, 1**	BPWMx_BA+0xE0	R/W	BPWM Interrupt Enable Register	0x0000_0000
BPWM_INTSTS0 **x=0, 1**	BPWMx_BA+0xE8	R/W	BPWM Interrupt Flag Register 0	0x0000_0000
BPWM_ADCTS0 **x=0, 1**	BPWMx_BA+0xF8	R/W	BPWM Trigger ADC Source Select Register 0	0x0000_0000
BPWM_ADCTS1 **x=0, 1**	BPWMx_BA+0xFC	R/W	BPWM Trigger ADC Source Select Register 1	0x0000_0000
BPWM_SSCTL **x=0, 1**	BPWMx_BA+0x110	R/W	BPWM Synchronous Start Control Register	0x0000_0000
BPWM_SSTRG **x=0, 1**	BPWMx_BA+0x114	W	BPWM Synchronous Start Trigger Register	0x0000_0000
BPWM_STATUS **x=0, 1**	BPWMx_BA+0x120	R/W	BPWM Status Register	0x0000_0000
BPWM_CAPINEN **x=0, 1**	BPWMx_BA+0x200	R/W	BPWM Capture Input Enable Register	0x0000_0000
BPWM_CAPCTL **x=0, 1**	BPWMx_BA+0x204	R/W	BPWM Capture Control Register	0x0000_0000
BPWM_CAPSTS **x=0, 1**	BPWMx_BA+0x208	R	BPWM Capture Status Register	0x0000_0000
BPWM_RCAPDAT0 **x=0, 1**	BPWMx_BA+0x20C	R	BPWM Rising Capture Data Register 0	0x0000_0000
BPWM_FCAPDAT0 **x=0, 1**	BPWMx_BA+0x210	R	BPWM Falling Capture Data Register 0	0x0000_0000
BPWM_RCAPDAT1 **x=0, 1**	BPWMx_BA+0x214	R	BPWM Rising Capture Data Register 1	0x0000_0000
BPWM_FCAPDAT1 **x=0, 1**	BPWMx_BA+0x218	R	BPWM Falling Capture Data Register 1	0x0000_0000
BPWM_RCAPDAT2 **x=0, 1**	BPWMx_BA+0x21C	R	BPWM Rising Capture Data Register 2	0x0000_0000
BPWM_FCAPDAT2 **x=0, 1**	BPWMx_BA+0x220	R	BPWM Falling Capture Data Register 2	0x0000_0000
BPWM_RCAPDAT3 **x=0, 1**	BPWMx_BA+0x224	R	BPWM Rising Capture Data Register 3	0x0000_0000
BPWM_FCAPDAT3	BPWMx_BA+0x228	R	BPWM Falling Capture Data Register 3	0x0000_0000

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

x=0, 1				
BPWM_RCAPDAT4 **x=0, 1**	BPWMx_BA+0x22C	R	BPWM Rising Capture Data Register 4	0x0000_0000
BPWM_FCAPDAT4 **x=0, 1**	BPWMx_BA+0x230	R	BPWM Falling Capture Data Register 4	0x0000_0000
BPWM_RCAPDAT5 **x=0, 1**	BPWMx_BA+0x234	R	BPWM Rising Capture Data Register 5	0x0000_0000
BPWM_FCAPDAT5 **x=0, 1**	BPWMx_BA+0x238	R	BPWM Falling Capture Data Register 5	0x0000_0000
BPWM_CAPIEN **x=0, 1**	BPWMx_BA+0x250	R/W	BPWM Capture Interrupt Enable Register	0x0000_0000
BPWM_CAPIF **x=0, 1**	BPWMx_BA+0x254	R/W	BPWM Capture Interrupt Flag Register	0x0000_0000
BPWM_PBUF **x=0, 1**	BPWMx_BA+0x304	R	BPWM PERIOD Buffer	0x0000_0000
BPWM_CMPBUF0 **x=0, 1**	BPWMx_BA+0x31C	R	BPWM CMPDAT0 Buffer	0x0000_0000
BPWM_CMPBUF1 **x=0, 1**	BPWMx_BA+0x320	R	BPWM CMPDAT1 Buffer	0x0000_0000
BPWM_CMPBUF2 **x=0, 1**	BPWMx_BA+0x324	R	BPWM CMPDAT2 Buffer	0x0000_0000
BPWM_CMPBUF3 **x=0, 1**	BPWMx_BA+0x328	R	BPWM CMPDAT3 Buffer	0x0000_0000
BPWM_CMPBUF4 **x=0, 1**	BPWMx_BA+0x32C	R	BPWM CMPDAT4 Buffer	0x0000_0000
BPWM_CMPBUF5 **x=0, 1**	BPWMx_BA+0x330	R	BPWM CMPDAT5 Buffer	0x0000_0000

6.8.7 Register Description

BPWM Control Register 0 (BPWM_CTL0)

Register	Offset	R/W	Description	Reset Value
BPWM_CTL0 **x=0, 1**	BPWMx_BA+0x00	R/W	BPWM Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	DBGHALT	Reserved					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		IMMLDEN5	IMMLDEN4	IMMLDEN3	IMMLDEN2	IMMLDEN1	IMMLDEN0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		CTRLD5	CTRLD4	CTRLD3	CTRLD2	CTRLD1	CTRLD0

Bits	Description	
[31]	**DBGTRIOFF**	**ICE Debug Mode Acknowledge Disable (Write Protect)** 0 = ICE debug mode acknowledgement effects BPWM output. BPWM pin will be forced as tri-state while ICE debug mode acknowledged. 1 = ICE debug mode acknowledgement disabled. BPWM pin will keep output no matter ICE debug mode acknowledged or not. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[30]	**DBGHALT**	**ICE Debug Mode Counter Halt (Write Protect)** If counter halt is enabled, BPWM all counters will keep current value until exit ICE debug mode. 0 = ICE debug mode counter halt disable. 1 = ICE debug mode counter halt enable. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[29:22]	**Reserved**	Reserved.
[n+16] n=0,1…5	**IMMLDENn**	**Immediately Load Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = PERIOD will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the end point or center point of each period by setting CTRLD bit. 1 = PERIOD/CMPDAT will load to PBUF and CMPBUF immediately when software update PERIOD/CMPDAT. **Note:** If IMMLDENn is enabled, CTRLDn will be invalid.
[15:6]	**Reserved**	Reserved.
[n] n=0,1…5	**CTRLDn**	**Center Load Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Center Lodaing mode is disable for corresponding BPWM channel n.

		1 = Center Lodaing mode is enable for corresponding BPWM channel n. In up-down counter type, PERIOD will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the center point of a period.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

BPWM Control Register 1 (BPWM_CTL1)

Register	Offset	R/W	Description	Reset Value
BPWM_CTL1 **x=0, 1**	BPWMx_BA+0x04	R/W	BPWM 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						**CNTTYPE0**	

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1:0]	**CNTTYPE0**	**BPWM Counter Behavior Type 0** 00 = Up counter type (supports in capture mode). 01 = Down count type (supports in capture mode). 10 = Up-down counter type. 11 = Reserved.

BPWM Clock Source Register (BPWM_CLKSRC)

Register	Offset	R/W	Description	Reset Value
BPWM_CLKSRC **x=0, 1**	BPWMx_BA+0x10	R/W	BPWM Clock Source 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					**ECLKSRC0**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**ECLKSRC0**	**BPWM_CH0/1 External Clock Source Select** 000 = BPWMx_CLK, x denotes 0, 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.

BPWM Clock Pre-Scale Register (BPWM_CLKPSC)

Register	Offset	R/W	Description	Reset Value
BPWM_CLKPSC **x=0, 1**	BPWMx_BA+0x14	R/W	BPWM Clock Pre-scale 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				**CLKPSC**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**CLKPSC**	**BPWM Counter Clock Pre-scale** The clock of BPWM counter is decided by clock prescaler. Each BPWM pair share one BPWM counter clock prescaler. The clock of BPWM counter is divided by (CLKPSC+ 1).

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

BPWM Counter Enable Register (BPWM_CNTEN)

Register	Offset	R/W	Description	Reset Value
BPWM_CNTEN **x=0, 1**	BPWMx_BA+0x20	R/W	BPWM Counter 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							CNTEN0

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTEN0**	**BPWM Counter Enable 0** 0 = BPWM Counter and clock prescaler Stop Running. 1 = BPWM Counter and clock prescaler Start Running.

BPWM Clear Counter Register (BPWM_CNTCLR)

Register	Offset	R/W	Description	Reset Value
BPWM_CNTCLR **x=0, 1**	BPWMx_BA+0x24	R/W	BPWM Clear Counter Register	0x0000_0000

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="7">Reserved</td><td>CNTCLR0</td></tr>
</table>

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTCLR0**	**Clear BPWM Counter Control Bit 0** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit BPWM counter to 0000H.

BPWM Period Register (BPWM_PERIOD)

Register	Offset	R/W	Description	Reset Value
BPWM_PERIOD **x=0, 1**	BPWMx_BA+0x30	R/W	BPWM Period 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**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**BPWM Period Register** Up-Count mode: In this mode, BPWM counter counts from 0 to PERIOD, and restarts from 0. Down-Count mode: In this mode, BPWM counter counts from PERIOD to 0, and restarts from PERIOD. BPWM period time = (PERIOD+1) * BPWM_CLK period. Up-Down-Count mode: In this mode, BPWM counter counts from 0 to PERIOD, then decrements to 0 and repeats again. BPWM period time = 2 * PERIOD * BPWM_CLK period.

BPWM Comparator Register 0~5 (BPWM_CMPDAT0~5)

Register	Offset	R/W	Description	Reset Value
BPWM_CMPDAT0 **x=0, 1**	BPWMx_BA+0x50	R/W	BPWM Comparator Register 0	0x0000_0000
BPWM_CMPDAT1 **x=0, 1**	BPWMx_BA+0x54	R/W	BPWM Comparator Register 1	0x0000_0000
BPWM_CMPDAT2 **x=0, 1**	BPWMx_BA+0x58	R/W	BPWM Comparator Register 2	0x0000_0000
BPWM_CMPDAT3 **x=0, 1**	BPWMx_BA+0x5C	R/W	BPWM Comparator Register 3	0x0000_0000
BPWM_CMPDAT4 **x=0, 1**	BPWMx_BA+0x60	R/W	BPWM Comparator Register 4	0x0000_0000
BPWM_CMPDAT5 **x=0, 1**	BPWMx_BA+0x64	R/W	BPWM Comparator Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMP**	**BPWM Comparator Register** CMP use to compare with CNT to generate BPWM waveform, interrupt and trigger ADC. In independent mode, BPWM_CMPDAT0~5 denote as 6 independent BPWM_CH0~5 compared point.

BPWM Counter Register (BPWM_CNT)

Register	Offset	R/W	Description	Reset Value
BPWM_CNT **x=0, 1**	BPWMx_BA+0x90	R	BPWM Counter Register 0	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**DIRF**	**BPWM Direction Indicator Flag (Read Only)** 0 = Counter is Down count. 1 = Counter is UP count.
[15:0]	**CNT**	**BPWM Data Register (Read Only)** User can monitor CNT to know the current value in 16-bit period counter.

BPWM Generation Register 0 (BPWM_WGCTL0)

Register	Offset	R/W	Description	Reset Value
BPWM_WGCTL0 **x=0, 1**	BPWMx_BA+0xB0	R/W	BPWM Generation Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				PRDPCTL5		PRDPCTL4	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRDPCTL3		PRDPCTL2		PRDPCTL1		PRDPCTL0	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				ZPCTL5		ZPCTL4	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ZPCTL3		ZPCTL2		ZPCTL1		ZPCTL0	

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**PRDPCTLn**	**BPWM Period (Center) Point Control** Each bit n controls the corresponding BPWM channel n. 00 = Do nothing. 01 = BPWM period (center) point output Low. 10 = BPWM period (center) point output High. 11 = BPWM period (center) point output Toggle. BPWM can control output level when BPWM counter count to (PERIODn+1). **Note:** This bit is center point control when BPWM counter operating in up-down counter type.
[15:12]	**Reserved**	Reserved.
[11:0]	**ZPCTLn**	**BPWM Zero Point Control** Each bit n controls the corresponding BPWM channel n. 00 = Do nothing. 01 = BPWM zero point output Low. 10 = BPWM zero point output High. 11 = BPWM zero point output Toggle. BPWM can control output level when BPWM counter count to zero.

BPWM Generation Register 1 (BPWM_WGCTL1)

Register	Offset	R/W	Description	Reset Value
BPWM_WGCTL1 **x=0, 1**	BPWMx_BA+0xB4	R/W	BPWM Generation Register 1	0x0000_0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Reserved				**CMPDCTL5**		**CMPDCTL4**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPDCTL3		**CMPDCTL2**		**CMPDCTL1**		**CMPDCTL0**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CMPUCTL5**		**CMPUCTL4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPUCTL3		**CMPUCTL2**		**CMPUCTL1**		**CMPUCTL0**	

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**CMPDCTLn**	**BPWM Compare Down Point Control** Each bit n controls the corresponding BPWM channel n. 00 = Do nothing. 01 = BPWM compare down point output Low. 10 = BPWM compare down point output High. 11 = BPWM compare down point output Toggle. BPWM can control output level when BPWM counter down count to CMPDAT.
[15:12]	**Reserved**	Reserved.
[11:0]	**CMPUCTLn**	**BPWM Compare Up Point Control** Each bit n controls the corresponding BPWM channel n. 00 = Do nothing. 01 = BPWM compare up point output Low. 10 = BPWM compare up point output High. 11 = BPWM compare up point output Toggle. BPWM can control output level when BPWM counter up count to CMPDAT.

BPWM Mask Enable Register (BPWM_MSKEN)

Register	Offset	R/W	Description	Reset Value
BPWM_MSKEN **x=0, 1**	BPWMx_BA+0xB8	R/W	BPWM Mask 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		**MSKEN5**	**MSKEN4**	**MSKEN3**	**MSKEN2**	**MSKEN1**	**MSKEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**MSKENn**	**BPWM Mask Enable Bits** Each bit n controls the corresponding BPWM channel n. The BPWM output signal will be masked when this bit is enabled. The corresponding BPWM channel n will output MSKDATn (BPWM_MSK[5:0]) data. 0 = BPWM output signal is non-masked. 1 = BPWM output signal is masked and output MSKDATn data.

BPWM Mask DATA Register (BPWM_MSK)

Register	Offset	R/W	Description	Reset Value
BPWM_MSK **x=0, 1**	BPWMx_BA+0xBC	R/W	BPWM Mask 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**
Reserved		**MSKDAT5**	**MSKDAT4**	**MSKDAT3**	**MSKDAT2**	**MSKDAT1**	**MSKDAT0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**MSKDATn**	**BPWM Mask Data Bit** This data bit control the state of BPWMn output pin, if corresponding mask function is enabled. Each bit n controls the corresponding BPWM channel n. 0 = Output logic low to BPWMn. 1 = Output logic high to BPWMn.

BPWM Pin Polar Inverse Control Register (BPWM_POLCTL)

Register	Offset	R/W	Description	Reset Value
BPWM_POLCTL **x=0, 1**	BPWMx_BA+0xD4	R/W	BPWM Pin Polar Inverse 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		**PINV5**	**PINV4**	**PINV3**	**PINV2**	**PINV1**	**PINV0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**PINVn**	**BPWM PIN Polar Inverse Control Bits** The register controls polarity state of BPWM output. Each bit n controls the corresponding BPWM channel n. 0 = BPWM output polar inverse Disabled. 1 = BPWM output polar inverse Enabled.

BPWM Output Enable Register (BPWM_POEN)

Register	Offset	R/W	Description	Reset Value
BPWM_POEN **x=0, 1**	BPWMx_BA+0xD8	R/W	BPWM Output 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		**POEN5**	**POEN4**	**POEN3**	**POEN2**	**POEN1**	**POEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**POENn**	**BPWM Pin Output Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = BPWM pin at tri-state. 1 = BPWM pin in output mode.

BPWM Interrupt Enable Register (BPWM_INTEN)

Register	Offset	R/W	Description	Reset Value
BPWM_INTEN **x=0, 1**	BPWMx_BA+0xE0	R/W	BPWM Interrupt Enable Register	0x0000_0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Reserved		**CMPDIEN5**	**CMPDIEN4**	**CMPDIEN3**	**CMPDIEN2**	**CMPDIEN1**	**CMPDIEN0**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**CMPUIEN5**	**CMPUIEN4**	**CMPUIEN3**	**CMPUIEN2**	**CMPUIEN1**	**CMPUIEN0**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**PIEN0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**ZIEN0**

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**CMPDIENn**	**BPWM Compare Down Count Interrupt Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Compare down count interrupt Disabled. 1 = Compare down count interrupt Enabled.
[23:22]	**Reserved**	Reserved.
[21:16]	**CMPUIENn**	**BPWM Compare Up Count Interrupt Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Compare up count interrupt Disabled. 1 = Compare up count interrupt Enabled. .
[15:9]	**Reserved**	Reserved.
[8]	**PIEN0**	**BPWM Period Point Interrupt Enable 0** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When up-down counter type period point means center point.
[7:1]	**Reserved**	Reserved.
[0]	**ZIEN0**	**BPWM Zero Point Interrupt Enable 0** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled.

BPWM Interrupt Flag Register 0 (BPWM_INTSTS0)

Register	Offset	R/W	Description	Reset Value
BPWM_INTSTS0 x=0, 1	BPWMx_BA+0xE8	R/W	BPWM Interrupt Flag Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							PIF0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							ZIF0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**CMPDIFn**	**BPWM Compare Down Count Interrupt Flag** Each bit n controls the corresponding BPWM channel n. Flag is set by hardware when BPWM counter down count and reaches BPWM_CMPDATn, software can clear this bit by writing 1 to it. **Note1:** If CMPDAT equal to PERIOD, this flag is not working in down counter type selection.
[23:22]	**Reserved**	Reserved.
[21:16]	**CMPUIFn**	**BPWM Compare Up Count Interrupt Flag** Flag is set by hardware when BPWM counter up count and reaches BPWM_CMPDATn, software can clear this bit by writing 1 to it. Each bit n controls the corresponding BPWM channel n. **Note1:** If CMPDAT equal to PERIOD, this flag is not working in up counter type selection.
[15:9]	**Reserved**	Reserved.
[8]	**PIF0**	**BPWM Period Point Interrupt Flag 0** This bit is set by hardware when BPWM_CH0 counter reaches BPWM_PERIOD0, software can write 1 to clear this bit to zero.
[7:1]	**Reserved**	Reserved.
[0]	**ZIF0**	**BPWM Zero Point Interrupt Flag 0** This bit is set by hardware when BPWM_CH0 counter reaches zero, software can write 1 to clear this bit to zero.

BPWM Trigger ADC Source Select Register 0 (BPWM_ADCTS0)

Register	Offset	R/W	Description	Reset Value
BPWM_ADCTS0 **x=0, 1**	BPWMx_BA+0xF8	R/W	BPWM Trigger ADC Source Select Register 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved			TRGSEL3			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TRGEN2	Reserved			TRGSEL2			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TRGEN1	Reserved			TRGSEL1			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRGEN0	Reserved			TRGSEL0			

Bits	Description	
[31]	**TRGEN3**	**BPWM_CH3 Trigger ADC Enable Bit**
[30:28]	**Reserved**	Reserved.
[27:24]	**TRGSEL3**	**BPWM_CH3 Trigger ADC Source Select** 0000 = BPWM_CH2 zero point. 0001 = BPWM_CH2 period point. 0010 = BPWM_CH2 zero or period point. 0011 = BPWM_CH2 up-count CMPDAT point. 0100 = BPWM_CH2 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH3 up-count CMPDAT point. 1001 = BPWM_CH3 down-count CMPDAT point. Others = reserved.
[23]	**TRGEN2**	**BPWM_CH2 Trigger ADC Enable Bit**
[22:20]	**Reserved**	Reserved.
[19:16]	**TRGSEL2**	**BPWM_CH2 Trigger ADC Source Select** 0000 = BPWM_CH2 zero point. 0001 = BPWM_CH2 period point. 0010 = BPWM_CH2 zero or period point. 0011 = BPWM_CH2 up-count CMPDAT point. 0100 = BPWM_CH2 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved.

		1000 = BPWM_CH3 up-count CMPDAT point. 1001 = BPWM_CH3 down-count CMPDAT point. Others reserved.
[15]	**TRGEN1**	**BPWM_CH1 Trigger ADC Enable Bit**
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL1**	**BPWM_CH1 Trigger ADC Source Select** 0000 = BPWM_CH0 zero point. 0001 = BPWM_CH0 period point. 0010 = BPWM_CH0 zero or period point. 0011 = BPWM_CH0 up-count CMPDAT point. 0100 = BPWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH1 up-count CMPDAT point. 1001 = BPWM_CH1 down-count CMPDAT point. Others = reserved.
[7]	**TRGEN0**	**BPWM_CH0 Trigger ADC Enable Bit**
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL0**	**BPWM_CH0 Trigger ADC Source Select** 0000 = BPWM_CH0 zero point. 0001 = BPWM_CH0 period point. 0010 = BPWM_CH0 zero or period point. 0011 = BPWM_CH0 up-count CMPDAT point. 0100 = BPWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH1 up-count CMPDAT point. 1001 = BPWM_CH1 down-count CMPDAT point. Others = reserved.

BPWM Trigger ADC Source Select Register 1 (BPWM_ADCTS1)

Register	Offset	R/W	Description	Reset Value
BPWM_ADCTS1	BPWMx_BA+0xFC	R/W	BPWM Trigger ADC Source Select 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
TRGEN5	Reserved			TRGSEL5			
7	6	5	4	3	2	1	0
TRGEN4	Reserved			TRGSEL4			

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**TRGEN5**	**BPWM_CH5 Trigger ADC Enable Bit**
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL5**	**BPWM_CH5 Trigger ADC Source Select** 0000 = BPWM_CH4 zero point. 0001 = BPWM_CH4 period point. 0010 = BPWM_CH4 zero or period point. 0011 = BPWM_CH4 up-count CMPDAT point. 0100 = BPWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH5 up-count CMPDAT point. 1001 = BPWM_CH5 down-count CMPDAT point. Others = reserved.
[7]	**TRGEN4**	**BPWM_CH4 Trigger ADC Enable Bit**
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL4**	**BPWM_CH4 Trigger ADC Source Select** 0000 = BPWM_CH4 zero point. 0001 = BPWM_CH4 period point. 0010 = BPWM_CH4 zero or period point. 0011 = BPWM_CH4 up-count CMPDAT point. 0100 = BPWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved.

		0111 = Reserved. 1000 = BPWM_CH5 up-count CMPDAT point. 1001 = BPWM_CH5 down-count CMPDAT point. Others = reserved.

BPWM Synchronous Start Control Register (BPWM_SSCTL)

Register	Offset	R/W	Description	Reset Value
BPWM_SSCTL **x=0, 1**	BPWMx_BA+0x110	R/W	BPWM Synchronous Start 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						**SSRC**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SSEN0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**SSRC**	**BPWM Synchronous Start Source Select** 00 = Synchronous start source come from BPWM0. 01 = Synchronous start source come from BPWM1. 10 = Synchronous start source come from PWM0. 11 = Synchronous start source come from PWM1.
[7:1]	**Reserved**	Reserved.
[0]	**SSEN0**	**BPWM Synchronous Start Function Enable 0** When synchronous start function is enabled, the BPWM_CH0 counter enable bit (CNTEN0) can be enabled by writing BPWM synchronous start trigger bit (CNTSEN). 0 = BPWM synchronous start function Disabled. 1 = BPWM synchronous start function Enabled.

BPWM Synchronous Start Trigger Register (BPWM_SSTRG)

Register	Offset	R/W	Description	Reset Value
BPWM_SSTRG **x=0, 1**	BPWMx_BA+0x114	W	BPWM Synchronous Start Trigger 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							CNTSEN

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTSEN**	**BPWM Counter Synchronous Start Enable (Write Only)** If BPWM synchronous start function is Enabled. Set this bit is used to make selected BPWM channels (include BPWM0_CHn, BPWM1_CHn, n=0~5) start counting at the same time. Writing this bit to 1 will also set the counter enable bit (CNTENn, n denotes channel 0 to 5) if correlated BPWM channel counter synchronous start function is enabled.

BPWM Status Register (BPWM_STATUS)

Register	Offset	R/W	Description	Reset Value
BPWM_STATUS **x=0, 1**	BPWMx_BA+0x120	R/W	BPWM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**ADCTRG5**	**ADCTRG4**	**ADCTRG3**	**ADCTRG2**	**ADCTRG1**	**ADCTRG0**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**CNTMAX0**

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21:16]	**ADCTRGn**	**ADC Start of Conversion Status** Each bit n controls the corresponding BPWM channel n. 0 = Indicates no ADC start of conversion trigger event has occurred. 1 = Indicates an ADC start of conversion trigger event has occurred, software can write 1 to clear this bit.
[15:1]	**Reserved**	Reserved.
[0]	**CNTMAX0**	**Time-base Counter 0 Equal to 0xFFFF Latched Status** 0 = Indicates the time-base counter never reached its maximum value 0xFFFF. 1 = Indicates the time-base counter reached its maximum value, software can write 1 to clear this bit.

BPWM Capture Input Enable Register (BPWM_CAPINEN)

Register	Offset	R/W	Description	Reset Value
BPWM_CAPINEN **x=0, 1**	BPWMx_BA+0x200	R/W	BPWM Capture Input 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		**CAPINEN5**	**CAPINEN4**	**CAPINEN3**	**CAPINEN2**	**CAPINEN1**	**CAPINEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**CAPINENn**	**Capture Input Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = BPWM Channel capture input path Disabled. The input of BPWM channel capture function is always regarded as 0. 1 = BPWM Channel capture input path Enabled. The input of BPWM channel capture function comes from correlative multifunction pin.

BPWM Capture Control Register (BPWM_CAPCTL)

Register	Offset	R/W	Description	Reset Value
BPWM_CAPCTL **x=0, 1**	BPWMx_BA+0x204	R/W	BPWM Capture Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		FCRLDEN5	FCRLDEN4	FCRLDEN3	FCRLDEN2	FCRLDEN1	FCRLDEN0
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		RCRLDEN5	RCRLDEN4	RCRLDEN3	RCRLDEN2	RCRLDEN1	RCRLDEN0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		CAPINV5	CAPINV4	CAPINV3	CAPINV2	CAPINV1	CAPINV0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		CAPEN5	CAPEN4	CAPEN3	CAPEN2	CAPEN1	CAPEN0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**FCRLDENn**	**Falling Capture Reload Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[23:22]	**Reserved**	Reserved.
[21:16]	**RCRLDENn**	**Rising Capture Reload Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[15:14]	**Reserved**	Reserved.
[13:8]	**CAPINVn**	**Capture Inverter Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[7:6]	**Reserved**	Reserved.
[5:0]	**CAPENn**	**Capture Function Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Capture function Disabled. RCAPDAT/FCAPDAT register will not be updated. 1 = Capture function Enabled. Capture latched the BPWM counter value when detected rising or falling edge of input signal and saved to RCAPDAT (Rising latch) and FCAPDAT (Falling latch).

BPWM Capture Status Register (BPWM_CAPSTS)

Register	Offset	R/W	Description	Reset Value
BPWM_CAPSTS **x=0, 1**	BPWMx_BA+0x208	R	BPWM Capture 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		CFLIFOV5	CFLIFOV4	CFLIFOV3	CFLIFOV2	CFLIFOV1	CFLIFOV0
7	6	5	4	3	2	1	0
Reserved		CRLIFOV5	CRLIFOV4	CRLIFOV3	CRLIFOV2	CRLIFOV1	CRLIFOV0

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:8]	**CFLIFOVn**	**Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIF is 1. Each bit n controls the corresponding BPWM channel n. **Note:** This bit will be cleared automatically when user clear corresponding CFLIF.
[7:6]	**Reserved**	Reserved.
[5:0]	**CRLIFOVn**	**Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIF is 1. Each bit n controls the corresponding BPWM channel n. **Note:** This bit will be cleared automatically when user clear corresponding CRLIF.

BPWM Rising Capture Data Register 0~5 (BPWM_RCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
BPWM_RCAPDAT0 **x=0, 1**	BPWMx_BA+0x20C	R	BPWM Rising Capture Data Register 0	0x0000_0000
BPWM_RCAPDAT1 **x=0, 1**	BPWMx_BA+0x214	R	BPWM Rising Capture Data Register 1	0x0000_0000
BPWM_RCAPDAT2 **x=0, 1**	BPWMx_BA+0x21C	R	BPWM Rising Capture Data Register 2	0x0000_0000
BPWM_RCAPDAT3 **x=0, 1**	BPWMx_BA+0x224	R	BPWM Rising Capture Data Register 3	0x0000_0000
BPWM_RCAPDAT4 **x=0, 1**	BPWMx_BA+0x22C	R	BPWM Rising Capture Data Register 4	0x0000_0000
BPWM_RCAPDAT5 **x=0, 1**	BPWMx_BA+0x234	R	BPWM Rising Capture Data Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RCAPDAT**	**BPWM Rising Capture Data Register (Read Only)** When rising capture condition happened, the BPWM counter value will be saved in this register.

BPWM Falling Capture Data Register 0~5 (BPWM_FCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
BPWM_FCAPDAT0 **x=0, 1**	BPWMx_BA+0x210	R	BPWM Falling Capture Data Register 0	0x0000_0000
BPWM_FCAPDAT1 **x=0, 1**	BPWMx_BA+0x218	R	BPWM Falling Capture Data Register 1	0x0000_0000
BPWM_FCAPDAT2 **x=0, 1**	BPWMx_BA+0x220	R	BPWM Falling Capture Data Register 2	0x0000_0000
BPWM_FCAPDAT3 **x=0, 1**	BPWMx_BA+0x228	R	BPWM Falling Capture Data Register 3	0x0000_0000
BPWM_FCAPDAT4 **x=0, 1**	BPWMx_BA+0x230	R	BPWM Falling Capture Data Register 4	0x0000_0000
BPWM_FCAPDAT5 **x=0, 1**	BPWMx_BA+0x238	R	BPWM Falling Capture Data Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FCAPDAT**	**BPWM Falling Capture Data Register (Read Only)** When falling capture condition happened, the BPWM counter value will be saved in this register.

BPWM Capture Interrupt Enable Register (BPWM_CAPIEN)

Register	Offset	R/W	Description	Reset Value
BPWM_CAPIEN **x=0, 1**	BPWMx_BA+0x250	R/W	BPWM Capture 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		CAPFIEN5	CAPFIEN4	CAPFIEN3	CAPFIEN2	CAPFIEN1	CAPFIEN0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		CAPRIEN5	CAPRIEN4	CAPRIEN3	CAPRIEN2	CAPRIEN1	CAPRIEN0

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:8]	**CAPFIENn**	**BPWM Capture Falling Latch Interrupt Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled.
[7:6]	**Reserved**	Reserved.
[5:0]	**CAPRIENn**	**BPWM Capture Rising Latch Interrupt Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled.

BPWM Capture Interrupt Flag Register (BPWM_CAPIF)

Register	Offset	R/W	Description	Reset Value
BPWM_CAPIF **x=0, 1**	BPWMx_BA+0x254	R/W	BPWM Capture Interrupt Flag 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		**CFLIF5**	**CFLIF4**	**CFLIF3**	**CFLIF2**	**CFLIF1**	**CFLIF0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CRLIF5**	**CRLIF4**	**CRLIF3**	**CRLIF2**	**CRLIF1**	**CRLIF0**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:8]	**CFLIFn**	**BPWM Capture Falling Latch Interrupt Flag** This bit is writing 1 to clear. Each bit n controls the corresponding BPWM channel n. 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high.
[7:6]	**Reserved**	Reserved.
[5:0]	**CRLIFn**	**BPWM Capture Rising Latch Interrupt Flag** This bit is writing 1 to clear. Each bit n controls the corresponding BPWM channel n. 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high.

BPWM Period Register Buffer (BPWM_PBUF)

Register	Offset	R/W	Description	Reset Value
BPWM_PBUF **x=0, 1**	BPWMx_BA+0x30 4	R	BPWM PERIOD Buffer	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PBUF**	**BPWM Period Register Buffer (Read Only)** Used as PERIOD active register.

BPWM Comparator Register Buffer 0~5 (BPWM_CMPBUF0~5)

Register	Offset	R/W	Description	Reset Value
BPWM_CMPBUF0 **x=0, 1**	BPWMx_BA+0x31C	R	BPWM CMPDAT0 Buffer	0x0000_0000
BPWM_CMPBUF1 **x=0, 1**	BPWMx_BA+0x320	R	BPWM CMPDAT1 Buffer	0x0000_0000
BPWM_CMPBUF2 **x=0, 1**	BPWMx_BA+0x324	R	BPWM CMPDAT2 Buffer	0x0000_0000
BPWM_CMPBUF3 **x=0, 1**	BPWMx_BA+0x328	R	BPWM CMPDAT3 Buffer	0x0000_0000
BPWM_CMPBUF4 **x=0, 1**	BPWMx_BA+0x32C	R	BPWM CMPDAT4 Buffer	0x0000_0000
BPWM_CMPBUF5 **x=0, 1**	BPWMx_BA+0x330	R	BPWM CMPDAT5 Buffer	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPBUF**	**BPWM Comparator Register Buffer (Read Only)** Used as CMP active register.

6.9 PWM Generator and Capture Timer (PWM)

6.9.1 Overview

The NUC121/125 series provides two PWM generators – PWM0 and PWM1 as shown in Figure 6.9-1. Each PWM supports 6 channels of PWM output or input capture. There is a 12-bit prescaler to support flexible clock to the 16-bit PWM counter with 16-bit comparator. The PWM counter supports up, down and up-down counter types. PWM uses the comparator compared with counter to generate events. These events are used to generate PWM pulse, interrupt and trigger signal for ADC to start conversion.

The PWM generator supports two standard PWM output modes: Independent mode and Complementary mode, which have difference architecture. In Complementary mode, there are two comparators to generate various PWM pulse with 12-bit dead-time generator. For PWM output control unit, it supports polarity output, independent pin mask, tri-state output enable and brake functions.

The PWM generator also supports input capture function to latch PWM counter value to the corresponding register when input channel has a rising transition, falling transition or both transition is happened.

6.9.2 Features

6.9.2.1 PWM function features

- Supports maximum clock frequency up to100 MHz
- Supports up to two PWM modules, each module provides 6 output channels
- Supports independent mode for PWM output/Capture input channel
- Supports complementary mode for 3 complementary paired PWM output channel
  - Dead-time insertion with 12-bit resolution
- Two compared values during one period
- Supports 12-bit pre-scalar from 1 to 4096
- Supports 16-bit resolution PWM counter, each module provides 3 PWM counters
  - Up, down and up/down counter operation type
- Supports mask function and tri-state enable for each PWM pin
- Supports brake function
  - Brake source from pin and system safety events (clock failed, Brown-out detection and CPU lockup)
  - Noise filter for brake source from pin
  - Edge detect brake source to control brake state until brake interrupt cleared
  - Level detect brake source to auto recover function after brake condition removed
- Supports interrupt on the following events:
  - PWM counter match zero, period value or compared value
  - Brake condition happened
- Supports trigger ADC on the following events:
  - PWM counter match zero, period value or compared value

6.9.2.2 Capture Function Features

- Supports up to 12 capture input channels with 16-bit resolution
- Supports rising or falling capture condition
- Supports input rising/falling capture interrupt
- Supports rising/falling capture with counter reload option

6.9.3 Block Diagram

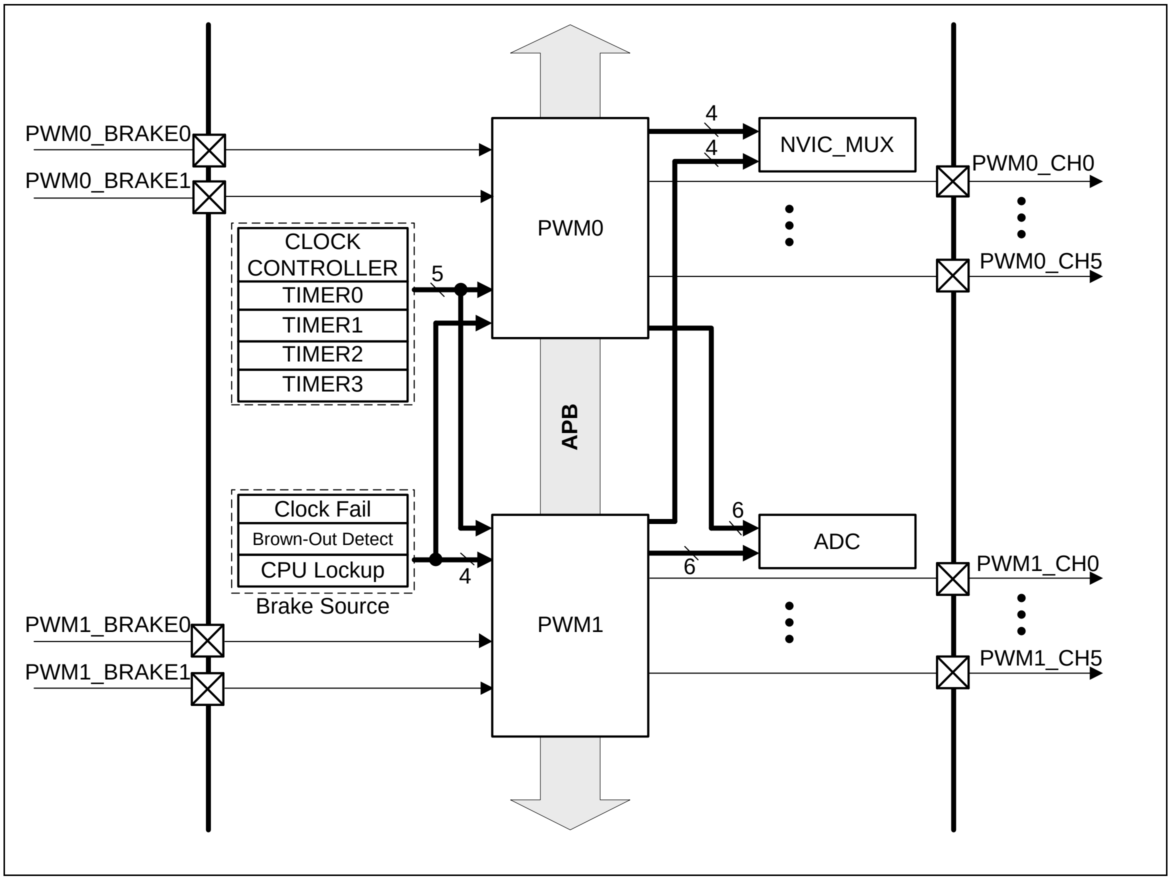


Figure 6.9-1 PWM Generator Overview Block Diagram

PWM system clock frequency can be set equal or double to HCLK frequency as Figure 6.9-2, the detail register setting, please refer to Table 6.9-1.

Each PWM generator has three clock source inputs, each clock source can be selected from system clock or four TIMER trigger PWM outputs as Figure 6.9-3 by ECLKSRC0 (PWM_CLKSRC[2:0]) for PWM_CLK0, ECLKSRC2 (PWM_CLKSRC[10:8]) for PWM_CLK2 and ECLKSRC4 (PWM_CLKSRC[18:16]) for PWM_CLK4.

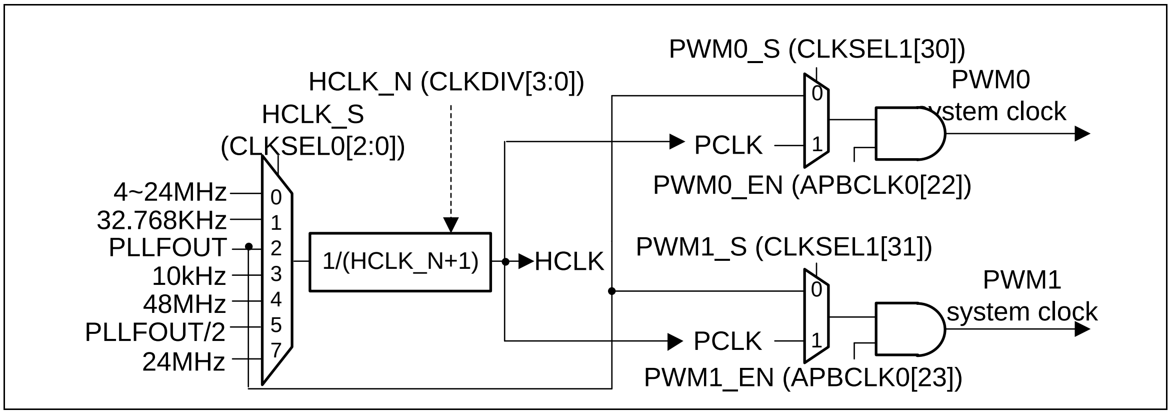


Figure 6.9-2 PWM System Clock Source Control

PWM System Clock/HCLK Frequency Ratio	HCLK_S (CLK_CLKSEL0[2:0])	HCLK_N (CLK_CLKDIV0[3:0])	PWMn_S (CLKSEL3[X]), (N, X) Denotes (0, 16) Or (1, 17)
1/1	Don't care	Don't care	1
2/1	2	1	0

Table 6.9-1 PWM System Clock Source Control Registers Setting Table

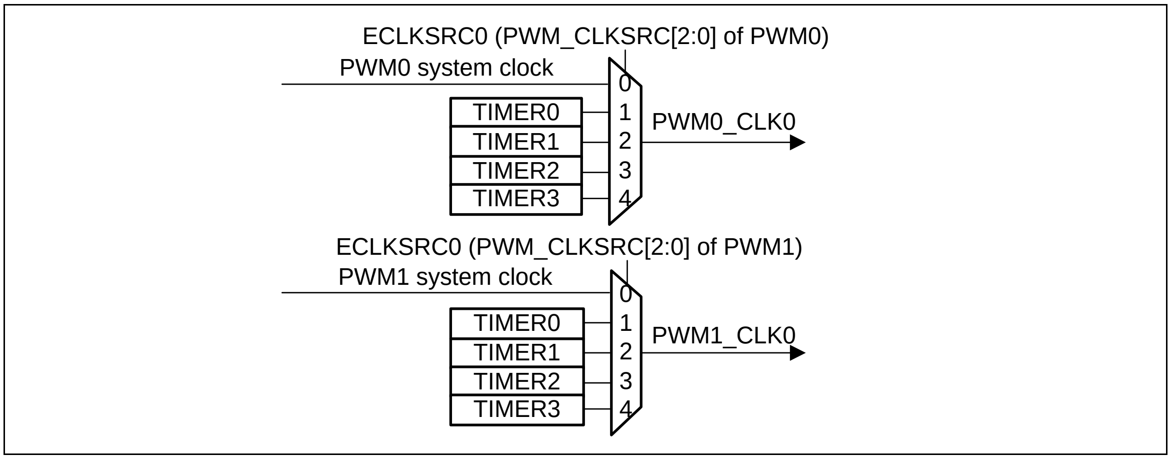


Figure 6.9-3 PWM Clock Source Control

Figure 6.9-4 and Figure 6.9-5 illustrate the architecture of PWM Independent mode and Complementary mode. Regardless of Independent mode or Complementary mode, paired channels' (PWM_CH0 and PWM_CH1, PWM_CH2 and PWM_CH3, PWM_CH4 and PWM_CH5) share the same counter. When the counter counts to 0, PERIOD (PWM_PERIODn[15:0]) or equal to comparator, events will be generated. These events are passed to the corresponding generators to generate PWM pulse, interrupt signal and trigger signal for ADC to start conversion. Output control is used to changing PWM pulse output state; brake function in output control also generates interrupt events. In Complementary mode, even channel use odd channel comparator to generate events.

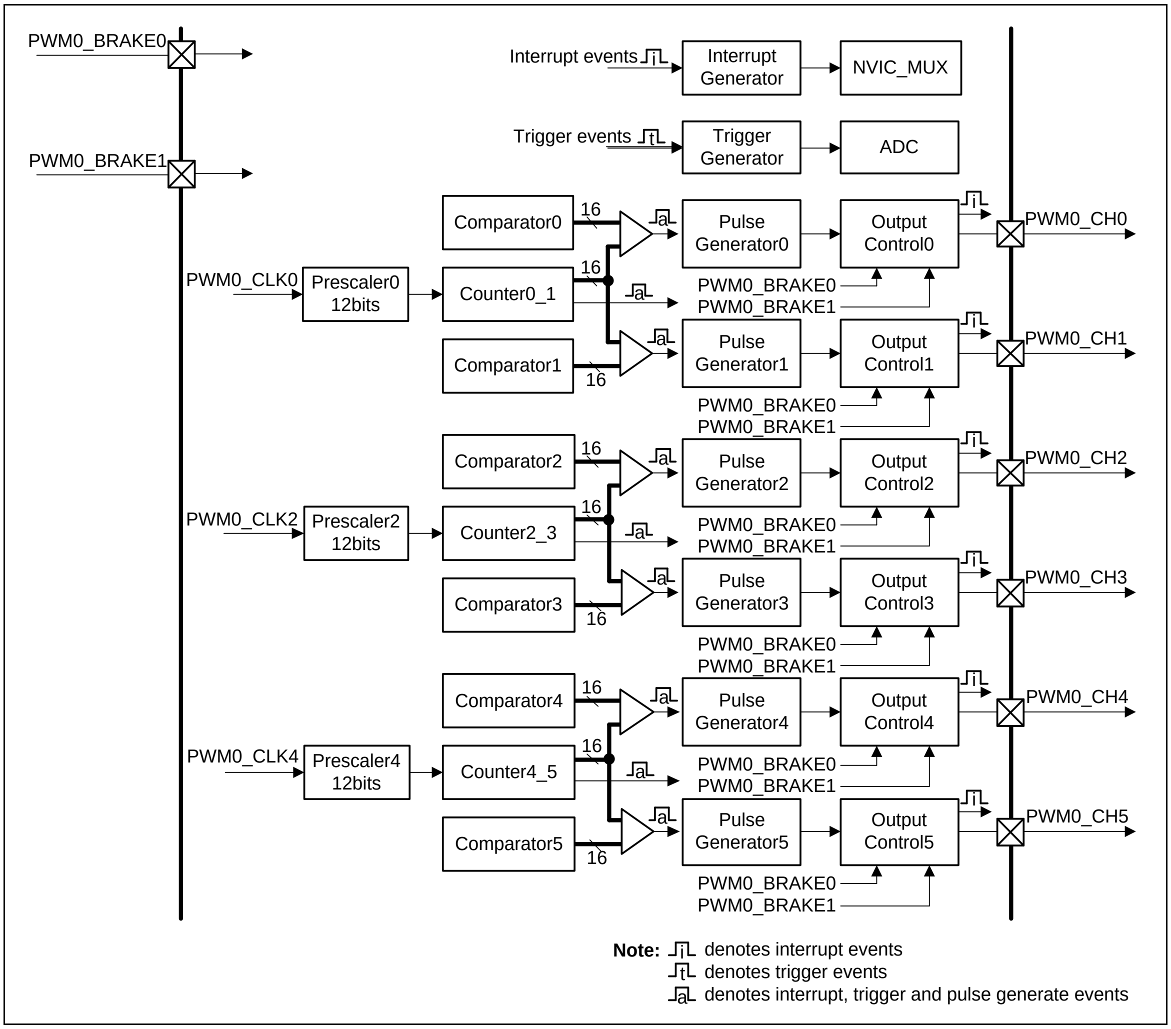


Figure 6.9-4 PWM Independent Mode Architecture Diagram

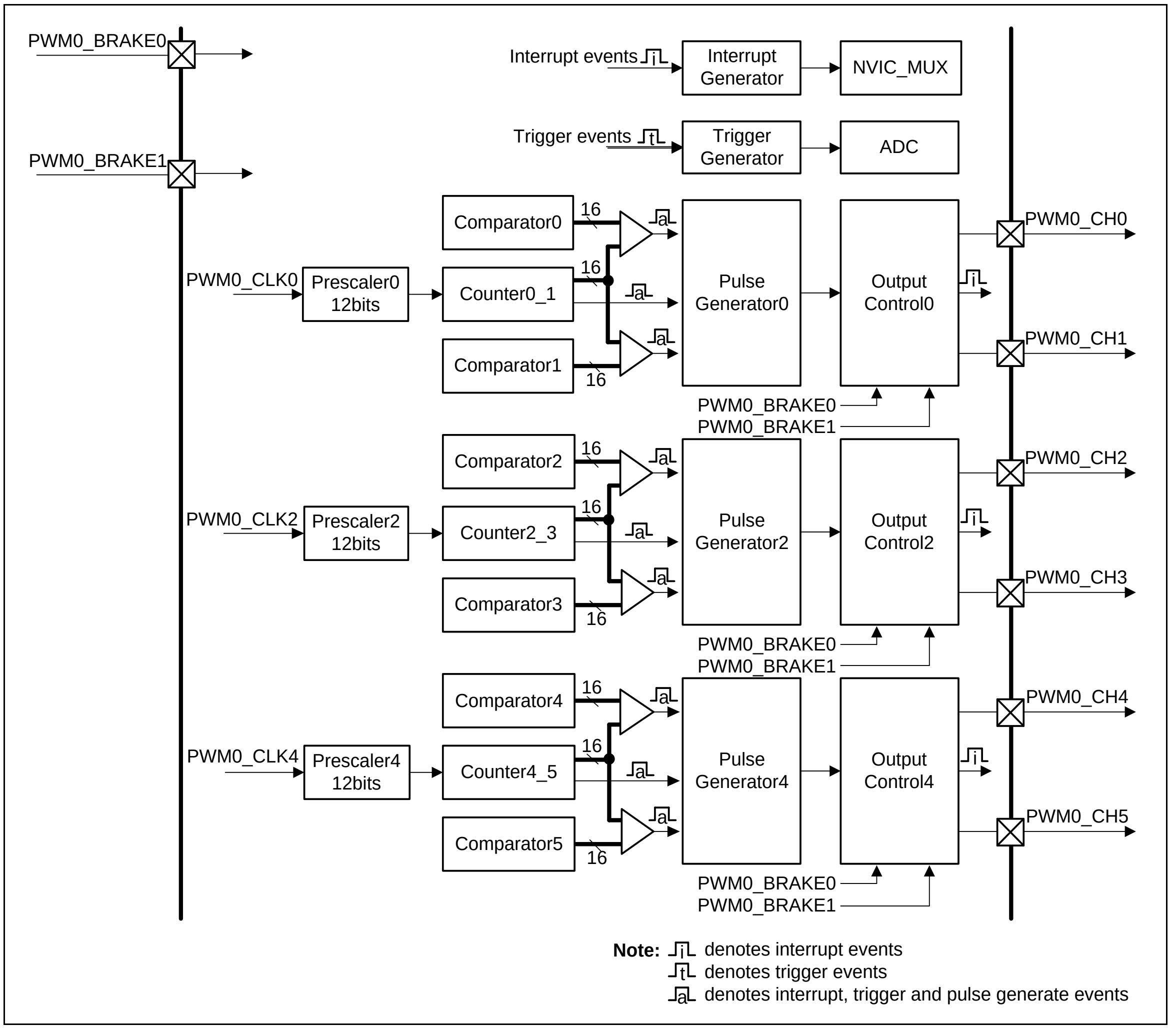


Figure 6.9-5 PWM Complementary Mode Architecture Diagram

6.9.4 Basic Configuration

The PWM pin function is configured in GPA_MFP register, PWM_BRAKE0 and PWM_BRAKE1 pin functions are configured in GPB_MFP and GPC_MFP registers.

The PWM clock can be enabled in APBCLK0[23:22]. The PWM clock source is selected by CLKSEL1[31:30].

6.9.5 Functional Description

6.9.5.1 PWM Prescaler

The PWM prescaler is used to divide clock source, prescaler counting CLKPSC +1 times, PWM counter only count once. CLKPSC (Clock Pre-scale Register) is set by CLKPSC (PWM_CLKPSCn[11:0], n denotes 0_1, 2_3, 4_5). Figure 6.9-6 shows an example of PWM channel 0 CLKPSC waveform.

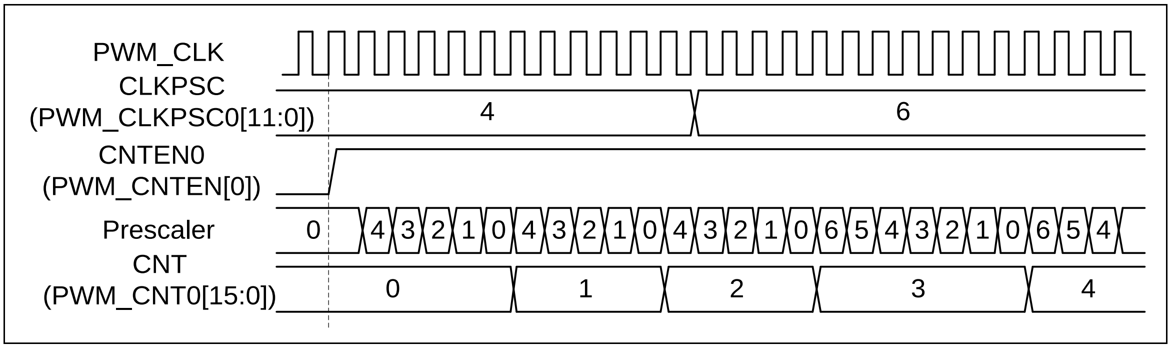


Figure 6.9-6 PWM_CH0 CLKPSC waveform

6.9.5.2 PWM Counter

PWM supports 3 counter types operation: Up Counter, Down Counter and Up-Down Counter types.

6.9.5.3 Up Counter Type

In the up counter operation, the 16 bits PWM counter is an up counter and starts up-counting from zero to PERIOD (PWM_PERIODn[15:0], where n denotes channel number) to finish a PWM period. The current counter value can be found by reading the CNT (PWM_CNTn[15:0]). PWM generates zero point event when counter counts to 0 and generates period point event when counting to PERIOD. Figure 6.9-7 shows an example of up counter, wherein PWM period time = (PERIOD+1) * PWM clock time.

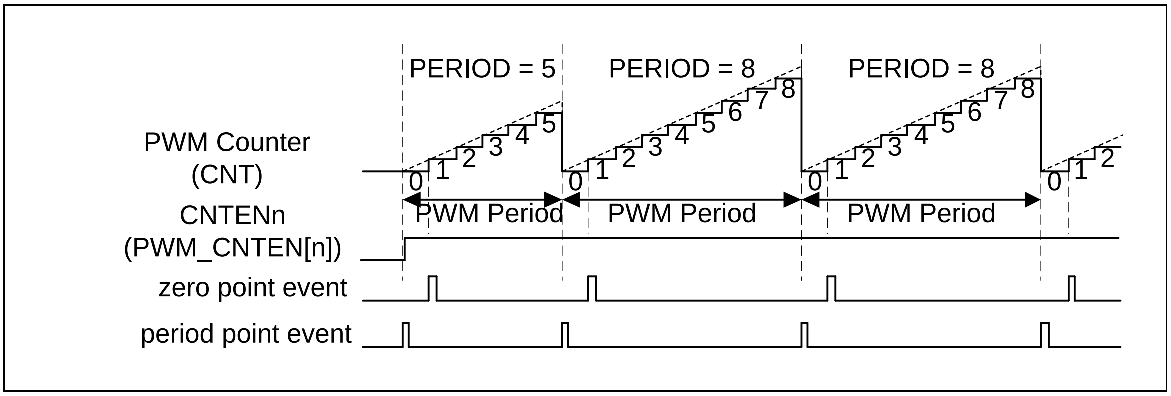


Figure 6.9-7 PWM Up Counter Type

6.9.5.4 Down Counter Type

In the down counter type, the 16 bits PWM counter is a down counter and starts down-counting from PERIOD to zero to finish a PWM period, current counter value can read CNT to know. PWM generates zero point event when counter counts to 0 and period point event when counts to PERIOD. Figure 6.9-8 is an example of down counter, a PWM period time = (PERIOD+1) * PWM clock time.

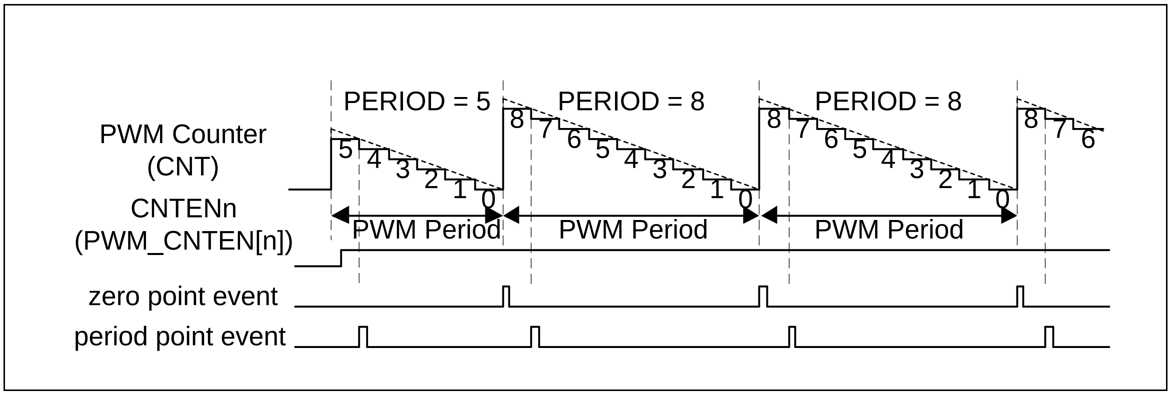


Figure 6.9-8 PWM Down Counter Type

6.9.5.5 Up-Down Counter Type

In up-down counter operation, the 16 bits PWM counter is an up-down counter and starts counting-up from zero to PERIOD and then starts counting down to zero to finish a PWM period. The current counter value can be found by reading the CNT. PWM generates zero point event when counter counts to 0 and generates center point event when counting to PERIOD. Figure 6.9-9 shows an example of up-down counter, wherein PWM period time = (2*PERIOD) * PWM clock time. The DIRF (PWM_CNTn[16]) is counter direction indicator flag, where high is up counting, and low is down counting.

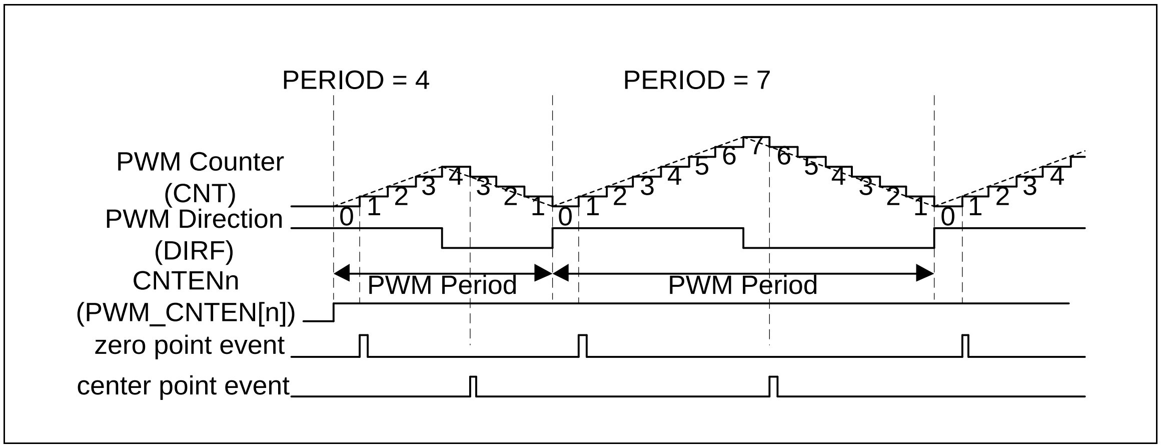


Figure 6.9-9 PWM Up-Down Counter Type

6.9.5.6 PWM Comparator

The CMPDAT (PWM_CMPDATn[15:0]) is a basic comparator register of PWM channel n; each channel only has one CMPDAT. The CMPDAT’s value is continuously compared to the corresponding complementary channel’s counter value. When the counter is equal to compared register, PWM generates an event and uses the event to generate PWM pulse, interrupt or use to trigger ADC. In up-down counter type, two events will be generated in a PWM period as shown in Figure 6.9-10.

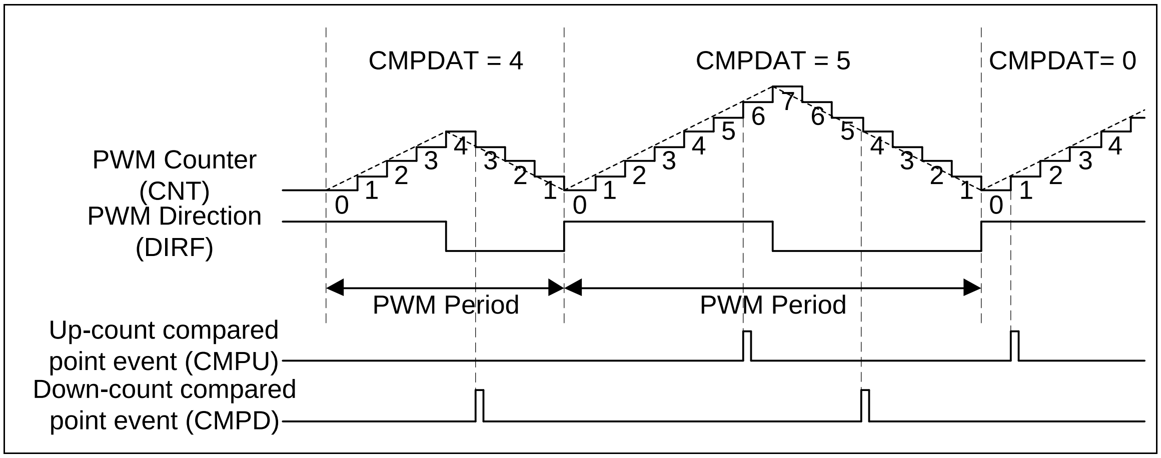


Figure 6.9-10 PWM CMPDAT Events in Up-Down Counter Type

6.9.5.7 *PWM Double Buffering*

The double buffering uses double buffers to separate software writing and hardware action operation timing. After registers are modified through software, hardware will load register value to the buffer register according to the loading mode timing. The hardware action is based on the buffer value. This can prevent asynchronously operation problem due to software and hardware asynchronism.

The PWM has double buffering function for PERIOD and CMPDAT. The concept of double buffering is used in loading modes, which are described in the following sections. For example, as shown in Figure 6.9-11, in period loading mode, writing PERIOD and CMPDAT through software, PWM will load new values to their buffer PBUF (PWM_PBUFn[15:0]) and CMPBUF (PWM_CMPBUFn[15:0]) at start of the next period without affecting the current period counter operation. There are 3 loading modes for loading value to buffer: period loading mode, immediately loading mode and center loading mode.

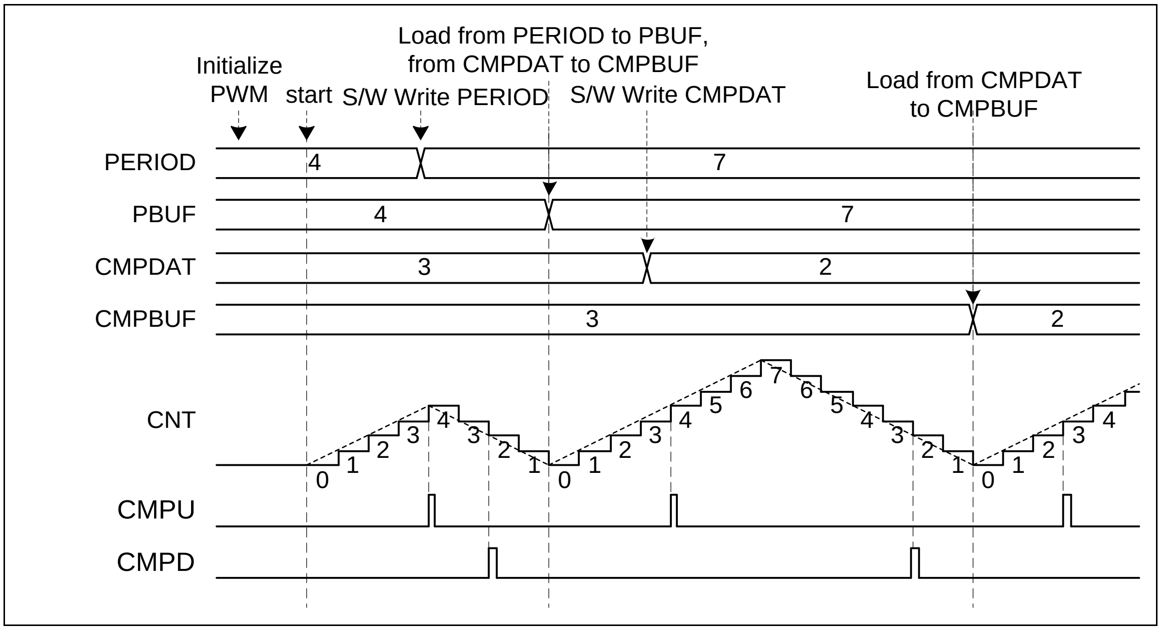


Figure 6.9-11 PWM Double Buffering Illustration

6.9.5.8 Period Loading Mode

Period Loading Mode is the default loading mode. It has lowest priority in loading modes. PERIOD and CMPDAT both will both load to their buffer while a period is completed. For example, after PWM counter up counts from zero to PERIOD in the up-counter operation or down counts from PERIOD to zero in the down-counter operation or up counts from zero to PERIOD and then down counts to zero in up-down counter operation.

Figure 6.9-12 shows period loading timing of up-count operation, where PERIOD DATA0 denotes the initial data of PERIOD, PERIOD DATA1 denotes the first updated PERIOD data by software and so on, CMPDAT also follows this rule. The following describes steps sequence of Figure 6.9-12. User can know the PERIOD and CMPDAT update condition, by watching PWM period and CMPU event.

1. Software writes CMPDAT DATA1 to CMPDAT at point 1.
2. Period loading CMPDAT DATA1 to CMPBUF at the end of PWM period at point 2.
3. Software writes PERIOD DATA1 to PERIOD at point 3.
4. Period loading PERIOD DATA1 to PBUF at the end of PWM period at point 4.
5. Software writes DATA2 to PERIOD at point 5.
6. Period loading DATA2 to PBUF at the end of PWM period at point 6.

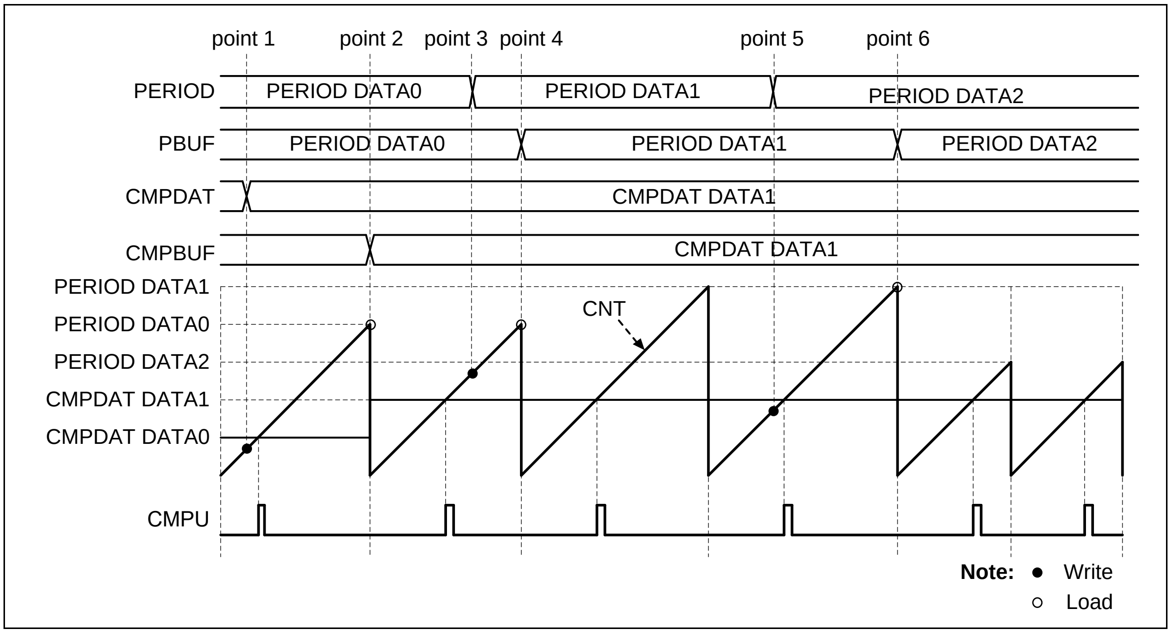


Figure 6.9-12 Period Loading Mode with Up-Counter Type

6.9.5.9 Immediately Loading Mode

If the IMMLDENn (PWM_CTL0[21:16]) bit which corresponds to PWM channel n is set to 1, software will load a value to buffer from PERIOD and CMPDAT immediately while software updates PERIOD or CMPDAT. If the update PERIOD value is less than current counter value, counter will count wraparound. Immediately loading mode has the highest priority. If IMMLDENn has been set, other loading mode for channel n will become invalid. Figure 6.9-13 shows an example and its steps sequence is described below.

1. Software writes CMPDAT DATA1 and hardware immediately loading CMPDAT DATA1 to CMPBUF at point 1.

2. Software writes PERIOD DATA1 which is greater than current counter value at point 2; counter will continue counting until equal to PERIOD DATA1 to finish a period loading.
3. Software writes PERIOD DATA2 which is less than the current counter value at point 3; counter will continue counting to its maximum value 0xFFFF and count wraparound from 0 to PERIOD DATA2 to finish this period loading.

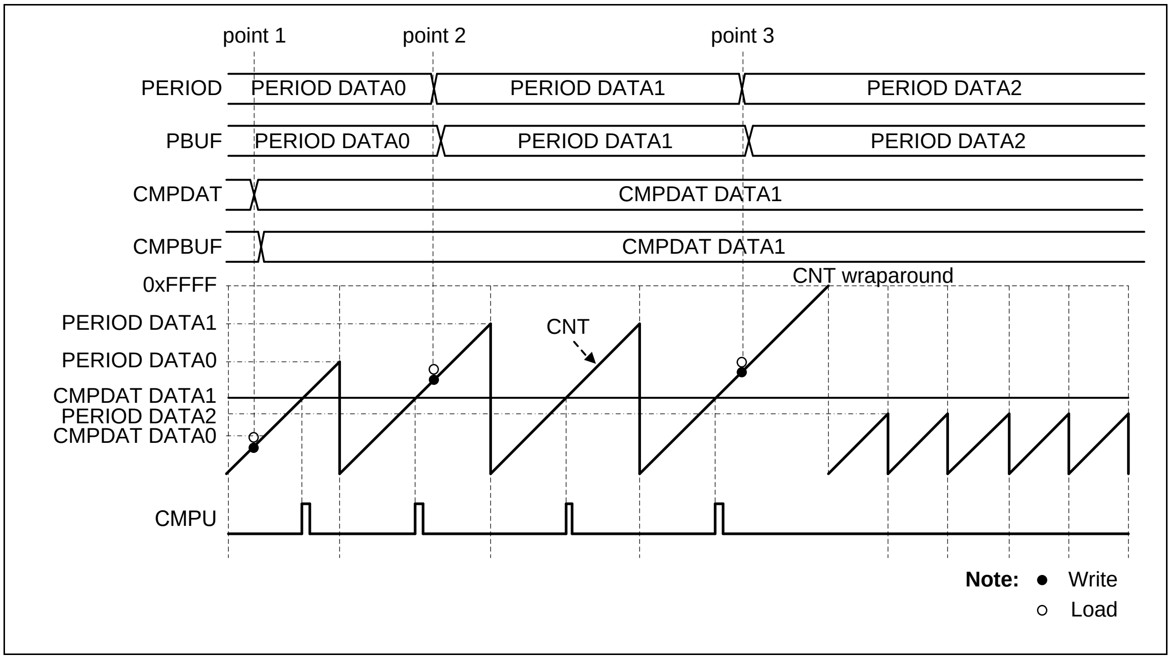


Figure 6.9-13 Immediately Loading Mode with Up-Counter Type

6.9.5.10 *Center Loading Mode*

If the CTRLDn (PWM_CTL0[5:0]) bit which corresponds to PWM channel n is set to 1 and in up-down counter type, CMPDAT will load to CMPBUFn in center of a period, that is, counter counts to PERIOD. PERIOD loading timing is the same as period loading mode. Figure 6.9-14 shows an example and its steps sequence is described below.

1. Software writes CMPDAT DATA1 at point 1.
2. Hardware loads CMPDAT DATA1 to CMPBUF at center of PWM period at point 2.
3. Software writes PERIOD DATA1 at point 3.
4. Hardware loads PERIOD DATA1 to PBUF at the end of PWM period at point 4.
5. Software writes CMPDAT DATA2 at point 5.
6. Hardware loads CMPDAT DATA2 to CMPBUF at center of PWM period at point 6.
7. Software writes PERIOD DATA2 at point 7.
8. Hardware loads PERIOD DATA2 to PBUF at the end of PWM period at point 8.

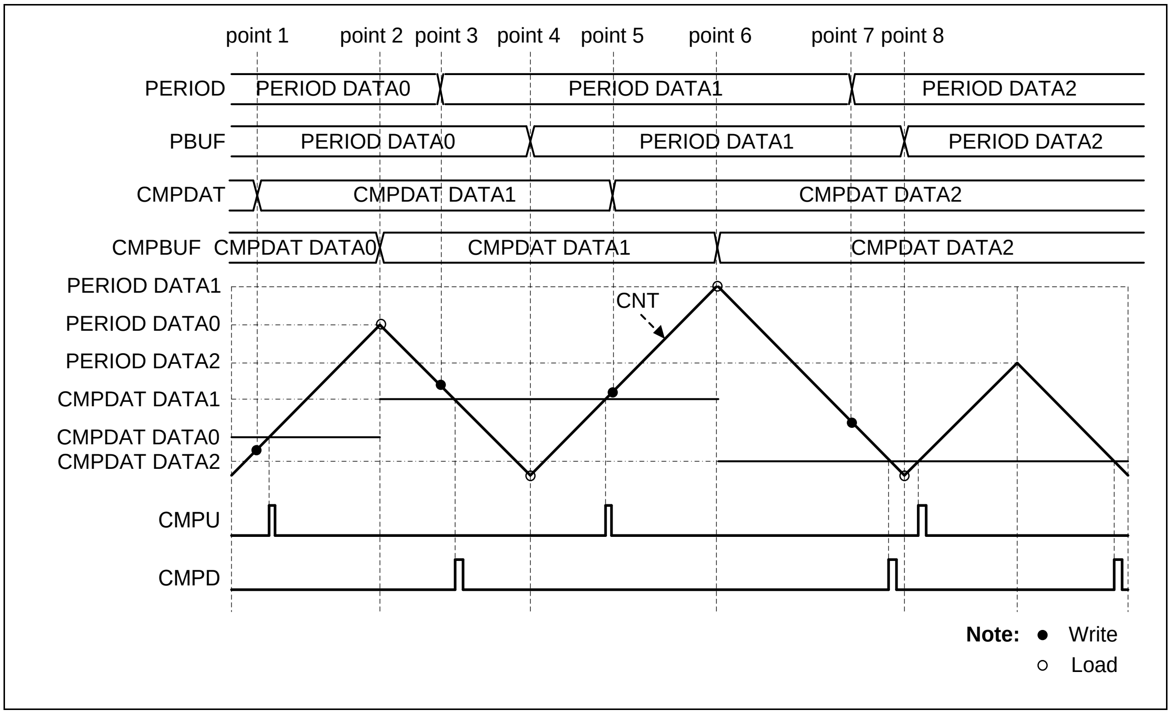


Figure 6.9-14 Center Loading Mode with Up-Down-Counter Type

6.9.5.11 PWM Pulse Generator

The PWM pulse generator uses counter and comparator events to generate PWM pulse. The events are: zero point, period point in up counter type and down counter type, center point in up-down counter type and counter equal to comparator point in three types. As to up-down counter type, there are two counter equal comparator points, one at up count and another at down count. Besides, Complementary mode has two comparators compared with counter, and thus comparing equal points will become four in up-down counter type and two for up or down counter type.

Each event point can decide PWM waveform to do nothing (X), set Low (L), set High (H) or toggle (T) by setting PWM_WGCTL0 and PWM_WGCTL1 registers. Using these points can easily generate asymmetric PWM pulse or variant waveform, as shown in Figure 6.9-15. In the figure, PWM is in Complementary mode. There are two comparators n and m to generate PWM pulse. n denotes even channel number 0, 2, 4, m denotes odd channel number 1, 3, 5. n and m channels are complementary paired. Complementary mode uses two channels (CH0 and CH1, CH2 and CH3, CH4 and CH5) as a pair of PWM outputs to generate complement paired waveforms. CMPU denotes CNT is equal to CMPDAT when counting up. CMPD denotes CNT is equal to CMPDAT when counting down.

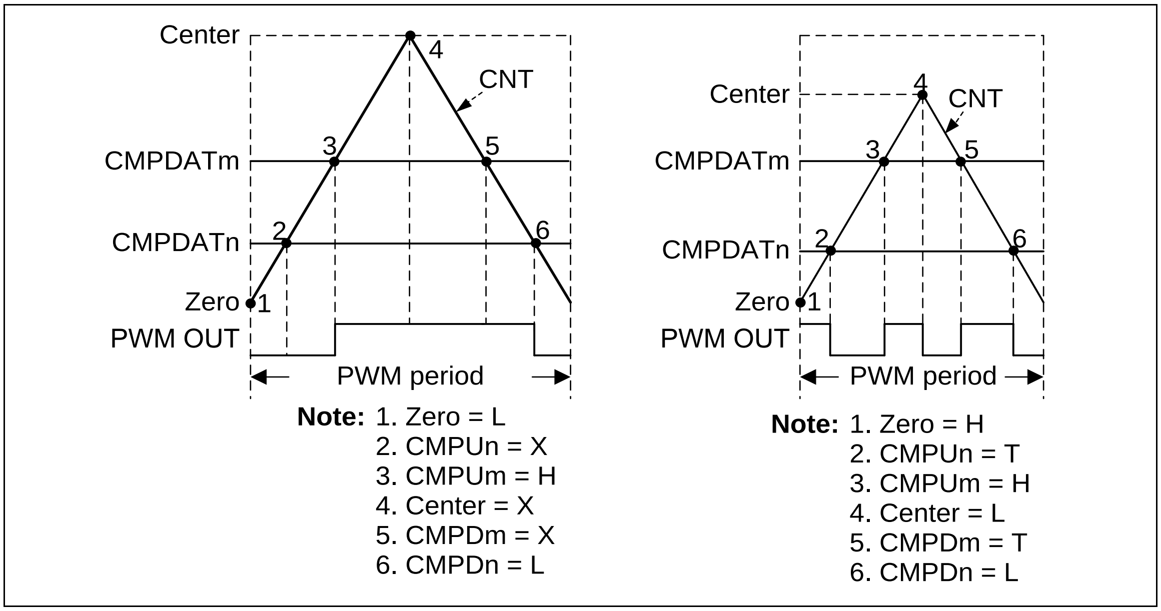


Figure 6.9-15 PWM Pulse Generation

The generation events may be sometimes set to the same value, as the reason, events priority between different counter types are list below, up counter type (Table 6.9-2), down counter type (Table 6.9-3) and up-down counter type (Table 6.9-4). By using event priority, user can easily generate 0% to 100% duty pulse as shown in Figure 6.9-16.

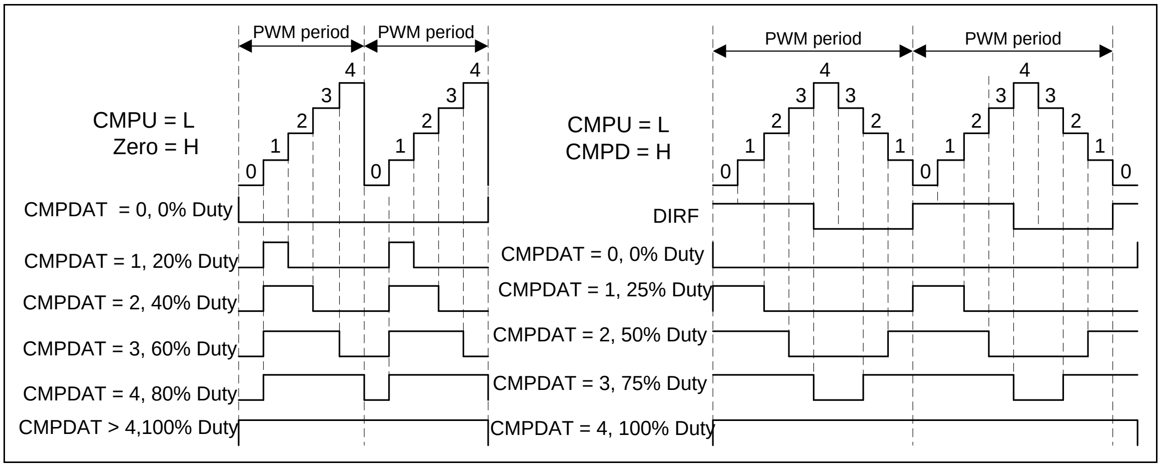


Figure 6.9-16 PWM 0% to 100% Pulse Generation

Priority	Up Event
1 (Highest)	CNT = period (PERIOD)
2	Compare up event of odd channel (CNT = CMPUm)
3	Compare up event of even channel (CNT = CMPUn)
4 (Lowest)	CNT = zero

Table 6.9-2 PWM Pulse Generation Event Priority for Up-Counter

Priority	Down Event
1 (Highest)	CNT = zero
2	Compare up event of odd channel (CNT = CMPDm)
3	Compare up event of even channel (CNT = CMPDn)
4 (Lowest)	CNT = period (PERIOD)

Table 6.9-3 PWM Pulse Generation Event Priority for Down-Counter

Priority	Up Event	Down Event
1 (Highest)	CNT = CMPUm	CNT = CMPDm
2	CNT= CMPUn	CNT = CMPDn
3	CNT = zero	CNT = center (PERIOD)
4	CNT = CMPDm	CNT = CMPUm
5 (Lowest)	PERIOD = CMPDn	CNT = CMPUn

Table 6.9-4 PWM Pulse Generation Event Priority for Up-Down-Counter

6.9.5.12 PWM Output Mode

The PWM supports two output modes: Independent mode which may be applied to DC motor system, Complementary mode with dead-time insertion which may be used in the application of AC induction motor and permanent magnet synchronous motor.

6.9.5.13 Independent mode

By default, the PWM is operating in Independent mode, Independent mode is enabled when channel n corresponding PWMMODEn (PWM_CTL1[26:24]) bit set to 0. In this mode six PWM channels: PWM_CH0, PWM_CH1, PWM_CH2, PWM_CH3, PWM_CH4 and PWM_CH5 are running off its own period and duty as shown in Figure 6.9-17.

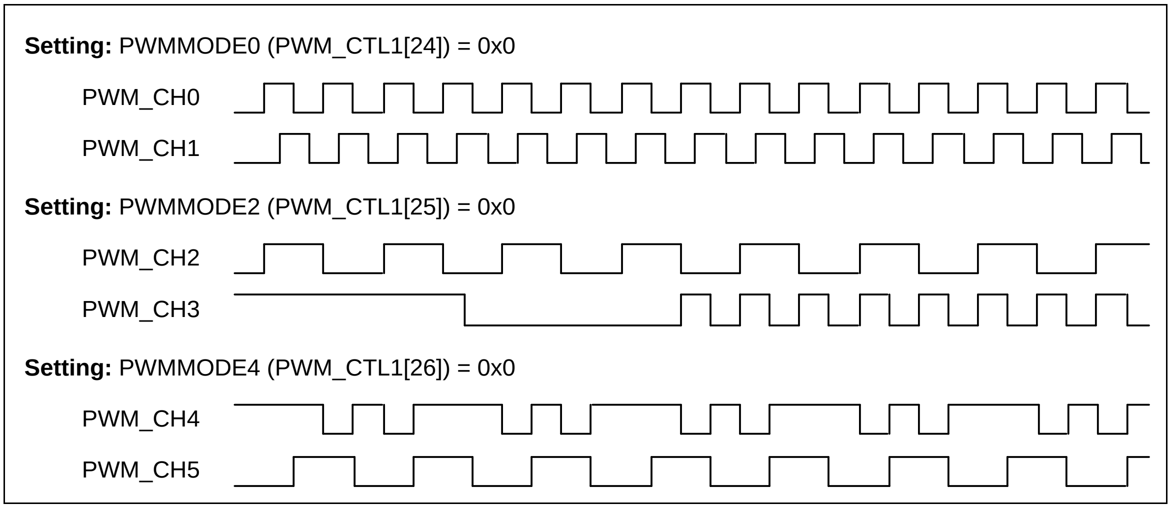

Setting: PWMMODE0 (PWM_CTL1[24]) = 0x0
PWM_CH0
PWM_CH1
Setting: PWMMODE2 (PWM_CTL1[25]) = 0x0
PWM_CH2
PWM_CH3
Setting: PWMMODE4 (PWM_CTL1[26]) = 0x0
PWM_CH4
PWM_CH5

Figure 6.9-17 PWM Independent Mode Waveform

6.9.5.14 Complementary mode

Complementary mode is enabled when the pair channel corresponding PWMMODEn (PWM_CTL1[26:24]) bit set to 1. In this mode there are 3 PWM generators utilized for Complementary mode, with total of 3 PWM output paired pins in this module. In Complimentary mode, the internal odd PWM signal must always be the complement of the corresponding even PWM signal. PWM_CH1 will be the complement of PWM_CH0. PWM_CH3 will be the complement of PWM_CH2 and PWM_CH5 will be the complement of PWM_CH4 as shown in Figure 6.9-18.

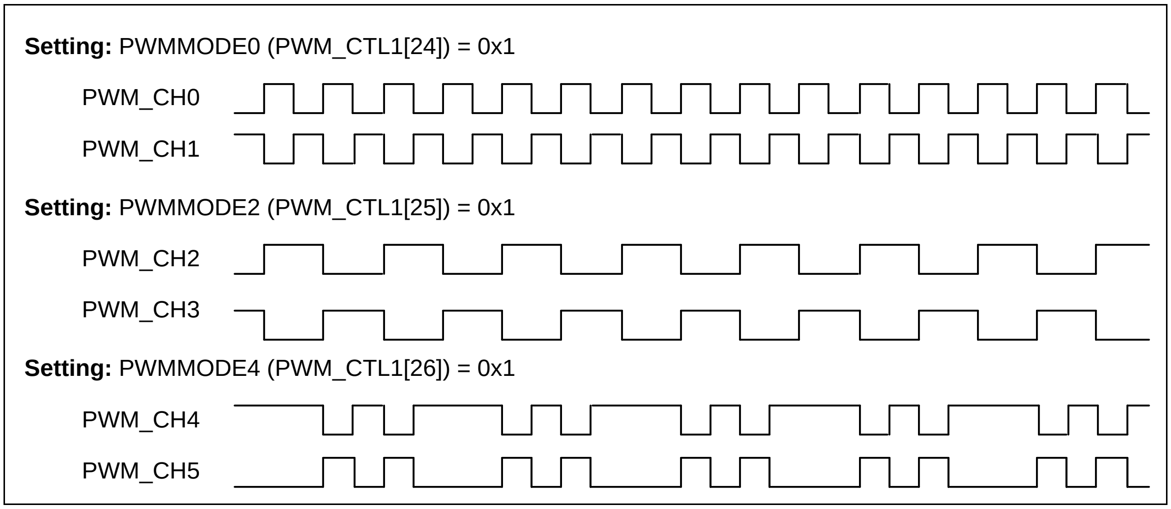

Setting: PWMMODE0 (PWM_CTL1[24]) = 0x1
PWM_CH0
PWM_CH1
Setting: PWMMODE2 (PWM_CTL1[25]) = 0x1
PWM_CH2
PWM_CH3
Setting: PWMMODE4 (PWM_CTL1[26]) = 0x1
PWM_CH4
PWM_CH5

Figure 6.9-18 PWM Complementary Mode Waveform

6.9.5.15 PWM Output Control

After PWM pulse generation, there are four to six steps to control the output of PWM channels. In Independent mode, there are Mask, Brake, Pin Polarity and Output Enable four steps as shown in Figure 6.9-19. In Complementary mode, it needs two more steps to precede these four steps, Complementary channels and Dead-Time Insertion as shown in Figure 6.9-20.

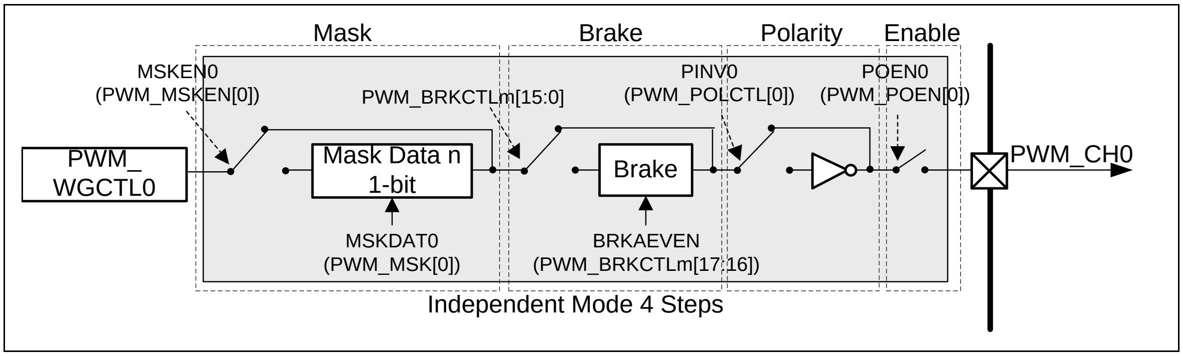


Figure 6.9-19 PWM_CH0 Output Control in Independent Mode

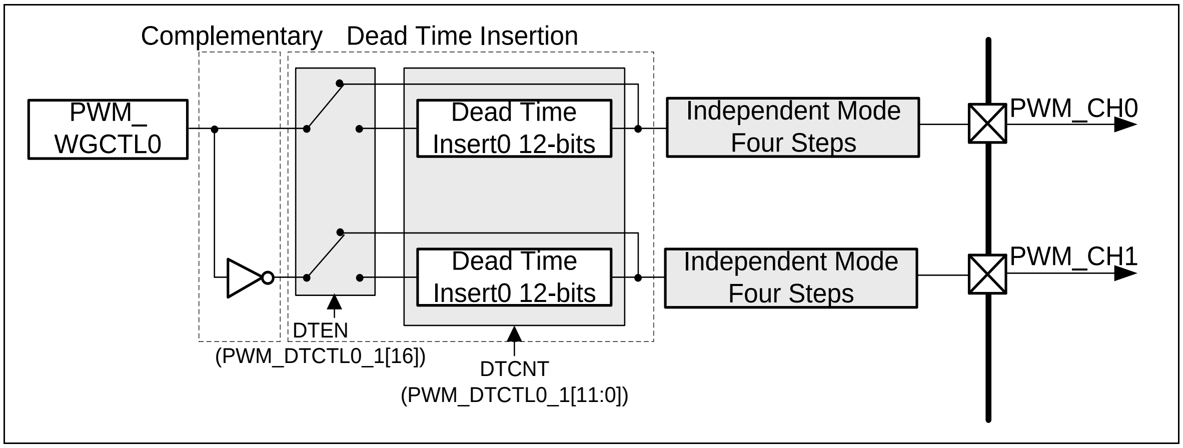


Figure 6.9-20 PWM_CH0 and PWM_CH1 Output Control in Complementary Mode

6.9.5.16 Dead-Time Insertion

In the complementary application, the complement channels may drive the external devices like power switches. The dead-time generator inserts a low level period called “dead-time” between complementary outputs to drive these devices safely and to prevent system or devices from the burn-out damage. Hence the dead-time control is a crucial mechansism to the proper operation of the complementary system. By setting corresponding channel n DTEN (PWM_DTCTLn[16]) bit to enable dead-time function and DTCNT (PWM_DTCTLn[11:0]) to control dead-time period, the dead-time can be calculated from the following formula:

Dead-time = (DTCNT[11:0]+1) * PWMx_CLK period

Figure 6.9-21 indicates the dead-time insertion for one pair of PWM signals.

Dead-time insertion clock source can be selected from prescaler output by setting DTCKSEL (PWM_DTCTLn[24]) to 1. By default, clock source is coming from PWM_CLK, which is prescaler input. Please note that the PWM_DTCTLn is a write-protected register. The dead-time can be calculated from the following formula:

Dead-time = (DTCNT (PWM_DTCTLn[11:0])+1) *

(CLKPSC (PWM_CLKPSCn [11:0])+1)*PWMx_CLK period

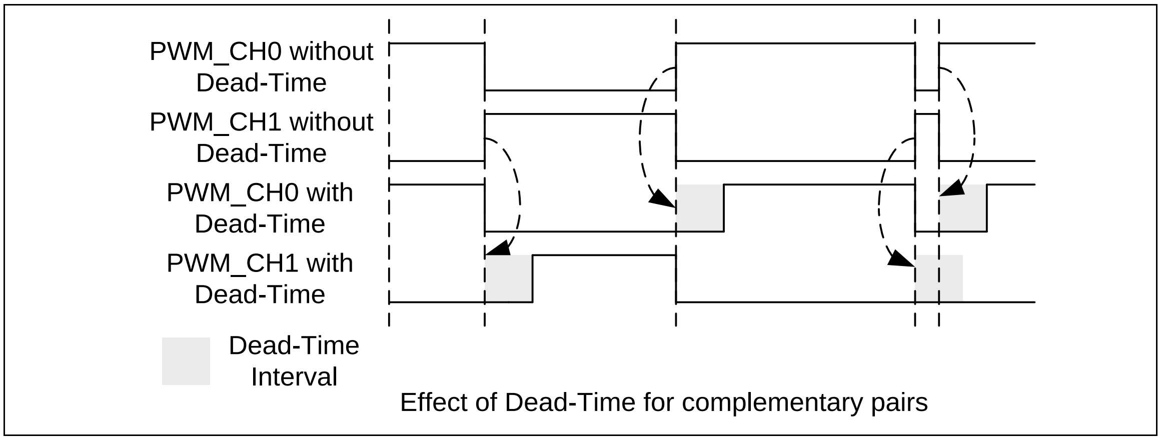

PWM_CH0 without Dead-Time
PWM_CH1 without Dead-Time
PWM_CH0 with Dead-Time
PWM_CH1 with Dead-Time
Dead-Time Interval
Effect of Dead-Time for complementary pairs

Figure 6.9-21 Dead-Time Insertion

6.9.5.17 *PWM Mask Output Function*

Each of the PWM channel output value can be manually overridden with the settings in the PWM Mask Enable Control Register (PWM_MSKEN) and the PWM Masked Data Register (PWM_MSK) With these settings, the PWM channel outputs can be assigned to specified logic states independent of the duty cycle comparison units. The PWM mask bits are useful when controlling various types of Electrically Commutated Motor (ECM) like a BLDC motor. The PWM_MSKEN register contains six bits, MSKENn (PWM_MSKEN[5:0]). If the MASKENn is set to active-high, the PWM channel n output will be overridden. The PWM_MSK register contains six bits, MSKDATn (PWM_MSK[5:0]). The bit value of the MSKDATn determines the state value of the PWM channel n output when the channel is overridden. Figure 6.9-22 shows an example of how PWM mask control can be used for the override feature.

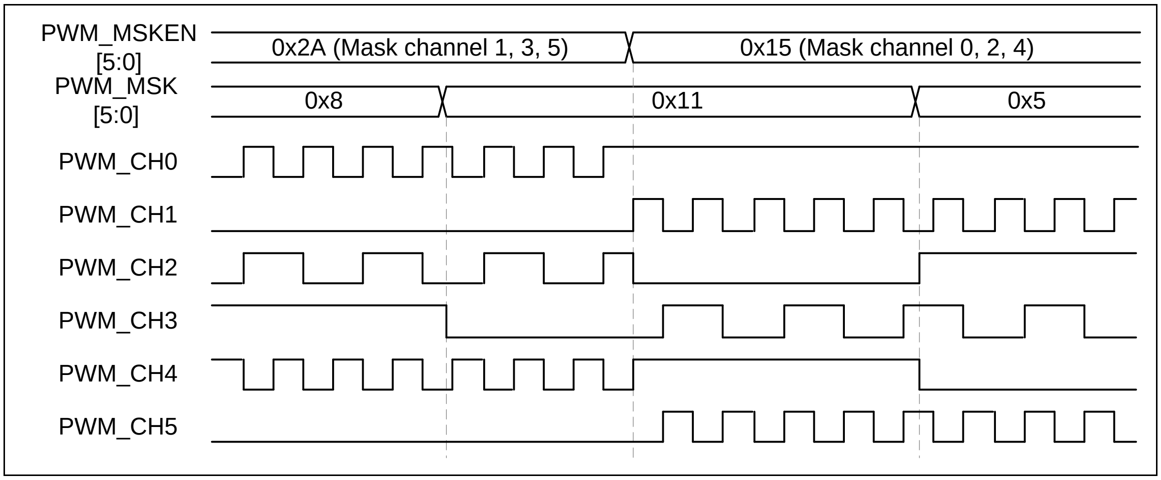


Figure 6.9-22 Illustration of Mask Control Waveform

6.9.5.18 *PWM Brake*

Each PWM module has two external input brake control signals. The external signals will be filtered by a 3-bit noise filter. In addition, it can be inversed by setting the bit BRKxPINV (PWM_BNF[15, 7], x denotes input external pin 0 or 1) to realize the polarity setup for the brake control signals. The noise filter sampling clock can be selected by setting bits BRKxFCS (PWM_BNF[11:9, 3:1]) to fit different noise properties. Moreover, by setting the bits BRKxFCNT

(PWM_BNF[14:12, 6:4]), user can define by how many sampling clock cycles a filter will recognize the effective edge of the brake signal. Configuring the BRKxFEN (PWM_BNF[8, 0]) will enable the noise filter function. By default, it is disabled.

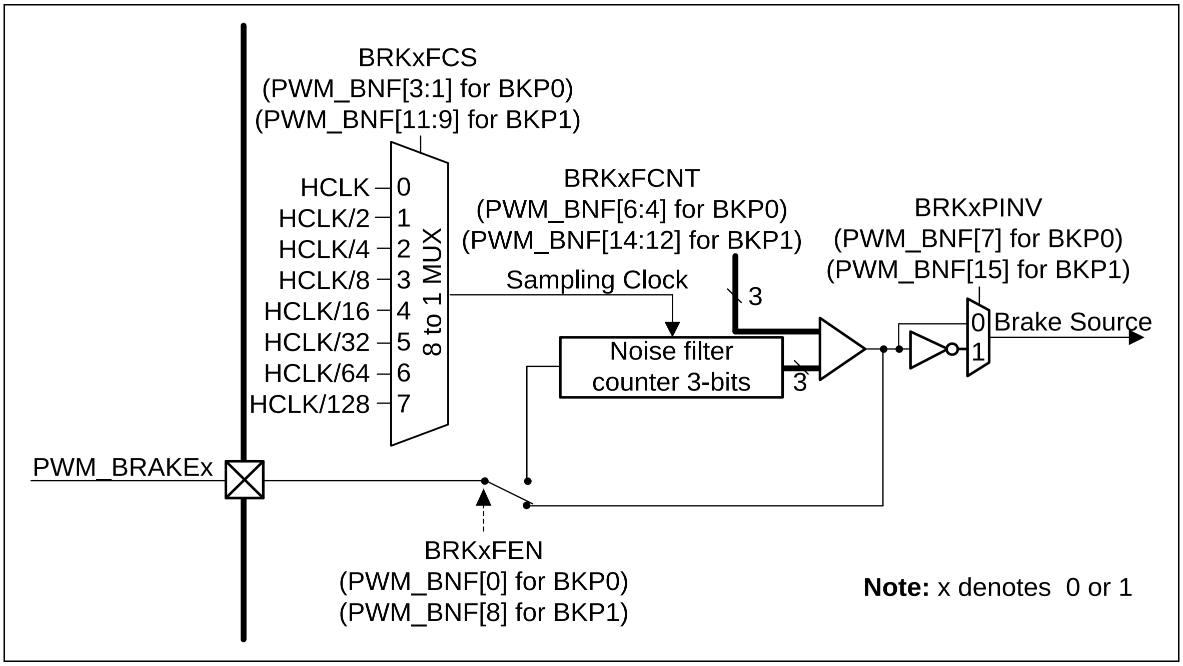


Figure 6.9-23 Brake Noise Filter Block Diagram

Each complementary channel pair shares a PWM brake function, as shown in Figure 6.9-24. To control paired channels to output safety state, user can setup BRKAEVEN (PWM_BRKCTL0_1[17:16]) for even channels and BRKAODD (PWM_BRKCTL0_1[19:18]) for odd channels when the fault brake event happens. There are two brake detectors: Edge detector and Level detector. When the edge detector detects the brake signal and BRKEIENn_m (PWM_INTEN1[2:0]) is enabled, the brake function generates BRK_INT. This interrupt needs software to clear, and the BRKESTSn (PWM_INTSTS1[21:16]) brake state will keep until the next PWM period starts after the interrupt cleared. The brake function can also operate in another way through the level detector. Once the level detector detects the brake signal and the BRKLIENn_m (PWM_INTEN1[10:8]) is also enabled, the brake function will generate BRK_INT, but BRKLSTSn (PWM_INTSTS1[29:24]) brake state will auto recovery to normal output while level brake source recovery to high level and pass through “Low Level Detection” at the PWM waveform period when brake condition removed without clear interrupt.

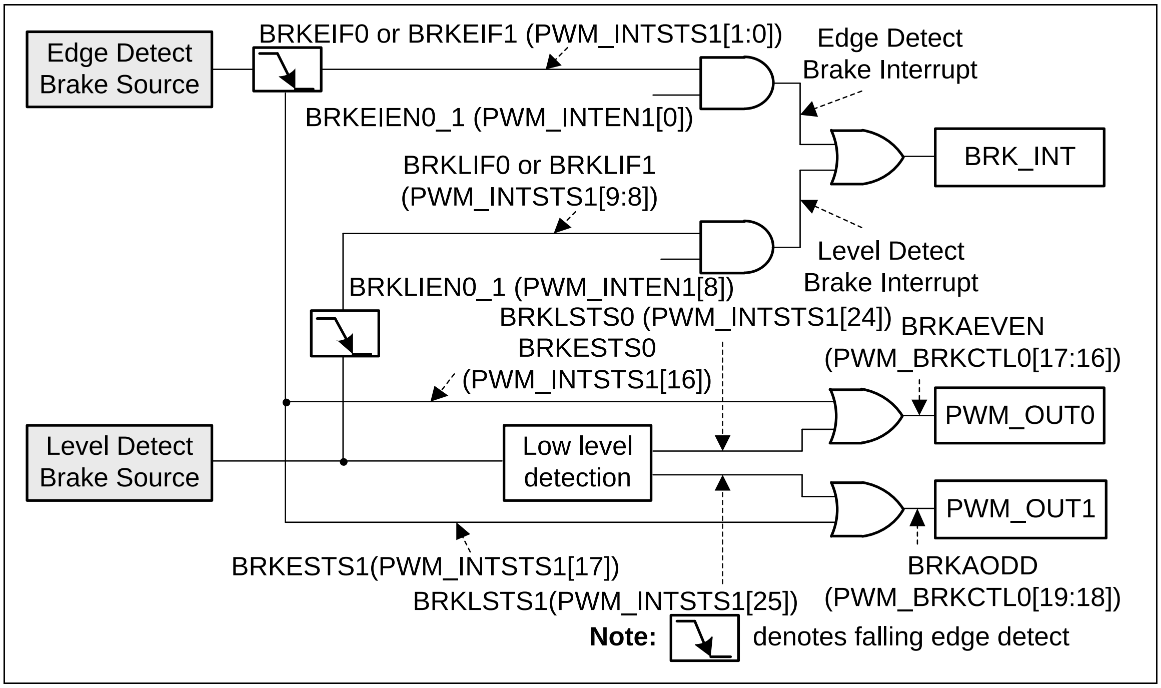


Figure 6.9-24 Brake Block Diagram for PWM_CH0 and PWM_CH1 Pair

Figure 6.9-25 illustrates the edge detector waveform for PWM_CH0 and PWM_CH1 pair. In this case, the edge detect brake source has occurred twice for the brake events. When the event occurs, both of the BRKEIF0 and BRKEIF1 flags are set and BRKESTS0 and BRKESTS1 are also set to indicate brake state of PWM_CH0 and PWM_CH1. For the first occurring event, software writes 1 to clear the BRKEIF0. After that, the BRKESTS0 is cleared by hardware at the next start of the PWM period. At the same moment, the PWM_CH0 outputs the normal waveform even though the brake event is still occurring. The second event also triggers the same flags, but at this time, software writes 1 to clear the BRKEIF1. Afterward, PWM_CH1 outputs normally at the next start of the PWM period.

As a contrast to the edge detector example, Figure 6.9-26 illustrates the level detector waveform for PWM_CH0 and PWM_CH1 pair. In this case, the BRKLIF0 and BRKLIF1 can only indicate the brake event having occurred. The BRKLSTS0 and BRKLSTS1 brake states will automatically recover at the start of the next PWM period no matter at what states the BRKLIF0 and BRKLIF1 are at that moment.

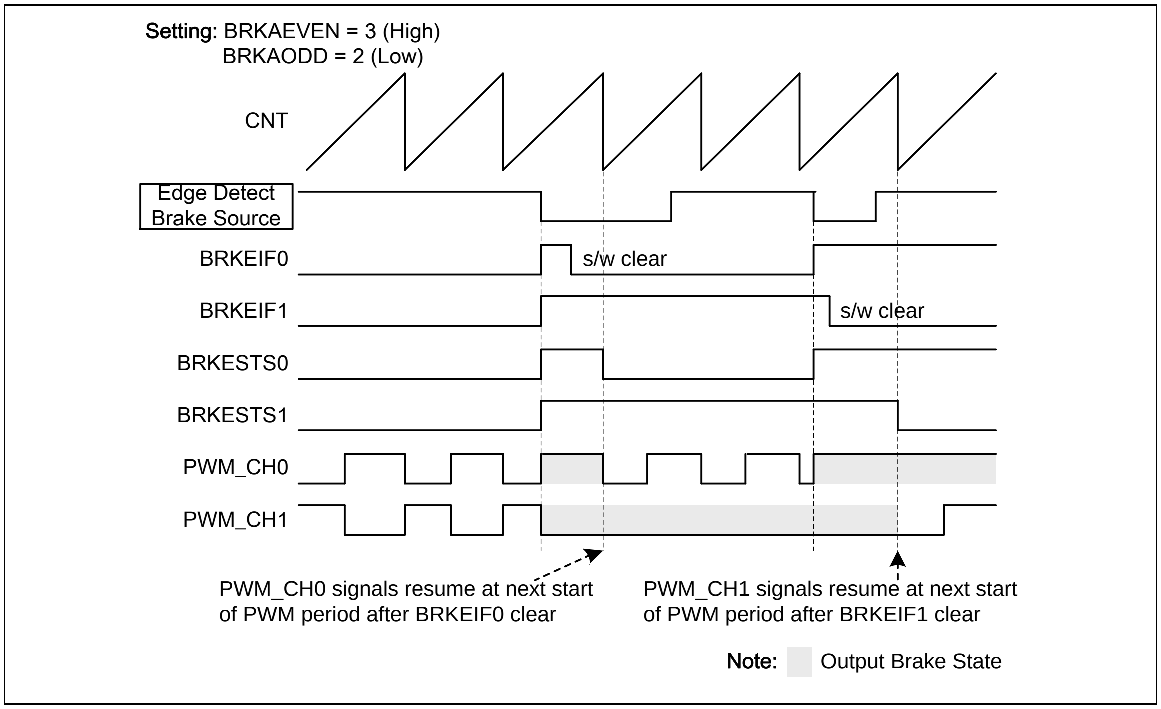

Setting: BRKAEVEN = 3 (High)
BRKAODD = 2 (Low)
CNT
Edge Detect Brake Source
BRKEIF0
s/w clear
BRKEIF1
s/w clear
BRKESTS0
BRKESTS1
PWM_CH0
PWM_CH1
PWM_CH0 signals resume at next start of PWM period after BRKEIF0 clear
PWM_CH1 signals resume at next start of PWM period after BRKEIF1 clear
Note: ▨ Output Brake State

Figure 6.9-25 Edge Detector Waveform for PWM_CH0 and PWM_CH1 Pair

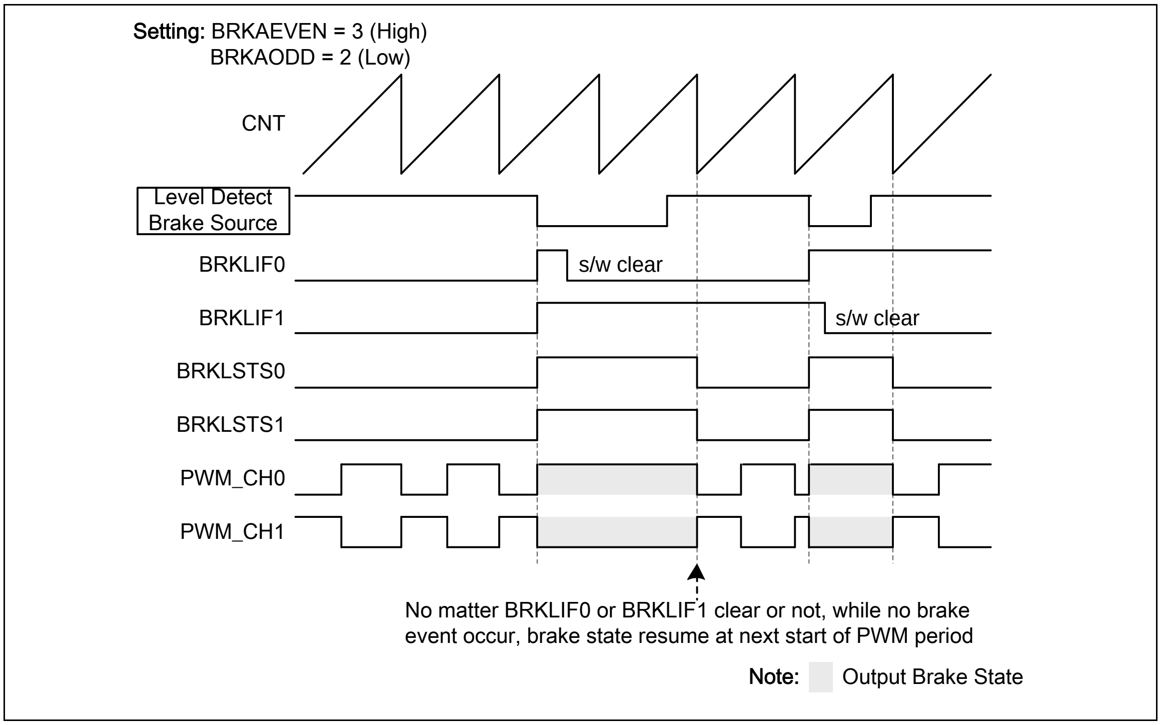

Setting: BRKAEVEN = 3 (High)
BRKAODD = 2 (Low)
CNT
Level Detect Brake Source
BRKLIF0
s/w clear
BRKLIF1
s/w clear
BRKLSTS0
BRKLSTS1
PWM_CH0
PWM_CH1
No matter BRKLIF0 or BRKLIF1 clear or not, while no brake event occur, brake state resume at next start of PWM period
Note: ▨ Output Brake State

Figure 6.9-26 Level Detector Waveform for PWM_CH0 and PWM_CH1 Pair

The two kinds of detectors detect the same three brake sources: two from external input signals and one from system fail but with different brake sources enable. In addition to the three sources, these two detectors have one more brake condition triggered by software, as shown in Figure

6.9-27.

Among the above described brake sources, the brake source coming from system fail can still be specified to several different system fail conditions. These conditions include clock fail, Brown-out detect and Cortex-M0 lockup. Figure 6.9-28 shows that by setting corresponding enable bits, the enabled system fail condition can be one of the sources to issue the Brake system fail to the PWM brake.

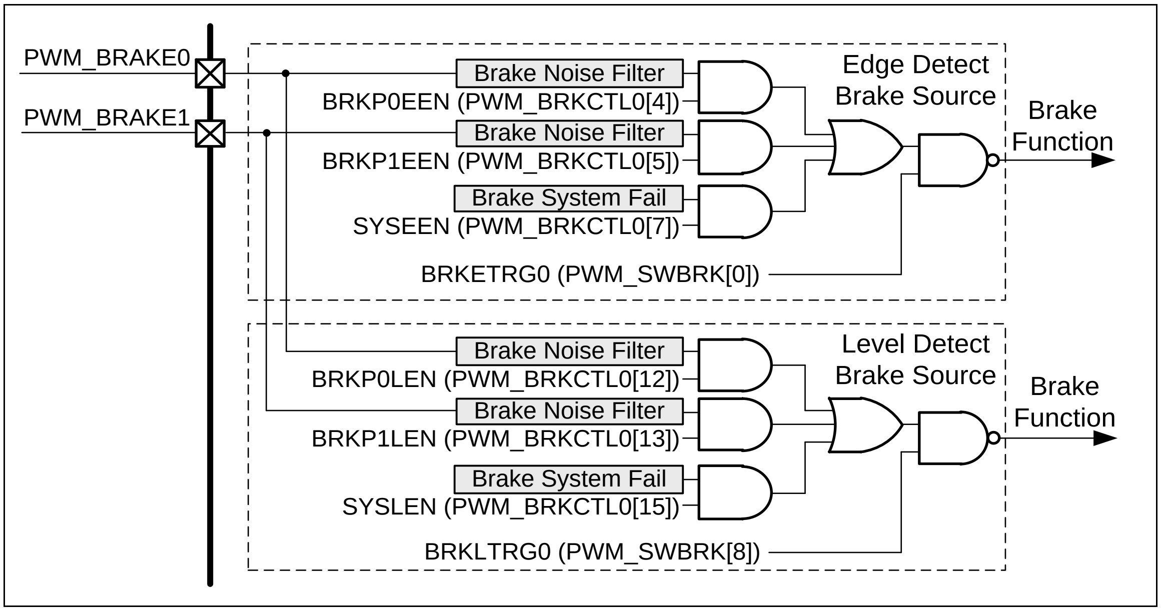


Figure 6.9-27 Brake Source Block Diagram for PWM_CH0 and PWM_CH1 Pair

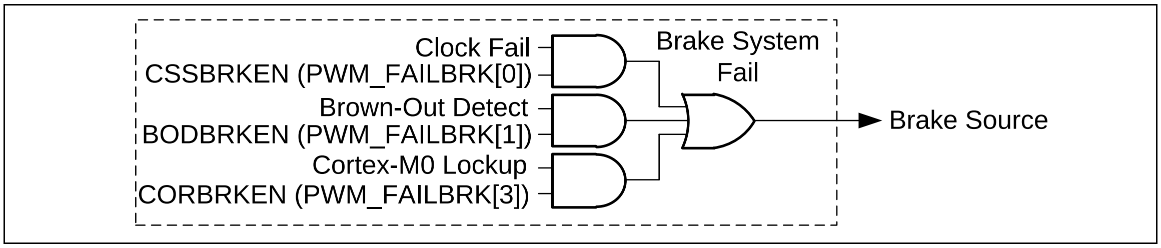
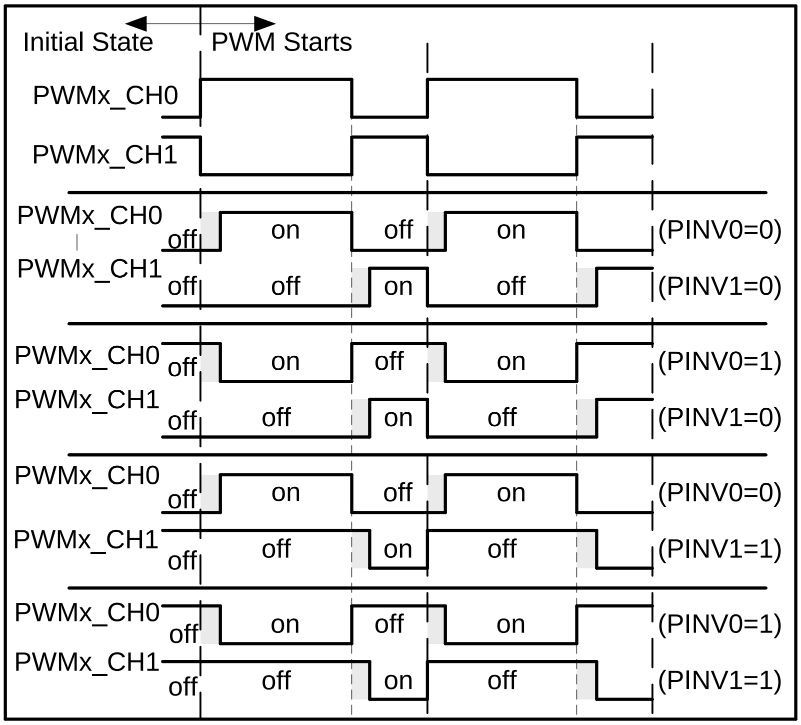

Clock Fail
CSSBRKEN (PWM_FAILBRK[0])
Brown-Out Detect
BODBRKEN (PWM_FAILBRK[1])
Cortex-M0 Lockup
CORBRKEN (PWM_FAILBRK[3])
Brake System
Fail
Brake Source

Figure 6.9-28 Brake System Fail Block Diagram

6.9.5.19 Polarity Control

Each PWM port, from PWM_CH0 to PWM_CH5, has an independent polarity control module to configure the polarity of the active state of PWM output. By default, the PWM output is active high. This implies the PWM OFF state is low and ON state is high. This definition is variable through setting the PWM Negative Polarity Control Register (PWM_POLCTL), for each individual PWM channel. Figure 6.9-29 shows the initial state before PWM starting with different polarity settings.



▨ Dead-time insertion; It is only effective in complementary mode
Note: PINVx: Negative Polarity control bits; It controls the PWM output initial state and polarity, x denotes 0 or 1.

Figure 6.9-29 Initial State and Polarity Control with Rising Edge Dead-Time Insertion

6.9.5.20 Synchronous start function

The synchronous start function can be enabled when SSEN0 (PWM_SSCTL[0]) is set. User can select synchronous source which is from PWM0, PWM1, BPWM0, or BPWM1 by SSRC (PWM_SSCTL[9:8]). The selected PWM or BPWM channels (include channel0 to channel5 of each BPWM or PWM) will start counting at the same time once the synchronous start function is enabled and set CNTSEN (PWM_SSTRG). It is noted that set CNTSEN (PWM_SSTRG) will also set the counter enable bit (CNTENn, n denotes channel 0 to 5) to start counting.

6.9.5.21 PWM Interrupt Generator

There are three independent interrupts for each PWM as shown in Figure 6.9-30.

The 1st PWM interrupt (PWM_INT) comes from PWM complementary pair events. The counter can generate the Zero point Interrupt Flag ZIFn (PWM_INTSTS0[n], n=0,2,4) and the Period point Interrupt Flag PIFn (PWM_INTSTS0[n+8], n=0,2,4). When PWM channel n's counter equals to the comparator value stored in PWM_CMPDATn, the different interrupt flags will be triggered depending on the counting direction. If the matching occurs at up-count direction, the Up Interrupt Flag CMPUIFn (PWM_INTSTS0[21:16]) is set and if matching at the opposite direction, the Down Interrupt Flag CMPDIFn (PWM_INTSTS0[29:24]) is set. Channel n's complementary channel m's comparator also generates the CMPUIFm and CMPDIFm in the same way. If the correspond interrupt enable bits are set, the trigger events will generates interrupt signals.

The 2nd interrupt is the capture interrupt (CAP_INT). It shares the PWM_INT vector in NVIC. The CAP_INT can be generated when the CRLIFn (PWM_CAPIF[5:0]) is triggered and the Capture Rising Interrupt Enable bit CAPRIENn (PWM_CAPIEN[5:0]) is set to 1. Or in the falling edge condition, the CFLIFn (PWM_CAPIF[13:8]) can be triggered when the Capture Falling Interrupt

Enable bit CAPFIENn (PWM_CAPIEN[13:8]) is set to 1.

The last one is the brake interrupt (BRK_INT). The detail of the BRK_INT is described in the PWM Brake section.

Figure 6.9-30 demonstrates the architecture of the PWM interrupts.

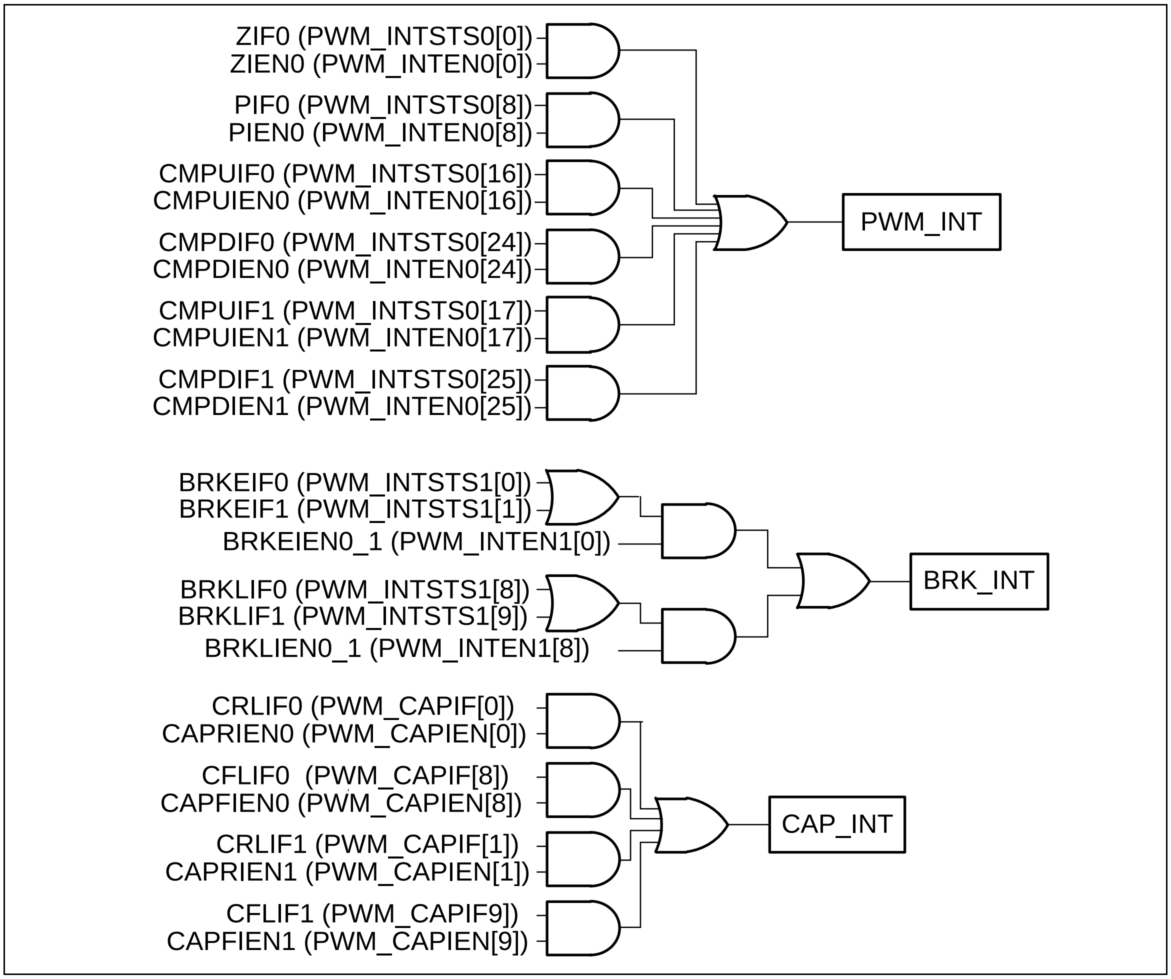


Figure 6.9-30 PWM_CH0 and PWM_CH1 Pair Interrupt Architecture Diagram

6.9.5.22 PWM Trigger ADC Generator

PWM can be one of the ADC conversion trigger source. Each PWM pair channels share the same trigger source. Setting TRGSELn is to select the trigger sources, where TRGSELn is TRGSEL0, TRGSEL1, …, and TRGSEL5, which are located in PWM_ADCTS0[3:0], PWM_ADCTS0[11:8], PWM_ADCTS0[19:16], PWM_ADCTS0[27:24], PWM_ADCTS1[3:0] and PWM_EADTS1[11:8], respectively. Setting TRGENn is to enable the trigger output to ADC, where TRGENn is TRGEN0, TRGEN1, …, TRGEN5, which are located in PWM_ADCTS0[7], PWM_ADCTS0[15], PWM_ADCTS0[23], PWM_ADCTS0[31], PWM_ADCTS1[7] and PWM_ADCTS1[15], respectively. The number n (n = 0,1, ..,5) denotes PWM channel number.

There are 7 PWM events can be selected as the trigger source for one pair of channels. Figure 6.9-31 is an example of PWM_CH0 and PWM_CH1. PWM can trigger ADC to start conversion in different timings by setting PERIOD and CMPDAT. Figure 6.9-32 is the trigger ADC timing waveform in the up-down counter type.

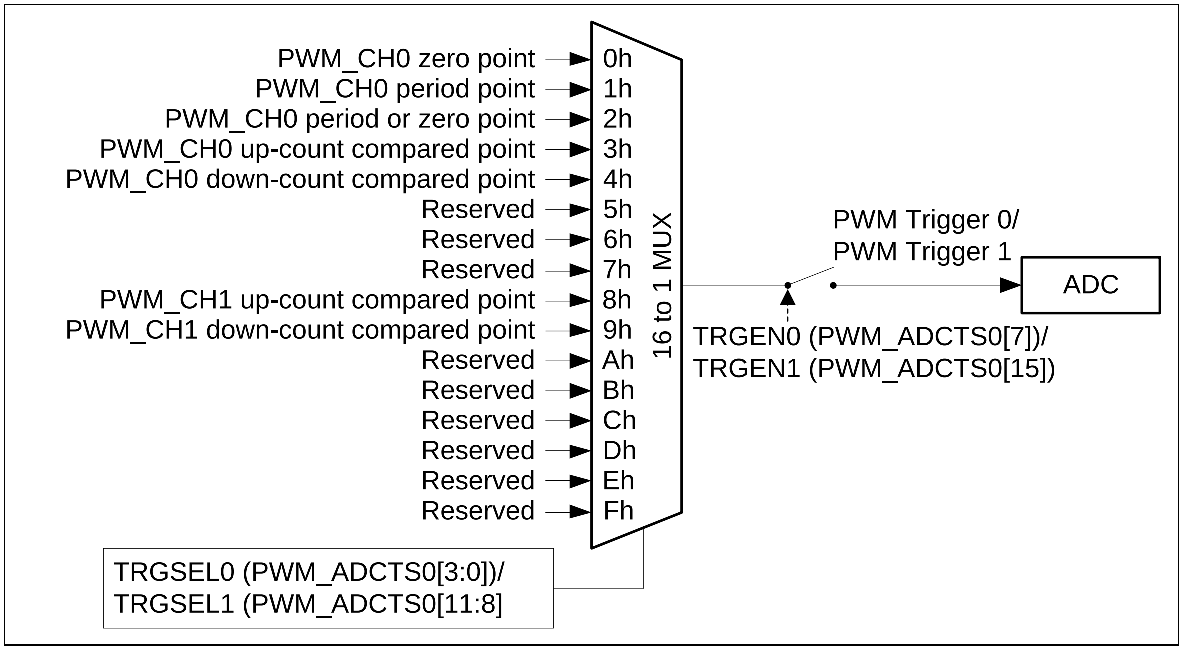


Figure 6.9-31 PWM_CH0 and PWM_CH1 Pair Trigger ADC Block Diagram

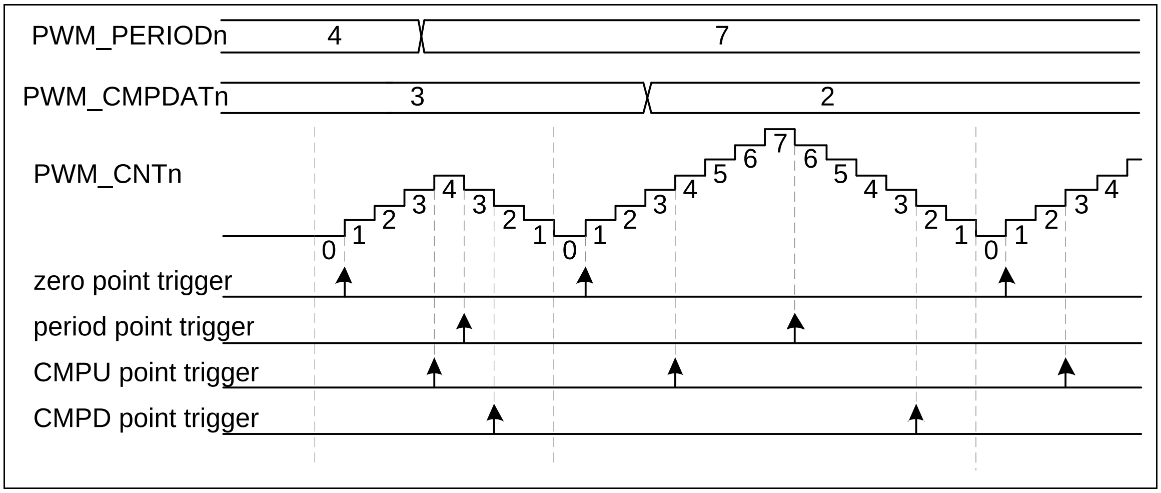


Figure 6.9-32 PWM Trigger ADC in Up-Down Counter Type Timing Waveform

6.9.5.23 Capture Operation

The channels of the capture input and the PWM output share the same pin and counter. The counter can operate in up or down counter type. The capture function will always latch the PWM counter to the register RCAPDATn (PWM_RCAPDATn[15:0]) or the register FCAPDATn (PWM_FCAPDATn[15:0]) if the input channel has a rising transition or a falling transition, respectively. The capture function will also generate an interrupt CAP_INT (using PWM_INT vector) if the rising or falling latch occurs and the corresponding channel n’s rising or falling interrupt enable bits are set, where the CAPRIENn (PWM_CAPIEN[5:0]) is for the rising edge and the CAPFIENn (PWM_CAPIEN[13:8]) is for the falling edge. When rising or falling latch occurs, the corresponding PWM counter may be reloaded with the value PWM_PERIODn, depending on the setting of RCRLDENn or FCRLDENn (where RCRLDENn and FCRLDENn are located at

PWM_CAPCTL[21:16] and PWM_CAPCTL[29:24], respectively). Note that the corresponding GPIO pins must be configured as the capture function by enable the CAPINENn (PWM_CAPINEN[5:0]) for the corresponding capture channel n. Figure 6.9-33 is the capture block diagram of channel 0.

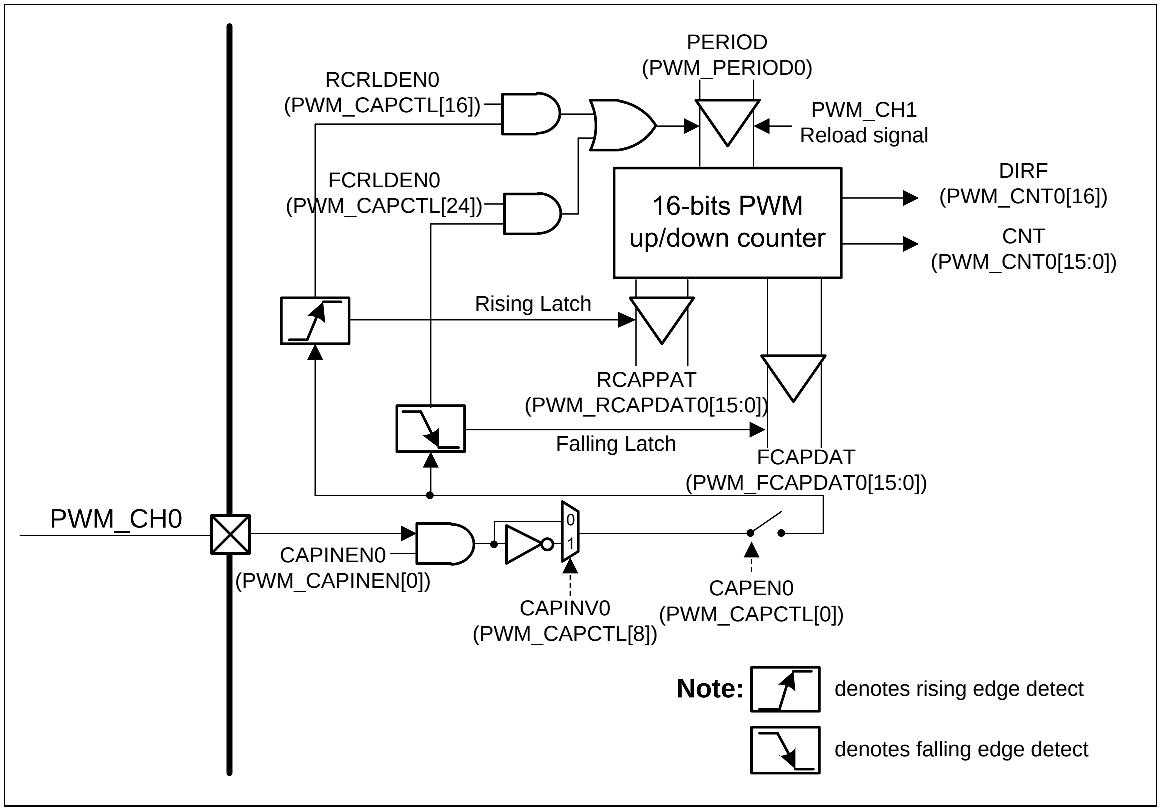


Figure 6.9-33 PWM_CH0 Capture Block Diagram

Figure 6.9-34 illustrates the capture function timing. In this case, the capture counter is set as PWM down counter type and the PERIOD is set to 8 so that the counter counts in the down direction, from 8 to 0. When detecting a falling edge at the capture input pin, the capture function latches counter value to the PWM_FCAPDATn. When detecting the rising edge, it latches the counter value to the PWM_RCAPDATn. In this timing diagram, when the falling edge is detected at the first time, the capture function will reload the counter value from the PERIOD setting because the FCRLDENn is enabled. But at the second time, the falling edge does not result in a reload because of the disabled FCRLDENn. In this example, the counter also reloads at the rising edge of the capture input because the RCRLDENn is enabled, too.

Moreover, if the case is setup as the up counter type, the counter will reload the value zero and count up to the value PERIOD. It is important that the counter is shared by two complement channels, and thus the counter reloads time also controlled by another channel’s reload signal.

Figure 6.9-34 also illustrates the timing example for the interrupt and interrupt flag generation. When the rising edge at channel n is detected, the corresponding bit CRLIFn (PWM_CAPIF[5:0]) is set by hardware. Similarly, a falling edge detection at chnnel n causes the corresponding bit CFLIFn (PWM_CAPIF[13:8]) set by hardware. CRLIFn and CFLIFn can be cleared by software by writing ‘1’. If the CRLIFn is set and the CRLIENn is enabled, the capture function generates an interrupt. If the CFLIFn is set and the CAPFIENn is enabled, the interrupt also happens.

A condition which is not shown in this figure is: if the rising latch happens again when the CRLIF is already set, the Over run status CRLIFOVn (PWM_CAPSTS[5:0]) will be set to 1 by hardware to indicate the CRLIF overrunning. Also, if the falling latch happens again, the same hardware operation occurs for the interrupt flag CFLIF and the Over run status CFLIFOVn (PWM_CAPSTS[13:8]).

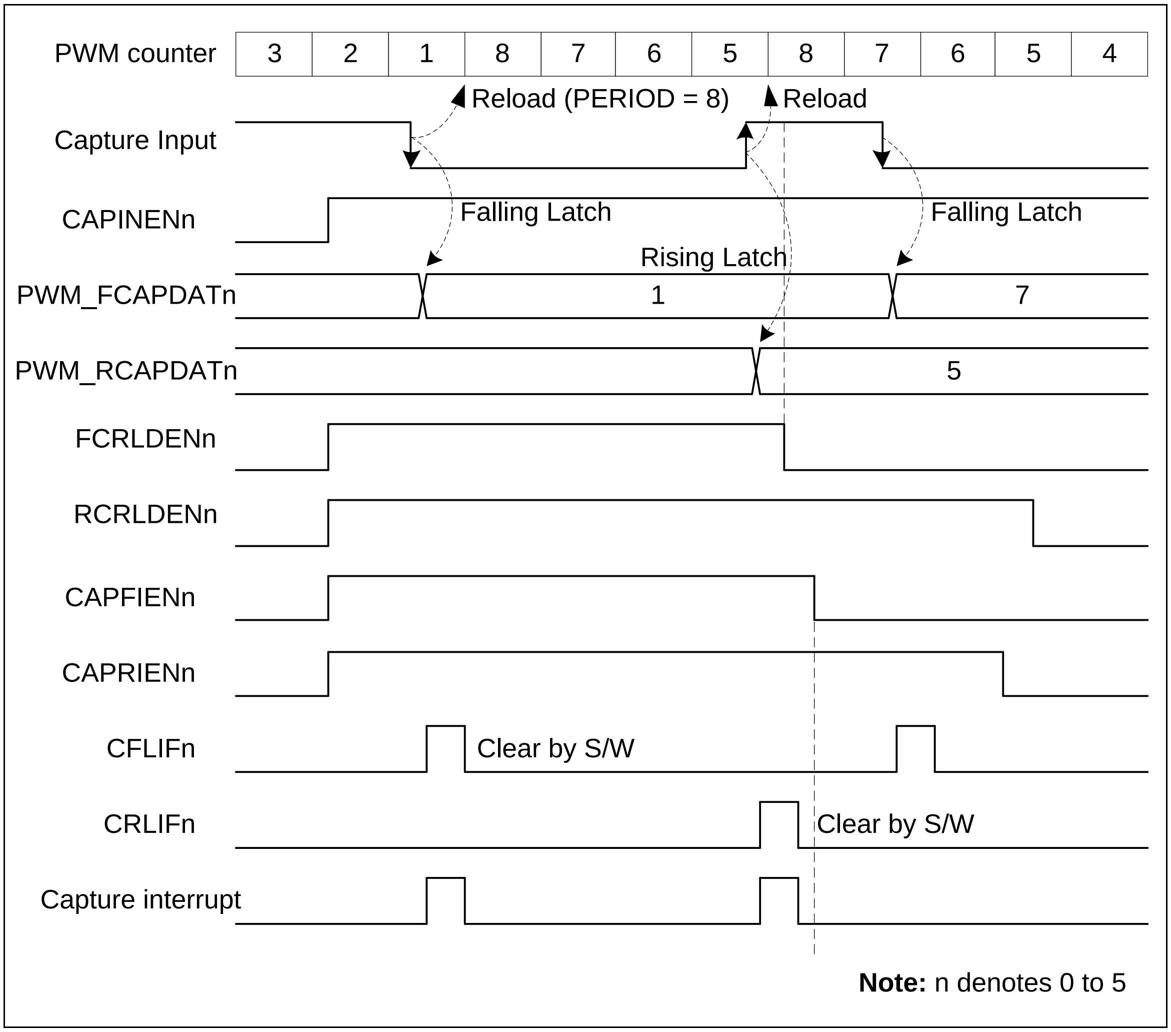


Figure 6.9-34 Capture Operation Waveform

The capture pulse width can be calculated according to the following formula:

For the negative pulse case, the channel low pulse width is calculated as (PWM_PERIODn + 1 - PWM_RCAPDATn). In Figure 6.9-34 case, the low pulse width is 8+1-5 = 4

For the positive pulse case, the channel high pulse width is calculated as (PWM_PERIODn + 1 - PWM_FCAPDATn). In Figure 6.9-34 case, high pulse width is 8+1-7 = 2

6.9.5.24 Capture PDMA Function

The PWM module supports the PDMA transfer function when operating in the capture mode. When the corresponding PDMA enable bit CHENn_m (CHEN0_1 at PWM_PDMACTL[0], CHEN2_3 at PWM_PDMACTL[8] and CHEN4_5 at PWM_PDMACTL[16], where n and m denote complement pair channels) is set, the capture module will issue a request to PDMA controller when the preceding capture event has happened. The PDMA controller will issue an acknowledgement to the capture module after it has read back the CAPBUF (PWM_PDMACAPn_m[15:0], n, m denotes complement pair channels) register in the capture module and has sent the register value to the memory. By setting CAPMODn_m (CAPMOD0_1 at PWM_PDMACTL[2:1], CAPMOD2_3 at PWM_PDMACTL[10:9] and CAPMOD4_5 at PWM_PDMACTL[18:17]), the PDMA can transfer the rising edge captured data or falling edge captured data or both of them to the memory. When using the PDMA to transfer both of the falling and rising edge data, remember to set CAPORDn_m (CAPORD0_1 at PWM_PDMACTL[3], CAPORD2_3 at PWM_PDMACTL[11] and CAPORD4_5 at PWM_PDMACTL[19]) to decide the order of the transferred data (falling edge captured is first or rising edge captured first). The complement pair channels share a PDMA channel. Therefore, a selection bit CHSELn_m (CHSEL0_1 (PWM_PDMACTL[4]), CHSEL2_3 (PWM_PDMACTL[12]) and CHSEL4_5 (PWM_PDMACTL[20])) is used to decide either channel n or channel m can be serviced by the PDMA channel.

Figure 6.9-35 is capture PDMA waveform. In this case, the CHSEL0_1 (PWM_PDMACTL[4]) is set to 0. Hence the PDMA will service channel 0 for the capture data transfer. CAPMOD0_1 (PWM_PDMACTL[2:1]) is set to 3. That means both of the rising and falling edge captured data will be transferred to the memory. The CAPORD0_1 (PWM_PDMACTL[3]) is set to 1, and thus the rising edge data will be the first data to transfer and following is the falling edge data to transfer. As shown in Figure 6.9-35, the last assertions of the CAPRIF0 CRLIF0 and CAPFIF0 CFLIF0 signal have some overlap. The PWM_RCAPDAT0 value 11 will be loaded to PWM_PDMACAP0_1 to wait for transfer but not the PWM_FCAPDAT0 value 6. The PWM_PDMACAP0_1 saves the data which will be transferred to the memory by PDMA. The HWDATA in this figure denotes the data which are being transferred by PDMA.

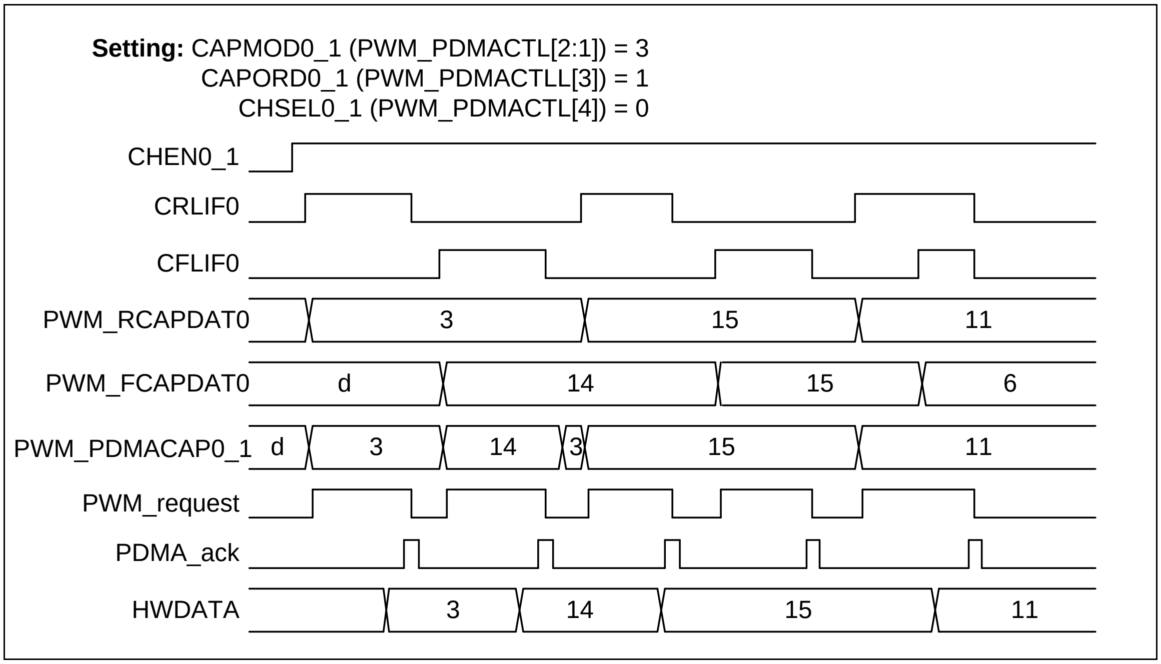


Figure 6.9-35 Capture PDMA Operation Waveform of Channel 0

6.9.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
PWM Base Address: **PWM0_BA = 0x4004_4000** **PWM1_BA = 0x4014_4000**				
PWM_CTL0 **x=0, 1**	PWMx_BA+0x00	R/W	PWM Control Register 0	0x0000_0000
PWM_CTL1 **x=0, 1**	PWMx_BA+0x04	R/W	PWM Control Register 1	0x0000_0000
PWM_CLKSRC **x=0, 1**	PWMx_BA+0x10	R/W	PWM Clock Source Register	0x0000_0000
PWM_CLKPSC0_1 **x=0, 1**	PWMx_BA+0x14	R/W	PWM Clock Pre-scale Register 0_1	0x0000_0000
PWM_CLKPSC2_3 **x=0, 1**	PWMx_BA+0x18	R/W	PWM Clock Pre-scale Register 2_3	0x0000_0000
PWM_CLKPSC4_5 **x=0, 1**	PWMx_BA+0x1C	R/W	PWM Clock Pre-scale Register 4_5	0x0000_0000
PWM_CNTEN **x=0, 1**	PWMx_BA+0x20	R/W	PWM Counter Enable Register	0x0000_0000
PWM_CNTCLR **x=0, 1**	PWMx_BA+0x24	R/W	PWM Clear Counter Register	0x0000_0000
PWM_PERIOD0 **x=0, 1**	PWMx_BA+0x30	R/W	PWM Period Register 0	0x0000_0000
PWM_PERIOD2 **x=0, 1**	PWMx_BA+0x38	R/W	PWM Period Register 2	0x0000_0000
PWM_PERIOD4 **x=0, 1**	PWMx_BA+0x40	R/W	PWM Period Register 4	0x0000_0000
PWM_CMPDAT0 **x=0, 1**	PWMx_BA+0x50	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1 **x=0, 1**	PWMx_BA+0x54	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2 **x=0, 1**	PWMx_BA+0x58	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3 **x=0, 1**	PWMx_BA+0x5C	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CMPDAT4 **x=0, 1**	PWMx_BA+0x60	R/W	PWM Comparator Register 4	0x0000_0000
PWM_CMPDAT5 **x=0, 1**	PWMx_BA+0x64	R/W	PWM Comparator Register 5	0x0000_0000

PWM_DTCTL0_1 **x=0, 1**	PWMx_BA+0x70	R/W	PWM Dead-Time Control Register 0_1	0x0000_0000
PWM_DTCTL2_3 **x=0, 1**	PWMx_BA+0x74	R/W	PWM Dead-Time Control Register 2_3	0x0000_0000
PWM_DTCTL4_5 **x=0, 1**	PWMx_BA+0x78	R/W	PWM Dead-Time Control Register 4_5	0x0000_0000
PWM_CNT0 **x=0, 1**	PWMx_BA+0x90	R	PWM Counter Register 0	0x0000_0000
PWM_CNT2 **x=0, 1**	PWMx_BA+0x98	R	PWM Counter Register 2	0x0000_0000
PWM_CNT4 **x=0, 1**	PWMx_BA+0xA0	R	PWM Counter Register 4	0x0000_0000
PWM_WGCTL0 **x=0, 1**	PWMx_BA+0xB0	R/W	PWM Generation Register 0	0x0000_0000
PWM_WGCTL1 **x=0, 1**	PWMx_BA+0xB4	R/W	PWM Generation Register 1	0x0000_0000
PWM_MSKEN **x=0, 1**	PWMx_BA+0xB8	R/W	PWM Mask Enable Register	0x0000_0000
PWM_MSK **x=0, 1**	PWMx_BA+0xBC	R/W	PWM Mask Data Register	0x0000_0000
PWM_BNF **x=0, 1**	PWMx_BA+0xC0	R/W	PWM Brake Noise Filter Register	0x0000_0000
PWM_FAILBRK **x=0, 1**	PWMx_BA+0xC4	R/W	PWM System Fail Brake Control Register	0x0000_0000
PWM_BRKCTL0_1 **x=0, 1**	PWMx_BA+0xC8	R/W	PWM Brake Edge Detect Control Register 0_1	0x0000_0000
PWM_BRKCTL2_3 **x=0, 1**	PWMx_BA+0xCC	R/W	PWM Brake Edge Detect Control Register 2_3	0x0000_0000
PWM_BRKCTL4_5 **x=0, 1**	PWMx_BA+0xD0	R/W	PWM Brake Edge Detect Control Register 4_5	0x0000_0000
PWM_POLCTL **x=0, 1**	PWMx_BA+0xD4	R/W	PWM Pin Polar Inverse Register	0x0000_0000
PWM_POEN **x=0, 1**	PWMx_BA+0xD8	R/W	PWM Output Enable Register	0x0000_0000
PWM_SWBRK **x=0, 1**	PWMx_BA+0xDC	W	PWM Software Brake Control Register	0x0000_0000
PWM_INTEN0 **x=0, 1**	PWMx_BA+0xE0	R/W	PWM Interrupt Enable Register 0	0x0000_0000
PWM_INTEN1 **x=0, 1**	PWMx_BA+0xE4	R/W	PWM Interrupt Enable Register 1	0x0000_0000
PWM_INTSTS0	PWMx_BA+0xE8	R/W	PWM Interrupt Flag Register 0	0x0000_0000

x=0, 1				
PWM_INTSTS1 **x=0, 1**	PWMx_BA+0xEC	R/W	PWM Interrupt Flag Register 1	0x0000_0000
PWM_ADCTS0 **x=0, 1**	PWMx_BA+0xF8	R/W	PWM Trigger ADC Source Select Register 0	0x0000_0000
PWM_ADCTS1 **x=0, 1**	PWMx_BA+0xFC	R/W	PWM Trigger ADC Source Select Register 1	0x0000_0000
PWM_SSCTL **x=0, 1**	PWMx_BA+0x110	R/W	PWM Synchronous Start Control Register	0x0000_0000
PWM_SSTRG **x=0, 1**	PWMx_BA+0x114	W	PWM Synchronous Start Trigger Register	0x0000_0000
PWM_STATUS **x=0, 1**	PWMx_BA+0x120	R/W	PWM Status Register	0x0000_0000
PWM_CAPINEN **x=0, 1**	PWMx_BA+0x200	R/W	PWM Capture Input Enable Register	0x0000_0000
PWM_CAPCTL **x=0, 1**	PWMx_BA+0x204	R/W	PWM Capture Control Register	0x0000_0000
PWM_CAPSTS **x=0, 1**	PWMx_BA+0x208	R	PWM Capture Status Register	0x0000_0000
PWM_RCAPDAT0 **x=0, 1**	PWMx_BA+0x20C	R	PWM Rising Capture Data Register 0	0x0000_0000
PWM_FCAPDAT0 **x=0, 1**	PWMx_BA+0x210	R	PWM Falling Capture Data Register 0	0x0000_0000
PWM_RCAPDAT1 **x=0, 1**	PWMx_BA+0x214	R	PWM Rising Capture Data Register 1	0x0000_0000
PWM_FCAPDAT1 **x=0, 1**	PWMx_BA+0x218	R	PWM Falling Capture Data Register 1	0x0000_0000
PWM_RCAPDAT2 **x=0, 1**	PWMx_BA+0x21C	R	PWM Rising Capture Data Register 2	0x0000_0000
PWM_FCAPDAT2 **x=0, 1**	PWMx_BA+0x220	R	PWM Falling Capture Data Register 2	0x0000_0000
PWM_RCAPDAT3 **x=0, 1**	PWMx_BA+0x224	R	PWM Rising Capture Data Register 3	0x0000_0000
PWM_FCAPDAT3 **x=0, 1**	PWMx_BA+0x228	R	PWM Falling Capture Data Register 3	0x0000_0000
PWM_RCAPDAT4 **x=0, 1**	PWMx_BA+0x22C	R	PWM Rising Capture Data Register 4	0x0000_0000
PWM_FCAPDAT4 **x=0, 1**	PWMx_BA+0x230	R	PWM Falling Capture Data Register 4	0x0000_0000
PWM_RCAPDAT5 **x=0, 1**	PWMx_BA+0x234	R	PWM Rising Capture Data Register 5	0x0000_0000

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

PWM_FCAPDAT5 x=0, 1	PWMx_BA+0x238	R	PWM Falling Capture Data Register 5	0x0000_0000
PWM_PDMACTL x=0, 1	PWMx_BA+0x23C	R/W	PWM PDMA Control Register	0x0000_0000
PWM_PDMACAP0_1 x=0, 1	PWMx_BA+0x240	R	PWM Capture Channel 01 PDMA Register	0x0000_0000
PWM_PDMACAP2_3 x=0, 1	PWMx_BA+0x244	R	PWM Capture Channel 23 PDMA Register	0x0000_0000
PWM_PDMACAP4_5 x=0, 1	PWMx_BA+0x248	R	PWM Capture Channel 45 PDMA Register	0x0000_0000
PWM_CAPIEN x=0, 1	PWMx_BA+0x250	R/W	PWM Capture Interrupt Enable Register	0x0000_0000
PWM_CAPIF x=0, 1	PWMx_BA+0x254	R/W	PWM Capture Interrupt Flag Register	0x0000_0000
PWM_PBUF0 x=0, 1	PWMx_BA+0x304	R	PWM PERIOD0 Buffer	0x0000_0000
PWM_PBUF2 x=0, 1	PWMx_BA+0x30C	R	PWM PERIOD2 Buffer	0x0000_0000
PWM_PBUF4 x=0, 1	PWMx_BA+0x314	R	PWM PERIOD4 Buffer	0x0000_0000
PWM_CMPBUF0 x=0, 1	PWMx_BA+0x31C	R	PWM CMPDAT0 Buffer	0x0000_0000
PWM_CMPBUF1 x=0, 1	PWMx_BA+0x320	R	PWM CMPDAT1 Buffer	0x0000_0000
PWM_CMPBUF2 x=0, 1	PWMx_BA+0x324	R	PWM CMPDAT2 Buffer	0x0000_0000
PWM_CMPBUF3 x=0, 1	PWMx_BA+0x328	R	PWM CMPDAT3 Buffer	0x0000_0000
PWM_CMPBUF4 x=0, 1	PWMx_BA+0x32C	R	PWM CMPDAT4 Buffer	0x0000_0000
PWM_CMPBUF5 x=0, 1	PWMx_BA+0x330	R	PWM CMPDAT5 Buffer	0x0000_0000

6.9.7 Register Description

PWM Control Register 0 (PWM_CTL0)

Register	Offset	R/W	Description	Reset Value
PWM_CTL0 **x=0, 1**	PWMx_BA+0x00	R/W	PWM Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	DBGHALT	Reserved					
23	22	21	20	19	18	17	16
Reserved		IMMLDEN5	IMMLDEN4	IMMLDEN3	IMMLDEN2	IMMLDEN1	IMMLDEN0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CTRLD5	CTRLD4	CTRLD3	CTRLD2	CTRLD1	CTRLD0

Bits	Description	
[31]	**DBGTRIOFF**	**ICE Debug Mode Acknowledge Disable (Write Protect)** 0 = ICE debug mode acknowledgement affects PWM output. PWM pin will be forced as tri-state while ICE debug mode acknowledged. 1 = ICE debug mode acknowledgement disabled. PWM pin will keep output no matter ICE debug mode acknowledged or not. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[30]	**DBGHALT**	**ICE Debug Mode Counter Halt (Write Protect)** If counter halt is enabled, PWM all counters will keep current value until exit ICE debug mode. 0 = ICE debug mode counter halt disable. 1 = ICE debug mode counter halt enable. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[29:22]	**Reserved**	Reserved.
[n+16] n=0,1…5	**IMMLDENn**	**Immediately Load Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = PERIOD will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the end point or center point of each period by setting CTRLD bit. 1 = PERIOD/CMPDAT will load to PBUF and CMPBUF immediately when software update PERIOD/CMPDAT. **Note:** If IMMLDENn is enabled, WINLDENn and CTRLDn will be invalid.
[15:6]	**Reserved**	Reserved.
[n] n=0,1…5	**CTRLDn**	**Center Load Enable Bits** 0 = Center Lodaing mode is disable for corresponding PWM channel n. 1 = Center Lodaing mode is enable for corresponding PWM channel n.

		Each bit n controls the corresponding PWM channel n. In up-down counter type, PERIOD will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the center point of a period.

PWM Control Register 1 (PWM_CTL1)

Register	Offset	R/W	Description	Reset Value
PWM_CTL1 **x=0, 1**	PWMx_BA+0x04	R/W	PWM Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					**PWMMODE4**	**PWMMODE2**	**PWMMODE0**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**CNTTYPE4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CNTTYPE2**		**Reserved**		**CNTTYPE0**	

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:24]	**PWMMODEn**	**PWM Mode** Each bit n controls the corresponding PWM channel n. 0 = PWM independent mode. 1 = PWM complementary mode. **Note:** When operating in group function, these bits must all set to the same mode.
[23:10]	**Reserved**	Reserved.
[9:8]	**CNTTYPE4**	**PWM Counter Behavior Type 4** The two bits control channel5 and channel4 00 = Up counter type (supports in capture mode). 01 = Down count type (supports in capture mode). 10 = Up-down counter type. 11 = Reserved.
[7:6]	**Reserved**	Reserved.
[5:4]	**CNTTYPE2**	**PWM Counter Behavior Type 2** The two bits control channel3 and channel2 00 = Up counter type (supports in capture mode). 01 = Down count type (supports in capture mode). 10 = Up-down counter type. 11 = Reserved.
[3:2]	**Reserved**	Reserved.
[1:0]	**CNTTYPE0**	**PWM Counter Behavior Type 0** The two bits control channel1 and channel0 00 = Up counter type (supports in capture mode). 01 = Down count type (supports in capture mode).

		10 = Up-down counter type. 11 = Reserved.

PWM Clock Source Register (PWM_CLKSRC)

Register	Offset	R/W	Description	Reset Value
PWM_CLKSRC **x=0, 1**	PWMx_BA+0x10	R/W	PWM Clock Source Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**ECLKSRC4**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**ECLKSRC2**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ECLKSRC0**		

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18:16]	**ECLKSRC4**	**PWM_CH45 External Clock Source Select** 000 = PWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.
[15:11]	**Reserved**	Reserved.
[10:8]	**ECLKSRC2**	**PWM_CH23 External Clock Source Select** 000 = PWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.
[7:3]	**Reserved**	Reserved.
[2:0]	**ECLKSRC0**	**PWM_CH01 External Clock Source Select** 000 = PWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.

PWM Clock Pre-Scale Register 0_1, 2_3, 4_5 (PWM_CLKPSC0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_CLKPSC0_1 **x=0, 1**	PWMx_BA+0x14	R/W	PWM Clock Pre-scale Register 0_1	0x0000_0000
PWM_CLKPSC2_3 **x=0, 1**	PWMx_BA+0x18	R/W	PWM Clock Pre-scale Register 2_3	0x0000_0000
PWM_CLKPSC4_5 **x=0, 1**	PWMx_BA+0x1C	R/W	PWM Clock Pre-scale Register 4_5	0x0000_0000

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**CLKPSC**	**PWM Counter Clock Pre-scale** The clock of PWM counter is decided by clock prescaler. Each PWM pair share one PWM counter clock prescaler. The clock of PWM counter is divided by (CLKPSC+ 1).

PWM Counter Enable Register (PWM_CNTEN)

Register	Offset	R/W	Description	Reset Value
PWM_CNTEN **x=0, 1**	PWMx_BA+0x20	R/W	PWM Counter 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			**CNTEN4**	**Reserved**	**CNTEN2**	**Reserved**	**CNTEN0**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**CNTEN4**	**PWM Counter Enable 4** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[3]	**Reserved**	Reserved.
[2]	**CNTEN2**	**PWM Counter Enable 2** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[1]	**Reserved**	Reserved.
[0]	**CNTEN0**	**PWM Counter Enable 0** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.

PWM Clear Counter Register (PWM_CNTCLR)

Register	Offset	R/W	Description	Reset Value
PWM_CNTCLR **x=0, 1**	PWMx_BA+0x24	R/W	PWM Clear 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			CNTCLR4	Reserved	CNTCLR2	Reserved	CNTCLR0

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**CNTCLR4**	**Clear PWM Counter Control Bit 4** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[3]	**Reserved**	Reserved.
[2]	**CNTCLR2**	**Clear PWM Counter Control Bit 2** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[1]	**Reserved**	Reserved.
[0]	**CNTCLR0**	**Clear PWM Counter Control Bit 0** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.

PWM Period Register 0, 2, 4 (PWM_PERIOD0, 2, 4)

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD0 **x=0, 1**	PWMx_BA+0x30	R/W	PWM Period Register 0	0x0000_0000
PWM_PERIOD2 **x=0, 1**	PWMx_BA+0x38	R/W	PWM Period Register 2	0x0000_0000
PWM_PERIOD4 **x=0, 1**	PWMx_BA+0x40	R/W	PWM Period Register 4	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**PWM Period Register** Up-Count mode: In this mode, PWM counter counts from 0 to PERIOD, and restarts from 0. Down-Count mode: In this mode, PWM counter counts from PERIOD to 0, and restarts from PERIOD. PWM period time = (PERIOD+1) * PWM_CLK period. Up-Down-Count mode: In this mode, PWM counter counts from 0 to PERIOD, then decrements to 0 and repeats again. PWM period time = 2 * PERIOD * PWM_CLK period.

PWM Comparator Register 0~5 (PWM_CMPDAT0~5)

Register	Offset	R/W	Description	Reset Value
PWM_CMPDAT0 **x=0, 1**	PWMx_BA+0x50	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1 **x=0, 1**	PWMx_BA+0x54	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2 **x=0, 1**	PWMx_BA+0x58	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3 **x=0, 1**	PWMx_BA+0x5C	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CMPDAT4 **x=0, 1**	PWMx_BA+0x60	R/W	PWM Comparator Register 4	0x0000_0000
PWM_CMPDAT5 **x=0, 1**	PWMx_BA+0x64	R/W	PWM Comparator Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMP**	**PWM Comparator Register** CMP use to compare with CNT to generate PWM waveform, interrupt and trigger ADC. In independent mode, PWM_CMPDAT0~5 denote as 6 independent PWM_CH0~5 compared point. In complementary mode, PWM_CMPDAT0, 2, 4 denote as first compared point, and PWM_CMPDAT1, 3, 5 denote as second compared point for the corresponding 3 complementary pairs PWM_CH0 and PWM_CH1, PWM_CH2 and PWM_CH3, PWM_CH4 and PWM_CH5.

PWM Dead-Time Control Register 0_1, 2_3, 4_5 (PWM_DTCTL0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_DTCTL0_1 **x=0, 1**	PWMx_BA+0x70	R/W	PWM Dead-Time Control Register 0_1	0x0000_0000
PWM_DTCTL2_3 **x=0, 1**	PWMx_BA+0x74	R/W	PWM Dead-Time Control Register 2_3	0x0000_0000
PWM_DTCTL4_5 **x=0, 1**	PWMx_BA+0x78	R/W	PWM Dead-Time Control Register 4_5	0x0000_0000

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**DTCKSEL**	**Dead-time Clock Select (Write Protect)** 0 = Dead-time clock source from PWM_CLK. 1 = Dead-time clock source from prescaler output. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[23:17]	**Reserved**	Reserved.
[16]	**DTEN**	**Enable Dead-time Insertion for PWM Pair (PWM_CH0, PWM_CH1) (PWM_CH2, PWM_CH3) (PWM_CH4, PWM_CH5) (Write Protect)** Dead-time insertion is only active when this pair of complementary PWM is enabled. If dead- time insertion is inactive, the outputs of pin pair are complementary without any delay. 0 = Dead-time insertion Disabled on the pin pair. 1 = Dead-time insertion Enabled on the pin pair. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[15:12]	**Reserved**	Reserved.
[11:0]	**DTCNT**	**Dead-time Counter (Write Protect)** The dead-time can be calculated from the following formula: Dead-time = (DTCNT[11:0]+1) * PWM_CLK period. If DTCKSEL bit is 0. Dead-time = (DTCNT+1) * (CLKPSC (PWM_CLKPSCn [11:0])+1)*PWMx_CLK period, If DTCKSEL bit is 1. **Note:** These bits are write protected. Refer to SYS_REGLCTL register.

PWM Counter Register 0, 2, 4 (PWM_CNT0, 2, 4)

Register	Offset	R/W	Description	Reset Value
PWM_CNT0 **x=0, 1**	PWMx_BA+0x90	R	PWM Counter Register 0	0x0000_0000
PWM_CNT2 **x=0, 1**	PWMx_BA+0x98	R	PWM Counter Register 2	0x0000_0000
PWM_CNT4 **x=0, 1**	PWMx_BA+0xA0	R	PWM Counter Register 4	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**DIRF**	**PWM Direction Indicator Flag (Read Only)** 0 = Counter is Down count. 1 = Counter is UP count.
[15:0]	**CNT**	**PWM Data Register (Read Only)** User can monitor CNT to know the current value in 16-bit period counter.

PWM Generation Register 0 (PWM_WGCTL0)

Register	Offset	R/W	Description	Reset Value
PWM_WGCTL0 **x=0, 1**	PWMx_BA+0xB0	R/W	PWM Generation Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**PRDPCTL5**		**PRDPCTL4**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRDPCTL3		**PRDPCTL2**		**PRDPCTL1**		**PRDPCTL0**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ZPCTL5**		**ZPCTL4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ZPCTL3		**ZPCTL2**		**ZPCTL1**		**ZPCTL0**	

Bits	Description	
[31:28]	**Reserved**	Reserved.
[2n+17:2n+16] n=0,1…5	**PRDPCTLn**	**PWM Period (Center) Point Control** Each bit n controls the corresponding PWM channel n. 00 = Do nothing. 01 = PWM period (center) point output Low. 10 = PWM period (center) point output High. 11 = PWM period (center) point output Toggle. PWM can control output level when PWM counter count to (PERIODn+1). **Note:** This bit is center point control when PWM counter operating in up-down counter type.
[15:12]	**Reserved**	Reserved.
[2n+1:2n] n=0,1…5	**ZPCTLn**	**PWM Zero Point Control** Each bit n controls the corresponding PWM channel n. 00 = Do nothing. 01 = PWM zero point output Low. 10 = PWM zero point output High. 11 = PWM zero point output Toggle. PWM can control output level when PWM counter count to zero.

PWM Generation Register 1 (PWM_WGCTL1)

Register	Offset	R/W	Description	Reset Value
PWM_WGCTL1 **x=0, 1**	PWMx_BA+0xB4	R/W	PWM Generation Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**CMPDCTL5**		**CMPDCTL4**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPDCTL3		**CMPDCTL2**		**CMPDCTL1**		**CMPDCTL0**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CMPUCTL5**		**CMPUCTL4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPUCTL3		**CMPUCTL2**		**CMPUCTL1**		**CMPUCTL0**	

Bits	Description	
[31:28]	**Reserved**	Reserved.
[2n+17:2n+16] n=0,1…5	**CMPDCTLn**	**PWM Compare Down Point Control** Each bit n controls the corresponding PWM channel n. 00 = Do nothing. 01 = PWM compare down point output Low. 10 = PWM compare down point output High. 11 = PWM compare down point output Toggle. PWM can control output level when PWM counter down count to CMPDAT. **Note:** In complementary mode, CMPDCTL1, 3, 5 use as another CMPDCTL for channel 0, 2, 4.
[15:12]	**Reserved**	Reserved.
[2n+1:2n] n=0,1…5	**CMPUCTLn**	**PWM Compare Up Point Control** Each bit n controls the corresponding PWM channel n. 00 = Do nothing. 01 = PWM compare up point output Low. 10 = PWM compare up point output High. 11 = PWM compare up point output Toggle. PWM can control output level when PWM counter up count to CMPDAT. **Note:** In complementary mode, CMPUCTL1, 3, 5 use as another CMPUCTL for channel 0, 2, 4.

PWM Mask Enable Register (PWM_MSKEN)

Register	Offset	R/W	Description	Reset Value
PWM_MSKEN x=0, 1	PWMx_BA+0xB8	R/W	PWM Mask 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		**MSKEN5**	**MSKEN4**	**MSKEN3**	**MSKEN2**	**MSKEN1**	**MSKEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**MSKENn**	**PWM Mask Enable Bits** Each bit n controls the corresponding PWM channel n. The PWM output signal will be masked when this bit is enabled. The corresponding PWM channel n will output MSKDATn (PWM_MSK[5:0]) data. 0 = PWM output signal is non-masked. 1 = PWM output signal is masked and output MSKDATn data.

PWM Mask DATA Register (PWM_MSK)

Register	Offset	R/W	Description	Reset Value
PWM_MSK **x=0, 1**	PWMx_BA+0xBC	R/W	PWM Mask 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**
Reserved		**MSKDAT5**	**MSKDAT4**	**MSKDAT3**	**MSKDAT2**	**MSKDAT1**	**MSKDAT0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**MSKDATn**	**PWM Mask Data Bit** This data bit control the state of PWMn output pin, if corresponding mask function is enabled. Each bit n controls the corresponding PWM channel n. 0 = Output logic low to PWMn. 1 = Output logic high to PWMn.

PWM Brake Noise Filter Register (PWM_BNF)

Register	Offset	R/W	Description	Reset Value
PWM_BNF **x=0, 1**	PWMx_BA+0xC0	R/W	PWM Brake Noise Filter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							**BK1SRC**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**BK0SRC**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BRK1PINV	**BRK1FCNT**			**BRK1FCS**			**BRK1FEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BRK0PINV	**BRK0FCNT**			**BRK0FCS**			**BRK0FEN**

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**BK1SRC**	**Brake 1 Pin Source Select** For PWM0 setting: 0 = Brake 1 pin source come from PWM0_BRAKE1. 1 = Brake 1 pin source come from PWM1_BRAKE1. For PWM1 setting: 0 = Brake 1 pin source come from PWM1_BRAKE1. 1 = Brake 1 pin source come from PWM0_BRAKE1.
[23:17]	**Reserved**	Reserved.
[16]	**BK0SRC**	**Brake 0 Pin Source Select** For PWM0 setting: 0 = Brake 0 pin source come from PWM0_BRAKE0. 1 = Brake 0 pin source come from PWM1_BRAKE0. For PWM1 setting: 0 = Brake 0 pin source come from PWM1_BRAKE0. 1 = Brake 0 pin source come from PWM0_BRAKE0.
[15]	**BRK1PINV**	**Brake 1 Pin Inverse** 0 = The state of pin PWMx_BRAKE1 is passed to the negative edge detector. 1 = The inversed state of pin PWMx_BRAKE1 is passed to the negative edge detector.
[14:12]	**BRK1FCNT**	**Brake 1 Edge Detector Filter Count** The register bits control the Brake1 filter counter to count from 0 to BRK1FCNT.
[11:9]	**BRK1FCS**	**Brake 1 Edge Detector Filter Clock Selection** 000 = Filter clock = HCLK. 001 = Filter clock = HCLK/2.

		010 = Filter clock = HCLK/4. 011 = Filter clock = HCLK/8. 100 = Filter clock = HCLK/16. 101 = Filter clock = HCLK/32. 110 = Filter clock = HCLK/64. 111 = Filter clock = HCLK/128.
[8]	**BRK1FEN**	**PWM Brake 1 Noise Filter Enable Bit** 0 = Noise filter of PWM Brake 1 Disabled. 1 = Noise filter of PWM Brake 1 Enabled.
[7]	**BRK0PINV**	**Brake 0 Pin Inverse** 0 = The state of pin PWMx_BRAKE0 is passed to the negative edge detector. 1 = The inversed state of pin PWMx_BRAKE10 is passed to the negative edge detector.
[6:4]	**BRK0FCNT**	**Brake 0 Edge Detector Filter Count** The register bits control the Brake0 filter counter to count from 0 to BRK1FCNT.
[3:1]	**BRK0FCS**	**Brake 0 Edge Detector Filter Clock Selection** 000 = Filter clock is HCLK. 001 = Filter clock is HCLK/2. 010 = Filter clock is HCLK/4. 011 = Filter clock is HCLK/8. 100 = Filter clock is HCLK/16. 101 = Filter clock is HCLK/32. 110 = Filter clock is HCLK/64. 111 = Filter clock is HCLK/128.
[0]	**BRK0FEN**	**PWM Brake 0 Noise Filter Enable Bit** 0 = Noise filter of PWM Brake 0 Disabled. 1 = Noise filter of PWM Brake 0 Enabled.

PWM System Fail Brake Control Register (PWM_FAILBRK)

Register	Offset	R/W	Description	Reset Value
PWM_FAILBRK	PWMx_BA+0xC4	R/W	PWM System Fail Brake 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				CORBRKEN	Reserved	BODBRKEN	CSSBRKEN

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**CORBRKEN**	**Core Lockup Detection Trigger PWM Brake Function 0 Enable Bit** 0 = Brake Function triggered by Core lockup detection Disabled. 1 = Brake Function triggered by Core lockup detection Enabled.
[2]	**Reserved**	Reserved.
[1]	**BODBRKEN**	**Brown-out Detection Trigger PWM Brake Function 0 Enable Bit** 0 = Brake Function triggered by BOD Disabled. 1 = Brake Function triggered by BOD Enabled.
[0]	**CSSBRKEN**	**Clock Security System Detection Trigger PWM Brake Function 0 Enable Bit** 0 = Brake Function triggered by CSS detection Disabled. 1 = Brake Function triggered by CSS detection Enabled.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

PWM Brake Edge Detect Control Register 0_1, 2_3, 4_5(PWM_BRKCTL0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_BRKCTL0_1	PWMx_BA+0xC8	R/W	PWM Brake Edge Detect Control Register 0_1	0x0000_0000
PWM_BRKCTL2_3	PWMx_BA+0xCC	R/W	PWM Brake Edge Detect Control Register 2_3	0x0000_0000
PWM_BRKCTL4_5	PWMx_BA+0xD0	R/W	PWM Brake Edge Detect Control Register 4_5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				BRKAODD		BRKAEVEN	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SYSLEN	Reserved	BRKP1LEN	BRKP0LEN	Reserved			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SYSEEN	Reserved	BRKP1EEN	BRKP0EEN	Reserved			

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:18]	**BRKAODD**	**PWM Brake Action Select for Odd Channel (Write Protect)** 00 = PWM odd channel level-detect brake function not affect channel output. 01 = PWM odd channel output tri-state when level-detect brake happened. 10 = PWM odd channel output low level when level-detect brake happened. 11 = PWM odd channel output high level when level-detect brake happened. **Note:** These bits are write protected. Refer to SYS_REGLCTL register.
[17:16]	**BRKAEVEN**	**PWM Brake Action Select for Even Channel (Write Protect)** 00 = PWM even channel level-detect brake function not affect channel output. 01 = PWM even channel output tri-state when level-detect brake happened. 10 = PWM even channel output low level when level-detect brake happened. 11 = PWM even channel output high level when level-detect brake happened. **Note:** These bits are write protected. Refer to SYS_REGLCTL register.
[15]	**SYSLEN**	**Enable System Fail As Level-detect Brake Source (Write Protect)** 0 = System Fail condition as level-detect brake source Disabled. 1 = System Fail condition as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[14]	**Reserved**	Reserved.
[13]	**BRKP1LEN**	**Enable BKP1 Pin As Level-detect Brake Source (Write Protect)** 0 = PWMx_BRAKE1 pin as level-detect brake source Disabled. 1 = PWMx_BRAKE1 pin as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

[12]	**BRKP0LEN**	**Enable BKP0 Pin As Level-detect Brake Source (Write Protect)** 0 = PWMx_BRAKE0 pin as level-detect brake source Disabled. 1 = PWMx_BRAKE0 pin as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[11:8]	**Reserved**	Reserved.
[7]	**SYSEEN**	**Enable System Fail As Edge-detect Brake Source (Write Protect)** 0 = System Fail condition as edge-detect brake source Disabled. 1 = System Fail condition as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[6]	**Reserved**	Reserved.
[5]	**BRKP1EEN**	**Enable PWMx_BRAKE1 Pin As Edge-detect Brake Source (Write Protect)** 0 = BKP1 pin as edge-detect brake source Disabled. 1 = BKP1 pin as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[4]	**BRKP0EEN**	**Enable PWMx_BRAKE0 Pin As Edge-detect Brake Source (Write Protect)** 0 = BKP0 pin as edge-detect brake source Disabled. 1 = BKP0 pin as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[3:0]	**Reserved**	Reserved.

PWM Pin Polar Inverse Control (PWM_POLCTL)

Register	Offset	R/W	Description	Reset Value
PWM_POLCTL	PWMx_BA+0xD4	R/W	PWM Pin Polar Inverse 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		PINV5	PINV4	PINV3	PINV2	PINV1	PINV0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**PINVn**	**PWM PIN Polar Inverse Control** The register controls polarity state of PWM output. Each bit n controls the corresponding PWM channel n. 0 = PWM output polar inverse Disabled. 1 = PWM output polar inverse Enabled.

PWM Output Enable Register (PWM_POEN)

Register	Offset	R/W	Description	Reset Value
PWM_POEN	PWMx_BA+0xD8	R/W	PWM Output 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		**POEN5**	**POEN4**	**POEN3**	**POEN2**	**POEN1**	**POEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**POENn**	**PWM Pin Output Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = PWM pin at tri-state. 1 = PWM pin in output mode.

PWM Software Brake Control Register (PWM_SWBRK)

Register	Offset	R/W	Description	Reset Value
PWM_SWBRK	PWMx_BA+0xDC	W	PWM Software Brake 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					BRKLTRG4	BRKLTRG2	BRKLTRG0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					BRKETRG4	BRKETRG2	BRKETRG0

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**BRKLTRGn**	**PWM Level Brake Software Trigger (Write Only) (Write Protect)** Each bit n controls the corresponding PWM pair n. Write 1 to this bit will trigger level brake, and set BRKLIFn to 1 in PWM_INTSTS1 register. **Note:** These bits are write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[2:0]	**BRKETRGn**	**PWM Edge Brake Software Trigger (Write Only) (Write Protect)** Each bit n controls the corresponding PWM pair n. Write 1 to this bit will trigger Edge brake, and set BRKEIFn to 1 in PWM_INTSTS1 register. **Note:** These bits are write protected. Refer to SYS_REGLCTL register.

PWM Interrupt Enable Register 0 (PWM_INTEN0)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN0	PWMx_BA+0xE0	R/W	PWM Interrupt Enable Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIEN5	CMPDIEN4	CMPDIEN3	CMPDIEN2	CMPDIEN1	CMPDIEN0
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		CMPUIEN5	CMPUIEN4	CMPUIEN3	CMPUIEN2	CMPUIEN1	CMPUIEN0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			PIEN4	Reserved	PIEN2	Reserved	PIEN0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			ZIEN4	Reserved	ZIEN2	Reserved	ZIEN0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**CMPDIENn**	**PWM Compare Down Count Interrupt Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = Compare down count interrupt Disabled. 1 = Compare down count interrupt Enabled. **Note:** In complementary mode, CMPDIEN1, 3, 5 use as another CMPDIEN for channel 0, 2, 4.
[23:22]	**Reserved**	Reserved.
[21:16]	**CMPUIENn**	**PWM Compare Up Count Interrupt Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = Compare up count interrupt Disabled. 1 = Compare up count interrupt Enabled. **Note:** In complementary mode, CMPUIEN1, 3, 5 use as another CMPUIEN for channel 0, 2, 4.
[15:13]	**Reserved**	Reserved.
[12]	**PIEN4**	**PWM Period Point Interrupt Enable 4** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When counter type is up-down, period point means center point.
[11]	**Reserved**	Reserved.
[10]	**PIEN2**	**PWM Period Point Interrupt Enable 2** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When counter type is up-down, period point means center point.
[9]	**Reserved**	Reserved.

[8]	**PIEN0**	**PWM Period Point Interrupt Enable 0** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When counter type is up-down, period point means center point.
[7:5]	**Reserved**	Reserved.
[4]	**ZIEN4**	**PWM Zero Point Interrupt Enable 4** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.
[3]	**Reserved**	Reserved.
[2]	**ZIEN2**	**PWM Zero Point Interrupt Enable 2** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.
[1]	**Reserved**	Reserved.
[0]	**ZIEN0**	**PWM Zero Point Interrupt Enable 0** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.

PWM Interrupt Enable Register 1 (PWM_INTEN1)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN1	PWMx_BA+0xE4	R/W	PWM Interrupt Enable 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					**BRKLIEN4_5**	**BRKLIEN2_3**	**BRKLIEN0_1**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**BRKEIEN4_5**	**BRKEIEN2_3**	**BRKEIEN0_1**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**BRKLIEN4_5**	**PWM Level-detect Brake Interrupt Enable for Channel4/5 (Write Protect)** 0 = Level-detect Brake interrupt for channel4/5 Disabled. 1 = Level-detect Brake interrupt for channel4/5 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[9]	**BRKLIEN2_3**	**PWM Level-detect Brake Interrupt Enable for Channel2/3 (Write Protect)** 0 = Level-detect Brake interrupt for channel2/3 Disabled. 1 = Level-detect Brake interrupt for channel2/3 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[8]	**BRKLIEN0_1**	**PWM Level-detect Brake Interrupt Enable for Channel0/1 (Write Protect)** 0 = Level-detect Brake interrupt for channel0/1 Disabled. 1 = Level-detect Brake interrupt for channel0/1 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[2]	**BRKEIEN4_5**	**PWM Edge-detect Brake Interrupt Enable for Channel4/5 (Write Protect)** 0 = Edge-detect Brake interrupt for channel4/5 Disabled. 1 = Edge-detect Brake interrupt for channel4/5 Enabled. **Note:** This bitr is write protected. Refer to SYS_REGLCTL register.
[1]	**BRKEIEN2_3**	**PWM Edge-detect Brake Interrupt Enable for Channel2/3 (Write Protect)** 0 = Edge-detect Brake interrupt for channel2/3 Disabled. 1 = Edge-detect Brake interrupt for channel2/3 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

[0]	**BRKEIEN0_1**	**PWM Edge-detect Brake Interrupt Enable for Channel0/1 (Write Protect)** 0 = Edge-detect Brake interrupt for channel0/1 Disabled. 1 = Edge-detect Brake interrupt for channel0/1 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

PWM Interrupt Flag Register 0 (PWM_INTSTS0)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS0	PWMx_BA+0xE8	R/W	PWM Interrupt Flag Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	22	21	20	19	18	17	16
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
15	14	13	12	11	10	9	8
Reserved			PIF4	Reserved	PIF2	Reserved	PIF0
7	6	5	4	3	2	1	0
Reserved			ZIF4	Reserved	ZIF2	Reserved	ZIF0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**CMPDIFn**	**PWM Compare Down Count Interrupt Flag** Each bit n controls the corresponding PWM channel n. Flag is set by hardware when PWM counter down count and reaches PWM_CMPDATn, software can clear this bit by writing 1 to it. **Note1:** If CMPDAT equal to PERIOD, this flag is not working in down counter type selection. **Note2:** In complementary mode, CMPDIF1, 3, 5 use as another CMPDIF for channel 0, 2, 4.
[23:22]	**Reserved**	Reserved.
[21:16]	**CMPUIFn**	**PWM Compare Up Count Interrupt Flag** Flag is set by hardware when PWM counter up count and reaches PWM_CMPDATn, software can clear this bit by writing 1 to it. Each bit n controls the corresponding PWM channel n. **Note1:** If CMPDAT equal to PERIOD, this flag is not working in up counter type selection. **Note2:** In complementary mode, CMPUIF1, 3, 5 use as another CMPUIF for channel 0, 2, 4.
[15:13]	**Reserved**	Reserved.
[12]	**PIF4**	**PWM Period Point Interrupt Flag 4** This bit is set by hardware when PWM_CH4 counter reaches PWM_PERIOD4, software can write 1 to clear this bit to zero.
[11]	**Reserved**	Reserved.
[10]	**PIF2**	**PWM Period Point Interrupt Flag 2** This bit is set by hardware when PWM_CH2 counter reaches PWM_PERIOD2, software can write 1 to clear this bit to zero.
[9]	**Reserved**	Reserved.
[8]	**PIF0**	**PWM Period Point Interrupt Flag 0**

		This bit is set by hardware when PWM_CH0 counter reaches PWM_PERIOD0, software can write 1 to clear this bit to zero.
[7:5]	**Reserved**	Reserved.
[4]	**ZIF4**	**PWM Zero Point Interrupt Flag 4** This bit is set by hardware when PWM_CH4 counter reaches zero, software can write 1 to clear this bit to zero.
[3]	**Reserved**	Reserved.
[2]	**ZIF2**	**PWM Zero Point Interrupt Flag 2** This bit is set by hardware when PWM_CH2 counter reaches zero, software can write 1 to clear this bit to zero.
[1]	**Reserved**	Reserved.
[0]	**ZIF0**	**PWM Zero Point Interrupt Flag 0** This bit is set by hardware when PWM_CH0 counter reaches zero, software can write 1 to clear this bit to zero.

PWM Interrupt Flag Register 1 (PWM_INTSTS1)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS1	PWMx_BA+0xEC	R/W	PWM Interrupt Flag Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		BRKLSTS5	BRKLSTS4	BRKLSTS3	BRKLSTS2	BRKLSTS1	BRKLSTS0
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		BRKESTS5	BRKESTS4	BRKESTS3	BRKESTS2	BRKESTS1	BRKESTS0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		BRKLIF5	BRKLIF4	BRKLIF3	BRKLIF2	BRKLIF1	BRKLIF0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		BRKEIF5	BRKEIF4	BRKEIF3	BRKEIF2	BRKEIF1	BRKEIF0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**BRKLSTS5**	**PWM Channel5 Level-detect Brake Status (Read Only)** 0 = PWM channel5 level-detect brake state is released. 1 = When PWM channel5 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel5 at brake state. **Note:** This bit is read only and auto cleared by hardware. When enabled brake source return to high level, PWM will release brake state until current PWM period finished. The PWM waveform will start output from next full PWM period.
[28]	**BRKLSTS4**	**PWM Channel4 Level-detect Brake Status (Read Only)** 0 = PWM channel4 level-detect brake state is released. 1 = When PWM channel4 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel4 at brake state. **Note:** This bit is read only and auto cleared by hardware. When enabled brake source return to high level, PWM will release brake state until current PWM period finished. The PWM waveform will start output from next full PWM period.
[27]	**BRKLSTS3**	**PWM Channel3 Level-detect Brake Status (Read Only)** 0 = PWM channel3 level-detect brake state is released. 1 = When PWM channel3 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel3 at brake state. **Note:** This bit is read only and auto cleared by hardware. When enabled brake source return to high level, PWM will release brake state until current PWM period finished. The PWM waveform will start output from next full PWM period.
[26]	**BRKLSTS2**	**PWM Channel2 Level-detect Brake Status (Read Only)** 0 = PWM channel2 level-detect brake state is released. 1 = When PWM channel2 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel2 at brake state. **Note:** This bit is read only and auto cleared by hardware. When enabled brake source return to high level, PWM will release brake state until current PWM period finished. The PWM waveform will start output from next full PWM period.

[25]	**BRKLSTS1**	**PWM Channel1 Level-detect Brake Status (Read Only)** 0 = PWM channel1 level-detect brake state is released. 1 = When PWM channel1 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel1 at brake state. **Note:** This bit is read only and auto cleared by hardware. When enabled brake source return to high level, PWM will release brake state until current PWM period finished. The PWM waveform will start output from next full PWM period.
[24]	**BRKLSTS0**	**PWM Channel0 Level-detect Brake Status (Read Only)** 0 = PWM channel0 level-detect brake state is released. 1 = When PWM channel0 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel0 at brake state. **Note:** This bit is read only and auto cleared by hardware. When enabled brake source return to high level, PWM will release brake state until current PWM period finished. The PWM waveform will start output from next full PWM period.
[23:22]	**Reserved**	Reserved.
[21]	**BRKESTS5**	**PWM Channel5 Edge-detect Brake Status** 0 = PWM channel5 edge-detect brake state is released. 1 = When PWM channel5 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel5 at brake state, writing 1 to clear.
[20]	**BRKESTS4**	**PWM Channel4 Edge-detect Brake Status** 0 = PWM channel4 edge-detect brake state is released. 1 = When PWM channel4 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel4 at brake state, writing 1 to clear.
[19]	**BRKESTS3**	**PWM Channel3 Edge-detect Brake Status** 0 = PWM channel3 edge-detect brake state is released. 1 = When PWM channel3 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel3 at brake state, writing 1 to clear.
[18]	**BRKESTS2**	**PWM Channel2 Edge-detect Brake Status** 0 = PWM channel2 edge-detect brake state is released. 1 = When PWM channel2 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel2 at brake state, writing 1 to clear.
[17]	**BRKESTS1**	**PWM Channel1 Edge-detect Brake Status** 0 = PWM channel1 edge-detect brake state is released. 1 = When PWM channel1 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel1 at brake state, writing 1 to clear.
[16]	**BRKESTS0**	**PWM Channel0 Edge-detect Brake Status** 0 = PWM channel0 edge-detect brake state is released. 1 = When PWM channel0 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel0 at brake state, writing 1 to clear.
[15:14]	**Reserved**	Reserved.
[13]	**BRKLIF5**	**PWM Channel5 Level-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel5 level-detect brake event do not happened. 1 = When PWM channel5 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

[12]	**BRKLIF4**	**PWM Channel4 Level-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel4 level-detect brake event do not happened. 1 = When PWM channel4 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[11]	**BRKLIF3**	**PWM Channel3 Level-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel3 level-detect brake event do not happened. 1 = When PWM channel3 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[10]	**BRKLIF2**	**PWM Channel2 Level-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel2 level-detect brake event do not happened. 1 = When PWM channel2 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[9]	**BRKLIF1**	**PWM Channel1 Level-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel1 level-detect brake event do not happened. 1 = When PWM channel1 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[8]	**BRKLIF0**	**PWM Channel0 Level-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel0 level-detect brake event do not happened. 1 = When PWM channel0 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:6]	**Reserved**	Reserved.
[5]	**BRKEIF5**	**PWM Channel5 Edge-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel5 edge-detect brake event do not happened. 1 = When PWM channel5 edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[4]	**BRKEIF4**	**PWM Channel4 Edge-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel4 edge-detect brake event do not happened. 1 = When PWM channel4 edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[3]	**BRKEIF3**	**PWM Channel3 Edge-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel3 edge-detect brake event do not happened. 1 = When PWM channel3 edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[2]	**BRKEIF2**	**PWM Channel2 Edge-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel2 edge-detect brake event do not happened. 1 = When PWM channel2 edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

[1]	**BRKEIF1**	**PWM Channel1 Edge-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel1 edge-detect brake event do not happened. 1 = When PWM channel1 edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[0]	**BRKEIF0**	**PWM Channel0 Edge-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel0 edge-detect brake event do not happened. 1 = When PWM channel0 edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

PWM Trigger ADC Source Select Register 0 (PWM_ADCTS0)

Register	Offset	R/W	Description	Reset Value
PWM_ADCTS0	PWMx_BA+0xF8	R/W	PWM Trigger ADC Source Select Register 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved			TRGSEL3			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TRGEN2	Reserved			TRGSEL2			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TRGEN1	Reserved			TRGSEL1			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRGEN0	Reserved			TRGSEL0			

Bits	Description	
[31]	**TRGEN3**	**PWM_CH3 Trigger ADC Enable Bit**
[30:28]	**Reserved**	Reserved.
[27:24]	**TRGSEL3**	**PWM_CH3 Trigger ADC Source Select** 0000 = PWM_CH2 zero point. 0001 = PWM_CH2 period point. 0010 = PWM_CH2 zero or period point. 0011 = PWM_CH2 up-count CMPDAT point. 0100 = PWM_CH2 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH3 up-count CMPDAT point. 1001 = PWM_CH3 down-count CMPDAT point. Others = reserved.
[23]	**TRGEN2**	**PWM_CH2 Trigger ADC Enable Bit**
[22:20]	**Reserved**	Reserved.
[19:16]	**TRGSEL2**	**PWM_CH2 Trigger ADC Source Select** 0000 = PWM_CH2 zero point. 0001 = PWM_CH2 period point. 0010 = PWM_CH2 zero or period point. 0011 = PWM_CH2 up-count CMPDAT point. 0100 = PWM_CH2 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH3 up-count CMPDAT point.

		1001 = PWM_CH3 down-count CMPDAT point. Others = reserved.
[15]	**TRGEN1**	**PWM_CH1 Trigger ADC Enable Bit**
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL1**	**PWM_CH1 Trigger ADC Source Select** 0000 = PWM_CH0 zero point. 0001 = PWM_CH0 period point. 0010 = PWM_CH0 zero or period point. 0011 = PWM_CH0 up-count CMPDAT point. 0100 = PWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH1 up-count CMPDAT point. 1001 = PWM_CH1 down-count CMPDAT point. Others = reserved.
[7]	**TRGEN0**	**PWM_CH0 Trigger ADC Enable Bit**
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL0**	**PWM_CH0 Trigger ADC Source Select** 0000 = PWM_CH0 zero point. 0001 = PWM_CH0 period point. 0010 = PWM_CH0 zero or period point. 0011 = PWM_CH0 up-count CMPDAT point. 0100 = PWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH1 up-count CMPDAT point. 1001 = PWM_CH1 down-count CMPDAT point. Others = reserved.

PWM Trigger ADC Source Select Register 1 (PWM_ADCTS1)

Register	Offset	R/W	Description	Reset Value
PWM_ADCTS1	PWMx_BA+0xFC	R/W	PWM Trigger ADC Source Select 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**
TRGEN5	Reserved			TRGSEL5			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRGEN4	Reserved			TRGSEL4			

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**TRGEN5**	**PWM_CH5 Trigger ADC Enable Bit**
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL5**	**PWM_CH5 Trigger ADC Source Select** 0000 = PWM_CH4 zero point. 0001 = PWM_CH4 period point. 0010 = PWM_CH4 zero or period point. 0011 = PWM_CH4 up-count CMPDAT point. 0100 = PWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH5 up-count CMPDAT point. 1001 = PWM_CH5 down-count CMPDAT point. Others = reserved.
[7]	**TRGEN4**	**PWM_CH4 Trigger ADC Enable Bit**
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL4**	**PWM_CH4 Trigger ADC Source Select** 0000 = PWM_CH4 zero point. 0001 = PWM_CH4 period point. 0010 = PWM_CH4 zero or period point. 0011 = PWM_CH4 up-count CMPDAT point. 0100 = PWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved.

		0111 = Reserved. 1000 = PWM_CH5 up-count CMPDAT point. 1001 = PWM_CH5 down-count CMPDAT point. Others = reserved.

PWM Synchronous Start Control Register (PWM_SSCTL)

Register	Offset	R/W	Description	Reset Value
PWM_SSCTL	PWMx_BA+0x110	R/W	PWM Synchronous Start 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						**SSRC**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**SSEN4**	**Reserved**	**SSEN2**	**Reserved**	**SSEN0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**SSRC**	**PWM Synchronous Start Source Select** 00 = Synchronous start source come from PWM0. 01 = Synchronous start source come from PWM1. 10 = Synchronous start source come from BPWM0. 11 = Synchronous start source come from BPWM1.
[7:5]	**Reserved**	Reserved.
[4]	**SSEN4**	**PWM Synchronous Start Function Enable 4** When synchronous start function is enabled, the PWM_CH4 counter enable bit (CNTEN4) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.
[3]	**Reserved**	Reserved.
[2]	**SSEN2**	**PWM Synchronous Start Function Enable 2** When synchronous start function is enabled, the PWM_CH2 counter enable bit (CNTEN2) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.
[1]	**Reserved**	Reserved.
[0]	**SSEN0**	**PWM Synchronous Start Function Enable 0** When synchronous start function is enabled, the PWM_CH0 counter enable bit (CNTEN0) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.

PWM Synchronous Start Trigger Register (PWM_SSTRG)

Register	Offset	R/W	Description	Reset Value
PWM_SSTRG	PWMx_BA+0x114	W	PWM Synchronous Start Trigger 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							**CNTSEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTSEN**	**PWM Counter Synchronous Start Enable (Write Only)** PMW counter synchronous enable function is used to make selected PWM channels (include PWM0_CHx and PWM1_CHx) start counting at the same time. Writing this bit to 1 will also set the counter enable bit (CNTENn, n denotes channel 0 to 5) if correlated PWM channel counter synchronous start function is enabled.

PWM Status Register (PWM_STATUS)

Register	Offset	R/W	Description	Reset Value
PWM_STATUS	PWMx_BA+0x120	R/W	PWM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		ADCTRG5	ADCTRG4	ADCTRG3	ADCTRG2	ADCTRG1	ADCTRG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			CNTMAX4	Reserved	CNTMAX2	Reserved	CNTMAX0

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21:16]	**ADCTRGn**	**ADC Start of Conversion Status** Each bit n controls the corresponding PWM channel n. 0 = Indicates no ADC start of conversion trigger event has occurred. 1 = Indicates an ADC start of conversion trigger event has occurred, software can write 1 to clear this bit.
[15:5]	**Reserved**	Reserved.
[4]	**CNTMAX4**	**Time-base Counter 4 Equal to 0xFFFF Latched Status** 0 = indicates the time-base counter never reached its maximum value 0xFFFF. 1 = indicates the time-base counter reached its maximum value, software can write 1 to clear this bit.
[3]	**Reserved**	Reserved.
[2]	**CNTMAX2**	**Time-base Counter 2 Equal to 0xFFFF Latched Status** 0 = indicates the time-base counter never reached its maximum value 0xFFFF. 1 = indicates the time-base counter reached its maximum value, software can write 1 to clear this bit.
[1]	**Reserved**	Reserved.
[0]	**CNTMAX0**	**Time-base Counter 0 Equal to 0xFFFF Latched Status** 0 = indicates the time-base counter never reached its maximum value 0xFFFF. 1 = indicates the time-base counter reached its maximum value, software can write 1 to clear this bit.

PWM Capture Input Enable Register (PWM_CAPINEN)

Register	Offset	R/W	Description	Reset Value
PWM_CAPINEN	PWMx_BA+0x200	R/W	PWM Capture Input 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		CAPINEN5	CAPINEN4	CAPINEN3	CAPINEN2	CAPINEN1	CAPINEN0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**CAPINENn**	**Capture Input Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = PWM Channel capture input path Disabled. The input of PWM channel capture function is always regarded as 0. 1 = PWM Channel capture input path Enabled. The input of PWM channel capture function comes from correlative multifunction pin.

PWM Capture Control Register (PWM_CAPCTL)

Register	Offset	R/W	Description	Reset Value
PWM_CAPCTL	PWMx_BA+0x204	R/W	PWM Capture Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		FCRLDEN5	FCRLDEN4	FCRLDEN3	FCRLDEN2	FCRLDEN1	FCRLDEN0
23	22	21	20	19	18	17	16
Reserved		RCRLDEN5	RCRLDEN4	RCRLDEN3	RCRLDEN2	RCRLDEN1	RCRLDEN0
15	14	13	12	11	10	9	8
Reserved		CAPINV5	CAPINV4	CAPINV3	CAPINV2	CAPINV1	CAPINV0
7	6	5	4	3	2	1	0
Reserved		CAPEN5	CAPEN4	CAPEN3	CAPEN2	CAPEN1	CAPEN0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**FCRLDENn**	**Falling Capture Reload Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[23:22]	**Reserved**	Reserved.
[21:16]	**RCRLDENn**	**Rising Capture Reload Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[15:14]	**Reserved**	Reserved.
[13:8]	**CAPINVn**	**Capture Inverter Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[7:6]	**Reserved**	Reserved.
[5:0]	**CAPENn**	**Capture Function Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = Capture function Disabled. RCAPDAT/FCAPDAT register will not be updated. 1 = Capture function Enabled. Capture latched the PWM counter value when detected rising or falling edge of input signal and saved to RCAPDAT (Rising latch) and FCAPDAT (Falling latch).

PWM Capture Status Register (PWM_CAPSTS)

Register	Offset	R/W	Description	Reset Value
PWM_CAPSTS	PWMx_BA+0x208	R	PWM Capture 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		CFLIFOV5	CFLIFOV4	CFLIFOV3	CFLIFOV2	CFLIFOV1	CFLIFOV0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		CRLIFOV5	CRLIFOV4	CRLIFOV3	CRLIFOV2	CRLIFOV1	CRLIFOV0

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:8]	**CFLIFOVn**	**Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIF is 1. Each bit n controls the corresponding PWM channel n. **Note:** This bit will be cleared automatically when user clear corresponding CFLIF.
[7:6]	**Reserved**	Reserved.
[5:0]	**CRLIFOVn**	**Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIF is 1. Each bit n controls the corresponding PWM channel n. **Note:** This bit will be cleared automatically when user clear corresponding CRLIF.

PWM Rising Capture Data Register 0~5 (PWM_RCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
PWM_RCAPDAT0	PWMx_BA+0x20C	R	PWM Rising Capture Data Register 0	0x0000_0000
PWM_RCAPDAT1	PWMx_BA+0x214	R	PWM Rising Capture Data Register 1	0x0000_0000
PWM_RCAPDAT2	PWMx_BA+0x21C	R	PWM Rising Capture Data Register 2	0x0000_0000
PWM_RCAPDAT3	PWMx_BA+0x224	R	PWM Rising Capture Data Register 3	0x0000_0000
PWM_RCAPDAT4	PWMx_BA+0x22C	R	PWM Rising Capture Data Register 4	0x0000_0000
PWM_RCAPDAT5	PWMx_BA+0x234	R	PWM Rising Capture Data Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RCAPDAT**	**PWM Rising Capture Data Register (Read Only)** When rising capture condition happened, the PWM counter value will be saved in this register.

PWM Falling Capture Data Register 0~5 (PWM_FCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
PWM_FCAPDAT0	PWMx_BA+0x210	R	PWM Falling Capture Data Register 0	0x0000_0000
PWM_FCAPDAT1	PWMx_BA+0x218	R	PWM Falling Capture Data Register 1	0x0000_0000
PWM_FCAPDAT2	PWMx_BA+0x220	R	PWM Falling Capture Data Register 2	0x0000_0000
PWM_FCAPDAT3	PWMx_BA+0x228	R	PWM Falling Capture Data Register 3	0x0000_0000
PWM_FCAPDAT4	PWMx_BA+0x230	R	PWM Falling Capture Data Register 4	0x0000_0000
PWM_FCAPDAT5	PWMx_BA+0x238	R	PWM Falling Capture Data Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FCAPDAT**	**PWM Falling Capture Data Register (Read Only)** When falling capture condition happened, the PWM counter value will be saved in this register.

PWM PDMA Control Register (PWM_PDMACTL)

Register	Offset	R/W	Description	Reset Value
PWM_PDMACTL	PWMx_BA+0x23C	R/W	PWM PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**CHSEL4_5**	**CAPORD4_5**	**CAPMOD4_5**		**CHEN4_5**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**CHSEL2_3**	**CAPORD2_3**	**CAPMOD2_3**		**CHEN2_3**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**CHSEL0_1**	**CAPORD0_1**	**CAPMOD0_1**		**CHEN0_1**

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**CHSEL4_5**	**Select Channel 4/5 to Do PDMA Transfer** 0 = Channel4. 1 = Channel5.
[19]	**CAPORD4_5**	**Capture Channel 4/5 Rising/Falling Order** Set this bit to determine whether the PWM_RCAPDAT4/5 or PWM_FCAPDAT4/5 is the first captured data transferred to memory through PDMA when CAPMOD4_5 =11. 0 = PWM_FCAPDAT4/5 is the first captured data to memory. 1 = PWM_RCAPDAT4/5 is the first captured data to memory.
[18:17]	**CAPMOD4_5**	**Select PWM_RCAPDAT4/5 or PWM_FCAPDAT4/5 to Do PDMA Transfer** 00 = Reserved. 01 = PWM_RCAPDAT4/5. 10 = PWM_FCAPDAT4/5. 11 = Both PWM_RCAPDAT4/5 and PWM_FCAPDAT4/5.
[16]	**CHEN4_5**	**Channel 4/5 PDMA Enable Bit** 0 = Channel 4/5 PDMA function Disabled. 1 = Channel 4/5 PDMA function Enabled for the channel 4/5 captured data and transfer to memory.
[15:13]	**Reserved**	Reserved.
[12]	**CHSEL2_3**	**Select Channel 2/3 to Do PDMA Transfer** 0 = Channel2. 1 = Channel3.
[11]	**CAPORD2_3**	**Capture Channel 2/3 Rising/Falling Order**

		Set this bit to determine whether the PWM_RCAPDAT2/3 or PWM_FCAPDAT2/3 is the first captured data transferred to memory through PDMA when CAPMOD2_3 =11. 0 = PWM_FCAPDAT2/3 is the first captured data to memory. 1 = PWM_RCAPDAT2/3 is the first captured data to memory.
[10:9]	**CAPMOD2_3**	**Select PWM_RCAPDAT2/3 or PWM_FCAODAT2/3 to Do PDMA Transfer** 00 = Reserved. 01 = PWM_RCAPDAT2/3. 10 = PWM_FCAPDAT2/3. 11 = Both PWM_RCAPDAT2/3 and PWM_FCAPDAT2/3.
[8]	**CHEN2_3**	**Channel 2/3 PDMA Enable Bit** 0 = Channel 2/3 PDMA function Disabled. 1 = Channel 2/3 PDMA function Enabled for the channel 2/3 captured data and transfer to memory.
[7:5]	**Reserved**	Reserved.
[4]	**CHSEL0_1**	**Select Channel 0/1 to Do PDMA Transfer** 0 = Channel0. 1 = Channel1.
[3]	**CAPORD0_1**	**Capture Channel 0/1 Rising/Falling Order** Set this bit to determine whether the PWM_RCAPDAT0/1 or PWM_FCAPDAT0/1 is the first captured data transferred to memory through PDMA when CAPMOD0_1 =11. 0 = PWM_FCAPDAT0/1 is the first captured data to memory. 1 = PWM_RCAPDAT0/1 is the first captured data to memory.
[2:1]	**CAPMOD0_1**	**Select PWM_RCAPDAT0/1 or PWM_FCAPDAT0/1 to Do PDMA Transfer** 00 = Reserved. 01 = PWM_RCAPDAT0/1. 10 = PWM_FCAPDAT0/1. 11 = Both PWM_RCAPDAT0/1 and PWM_FCAPDAT0/1.
[0]	**CHEN0_1**	**Channel 0/1 PDMA Enable Bit** 0 = Channel 0/1 PDMA function Disabled. 1 = Channel 0/1 PDMA function Enabled for the channel 0/1 captured data and transfer to memory.

PWM Capture Channel 0_1, 2_3, 4_5 PDMA Register (PWM_PDMACAP 0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_PDMACAP0_1	PWMx_BA+0x240	R	PWM Capture Channel 01 PDMA Register	0x0000_0000
PWM_PDMACAP2_3	PWMx_BA+0x244	R	PWM Capture Channel 23 PDMA Register	0x0000_0000
PWM_PDMACAP4_5	PWMx_BA+0x248	R	PWM Capture Channel 45 PDMA 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
CAPBUF							
7	6	5	4	3	2	1	0
CAPBUF							

Bits	Description	
[31:16]	Reserved	Reserved.
[15:0]	CAPBUF	**PWM Capture PDMA Register (Read Only)** This register is use as a buffer to transfer PWM capture rising or falling data to memory by PDMA.

PWM Capture Interrupt Enable Register (PWM_CAPIEN)

Register	Offset	R/W	Description	Reset Value
PWM_CAPIEN	PWMx_BA+0x250	R/W	PWM Capture 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		CAPFIEN5	CAPFIEN4	CAPFIEN3	CAPFIEN2	CAPFIEN1	CAPFIEN0
7	6	5	4	3	2	1	0
Reserved		CAPRIEN5	CAPRIEN4	CAPRIEN3	CAPRIEN2	CAPRIEN1	CAPRIEN0

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:8]	**CAPFIENn**	**PWM Capture Falling Latch Interrupt Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled.
[7:6]	**Reserved**	Reserved.
[5:0]	**CAPRIENn**	**PWM Capture Rising Latch Interrupt Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled.

PWM Capture Interrupt Flag Register (PWM_CAPIF)

Register	Offset	R/W	Description	Reset Value
PWM_CAPIF	PWMx_BA+0x254	R/W	PWM Capture Interrupt Flag 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		CFLIF5	CFLIF4	CFLIF3	CFLIF2	CFLIF1	CFLIF0
7	6	5	4	3	2	1	0
Reserved		CRLIF5	CRLIF4	CRLIF3	CRLIF2	CRLIF1	CRLIF0

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:8]	**CFLIFn**	**PWM Capture Falling Latch Interrupt Flag** This bit is writing 1 to clear. Each bit n controls the corresponding PWM channel n. 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high.
[7:6]	**Reserved**	Reserved.
[5:0]	**CRLIFn**	**PWM Capture Rising Latch Interrupt Flag** This bit is writing 1 to clear. Each bit n controls the corresponding PWM channel n. 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high.

PWM Period Register Buffer 0, 2, 4 (PWM_PBUF0, 2, 4)

Register	Offset	R/W	Description	Reset Value
PWM_PBUF0	PWMx_BA+0x304	R	PWM PERIOD0 Buffer	0x0000_0000
PWM_PBUF2	PWMx_BA+0x30C	R	PWM PERIOD2 Buffer	0x0000_0000
PWM_PBUF4	PWMx_BA+0x314	R	PWM PERIOD4 Buffer	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PBUF**	**PWM Period Register Buffer (Read Only)** Used as PERIOD active register.

PWM Comparator Register Buffer 0~5 (PWM_CMPBUF0~5)

Register	Offset	R/W	Description	Reset Value
PWM_CMPBUF0	PWMx_BA+0x31C	R	PWM CMPDAT0 Buffer	0x0000_0000
PWM_CMPBUF1	PWMx_BA+0x320	R	PWM CMPDAT1 Buffer	0x0000_0000
PWM_CMPBUF2	PWMx_BA+0x324	R	PWM CMPDAT2 Buffer	0x0000_0000
PWM_CMPBUF3	PWMx_BA+0x328	R	PWM CMPDAT3 Buffer	0x0000_0000
PWM_CMPBUF4	PWMx_BA+0x32C	R	PWM CMPDAT4 Buffer	0x0000_0000
PWM_CMPBUF5	PWMx_BA+0x330	R	PWM CMPDAT5 Buffer	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPBUF**	**PWM Comparator Register Buffer (Read Only)** Used as CMP active register.

6.10 Watchdog Timer (WDT)

6.10.1 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.

6.10.2 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[2:0] in Config0 register.
- Supports WDT time-out wake-up function only if WDT clock source is selected as LIRC or LXT.

6.10.3 Block Diagram

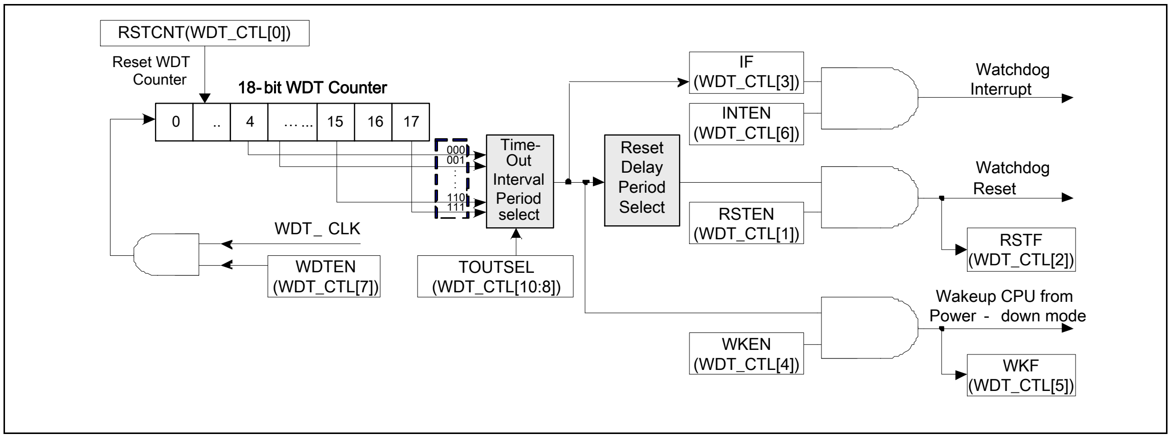


Figure 6.10-1 Watchdog Timer Block Diagram

Note1: WDT resets CPU and lasts 63 WDT_CLK.

Note2: The WDT reset delay period can be selected as 3/18/130/1026 WDT_CLK.

6.10.4 Clock Control

The WDT clock control and block diagram are shown as follows.

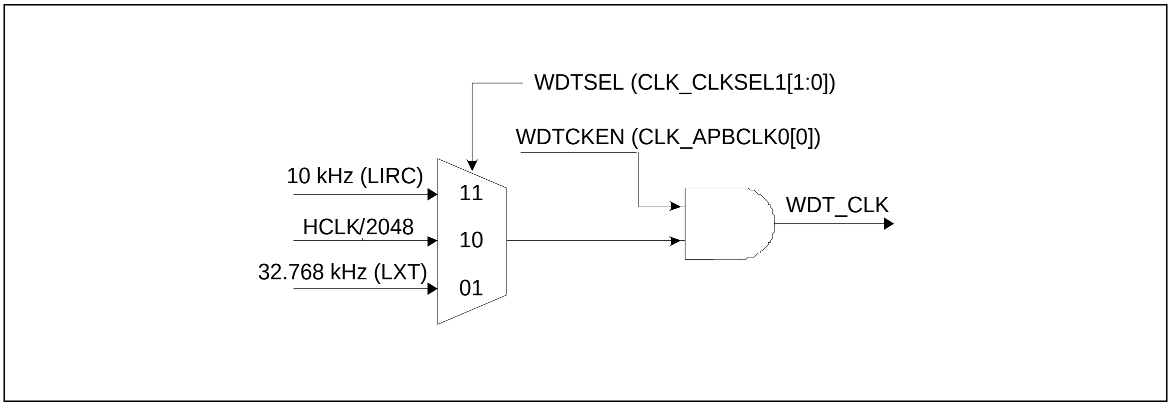


Figure 6.10-2 Watchdog Timer Clock Control

6.10.5 Basic Configuration

The WDT peripheral clock is enabled in WDTCKEN (CLK_APBCLK0[0]) and clock source can be selected in WDTSEL (CLK_CLKSEL1[1:0]).

WDT controller also can be forced enabled and active in 10 kHz after chip powered on or reset while CWDTEN[2:0] (i.e. CWDTEN[2] is Config0[31], CWDTEN[1:0] is Config0[4:3]) is not configure to b'111.

6.10.6 Functional Description

The WDT includes an 18-bit free running up counter with programmable time-out intervals. Table 6.10-1 shows the WDT time-out interval period selection and Figure 6.10-3 shows the WDT time-out interval and reset period timing.

6.10.6.1 WDT Time-out Interrupt

Setting WDTEN (WDTCR[7]) to 1 will enable the WDT function and the WDT counter to start counting up. When WDTEN is synchronizing, WSYNC (WDT_CTL[30]) will be set to 1. After synchronized, WSYNC will be cleared. There are eight time-out interval period can be selected by setting TOUTSEL (WDTCR[10:8]). When the WDT up counter reaches the TOUTSEL (WDTCR[10:8]) settings, WDT time-out interrupt will occur then WDT time-out interrupt flag IF (WDT_CTL[3]) will be set to 1 immediately. If INTEN (WDT_CTL[6]) is enabled, WDT time-out interrupt will inform to CPU.

6.10.6.2 WDT Reset Delay Period and Reset System

A specified T_{RSTD} reset delay period occurs when the IF (WDT_CTL[3]) is set to 1. User should set RSTCNT (WDT_CTL[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 RSTDSEL (WDT_ALTCTL [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 RSTF (WDT_CTL[2]) to 1 if RSTEN (WDT_CTL[1]) bit is enabled, then chip enters to reset state immediately. Refer to Figure 6.10-3, T_{RST} reset period will keep last 63 WDT clocks then chip restart executing program from reset vector (0x0000_0000). The RSTF (WDT_CTL[2]) will keep 1 after WDT time-out reset the chip, user can check RSTF (WDT_CTL[2]) by software to recognize the system has been reset by WDT time-out reset or not. Whatever which clock source be selected, LIRC is selected to WDT clock in reset period. Thus, the T_{RST} is about 6.3 ms.

TOUTSEL	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 6.10-1 Watchdog Timer Time-out Interval Period Selection

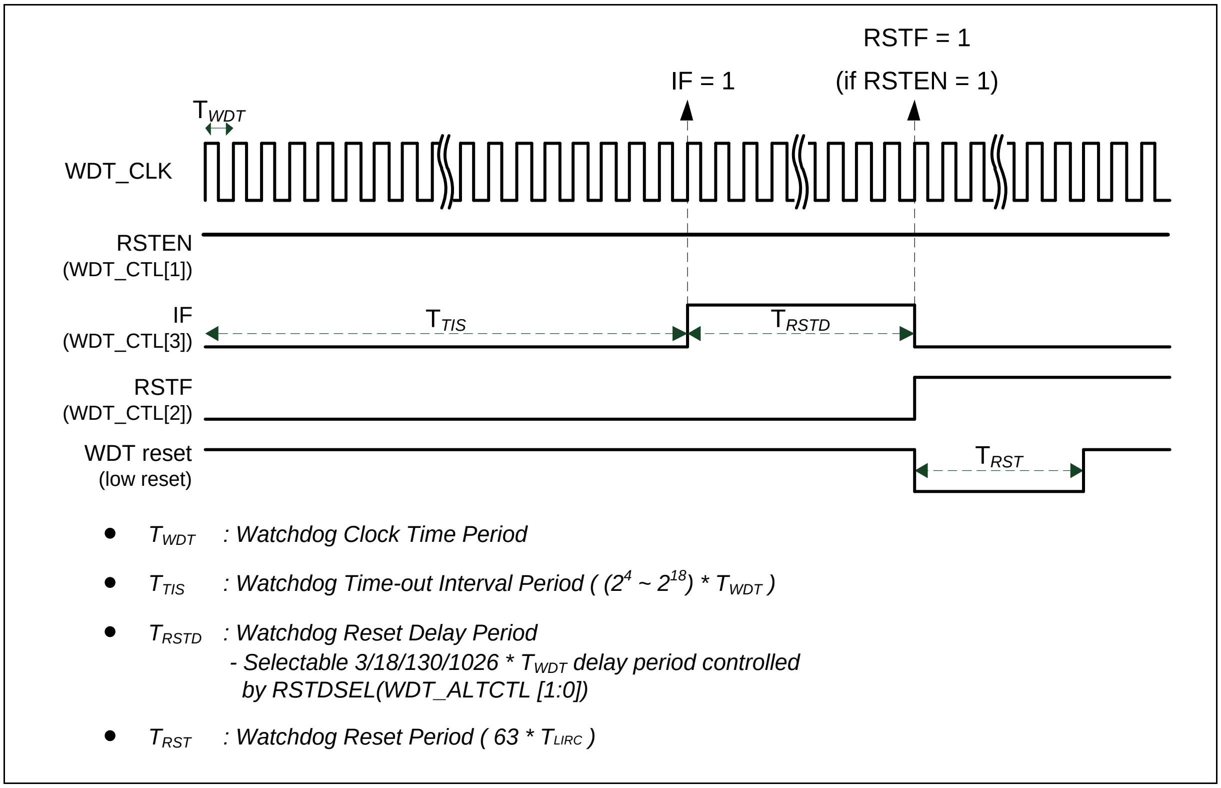

RSTF = 1
(if RSTEN = 1)
IF = 1
T_{WDT}
WDT_CLK
RSTEN
(WDT_CTL[1])
IF
(WDT_CTL[3])
T_{TIS}
T_{RSTD}
RSTF
(WDT_CTL[2])
WDT reset
(low reset)
T_{RST}
● T_{WDT} : Watchdog Clock Time Period
● T_{TIS} : Watchdog Time-out Interval Period ($(2^4$ ~ $2^{18}) * T_{WDT}$)
● T_{RSTD} : Watchdog Reset Delay Period
- Selectable 3/18/130/1026 * T_{WDT} delay period controlled by RSTDSEL(WDT_ALTCTL [1:0])
● T_{RST} : Watchdog Reset Period (63 * T_{LIRC})

Figure 6.10-3 Watchdog Timer Time-out Interval and Reset Period Timing

6.10.6.3 WDT Wake-up

If WDT clock source is selected to 10 kHz or LXT, system can be waken-up from Power-down mode while WDT time-out interrupt signal is generated and WKEN (WDT_CTL[4]) enabled. In the meanwhile, the WKF (WDT_CTL[5]) will set to 1 automatically, user can check WKF (WDT_CTL[5]) status by software to recognize the system has been waken-up by WDT time-out interrupt or not.

6.10.6.4 WDT ICE Debug

When ICE is connected to MCU, WDT counter is counting or not by ICEDEBUG (WDT_CTL[31]). The default value of ICEDEBUG is 0, WDT counter will stop counting when CPU is held by ICE. If ICEDEBUG is set to 1, WDT counter will keep counting no matter CPU is held by ICE or not.

6.10.7 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**				
WDT_CTL	WDT_BA+0x00	R/W	WDT Control Register	0x0000_0700
WDT_ALTCTL	WDT_BA+0x04	R/W	WDT Alternative Control Register	0x0000_0000
WDT_RSTCNT	WDT_BA+0x08	W	WDT Reset Counter Register	0x0000_0000
WDT_VERSION	WDT_BA+0xFFC	R	WDT RTL Design Version Register	0x0201_0000

6.10.8 Register Description

WDT Control Register (WDT_CTL)

Register	Offset	R/W	Description	Reset Value
WDT_CTL	WDT_BA+0x00	R/W	WDT Control Register	0x0000_0700

31	30	29	28	27	26	25	24
ICEDEBUG	WSYNC	Reserved					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					TOUTSEL		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WDTEN	INTEN	WKF	WKEN	IF	RSTF	RSTEN	RSTCNT

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Control (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 SYS_REGLCTL register.
[30]	**WSYNC**	**WDT Enable Control SYNC Flag Indicator (Read Only)** Due to synchronization, software can check this flag after enable WDTEN(WDT_CTL[7]), SYNC delay is 0 = WDT enable control synchronizing is completion. 1 = WDT enable control is synchronizing. **Note:** This bit enabled needs 2 * WDT_CLK
[29:11]	**Reserved**	Reserved.
[10:8]	**TOUTSEL**	**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 SYS_REGLCTL register.

[7]	**WDTEN**	**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 SYS_REGLCTL register. **Note2:** If CWDTEN[2:0] (combined by Config0[31] and Config0[4:3]) bits is not configure to 111B, this bit is forced as 1 and user cannot change this bit to 0. **Note3:** Disabled this bit needs 2 * WDT_CLK.
[6]	**INTEN**	**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 SYS_REGLCTL register.
[5]	**WKF**	**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 SYS_REGLCTL register. **Note2:** This bit is cleared by writing 1 to it.
[4]	**WKEN**	**WDT Time-out Wake-up Function Control (Write Protect)** If this bit is set to 1, while WDT time-out interrupt flag IF (WDT_CTL[3]) is generated to 1 and interrupt enable bit INTEN (WDT_CTL[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 SYS_REGLCTL register. **Note2:** Chip can be woken-up by WDT time-out interrupt signal generated only if WDT clock source is selected to 10 kHz (LIRC) or LXT.
[3]	**IF**	**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]	**RSTF**	**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]	**RSTEN**	**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 SYS_REGLCTL register.
[0]	**RSTCNT**	**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 SYS_REGLCTL register. **Note2:** This bit will be automatically cleared by hardware.

WDT Alternative Control Register (WDT_ALTCTL)

Register	Offset	R/W	Description	Reset Value
WDT_ALTCTL	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						**RSTDSEL**	

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1:0]	**RSTDSEL**	**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 RSTCNT (WDT_CTL[0]) to prevent WDT time-out reset happened. User can select a suitable setting of RSTDSEL 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 SYS_REGLCTL register. **Note2:** This register will be reset to 0 if WDT time-out reset happened.

WDT Reset Counter Register (WDT_RSTCNT)

Register	Offset	R/W	Description	Reset Value
WDT_RSTCNT	WDT_BA+0x08	W	WDT Reset Counter Register	0x0000_0000

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

Bits	Description	
[31:0]	**RSTCNT**	**WDT Reset Counter Register** Writing 0x00005AA5 to this register will reset the internal 18-bit WDT up counter value to 0. **Note:** Perform RSTCNT to reset counter needs 2 * WDT_CLK period to become active.

6.11 Window Watchdog Timer (WWDT)

6.11.1 Overview

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

6.11.2 Features

- 6-bit down counter value CNTDAT(WWDT_CNT[5:0]) and maximum 6-bit compare value CMPDAT(WWDT_CTL[21:16]) to make the WWDT time-out window period flexible
- Supports 4-bit value PSCSEL(WWDT_CTL[11:8]) to programmable maximum 11-bit prescale counter period of WWDT counter
- WWDT counter suspends in Idle/Power-down mode
- WWDT counter only can be reloaded within in valid window period to prevent system reset

6.11.3 Block Diagram

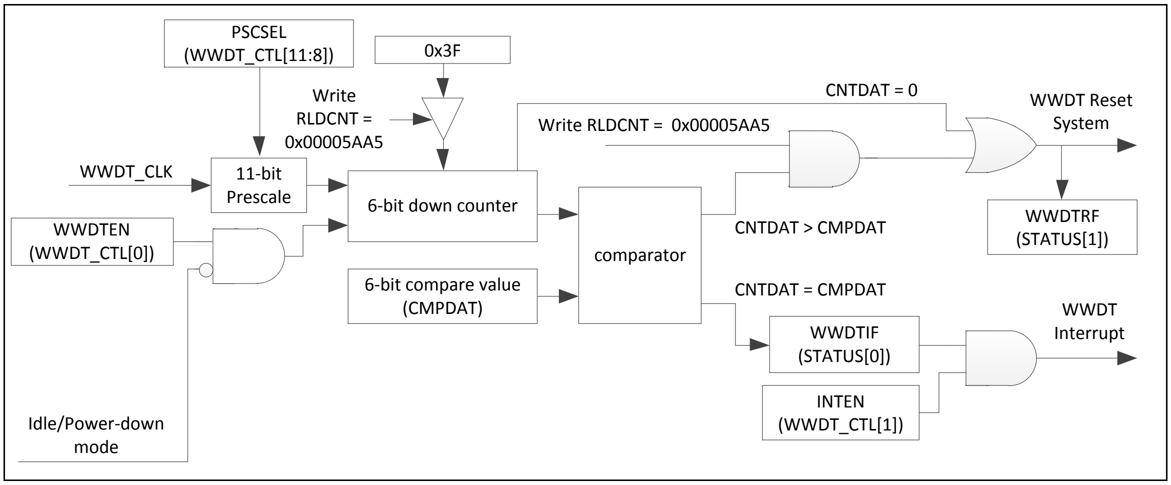


Figure 6.11-1 WWDT Block Diagram

6.11.4 Clock Control

The WWDT clock control and block diagram are shown as follows.

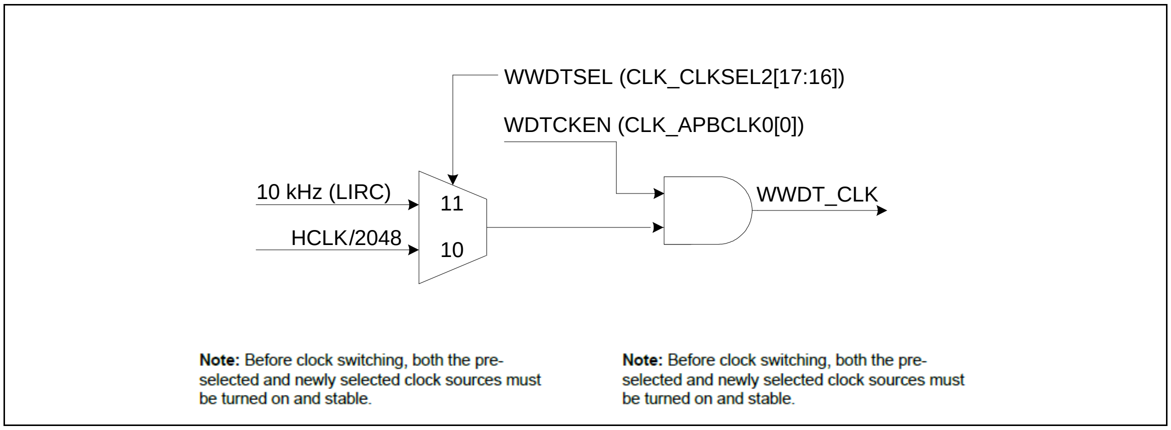

WWDTSEL (CLK_CLKSEL2[17:16])
WDTCKEN (CLK_APBCLK0[0])
10 kHz (LIRC)
HCLK/2048
11
10
WWDT_CLK
Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.
Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

Figure 6.11-2 WWDT Clock Control

6.11.5 Basic Configuration

The WWDT peripheral clock is enabled in WDTCKEN (CLK_APBCLK0[0]) and clock source can be selected in WWDTSEL (CLK_CLKSEL2[17:16]).

6.11.6 Functional Description

The WWDT includes a 6-bit down counter with programmable prescale value to define different WWDT time-out intervals. The clock source of 6-bit WWDT is based on system clock divided by 2048 (HCLK/2048) or 10 kHz internal low speed RC oscillator (LIRC) with a programmable 11-bit prescale counter value which controlled by PSCSEL (WWDT_CTL[11:8]). Also, the correlate of PSCSEL (WWDT_CTL[11:8]) and prescale value are listed in Table 6.11-1.

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

Table 6.11-1 WWDT Prescaler Value Selection

6.11.6.1 WWDT Counting

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

To avoid reseting the system while CPU clock is disabled, the WWDT counter will stop counting when CPU enters Idle/Power-down mode. After CPU enters normal mode, the WWDT counter will continue down counting.

6.11.6.2 WWDT Compare Match Interrupt

When WWDT counter value CNTDAT (WWDT_CNT[5:0]) down counts equal to window compare value CMPDAT (WWDT_CTL[21:16]) and internal prescale counter counts to 0, the WWDT counter compare match interrupt WWDTIF (WWDT_STATUS[0]) will be generated and it can be cleared by writing 1. Figure 6.11-3 shows an example of WWDT compare match interrupt when PSCSEL (WWDT_CTL[11:8]) is 0x2 and prescale value is 4.

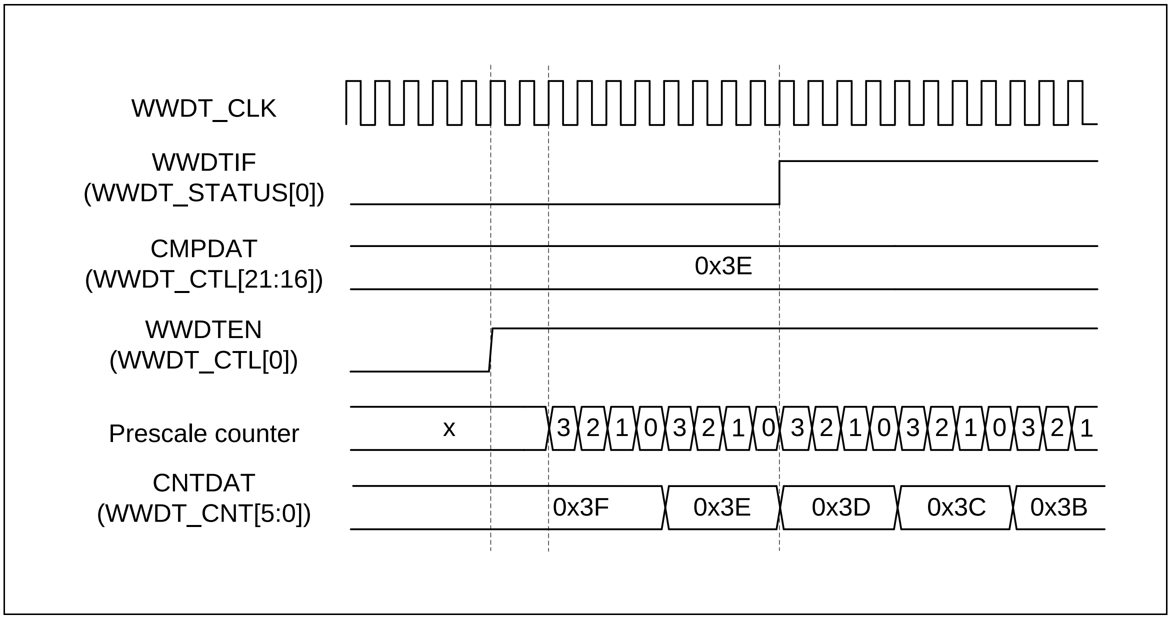


Figure 6.11-3 WWDT Compare Match Interrupt when CMPDAT is 0x3E

6.11.6.3 WWDT Reset System

When WWDTIF (WWDT_STATUS[0]) is generated, user must write 0x00005AA5 in WWDT_RLDCNT register to reload CNTDAT (WWDT_CNT[5:0]) to 0x3F, and also to prevent WWDT reset system signal occurred while CNTDAT reached to 0 and internal pre-scale counter value down count to 0.

If current CNTDAT is larger than CMPDAT (WWDT_CTL[21:16]) and user writes 0x00005AA5 to the WWDT_RLDCNT register, the WWDT reset system signal will be generated immediately to cause chip reset also. After reset system, WWDTRF (WWDT_STATUS[1]) will be set to 1.

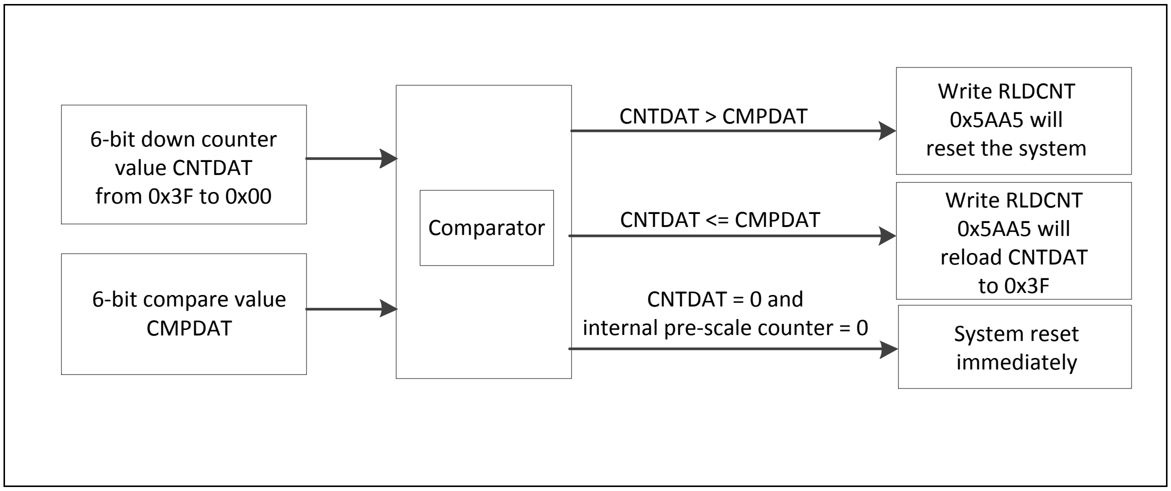


Figure 6.11-4 WWDT Reset and Reload Behavior

6.11.6.4 WWDT Window Setting Limitation

When user writes 0x00005AA5 to WWDT_RLDCNT register to reload WWDT counter value to 0x3F, it needs (WWDT_CLK * 3) period to sync the reload command to actually perform reload action. Note that if user sets PSCSEL (WWDT_CTL[11:8]) to 0000, the counter prescale value should be as 1, and the CMPDAT (WWDT_CTL[21:16]) must be larger than 2. Otherwise, writing WWDT_RLDCNT register to reload WWDT counter value to 0x3F is unavailable, WWDTIF (WWDT_STATUS[0]) is generated, and WWDT reset system event always happened after CNTDAT (WWDT_CNT[5:0]) equals to 0. Table 6.11-2 shows the detail relation between valid CMPDAT value and 4-bit PSCSEL setting.

PSCSEL	Prescale Value	Valid CMPDAT Value
0000	1	0x3 ~ 0x3E
0001	2	0x2 ~ 0x3E
Others	Others	0x1 ~ 0x3E

Table 6.11-2 CMPDAT Setting Limitation

6.11.6.5 WWDT ICE Debug

When ICE is connected to MCU, WWDT counter is counting or not by ICEDEBUG (WWDT_CTL[31]). The default value of ICEDEBUG is 0, WWDT counter will stop counting when CPU is held by ICE. If ICEDEBUG is set to 1, WWDT counter will keep counting no matter CPU is held by ICE or not.

6.11.7 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**				
WWDT_RLDCNT	WWDT_BA+0x00	W	WWDT Reload Counter Register	0x0000_0000
WWDT_CTL	WWDT_BA+0x04	R/W	WWDT Control Register	0x003F_0800
WWDT_STATUS	WWDT_BA+0x08	R/W	WWDT Status Register	0x0000_0000
WWDT_CNT	WWDT_BA+0x0C	R	WWDT Counter Value Register	0x0000_003F

6.11.8 Register Description

WWDT Reload Counter Register (WWDT_RLDCNT)

Register	Offset	R/W	Description	Reset Value
WWDT_RLDCNT	WWDT_BA+0x00	W	WWDT Reload Counter Register	0x0000_0000

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

Bits	Description	
[31:0]	**RLDCNT**	**WWDT Reload Counter Register** Writing 0x00005AA5 to this register will reload the WWDT counter value to 0x3F. **Note1:** User can only execute the reload WWDT counter value command when current CNTDAT (WWDT_CNT[5:0]) is between 1 and CMPDAT (WWDT_CTL[21:16]). If user writes 0x00005AA5 in WWDT_RLDCNT register when current CNTDAT is larger than CMPDAT, WWDT reset system event will be generated immediately. **Note2:** Execute WWDT counter reload always needs (WWDT_CLK *3) period to reload CNTDAT to 0x3F and internal prescale counter will be reset also.

WWDT Control Register (WWDT_CTL)

Register	Offset	R/W	Description	Reset Value
WWDT_CTL	WWDT_BA+0x04	R/W	WWDT Control Register	0x003F_0800

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

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

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Control** 0 = ICE debug mode acknowledgement effects WWDT counting. WWDT down counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. WWDT down counter will keep going no matter CPU is held by ICE or not.
[30:22]	**Reserved**	Reserved.
[21:16]	**CMPDAT**	**WWDT Window Compare Value** Set this field to adjust the valid reload window interval when WWDTIF (WWDT_STATUS[0]) is generated. **Note:** User can only write WWDT_RLDCNT register to reload WWDT counter value when current CNTDAT (WWDT_CNT[5:0]) is between 1 and CMPDAT. If user writes 0x00005AA5 in WWDT_RLDCNT register when current CNTDAT is larger than CMPDAT, WWDT reset system event will be generated immediately.
[15:12]	**Reserved**	Reserved.
[11:8]	**PSCSEL**	**WWDT Counter Prescale Period Selection** 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.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

		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]	**INTEN**	**WWDT Interrupt Enable Control Bit** If this bit is enabled, when WWDTIF (WWDT_STATUS[0]) is set to 1, the WWDT counter compare match interrupt signal is generated and inform to CPU. 0 = WWDT counter compare match interrupt Disabled. 1 = WWDT counter compare match interrupt Enabled.
[0]	**WWDTEN**	**WWDT Enable Control Bit** Set this bit to enable WWDT counter counting. 0 = WWDT counter is stopped. 1 = WWDT counter is starting counting.

WWDT Status Register (WWDT_STATUS)

Register	Offset	R/W	Description	Reset Value
WWDT_STATUS	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 Timer-out Reset Flag** If this bit is set to 1, it indicates that system has been reset by WWDT counter time-out reset system event. 0 = WWDT time-out reset system event did not occur. 1 = WWDT time-out reset system event occurred. **Note:** This bit is cleared by writing 1 to it.
[0]	**WWDTIF**	**WWDT Compare Match Interrupt Flag** This bit indicates that current CNTDAT (WWDT_CNT[5:0]) matches the CMPDAT (WWDT_CTL[21:16]). 0 = No effect. 1 = WWDT counter value matches CMPDAT. **Note:** This bit is cleared by writing 1 to it.

WWDT Counter Value Register (WWDT_CNT)

Register	Offset	R/W	Description	Reset Value
WWDT_CNT	WWDT_BA+0x0C	R	WWDT Counter Value Register	0x0000_003F

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**CNTDAT**	**WWDT Counter Value** CNTDAT will be updated continuously

6.12 USCI - Universal Serial Control Interface Controller

6.12.1 Overview

The Universal Serial Control Interface (USCI) is a flexible interface module covering several serial communication protocols. The user can configure this controller as UART, SPI, or I^{2}C functional protocol.

6.12.2 Features

The controller can be individually configured to match the application needs. The following protocols are supported:

- UART
- SPI
- I^{2}C

To increase readability, the registers of USCI have different alias names that depending on the selected protocol. For example, register USCI_CTL has alias name *UUART*_CTL for protocol UART, has alias name *USPI*_CTL for protocol SPI, and has alias name *UI2C*_CTL for protocol I^{2}C.

6.12.3 Block Diagram

The basic configurations of USCI are as follows:

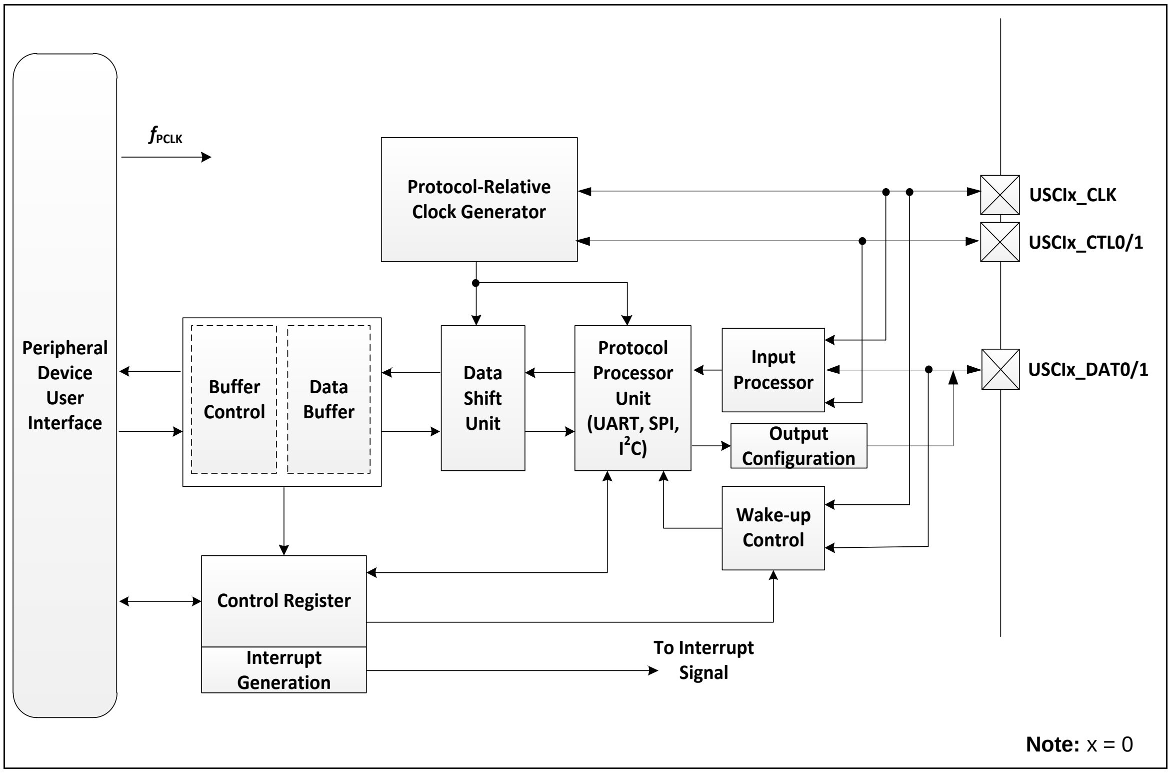


Figure 6.12-1 USCI Block Diagram

6.12.4 Functional Description

The structure of the Universal Serial Control Interface (USCI) controller is shown in Figure 6.12-1. The input signal is implemented in input processor. The data buffers and the data shift unit support the data transfers. Each protocol-specific function is handled by the protocol processor unit. The timing and time event control signals of the specific protocol are handled by the protocol-relative clock generator. All the protocol-specific events are processed in the interrupt generation unit. The wake-up function of the specific protocol is implemented in the wake-up control unit.

The USCI is equipped with three protocols including UART, SPI, and I^{2}C. They can be selected by FUNMODE (USCI_CTL [2:0]). Note that the FUNMODE must be set to 0 before changing protocol.

6.12.4.1 I/O Processer

Input Signal

All input stages offer the similar feature set. They are used for all protocols.

Table 6.12-1 lists the relative input signals for each selected protocol. Each input signal is handled by an input processor for signal conditioning, such as signal inverse selection control, or a digital input filter.

Selected Protocol		**UART**	**SPI**	**I^{2}C**
Serial Bus Clock Input	USCIx_CLK	-	SPI_CLK	SCL
Control Input	USCIx_CTL0	nCTS	SPI_SS	-
	USCIx_CTL1	-	-	-
Data Input	USCIx_DAT0	RX	SPI_MOSI_0	SDA
	USCIx_DAT1	-	SPI_MISO_0	-

Table 6.12-1 Input Signals for Different Protocols

Note: The descriptions of protocol-specific items are given in the related protocol chapters.

General Input Structure

The input structures of data and control signals include inverter, digital filter and edge detection (data signal only).

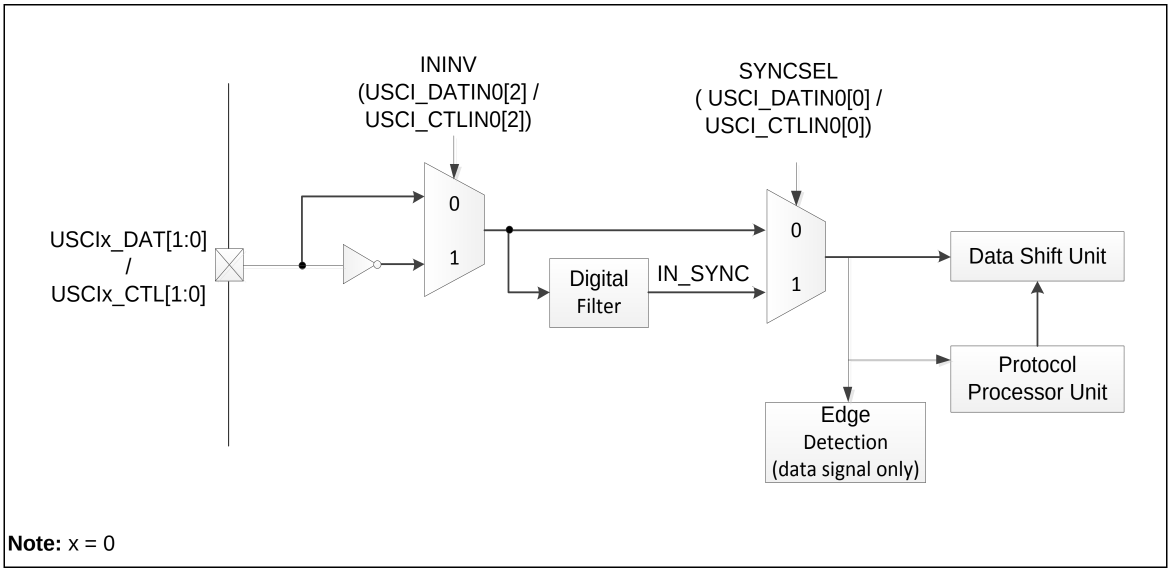


Figure 6.12-2 Input Conditioning for USCIx_DAT[1:0] and USCIx_CTL[1:0]

The input structure of USCIx_CLK is similar to USCIx_CTL[1:0] input structure, except it does not support inverse function.

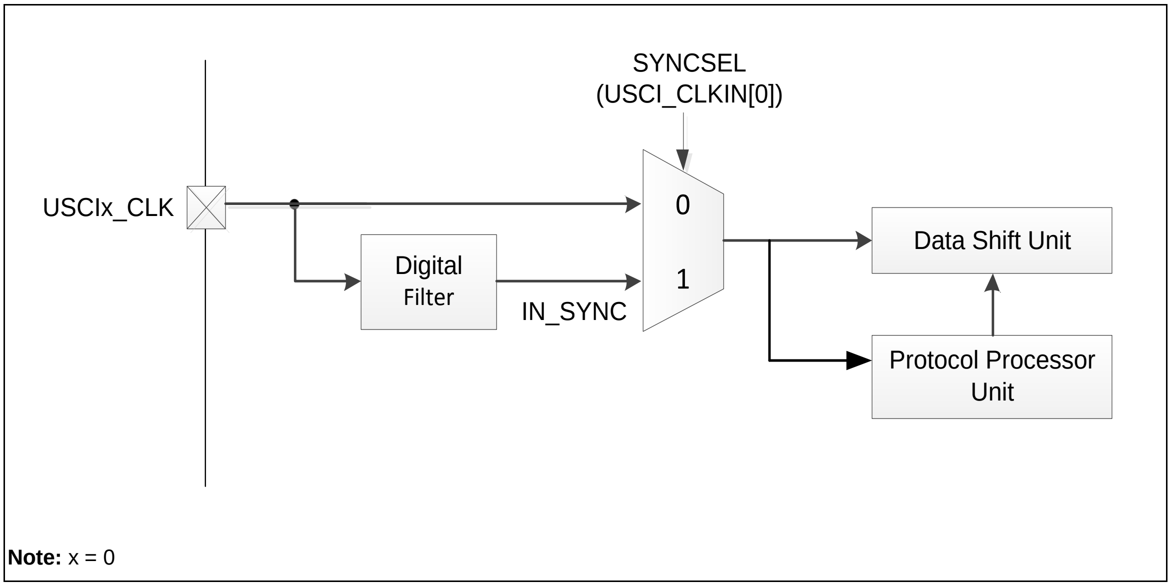


Figure 6.12-3 Input Conditioning for USCIx_CLK

All configurations of control, clock and data input structures are in USCI_CTLIN0, USCI_CLKIN and USCI_DATIN0 registers respectively. EDGEDET (USCI_DATIN0[4:3]) is used to select the edge detection condition. Note that the EDGEDET for USCI_DATIN0 must be set to 2'b10 in UART mode. The programmable edge detection indicates that the desired event has occurred by activating the trigger signal.

ININV (USCI_DATIN0[2] / USCI_CTLIN0[2]) allows a polarity inversion of the selected input signal to adapt the input signal polarity to the internal polarity of the data shift unit and the protocol state machine.

If the SYNCSEL (USCI_DATIN0[0] / USCI_CTLIN0[0] / USCI_CLKIN[0]) is set to 0, the paths of input signals do not contain any delay due to synchronization or filtering. If there is noise on the input signals, there is the possibility to synchronize the input signal (signal IN_SYNC is synchronized to f_{PCLK}). The synchronized input signal is taken into account by SYNCSEL = 1. The synchronization leads to a delay in the signal path of 2-3 times the period of f_{PCLK}.

Output Signals

Table 6.12-2 shows the relative output signals for each protocol. The number of actually used outputs depends on the selected protocol and they can be classified according to their meaning for the protocols.

Selected Protocol		UART	SPI	I^2C
Serial Bus Clock Output	USCIx_CLK	-	SPI_CLK	SCL
Control Output	USCIx_CTL0	-	SPI_SS	-
	USCIx_CTL1	nRTS	-	-
Data Output	USCIx_DAT0	-	SPI_MOSI_0	SDA
	USCIx_DAT1	TX	SPI_MISO_0	-

Table 6.12-2 Output Signals for Different Protocols

Note: The descriptions of protocol-specific items are given in the related protocol chapters.

6.12.4.2 Data Buffering

The data handling of the USCI controller is based on a Data Shift Unit (DSU) and a buffer structure. Both of the data shift and buffer registers are 16-bit wide. The inputs of Data Shift Unit include the shift data, the serial bus clock, and the shift control. The output pin of transmission can be USCIx_DAT0 pin or USCIx_DAT1 pin depends on what protocol is selected.

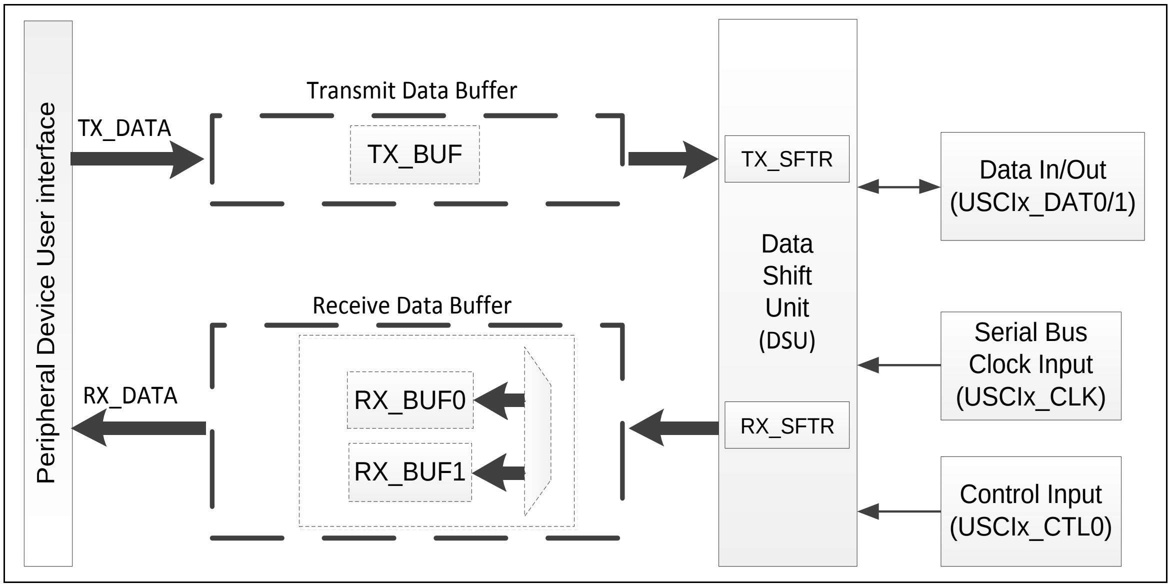


Figure 6.12-4 Block Diagram of Data Buffering

The operation of data handling includes:

- The peripheral device user interface (APB) is used to handle data, interrupts, status and control information.
- A transmitter includes transmit shift register (TX_SFTR) and a transmit data buffer (TX_BUF). The TXFULL (USCI_BUFSTS[9], TXEMPTY (USCI_BUFSTS[8]) and TXENDIF (USCI_PROTSTS[2]) can indicate the status of transmitter.
- A receiver includes receive shift register (RX_SFTR) and a double receive buffer structure (RX_BUF0, RX_BUF1). In double buffer structure, user need no care about the reception sequence and two received data can be hold if user does not read the data of USCI_RXDAT register in time.

Data Access Structure

The Data Access Structure includes read access to received data and write access of data to be transmitted. The received data is stored in the receiver buffers including RX_BUF0 and RX_BUF1. User need no care about the reception sequence. The receive buffer can be accessed by reading USCI_RXDAT register. The first received data is read out first and the next received data becomes visible in USCI_RXDAT and can be read out next.

Transmitted data can be loaded to TX_BUF by writing to the transmit register USCI_TXDAT.

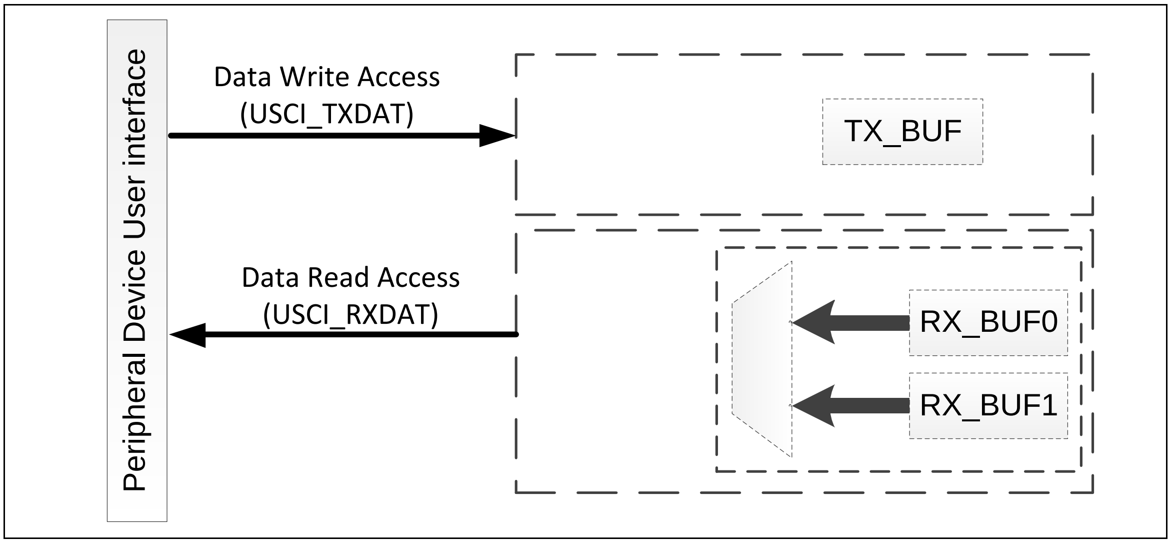


Figure 6.12-5 Data Access Structure

Transmit Data Path

The transmit data path is based on 16-bit wide transmit shift register (TX_SFTR) and transmit buffer TX_BUF. The data transfer parameters like data word length is controlled commonly for transmission and reception by the line control register USCI_LINECTL.

Transmit Buffering

The transmit shift register cannot be directly accessed by user. It is updated automatically with the value stored in the transmit buffer (TX_BUF) if a currently transmitted data is finished and new data is valid for transmission.

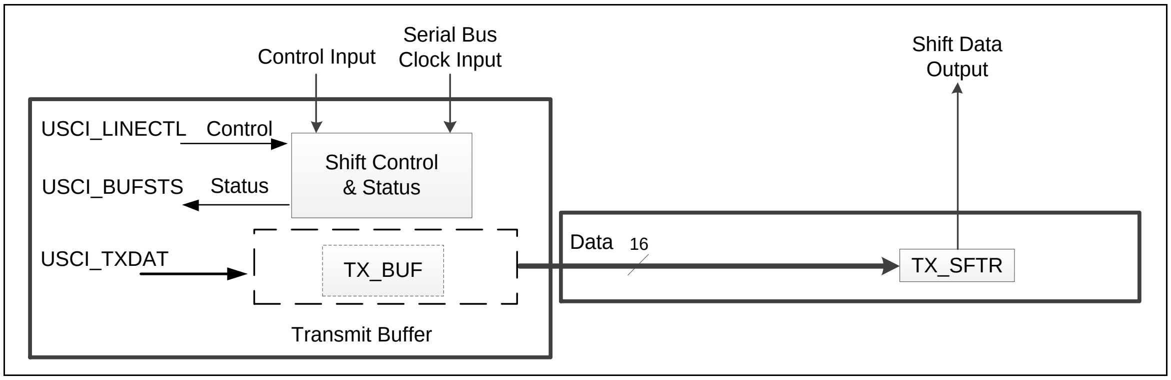


Figure 6.12-6 Transmit Data Path

Transmit Data Validation

The status of TXEMPTY (USCI_BUFSTS[8]) indicates the transmission data is valid or not in the transmit buffer (TX_BUF) and the TXSTIF (USCI_PROTSTS[1]) labels the start conditions for each data.

- If the USCI controller is a Master, the data transfer can only be started with valid data in the transmit buffer (TX_BUF). In this case, the transmit shift register is loaded with the content of transmit buffer.

  Note: Master defines the start of data transfer.

- If the USCI controller is a Slave, a data transfer requested by Master and it has to be started independently of the status in transmit buffer (TX_BUF). If a data transfer is requested and started by the Master, the transmit shift register is loaded from specific protocol control signal if it is valid for transmission.

  Note: Slave can not define the start itself, but has to react.

- The timing of loading data from transmit buffer to data shift unit depends on protocol configurations.

  UART: A transmission of the data word in transmit buffer can be started if TXEMPTY = 0 in normal operation. In auto flow control, A transmission of the data word in transmit buffer can be started while TXEMPTY = 0 and USCIx_CTL0 in active stage.

  SPI: In Master mode, data transmission will be started when TXEMPTY (USCI_BUFSTS[8]) is 0. In Slave mode, the data transmission can be started only when slave selection signal is at active state and clock is presented on USCIx_CLK pin.

  I^{2}C: A transmission of the data byte in transmit buffer can be started if TXEMPTY = 0.

- A transmission data which is located in transmit buffer can be started if the TXEMPTY (USCI_BUFSTS [8]) = 0. The content of the transmit buffer (in TX_BUF condition) should not be overwritten with new data while it is valid for transmission and a new transmission can start. If the content of TX_BUF has to be changed, user can set TXRST (USCI_BUFCTL [16]) to 1 to clear the content of TX_BUF before updating the data. Moreover, TXEMPTY (USCI_BUFSTS [8]) will be cleared automatically when transmit buffer (TX_BUF) is updated with new data. While a transmission is in progress, TX_BUF can be loaded with new data. User has to update the TX_BUF before a new transmission.

Receive Data Path

The receive data path is based on 16-bit wide receive shift register RX_SFTR and receive buffers RX_BUF0 and RX_BUF1. The data transfer parameters like data word length, or the shift direction are controlled commonly for transmission and reception by the line control register USCI_LINECTL. Register USCI_BUFSTS monitors the data validation of USCI_RXDAT.

Receive Buffering

The receive shift register cannot be directly accessed by user, but its content is automatically loaded into the receive buffer if a complete data word has been received or the frame is finished. The received data words in Receive Buffer can be read out automatically from register USCI_RXDAT.

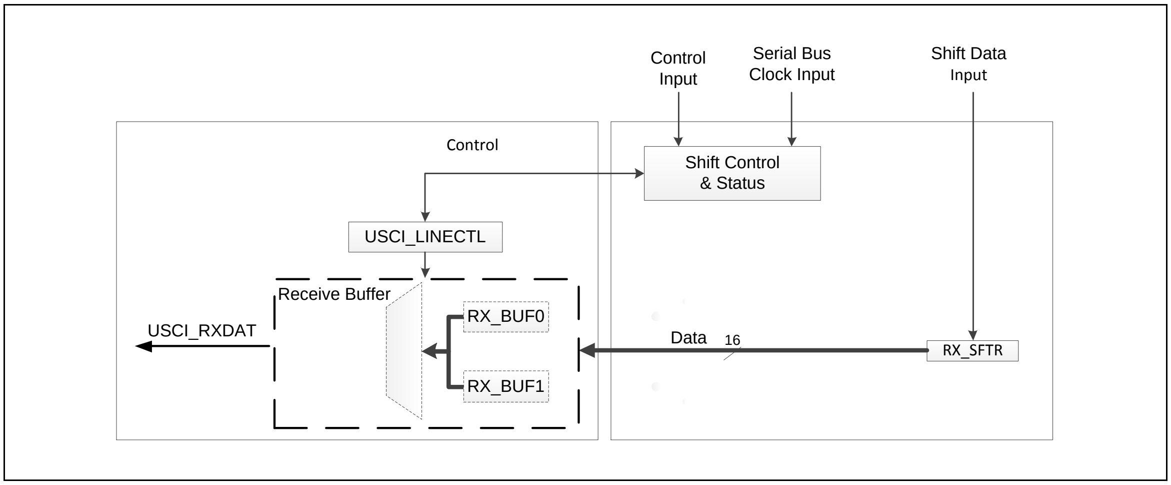


Figure 6.12-7 Receive Data Path

6.12.4.3 Port Direction Control

In SPI protocol with half-duplex configurations, the data port is bidirectional. Port direction control is intended to control the pin direction through a dedicated hardware interface.

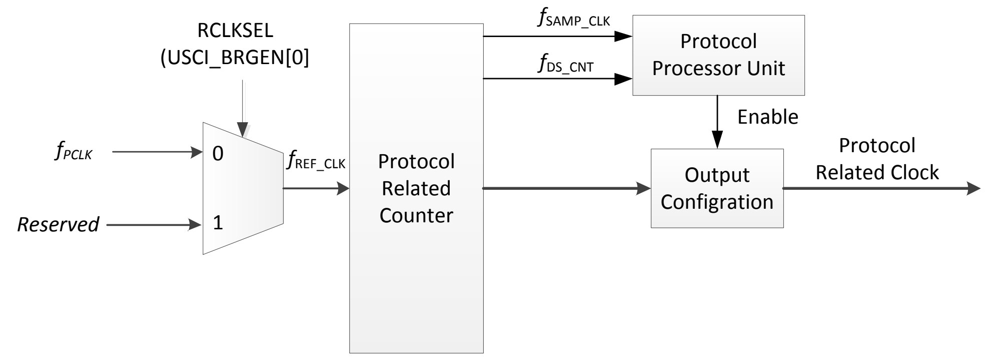
The direction of selected pin is controlled by PORTDIR (USCI_TXDAT[16]). When user writes USCI_TXDAT register, the transmit data and its port direction are settled simultaneously.

6.12.4.4 Protocol Control and Status

The protocol-related control and status information are located in the protocol control register USCI_PROTCTL and in the protocol status register USCI_PROTSTS. These registers are shared between the available protocols. As a consequence, the meaning of the bit positions in these registers is different within the protocols. Refer to each protocol’s relative register for detail information.

6.12.4.5 Protocol-Relative Clock Generator

USCI controller contains a protocol-relative clock generator and it is controlled by register USCI_BRGEN. It is reset when the USCI_BRGEN register is written. The structured of protocol-relative clock generator is shown in Figure 6.12-8.



Note: Refer the **Basic Clock Divider Counter** section to get the f_{SAMP_CLK}

Figure 6.12-8 Protocol-Relative Clock Generator

The protocol related counter contains basic clock divider counter and timing measurement counter. It is based on a divider stages, providing the frequencies needed for the different protocols. It contains:

- The basic clock divider counter provides the protocol relative clock signal and other protocol-related signals (f_{SAMP_CLK} and f_{DS_CNT}).
- The timing measurement counter for time interval measurement, e.g. baud rate detection on UART protocol.
- The output signals of protocol relative clock generator can be made available on pins (e.g USCIx_CLK for SPI).

Basic Clock Divider Counter

The basic clock divider counter is used for an integer division delivering f_{REF_CLK2}, f_{REF_CLK}, f_{DIV_CLK}, f_{SCLK}, and f_{SAMP_CLK} . The frequencies of this divider are controlled by PTCLKSEL (USCI_BRGEN [1]), CLKDIV (USCI_BRGEN [25:16]), SPCLKSEL (USCI_BRGEN [3:2]).

The basic clock divider counter is used to generate the relative protocol timing signals.

$$f_{\mathrm{DIV\_CLK}} = f_{REF\_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \text{ if PTCLKSEL } = 0$$

$$f_{\mathrm{DIV\_CLK}} = f_{REF\_CLK} \times \frac{1}{(\mathrm{CLKDIV}+1)\times 2} \text{ if PTCLKSEL } = 1$$

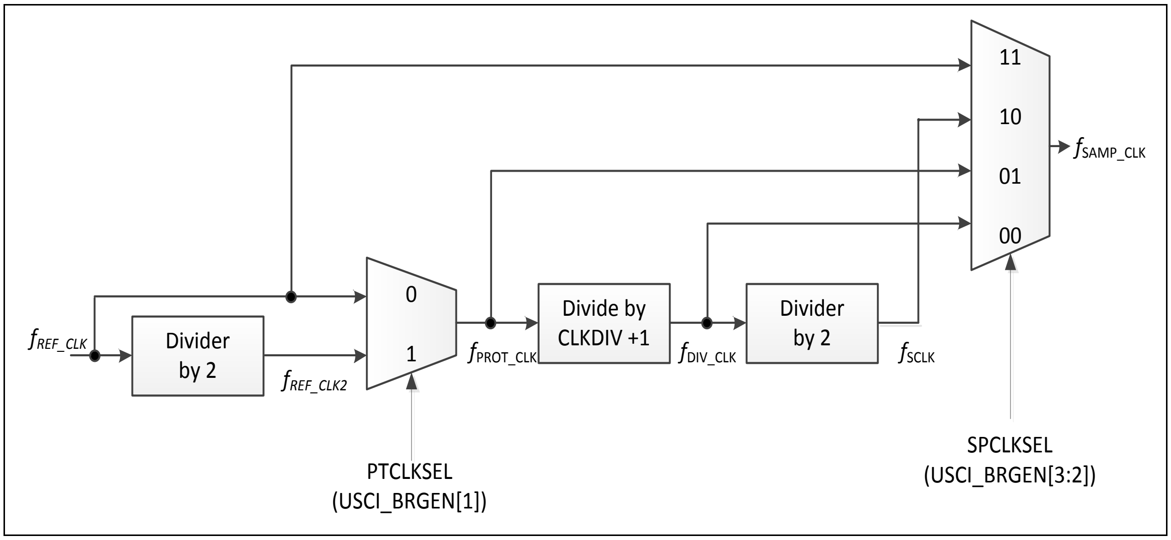


Figure 6.12-9 Basic Clock Divider Counter

Timing Measurement Counter

The timing measurement counter is used for time interval measurement and is enabled by TMCNTEN (USCI_BRGEN [4]) = 1. When TMCNTSRC (USCI_BRGEN [5]) is set to 1, the timer works on f_{DIV_CLK}, otherwise, the timer works independently from f_{PROT_CLK}. Therefore, any serial data reception or transmission can continue while the timer is performing timing measurements. The timer counts the length of protocol-related signals with f_{PROT_CLK} or f_{DIV_CLK}. It stops counting when it reaches the user-specified value.

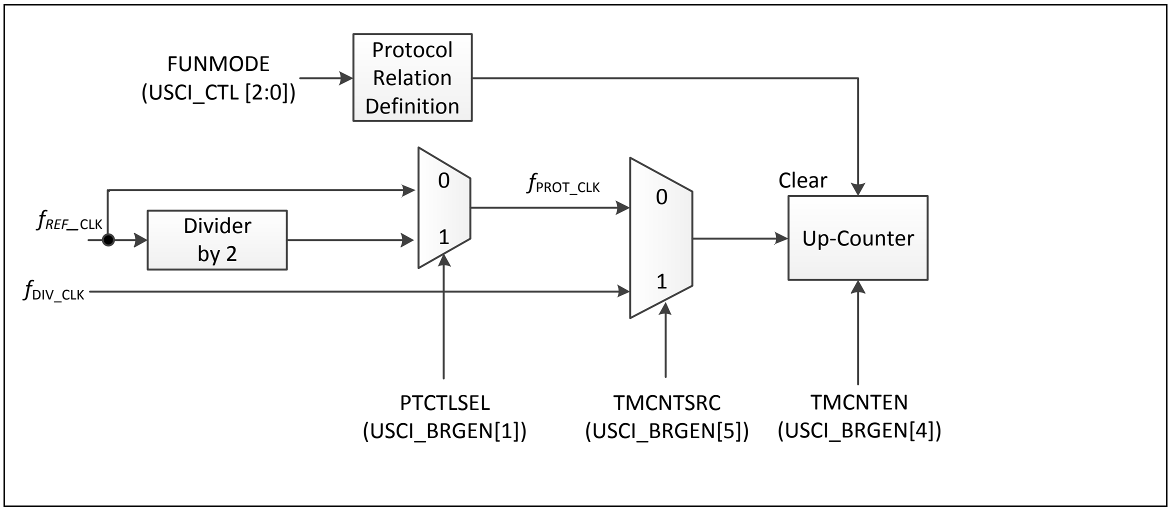


Figure 6.12-10 Block of Timing Measurement Counter

The timing measurement counter is used to perform time-out function or auto-baud rate mechanism. Its functionality depends on the selected protocol as shown below.

- UART: The timing measurement counter is used in auto baud rate detection.
- SPI: The timing measurement counter is used for counting the slave time-out period.
- I^{2}C: The timing measurement counter indicates time-out clock cycle.

Sample Time Counter

A sample time counter associated to the protocol related counter defining protocol specific timings, such shift control signals or bit timings, based on the input frequency f_{SAMP_CLK}. The sample time counter allows generating time intervals for protocol-specific purposes. The period of a sample frequency f_{PDS_CNT} is given by the selected input frequency f_{SAMP_CLK} and the programmed pre-divider value (PDSCNT (USCI_BRGEN [9:8])). The meaning of the sample time depends on the selected protocol. Please refer to the corresponding chapters for more protocol-specific information.

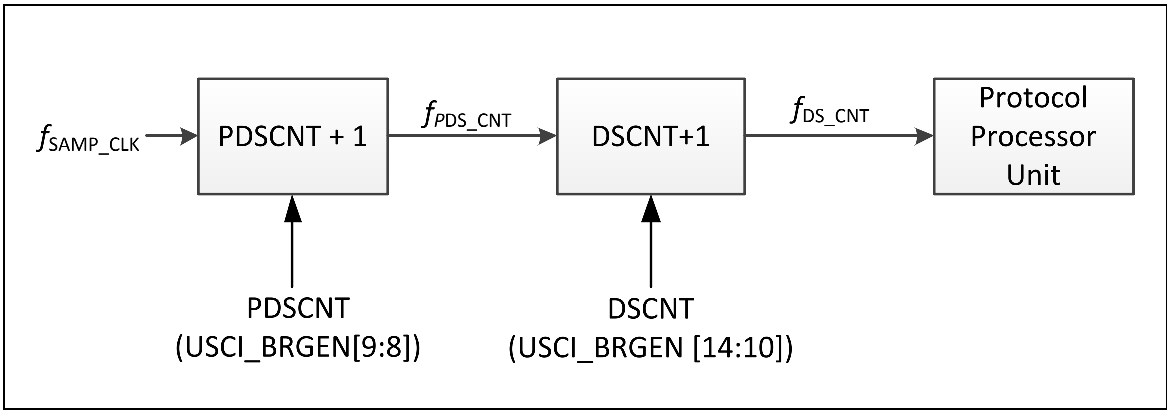


Figure 6.12-11 Sample Time Counter

6.12.4.6 *Data Transfer Events and Interrupts*

The data transfer events are based on the transmission or reception of a data word. The related indication flags are located in register USCI_PROTSTS. All events can be individually enabled for interrupt generation. If the FUNMODE (USCI_CTL [2:0]) is set to 0, the USCI is disabled. When FUNMODE (USCI_CTL [2:0]) is setting for a protocol port, the internal states will be controlled by logic hardware of the selected protocol.

- Transmit start interrupt event to indicate that a data word has been started:

  A transmit start interrupt event occurs when the data is loaded into transmitted shift register. It is indicated by flag TXSTIF (USCI_PROTSTS [1]) and, if enabled, leads to transmit start interrupt.

- Transmit end interrupt event to indicate that a data word transmission has been done:

  A transmit end interrupt event occurs when the current transmit data in shift register had finished. It is indicated by flag TXENDIF (USCI_PROTSTS [2]) and, if enabled, leads to transmit end interrupt. This event also indicates when the shift control settings (word length, shift direction, etc.) are internally “frozen” for the current data word transmission. In UART and I^2C mode, the transmit data valid is according to TXEMPTY (USCI_BUFSTS [8]) and protocol relative internal signal with the transmit end interrupt event.

- Receiver start event to indicate that a data word reception has started:

  When the receive clock edge that shifts in the first bit of a new data word is detected and reception is enabled, a receiver start event occurs. It is indicated by flag RXSTIF

(USCI_PROTSTS [3]) and, if enabled, leads to receiver start interrupt.

- Receive event to indicate that a data word has been received:

  If a new received word becomes available in the receive buffer, a receive event occurs. It is indicated by flag RXENDIF (USCI_PROTSTS [4]) and, if enabled, leads to receive interrupt.

- Data lost event to indicate a loss of the newest received data word:

  If the data word available in register USCI_RXDAT (oldest data word from RX_BUF0 or RX_BUF1) has not been read out and the receive buffer is FULL, the new incoming data will lose and this event occurs. It is indicated by flag RXOVIF (USCI_BUFSTS[3]) and, if enabled, leads to a protocol interrupt.

The general event and interrupt structure is shown in Figure 6.12-12.

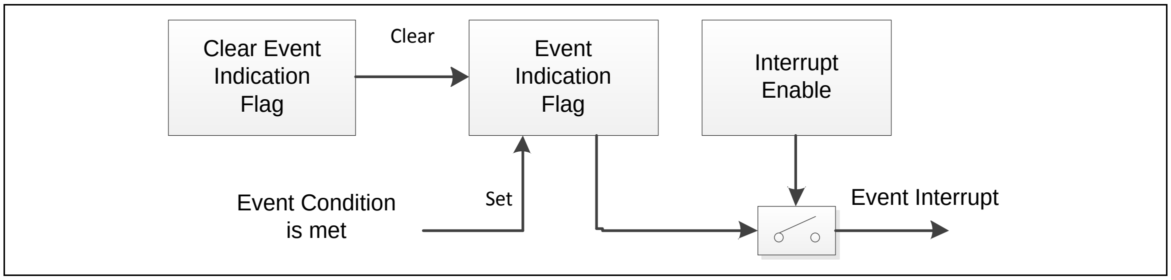


Figure 6.12-12 Event and Interrupt Structure

Each general interrupt enable can set by RXENDIEN, RXSTIEN, TXENDIEN, and TXSTIEN of USCI_INTEN [4:1]. The events are including receive end interrupt event, receive start interrupt event, transmit end interrupt event, and transmit start interrupt event. For protocol-specific interrupt, it is specified in each protocol interrupt enable register.

If a defined condition is met, an event is detected and an event indication flag becomes automatically set. The flag stays set until it is cleared by software. If interrupt is enabled, an interrupt can be generated if an event is detected.

The registers, bits and bit fields indicate the data transfer events and control the general interrupts of a USCI are shown in Table 6.12-3.

Event	Indication Flag	Indication Cleared By	Interrupt Enabled By
Transmit start interrupt event	TXSTIF (USCI_PROTSTS [1])	It is cleared by software writes 1 to corresponding interrupt bit of USCI_PROTSTS.	TXSTIEN (USCI_INTEN [1])
Transmit end interrupt event	TXENDIF (USCI_PROTSTS [2])		TXENDIEN (USCI_INTEN [2])
Receive start interrupt event	RXSTIF (USCI_PROTSTS [3])		RXSTIEN (USCI_INTEN [3])
Receive end interrupt event	RXENDIF (USCI_PROTSTS [4])		RXENDIEN (USCI_INTEN [4])

Table 6.12-3 Data Transfer Events and Interrupt Handling

6.12.4.7 Protocol-specific Events and Interrupts

These events are related to protocol-specific actions that are described in the corresponding protocol chapters. The related indication flags are located in register USCI_PROTSTS. All events can be individually enabled for the generation of the common protocol interrupt.

Event	Indication Flag	Indication Cleared By	Interrupt Enabled By
Protocol-specific events in UART mode	USCI_PROTSTS [17:16] and USCI_PROTSTS [11:5]	It is cleared by software writes 1 to corresponding interrupt bit of USCI_PROTSTS.	USCI_PROTIEN[2:1]
Protocol-specific events in SPI mode	USCI_PROTSTS [9:8], USCI_PROTSTS [6:5]		USCI_PROTIEN [3:0]
Protocol-specific events in I^2C mode	USCI_PROTSTS [13:8], USCI_PROTSTS [5]		USCI_PROTIEN [6:0]

Table 6.12-4 Protocol-specific Events and Interrupt Handling

6.12.4.8 Wake-up

The protocol-related wake-up functional information is located in the Wake-up Control Register (USCI_WKCTL) and in the Wake-up Status Register (USCI_WKSTS). These registers are shared between the available protocols. As a consequence, the meaning of the bit positions in these registers is different within the protocols.

6.12.4.9 PDMA

USCI supports PDMA transfer function. When PDMAEN (USCI_PDMACTL [3]) is set to 1, the PDMA function is enabled.

When TXPDMAEN (USCI_PDMACTL [1]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (USCI_PDMACTL [2]) is set to 1, the controller will start the PDMA reception process. USCI will issue request to PDMA controller automatically when there is data in the receive FIFO buffer.

In UART function, the requirement of RXPDMAEN will be cleared and hold if there is any error condition events including frame error, parity error or break detection. The user shall read out the current data and then the requirement of RXPDMAEN will send to the PDMA module in the next data.

6.13 USCI - UART Mode

6.13.1 Overview

The asynchronous serial channel UART covers the reception and the transmission of asynchronous data frames. It performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the controller. The receiver and transmitter are independent, frames can start at different points in time for transmission and reception.

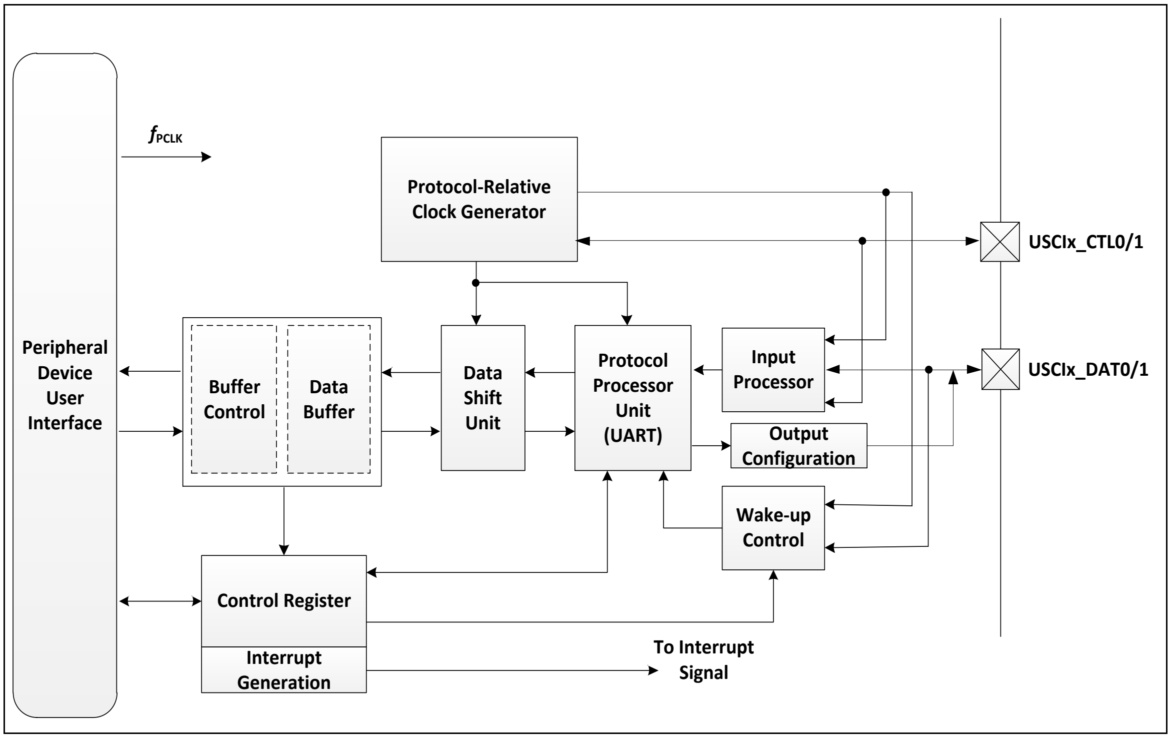
The UART controller also provides auto flow control. There are two conditions to wake-up the system by incoming data or nCTS.

6.13.2 Features

- Supports one transmit buffer and two receive buffer for data payload
- Supports hardware auto flow control function
- Supports programmable baud-rate generator
- Supports 9-Bit Data Transfer (9-Bit RS-485)
- Supports baud rate detection by built-in capture event of baud rate generator
- Supports PDMA capability
- Supports Wake-up function (Data and nCTS Wakeup Only)

6.13.3 Block Diagram

The basic configurations of USCI are as follows:



Note: x = 0

Figure 6.13-1 Block Diagram

6.13.4 Basic Configuration

The basic configurations of USCI0_UART are as follows:

- USCI0 pins are configured in SYS_GPA_MFPH, SYS_GPB_MFPL, SYS_GPB_MFPH, SYS_GPC_MFPL, SYS_GPC_MFPH, SYS_GPD_MFPL, SYS_GPD_MFPH, and SYS_GPE_MFPL registers.
- Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK1[8]).
- Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).

6.13.5 Functional Description

6.13.5.1 USCI Common Function Description

Please refer to section 6.12.4 for detailed information.

6.13.5.2 Signal Description

An UART connection is characterized by the use of a single connection line between a transmitter and a receiver. The receiver input signal (RXD) is handled by the input stage USCIx_DAT0 and the transmit output (TXD) signal is handled by the output stage of USCIx_DAT1.

For full-duplex communication, an independent communication line is needed for each transfer direction. Figure 6.13-2 shows an example with a point-to-point full-duplex connection between two communication partners UART module A and UART module B.

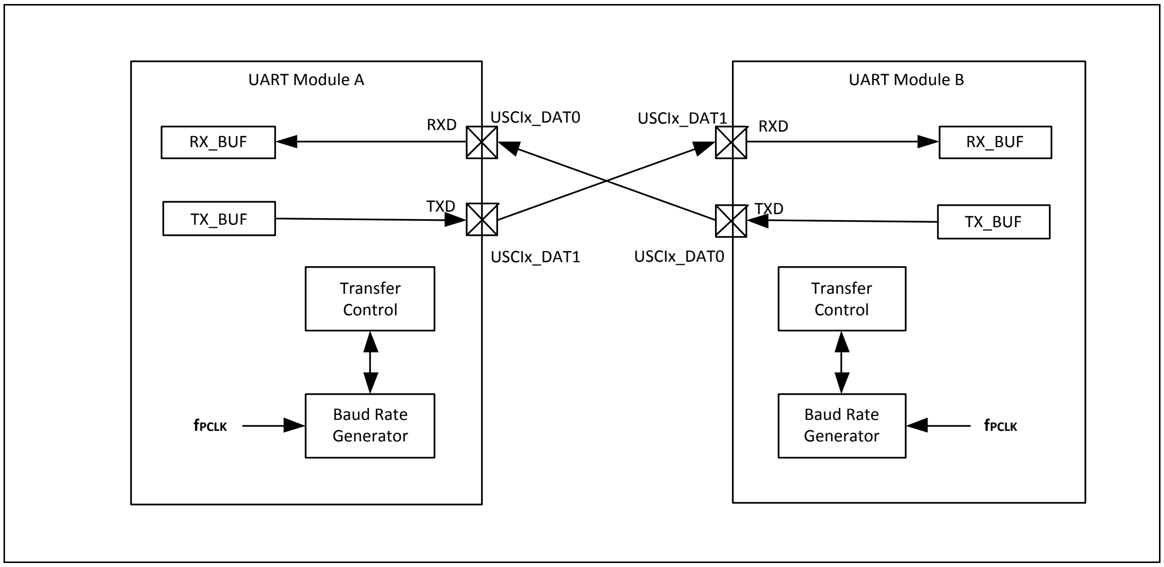


Figure 6.13-2 UART Signal Connection for Full-Duplex Communication

Input Signal

For UART protocol, the number of input signals is figured in Table 6.13-1. Each input signal is handled by an input processor for signal conditioning, such as signal inverse selection control, or a digital input filter. They can be classified according to their meaning for the protocols, as shown in Table 6.13-1.

Selected Protocol		UART
Control Input	USCIx_CTL0	nCTS
	USCIx_CTL1	X
Data Input(s)	USCIx_DAT0	RX
	USCIx_DAT1	X

Table 6.13-1 Input Signals for UART Protocols

Output Signals

For UART protocol, up to each protocol-related output signals are available. The number of actually used outputs depends on the selected protocol. They can be classified according to their meaning for the protocols as shown in Table 6.13-2.

Selected Protocol		UART
Control Output	USCI_CTL0	X
	USCI_CTL1	nRTS
Data Output (s)	USCI_DAT0	X
	USCI_DAT1	TX

Table 6.13-2 Output Signals for Different Protocols

6.13.5.3 Frame Format

- A standard UART frame is shown in Figure 6.13-3. It consists of:
- An idle time with the signal level 1.
- One start of frame bit (SOF) with the signal level 0.
- 6~13 bit data
- A parity bit (P), programmable for either even or odd parity. It is optionally possible to handle frames without parity bit.
- One or two stop bits with the signal level 1.

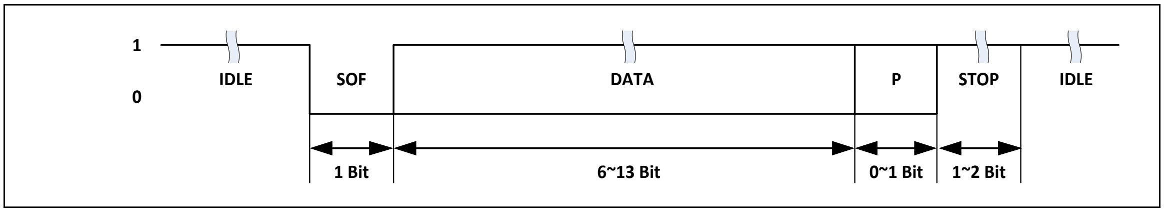


Figure 6.13-3 UART Standard Frame Format

The protocol specific bits (SOF, P, STOP) are automatically handled by the UART protocol state machine and do not appear in the data flow via the receive and transmit buffers.

Start Bit

The receiver input signal USCIx_DAT0 is checked for a falling edge. An SOF bit is detected when a falling edge occurs while the receiver is idle or after the sampling point of the last stop bit. To increase noise immunity, the SOF bit timing starts with the first falling edge that is detected. If the sampled bit value of the SOF is 1, the previous falling edge is considered to be due to noise and the receiver is considered to be idle again.

Data Field

The length of the data field (number of data bits) can be programmed by the bit field of DWIDTH (UUART_LINECTL[11:8]). It can vary between 6 to 13 data bits.

Note: In UART protocol, the data transmission order is LSB first by setting LSB (UUART_LINECTL[0]) to 1.

Parity Bit

The UART allows parity generation for transmission and parity check for reception on frame base. The type of parity can be selected by bit field PARITYEN (UUART_PROTCTL[1]) and EVENPARITY (UUART_PROTCTL[2]), common for transmission and reception (no parity, even or odd parity). If the parity handling is disabled, the UART frame does not contain any parity bit. For consistency reasons, all communication partners have to be programmed to the same parity mode.

After the last data bit of the data field, the transmitter automatically sends out its calculated parity bit if parity generation has been enabled. The receiver interprets this bit as received parity and compares it to its internally calculated one. The result of the parity check and frame check (STOP bit) are monitored in the protocol status registers (UUART_PROTSTS). The register contains bits to monitor a protocol-related status and protocol-related error indication (FRMERR, PARITYERR).

Stop Bit

Each UART frame is completed by 1 or 2 of stop bits with the signal level 1 (same level as the idle level). The number of stop bits is programmable by bit STOPB (UUART_PROTCTL[0]). A new start bit can be transferred directly after the last stop bit.

Transfer Status Indication

RXBUSY (UUART_PROTSTS[10]) indicates the receiver status.

The receiver status can be monitored by RXBUSY bit. In this case, bit RXBUSY is set during a complete frame reception from the beginning of the start of frame bit to the end of the last stop bit.

6.13.5.4 Operating Mode

In order to operate the UART protocol, the following issues have to be considered:

Select UART Mode:

The UART protocol can be selected by setting FUNMODOE (UUART_CTL[2:0]) to 0x2 and the UART protocol can be enabled by setting PROTEN (UUART_PROTCTL [31]) to 1. Note that the FUNMODE must be set to 0 before protocol changing and it is recommended to configure all parameters of the UART before UART protocol is enabled.

Pin Connections:

The USCI_DAT0 pin is used for UART receive data input signal (RX) in UART protocol. The property of input data signal can be configured in UUART_DATIN0. It is suggested to set EDGEDET (UUART_DATIN0[4:3]) as 0x2 for start bit detection.

The USCI_DAT1 pin is used for UART transmit data output signal (TX) in UART protocol. The property of output data signal can be configured in UUART_LINECTL.

The USCI_CTL0 pin is used for UART clear to send signal (nCTS) in UART protocol. The property of input control signal can be configured in UUART_CTLIN0.

The USCI_CTL1 pin is used for UART request to send signal (nRTS) in UART protocol. The property of output control signal can be configured in UUART_LINECTL.

Bit Timing Configuration:

The desired baud rate setting has to be selected, comprising the baud rate generator and the bit timing. Please refer to section 6.13.5 for detail descriptions.

Frame Format Configuration:

The word length, the stop bit number, and the parity mode has to be set up according to the application requirements by programming UUART_LINECTL and the UUART_PROTCTL register. It is required by the application, the data input and output signals can be inverted. The data transmission order is LSB first by setting LSB (UUART_LINECTL[0]) to 1.

6.13.5.5 Bit Timing

In UART mode, each frame bit is divided into data sample time in order to provide granularity in the sub-bit range to adjust the sample point to the application requirements. The number of data sample time per bit is defined by bit fields DSCNT (UUART_BRGEN[14:10]) and the length of a data sample time is given by PDSCNT (UUART_BRGEN[9:8]).

In the example given in Figure 6.13-4, one bit time is composed of 16 data sample time DSCNT(UUART_BRGEN[14:10]) = 15. It is not recommended to program less and equal than 4 data sample time per bit time.

The position of the sampling point for the bit value is fixed in 1/2 samples time. It is possible to sample the bit value to take the average of samples.

The bit timing setup (number of data sample time) is common for the transmitter and the receiver because they use the same hardware circuit.

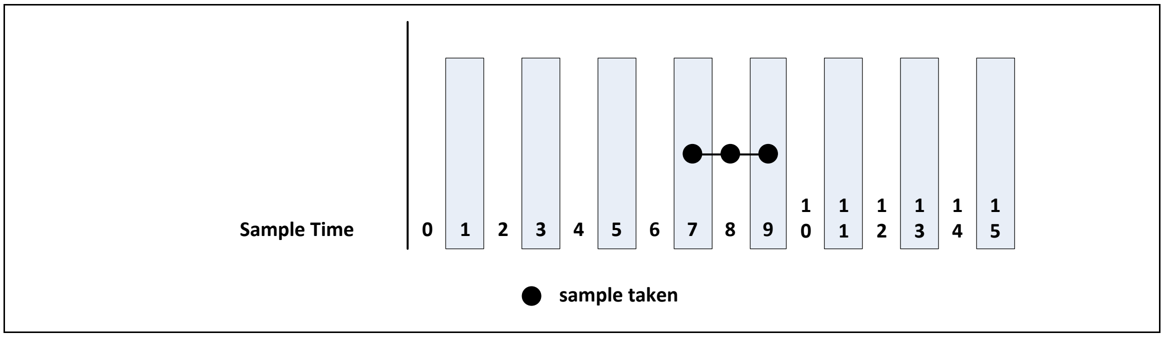


Figure 6.13-4 UART Bit Timing (data sample time)

6.13.5.6 Baud Rate Generation

The baud rate f_{UART} in UART mode depends on the number of data sample time per bit time and their timing. The baud rate setting should only be changed while the transmitter and the receiver are idle. The bits RCLKSEL, SPCLKSEL, PDSCNT, and DSCNT define the baud rate setting:

RCLKSEL (UUART_BRGEN [0])

to define the input frequency f_{REF_CLK}

SPCLKSEL (UUART_BRGEN[3:2])

to define the multiple source of the sample clock f_{SAM_CLK}

PDSCNT (UUART_BRGEN [9:8])

to define the length of a data sample time (division of f_{REF_CLK} by 1, 2, 3, or 4)

DSCNT (UUART_BRGEN [14:10])

to define the number of data sample time per bit time

The standard setting is given by RCLKSEL = 0 (f_{REF_CLK} = f_{PCLK}), PTCLKSEL = 0 (f_{PROT_CLK} = f_{REF_CLK}) and SPCLKSEL = 0x0 (f_{SAMP_CLK} = f_{DIV_CLK}). Under these conditions, the baud rate is given by:

$$f_{\mathrm{UART}} = f_{REF\_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

To generate slower frequencies, additional divide-by-2 stages can be selected by PTCLKSEL = 1 (f_{PROT_CLK} = f_{REF_CLK2}), leading to:

$$f_{\mathrm{UART}} = \frac{f_{REF\_CLK}}{2} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

If SPCLKSEL = 0x2 (f_{SAMP_CLK} = f_{SCLK}), and RCLKSEL = 0 (f_{REF_CLK} = f_{PCLK}), PTCLKSEL = 0 (f_{PROT_CLK} = f_{REF_CLK}). The baud rate is given by:

$$f_{\mathrm{UART}} = f_{REF\_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{2} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

There is error tolerance for the UART baud rate after setting the baud rate parameter. Table 6.13-3 lists the relative error percentage examples for user to calculate his relative baud rate setting.

HCLK Source	PCLK Source	Expect Baud Rate	CLKDIV (UUART_BRGEN[25:16])	DSCNT (UUART_BRGEN[14:10])	PDSCNT	Active Baud Rate	Error Percentage
HXT12M	12M (HCLK)	115200	0xC	0x7	0x0	115384	1.6%
HIRC/2, 24M	24M (HCLK)	9600	0XF9	0x9	0x0	9600	0%
HIRC48M	24M (HCLK/2)	115200	0xC	0xf	0x0	115384	1.6%

Table 6.13-3 Baud rate Relationship

Note: SPCLKSEL = 0x0, PTCLKSEL = 0, RCLKSEL = 0

6.13.5.7 Auto Baud Rate Detection

The UART controller supports auto baud rate detection function. It is used to identify the input baud rate from the receiver signal (USCIx_DAT0) and then revised the baud rate clock divider CLKDIV (UUART_BRGEN[25:16]) after the baud rate function done to meet the detected baud rate information. According the Timing Measurement Counter section, the timing measurement counter is used for time interval measurement of the input signal (USCIx_DAT0) and the actual timer value is captured into bit field BRDETITV (UUART_PROTCTL [24:16]) in each falling edge of the detected signal.

When the ABREN (PROTOCOL[6]) bit is enabled, the 0x55 data patterns is necessary for auto baud rate detection. The falling edge of input signal starts the baud rate counter and it loads the timing measurement counter value into the BRDETITV (UUART_PROTCTL [24:16]) in the next falling edge. It is suggested to use the f_{DIV_CLK} (TMCNTSRC (UUART_BRGENC[5]) =1) as the counter source.

The CLKDIV (UUART_BRGEN[25:16]) will be revised by BRDETITV (UUART_PROTCTL [25:16]) after the auto baud rate function done (the time of 4th falling edge of input signal). If the user want to receive the next successive frame correctly, it is better to set the value of CLKDIV (UUART_BRGEN[25:16]) and DSCNT (UUART_BRGEN[14:10]) as the same value (the value shall be among the range of 0xF and 0x5 because the DSCNT is used to define the sample counter of each bit and the PDSCNT (UUART_BRGEN[9:8]) is 0x0.

During the auto baud rate detection, the ABRDETIF (UUART_PROTSTS[9]) and the BRDETITV (UUART_PROTCTL [24:16]) will be updated after each falling edge of input signal and the auto baud rate pattern, 0x55, won't be received into the receiver buffer after the frame done. The bit of ABREN will be cleared by hardware after the 4th falling edge of input signal is detected thus the user can read the status of ABREN to know the auto baud rate function is done or not.

If the CLKDIV and DSCNT are not set as the same value in calculation the auto baud rate function, the user shall calculate the proper average baud rate by the value of BRDETITV and CLKDIV after the auto baud rate function done.

If the baud rate of input signal is very slow and the bit time of timing measurement counter can't calculate the correct period of the input bit time, there is a ABERRSTS bit (UUART_PROTSTS[11]) to indicate the error information of the auto baud rate detection. At this time, the user shall revise the value of CLKDIV and require the Host device to send the 0x55 pattern again.

According the limitation of timing measurement counter, the maximum auto baud rate detection is 0x1FE for BRDETITV.

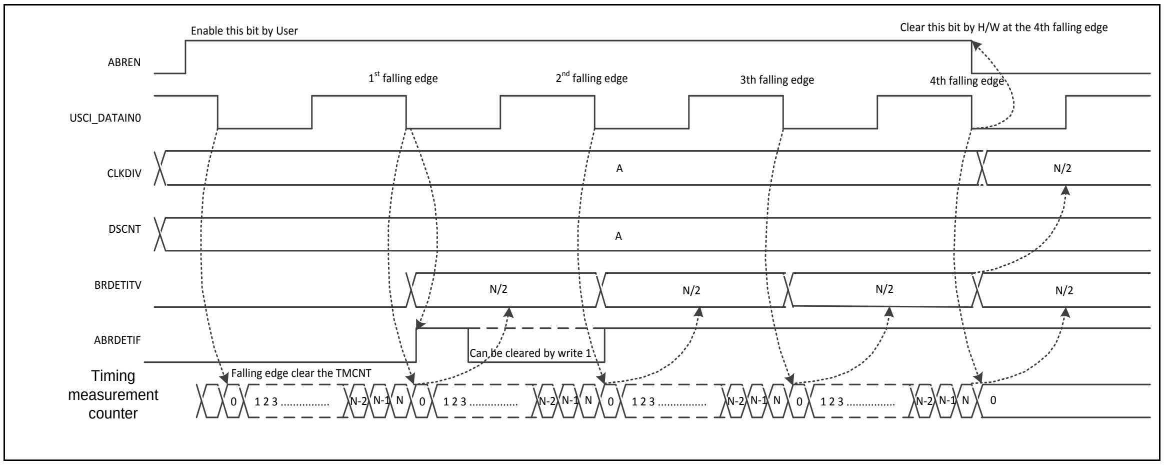

Enable this bit by User
Clear this bit by H/W at the 4th falling edge
ABREN
1st falling edge
2nd falling edge
3th falling edge
4th falling edge
USCI_DATAIN0
CLKDIV
A
N/2
DSCNT
A
BRDETITV
N/2
N/2
N/2
N/2
ABRDETIF
Can be cleared by write 1
Falling edge clear the TMCNT
Timing measurement counter
0 1 2 3 .... N-2 N-1 N 0 1 2 3 .... N-2 N-1 N 0 1 2 3 .... N-2 N-1 N 0 1 2 3 .... N-2 N-1 N 0

Figure 6.13-5 UART Auto Baud Rate Control

6.13.5.8 Auto Flow Control

The UART supports hardware auto-flow control that provides nRTS flow control on receiver buffer indicator RXFULL (UUART_BUFSTS[1]). When the buffer is full (RXFULL = 1), the nRTS is de-asserted.

The UART also provides nCTS flow control on transmitter. The nCTS is used to control the transmitted data is sent out when the nCTS is asserted.

6.13.5.9 RS-485 Support

The UART controller can play the role of the RS-485 master transmitter and 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 the bit15 of each data to control the parity bit (PARITYEN (UUART_PROTCTL[1]) be set) when the STICKEN (UUART_PROTCTL[26]) is set. For example, if the STICKEN is set to 1 and data sequence are 0x8015, 0x8033, 0x0055, 0x0033 and 0x80AA the transmitted parity of data 0x15, 0x33, 0x55 0x33 and 0xAA will be 1, 1, 0, 0 and 1.

The UART controller also can play as an RS-485 addressable slave, the protocol-related error of PARITYERR (UUART_PROTSTS[5]) can be acted as the address bit detection when the PARITYEN (UUART_PROTCTL[1]), EVENPARITY (UUART_PROTCTL[2]) and STICKEN (UUART_PROTCTL[26]) were set. If the PARITYERR was set, it means that the address bit in the received bus is detected otherwise, the data is received into Buffer.

6.13.5.10 Wake-up Function

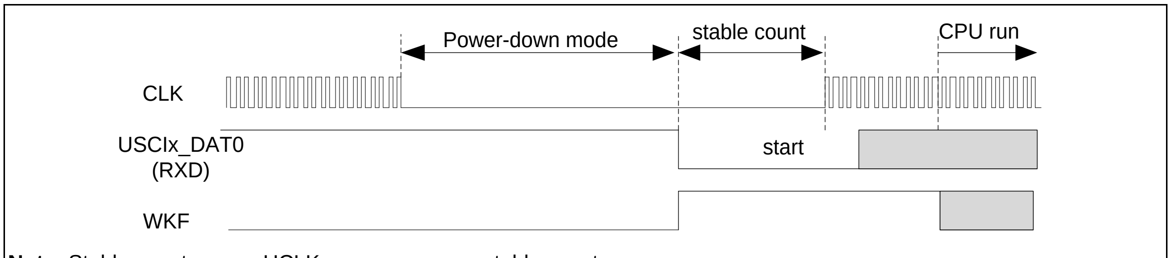
The USCI Controller in UART mode supports wake-up system function. The wake-up source includes incoming data and nCTS pin. Each wake-up source description as following:

(a) Incoming data wake-up

When system is in power-down and both of the WKEN (UUART_WKCTL [0]) and DATWKEN (UUART_PROTCTL[9]) are set, the toggle of incoming data pin can wake up the system. In order to receive the incoming data after the system wake-up, the WAKECNT (UUART_PROTCTL[14:11]) shall be set. These bits field of WAKECNT (UUART_PROCTL[14:11]) indicate how many clock cycle selected by f_{PDS_CLK} do the controller can get the 1st bit (start bit) when the device is wakeup from Power-down mode.

Note 1: By the WAKECNT is loaded into the hardware counter at the time of WKF (UUART_WKSTS[0]) is cleared so that the user shall clear the wakeup flag first to make sure the time period of WAKECNT is closed to the wake time of system.

Note 2: In order to receive the incoming data, the relation between the selected clock stable and the baud rate shall be take care. (for example: The stable time of HXT is 4096 clock period).



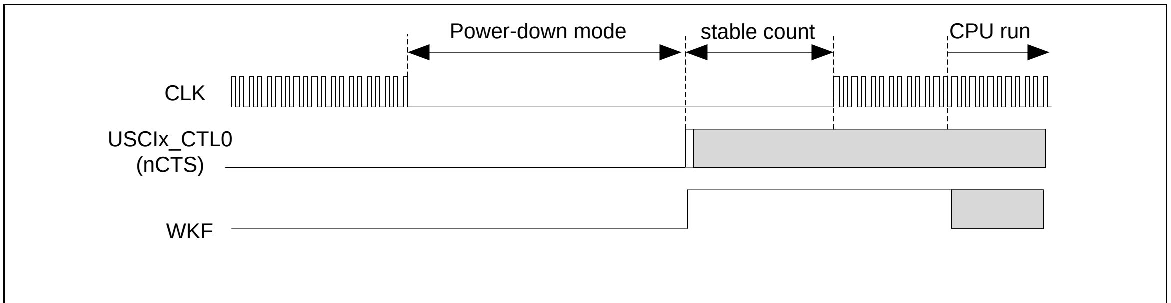
Note: Stable count means HCLK source recovery stable count.

Figure 6.13-6 Incoming Data Wake-Up

(b) nCTS pin wake-up

When system is in power-down and both of the WKEN (UUART_WKCTL [0]) and CTSWKEN (UUART_PROTCTL[10]) are set, the toggle of nCTS pin can wake up the system.

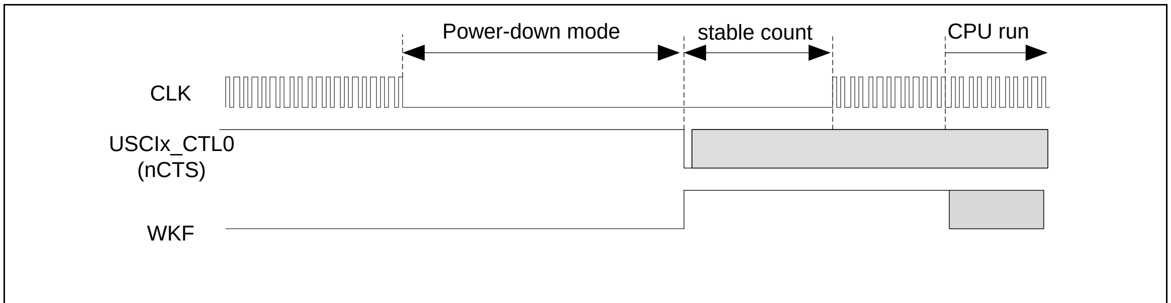
Case 1 (nCTS transition from low to high):



Note: Stable count means HCLK source recovery stable count.

Figure 6.13-7 nCTS Wake-Up Case 1

Case 2 (nCTS transition from high to low):



Note 1: Stable count means HCLK source recovery stable count.

Figure 6.13-8 nCTS Wake-Up Case 2

6.13.5.11 Interrupt Events

Protocol Interrupt Events

The following protocol-related events are generated in UART mode and can lead to a protocol interrupt.

Please note that the bits in register UUART_PROTSTS are not automatically cleared by hardware and have to be cleared by software in order to monitor new incoming events.

Receiver Line Status:

The protocol-related error FRMERR (UUART_PROTSTS[6]) or PARITYERR (UUART_PROTSTS[5]) are two flags that are assigned to each received data word in the corresponding receiver buffer status registers.

In UART mode, the result of the parity check by the protocol-related error indication (0 = received parity bit equal to calculated parity value), and the result of frame check by the protocol-related error indication (0 = received stop bit equal to the format value ‘1’). This information is elaborated for each data frame.

The break error flag (BREAK (UUART_PROTSTS[7])) is assigned when the receive data is 0, the received parity and the stop bit are also 0.

The interrupt indicates that there are parity error, frame error or the break data detection in the BREAK, FRMERR, PARITYERR (UUART_PROTSTS[7:5]) bits. The controller wil issue an interrupt if RLSIEN (UUART_PROTIEN[2]) is also set to 1.

Auto Baud Rate Detection:

The auto baud rate interrupt flag, ABRDETIF (UUART_PROTSTS [9]), indicates that the timing measurement counter has getting 2-bit duration for auto baud rate capture function. The controller wil issue an interrupt if ABRIEN (UUART_PROTIEN[1]) is also set to 1.

The auto baud rate detection function will be enabled in the first falling edge of receiver signal. The auto baud rate detection function is measurement after the next following falling is detected and it is finished when the frame transfer done. After the transfer done, the timing measurement counter value divided by twice is equal to the number of sample time per bit. The user can read the value of BRDETITV (UUART_PROTCTL[24:16]) and write into the baud rate generator register CLKDIV (UUART_BRGEN[25:16]).

Data Transfer Interrupt Handling

The data transfer interrupts indicate events related to UART frame handling.

Transmit Start Interrupt:

Bit TXSTIF (UUART_PROTSTS[1]) is set after the start bit of a data word. In buffer mode, this is the earliest point in time when a new data word can be written to UUART_TXDAT. The controller wil issue an interrupt if TXSTIEN (UUART_INTEN[1]) is also set to 1.

Transmitter Finished:

This interrupt indicates that the transmitter has completely finished all data in the buffer. Bit TXENDIF (UUART_PROTSTS[2]) becomes set at the end of the last stop bit. The controller wil issue an interrupt if TXENDIEN (UUART_INTEN[2]) is also set to 1.

Receiver Starts Interrupt:

Bit RXSTIF (UUART_PROTSTS[3]) is set after the sample point of the start bit. The controller wil issue an interrupt if RXSTIEN (UUART_INTEN[3]) is also set to 1.

Receiver Frame Finished:

This interrupt indicates that the receiver has completely finished a frame. Bit RXENDIF (UUART_PROTSTS[4]) becomes set at the end of the last receive bit. The controller wil issue an interrupt if RXENDIEN (UUART_INTEN[4]) is also set to 1.

6.13.5.12 Programming Example

It is recommended to configure all the setting of UART protocol in idle state of USCI controller with FUNMODE = 0x0 (UUART_CTL[2:1] = 0x0), and then enable the UART transmission by setting FUNMODE (UUART_CTL[2:0]) as 0x2.

The following steps are used to configure the UART protocol setting and the data transmission.

1) Set FUNMODE (UUART_CTL[2:0]) to 0x2 to select UART protocol.

2) Write baud rate generator register UUART_BRGEN to select desired baud rate.

   1. Set SPCLKSEL (UUART_BRGEN[3:2]), PTCLKSEL (UUART_BRGEN[1]) and RCLKSEL (UUART_BRGEN[0]) to select the clock source.

   2. Configure CLKDIV (UUART_BRGEN[25:16]), DSCNT (UUART_BRGEN[14:10]) and PDSCNT (UUART_BRGEN[9:8]) to determine the baud rate divider.

3) Write line control register UUART_LINECTL and protocol control register UUART_PROTCTL to configure the transmission data format and UART protocol setting.

   1. Program data field length in DWIDTH (UUART_LINECTL[11:8]).

   2. Enable parity bit and determine the parity bit type by setting EVENPARITY (UUART_PROTCTL[2]) and PARITYEN (UUART_PROTCTL[1]).

   3. Configure stop bit length by setting STOPB (UUART_PROTCTL[0]).

   4. Enable LSB (UUART_LINECTL[0]) to select LSB first transmission for UART protocol.

   5. Set EDGEDET (UUART_DATIN0[4:3]) to 0x2 to select the detected edge as falling edge for receiver start bit detection.

4) Set PROTEN (UUART_PROTCTL[31]) to 1 to enable UART protocol.

5) Transmit and receive data.

   1. Write transmit data register UUART_TXDAT to transmit data.

   2. Wait until TXSTIF(UUART_PROTSTS[1]) is set and then user can write the next data in UUART_TXDAT.

   3. Enable parity bit and determine the parity bit type by setting EVENPARITY (UUART_PROTCTL[2]) and PARITYEN (UUART_PROTCTL[1]).

   4. When TXENDIF(UUART_PROTSTS[2]) is set, the transmit buffer is empty and the stop bit of stop bit of the last data has been transmitted.

   5. If RXENDIF(UUART_PROTSTS[4]) is set, the receiver has finished a data frame completely. User can get the data by reading receive data register UUART_RXDAT.

6.13.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UUART Base Address: **UUARTn_BA = 0x4007_0000 + (0x10_0000 * n)** **n=0**				
UUART_CTL **n=0**	UUARTn_BA+0x00	R/W	USCI Control Register	0x0000_0000
UUART_INTEN **n=0**	UUARTn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000
UUART_BRGEN **n=0**	UUARTn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
UUART_DATIN0 **n=0**	UUARTn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000
UUART_CTLIN0 **n=0**	UUARTn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000
UUART_CLKIN **n=0**	UUARTn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000
UUART_LINECTL **n=0**	UUARTn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
UUART_TXDAT **n=0**	UUARTn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
UUART_RXDAT **n=0**	UUARTn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
UUART_BUFCTL **n=0**	UUARTn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000
UUART_BUFSTS **n=0**	UUARTn_BA+0x3C	R/W	USCI Transmit/Receive Buffer Status Register	0x0000_0101
UUART_PDMACTL **n=0**	UUARTn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000
UUART_WKCTL **n=0**	UUARTn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
UUART_WKSTS **n=0**	UUARTn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
UUART_PROTCTL **n=0**	UUARTn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000
UUART_PROTIEN **n=0**	UUARTn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
UUART_PROTSTS **n=0**	UUARTn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

6.13.7 Register Description

USCI Control Register (UUART_CTL)

Register	Offset	R/W	Description	Reset Value
UUART_CTL **n=0**	UUARTn_BA+0x00	R/W	USCI 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					**FUNMODE**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 0x0 to FUNMODE. 0x0 = The USCI is disabled. All protocol related state machines are set to idle state. 0x1 = The SPI protocol is selected. 0x2 = The UART protocol is selected. 0x4 = The I^{2}C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Interrupt Enable Register (UUART_INTEN)

Register	Offset	R/W	Description	Reset Value
UUART_INTEN **n=0**	UUARTn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**RXENDIEN**	**Receive End Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive finish event. 0 = The receive end interrupt is disabled. 1 = The receive end interrupt is enabled.
[3]	**RXSTIEN**	**Receive Start Interrupt Enable BIt** This bit enables the interrupt generation in case of a receive start event. 0 = The receive start interrupt is disabled. 1 = The receive start interrupt is enabled.
[2]	**TXENDIEN**	**Transmit End Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit finish event. 0 = The transmit finish interrupt is disabled. 1 = The transmit finish interrupt is enabled.
[1]	**TXSTIEN**	**Transmit Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit start event. 0 = The transmit start interrupt is disabled. 1 = The transmit start interrupt is enabled.
[0]	**Reserved**	Reserved.

USCI Baud Rate Generator Register (UUART_BRGEN)

Register	Offset	R/W	Description	Reset Value
UUART_BRGEN **n=0**	UUARTn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

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

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} ($f_{DIV\_CLK} = f_{PROT\_CLK} / (CLKDIV+1)$). **Note:** In UART function, it can be updated by hardware in the 4th falling edge of the input data 0x55 when the auto baud rate function (ABREN(UUART_PROTCTL[6])) is enabled. The revised value is the average bit time between bit 5 and bit 6. The user can use revised CLKDIV and new BRDETITV (UUART_PROTCTL[24:16]) to calculate the precise baud rate.
[15]	**Reserved**	Reserved.
[14:10]	**DSCNT**	**Denominator for Sample Counter** This bit field defines the divide ratio of the sample clock f_{SAMP_CLK}. The divided frequency $f_{DS\_CNT} = f_{PDS\_CNT} / (DSCNT+1)$. **Note:** The maximum value of DSCNT is 0xF on UART mode and suggest to set over 4 to confirm the receiver data is sampled in right value.
[9:8]	**PDSCNT**	**Pre-divider for Sample Counter** This bit field defines the divide ratio of the clock division from sample clock f_{SAMP_CLK}. The divided frequency $f_{PDS\_CNT} = f_{SAMP\_CLK} / (PDSCNT+1)$.
[7:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Timing Measurement Counter Clock Source Selection** 0 = Timing measurement counter with f_{PROT_CLK}. 1 = Timing measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Timing Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Timing measurement counter is Disabled. 1 = Timing measurement counter is Enabled.

[3:2]	**SPCLKSEL**	**Sample Clock Source Selection** This bit field used for the clock source selection of a sample clock (f_{SAMP_CLK}) for the protocol processor. 00 = f_{SAMP_CLK} = f_{DIV_CLK}. 01 = f_{SAMP_CLK} = f_{PROT_CLK}. 10 = f_{SAMP_CLK} = f_{SCLK}. 11 = f_{SAMP_CLK} = f_{REF_CLK}.
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source signal of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source signal of reference clock (f_{REF_CLK}). 0 = Peripheral device clock f_{PCLK}. 1 = Reserved.

USCI Input Data Signal Configuration (UUART_DATIN0)

Register	Offset	R/W	Description	Reset Value
UUART_DATIN0 n=0	UUARTn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**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			EDGEDET		ININV	Reserved	SYNCSEL

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:3]	**EDGEDET**	**Input Signal Edge Detection Mode** This bit field selects which edge actives the trigger event of input data signal. 00 = The trigger event activation is disabled. 01 = A rising edge activates the trigger event of input data signal. 10 = A falling edge activates the trigger event of input data signal. 11 = Both edges activate the trigger event of input data signal. **Note:** In UART function mode, it is suggested to set this bit field as 10.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Signal Synchronization Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Input Control Signal Configuration (UUART_CTLIN0)

Register	Offset	R/W	Description	Reset Value
UUART_CTLIN0 **n=0**	UUARTn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Input Clock Signal Configuration (UUART_CLKIN)

Register	Offset	R/W	Description	Reset Value
UUART_CLKIN **n=0**	UUARTn_BA+0x28	R/W	USCI Input Clock Signal 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SYNCSEL**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Line Control Register (UUART_LINECTL)

Register	Offset	R/W	Description	Reset Value
UUART_LINECTL **n=0**	UUARTn_BA+0x2C	R/W	USCI 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				**DWIDTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTLOINV	**Reserved**	**DATOINV**	**Reserved**				**LSB**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0x0: The data word contains 16 bits located at bit positions [15:0]. 0x1: Reserved. 0x2: Reserved. 0x3: Reserved. 0x4: The data word contains 4 bits located at bit positions [3:0]. 0x5: The data word contains 5 bits located at bit positions [4:0]. ... 0xF: The data word contains 15 bits located at bit positions [14:0]. **Note:** In UART protocol, the length can be configured as 6~13 bits.
[7]	**CTLOINV**	**Control Signal Output Inverse Selection** This bit defines the relation between the internal control signal and the output control signal. 0 = No effect. 1 = The control signal will be inverted before its output. **Note:** In UART protocol, the control signal means nRTS signal.
[6]	**Reserved**	Reserved.
[5]	**DATOINV**	**Data Output Inverse Selection** This bit defines the relation between the internal shift data value and the output data signal of USCIx_DAT1 pin. 0 = The value of USCIx_DAT1 is equal to the data shift register. 1 = The value of USCIx_DAT1 is the inversion of data shift register.
[4:1]	**Reserved**	Reserved.

[0]	**LSB**	**LSB First Transmission Selection** 0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (UUART_TXDAT)

Register	Offset	R/W	Description	Reset Value
UUART_TXDAT **n=0**	UUARTn_BA+0x30	W	USCI Transmit 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**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission.

USCI Receive Data Register (UUART_RXDAT)

Register	Offset	R/W	Description	Reset Value
UUART_RXDAT **n=0**	UUARTn_BA+0x34	R	USCI 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**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer. **Note:** RXDAT[15:13] indicate the same frame status of BREAK, FRMERR and PARITYERR (UUART_PROTSTS[7:5]).

USCI Transmitter/Receive Buffer Control Register (UUART_BUFCTL)

Register	Offset	R/W	Description	Reset Value
UUART_BUFCTL **n=0**	UUARTn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset the receive-related counters, state machine, and the content of receive shift register and data buffer. **Note 1:** It is cleared automatically after one PCLK cycle. **Note 2:** It is suggested to check the RXBUSY (UUART_PROTSTS[10]) before this bit will be set to 1.
[16]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset the transmit-related counters, state machine, and the content of transmit shift register and data buffer**.** **Note:** It is cleared automatically after one PCLK cycle.
[15]	**RXCLR**	**Clear Receive Buffer** 0 = No effect. 1 = The receive buffer is cleared (filling level is cleared and output pointer is set to input pointer value). Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[14]	**RXOVIEN**	**Receive Buffer Overrun Error Interrupt Enable Control** 0 = Receive overrun interrupt Disabled. 1 = Receive overrun interrupt Enabled.
[13:8]	**Reserved**	Reserved.
[7]	**TXCLR**	**Clear Transmit Buffer** 0 = No effect. 1 = The transmit buffer is cleared (filling level is cleared and output pointer is set to input pointer value). Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.

[6:0]	**Reserved**	Reserved.

USCI Transmit/Receive Buffer Status Register (UUART_BUFSTS)

Register	Offset	R/W	Description	Reset Value
UUART_BUFSTS **n=0**	UUARTn_BA+0x3C	R/W	USCI Transmit/Receive Buffer Status Register	0x0000_0101

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**TXFULL**	**Transmit Buffer Full Indicator** 0 = Transmit buffer is not full. 1 = Transmit buffer is full.
[8]	**TXEMPTY**	**Transmit Buffer Empty Indicator** 0 = Transmit buffer is not empty. 1 = Transmit buffer is empty.
[7:4]	**Reserved**	Reserved.
[3]	**RXOVIF**	**Receive Buffer Over-run Error Interrupt Status** This bit indicates that a receive buffer overrun error event has been detected. If RXOVIEN (UUART_BUFCTL[14]) is enabled, the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit. 0 = A receive buffer overrun error event has not been detected. 1 = A receive buffer overrun error event has been detected.
[2]	**Reserved**	Reserved.
[1]	**RXFULL**	**Receive Buffer Full Indicator** 0 = Receive buffer is not full. 1 = Receive buffer is full.
[0]	**RXEMPTY**	**Receive Buffer Empty Indicator** 0 = Receive buffer is not empty. 1 = Receive buffer is empty.

USCI PDMA Control Register (UUART_PDMACTL)

Register	Offset	R/W	Description	Reset Value
UUART_PDMACTL **n=0**	UUARTn_BA+0x40	R/W	USCI 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				**PDMAEN**	**RXPDMAEN**	**TXPDMAEN**	**PDMARST**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PDMAEN**	**PDMA Mode Enable Bit** 0 = PDMA function Disabled. 1 = PDMA function Enabled.
[2]	**RXPDMAEN**	**PDMA Receive Channel Available** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[1]	**TXPDMAEN**	**PDMA Transmit Channel Available** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled.
[0]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the USCI’s PDMA control logic. This bit will be cleared to 0 automatically.

USCI Wake-up Control Register (UUART_WKCTL)

Register	Offset	R/W	Description	Reset Value
UUART_WKCTL **n=0**	UUARTn_BA+0x54	R/W	USCI 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					PDBOPT	Reserved	WKEN

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDBOPT**	**Power Down Blocking Option** 0 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, MCU will stop the transfer and enter Power-down mode immediately. 1 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, the on-going transfer will not be stopped and MCU will enter idle mode immediately.
[1]	**Reserved**	Reserved.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (UUART_WKSTS)

Register	Offset	R/W	Description	Reset Value
UUART_WKSTS **n=0**	UUARTn_BA+0x58	R/W	USCI 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							**WKF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – UART (UUART_PROTCTL)

Register	Offset	R/W	Description	Reset Value
UUART_PROTCTL **n=0**	UUARTn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	Reserved	BCEN	Reserved	Reserved	STICKEN	Reserved	BRDETITV
23	22	21	20	19	18	17	16
BRDETITV							
15	14	13	12	11	10	9	8
Reserved	WAKECNT				CTSWKEN	DATWKEN	Reserved
7	6	5	4	3	2	1	0
Reserved	ABREN	RTSAUDIREN	CTSAUTOEN	RTSAUTOEN	EVENPARITY	PARITYEN	STOPB

Bits	Description	
[31]	**PROTEN**	**UART Protocol Enable Bit** 0 = UART Protocol Disabled. 1 = UART Protocol Enabled.
[30]	**Reserved**	Reserved.
[29]	**BCEN**	**Transmit Break Control Enable Bit** 0 = Transmit Break Control Disabled. 1 = Transmit 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 line and has no effect on the transmitter logic.
[27]	**Reserved**	Reserved.
[26]	**STICKEN**	**Stick Parity Enable Bit** 0 = Stick parity Disabled. 1 = Stick parity Enabled. **Note:** Refer to section 6.13.5.9 for detailed information.
[25]	**Reserved**	Reserved.
[24:16]	**BRDETITV**	**Baud Rate Detection Interval** This bit fields indicate how many clock cycle selected by TMCNTSRC (UUART_BRGEN [5]) does the slave calculates the baud rate in one bits. The order of the bus shall be 1 and 0 step by step (e.g. the input data pattern shall be 0x55). The user can read the value to know the current input baud rate of the bus whenever the ABRDETIF (UUART_PROTCTL[9]) is set. **Note:** This bit can be cleared to 0 by software writing '0' to the **BRDETITV**.
[15]	**Reserved**	Reserved.
[14:11]	**WAKECNT**	**Wake-up Counter** These bits field indicate how many clock cycle selected by f_{PDS_CNT} do the slave can get the 1st bit (start bit) when the device is wake-up from Power-down mode.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

[10]	**CTSWKEN**	**nCTS Wake-up Mode Enable Bit** 0 = nCTS wake-up mode Disabled. 1 = nCTS wake-up mode Enabled.
[9]	**DATWKEN**	**Data Wake-up Mode Enable Bit** 0 = Data wake-up mode Disabled. 1 = Data wake-up mode Enabled.
[6]	**ABREN**	**Auto-baud Rate Detect Enable Bit** 0 = Auto-baud rate detect function Disabled. 1 = Auto-baud rate detect function Enabled. **Note:** When the auto baud rate detect operation finishes, hardware will clear this bit. The associated interrupt ABRDETIF (UUART_PROTST[9]) will be generated (If ARBIEN (UUART_PROTIEN [1]) is enabled).
[5]	**RTSAUDIREN**	**nRTS Auto Direction Enable Bit** When nRTS auto direction is enabled, if the transmitted bytes in the TX buffer is empty, the UART will reassert nRTS signal. 0 = nRTS auto direction control Disabled. 1 = nRTS auto direction control Enabled. **Note 1:** This bit is used for nRTS auto direction control for RS485. **Note 2:** This bit has effect only when the RTSAUTOEN is not set.
[4]	**CTSAUTOEN**	**nCTS Auto-flow Control Enable Bit** 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 if nCTS input is dis-asserted). 0 = nCTS auto-flow control Disabled. 1 = nCTS auto-flow control Enabled.
[3]	**RTSAUTOEN**	**nRTS Auto-flow Control Enable Bit** When nRTS auto-flow is enabled, if the receiver buffer is full (RXFULL (UUART_BUFSTS[1] = 1'b1)), the UART will de-assert nRTS signal. 0 = nRTS auto-flow control Disabled. 1 = nRTS auto-flow control Enabled. **Note:** This bit has effect only when the RTSAUDIREN is not set.
[2]	**EVENPARITY**	**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 PARITYEN is set.
[1]	**PARITYEN**	**Parity Enable Bit** This bit defines the parity bit is enabled in an UART frame. 0 = The parity bit Disabled. 1 = The parity bit Enabled.
[0]	**STOPB**	**Stop Bits** This bit defines the number of stop bits in an UART frame. 0 = The number of stop bits is 1. 1 = The number of stop bits is 2.

USCI Protocol Interrupt Enable Register – UART (UUART_PROTIEN)

Register	Offset	R/W	Description	Reset Value
UUART_PROTIEN **n=0**	UUARTn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable Bit** 0 = Receive line status interrupt Disabled. 1 = Receive line status interrupt Enabled. **Note:** UUART_PROTSTS[7:5] indicates the current interrupt event for receive line status interrupt.
[1]	**ABRIEN**	**Auto-baud Rate Interrupt Enable Bit** 0 = Auto-baud rate interrupt Disabled. 1 = Auto-baud rate interrupt Enabled.
[0]	**Reserved**	Reserved.

USCI Protocol Status Register – UART (UUART_PROTSTS)

Register	Offset	R/W	Description	Reset Value
UUART_PROTSTS **n=0**	UUARTn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**CTSLV**	**CTSSYNCLV**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ABERRSTS**	**RXBUSY**	**ABRDETIF**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BREAK	**FRMERR**	**PARITYERR**	**RXENDIF**	**RXSTIF**	**TXENDIF**	**TXSTIF**	**Reserved**

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**CTSLV**	**nCTS Pin Status (Read Only)** This bit used to monitor the current status of nCTS pin input. 0 = nCTS pin input is low level voltage logic state. 1 = nCTS pin input is high level voltage logic state.
[16]	**CTSSYNCLV**	**nCTS Synchronized Level Status (Read Only)** This bit used to indicate the current status of the internal synchronized nCTS signal. 0 = The internal synchronized nCTS is low. 1 = The internal synchronized nCTS is high.
[15:12]	**Reserved**	Reserved.
[11]	**ABERRSTS**	**Auto-baud Rate Error Status** This bit is set when auto-baud rate detection counter overrun. When the auto-baud rate counter overrun, the user shall revise the CLKDIV (UUART_BRGEN[25:16]) value and enable ABREN (UUART_PROTCTL[6]) to detect the correct baud rate again. 0 = Auto-baud rate detect counter is not overrun. 1 = Auto-baud rate detect counter is overrun. **Note 1:** This bit is set at the same time of ABRDETIF. **Note 2:** This bit can be cleared by writing “1” to ABRDETIF or ABERRSTS.
[10]	**RXBUSY**	**RX Bus Status Flag (Read Only)** This bit indicates the busy status of the receiver. 0 = The receiver is Idle. 1 = The receiver is BUSY.
[9]	**ABRDETIF**	**Auto-baud Rate Interrupt Flag** This bit is set when auto-baud rate detection is done among the falling edge of the input data. If the ABRIEN (UUART_PROTCTL[6]) is set, the auto-baud rate interrupt will be generated. This bit can be set 4 times when the input data pattern is 0x55 and it is cleared before the next falling edge of the input bus.

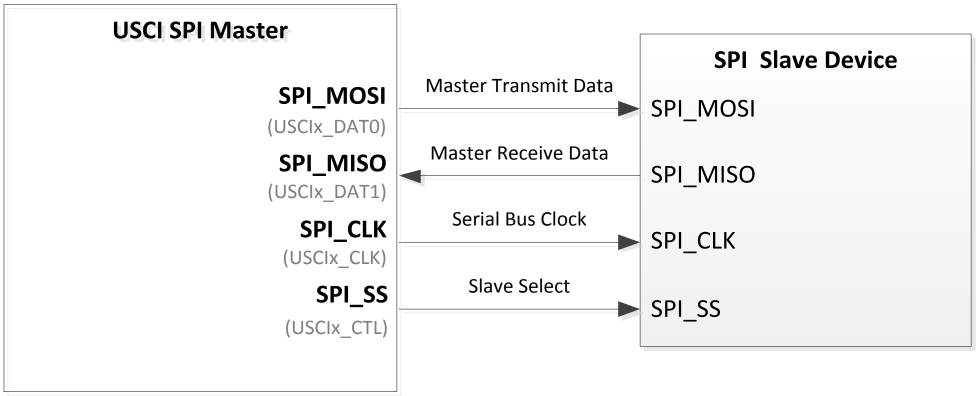
		0 = Auto-baud rate detect function is not done. 1 = One Bit auto-baud rate detect function is done. **Note:** This bit can be cleared by writing “1” to it.
[8]	**Reserved**	Reserved.
[7]	**BREAK**	**Break Flag** This bit is set to logic 1 whenever the received data input (RX) is held in the “spacing 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 is generated. 1 = Break is generated in the receiver bus. **Note:** This bit can be cleared by write “1” among the BREAK, FRMERR and PARITYERR bits.
[6]	**FRMERR**	**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 can be cleared by write “1” among the BREAK, FRMERR and PARITYERR bits.
[5]	**PARITYERR**	**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 write “1” among the BREAK, FRMERR and PARITYERR bits.
[4]	**RXENDIF**	**Receive End Interrupt Flag** 0 = A receive finish interrupt status has not occurred. 1 = A receive finish interrupt status has occurred. **Note:** It is cleared by software writing one into this bit.
[3]	**RXSTIF**	**Receive Start Interrupt Flag** 0 = A receive start interrupt status has not occurred. 1 = A receive start interrupt status has occurred. **Note:** It is cleared by software writing one into this bit.
[2]	**TXENDIF**	**Transmit End Interrupt Flag** 0 = A transmit end interrupt status has not occurred. 1 = A transmit end interrupt status has occurred. **Note:** It is cleared by software writing one into this bit.
[1]	**TXSTIF**	**Transmit Start Interrupt Flag** 0 = A transmit start interrupt status has not occurred. 1 = A transmit start interrupt status has occurred. **Note 1:** It is cleared by software writing one into this bit. **Note 2:** Used for user to load next transmit data when there is no data in transmit buffer.
[0]	**Reserved**	Reserved.

6.14 USCI - SPI Mode

6.14.1 Overview

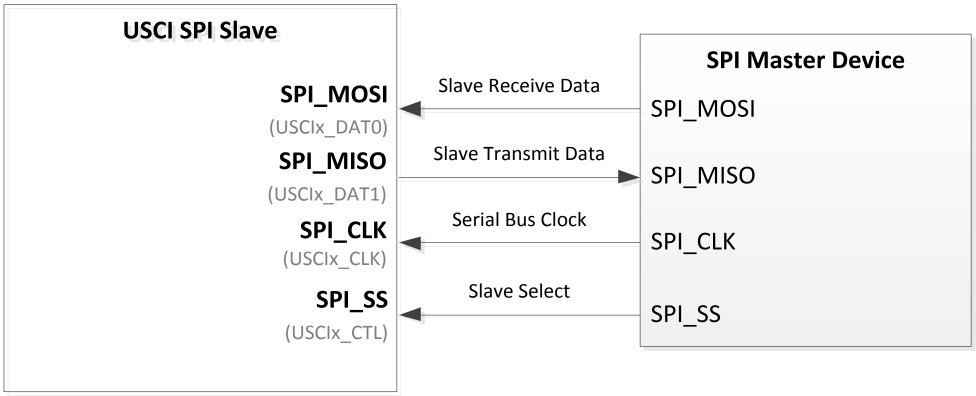
The SPI protocol of USCI controller applies to synchronous serial data communication and allows full duplex transfer. It supports both master and slave operation mode with the 4-wire bi-direction interface. SPI mode of USCI controller performs a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. The SPI mode is selected by FUNMODE (USPI_CTL[2:0]) = 0x1

This SPI protocol can operate as master or slave mode by setting the SLAVE (USPI_PROTCTL[0]) to communicate with the off-chip SPI slave or master device. The application block diagrams in master and slave mode are shown below.



Note: x = 0

Figure 6.14-1 SPI Master Mode Application Block Diagram



Note: x = 0

Figure 6.14-2 SPI Slave Mode Application Block Diagram

6.14.2 Features

- Supports master or slave mode operation (the maximum frequency for Master = f_{PCLK} / 2, for Slave < f_{PCLK} / 5)
- Configurable bit length of a transfer word from 4 to 16-bit

- Supports one transmit buffer and two receive buffers for data payload
- Supports MSB first or LSB first transfer sequence
- Supports word suspend function
- Supports PDMA transfer
- Supports 3-wire, no slave select signal, bi-direction interface
- Supports wake-up function by slave select signal in Slave mode
- Supports one data channel half-duplex transfer

6.14.3 Block Diagram

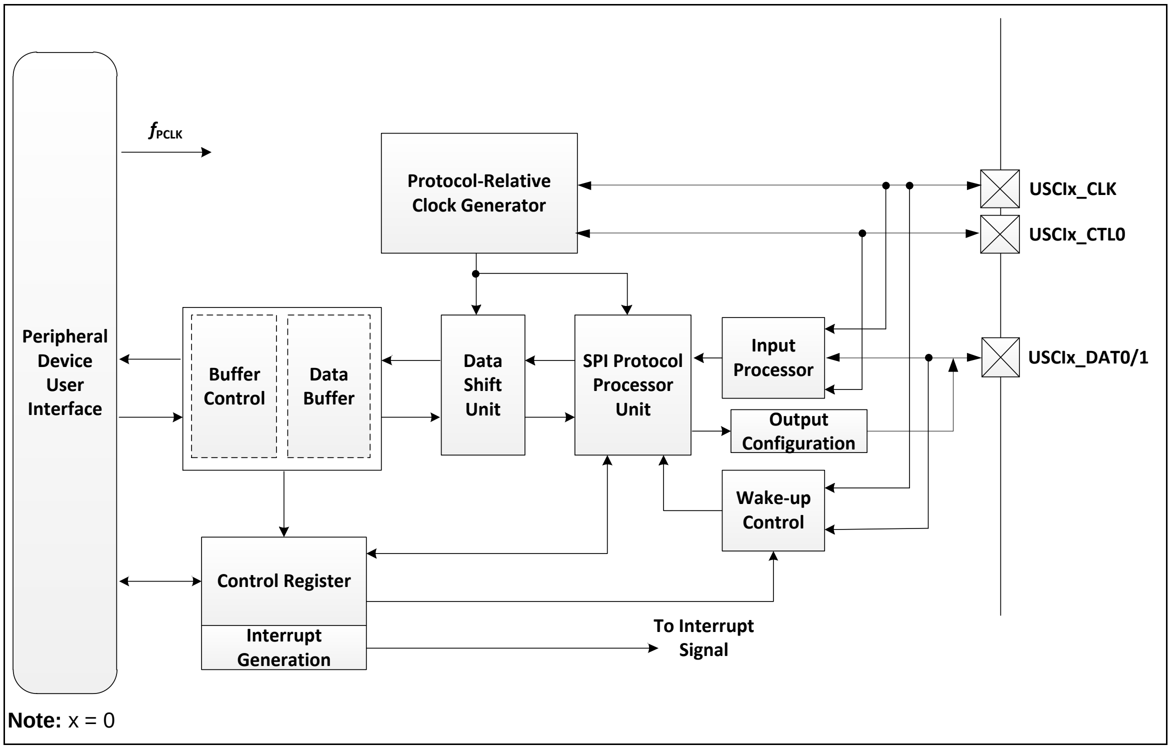


Figure 6.14-3 Block Diagram

6.14.4 Basic Configuration

The basic configurations of USCI0 for SPI mode are as follows.

- USCI0 pins are configured in SYS_GPA_MFPH, SYS_GPB_MFPL, SYS_GPB_MFPH, SYS_GPC_MFPL, SYS_GPC_MFPH, SYS_GPD_MFPL, SYS_GPD_MFPH, and SYS_GPE_MFPL registers.
- Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK1[8]).
- Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).

6.14.5 Functional Description

6.14.5.1 USCI Common Function Description

Please refer to section 6.12.4 for detailed information.

6.14.5.2 Signal Description

A device operating in master mode controls the start and end of a data transfer, as well as the generation of the SPI bus clock and slave select signal. The slave select signal indicates the start and the end of a data transfer, and the master device can use it to enable the transmitting or receiving operations of slave device. Slave device receives the SPI bus clock and optionally a slave select signal for data transaction. The signals for SPI communication are shown in Figure 6.14-4 and Figure 6.14-5.

SPI Mode	Receive Data	Transmit Data	Serial Bus Clock	Slave Select
Full-duplex SPI Master	SPI_MISO (USCIx_DAT1)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
Full-duplex SPI Slave	SPI_MOSI (USCIx_DAT0)	SPI_MISO (USCIx_DAT1)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
Half-duplex SPI Master/Slave	SPI_MOSI (USCIx_DAT0)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)

Table 6.14-1 SPI Communication Signals

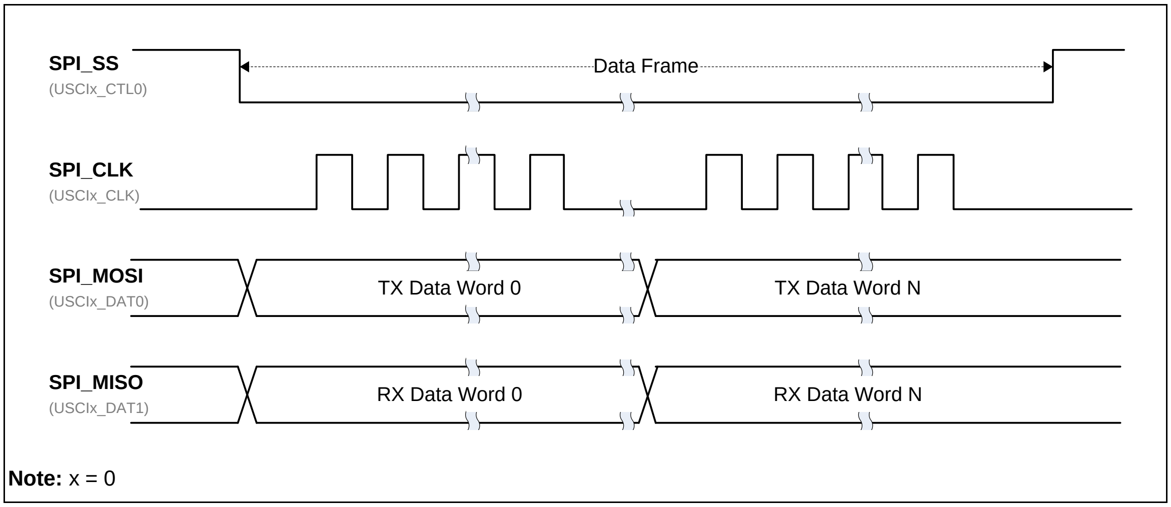


Figure 6.14-4 4-Wire Full-Duplex SPI Communication Signals (Master Mode)

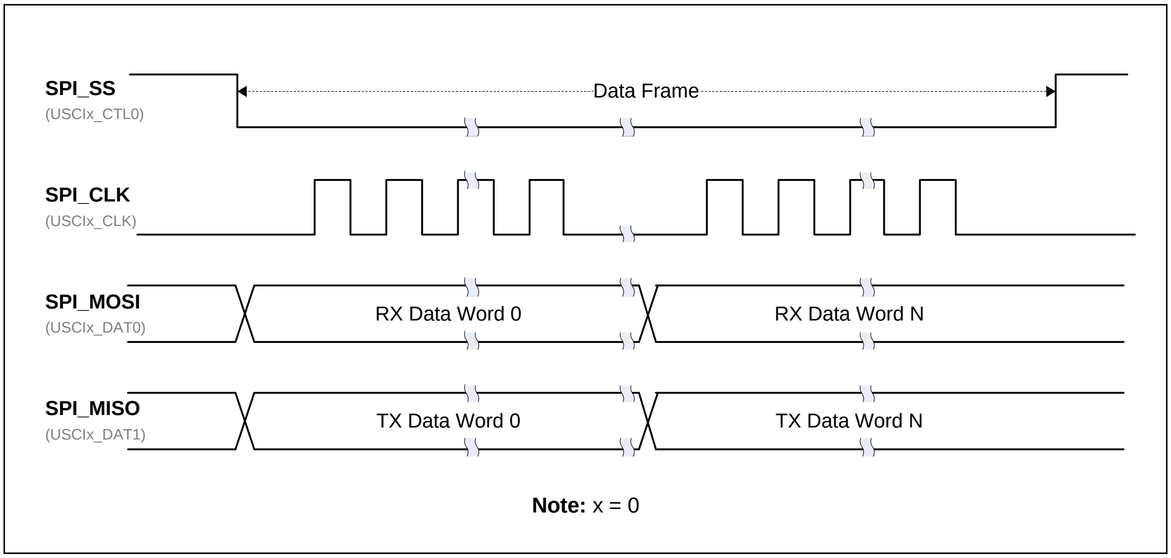


Figure 6.14-54-Wire Full-Duplex SPI Communication Signals (Slave Mode)

6.14.5.3 Serial Bus Clock Configuration

The USCI controller needs the peripheral clock to drive the USCI logic unit to perform the data transfer. The peripheral clock frequency is equal to PCLK frequency.

In Master mode, the frequency of the SPI bus clock is determined by protocol-relative clock generator. In general, the SPI bus clock is denoted as SPI clock. The frequency of SPI clock is half of f_{SAMP_CLK}, which can be selected by SPCLKSEL (USPI_BRGEN[3:2]). Refer to section 6.12.4.5 for details of protocol-relative clock generator.

In slave mode, the SPI bus clock is provided by an off-chip master device. The peripheral clock frequency, f_{PCLK}, of SPI slave device must be 5-times faster than the serial bus clock rate of the SPI master device connected together (i.e. the clock rate of serial bus clock < 1/5 peripheral clock f_{PCLK} in slave mode).

In SPI protocol, SCLKMODE (USPI_PROTCTL[7:6]) defines not only the idle state of serial bus clock but also the serial clock edge used for transmit and receive data. Both master and slave devices on the same communication bus should have the same SCLKMODE configuration. The four kinds of serial bus clock configuration are shown in Table 6.14-2, and timing diagrams are shown in Figure 6.14-6, Figure 6.14-7, Figure 6.14-8 and Figure 6.14-9.

SCLKMODE [1:0]	SPI Clock Idle State	Transmit Timing	Receive Timing
0x0	Low	Falling edge	Rising edge
0x1	Low	Rising edge	Falling edge
0x2	High	Rising edge	Falling edge
0x3	High	Falling edge	Rising edge

Table 6.14-2 Serial Bus Clock Configuration

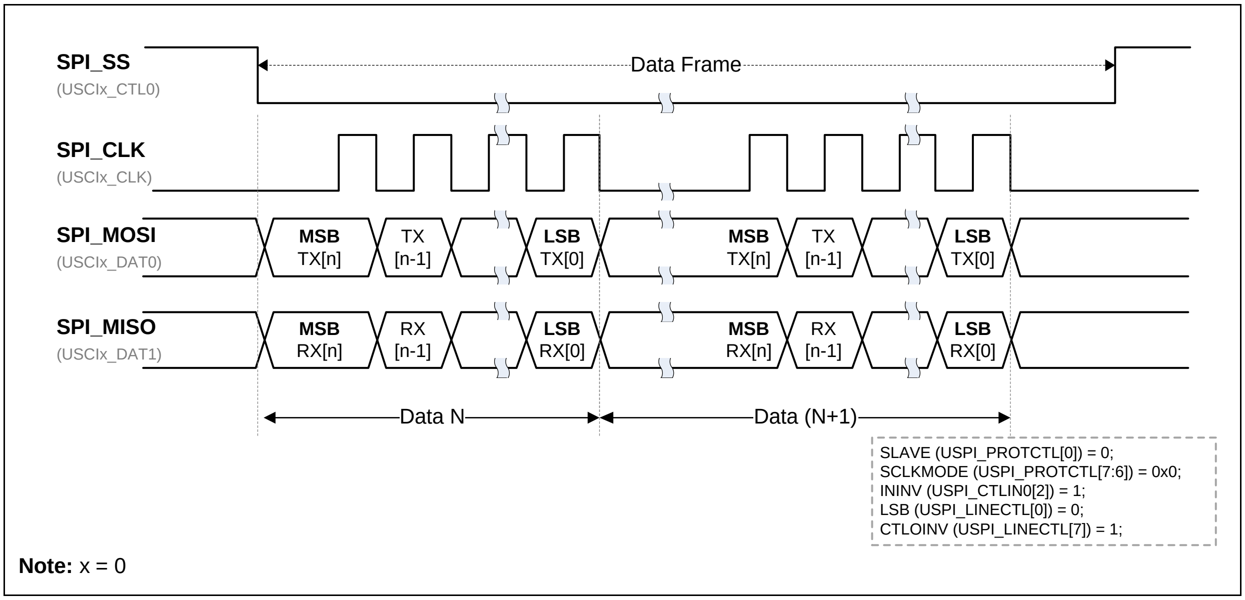


Figure 6.14-6 SPI Communication with Different SPI Clock Configuration (SCLKMODE = 0x0)

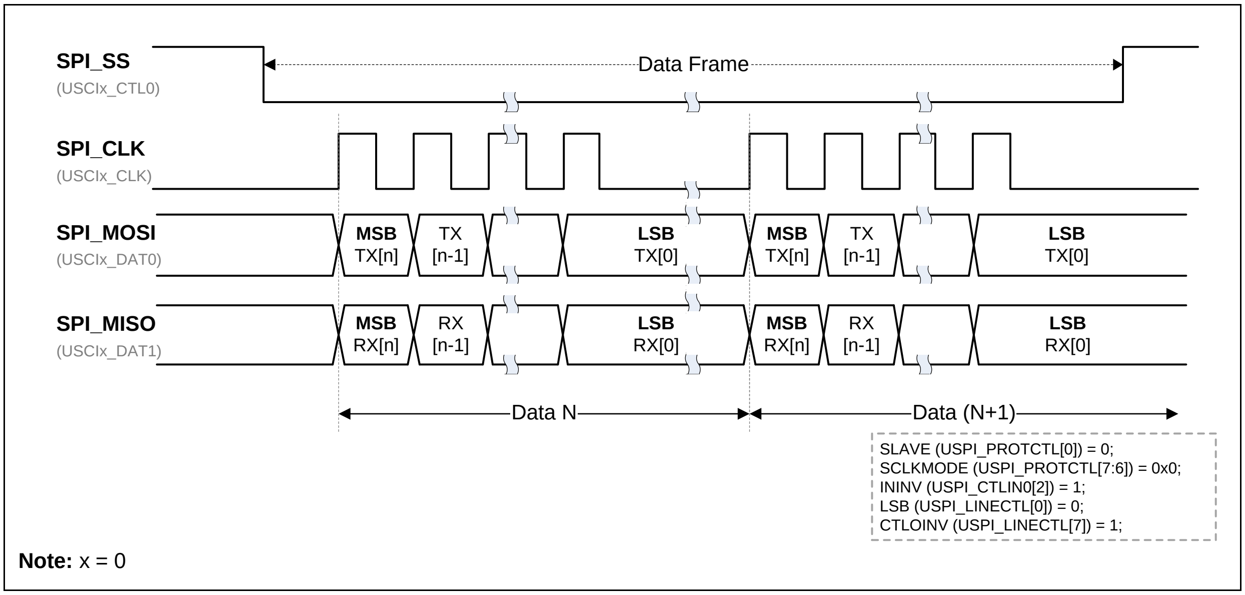


Figure 6.14-7 SPI Communication with Different SPI Clock Configuration (SCLKMODE = 0x1)

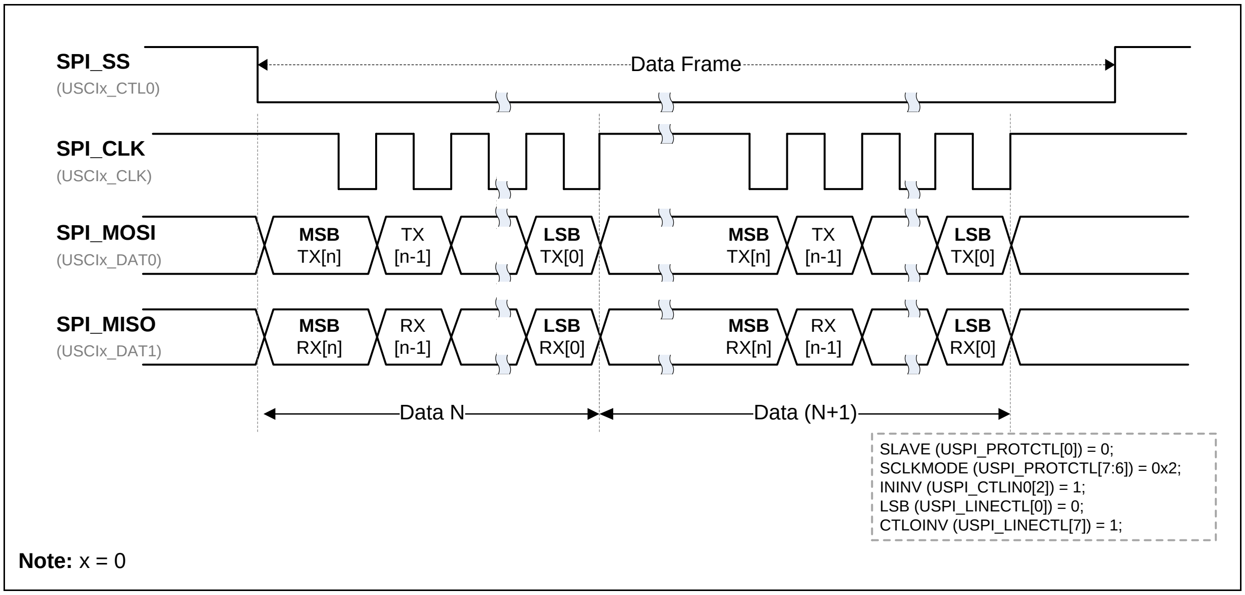


Figure 6.14-8 SPI Communication with Different SPI Clock Configuration (SCLKMODE = 0x2)

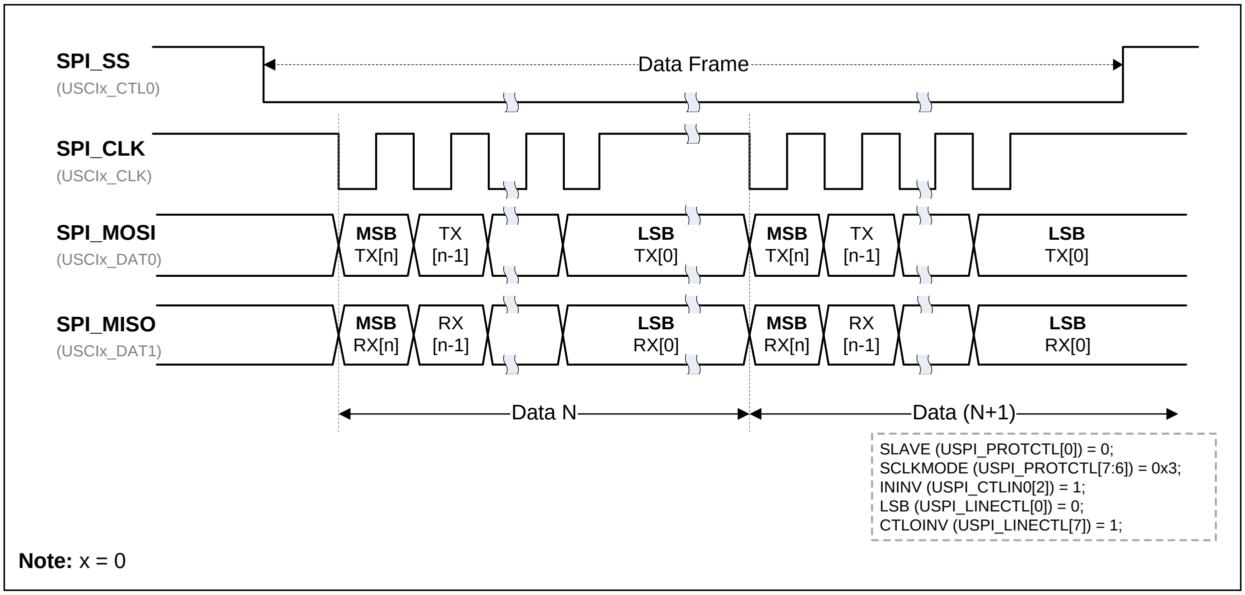


Figure 6.14-9 SPI Communication with Different SPI Clock Configuration (SCLKMODE = 0x3)

6.14.5.4 Slave Select Signal

The slave selection signal of SPI protocol is active high by default. In SPI master mode, the USCI controller can drive the control signal to off-chip SPI slave device through slave select pin SPI_SS (USCIx_CTL0). In SPI slave mode, the received slave select signal can be inverted by ININV (USPI_CTLIN0[2]).

If the slave select signal of external SPI master device is low active, the ININV (USPI_CTLIN[2]) setting of slave device should be set to 1 for the inversion of input control signal. If USCI operates as SPI master mode, the output slave select inversion CTLOINV (USPI_LINECTL[7]) is also needed to set as 1 for the external SPI slave device whose slave select signal is active low.

The duration between the slave select active edge and the first SPI clock input edge shall over 2

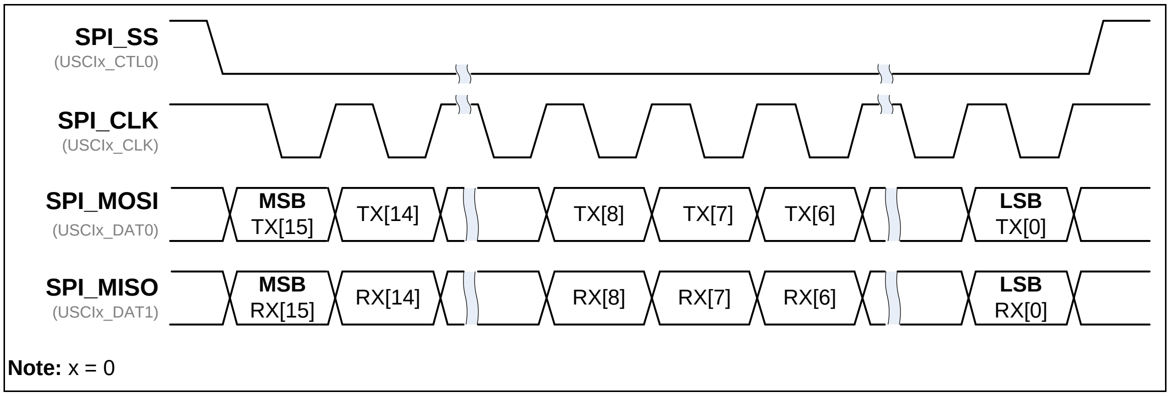
USCI peripheral clock cycles.

The input slave select signal of SPI Slave has to be keep inactive for at least 2 USCI peripheral clock cycles between two consecutive frames in order to correctly detect the end of a frame.

6.14.5.5 Transmit and Receive Data

The bit length of a transmit/receive data word in SPI protocol of USCI controller is defined in DWIDTH (USPI_LINECTL[11:8]), and it can be configured up to 16-bit length and not less than 4-bit length for transmitting and receiving data in SPI communication.

The LSB bit (USPI_LINECTL[0]) defines the order of transfer data bit. If the LSB bit is set to 1, the transmission data sequence is LSB first. If the LSB bit is cleared to 0, the transmission data sequence is MSB first.

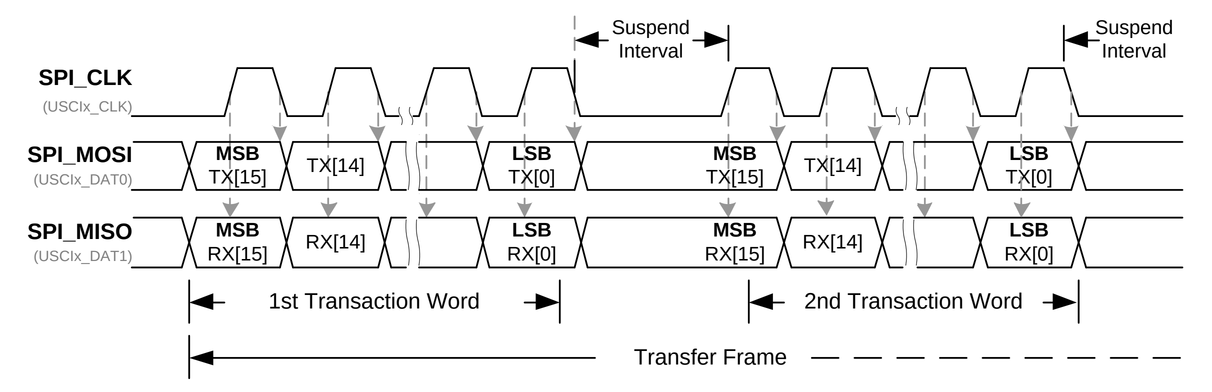


Note: x = 0

Figure 6.14-1016-bit data Length in One Word Transaction with MSB First Format

6.14.5.6 Word Suspend

SUSPITV (USPI_PROTCTL[11:8]) provide a configurable suspend interval, 0.5 ~ 15.5 SPI clock periods, between two successive transaction words in master mode. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SUSPITV (USPI_PROTCTL[11:8]) is 0x3 (3.5 SPI clock cycles).



Note 1: x = 0

Note 2: Timing Condition is SCLKMODE[1:0] = 0x0; LSB = 0; DWIDTH[3:0] = 0x0;

Figure 6.14-11 Word Suspend Interval between Two Transaction Words

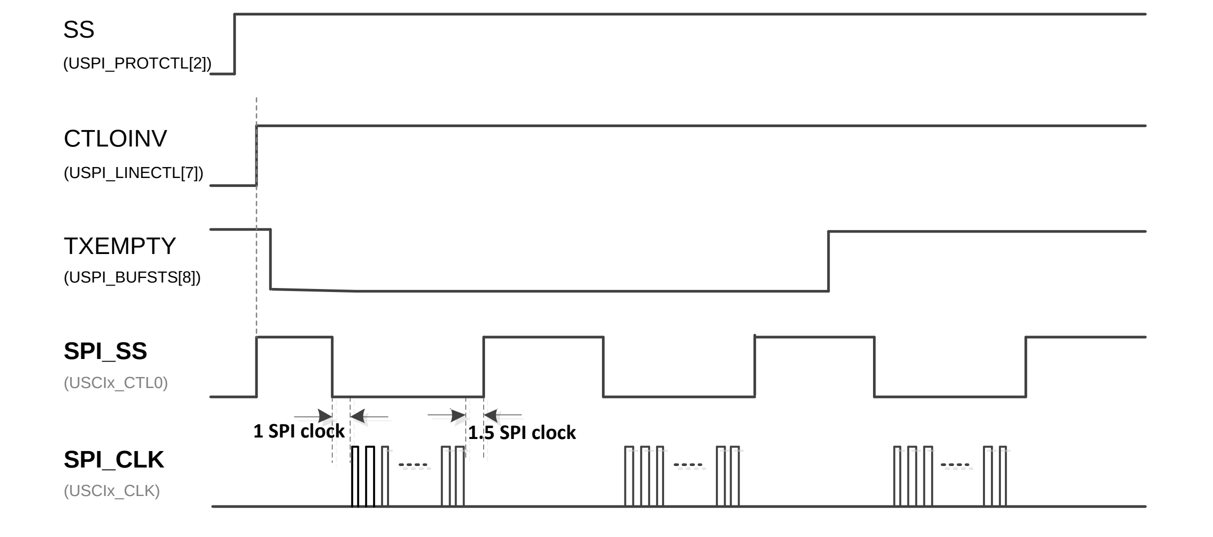
6.14.5.7 Automatic Slave Select Function

AUTOSS (USPI_PROTCTL[3]) is used for SPI master mode to enable the automatic slave select function. If the bit AUTOSS (USPI_PROTCTL[3]) is set, the slave select signal will be generated automatically and the setting value of SS (USPI_PROTCTL[2]) will not affect the output slave select (through USCIx_CTL0 line). This means that the slave select signal will be asserted by the USCI controller when the SPI data transfer is started by writing to the transmit buffer. Also, it will be de-asserted after either all transaction is finished or one word transaction done if the value of SUSPITV (USPI_PROTCTL[11:8]) is equal to or great than 3.

If the AUTOSS bit (USPI_PROTCTL[3]) is cleared, the slave select on USCIx_CTL0 pin will be asserted/de-asserted by setting/clearing the SS (USPI_PROTCTL[2]). The internal slave select signal is active high and the CTLOINV (USPI_LINECTL[7]) can be used for the inversion of the slave select signal.

In SPI master mode, if the value of SUSPITV (USPI_PROTCTL[11:8]) is less than 3 and the AUTOSS (USPI_PROTCTL[3]) is set as 1, the slave select signal will be kept at active state between two successive word transactions.

In SPI slave mode, to recognize the inactive state of the slave select signal, the inactive period of the received slave select signal must be larger than 2 peripheral clock cycles between two successive transactions.



Note: Automatic slave select is enabled

Figure 6.14-12 Auto Slave Select (SUSPITV ≧ 0x3)

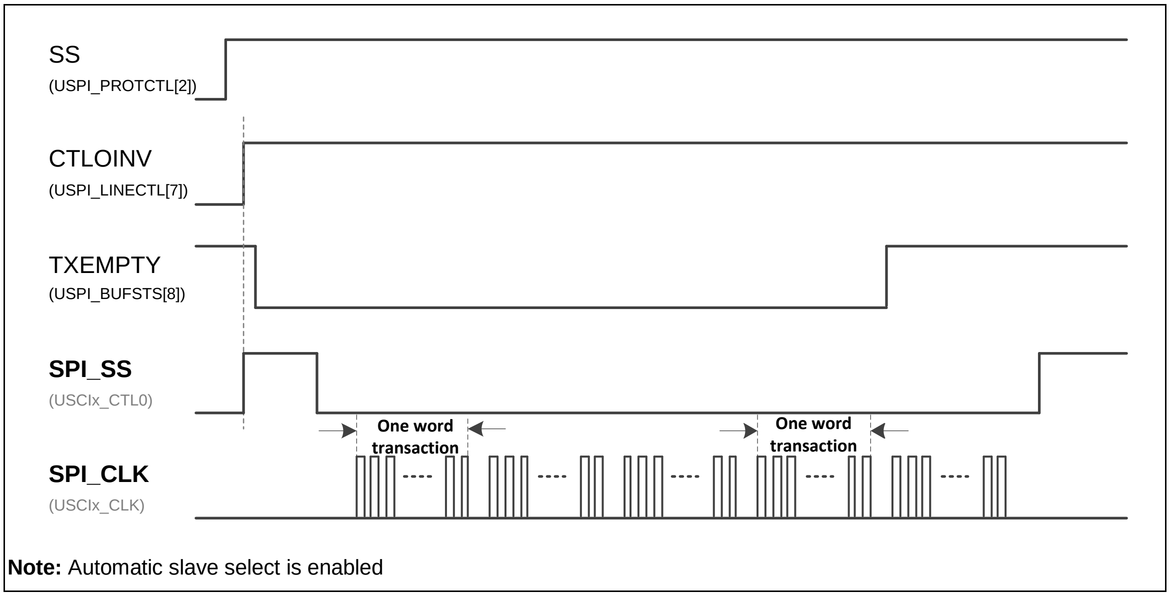


Figure 6.14-13 Auto Slave Select (SUSPITV < 0x3)

6.14.5.8 Slave 3-wire Mode

When the SLV3WIRE (USPI_PROTCTL[1]) is set by software to enable the slave 3-wire mode, the USCI SPI communication can work with no slave select signal in slave mode. The SLV3WIRE (USPI_PROTCTL[1]) only takes effect in SPI Slave mode. Only three pins, SPI_CLK (through USCIx_CLK line), SPI_MOSI (through USCIx_DAT0 line), and SPI_MISO (through USCIx_DAT1 line), are required to communicate with a SPI master. When the SLV3WIRE (USPI_PROTCTL[1]) is set to 1, the SPI slave will be ready to transmit/receive data after the SPI protocol is enabled by setting FUNMODE(USPI_CTL [2:0]) to 0x1.

6.14.5.9 Data Transfer Mode

USCI controller supports full-duplex SPI transfer and one data channel half-duplex SPI transfer.

Full-duplex SPI Transfer

In full-duplex SPI transfer, there are two data pins. One is used for transmitting data and the other is used for receiving data. Thus, data transmission and data reception can be performed simultaneously.

SCLKMODE (USPI_PROTCTL[7:6]) defines the transition timing of the data shift output signal on USCIx_DAT0 pin. The transition may happen at the corresponding edge of SPI bus clock or active edge of slave select signal. The level of the last data bit of a data word is held on USCIx_DAT0 pin until the next data word begins with the next corresponding edge of the serial bus clock.

One Data Channel Half-duplex SPI Transfer

In one data channel half-duplex SPI transfer, there is only one data pin for data transfer. Thus, the data transmission and data reception are at different time interval. The data shift direction is determined by PORTDIR (USPI_TXDAT[16]). If PORTDIR is set to 0, the data pin is configured as output mode; if PORTDIR is set to 1, the data pin is configured as input mode.

The function of one data channel half-duplex SPI transfer is similar to the full-duplex SPI protocol. All the transfer data timing is the same as the full-duplex SPI transfer.

Figure 6.14-16 are the one output data channel and one input data channel half-duplex transfer diagrams with the external device.

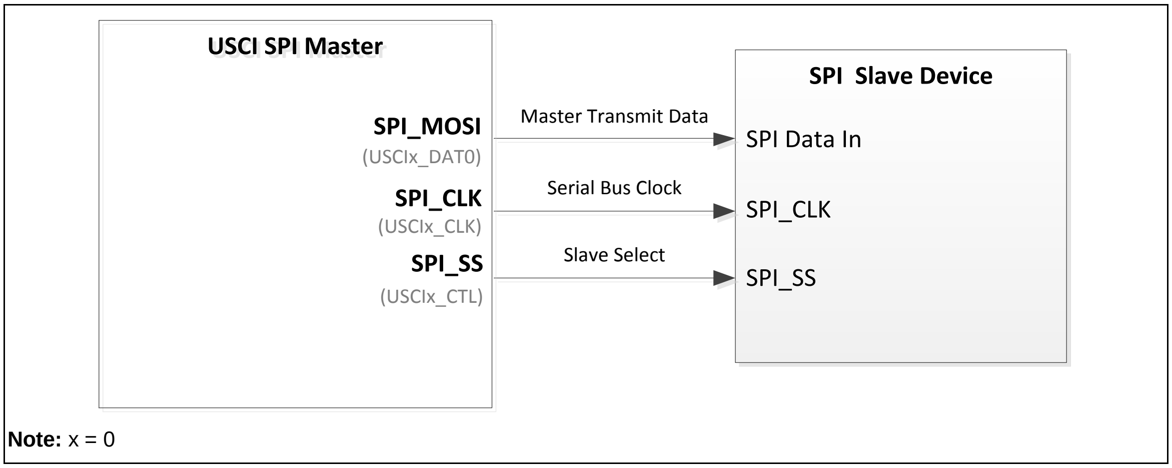


Figure 6.14-14 One Output Data Channel Half-duplex (SPI Master Mode)

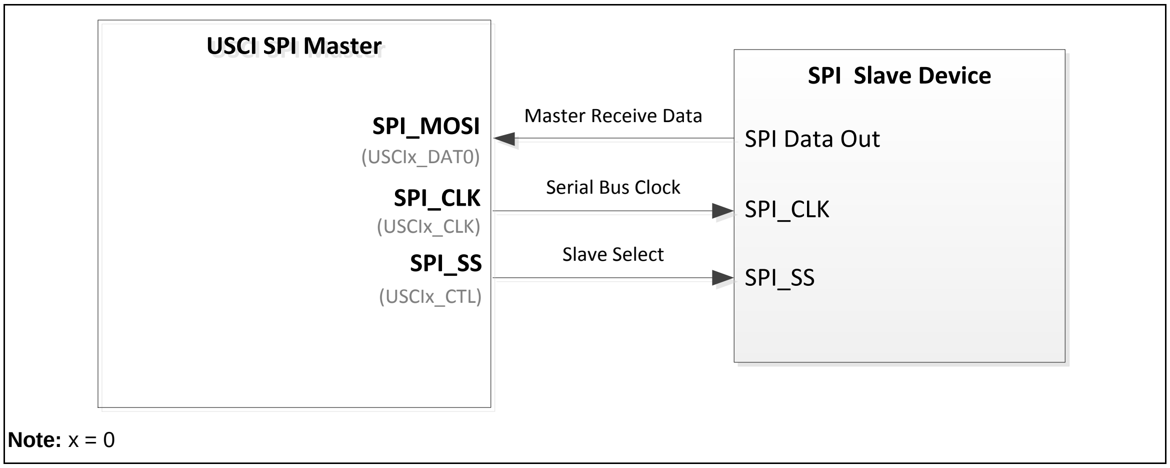


Figure 6.14-15 One Input Data Channel Half-duplex (SPI Master Mode)

The one data channel half-duplex transfer mode can be configured by TSMSEL[2:0] (USPI_PROTCTL[14:12]) and PORTDIR (USPI_TXDAT[16]) settings. When TSMSEL (USPI_PROTCTL[14:12]) is set to 0x4, one data channel half-duplex transfer mode is selected. The PORTDIR (USPI_TXDAT[16]) is used to define the direction of the corresponding transmit data. When the PORTDIR bit is set to 0, the USCI controller will send the corresponding data to external SPI device. When the PORTDIR bit is set to 1, the controller will read the corresponding data from the external SPI device.

For example, in one data channel half-duplex transfer mode with PORTDIR=0, USCI SPI transmits data through USCIx_DAT0 pin; if PORTDIR=1, USCI SPI receives data through USCIx_DAT0 pin.

6.14.5.10 Interrupt

Data Transfer Interrupts

Transmit Start Interrupt

The interrupt event TXSTIF (USPI_PROTSTS[1]) is set after the start of the first data bit of a transmit data word. It can be cleared only by writing 1 to it. The controller wil issue an interrupt if TXSTIEN (USPI_INTEN[1]) is also set to 1.

Transmit End Interrupt

The interrupt event TXENDIF (USPI_PROTSTS[2]) is set after the start of the last data bit of the last transmit data which has been stored in transmit buffer. It can be cleared only by writing 1 to it. The controller wil issue an interrupt if TENDIEN (USPI_INTEN[2]) is also set to 1.

Receive Start Interrupt

The interrupt event RXSTIF (USPI_PROTSTS[3]) is set after the start of the first data bit of a receive data word. It can be cleared only by writing 1 to it. The controller wil issue an interrupt if RXSTIEN (USPI_INTEN[3]) is also set to 1.

Receive End Interrupt

The interrupt event RXENDIF (USPI_PROTSTS[4]) is set after the start of the last data bit of a receive data word. It can be cleared only by writing 1 to it. The controller wil issue an interrupt if RXENDIEN (USPI_INTEN[4]) is also set to 1.

Protocol-related Interrupts

SPI Slave Select Interrupt

In SPI slave mode, there are slave select active and in-active interrupt flags, SSACTIF (USPI_PROTSTS[9]) and SSINAIF (USPI_PROTSTS[8]), will be set to 1 when SLAVE (USPI_PROTCTL [0]) is set to 1 and slave senses the slave select signal active or inactive. The SPI controller will issue an interrupt if SSINAIEN (USPI_PROTIEN[0]) or SSACTIEN (USPI_PROTIEN[1]) are set to 1. Because the internal slave select signal in SPI function is active high, the ININV (USPI_CTLIN0[2]) can be used for inverting the slave select signal comes from an active low device.

Slave Time-out Interrupt

In SPI slave mode, there is slave time-out function for user to know that there is no serial clock input during the period of one word transaction. The slave time-out function uses the timing measurement counter for the calculation of slave time-out period which is defined by SLVTOCNT (USPI_PROTCTL[25:16]). TMCNTSRC (USPI_BRGEN[5]) can be used for clock frequency selection of timing measurement counter to calculate the Slave time-out period.

When the timing measurement counter is enabled by TMCNTEN (USPI_BRGEN[4]) and the setting value of SLVTOCNT (USPI_PROTCTL[25:16]) is not 0 in SPI slave mode, the timing measurement counter will start counting after the first input serial clock of each received word data. This counter will be reset while receiving the following input serial clock and then keep counting. Finally, the timing measurement counter will be cleared and stopped after the finish of the current word transaction. If the value of the time-out counter is equal to or greater than the value of SLVTOCNT (USPI_PROTCTL[25:16]) before one word transaction is done, the Slave time-out interrupt event occurs and the SLVTOIF (USPI_PROTSTS[5]) will be set to 1. The controller wil issue an interrupt if SLVTOIEN (USPI_ PROTIEN[2]) is also set to 1.

Buffer-related Interrupts

The buffer-related interrupts are available if there is transmit/receive buffer in USCI controller.

Receive Buffer Overrun Interrupt

If there is receive buffer overrun event, RXOVIF (USPI_BUFSTS[3]) will be set as 1. It can be cleared by write 1 into it. The controller wil issue an interrupt if RXOVIEN (USPI_ BUFCTL[14]) is also set to 1.

Transmit Buffer Under-run Interrupt

If there is transmit buffer under-run event, TXUDRIF (USPI_BUFSTS[11]) will be set as 1. It can be cleared by write 1 into it. The controller wil issue an interrupt if TXUDRIEN (USPI_ BUFCTL[6]) is also set to 1.

6.14.5.11 Timing Diagram

The slave select signal of USCI SPI protocol is active high by default, and it can be inverted by CTLOINV (USPI_LINECTL[7]) setting.

The idle state of serial bus clock and the serial bus clock edge used for transmit/receive data can be configured by setting SCLKMODE (USPI_PROTCTL[7:6]). The bit length of a transaction word data is determined by DWIDTH (USPI_LINECTL[11:8]), and data bit transfer sequence is determined by LSB (USPI_LINECTL[0]). Four SPI timing diagrams for master/slave operations and the related settings are shown in Figure 6.14-16, Figure 6.14-17, Figure 6.14-18 and Figure 6.14-19.

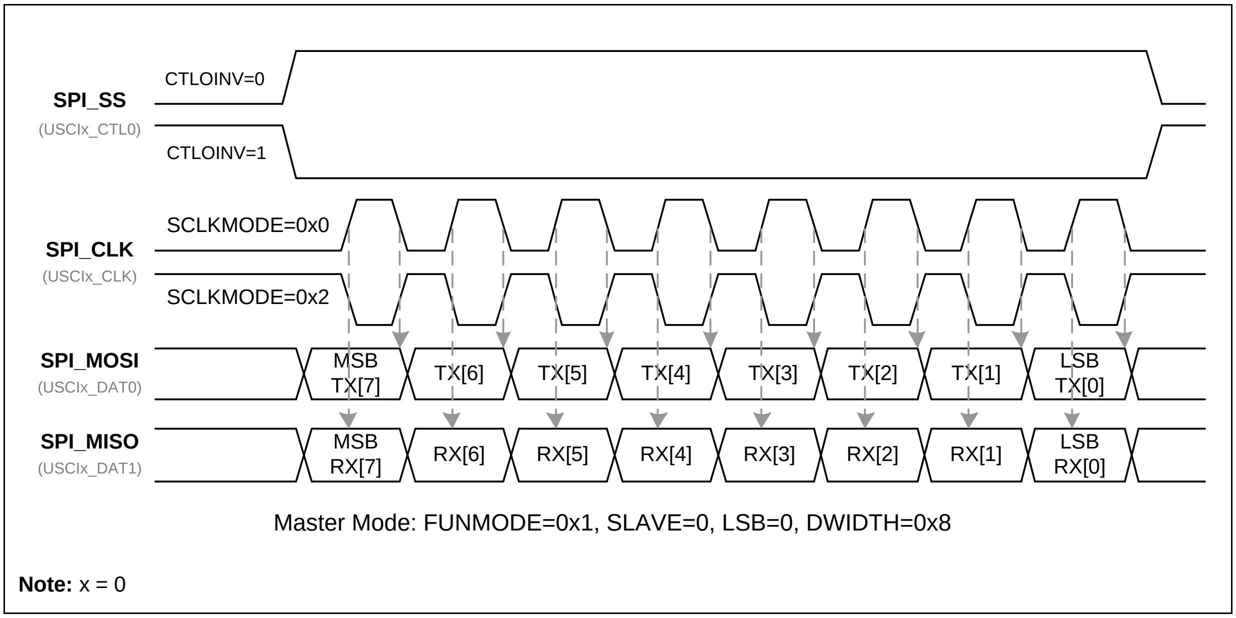


Figure 6.14-16 SPI Timing in Master Mode

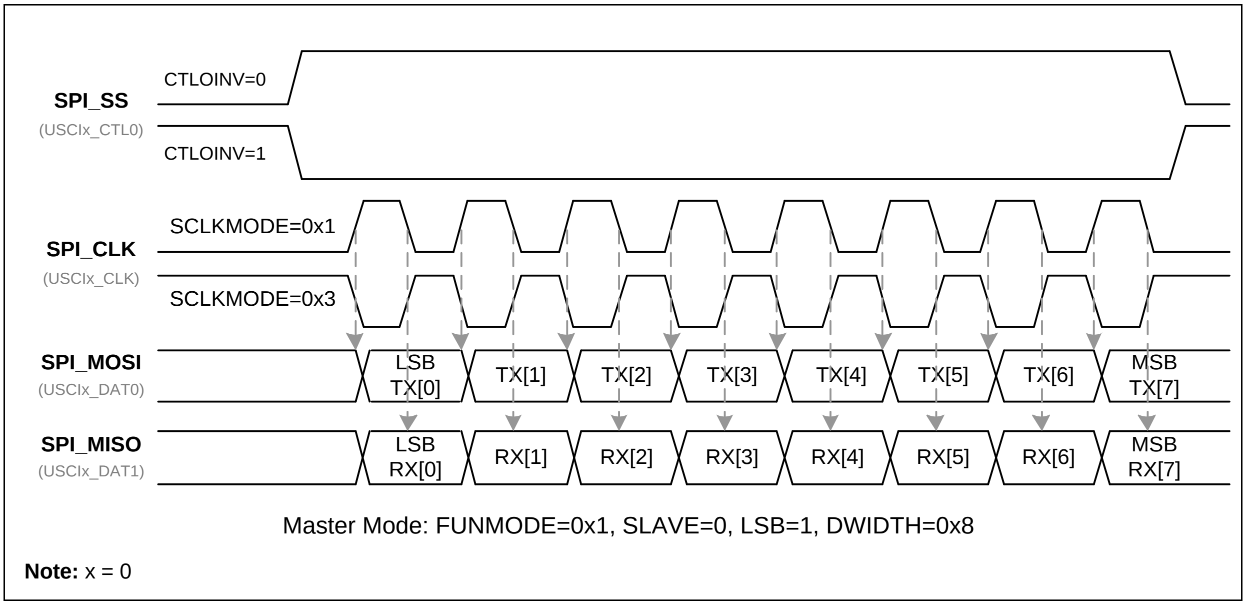


Figure 6.14-17 SPI Timing in Master Mode (Alternate Phase of Serial Bus Clock)

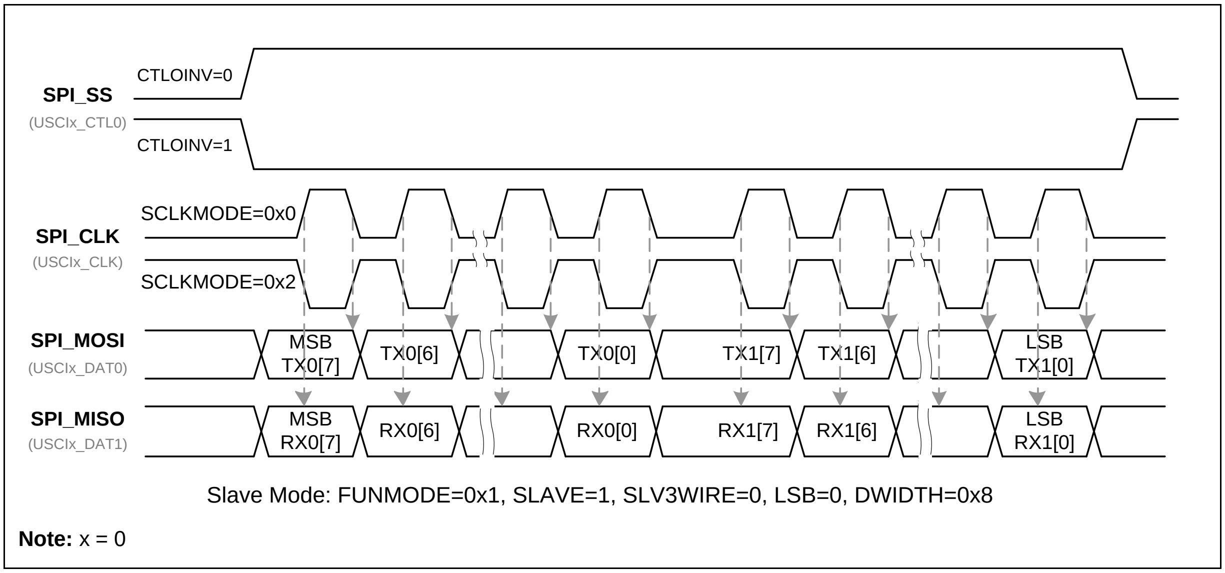


Figure 6.14-18 SPI Timing in Slave Mode

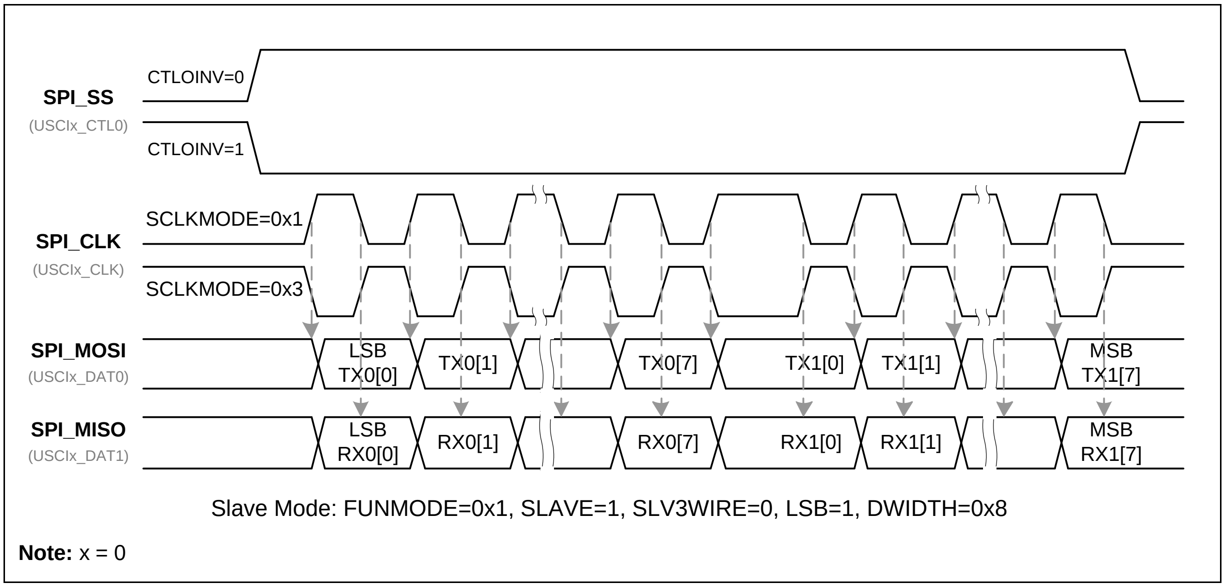


Figure 6.14-19 SPI Timing in Slave Mode (Alternate Phase of Serial Bus Clock)

6.14.5.12 Programming flow

This section describes the programming flow for USCI SPI data transfer.

For Master Mode:

1. Enable USCI peripheral clock by setting CLK_APBCLK1 register.
2. Configure user-specified pins as USCI function pins by setting corresponding multiple function control registers.
3. Set FUNMODE (USPI_CTL[2:0]) to 0x1 to select SPI mode.
4. Set USPI_BRGEN register to determine the SPI bus clock frequency.
5. According to the requirements of user’s application, configured the settings as follows.
   - CTLOINV (USPI_LINECTL[7]): If the slave selection signal is active low, set this bit to 1; otherwise, set it to 0.
   - DWIDTH (USPI_LINECTL[11:8]): Data width setting.
   - LSB (USPI_LINECTL[0]): LSB first or MSB first.
   - TSMSEL (USPI_PROTCTL[14:12]): Full-duplex SPI transfer or one channel half-duplex SPI transfer.
   - SCLKMODE (USPI_PROTCTL[7:6]): Determine the clock timing.
   - AUTOSS (USPI_PROTCTL[3]): Enable automatic slave select function or not.
   - SLAVE (USPI_PROTCTL[0]): Set to 0 for master mode.
6. Set PROTEN (USPI_PROTCTL[31]) to 1 to enable SPI protocol.
7. If automatic slave select function is disabled (AUTOSS=0), set SS (USPI_PROTCTL[2]) to 1 before data transfer; set SS to 0 to inactivate the slave selection signal by user’s application.
8. Write USPI_TXDAT register to trigger SPI transfer. In half-duplex SPI transfer, the data pin direction is determined by PORTDIR (USPI_TXDAT[16]) setting.
9. User can get the received data by reading USPI_RXDAT register as long as RXEMPTY (USPI_BUFSTS[0]) is 0. The SPI data transfer can be triggered by writing USPI_TXDAT register

as long as TXFULL (USPI_BUFSTS[9]) is 0.

For Slave Mode:

1. Enable USCI peripheral clock by setting CLK_APBCLK1 register.
2. Configure user-specified pins as USCI function pins by setting corresponding multiple function control registers.
3. Set FUNMODE (USPI_CTL[2:0]) to 0x1 to select SPI mode.
4. According to the requirements of user’s application, configured the settings as follows.
   - ININV (USPI_CTLIN0[2]): If the slave selection signal is active low, set this bit to 1; otherwise, set it to 0.
   - DWIDTH (USPI_LINECTL[11:8]): Data width setting.
   - LSB (USPI_LINECTL[0]): LSB first or MSB first.
   - TSMSEL (USPI_PROTCTL[14:12]): Full-duplex SPI transfer or one channel half-duplex SPI transfer.
   - SCLKMODE (USPI_PROTCTL[7:6]): Determine the clock timing.
   - SLAVE (USPI_PROTCTL[0]): Set to 1 for slave mode.
5. Set PROTEN (USPI_PROTCTL[31]) to 1 to enable SPI protocol.
6. Write USPI_TXDAT register for transmission. In half-duplex SPI transfer, the data pin direction is determined by PORTDIR (USPI_TXDAT[16]) setting.
7. User can get the received data by reading USPI_RXDAT register as long as RXEMPTY (USPI_BUFSTS[0]) is 0. The next datum for transmission can be written to USPI_TXDAT register as long as TXFULL (USPI_BUFSTS[9]) is 0.

6.14.5.13 Wake-up Function

The USCI Controller in SPI mode supports wake-up system function. The wake-up source in SPI protocol is the transition of input slave select signal in Slave mode.

6.14.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
USPI Base Address: **USPIn_BA = 0x4007_0000 + (0x10_0000 * n)** **n=0**				
USPI_CTL **n=0**	USPIn_BA+0x00	R/W	USCI Control Register	0x0000_0000
USPI_INTEN **n=0**	USPIn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000
USPI_BRGEN **n=0**	USPIn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
USPI_DATIN0 **n=0**	USPIn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000
USPI_CTLIN0 **n=0**	USPIn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000
USPI_CLKIN **n=0**	USPIn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000
USPI_LINECTL **n=0**	USPIn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
USPI_TXDAT **n=0**	USPIn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
USPI_RXDAT **n=0**	USPIn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
USPI_BUFCTL **n=0**	USPIn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000
USPI_BUFSTS **n=0**	USPIn_BA+0x3C	R/W	USCI Transmit/Receive Buffer Status Register	0x0000_0101
USPI_PDMACTL **n=0**	USPIn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000
USPI_WKCTL **n=0**	USPIn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
USPI_WKSTS **n=0**	USPIn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
USPI_PROTCTL **n=0**	USPIn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0300
USPI_PROTIEN **n=0**	USPIn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
USPI_PROTSTS **n=0**	USPIn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

6.14.7 Register Description

USCI Control Register (USPI_CTL)

Register	Offset	R/W	Description	Reset Value
USPI_CTL **n=0**	USPIn_BA+0x00	R/W	USCI 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					**FUNMODE**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 0x0 to FUNMODE. 0x0 = The USCI is disabled. All protocol related state machines are set to idle state. 0x1 = The SPI protocol is selected. 0x2 = The UART protocol is selected. 0x4 = The I^2C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Interrupt Enable Register (USPI_INTEN)

Register	Offset	R/W	Description	Reset Value
USPI_INTEN **n=0**	USPIn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**RXENDIEN**	**Receive End Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive finish event. 0 = The receive end interrupt Disabled. 1 = The receive end interrupt Enabled.
[3]	**RXSTIEN**	**Receive Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive start event. 0 = The receive start interrupt Disabled. 1 = The receive start interrupt Enabled.
[2]	**TXENDIEN**	**Transmit End Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit finish event. 0 = The transmit finish interrupt Disabled. 1 = The transmit finish interrupt Enabled.
[1]	**TXSTIEN**	**Transmit Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit start event. 0 = The transmit start interrupt Disabled. 1 = The transmit start interrupt Enabled.
[0]	**Reserved**	Reserved.

USCI Baud Rate Generator Register (USPI_BRGEN)

Register	Offset	R/W	Description	Reset Value
USPI_BRGEN **n=0**	USPIn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

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

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} ($f_{DIV\_CLK} = f_{PROT\_CLK}$ / (CLKDIV+1)).
[15:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Time Measurement Counter Clock Source Selection** 0 = Time measurement counter with f_{PROT_CLK}. 1 = Time measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Time Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Time measurement counter Disabled. 1 = Time measurement counter Enabled.
[3:2]	**SPCLKSEL**	**Sample Clock Source Selection** This bit field used for the clock source selection of sample clock (f_{SAMP_CLK}) for the protocol processor. 00 = f_{DIV_CLK}. 01 = f_{PROT_CLK}. 10 = f_{SCLK}. 11 = f_{REF_CLK}.
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source of reference clock (f_{REF_CLK}).

		0 = Peripheral device clock $f_{PCLK.}$ 1 = Reserved.

USCI Input Data Signal Configuration (USPI_DATIN0)

Register	Offset	R/W	Description	Reset Value
USPI_DATIN0 **n=0**	USPIn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	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					ININV	Reserved	SYNCSEL

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted. **Note:** In SPI protocol, we suggest this bit should be set as 0.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Signal Synchronization Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, we suggest this bit should be set as 0.

USCI Input Control Signal Configuration (USPI_CTLIN0)

Register	Offset	R/W	Description	Reset Value
USPI_CTLIN0 **n=0**	USPIn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, we suggest this bit should be set as 0.

USCI Input Clock Signal Configuration (USPI_CLKIN)

Register	Offset	R/W	Description	Reset Value
USPI_CLKIN **n=0**	USPIn_BA+0x28	R/W	USCI Input Clock Signal 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SYNCSEL**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, we suggest this bit should be set as 0.

USCI Line Control Register (USPI_LINECTL)

Register	Offset	R/W	Description	Reset Value
USPI_LINECTL **n=0**	USPIn_BA+0x2C	R/W	USCI 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				DWIDTH			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTLOINV	Reserved	DATOINV	Reserved				LSB

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0x0: The data word contains 16 bits located at bit positions [15:0]. 0x1: Reserved. 0x2: Reserved. 0x3: Reserved. 0x4: The data word contains 4 bits located at bit positions [3:0]. 0x5: The data word contains 5 bits located at bit positions [4:0]. ... 0xF: The data word contains 15 bits located at bit positions [14:0].
[7]	**CTLOINV**	**Control Signal Output Inverse Selection** This bit defines the relation between the internal control signal and the output control signal. 0 = No effect. 1 = The control signal will be inverted before its output. **Note:** The control signal has different definitions in different protocol. In SPI protocol, the control signal means slave select signal.
[6]	**Reserved**	Reserved.
[5]	**DATOINV**	**Data Output Inverse Selection** This bit defines the relation between the internal shift data value and the output data signal of USCIx_DAT0/1 pin. 0 = Data output level is not inverted. 1 = Data output level is inverted.
[4:1]	**Reserved**	Reserved.

[0]	**LSB**	**LSB First Transmission Selection** 0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (USPI_TXDAT)

Register	Offset	R/W	Description	Reset Value
USPI_TXDAT **n=0**	USPIn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**PORTDIR**	**Port Direction Control** This bit field is only available while USCI operates in SPI protocol (FUNMODE = 0x1) with half-duplex transfer. It is used to define the direction of the data port pin. When software writes USPI_TXDAT register, the transmit data and its port direction are settled simultaneously. 0 = The data pin is configured as output mode. 1 = The data pin is configured as input mode.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission. In order to avoid overwriting the transmit data, user have to check TXEMPTY (USPI_BUFSTS[8]) status before writing transmit data into this bit field.

USCI Receive Data Register (USPI_RXDAT)

Register	Offset	R/W	Description	Reset Value
USPI_RXDAT **n=0**	USPIn_BA+0x34	R	USCI 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**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer.

USCI Transmitter/Receive Buffer Control Register (USPI_BUFCTL)

Register	Offset	R/W	Description	Reset Value
USPI_BUFCTL **n=0**	USPIn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset the receive-related counters, state machine, and the content of receive shift register and data buffer. **Note:** It is cleared automatically after one PCLK cycle.
[16]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset the transmit-related counters, state machine, and the content of transmit shift register and data buffer**.** **Note1:** It is cleared automatically after one PCLK cycle. **Note2:** Write 1 to this bit will set the output data pin to zero if USPI_LINCTL[5]=0.
[15]	**RXCLR**	**Clear Receive Buffer** 0 = No effect. 1 = The receive buffer is cleared. Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[14]	**RXOVIEN**	**Receive Buffer Overrun Interrupt Enable Control** 0 = Receive overrun interrupt Disabled. 1 = Receive overrun interrupt Enabled.
[13:8]	**Reserved**	Reserved.
[7]	**TXCLR**	**Clear Transmit Buffer** 0 = No effect. 1 = The transmit buffer is cleared. Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.

[6]	**TXUDRIEN**	**Slave Transmit Under Run Interrupt Enable Bit** 0 = Transmit under-run interrupt Disabled. 1 = Transmit under-run interrupt Enabled.
[5:0]	**Reserved**	Reserved.

USCI Transmit/Receive Buffer Status Register (USPI_BUFSTS)

Register	Offset	R/W	Description	Reset Value
USPI_BUFSTS **n=0**	USPIn_BA+0x3C	R/W	USCI Transmit/Receive Buffer Status Register	0x0000_0101

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**TXUDRIF**	**Transmit Buffer Under-run Interrupt Status** This bit indicates that a transmit buffer under-run event has been detected. If enabled by TXUDRIEN (USPI_BUFCTL[6]), the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit 0 = A transmit buffer under-run event has not been detected. 1 = A transmit buffer under-run event has been detected.
[10]	**Reserved**	Reserved.
[9]	**TXFULL**	**Transmit Buffer Full Indicator** 0 = Transmit buffer is not full. 1 = Transmit buffer is full.
[8]	**TXEMPTY**	**Transmit Buffer Empty Indicator** 0 = Transmit buffer is not empty. 1 = Transmit buffer is empty and available for the next transmission datum.
[7:4]	**Reserved**	Reserved.
[3]	**RXOVIF**	**Receive Buffer Over-run Interrupt Status** This bit indicates that a receive buffer overrun event has been detected. If RXOVIEN (USPI_BUFCTL[14]) is enabled, the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit. 0 = A receive buffer overrun event has not been detected. 1 = A receive buffer overrun event has been detected.
[2]	**Reserved**	Reserved.
[1]	**RXFULL**	**Receive Buffer Full Indicator** 0 = Receive buffer is not full. 1 = Receive buffer is full.

[0]	**RXEMPTY**	**Receive Buffer Empty Indicator** 0 = Receive buffer is not empty. 1 = Receive buffer is empty.

USCI PDMA Control Register (USPI_PDMACTL)

Register	Offset	R/W	Description	Reset Value
USPI_PDMACTL **n=0**	USPIn_BA+0x40	R/W	USCI 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				**PDMAEN**	**RXPDMAEN**	**TXPDMAEN**	**PDMARST**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PDMAEN**	**PDMA Mode Enable Bit** 0 = PDMA function Disabled. 1 = PDMA function Enabled.
[2]	**RXPDMAEN**	**PDMA Receive Channel Available** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[1]	**TXPDMAEN**	**PDMA Transmit Channel Available** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled.
[0]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the USCI’s PDMA control logic. This bit will be cleared to 0 automatically.

USCI Wake-up Control Register (USPI_WKCTL)

Register	Offset	R/W	Description	Reset Value
USPI_WKCTL **n=0**	USPIn_BA+0x54	R/W	USCI 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					**PDBOPT**	**Reserved**	**WKEN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDBOPT**	**Power Down Blocking Option** 0 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, MCU will stop the transfer and enter Power-down mode immediately. 1 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, the on-going transfer will not be stopped and MCU will enter idle mode immediately.
[1]	**Reserved**	Reserved.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (USPI_WKSTS)

Register	Offset	R/W	Description	Reset Value
USPI_WKSTS **n=0**	USPIn_BA+0x58	R/W	USCI 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							**WKF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – USPI_PROTCTL (SPI)

Register	Offset	R/W	Description	Reset Value
USPI_PROTCTL **n=0**	USPIn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0300

31	30	29	28	27	26	25	24
PROTEN	Reserved		TXUDRPOL	Reserved		SLVTOCNT	
23	22	21	20	19	18	17	16
SLVTOCNT							
15	14	13	12	11	10	9	8
Reserved	TSMSEL			SUSPITV			
7	6	5	4	3	2	1	0
SCLKMODE		Reserved		AUTOSS	SS	SLV3WIRE	SLAVE

Bits	Description	
[31]	**PROTEN**	**SPI Protocol Enable Bit** 0 = SPI Protocol Disabled. 1 = SPI Protocol Enabled.
[30:29]	**Reserved**	Reserved.
[28]	**TXUDRPOL**	**Transmit Under-run Data Polarity (for Slave)** This bit defines the transmitting data level when no data is available for transferring. 0 = The output data level is 0 if TX under run event occurs. 1 = The output data level is 1 if TX under run event occurs.
[27:26]	**Reserved**	Reserved.
[25:16]	**SLVTOCNT**	**Slave Mode Time-out Period (Slave Only)** In Slave mode, this bit field is used for Slave time-out period. This bit field indicates how many clock periods (selected by TMCNTSRC, USPI_BRGEN[5]) between the two edges of input SCLK will assert the Slave time-out event. Writing 0x0 into this bit field will disable the Slave time-out function. Example: Assume SLVTOCNT is 0x0A and TMCNTSRC (USPI_BRGEN[5]) is 1, it means the time-out event will occur if the state of SPI bus clock pin is not changed more than (10+1) periods of f_{DIV_CLK}.
[15]	**Reserved**	Reserved.
[14:12]	**TSMSEL**	**Transmit Data Mode Selection** This bit field describes how receive and transmit data is shifted in and out. 0x0 = Full-duplex SPI. 0x4 = Half-duplex SPI. Other values are reserved. **Note:** Changing the value of this bit field will produce the TXRST and RXRST to clear the TX/RX data buffer automatically.
[11:8]	**SUSPITV**	**Suspend Interval (Master Only)** This bit field provides the 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.
		(SUSPITV[3:0] + 0.5) * period of SPI_CLK clock cycle
		Example:
		SUSPITV = 0x0 … 0.5 SPI_CLK clock cycle.
		SUSPITV = 0x1 … 1.5 SPI_CLK clock cycle.
		……
		SUSPITV = 0xE … 14.5 SPI_CLK clock cycle.
		SUSPITV = 0xF … 15.5 SPI_CLK clock cycle.

[7:6]	**SCLKMODE**	**Serial Bus Clock Mode** This bit field defines the SCLK idle status, data transmit, and data receive edge. MODE0 = The idle state of SPI clock is low level. Data is transmitted with falling edge and received with rising edge. MODE1 = The idle state of SPI clock is low level. Data is transmitted with rising edge and received with falling edge. MODE2 = The idle state of SPI clock is high level. Data is transmitted with rising edge and received with falling edge. MODE3 = The idle state of SPI clock is high level. Data is transmitted with falling edge and received with rising edge.
[5:4]	**Reserved**	Reserved.
[3]	**AUTOSS**	**Automatic Slave Select Function Enable (Master Only)** 0 = Slave select signal will be controlled by the setting value of SS (USPI_PROTCTL[2]) bit. 1 = Slave select signal will be generated automatically. The slave select signal will be asserted by the SPI controller when transmit/receive is started, and will be de-asserted after each transmit/receive is finished.
[2]	**SS**	**Slave Select Control (Master Only)** If AUTOSS bit is cleared, setting this bit to 1 will set the slave select signal to active state, and setting this bit to 0 will set the slave select back to inactive state. If the AUTOSS function is enabled (AUTOSS = 1), the setting value of this bit will not affect the current state of slave select signal. **Note:** In SPI protocol, the internal slave select signal is active high.
[1]	**SLV3WIRE**	**Slave 3-wire Mode Selection (Slave Only)** The SPI protocol can work with 3-wire interface (without slave select signal) in Slave mode. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface.
[0]	**SLAVE**	**Slave Mode Selection** 0 = Master mode. 1 = Slave mode.

USCI Protocol Interrupt Enable Register – USPI_PROTIEN (SPI)

Register	Offset	R/W	Description	Reset Value
USPI_PROTIEN **n=0**	USPIn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:5]	**Reserved**	Reserved.
[3]	**SLVBEIEN**	**Slave Mode Bit Count Error Interrupt Enable Control** If data transfer is terminated by slave time-out or slave select inactive event in Slave mode, so that the transmit/receive data bit count does not match the setting of DWIDTH (USPI_LINECTL[11:8]). Bit count error event occurs. 0 = The Slave mode bit count error interrupt Disabled. 1 = The Slave mode bit count error interrupt Enabled.
[2]	**SLVTOIEN**	**Slave Time-out Interrupt Enable Control** In SPI protocol, this bit enables the interrupt generation in case of a Slave time-out event. 0 = The Slave time-out interrupt Disabled. 1 = The Slave time-out interrupt Enabled.
[1]	**SSACTIEN**	**Slave Select Active Interrupt Enable Control** This bit enables/disables the generation of a slave select interrupt if the slave select changes to active. 0 = Slave select active interrupt generation Disabled. 1 = Slave select active interrupt generation Enabled.
[0]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable Control** This bit enables/disables the generation of a slave select interrupt if the slave select changes to inactive. 0 = Slave select inactive interrupt generation Disabled. 1 = Slave select inactive interrupt generation Enabled.

USCI Protocol Status Register – USPI_PROTSTS (SPI)

Register	Offset	R/W	Description	Reset Value
USPI_PROTSTS n=0	USPIn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					SLVUDR	BUSY	SSLINE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						SSACTIF	SSINAIF
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	SLVBEIF	SLVTOIF	RXENDIF	RXSTIF	TXENDIF	TXSTIF	Reserved

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**SLVUDR**	**Slave Mode Transmit Under-run Status (Read Only)** In Slave mode, if there is no available transmit data in buffer while transmit data shift out caused by input serial bus clock, this status flag will be set to 1. This bit indicates whether the current shift-out data of word transmission is switched to TXUDRPOL (USPI_PROTCTL[28]) or not. 0 = Slave transmit under run event does not occur. 1 = Slave transmit under run event occurs.
[17]	**BUSY**	**Busy Status (Read Only)** 0 = SPI is in idle state. 1 = SPI is in busy state. The following lists the bus busy conditions: a. PROTEN (USPI_PROTCTL[31]) = 1 and the TXEMPTY = 0. b. For SPI Master mode, the TXEMPTY = 1 but the current transaction is not finished yet. c. For SPI Slave mode, the PROTEN (USPI_PROTCTL[31]) = 1 and there is serial clock input into the SPI core logic when slave select is active. d. For SPI Slave mode, the PROTEN (USPI_PROTCTL[31]) = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive.
[16]	**SSLINE**	**Slave Select Line Bus Status (Read Only)** This bit is only available in Slave mode. It used to monitor the current status of the input slave select signal on the bus. 0 = The slave select line status is 0. 1 = The slave select line status is 1.
[15:10]	**Reserved**	Reserved.
[9]	**SSACTIF**	**Slave Select Active Interrupt Flag (for Slave Only)** This bit indicates that the internal slave select signal has changed to active. It is cleared by software writes one to this bit 0 = The slave select signal has not changed to active.

		1 = The slave select signal has changed to active. **Note:** The internal slave select signal is active high.
[8]	**SSINAIF**	**Slave Select Inactive Interrupt Flag (for Slave Only)** This bit indicates that the internal slave select signal has changed to inactive. It is cleared by software writes 1 to this bit 0 = The slave select signal has not changed to inactive. 1 = The slave select signal has changed to inactive. **Note:** The internal slave select signal is active high.
[7]	**Reserved**	Reserved.
[6]	**SLVBEIF**	**Slave Bit Count Error Interrupt Flag (for Slave Only)** 0 = Slave bit count error event does not occur. 1 = Slave bit count error event occurs. **Note:** It is cleared by software writes 1 to this bit.
[5]	**SLVTOIF**	**Slave Time-out Interrupt Flag (for Slave Only)** 0 = Slave time-out event does not occur. 1 = Slave time-out event occurs. **Note:** It is cleared by software writes 1 to this bit
[4]	**RXENDIF**	**Receive End Interrupt Flag** 0 = Receive end event does not occur. 1 = Receive end event occurs. **Note:** It is cleared by software writes 1 to this bit
[3]	**RXSTIF**	**Receive Start Interrupt Flag** 0 = Receive start event does not occur. 1 = Receive start event occurs. **Note:** It is cleared by software writes 1 to this bit
[2]	**TXENDIF**	**Transmit End Interrupt Flag** 0 = Transmit end event does not occur. 1 = Transmit end event occurs. **Note:** It is cleared by software writes 1 to this bit
[1]	**TXSTIF**	**Transmit Start Interrupt Flag** 0 = Transmit start event does not occur. 1 = Transmit start event occurs. **Note:** It is cleared by software writes 1 to this bit
[0]	**Reserved**	Reserved.

6.15 USCI - I^{2}C Mode

6.15.1 Overview

On I^{2}C bus, data is transferred between a Master and a Slave. Data bits transfer on the SCL and SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit. There is one SCL clock pulse for each data bit with the MSB being transmitted first, and an acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (START or STOP). Please refer to Figure 6.15-1 for more detailed I^{2}C BUS Timing.

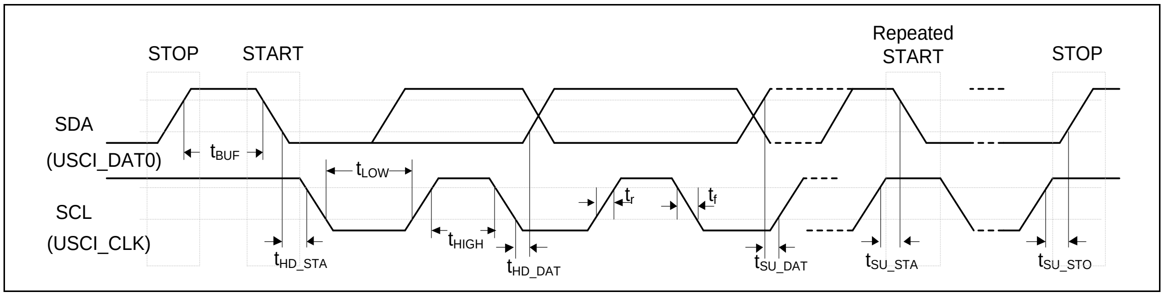


Figure 6.15-1 I^{2}C Bus Timing

The device's on-chip I^{2}C provides the serial interface that meets the I^{2}C bus standard mode specification. The I^{2}C port handles byte transfers autonomously. The I^{2}C mode is selected by FUNMODE (UI2C_CTL [2:0]) = 100b. When enable this port, the USCI interfaces to the I^{2}C bus via two pins: SDA and SCL. When I/O pins are used as I^{2}C ports, user must set the pins function to I^{2}C in advance.

Note: Pull-up resistor is needed for I^{2}C operation because the SDA and SCL are set to open-drain pins when USCI is selected to I^{2}C operation mode .

6.15.2 Features

- Full master and slave device capability
- Supports of 7-bit addressing, as well as 10-bit addressing
- Communication in standard mode (100 kBit/s) or in fast mode (up to 400 kBit/s)
- Supports multi-master bus
- Supports 10-bit bus time-out capability
- Supports bus monitor mode.
- Supports Power down wake-up by data toggle or address match
- Supports setup/hold time programmable
- Supports multiple address recognition (two slave address with mask option)

6.15.3 Block Diagram

The basic configurations of USCI are as follows:

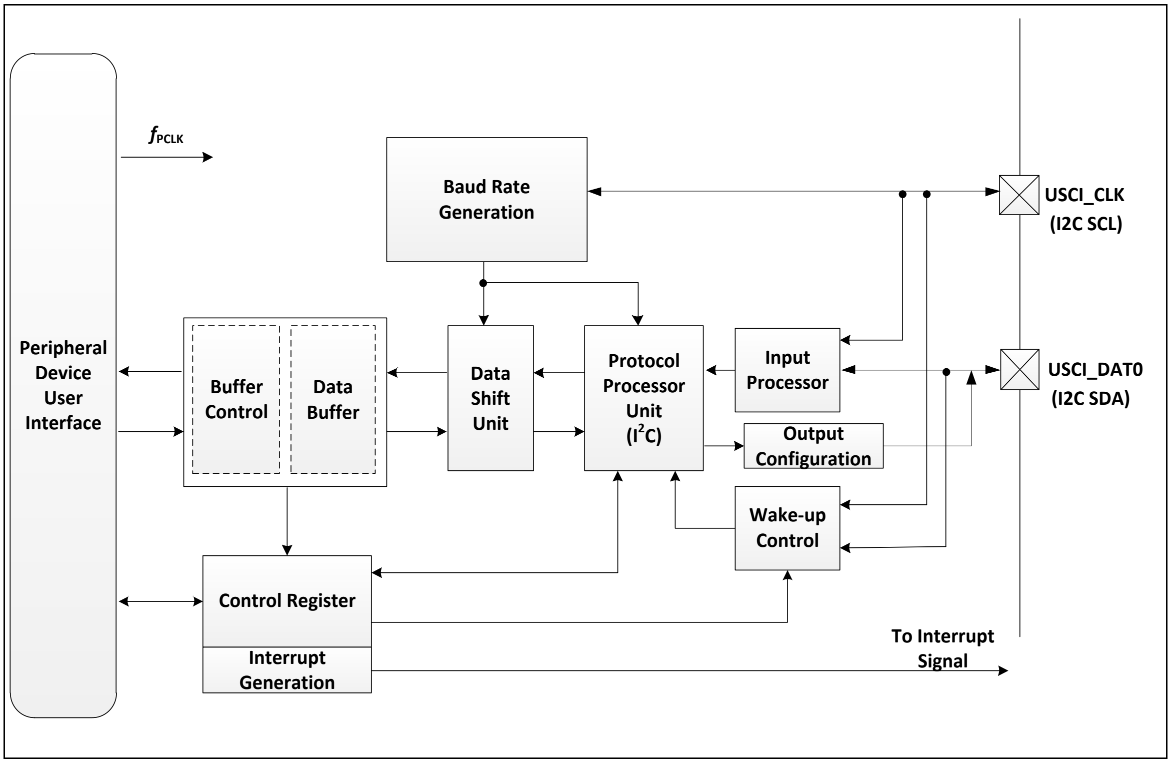


Figure 6.15-2 Block Diagram

6.15.4 Basic Configuration

The basic configurations of USCI0_I2C are as follows:

- USCI0 pins are configured in SYS_GPA_MFPH, SYS_GPB_MFPL, SYS_GPB_MFPH, SYS_GPC_MFPL, SYS_GPC_MFPH, SYS_GPD_MFPL, SYS_GPD_MFPH, and SYS_GPE_MFPL registers.
- Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK1[8]).
- Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).
- Enable USCI0_I2C function (FUNMODE=100b) on (UI2C_CTL[2:0]) register.

6.15.5 Functional Description

6.15.5.1 START or Repeated START signal

Figure 6.15-3 shows the typical I^2C protocol. Normally, a standard communication consists of four parts:

- START or Repeated START signal generation
- Slave address and R/W bit transfer
- Data transfer
- STOP signal generation

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

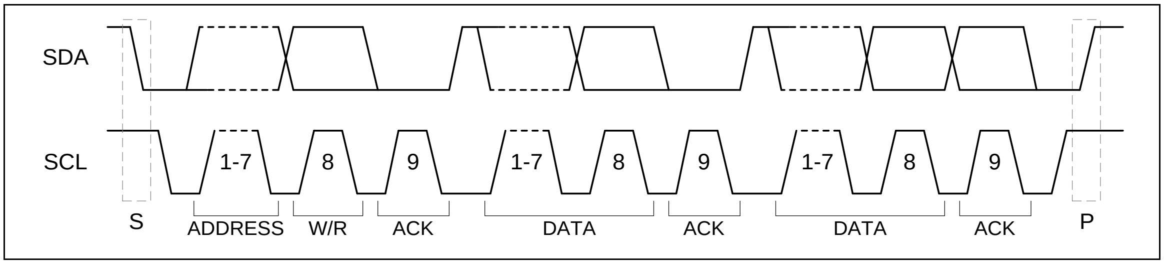


Figure 6.15-3 I^2C Protocol

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

A Repeated START is a START signal between START and STOP signals and usually referred to as the "Sr" bit. The master uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus idle flag.

6.15.5.2 STOP Signal

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

Figure 6.15-4 shows the waveform of START, Repeat START and STOP.

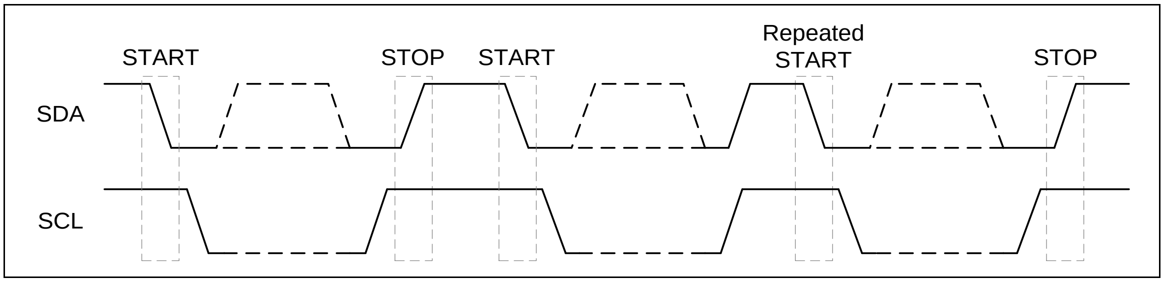


Figure 6.15-4 START and STOP Conditions

6.15.5.3 Slave Address Transfer

After a (repeated) start condition, the master sends a slave address to identify the target device of the communication. The start address can comprise one or two address bytes (for 7-bit or for 10-bit addressing schemes). After an address byte, a slave sensitive to the transmitted address has to acknowledge the reception.

Therefore, the slave's address can be programmed in the device, where it is compared to the received address. In case of a match, the slave answers with an acknowledge (SDA = 0). Slaves that are not targeted answer with a non-acknowledge (SDA = 1). In addition to the match of the programmed address, another address byte value has to be answered with an acknowledge if the slave is capable to handle the corresponding requests. The address byte 00H indicates a general call address that can be acknowledged.

To allow selective acknowledges for the different values of the address byte(s), the following control mechanism is implemented:

- If the GCFUNC bit (UI2C_PROTCTL [0]) is set the I^{2}C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.
- The I^{2}C port is equipped with one device address registers, UI2C_DEVADDRn (n = 0~1). In 7-bit address mode, the first 7 bits of a received first address byte are compared to the programmed slave address (UI2C_DEVADDRn [6:0]). If these bits match, the slave sends an acknowledge.
- In addition to this, if the slave address is programmed to 1111 0XXB, the XX bits are compared to the bits UI2C_DEVADDR [9:8] to check for address match and also sends an acknowledge when ADDR10EN (UI2C_PROTCTL [4]) is set. The slave waits for a second address byte compares it with UI2C_DEVADDR [7:0] and sends an acknowledge accordingly to cover the 10 bit addressing mode. The user has to take care about reserved addresses (refer to I^{2}C specification for more detailed description). Only the address 1111 0XXB is supported. Under each of these conditions, bit SLASEL (UI2C_PROTSTS [14]) will be set when the addressing delivered a match. This SLASEL (UI2C_PROTSTS [14]) bit is cleared automatically by a (repeated) start or stop condition.
- The I^{2}C port is equipped multiple address recognition with one address mask registers ADDRMSK (UI2C_DEVADDRn[9:0], n = 0~1). When the bit in the address mask register is set to 1, it means the received corresponding address bit is "Don’t care". If the bit is set to 0, it means the received corresponding register bit should be exactly the same as address register.

6.15.5.4 Data Transfer

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

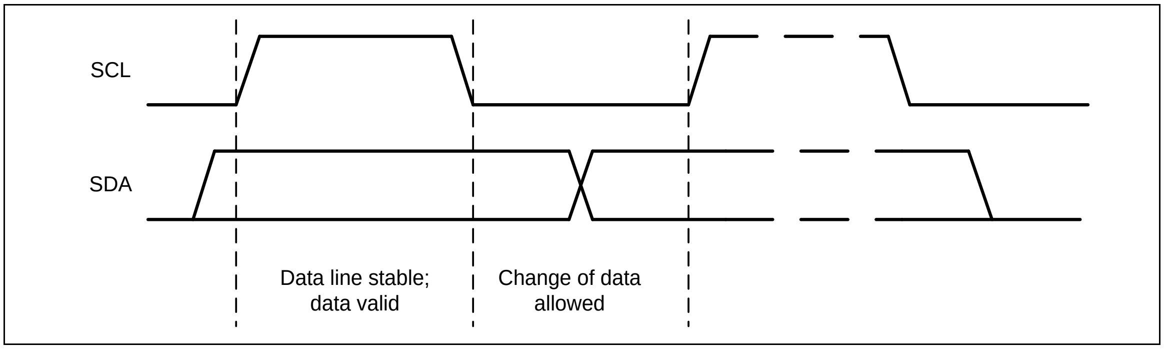


Figure 6.15-5 Bit Transfer on the I^{2}C Bus

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

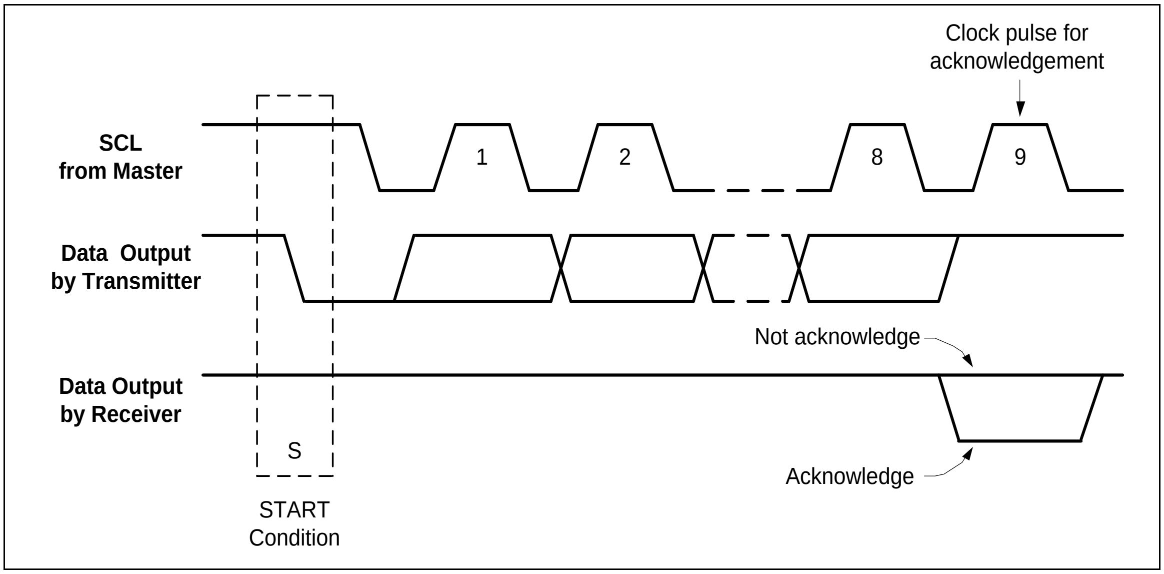


Figure 6.15-6 Acknowledge on the I2C Bus

6.15.5.5 *Clock Baud Rate Bits*

For this section, please refer to Figure 6.12-9 Basic Clock Divider Counter. The data baud rate of I^2C is determined by UI2C_BRGEN register when I^2C is in Master Mode, and it is not necessary in Slave mode. In the Slave mode, I^2C will automatically synchronize it with any clock frequency from master I^2C device. The bits RCLKSEL, SPCLKSEL, PDSCNT, and DSCNT define the baud rate setting:

RCLKSEL (UI2C_BRGEN [0])

to define the input frequency f_{REF_CLK}

SPCLKSEL (UI2C_BRGEN[3:2])

to define the multiple source of the sample clock f_{SAM_CLK}

PDSCNT (UI2C_BRGEN [9:8])

to define the length of a data sample time (division of f_{REF_CLK} by 1, 2, 3, or 4)

DSCNT (UI2C_BRGEN [14:10])

to define the number of data sample time per bit time

The standard setting is given by RCLKSEL = 0 ($f_{REF\_CLK} = f_{PCLK}$), PTCLKSEL = 0 ($f_{PROT\_CLK} = f_{REF\_CLK}$) and SPCLKSEL = 2’b00 ($f_{SAMP\_CLK} = f_{DIV\_CLK}$). Under these conditions, the baud rate is given by:

$$f_{I2C} = \frac{f_{REF\_CLK}}{2} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

To generate slower frequencies, additional divide-by-2 stages can be selected by PTCLKSEL = 1 ($f_{PROT\_CLK} = f_{REF\_CLK2}$), leading to:

$$f_{I2C} = \frac{f_{REF\_CLK}}{4} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

If SPCLKSEL = 2'b10 (f_{SAMP_CLK} = f_{SCLK}), and RCLKSEL = 0 (f_{REF_CLK} = f_{PCLK}), PTCLKSEL = 0 (f_{PROT_CLK} = f_{REF_CLK}). The baud rate is given by:

$$f_{\mathrm{I2C}} = \frac{f_{REF\_CLK}}{2} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{2} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

6.15.5.6 Byte Stretching

If a device is selected as transceiver and should transmit a data byte but the transmit buffer TXDAT does not contain valid data to be transmitted, the device ties down SCL = 0 at the end of the previous acknowledge bit. The waiting period is finished if software writes 1 to PTRG (UI2C_PROTCTL [5]).

6.15.5.7 Master arbitration

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 two masters sometimes initiate I^{2}C 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. Master I^{2}C 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 I^{2}C master must monitor the I^{2}C bus for collisions and act accordingly. Figure 6.15-7 illustrates that DATA1 and DATA2 are compete arbitration.

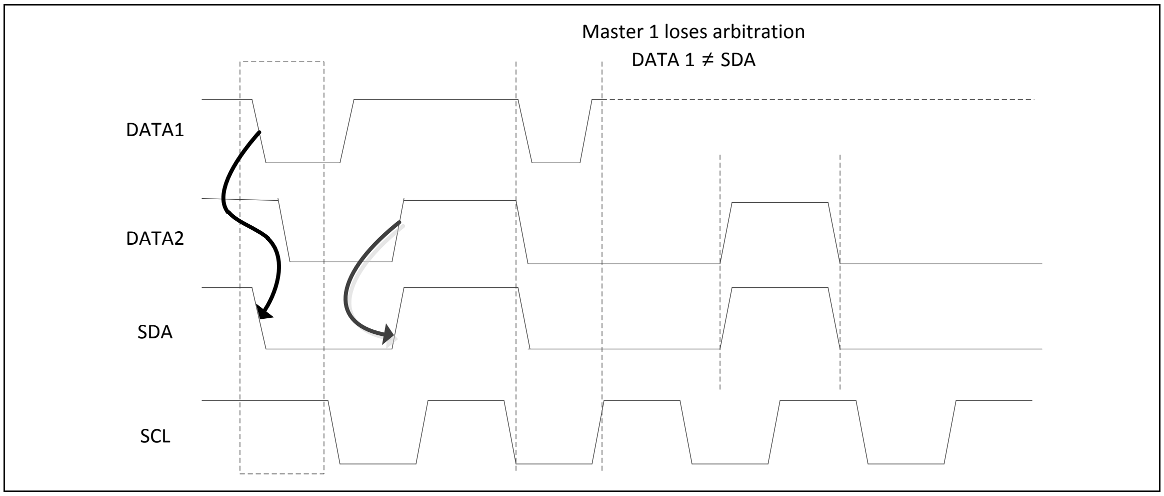


Figure 6.15-7 Arbitration Lost

In this case, during the address and data transmission, the master transmitter checks at the rising edge of SCL for each data bit if the value it is sending is equal to the value read on the SDA line. If yes, the next data bit values can be 0. If this is not the case (transmitted value = 1, value read = 0), the master has lost the transmit arbitration. This is indicated by interrupt flag ARBLOIF (UI2C_PROTSTS [11]) and can generate a protocol interrupt if enabled by ARBLOIEN (UI2C_PROTIEN [4]).

When the transmit arbitration has been lost, the software has to initialize the complete frame again, starting with the first address byte together with the start condition for a new master transmit attempt. Arbitration also takes place for the ACK bit. If master arbitration lost and match the device address, then master will turn to slave.

6.15.5.8 Transmission Chain

The I^{2}C bus protocol requiring a kind of in-bit-response during the arbitration phase and while a slave is transmitting, the resulting loop delay of the transmission chain can limit the reachable maximal baud rate, strongly depending on the bus characteristics (bus load, module frequency, etc.).

The shift clock SCL is generated by the master device, output on the wire, then it passes through the input stage and the input filter. Now, the edges can be detected and the SDA data signal can be generated accordingly. The SDA signal passes through the output stage and the wire to the master receiver part. There, it passes through the input stage and the input filter before it is sampled.

This complete loop has to be finished (including all settling times to obtain stable signal levels) before the SCL signal changes again. The delays in this path have to be taken into account for the calculation of the baud rate as a function of f_{PCLK} and f_{PROT_CLK}. We suggest user adopt f_{PCLK}.

6.15.5.9 Non-Acknowledge and Error Conditions

In case of a non-acknowledge (NACKIF (UI2C_PROTSTS [10])) or an error (ERRIF(UI2C_PROTSTS [12])), no further transmission will take place. User software doesn't invalidate the transmit buffer and disable transmissions, before configuring the transmission (by writing TXDAT) again with appropriate values to react on the previous event.

6.15.5.10 Interrupt

I^{2}C Protocol Interrupt Events

The following protocol-related events are generated in I^{2}C mode and can lead to a protocol interrupt.

Please note that the bits in register UI2C_PROTSTS are not all automatically cleared by hardware and have to be cleared by software in order to monitor new incoming events.

- Start condition received at a correct position in a frame (STARIF (UI2C_PROTSTS [8]))
- Stop condition transferred at a correct position in a frame (STORIF (UI2C_PROTSTS [9]))
- Master arbitration lost (ARBLOIF (UI2C_PROTSTS [11]))
- Slave read requested (SLAREAD (UI2C_PROTSTS [15]))
- Acknowledge received (ACKIF (UI2C_PROTSTS [13]))
- Non-acknowledge received (NACKIF (UI2C_PROTSTS [10]))
- Start condition not at the expected position in a frame (ERRIF (UI2C_PROTSTS [12]))
- Stop condition not at the expected position in a frame (ERRIF (UI2C_PROTSTS [12]))

6.15.5.11 Operating the I^{2}C

To operate the I^{2}C protocol, the following issues have to be considered:

Select I^{2}C Mode:

It is recommended to configure all parameters of the I^{2}C that do not change during run time while FUNMODE (UI2C_CTL [2:0]) = 000B. The I^{2}C control flow has to be done while FUNMODE

(UI2C_CTL [2:0]) = 000B to avoid unintended edges of the input signals and the I^{2}C mode can be enabled by FUNMODE (UI2C_CTL [2:0]) = 100b afterwards.

-Step 1. Set FUNMODE (UI2C_CTL [2:0]) = 000b

-Step 2. Set FUNMODE (UI2C_CTL [2:0]) = 100b

Pin Connections:

The pins used for SDA and SCL have to be set to open-drain mode by USCI controller to support the wired-AND structure of the I^{2}C bus lines.

Note: *The step to enable the alternate output port functions should only be done after the I^{2}C mode is enabled, to avoided unintended spikes on the output.*

Bit Timing Configuration:

In standard mode (100 kBit/s) a minimum module frequency of 2 MHz is necessary, whereas in fast mode (400 kBit/s) a minimum of 10 MHz is required. Additionally, if the digital filter stage should be used to eliminate spikes up to 50 ns, a filter frequency of 20 MHz is necessary. There could be an uncertainty in the SCL high phase timing of maximum $1/f_{PROT\_CLK}$ if another I^{2}C participant lengthens the SCL low phase on the bus. Note that the SCL maximum frequency is SAMP_CLK/2 and the SPCLKSEL (UI2C_BRGEN [3:2]) must be set to 0 for selecting $f_{SAMP\_CLK} = f_{DIV\_CLK}$.

Data Format Configuration:

The data format has to be configured for 8 data bits (DWIDTH (UI2C_LINECTL [11:8]) = 8), and MSB shifted first (LSB (UI2C_LINECTL [0]) = 0). As a result, UI2C_LINECTL has to be set to 0x800.

Control Flow:

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

In a given application, I^{2}C port may operate as a master or as a slave. In Slave mode, the I^{2}C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master(by setting the AA bit), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, hardware waits until the bus is free before entering Master mode so that a possible slave action is not be interrupted. If address 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 UI2C_PROTCTL, UI2C_PROTIEN, TXDAT registers according to current status of UI2C_PROTSTS register. In other words, for each I^{2}C bus action, user needs to check current status by UI2C_PROTSTS register, and then set UI2C_PROTCTL, UI2C_PROTIEN, TXDAT registers to take bus action. Finally, check the response status by UI2C_PROTSTS.

The bits, STA, STO and AA in UI2C_PROTCTL register are used to control the next state of the I^{2}C hardware after interrupt signal is cleared. Upon completion of the new action, a new status will be updated in UI2C_PROTSTS register will be set. If the I^{2}C interrupt control bit of UI2C_PROTIEN is set, appropriate action or software branch of the new status can be performed in the Interrupt service routine.

Figure 6.15-8 shows the current I^{2}C STARIF (UI2C_PROTSTS [8]) is set to 1 by hardware, and then set TXDAT = SLA+W (Slave address + Write bit), (PTRG, STA, STO, AA) = (1, 0, 0, x) to send the address to I^{2}C bus, and write 1 to STARIF (UI2C_PROTSTS [8]) to clear flag. If a slave on the bus matches the address and response ACK, the UI2C_PROTSTS will be updated by ACKIF (UI2C_PROTSTS [13]) setting.

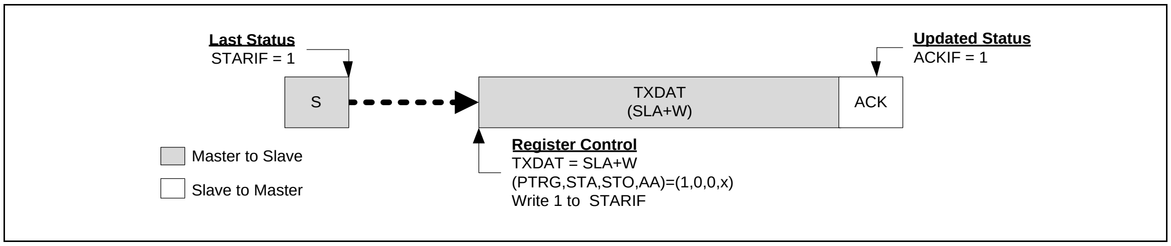


Figure 6.15-8 Control I^{2}C Bus according to Current I^{2}C Status

Data Transfer on the I^{2}C Bus

Figure 6.15-9 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.

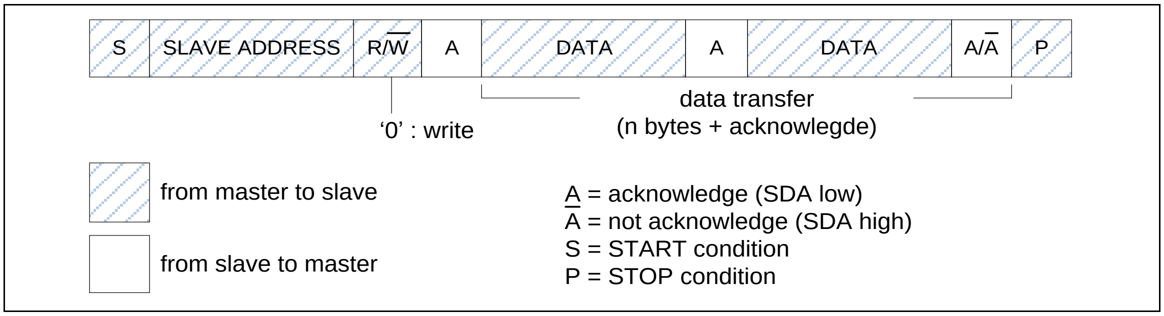


Figure 6.15-9 Master Transmits Data to Slave with a 7-bit Address

Figure 6.15-10 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.

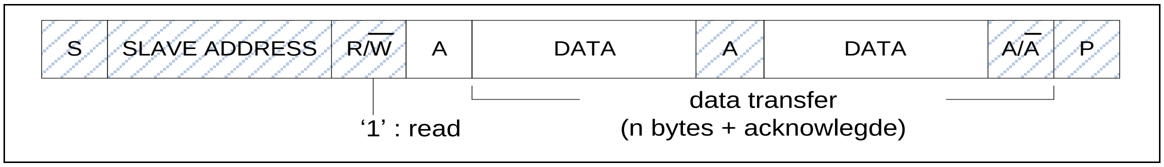


Figure 6.15-10 Master Reads Data from Slave with a 7-bit Address

Figure 6.15-11 shows a master transmits data to slave by 10-bit address. A master addresses a slave with a 10-bit address. First byte contains 10-bit address indicator (5'b11110) and 2-bit address with write index, second byte contains 8-bit address. The master keeps transmitting data after the second byte end. Note that 7-bit and 10-bit address device can work on the same bus.

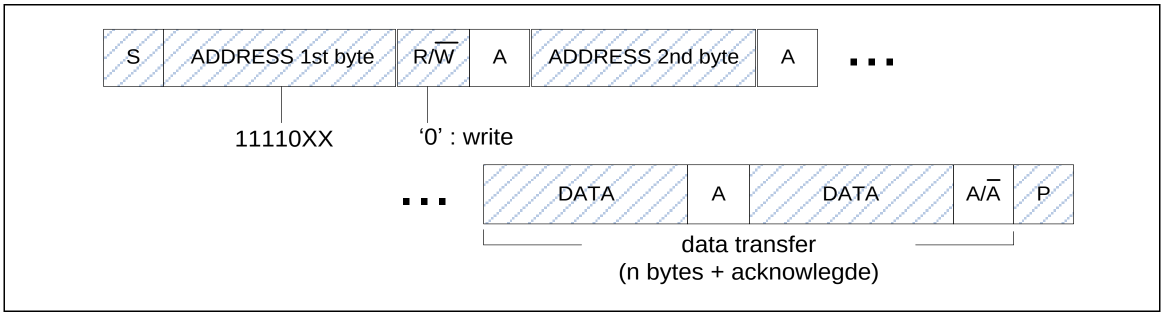


Figure 6.15-11 Master Transmits Data to Slave by 10-bit Address

Figure 6.15-12 shows a master read data from slave by 10-bit address. A master addresses a slave with a 10-bit address. First mater transmits 10-bit address to slave, after that master transmits first byte with read index. The slave will start transmitting data after the first byte with read index.

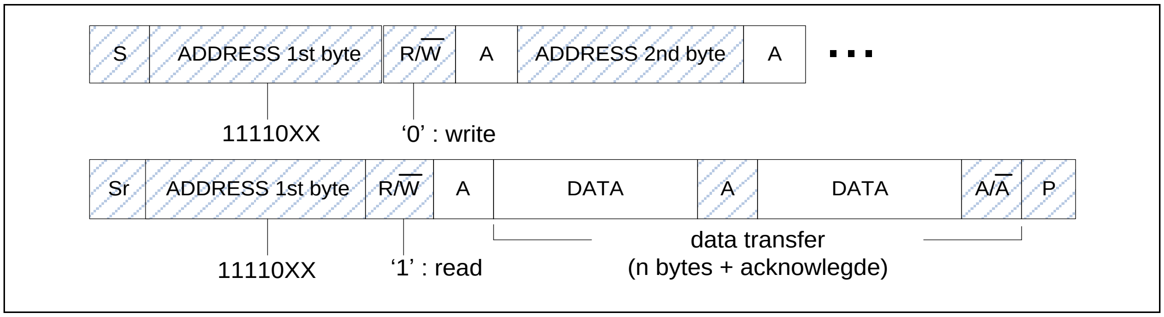


Figure 6.15-12 Master Reads Data from Slave by 10-bit Address

Master Mode

In Figure 6.15-13 and Figure 6.15-14, all possible protocols for I^{2}C master are shown. User needs to follow a proper path of the flow to implement required I^{2}C protocol.

In other words, user can send a START signal to bus and I^{2}C will be in Master Transmitter mode (Figure 6.15-13) or Master receiver mode (Figure 6.15-14) after START signal has been sent successfully and new status register would be set STARIF (UI2C_PROTSTS [8]). Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^{2}C protocol.

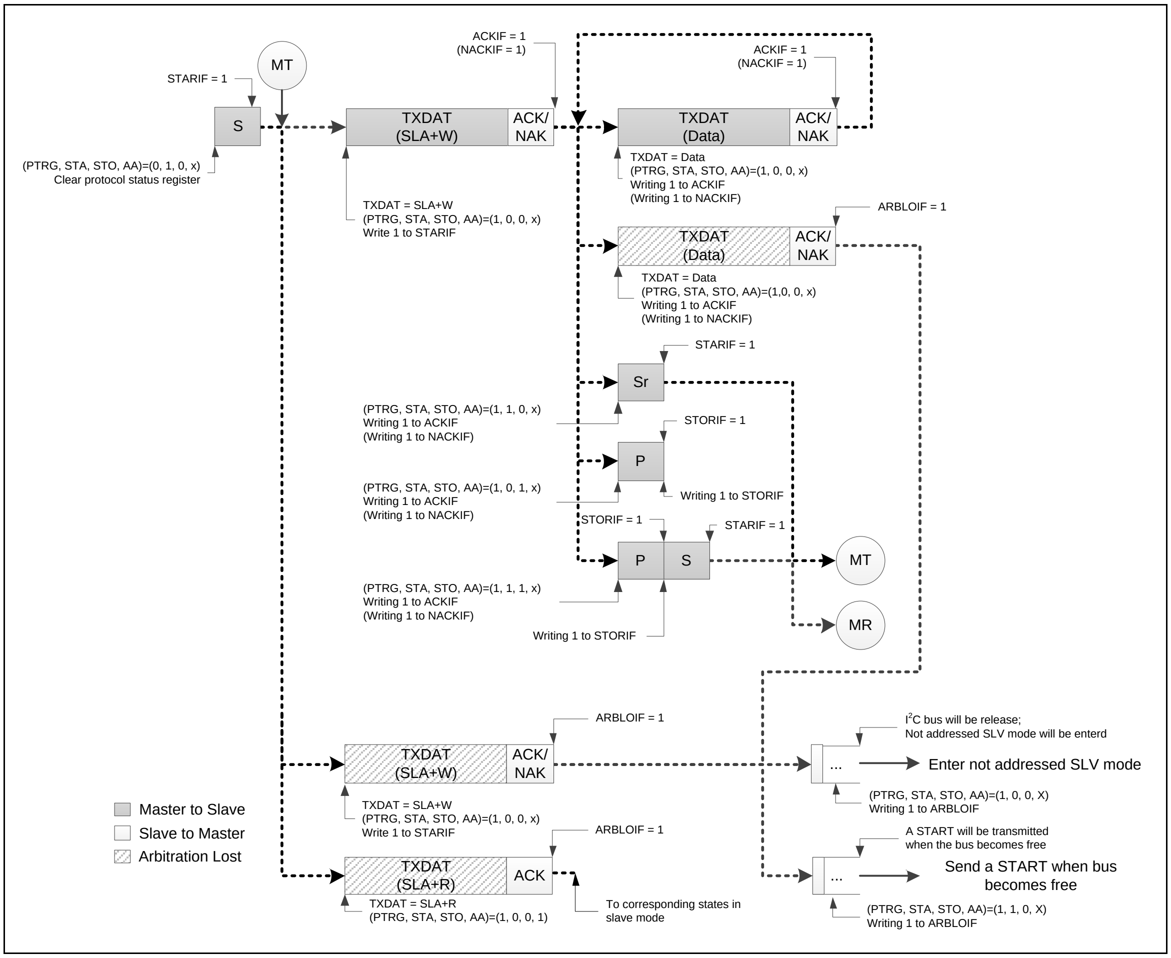


Figure 6.15-13 Master Transmitter Mode Control Flow with 7-bit Address

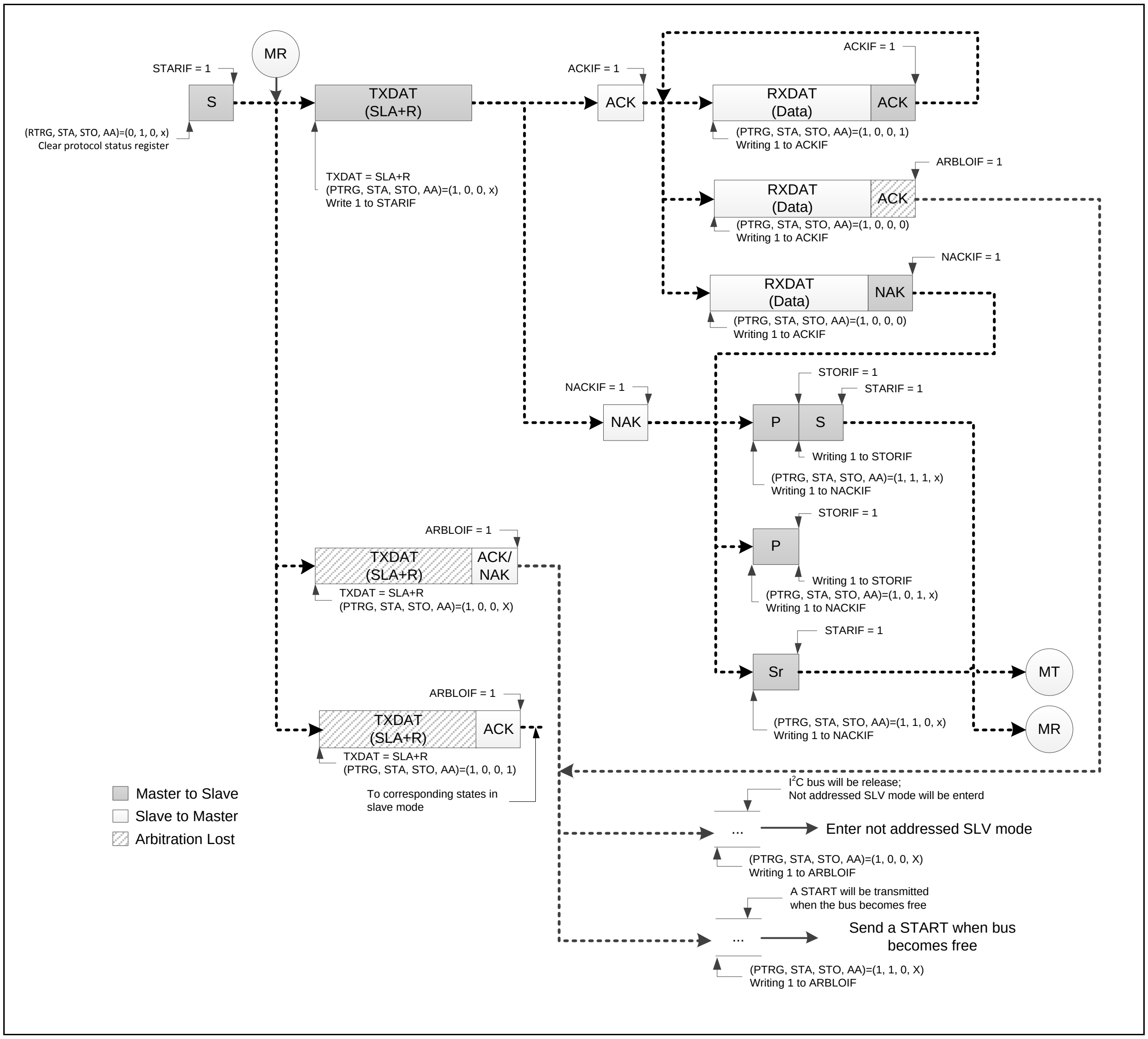


Figure 6.15-14 Master Receiver Mode Control Flow with 7-bit Address

If the I^{2}C is in Master mode and gets arbitration lost, the bit of ARBLOIF (UI2C_PROTSTS [11]) will be set. User may writing 1 to ARBLOIF (UI2C_PROTSTS [11]) and set (PTRG, STA, STO, AA) = (1, 1, 0, X) to send START to re-start Master operation when bus become free. Otherwise, user may writing 1 to ARBLOIF (UI2C_PROTSTS [11]) and set (PTRG, STA, STO, AA) = (1, 0, 0, X) to release I^{2}C bus and enter not addressed Slave mode.

Slave Mode

When reset, I^{2}C is not addressed and will not recognize the address on I^{2}C bus. User can set device address by UI2C_DEVADDRn and set (PTRG, STA, STO, AA) = (1, 0, 0, 1) to let I^{2}C recognize the address sent by master. Figure 6.15-15 shows all the possible flow for I^{2}C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.15-15 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) or SLA+R (Master want to read data from Slave) after arbitration lost, the ARBLOIF will be set to 1.

The I^{2}C controller supports two slave address match flags, are ADMAT0 and ADMAT1 on UI2C_ADMAT[1:0] register. Every control register represent which address is used and set 1 to inform software.

Note: During I^{2}C communication, the SCL clock will be released when writing ‘1’ to PTRG (UI2C_PROTCTL [5]) in Slave mode.

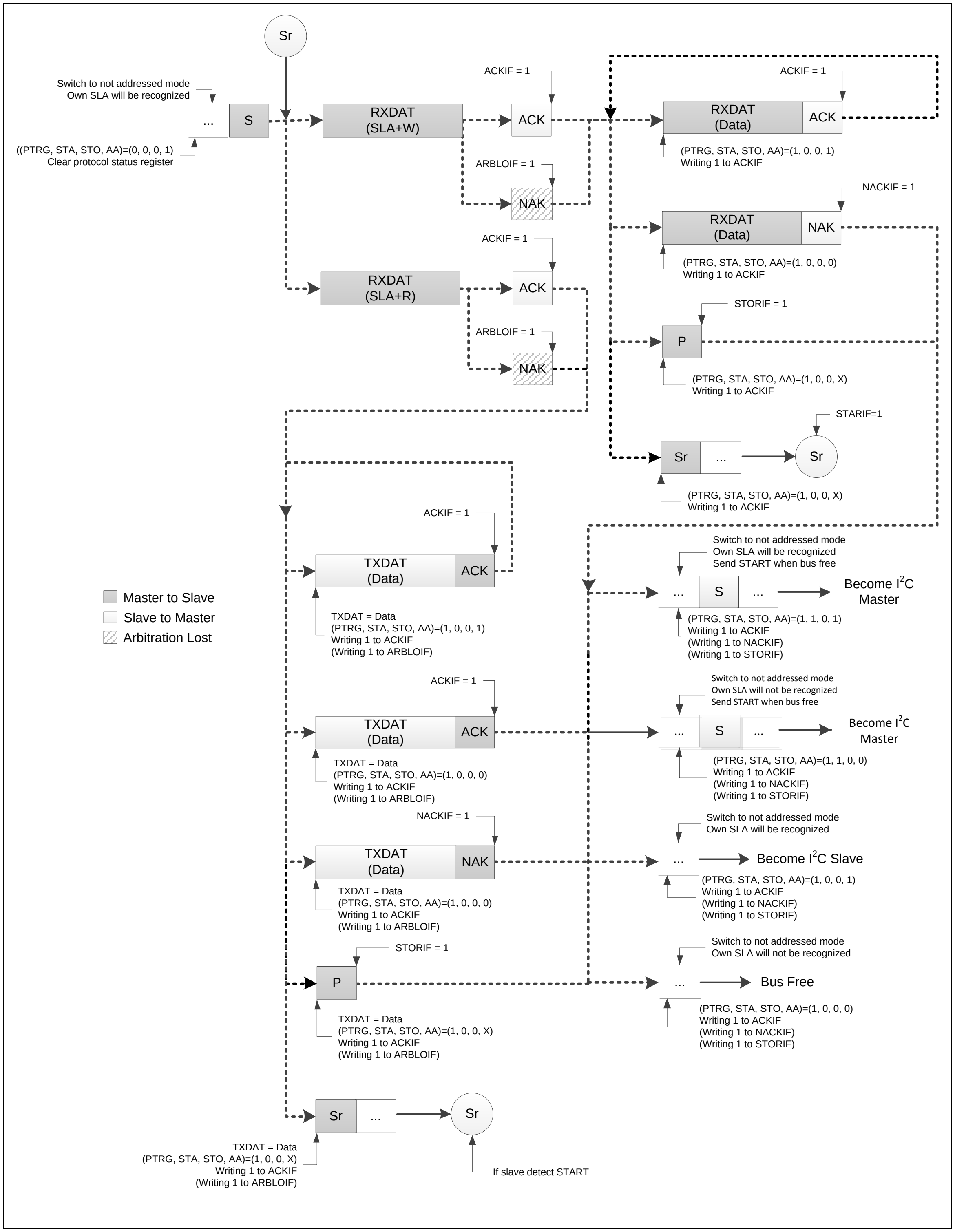


Figure 6.15-15 Save Mode Control Flow with 7-bit Address

If I^{2}C is still transmitting and receiving data in addressed Slave mode but got a STOP or Repeat START, the register STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) will be set. User could follow the action for NACKIF (UI2C_PROTSTS [10]) as shown in the above figure when got STARIF (UI2C_PROTSTS [8]) is set.

Note: After slave gets interrupt flag of NACKIF (UI2C_PROTSTS [10]) and start/stop symbol including STARIF (UI2C_PROTSTS [8]) and STORIF (UI2C_PROTSTS [9]), slave can switch to not address mode and own SLA will not be recognized. If setting this interrupt flag, slave will not receive any I^{2}C signal or address from master. At this status, I^{2}C should be reset by setting FUNMODE (UI2C_CTL [2:0]) = 000B to leave this status.

General Call (GC) Mode

If the GCFUNC bit (UI2C_PROTCTL [0]) is set, the I^{2}C port hardware will respond to General Call address (00H). User can clear GC bit to disable general call function. When the GC bit is set and the I^{2}C in slave mode, it can receive the general call address by 0x00 after master send general call address to I^{2}C bus, and then it also will follow protocol status register.

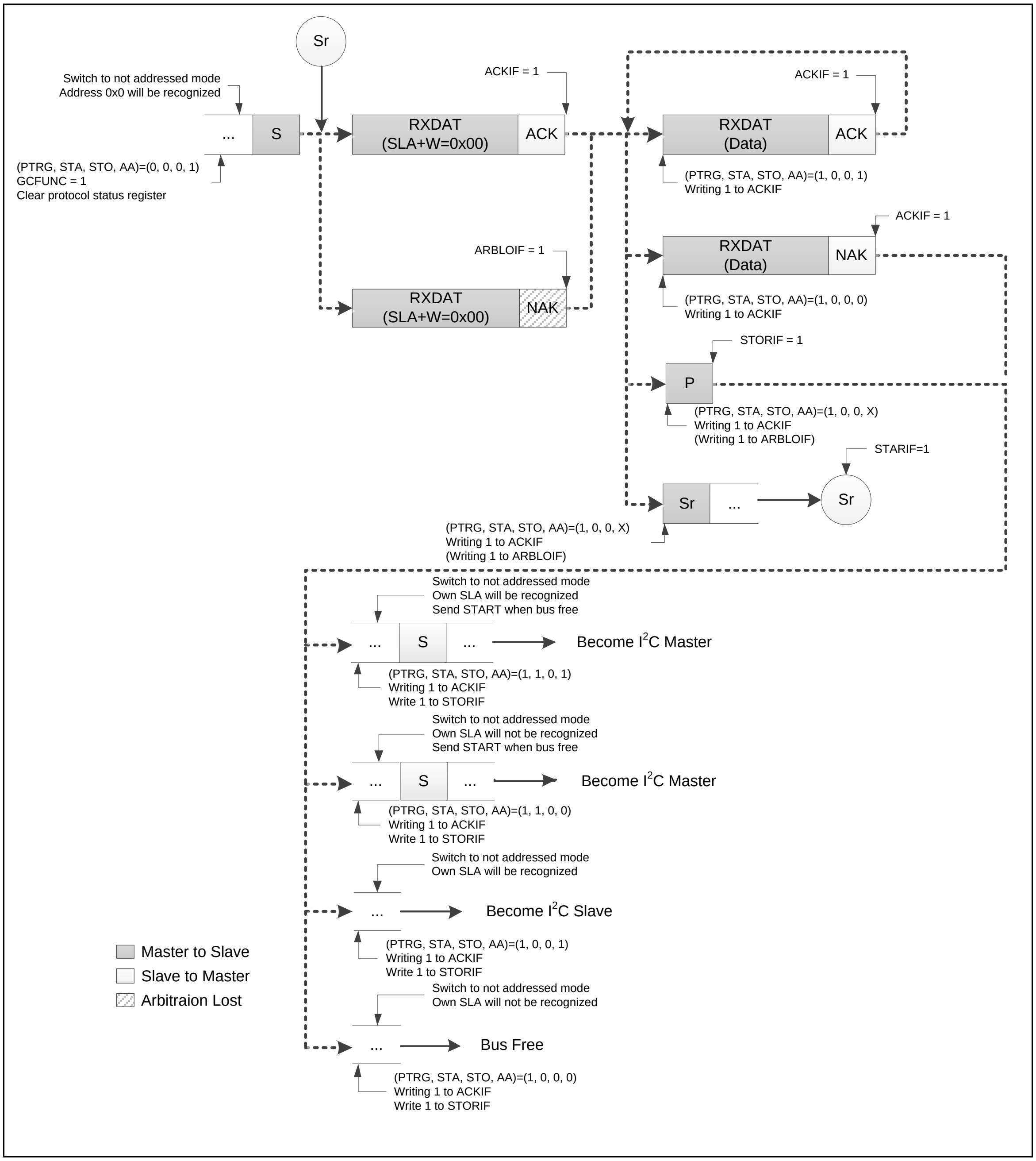

Sr
Switch to not addressed mode
Address 0x0 will be recognized
...
S
(PTRG, STA, STO, AA)=(0, 0, 0, 1)
GCFUNC = 1
Clear protocol status register
ACKIF = 1
RXDAT
(SLA+W=0x00)
ACK
ARBLOIF = 1
RXDAT
(SLA+W=0x00)
NAK
ACKIF = 1
RXDAT
(Data)
ACK
(PTRG, STA, STO, AA)=(1, 0, 0, 1)
Writing 1 to ACKIF
ACKIF = 1
RXDAT
(Data)
NAK
(PTRG, STA, STO, AA)=(1, 0, 0, 0)
Writing 1 to ACKIF
STORIF = 1
P
(PTRG, STA, STO, AA)=(1, 0, 0, X)
Writing 1 to ACKIF
(Writing 1 to ARBLOIF)
STARIF=1
Sr
...
Sr
(PTRG, STA, STO, AA)=(1, 0, 0, X)
Writing 1 to ACKIF
(Writing 1 to ARBLOIF)
Switch to not addressed mode
Own SLA will be recognized
Send START when bus free
...
S
...
Become I2C Master
(PTRG, STA, STO, AA)=(1, 1, 0, 1)
Writing 1 to ACKIF
Write 1 to STORIF
Switch to not addressed mode
Own SLA will not be recognized
Send START when bus free
...
S
...
Become I2C Master
(PTRG, STA, STO, AA)=(1, 1, 0, 0)
Writing 1 to ACKIF
Write 1 to STORIF
Switch to not addressed mode
Own SLA will be recognized
...
Become I2C Slave
(PTRG, STA, STO, AA)=(1, 0, 0, 1)
Writing 1 to ACKIF
Write 1 to STORIF
Switch to not addressed mode
Own SLA will not be recognized
...
Bus Free
(PTRG, STA, STO, AA)=(1, 0, 0, 0)
Writing 1 to ACKIF
Write 1 to STORIF
Master to Slave
Slave to Master
Arbitraion Lost

Figure 6.15-16 GC Mode with 7-bit Address

If I^{2}C is still receiving data in GC mode but got a STOP or Repeat START, the STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) will be set. User could follow the action for NACKIF (UI2C_PROTSTS [10]) in above figure when got STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) is set.

Note: After slave gets interrupt flag of NACKIF (UI2C_PROTSTS [10]) and start/stop symbol including STARIF (UI2C_PROTSTS [8]) and STORIF (UI2C_PROTSTS [9]), slave can switch to not address mode and own SLA will not be recognized. If setting this interrupt flag, slave will not receive any I^{2}C signal or address from master. At this time, the I^{2}C controller should be reset by setting FUNMODE (UI2C_CTL [2:0]) = 000B to leave this status.

Protocol Functional Description

Monitor Mode

When I^{2}C enters monitor mode, this device always returns NACK to master after each frame reception even address matching. Moreover, this device will store any receive data including address, command code, and data. Note that monitor mode does not support 10-bit mode.

Interrupt in Monitor Mode

All interrupts will occur as normal process when the MONEN (UI2C_PROTCTL [9]) is set. Note that the first interrupt will occur when initial START, it's not the same as I^{2}C slave, but the other interrupts are the same.

Subsequent to the address-match detection, interrupts will be generated after each data byte is received as slave mode control flow, or after each byte that the module believes it has transmitted for a slave-read transfer. In this second case, the data register will actually contain data transmitted by some other slave on the bus which was actually addressed by the master. If user wants to watch other device, user can set address mask and monitor.

If the monitor has not had time to respond to an interrupt, the SCL signal will be pulled to low when SCLOUTEN (UI2C_PROTCTL [8]) is set to 1. User must set PTRG (UI2C_PROTCTL [5]) to release bus when SCLOUTEN (UI2C_PROTCTL [8]) is set to 1. If SCLOUTEN (UI2C_PROTCTL [8]) is not set to 1, user doesn't need to set PTRG (UI2C_PROTCTL [5]) to 1.

When device address match, but the device response NACK, this address will be received into buffer and NACK interrupt will be generated.

Following all of these interrupts, the processor may read the data register to see what was actually transmitted on the bus.

Loss of arbitration in Monitor Mode

In monitor mode, the I^{2}C module will not be able to respond to a request for information by the bus master or issue an ACK. Some other slave on the bus will respond instead. Software should be aware of the fact that the module is in monitor mode and should not respond to any loss of arbitration state that is detected.

Programmable Setup and Hold Time

To guarantee a correct data setup and hold time, the timing must be configured by programming HTCTL [5:0] (UI2C_TMCTL[11:6]) to configure hold time and STCTL [5:0] (UI2C_TMCTL[5:0]) to configure setup time.

The delay timing refers to peripheral clock (PCLK). When device stretch master clock, the setup and hold time configuration value will not be affected by stretched.

User should focus on the limitation of setup and hold time configuration. The timing setting must follow I^{2}C protocol. Once setup time configuration is greater than design limitation, that means if setup time setting makes SCL output less than three PCLKs, the I^{2}C controller can not work normally due to SCL must sample three times. Once the hold time configuration is greater than I^{2}C clock limitation, I^{2}C will occur bus error. It is suggested that user calculate suitable timing with baud rate and protocol before setting timing. Table 6.15-1 shows the relationship between I^{2}C baud rate and PCLK, the number of table represent one clock duty contain how many PCLKs. Setup and hold time configuration even can program some extreme values in the design, but user should follow I^{2}C protocol standard.

I^2C Baud Rate PCLK	100k	200k	400k
12 MHz	120	60	30
24 MHz	240	120	60
48 MHz	480	240	120
72 MHz	720	360	180

Table 6.15-1 Relationship between I^2C Baud Rate and PCLK

For setup time wrong adjustment example, we assume one SCL cycle contains 10 PCLKs and set STCTL [5:0] (UI2C_TMCTL[5:0]) to 3 that stretch three PCLKs for setup time setting. The setup time setting limitation: ST_{limit} = (UI2C_BRGEN[25:16]+1) - 6. For example, if user determines PCLK = 12MHz and baud rate =100k, the UI2C_BRGEN[25:16] must be set to 59, and the STCTL [5:0] maximum value is 54.

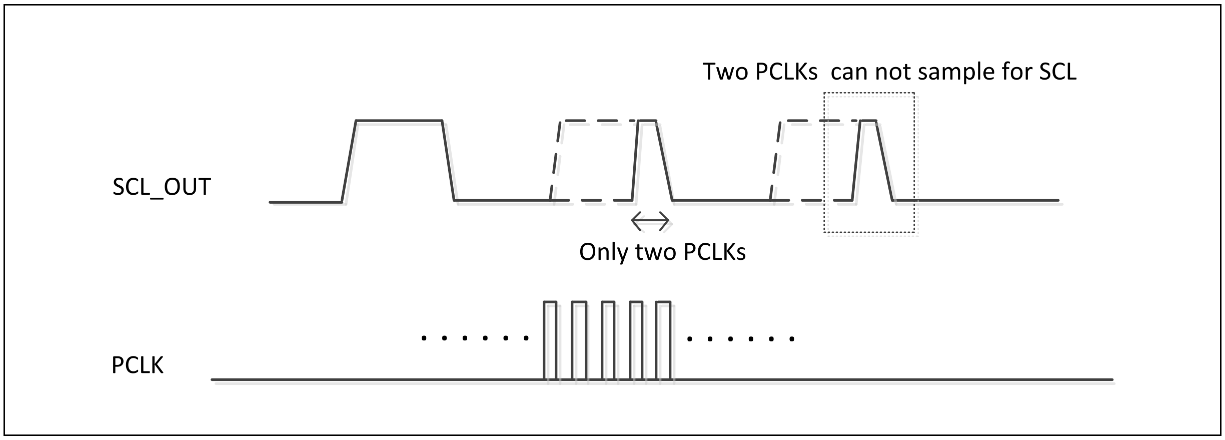


Figure 6.15-17 Setup Time Wrong Adjustment

For hold time wrong adjustment example, we use I^2C Baud Rate = 400k and PCLK = 24 MHz, the SCL high/low duty = 60 PCLK. When we set HTCTL [5:0] (UI2C_TMCTL[11:6]) to 63 and STCTL [5:0] (UI2C_TMCTL[5:0]) to 0, then SDA output delay will over SCL high duty and cause bus error. The hold time setting limitation: HT_{limit} = (UI2C_BRGEN[25:16]+1) - 9. For example, if user decide PCLK = 12MHz and baud rate =100k, the UI2C_BRGEN[25:16] must set 59, and the HTCTL [5:0] maximum value is 51.

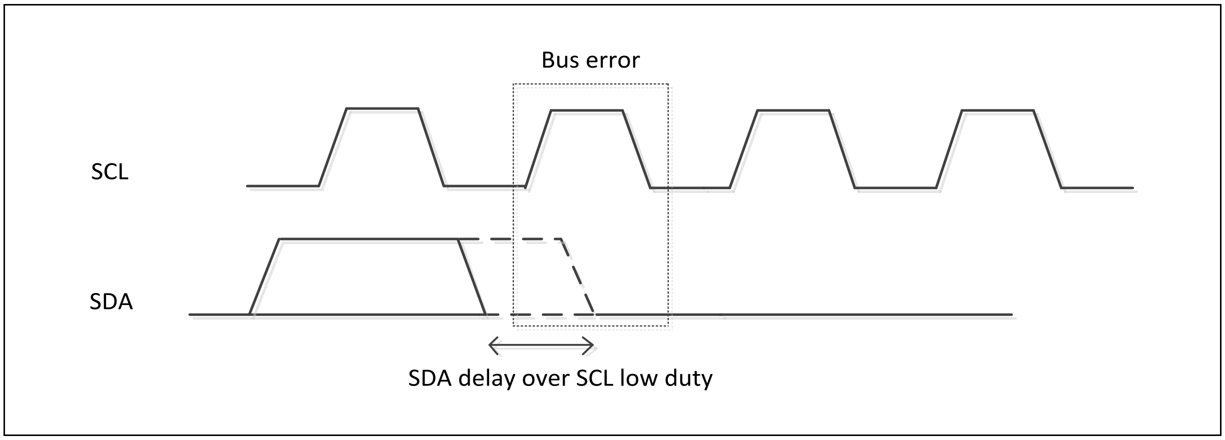


Figure 6.15-18 Hold Time Wrong Adjustment

I^{2}C Time-out Function

There is a time-out counter TOCNT (UI2C_PROTCTL [25:16]) which can be used to deal with the I^{2}C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it equals TOCNT (UI2C_PROTCTL [25:16]) and generates I^{2}C interrupt to CPU or stops counting by clearing TOIEN (UI2C_PROTIEN [0]) to 0 or setting all I^{2}C interrupt signal (ACKIF, ERRIF, ARBLOIF, NACKIF, STORIF, STARIF). User may write 1 to clear TOIF(UI2C_PROTSTS[5]) to 0. When time-out counter is enabled, writing 1 to the TOIF will reset counter and re-start up counting after TOIF is cleared. Refer to Figure 6.15-19 for the time-out counter TOCNT (UI2C_PROTCTL [25:16]). $T_{\mathbf{TOCNT}}$ = (TOCNT (UI2C_PROTCTL [25:16]) +1) x32 (5-bit) x $T_{\mathbf{PCLK}}$. Note that the time counter clock source TMCNTSRC (UI2C_ BRGEN [5]) must be set zero.

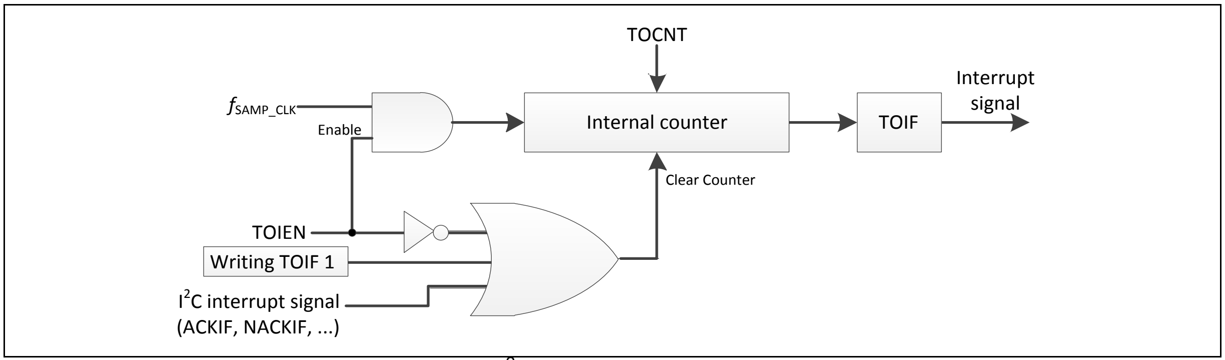


Figure 6.15-19 I^{2}C Time-out Count Block Diagram

Wake-up Functional Description

When chip enters Power-down mode and set WKEN (UI2C_WKCTL[0]) to 1, other I^{2}C master can wake up the chip by addressing the I^{2}C device, user must configure the related setting before entering sleep mode. The ACK bit cycle of address match frame is done in power-down. The controller will stretch the SCL to low when the address is matched with the device's address and the ACK cycle done. The SCL is stretched until the bit is cleared by user. If the frequency of SCL is low speed and the system has wakeup from address match frame, the user shall check this bit to confirm this frame has transaction done and then to do the wakeup procedure. Therefore, when the chip is woken-up by address match with one of the DEVADDR (UI2C_DEVADDRn) and set WKADDREN (UI2C_WKCTL[1]) to 1, the user shall check the WKAKDONE (UI2C_PROTSTS [16]) bit is set to 1 to confirm the address wakeup frame has done. The WKAKDONE bit indicates that the ACK bit cycle of address match frame is done in power-down. The controller will stretch the SCL to low when the address is matched with the device's slave address and the ACK cycle done. The SCL is stretched until the WKAKDONE bit is clear by user. If the frequency of SCL is low speed and the system has wakeup from address match frame, the user shall check this bit to confirm this frame has transaction done and then to do the wakeup procedure. Note that user must clear WKF after clearing the WKAKDONE bit to 0.

The WRSTSWK (UI2C_PROTSTS [17]) bit records the Read/Write command on the address match wake-up frame. The user can read this bit's status to prepare the next transmitted data (WRSTSWK = 0) or to wait the incoming data (WRSTSWK = 1) which can be stored in time after the system is wake-up by the address match frame.

When system is woken up by other I^{2}C master device, WKF (UI2C_WKSTS[0) is set to indicate this event. User needs write "1" to clear this bit.

I^{2}C also support data toggle mode. When system is in power-down and the WKEN (UI2C_WKCTL [0]) set to 1 and WKADDREN (UI2C_WKCTL[1]) set to 0, the toggle of incoming data pin can wake-up the system.

Example for Random Read on EEPROM

The following steps are used to configure the USCI0_I^{2}C related registers when using I^{2}C protocol to read data from EEPROM.

1. Set USCI0_I^{2}C the multi-function pin in the SYS_GPB_MFPL or SYS_GPBMFPH or SYS_GPC_MFPL or SYS_GPE_MFPL registers as SCL and SDA pins.
2. Enable USCI0 APB clock, USCI0CKEN =1 in the "CLK_APBCLK[8]" register.
3. Set USCI0RST=1 to reset USCI controller then set USCI0RST=0 let USCI controller to normal operation, in the "SYS_IPRST2[8]" register.
4. Set FUNMODE =100b to enable USCI0_I^{2}C controller in the "UI2C_CTL" register.
5. Give USCI0_I^{2}C clock a divided register value for USCI0_I^{2}C clock rate in the "UI2C_BRGEN".
6. Set SETENA =0x00400000 in the "NVIC_ISR" register to set UI2C_IRQ.
7. Set ACKIEN, ERRIEN, ARBLOIEN, NACKIEN, STORIEN, STARIEN, and TOIEN to enable I^{2}C Interrupt in the "UI2C_PROTIEN" register.
8. Set USCI address registers "UI2C_ADDR0 ~ UI2C_ADDR1".

Random read operation is one of the methods to access EEPROM. The method allows the master to access any address of EEPROM space. Figure 6.15-20 shows the EEPROM random read operation.

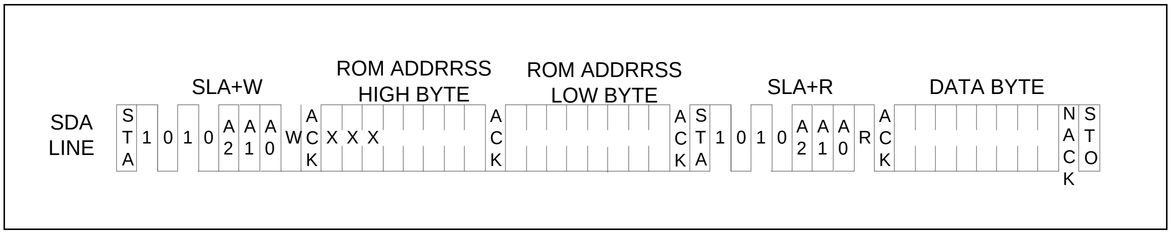


Figure 6.15-20 EEPROM Random Read

Figure 6.15-21 shows how to use the I^{2}C controller to implement the protocol of EEPROM random read.

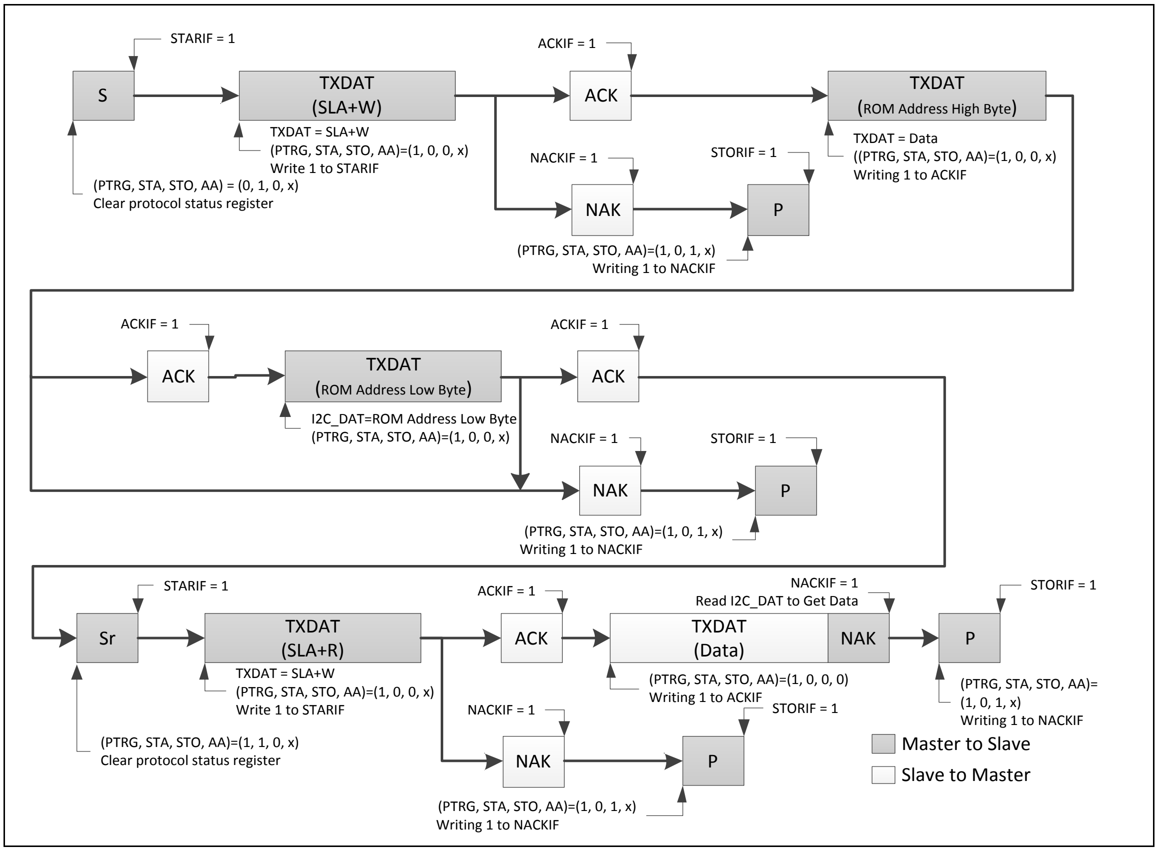


Figure 6.15-21 Protocol of EEPROM Random Read

The I^{2}C controller, which is a master, sends START to bus. Then, it sends a SLA+W (Slave address + Write bit) to EERPOM followed by two bytes data address to set the EEPROM address to read. Finally, a Repeat START followed by SLA+R is sent to read the data from EEPROM.

6.15.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UI2C Base Address: **UI2Cn_BA = 0x4007_0000 + (0x10_0000 * n)** **n = 0**				
UI2C_CTL **n = 0**	UI2Cn_BA+0x00	R/W	USCI Control Register	0x0000_0000
UI2C_BRGEN **n = 0**	UI2Cn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
UI2C_LINECTL **n = 0**	UI2Cn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
UI2C_TXDAT **n = 0**	UI2Cn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
UI2C_RXDAT **n = 0**	UI2Cn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
UI2C_DEVADDR0 **n = 0**	UI2Cn_BA+0x44	R/W	USCI Device Address Register 0	0x0000_0000
UI2C_DEVADDR1 **n = 0**	UI2Cn_BA+0x48	R/W	USCI Device Address Register 1	0x0000_0000
UI2C_ADDRMSK0 **n = 0**	UI2Cn_BA+0x4C	R/W	USCI Device Address Mask Register 0	0x0000_0000
UI2C_ADDRMSK1 **n = 0**	UI2Cn_BA+0x50	R/W	USCI Device Address Mask Register 1	0x0000_0000
UI2C_WKCTL **n = 0**	UI2Cn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
UI2C_WKSTS **n = 0**	UI2Cn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
UI2C_PROTCTL **n = 0**	UI2Cn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000
UI2C_PROTIEN **n = 0**	UI2Cn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
UI2C_PROTSTS **n = 0**	UI2Cn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000
UI2C_ADMAT **n = 0**	UI2Cn_BA+0x88	R/W	I^2C Slave Match Address Register	0x0000_0000
UI2C_TMCTL **n = 0**	UI2Cn_BA+0x8C	R/W	I^2C Timing Configure Control Register	0x0000_0000

6.15.7 Register Description

USCI Control Register (UI2C_CTL)

Register	Offset	R/W	Description	Reset Value
UI2C_CTL	UI2Cn_BA+0x00	R/W	USCI 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					**FUNMODE**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 000 to FUNMODE. 000 = The USCI is disabled. All protocol related state machines are set to idle state. 001 = The SPI protocol is selected. 010 = The UART protocol is selected. 100 = The I^{2}C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Baud Rate Generator Register (UI2C_BRGEN)

Register	Offset	R/W	Description	Reset Value
UI2C_BRGEN	UI2Cn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

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

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} (fDIV_CLK = fPROT_CLK / (CLKDIV+1)). **Note1:** I^{2}C function, the minimum value of CLKDIV is 8. **Note2:** In UART function, it can be updated by hardware in the 4th falling edge of the input data 0x55 when the auto baud rate function (ABREN(UI2C_PROTCTL[6])) is enabled. The revised value is the average bit time between bit 5 and bit 6. The user can use revised CLKDIV and new BRDETITV (UI2C_PROTCTL[24:16]) to calculate the precise baud rate.
[15]	**Reserved**	Reserved.
[14:10]	**DSCNT**	**Denominator for Sample Counter** This bit field defines the divide ratio of the sample clock f_{SAMP_CLK}. The divided frequency $f_{DS\_CNT} = f_{PDS\_CNT}$ / (DSCNT+1). **Note:** The maximum value of DSCNT is 0xF on UART mode and suggest to set over 4 to confirm the receiver data is sampled in right value.
[9:8]	**PDSCNT**	**Pre-divider for Sample Counter** This bit field defines the divide ratio of the clock division from sample clock f_{SAMP_CLK}. The divided frequency $f_{PDS\_CNT} = f_{SAMP\_CLK}$ / (PDSCNT+1).
[7:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Time Measurement Counter Clock Source Selection** 0 = Time measurement counter with f_{PROT_CLK}. 1 = Time measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Time Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Time measurement counter is Disabled. 1 = Time measurement counter is Enabled.
[3:2]	**SPCLKSEL**	**Sample Clock Source Selection**

		This bit field is used for the clock source selection of a sample clock (f_{SAMP_CLK}) for the protocol processor. 00 = f_{SAMP_CLK} = f_{DIV_CLK}. 01 = f_{SAMP_CLK} = f_{PROT_CLK}. 10 = f_{SAMP_CLK} = f_{SCLK}. 11 = f_{SAMP_CLK} = f_{REF_CLK}.
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source signal of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source signal of reference clock (f_{REF_CLK}). 0 = Peripheral device clock f_{PCLK}. 1 = Reserved.

USCI Line Control Register (UI2C_LINECTL)

Register	Offset	R/W	Description	Reset Value
UI2C_LINECTL	UI2Cn_BA+0x2C	R/W	USCI 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				**DWIDTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**LSB**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0x0: The data word contains 16 bits located at bit positions [15:0]. 0x1: Reserved. 0x2: Reserved. 0x3: Reserved. 0x4: The data word contains 4 bits located at bit positions [3:0]. 0x5: The data word contains 5 bits located at bit positions [4:0]. ... 0xF: The data word contains 15 bits located at bit positions [14:0]. **Note:** In I^2C protocol, the length must be configured as 8 bits.
[7:1]	**Reserved**	Reserved.
[0]	**LSB**	**LSB First Transmission Selection** 0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (UI2C_TXDAT)

Register	Offset	R/W	Description	Reset Value
UI2C_TXDAT	UI2Cn_BA+0x30	W	USCI Transmit 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**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 8-bit transmit data for transmission.

USCI Receive Data Register (UI2C_RXDAT)

Register	Offset	R/W	Description	Reset Value
UI2C_RXDAT	UI2Cn_BA+0x34	R	USCI 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**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer.

USCI Device Address Register (UI2C_DEVADDR)

Register	Offset	R/W	Description	Reset Value
UI2C_DEVADDR0	UI2Cn_BA+0x44	R/W	USCI Device Address Register 0	0x0000_0000
UI2C_DEVADDR1	UI2Cn_BA+0x48	R/W	USCI Device Address 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						DEVADDR	
7	6	5	4	3	2	1	0
DEVADDR							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**DEVADDR**	**Device Address** In I^{2}C protocol, this bit field contains the programmed slave address. If the first received address byte are 1111 0AAX$_B$, the AA bits are compared to the bits DEVADDR[9:8] to check for address match, where the X is R/W bit. Then the second address byte is also compared to DEVADDR[7:0]. **Note:** The DEVADDR [9:7] must be set 3'b000 when I^{2}C operating in 7-bit address mode.

USCI Device Address Mask Register (UI2C_ADDRMSK) – for I^{2}C Only

Register	Offset	R/W	Description	Reset Value
UI2C_ADDRMSK0	UI2Cn_BA+0x4C	R/W	USCI Device Address Mask Register 0	0x0000_0000
UI2C_ADDRMSK1	UI2Cn_BA+0x50	R/W	USCI Device Address Mask 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						**ADDRMSK**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDRMSK							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**ADDRMSK**	**USCI Device Address Mask** 0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don’t care.). USCI support multiple address recognition with two 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.

USCI Wake-up Control Register (UI2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_WKCTL	UI2Cn_BA+0x54	R/W	USCI 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						**WKADDREN**	**WKEN**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WKADDREN**	**Wake-up Address Match Enable Bit** 0 = The chip is woken up according data toggle. 1 = The chip is woken up according address match.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (UI2C_WKSTS)

Register	Offset	R/W	Description	Reset Value
UI2C_WKSTS	UI2Cn_BA+0x58	R/W	USCI 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							**WKF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – I^{2}C (UI2C_PROTCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_PROTCTL	UI2Cn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	Reserved					TOCNT	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TOCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						MONEN	SCLOUTEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		PTRG	ADDR10EN	STA	STO	AA	GCFUNC

Bits	Description	
[31]	**PROTEN**	**I^{2}C Protocol Enable Bit** 0 = I^{2}C Protocol disable. 1 = I^{2}C Protocol enable.
[30:26]	**Reserved**	Reserved.
[25:16]	**TOCNT**	**Time-out Clock Cycle** This bit field indicates how many clock cycle selected by TMCNTSRC (UI2C_BRGEN [5]) when each interrupt flags are clear. The time-out is enable when TOCNT bigger than 0. **Note:** The TMCNTSRC (UI2C_BRGEN [5]) must be set zero on I^{2}C mode.
[15:10]	**Reserved**	Reserved.
[9]	**MONEN**	**Monitor Mode Enable Bit** This bit enables monitor mode. In monitor mode the SDA output will be put in high impedance mode. This prevents the I^{2}C module from outputting data of any kind (including ACK) onto the I^{2}C data bus. 0 = The monitor mode is disabled. 1 = The monitor mode is enabled. **Note:** Depending on the state of the SCLOUTEN bit, the SCL output may be also forced high, preventing the module from having control over the I^{2}C clock line.
[8]	**SCLOUTEN**	**SCL Output Enable Bit** This bit enables monitor pulling SCL to low. This monitor will pull SCL to low until it has had time to respond to an I^{2}C interrupt. 0 = SCL output will be forced high due to open drain mechanism. 1 = I^{2}C module may act as a slave peripheral just like in normal operation, the I^{2}C holds the clock line low until it has had time to clear I^{2}C interrupt.
[7:6]	**Reserved**	Reserved.

[5]	**PTRG**	**I^{2}C Protocol Trigger** When a new state is present in the UI2C_PROTSTS register, if the related interrupt enable bits are set, the I^{2}C interrupt is requested. It must write one by software to this bit after the related interrupt flags are set to 1 and the I^{2}C protocol function will go ahead until the STOP is active or the PROTEN is disabled. 0 = I^{2}C's stretch disabled and the I^{2}C protocol function will go ahead. 1 = I^{2}C's stretch active.
[4]	**ADDR10EN**	**Address 10-bit Function Enable Bit** 0 = Address match 10 bit function is disabled. 1 = Address match 10 bit function is enabled.
[3]	**STA**	**I^{2}C START Control** Setting STA to logic 1 to enter Master mode, the I^{2}C hardware sends a START or repeat START condition to bus when the bus is free.
[2]	**STO**	**I^{2}C STOP Control** In Master mode, setting STO to transmit a STOP condition to bus then I^{2}C hardware will check the bus condition if a STOP condition is detected this bit will be cleared by hardware automatically. In a slave mode, setting STO resets I^{2}C hardware to the defined "not addressed" slave mode when bus error (UI2C_PROTSTS.ERRIF = 1).
[1]	**AA**	**Assert Acknowledge Control** When AA =1 prior to address or data received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.) A slave is acknowledging the address sent from master, 2.) The receiver devices are acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line.
[0]	**GCFUNC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled.

USCI Protocol Interrupt Enable Register – I^{2}C (UI2C_PROTIEN)

Register	Offset	R/W	Description	Reset Value
UI2C_PROTIEN	UI2Cn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**ACKIEN**	**Acknowledge Interrupt Enable Control** This bit enables the generation of a protocol interrupt if an acknowledge is detected by a master. 0 = The acknowledge interrupt is disabled. 1 = The acknowledge interrupt is enabled.
[5]	**ERRIEN**	**Error Interrupt Enable Control** This bit enables the generation of a protocol interrupt if an I^{2}C error condition is detected (indicated by ERR (UI2C_PROTSTS [16])). 0 = The error interrupt is disabled. 1 = The error interrupt is enabled.
[4]	**ARBLOIEN**	**Arbitration Lost Interrupt Enable Control** This bit enables the generation of a protocol interrupt if an arbitration lost event is detected. 0 = The arbitration lost interrupt is disabled. 1 = The arbitration lost interrupt is enabled.
[3]	**NACKIEN**	**Non - Acknowledge Interrupt Enable Control** This bit enables the generation of a protocol interrupt if a non - acknowledge is detected by a master. 0 = The non - acknowledge interrupt is disabled. 1 = The non - acknowledge interrupt is enabled.
[2]	**STORIEN**	**Stop Condition Received Interrupt Enable Control** This bit enables the generation of a protocol interrupt if a stop condition is detected. 0 = The stop condition interrupt is disabled. 1 = The stop condition interrupt is enabled.
[1]	**STARIEN**	**Start Condition Received Interrupt Enable Control** This bit enables the generation of a protocol interrupt if a start condition is detected.

		0 = The start condition interrupt is disabled. 1 = The start condition interrupt is enabled.
[0]	**TOIEN**	**Time-out Interrupt Enable Control** In I^{2}C protocol, this bit enables the interrupt generation in case of a time-out event. 0 = The time-out interrupt is disabled. 1 = The time-out interrupt is enabled.

USCI Protocol Status Register – I^{2}C (UI2C_PROTSTS)

Register	Offset	R/W	Description	Reset Value
UI2C_PROTSTS	UI2Cn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						WRSTSWK	WKAKDONE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SLAREAD	SLASEL	ACKIF	ERRIF	ARBLOIF	NACKIF	STORIF	STARIF
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	ONBUSY	TOIF	Reserved				

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**WRSTSWK**	**Read/Write Status Bit in Address Wakeup Frame** 0 = Write command be record on the address match wakeup frame. 1 = Read command be record on the address match wakeup frame.
[16]	**WKAKDONE**	**Wakeup Address Frame Acknowledge Bit Done** 0 = The ACK bit cycle of address match frame isn't done. 1 = The ACK bit cycle of address match frame is done in power-down. **Note:** This bit can't release when WKF is set.
[15]	**SLAREAD**	**Slave Read Request Status** This bit indicates that a slave read request has been detected. 0 = A slave read request has not been detected. 1 = A slave read request has been detected. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware.
[14]	**SLASEL**	**Slave Select Status** This bit indicates that this device has been selected as slave. 0 = The device is not selected as slave. 1 = The device is selected as slave. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware.
[13]	**ACKIF**	**Acknowledge Received Interrupt Flag** This bit indicates that an acknowledge has been received in master mode. A protocol interrupt can be generated if UI2C_PROTCTL.ACKIEN = 1. 0 = An acknowledge has not been received. 1 = An acknowledge has been received. It is cleared by software writing one into this bit
[12]	**ERRIF**	**Error Interrupt Flag** This bit indicates that 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. A protocol interrupt can be generated if UI2C_PROTCTL.ERRIEN = 1. 0 = An I^{2}C error has not been detected. 1 = An I^{2}C error has been detected. It is cleared by software writing one into this bit **Note:** This bit is set when slave mode, user must write one into STO register to the defined “not addressed” slave mode.
[11]	**ARBLOIF**	**Arbitration Lost Interrupt Flag** This bit indicates that an arbitration has been lost. A protocol interrupt can be generated if UI2C_PROTCTL.ARBLOIEN = 1. 0 = An arbitration has not been lost. 1 = An arbitration has been lost. It is cleared by software writing one into this bit
[10]	**NACKIF**	**Non - Acknowledge Received Interrupt Flag** This bit indicates that a non - acknowledge has been received in master mode. A protocol interrupt can be generated if UI2C_PROTCTL.NACKIEN = 1. 0 = A non - acknowledge has not been received. 1 = A non - acknowledge has been received. It is cleared by software writing one into this bit
[9]	**STORIF**	**Stop Condition Received Interrupt Flag** This bit indicates that a stop condition has been detected on the I^{2}C bus lines. A protocol interrupt can be generated if UI2C_PROTCTL.STORIEN = 1. 0 = A stop condition has not yet been detected. 1 = A stop condition has been detected. It is cleared by software writing one into this bit
[8]	**STARIF**	**Start Condition Received Interrupt Flag** This bit indicates that a start condition or repeated start condition has been detected on master mode. However, this bit also indicates that a repeated start condition has been detected on slave mode. A protocol interrupt can be generated if UI2C_PROTCTL.STARIEN = 1. 0 = A start condition has not yet been detected. 1 = A start condition has been detected. It is cleared by software writing one into this bit
[7]	**Reserved**	Reserved.
[6]	**ONBUSY**	**On Bus Busy** Indicates that a communication is in progress on the bus. It is set by hardware when a START condition is detected. It is cleared by hardware when a STOP condition is detected 0 = The bus is IDLE (both SCLK and SDA High). 1 = The bus is busy.
[5]	**TOIF**	**Time-out Interrupt Flag** 0 = A time-out interrupt status has not occurred. 1 = A time-out interrupt status has occurred. **Note:** It is cleared by software writing one into this bit
[0]	**Reserved**	Reserved.

USCI Slave match Address Register (UI2C_ADMAT)

Register	Offset	R/W	Description	Reset Value
UI2C_ADMAT	UI2Cn_BA+0x88	R/W	I^{2}C Slave Match 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						**ADMAT1**	**ADMAT0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**ADMAT1**	**USCI Address 1 Match Status Register** When address 1 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[0]	**ADMAT0**	**USCI Address 0 Match Status Register** When address 0 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.

USCI Timing Configure Control Register (UI2C_TMCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_TMCTL	UI2Cn_BA+0x8C	R/W	I^2C Timing Configure 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				HTCTL			
7	6	5	4	3	2	1	0
HTCTL		STCTL					

Bits	Description	
[31:8]	**Reserved**	Reserved.
[11:6]	**HTCTL**	**Hold Time Configure Control Register** This field is used to generate the delay timing between SCL falling edge SDA edge in transmission mode. The delay hold time is numbers of peripheral clock = HTCTL x f_{PCLK}.
[5:0]	**STCTL**	**Setup Time Configure Control Register** This field is used to generate a delay timing between SDA edge and SCL rising edge in transmission mode.. The delay setup time is numbers of peripheral clock = STCTL x f_{PCLK}.

6.16 UART Interface Controller (UART)

6.16.1 Overview

The NUC121/125 series provides one channel 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 ten types of interrupts. The UART controller also supports IrDA SIR, LIN and RS-485 function modes and auto-baud rate measuring function.

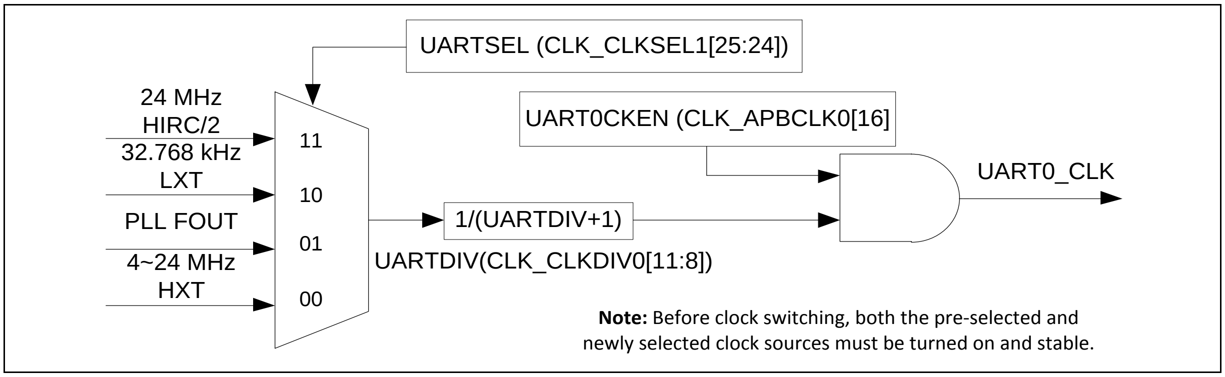
6.16.2 Features

- Full-duplex asynchronous communications
- Supports maximum clock frequency up to 10 Mbps
- Separates receive and transmit 16/16 bytes entry FIFO for data payloads
- Supports hardware auto-flow control
- Programmable receiver buffer trigger level
- Supports programmable baud rate generator for each channel individually
- Supports nCTS, incoming data, Received Data FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function
- Supports 8-bit receiver buffer time-out detection function
- Programmable transmitting data delay time between the last stop and the next start bit by setting DLY (UART_TOUT [15:8])
- Supports Auto-Baud Rate measurement and baud rate compensation function
- Supports break error, frame error, parity error and receive/transmit buffer overflow detection function
- Fully programmable serial-interface characteristics
  - Programmable number of data bit, 5-, 6-, 7-, 8- bit character
  - Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
  - Programmable stop bit, 1, 1.5, or 2 stop bit generation
- Supports IrDA SIR function mode
  - Supports for 3/16 bit duration for normal mode
- Supports LIN function mode
  - Supports LIN master/slave mode
  - Supports programmable break generation function for transmitter
  - Supports break detection function for receiver
- Supports RS-485 function mode
  - Supports RS-485 9-bit mode
  - Supports hardware or software enables to program nRTS pin to control RS-485 transmission direction
- Supports PDMA transfer function

6.16.3 Block Diagram

The UART clock control and block diagram are shown in Figure 6.16-1 and Figure 6.16-2 respectively.

Note: The frequency of UARTx_CLK should not be greater than 30 times HCLK.


UARTSEL (CLK_CLKSEL1[25:24])
24 MHz
HIRC/2
32.768 kHz
LXT
PLL FOUT
4~24 MHz
HXT
11
10
01
00
UART0CKEN (CLK_APBCLK0[16]
1/(UARTDIV+1)
UARTDIV(CLK_CLKDIV0[11:8])
UART0_CLK

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

Figure 6.16-1 UART Clock Control Diagram

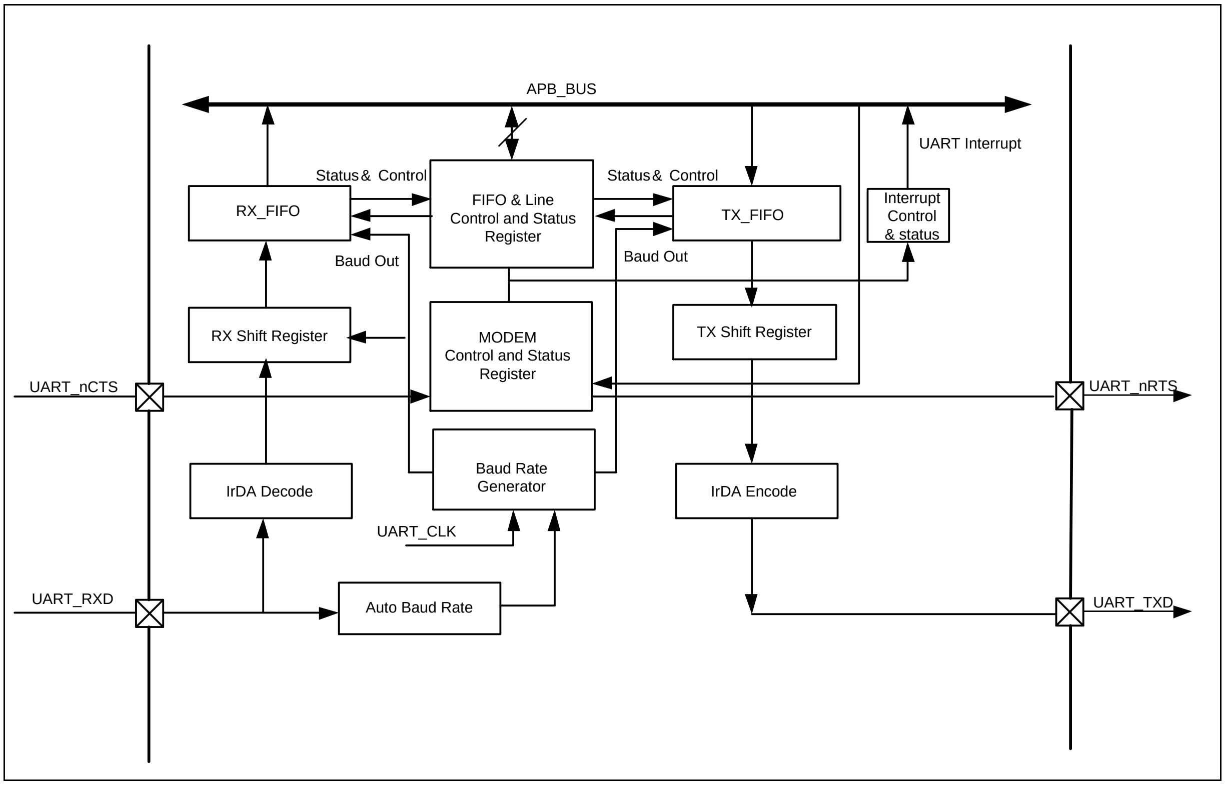


Figure 6.16-2 UART Block Diagram

Each block is described in detail as follows:

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

RX_FIFO

The receiver is buffered with a 16 bytes FIFO (plus three error bits, BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]), PEF (UART_FIFOSTS[4])) 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 encoding control block.

IrDA Decode

This block is IrDA decoding control block.

FIFO & Line Control and Status Register

This field is register set that including the FIFO control register (UART_FIFO), FIFO status register (UART_FIFOSTS), and line control register (UART_LINE) for transmitter and receiver. The time-out register (UART_TOUT) identifies the condition of time-out interrupt.

Auto-Baud Rate Measurement

This block is responsible for auto-baud rate measurement.

Interrupt Control and Status Register

There are ten types of interrupts, Receive Data Available Interrupt (RDAINT), Transmit Holding Register Empty Interrupt (THERINT), Transmitter Empty Interrupt (TXENDINT), Receive Line Status Interrupt (parity error or framing error or break interrupt) (RLSINT), MODEM Status Interrupt (MODEMINT), Receiver Buffer Time-out Interrupt (RXTOINT), Buffer Error Interrupt (BUFERRINT), LIN Bus Interrupt (LININT), Wake-up Interrupt (WKINT) and Auto-Baud Rate Interrupt (ABRINT). Interrupt enable register (UART_INTEN) enable or disable the responding interrupt and interrupt status register (UART_INTSTS) identifying the occurrence of the responding interrupt.

Interrupt	Description
RDAINT	Receive Data Available Interrupt.
THERINT	Transmit Holding Register Empty Interrupt.
TXENDINT	Transmitter Empty Interrupt.
RLSINT	Receive Line Status Interrupt (parity error or frame error or break error).
MODEMINT	MODEM Status Interrupt.
RXTOINT	Receiver Buffer Time-out Interrupt.
BUFERRINT	Buffer Error Interrupt.
LININT	LIN Bus Interrupt.
WKINT	Wake-up Interrupt.
ABRINT	Auto-Baud Rate Interrupt.

Table 6.16-1 UART Interrupt

6.16.4 Basic Configuration

The UART controller function pins are configured in SYS_GPA_MFPH, SYS_GPB_MFPL, SYS_GPB_MFPH, SYS_GPC_MFPL, SYS_GPD_MFPL and SYS_GPF_MFPL Multiple Function Registers.

The UART controller clock are enabled in UART0CKEN (CLK_APBCLK0[16]) for UART0.

The UART controller clock source is selected by UARTSEL (CLK_CLKSEL1[25:24]).

The UART controller clock pre-scale is determined by UARTDIV (CLK_CLKDIV0[11:8]).

UART Interface Controller Pin description is shown in Table 6.16-2:

Pin	Type	Description
UART_TXD	Output	UART transmit
UART_RXD	Input	UART receive
UART_nCTS	Input	UART modem clear to send
UART_nRTS	Output	UART modem request to send

Table 6.16-2 UART Interface Controller Pin

6.16.5 Functional Description

The UART controller supports four function modes including UART, IrDA, LIN and RS-485 mode. User can select a function by setting the UART_FUNCSEL register. The four function modes will be described in following section.

6.16.5.1 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. Table 6.16-3 list the

UART baud rate equations in the various conditions. Table 6.16-4 and Table 6.16-5 list the UART baud rate parameter and register setting example. In IrDA function mode, the baud rate generator must be set in Mode 0. More detail register description is shown in UART_BAUD register. There are three setting mode. Mode 0 is set by UART_BAUD[29:28] with 00. Mode 1 is set by UART_BAUD[29:28] with 10. Mode 2 is set by UART_BAUD[29:28] with 11.

Mode	BAUDM1	BAUDM0	Baud Rate Equation
Mode 0	0	0	UART_CLK / [16 * (BRD+2)].
Mode 1	1	0	UART_CLK / [(EDIVM1+1) * (BRD+2)], EDIVM1 must >= 8.
Mode 2	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. (If the UART_CLK is selected from LXT, BRD can be greater than or equal to 1)

Table 6.16-3 UART controller Baud Rate Equation Table

UART Peripheral Clock = 24 MHz			
Baud Rate	**Mode 0**	**Mode 1**	**Mode 2**
921600	Not support	BRD=0, EDIVM1=12	BRD=24
460800	Not recommended	BRD=2, EDIVM1 =12	BRD=50
230400	Not recommended	BRD=6, EDIVM1 =12	BRD =102
115200	BRD =11	BRD=14, EDIVM1 =12	BRD =206
57600	BRD =24	BRD=30, EDIVM1 =12	BRD =415
38400	BRD =37	BRD=50, EDIVM1 =11	BRD =623
19200	BRD =76	BRD=123, EDIVM1 =9	BRD =1248
9600	BRD =154	BRD=248, EDIVM1 =9	BRD =2498
4800	BRD =310	BRD=498, EDIVM1 =9	BRD =4998
Note: There are ±2.5% tolerance in this table.			

Table 6.16-4 UART controller Baud Rate Parameter Setting Example Table

UART Peripheral Clock = 24 MHz			
Baud Rate	**UART_BAUD Value**		
	Mode 0	**Mode 1**	**Mode 2**
921600	Not support	0x2C00_0000	0x3000_0018
460800	Not recommended	0x2C00_0002	0x3000_0032
230400	Not recommended	0x2C00_0006	0x3000_0066

115200	0x0000_000B	0x2C00_000E	0x3000_00CE
57600	0x0000_0018	0x2C00_001E	0x3000_019F
38400	0x0000_0025	0x2B00_0032	0x3000_026F
19200	0x0000_004C	0x2900_007B	0x3000_04E0
9600	0x0000_009A	0x2900_00F8	0x3000_09C2
4800	0x0000_0136	0x2900_01F2	0x3000_1386
Note: There are ±2.5% tolerance in this table.			

Table 6.16-5 UART controller Baud Rate Register Setting Example Table

6.16.5.2 UART Controller Baud Rate Compensation

The UART controller supports baud rate compensation function. It is used to optimize the precision in each bit. The precision of the compensation is half of UART module clock because there is BRCOMDEC bit (UART_BRCOMP[31]) to define the positive or negative compensation in each bit. If the BRCOMPDEC (UART_BRCOMP[31]) = 0, it is positive compensation for each bit, one more module clock will be append in the compensated bit. If the BRCOMPDEC (UART_BRCOMP[31]) = 1, it is negative compensation for each bit, decrease one module clock in the compensated bit.

There is 9-bits location, BRCOMP[8:0] (UART_BRCOMP[8:0]), can be configured by user to define the relative bit is compensated or not. BRCOMP[7:0] is used to define the compensation of UART_DAT[7:0] and BRCOMP[8] is used to define the parity bit.

Example:

(1). UART's peripheral clock = 32.768K and baud rate is 9600

Baud rate is 9600, UART peripheral clock is 32.768K → 3.413 peripheral clock/bit

if the baud divider is set 1 (3 peripheral clock/bit), the inaccuracy of each bit is -0.413 peripheral clock and BRCOMPDEC =0,

Bit	**Name**	**Total INACCURACY**	**BRCOMP Compensated**	**Final Inaccuracy**
0	Start	-0.413	x	-0.413
1	UART_DAT[0]	-0.826(-0.413-0.413)	1	0.174
2	UART_DAT[1]	-0.239(0.174-0.413)	0	-0.239
3	UART_DAT[2]	-0.652(-0.239-0.413)	1	0.348
4	UART_DAT[3]	-0.065(0.348-0.413)	0	-0.065
5	UART_DAT[4]	-0.478(-0.065-0.413)	0	-0.478
6	UART_DAT[5]	-0.891(-0.478-0.413)	1	0.109
7	UART_DAT[6]	-0.304(0.109-0.413)	0	-0.304
8	UART_DAT[7]	-0.717(-0.304-0.413)	1	0.283
9	Parity	-0.130(0.283-0.413)	0	-0.13

Table 6.16-6 Baud Rate Compensation Example Table 1

Thus, the BRCOMP (UART_BRCOMP[8:0]) can be set as 9’b010100101 = 0xa5.

(2). UART’s peripheral clock = 32.768K and baud rate is 4800

Baud rate is 4800, UART peripheral clock is 32.768K → 6.827 peripheral clock/bit

if the baud divider is set 5 (7 peripheral clock/bit), the inaccuracy of each bit is 0.173 peripheral clock and BRCOMPDEC =1,

Bit	Name	Total INACCURACY	BRCOMP Compensated	Final Inaccuracy
0	Start	0.173	x	0.173
1	UART_DAT[0]	0.346(0.173+0.173)	0	0.346
2	UART_DAT[1]	0.519(0.346+0.173)	1	-0.481
3	UART_DAT[2]	-0.308(-0.481+0.173)	0	-0.308
4	UART_DAT[3]	-0.135(-0.308+0.173)	0	-0.135
5	UART_DAT[4]	-0.038(-0.135+0.173)	0	0.038
6	UART_DAT[5]	0.211(0.038+0.173)	0	0.211
7	UART_DAT[6]	0.384(0.211+0.173)	0	0.384
8	UART_DAT[7]	0.557(0.384+0.173)	1	-0.443
9	Parity	-0.270(-0.443+0.173)	0	-0.270

Table 6.16-7 Baud Rate Compensation Example Table 2

Thus, the BRCOMP (UART_BRCOMP[8:0]) can be set as 9’b010000010 = 0x82.

6.16.5.3 UART Controller Auto-Baud Rate Function Mode

Auto-Baud Rate function can measure baud rate of receiving data from UART RX pin automatically. When the Auto-Baud Rate measurement is finished, the measuring baud rate is loaded to BRD (UART_BAUD[15:0]). Both of the BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) are set to 1 automatically. UART RX data from Start bit to 1st rising edge time is set by 2^{ABRDBITS} bit time in Auto-Baud Rate function detection frame.

2^{ABRDBITS} bit time from Start bit to the 1st rising edge is calculated by setting ABRDBITS (UART_ALTCTL[20:19]). Setting ABRDEN (UART_ALTCTL[18]) is to enable auto-baud rate function. In beginning stage, the UART RX is kept at 1. Once falling edge is detected, START bit is received. The auto-baud rate counter is reset and starts counting. The auto-baud rate counter will be stop when the 1 st rising edge is detected. Then, auto-baud rate counter value divided by ABRDBITS (UART_ALTCTL[20:19]) is loaded to BRD (UART_BAUD[15:0]) automatically. ABRDEN (UART_ALTCTL[18]) is cleared. Once the auto-baud rate measurement is finished, the ABRDIF (UART_FIFOSTS[1]) is set. When auto-baud rate counter is overflow, ABRDTOIF (UART_FIFOSTS[2]) is set. ABRDIF (UART_FIFOSTS[1]) or ABRDTOIF (UART_FIFOSTS[2]) cause the auto-baud rate flag ABRIF(UART_ALTCTL[17]) is generated. If the ABRIEN (UART_INTEN[18]) is enabled, ABRIF(UART_ALTCTL[17]) cause the auto-baud rate interrupt ABRINT (UART_INTSTS[31]) is generated.

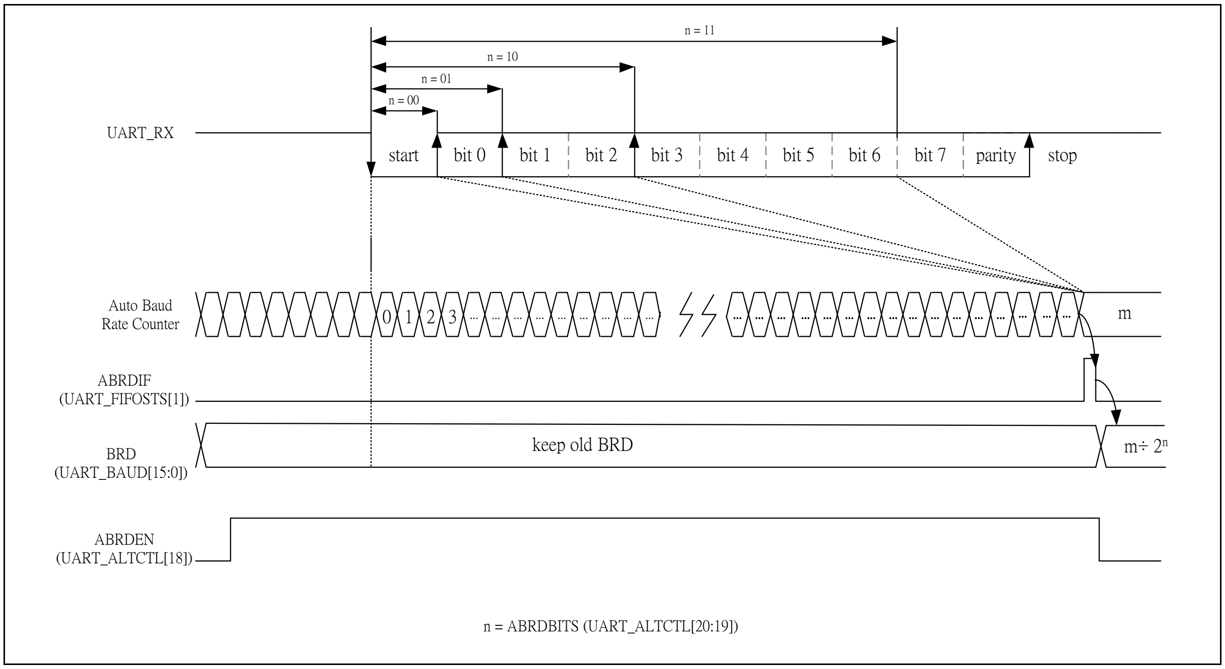


Figure 6.16-3 Auto-Baud Rate Measurement

Programming Sequence Example:

1. Program ABRDBITS (UART_ALTCTL[20:19]) to determines UART RX data 1st rising edge time from Start by $2^{ABRDBITS}$ bit time.
2. Set ABRIEN (UART_INTEN[18]) to enable auto-baud rate function interrupt.
3. Set ABRDEN (UART_ALTCTL[18]) to enable auto-baud rate function.
4. ABRDIF (UART_FIFOSTS[1]) is set, the auto-baud rate measurement is finished.
5. Operate UART transmit and receive action.
6. ABRDTOIF (UART_FIFOSTS[2]) is set, if auto-baud rate counter is overflow.
7. Go to Step 3.

6.16.5.4 UART Controller Transmit Delay Time Value

The UART controller programs DLY (UART_TOUT [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 Figure 6.16-4.

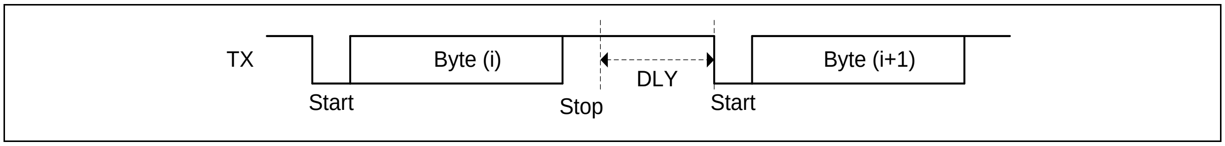


Figure 6.16-4 Transmit Delay Time Operation

6.16.5.5 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, LIN and RS-485 mode support FIFO control and status function.

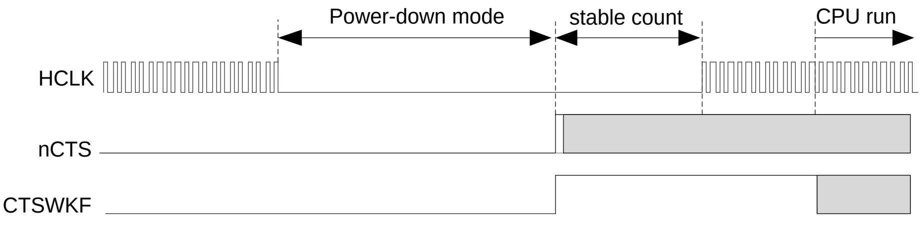
6.16.5.6 UART Controller Wake-up Function

The UART controller supports wake-up system function. The wake-up function includes nCTS pin, incoming data wake-up, Received Data FIFO reached threshold wake-up, RS-485 Address Match (AAD mode) wake-up and Received Data FIFO threshold time-out wake-up function. CTSWKF (UART_WKSTS[0]), DATWKF (UART_WKSTS[1]), RFRTWKF (UART_WKSTS[2]), RS485WKF (UART_WKSTS[3]) or TOUTWKF (UART_WKSTS[4]) cause the wake-up interrupt flag WKIF(UART_INTSTS[6]) is generated. If the WKIEN (UART_INTEN[6]) is enabled, the wake-up interrupt flag WKIF(UART_INTSTS[6]) cause the wake-up interrupt WKINT (UART_INTSTS[14]) is generated.

nCTS pin wake-up :

When the system is in Power-down mode and WKCTSEN (UART_WKCTL[0]) is set, the toggle of nCTS pin can wake-up system. If the WKCTSEN (UART_WKCTL[0]) is enabled, the toggle of nCTS pin cause the nCTS wake-up flag CTSWKF (UART_WKSTS[0]) is generated.

nCTS Wake-up Case 1 (nCTS transition from low to high)

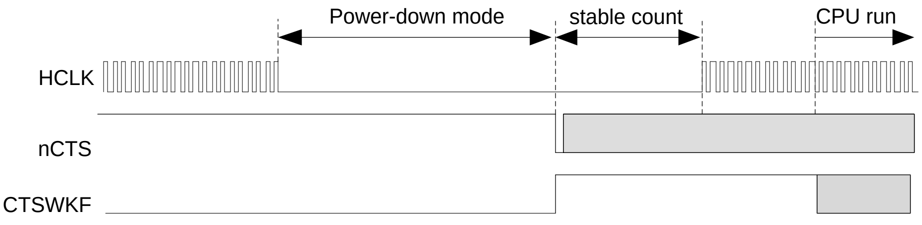


CTSACTLV (UART_MODEMSTS[8]) = 0

Note: Stable count means HCLK source recovery stable count.

Figure 6.16-5 UART nCTS Wake-up Case1

nCTS Wake-up Case 2 (nCTS transition from high to low)



CTSACTLV (UART_MODEM[8]) = 1

Note: Stable count means HCLK source recovery stable count.

Figure 6.16-6 UART nCTS Wake-up Case2

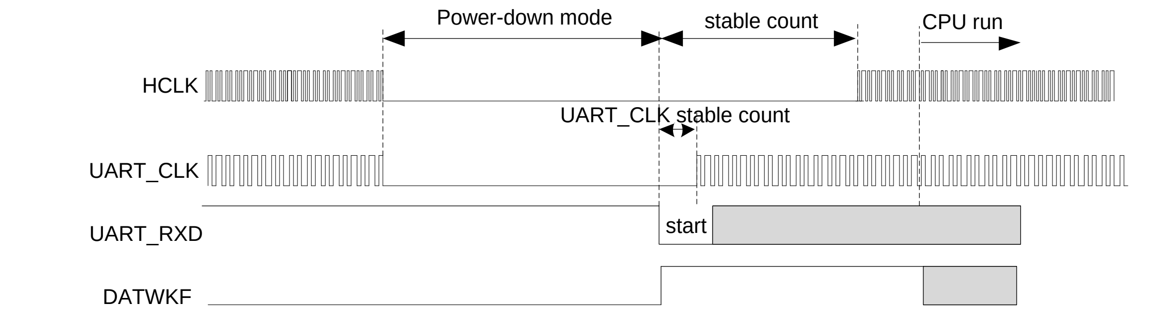
Incoming data wake-up:

When system is in Power-down mode and the WKDATEN (UART_WKCTL [1]) is set, the toggle of incoming data (UART_RXD) pin can wake-up the system. In order to receive the incoming data after the system wake-up, the STCOMP (UART_DWKCOMP[15:0]) shall be set. These bits field of STCOMP indicate how many clock cycle selected by UART_CLK do the UART controller can get the 1st bit (start bit) when the system is wakeup from power-down mode.

When incoming data wakes system up, the incoming data will be received and stored in FIFO. If the WKDATEN (UART_WKCTL[1]) is enabled, the toggle of incoming data (UART_RXD) pin cause the incoming data wake-up flag DATWKF (UART_WKSTS[1]) is generated.

Note1: The UART controller clock source should be selected as HIRC and the compensation time for start bit is about 20.32us. It means that the value of STCOMP (UART_DWKCOMP[15:0]) can be set as 488.

Note2: The value of BRD(UART_BAUD[15:0]) should be greater than STCOMP (UART_DWKCOMP[15:0]).



Note1: Stable count means HCLK source recovery stable count.

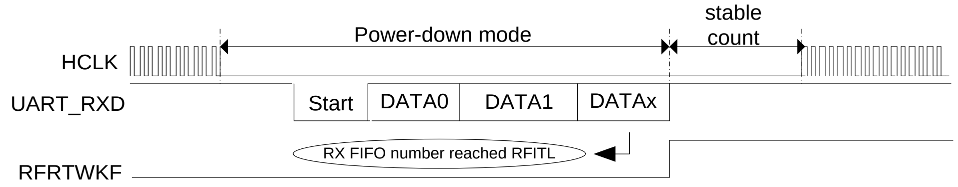
Note2: UART_CLK stable count means UART clock source recovery stable count.

Figure 6.16-7 UART Data Wake-up

Received Data FIFO reached threshold wake-up:

The received data FIFO threshold reached wake-up function is enabled by setting WKRFRTEN (UART_WKCTL[2]). In Power-down mode, when the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]), it can wake-up the system. If the WKRFRTEN (UART_WKCTL[2]) is enabled, the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]) cause the received data FIFO reached threshold wake-up flag RFRTWKF (UART_WKSTS[2]) is generated.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



Note: Stable count means HCLK source recovery stable count.

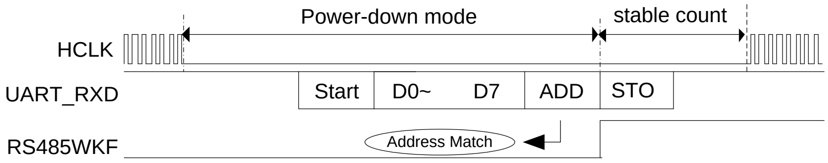
Figure 6.16-8 UART Received Data FIFO reached threshold wake-up

RS-485 Address Match (AAD mode) wake-up:

The RS-485 address match wake-up function is enabled by setting WKRS485EN (UART_WKCTL[3]). This function is used for RS-485 Auto Address Detection (AAD) mode in RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1. In Power-down mode, when an address byte is detected and matches the ADDRMV (UART_ALTCTL[31:24]), it can wake-up the system. If the WKRS485EN (UART_WKCTL[3]) is enabled, when an address byte is detected and matches the ADDRMV (UART_ALTCTL[31:24]) that cause the RS485 address match (AAD mode) wake-up flag RS485WKF (UART_WKSTS[3]) is generated.

Note1: The UART controller clock source should be selected as LXT in Power-down mode to receive data.

Note2: The UART controller can not receive any data after the address byte until HCLK source clock is stable. Since HCLK needs to wait for a stable time to recover source clock from Power-down mode, the incoming data transmission during the stable time will cause loss and error of the address byte.



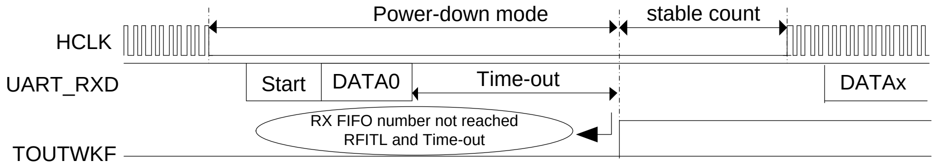
Note: Stable count means HCLK source recovery stable count.

Figure 6.16-9 UART RS-485 AAD Mode Address Match Wake-up

Received Data FIFO threshold time-out wake-up:

The received data FIFO threshold time-out wake-up function is enabled by setting WKRFRTEN (UART_WKCTL[2]) and WKTOUTEN (UART_WKCTL[4]). Setting TOCNTEN (UART_INTEN[11]) to enable receiver buffer time-out counter. In Power-down mode, when the number of received data in RX FIFO does not reach the threshold value RFITL (UART_FIFO[7:4]) and the time-out counter equals to the time-out value TOIC (UART_TOUT[7:0]), it can wake-up the system. If the WKTOUTEN (UART_WKCTL[4]) is enabled, when the time-out counter equals to the time-out value TOIC (UART_TOUT[7:0]) that cause the Received Data FIFO threshold time-out wake-up wake-up flag TOUTWKF (UART_WKSTS[4]) is generated.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



Note: Stable count means HCLK source recovery stable count.

Figure 6.16-10 UART Received Data FIFO threshold time-out wake-up

6.16.5.7 UART Controller Interrupt and Status

Each UART controller supports ten types of interrupts including:

- Receive Data Available Interrupt (RDAINT)

- Transmit Holding Register Empty Interrupt (THERINT)
- Transmitter Empty Interrupt (TXENDIF)
- Receive Line Status Interrupt (RLSINT)
  - Break Interrupt Flag (BIF)
  - Framing Error Flag (FEF)
  - Parity Error Flag (PEF)
  - RS-485 Address Byte Detect Flag (ADDRDETF)
- MODEM Status Interrupt (MODEMINT)
  - Detect nCTS State Change Flag (CTSDETF)
- Receiver Buffer Time-out Interrupt (RXTOINT)
- Buffer Error Interrupt (BUFERRINT)
  - TX Overflow Error Interrupt Flag (TXOVIF)
  - RX Overflow Error Interrupt Flag (RXOVIF)
- LIN Bus Interrupt (LININT)
  - LIN Break Detection Flag (BRKDETF)
  - Bit Error Detect Status Flag (BITEF)
  - LIN Slave ID Parity Error Flag (SLVIDPEF)
  - LIN Slave Header Error Flag (SLVHEF)
  - LIN Slave Header Detection Flag (SLVHDETF)
- Wake-up Interrupt (WKINT)
  - nCTS Wake-up Flag (CTSWKF)
  - Incoming Data Wake-up Flag (DATWKF)
  - Received Data FIFO Reached Threshold Wake-up Flag (RFRTWKF)
  - RS-485 Address Match (AAD mode) Wake-up Flag (RS485WKF)
  - Received Data FIFO Threshold Time-out Wake-up Flag (TOUTWKF)
- Auto-Baud Rate Interrupt (ABRINT)
  - Auto-baud Rate Detect Interrupt Flag (ABRDIF)
  - Auto-baud Rate Detect Time-out Interrupt Flag (ABRDTOIF)

Table 6.16-8 describes the interrupt sources and flags. The interrupt is generated when the interrupt flag is generated and the interrupt enable bit is set. User must clear the interrupt flag after the interrupt is generated.

Interrupt Source	**Interrupt Indicator**	**Interrupt Enable Bit**	**Interrupt Flag**	**Flag Caused By**	**Flag Cleared By**
Receive Data Available Interrupt	RDAINT	RDAIEN	**RDAIF**	N/A	Read UART_DAT
Transmit Holding Register Empty Interrupt	THERINT	THREIEN	**THREIF**	N/A	Write UART_DAT
Transmitter Empty Interrupt	TXENDINT	TXENDIEN	**TXENDIF**	N/A	Write UART_DAT

Receive Line Status Interrupt	RLSINT	RLSIEN	**RLSIF**	**RLSIF** = BIF	**Write ‘1’ to** BIF
				RLSIF = FEF	**Write ‘1’ to** FEF
				RLSIF = PEF	**Write ‘1’ to** PEF
				RLSIF = ADDRDETF	**Write ‘1’ to** ADDRDETF
Modem Status Interrupt	MODEMINT	MODEMIEN	**MODEMIF**	**MODEMIF** = CTSDETF	**Write ‘1’ to** CTSDETF
Receiver Buffer Time-out Interrupt	RXTOINT	RXTOIEN	**RXTOIF**	N/A	Read UART_DAT
Buffer Error Interrupt	BUFERRINT	BUFERRIEN	**BUFERRIF**	**BUFERRIF** = TXOVIF	**Write ‘1’ to** TXOVIF
				BUFERRIF = RXOVIF	**Write ‘1’ to** RXOVIF
LIN Bus Interrupt	LININT	LINIEN	**LINIF**	**LINIF =** BRKDETF	**Write ‘1’ to** LINIF **and** **Write ‘1’ to** BRKDETF
				LINIF = BITEF	**Write ‘1’ to** LINIF **and** **Write ‘1’ to** BITEF
				LINIF = SLVIDPEF	**Write ‘1’ to** LINIF **and** **Write ‘1’ to** SLVIDPEF
				LINIF = SLVHEF	**Write ‘1’ to** LINIF **and** **Write ‘1’ to** SLVHEF
				LINIF = SLVHDETF	**Write ‘1’ to** LINIF **and** **Write ‘1’ to** SLVHDETF
Wake-up Interrupt	WKINT	WKIEN	**WKIF**	**WKIF =** CTSWKF	**Write ‘1’ to** CTSWKF
				WKIF = DATWKF	**Write ‘1’ to** DATWKF
				WKIF = RFRTWKF	**Write ‘1’ to** RFRTWKF
				WKIF = RS485WKF	**Write ‘1’ to** RS485WKF
				WKIF = TOUTWKF	**Write ‘1’ to** TOUTWKF
Auto-Baud Rate Interrupt	ABRINT	ABRIEN	**ABRIF**	**ABRIF =** ABRDIF	**Write ‘1’ to** ABRDIF
				ABRIF = ABRDTOIF	**Write ‘1’ to** ABRDTOIF

Table 6.16-8 UART controller Interrupt Source and Flag List

6.16.5.8 UART Function Mode

The UART controller provides UART function (Setting FUNCSEL (UART_FUNCSEL [1:0]) to ’00’ to enable UART function mode). The UART baud rate is up to 1 Mbps.

The UART provides full-duplex and asynchronous communications. The transmitter and receiver contain 16 bytes FIFO for payloads. User can program receiver buffer trigger level and receiver buffer time-out detection for receiver. The transmitting data delay time between the last stop and the next start bit can be programed by setting DLY (UART_TOUT [15:8]) register. The UART supports hardware auto-flow control that provides programmable nRTS flow control trigger level. The number of data bytes in RX FIFO is equal to or greater than RTSTRGLV (UART_FIFO[19:16]), the nRTS is de-asserted.

UART Line Control Function

The UART controller supports fully programmable serial-interface characteristics by setting the UART_LINE register. User can program UART_LINE register for the word length, stop bit and parity bit setting. Table 6.16-9 and Table 6.16-10 list the UART word, stop bit length and the parity bit settings.

NSB (UART_LINE[2])	WLS (UART_LINE[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 6.16-9 UART Line Control of Word and Stop Length Setting

Parity Type	SPE (UART_LINE[5])	EPE (UART_LINE[4])	PSS (UART_LINE[7])	PBE (UART_LINE[3])	Description
No Parity	x	x	x	0	No parity bit output.
Parity source from UART_DAT	x	x	1	1	Parity bit is generated and checked by software.
Odd Parity	0	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	0	1	Even Parity is calculated by adding all the “1’s” in a data stream and adding a parity bit to the total bits, to make the count an even number.
Forced Mask Parity	1	0	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	1	1	0	1	Parity bit always logic 0.

Space Parity					Parity bit on the serial byte is set to “0” regardless of total number of “1’s” (even or odd counts).

Table 6.16-10 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 RTSTRGLV (UART_FIFO [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.

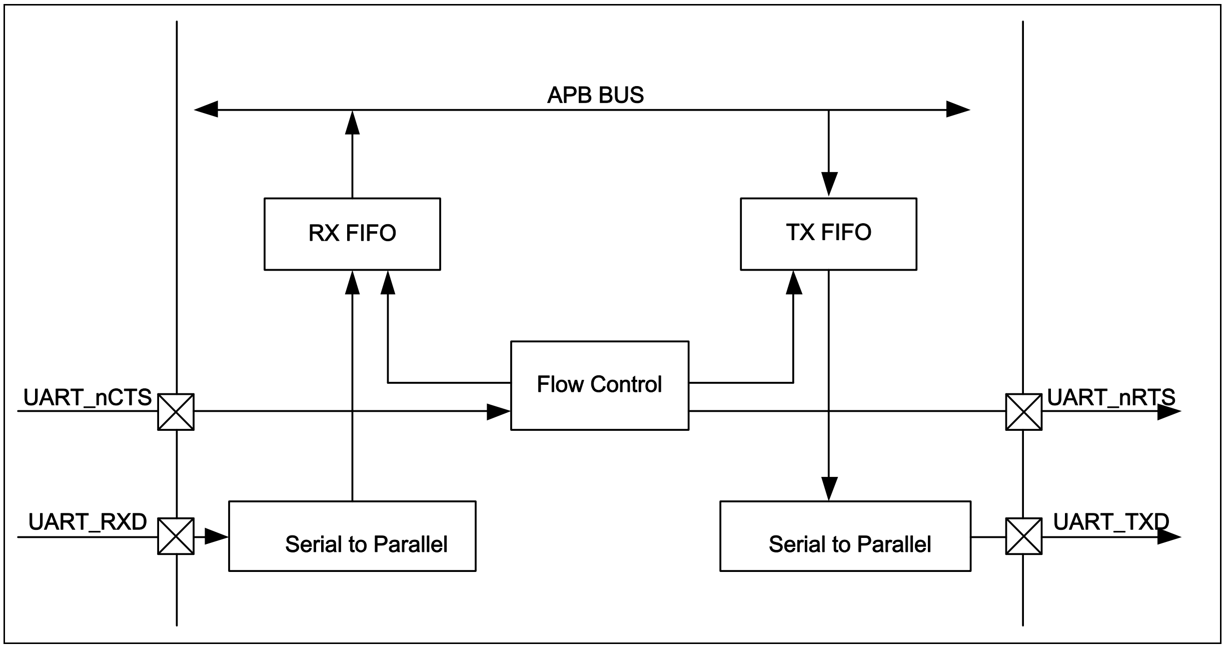


Figure 6.16-11 Auto-Flow Control Block Diagram

Figure 6.16-12 demonstrates the nCTS auto-flow control of UART function mode. User must set ATOCTSEN (UART_INTEN [13]) to enable nCTS auto-flow control function. The CTSACTLV (UART_MODEMSTS [8]) can set nCTS pin input active state. The CTSDETF (UART_MODEMSTS[0]) is set when any state change of nCTS pin input has occurred, and then TX data will be automatically transmitted from TX FIFO.

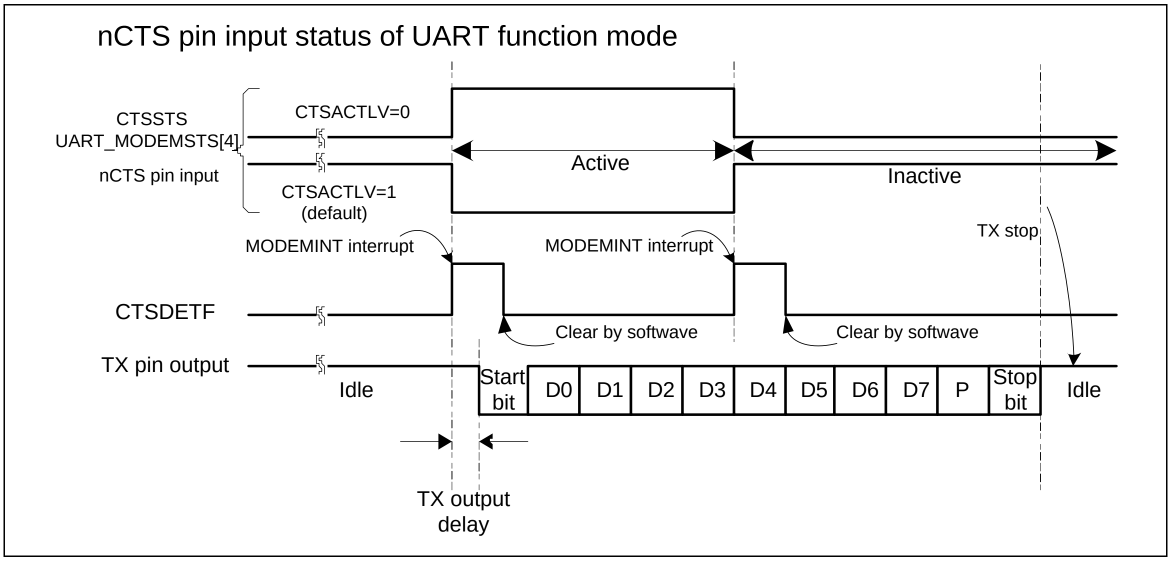


Figure 6.16-12 UART nCTS Auto-Flow Control Enabled

As shown in Figure 6.16-13, in UART nRTS auto-flow control mode (ATORTSEN(UART_INTEN[12])=1), the nRTS internal signal is controlled by UART FIFO controller with RTSTRGLV(UART_FIFO[19:16]) trigger level.

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

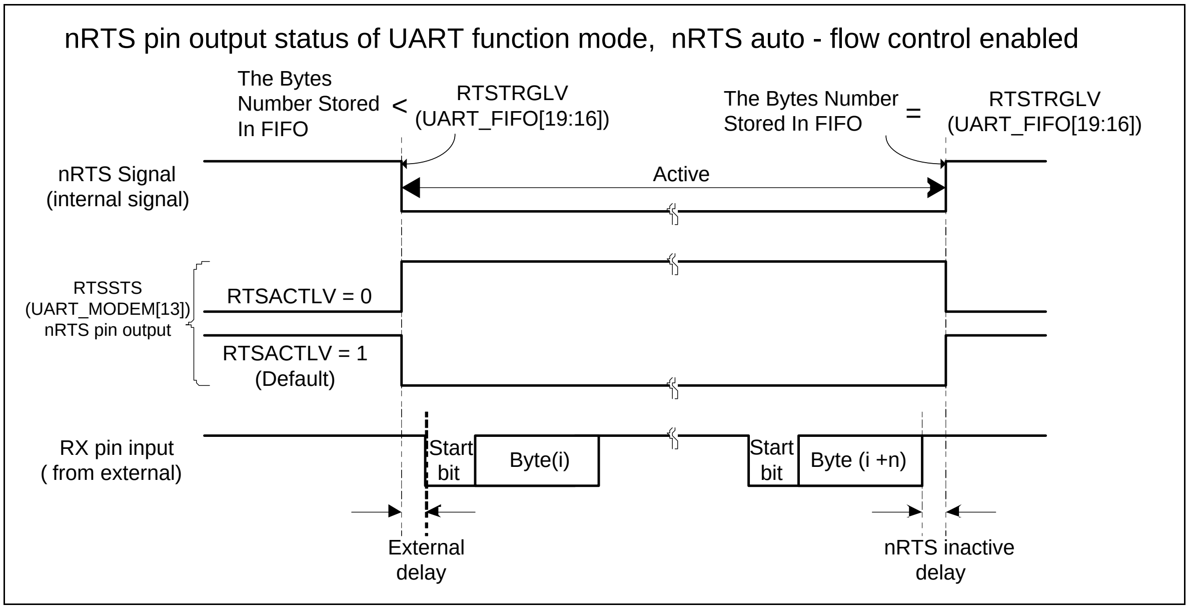

nRTS pin output status of UART function mode, nRTS auto - flow control enabled
The Bytes Number Stored In FIFO
<
RTSTRGLV (UART_FIFO[19:16])
The Bytes Number Stored In FIFO
=
RTSTRGLV (UART_FIFO[19:16])
nRTS Signal (internal signal)
Active
RTSSTS (UART_MODEM[13]) nRTS pin output
RTSACTLV = 0
RTSACTLV = 1 (Default)
RX pin input (from external)
Start bit
Byte(i)
Start bit
Byte (i +n)
External delay
nRTS inactive delay

Figure 6.16-13 UART nRTS Auto-Flow Control Enabled

As shown in Figure 6.16-14, in software mode (ATORTSEN(UART_INTEN[12])=0), the nRTS flow is directly controlled by software programming of RTS(UART_MODEM[1]) control bit.

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

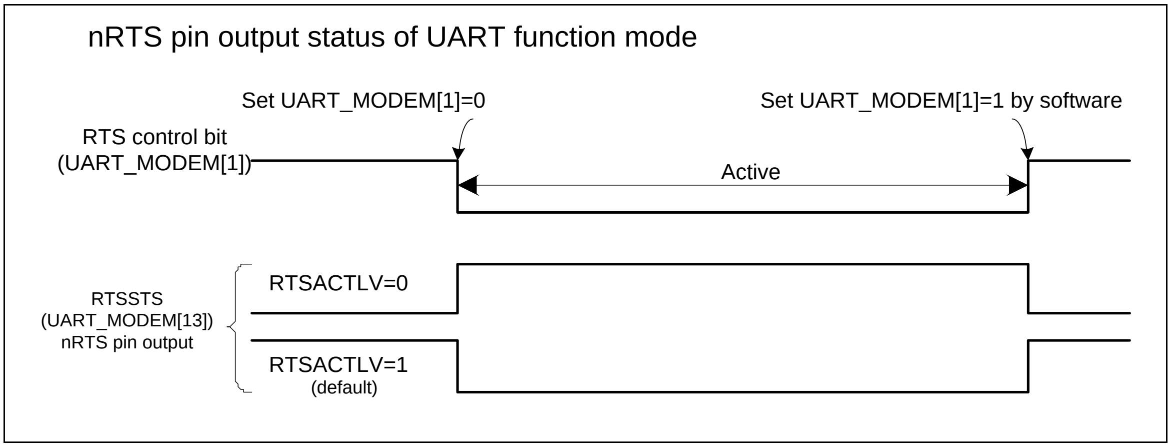


Figure 6.16-14 UART nRTS Auto-Flow with Software Control

6.16.5.9 IrDA Function Mode

The UART controller also provides Serial IrDA (SIR, Serial Infrared) function (Setting UART_FUNCSEL [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. Thus, 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 BAUDM1 (UART_BAUD [29]) must be cleared.

Baud Rate = Clock / (16 * (BRD +2)), where BRD (UART_BAUD[15:0]) is Baud Rate Divider in UART_BAUD register.

Note: The tolerance of baud-rate is ±5% between IrDA master and IrDA slave.

The IrDA control block diagram is shown in Figure 6.16-15.

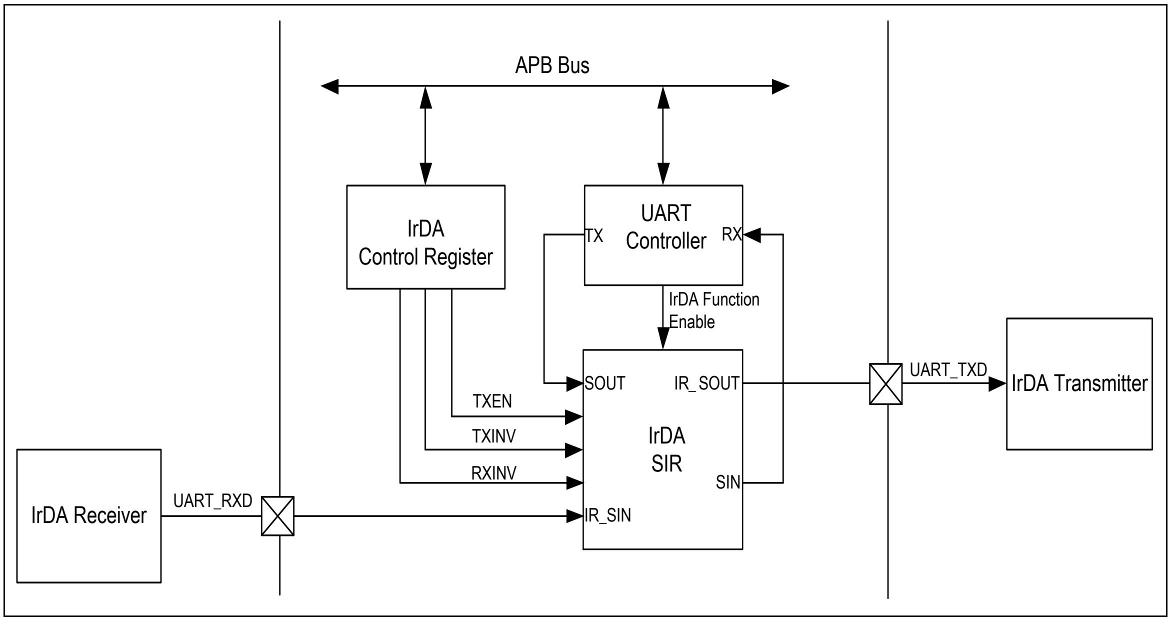


Figure 6.16-15 IrDA Control Block Diagram

IrDA SIR Transmit Encoder

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

The transmitted pulse width is specified as 3/16 period of baud rate.

IrDA SIR Receive Decoder

The IrDA SIR Receive Decoder demodulates the Return-to-Zero bit stream from the input detector and outputs the NRZ serial bits stream to the UART received data input.

In idle state, the decoder input is high. A start bit is detected when the decoder input is LOW. In normal operation, the RXINV (UART_IRDA[6]) is set to '1' and TXINV (UART_IRDA[5]) is set to '0'.

IrDA SIR Operation

The IrDA SIR encoder/decoder provides functionality which converts between UART data stream and half-duplex serial SIR interface. Figure 6.16-16 is IrDA encoder/decoder waveform.

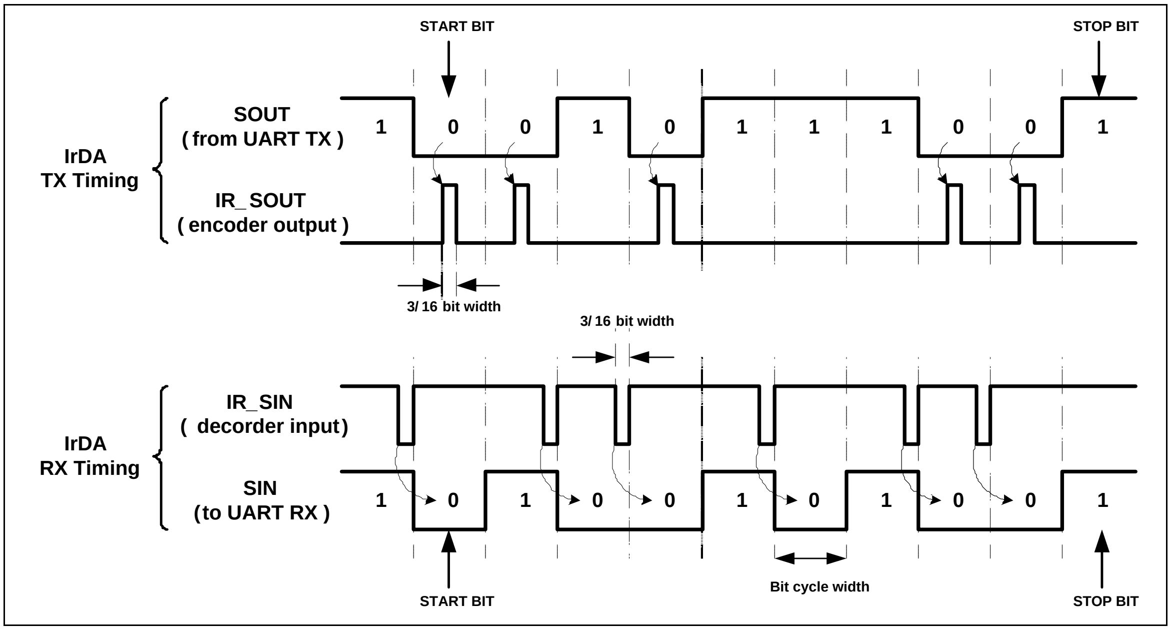


Figure 6.16-16 IrDA TX/RX Timing Diagram

6.16.5.10 LIN Function Mode (Local Interconnection Network)

The UART Controller supports LIN function. Setting FUNCSEL (UART_FUNCSEL[1:0]) to ’01’ to select LIN mode operation. The UART Controller supports LIN break/delimiter generation and break/delimiter detection in LIN master mode, and supports header detection and automatic resynchronization in LIN Slave mode.

Structure of LIN Frame

According to the LIN protocol, all information transmitted is packed as frames; a frame consists of a header (provided by the master task) and a response (provided by a slave task). The header (provided by the master task) consists of a break field and a sync field followed by a frame identifier (frame ID). The frame identifier uniquely defines the purpose of the frame. The slave task is appointed for providing the response associated with the frame ID. The response consists of a data field and a checksum field. Figure 6.16-17 is the structure of LIN Frame.

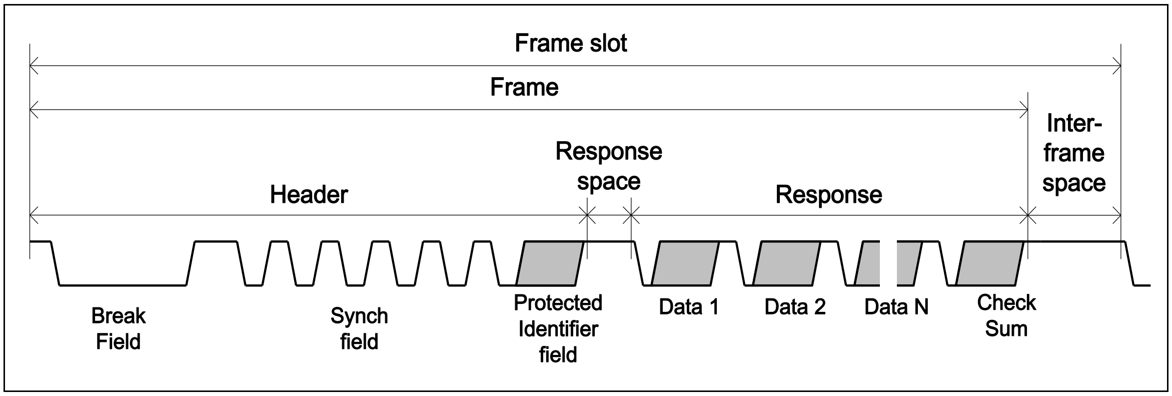


Figure 6.16-17 Structure of LIN Frame

Structure of LIN Byte

In LIN mode, each byte field is initiated by a START bit with value 0 (dominant), followed by 8 data bits and no parity bit, LSB is first and ended by 1 stop bit with value 1 (recessive) in accordance with the LIN standard. The structure of Byte is shown in Figure 6.16-18 .

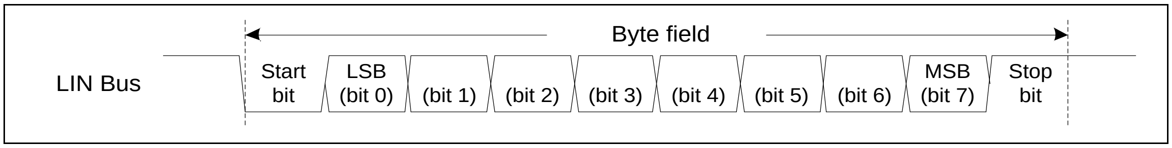


Figure 6.16-18 Structure of LIN Byte

LIN Master Mode

The UART Controller supports LIN Master mode. To enable and initialize the LIN Master mode, the following steps are necessary:

1. Setting the UART_BAUD register to select the desired baud rate.
2. Setting WLS (UART_LINE[1:0]) to ‘11’ to configure the word length with 8 bits, clearing PBE (UART_LINE[3]) bit to disable parity check and clearing NSB (UART_LINE[2]) bit to configure with one stop bit.
3. Setting FUNCSEL (UART_FUNCSEL[1:0]) to ‘01’ to select LIN function mode operation.

A complete header consists of a break field and sync field followed by a frame identifier (frame ID). The UART controller can be selected header sending by three header selected modes. The header selected mode can be “break field” or “break field and sync field” or “break field, sync field and frame ID field” by setting HSEL (UART_LINCTL[23:22]). If the selected header is “break field”, software must handle the following sequence to send a complete header to bus by filling sync data (0x55) and frame ID data to the UART_DAT register. If the selected header is “break field and sync field”, software must handle the sequence to send a complete header to bus by filling the frame ID data to UART_DAT register, and if the selected header is “break field, sync field and frame ID field”, hardware will control the header sending sequence automatically but software must filled frame ID data to PID (UART_LINCTL [31:24]). When operating in header selected mode in which the selected header is “break field, sync field and frame ID field”, the frame ID parity bit can be calculated by software or hardware depending whether the IDPEN (UART_LINCTL[9]) bit is set or not.

HSEL	Break Field	Sync Field	ID Field
0	Generated by Hardware	Handled by Software	Handled by Software
1	Generated by Hardware	Generated by Hardware	Handled by Software
2	Generated by Hardware	Generated by Hardware	Generated by Hardware (But Software needs to fill ID to PID (UART_LINCTL[31:24]) first)

Table 6.16-11 LIN Header Selection in Master Mode

When UART is operated in LIN data transmission, LIN bus transfer state can be monitored by hardware or software. User can enable hardware monitoring by setting BITERREN (UART_LINCTL [12]) to “1”, if the input pin (UART_RX) state is not equal to the output pin (UART_TX) state in LIN transmitter state that hardware will generate an interrupt to CPU. Software can also monitor the LIN bus transfer state by checking the read back data in UART_DAT register. The following sequence is a program sequence example.

The procedure without software error monitoring in Master mode:

1. Fill Protected Identifier to PID (UART_LINCTL[31:24]).

2. Select the hardware transmission header field including “break field + sync field + protected identifier field” by setting HSEL (UART_LINCTL [23:22]) to “10”.
3. Set SENDH (UART_LINCTL[8]) bit to 1 for requesting header transmission.
4. Wait until SENDH (UART_LINCTL[8]) bit cleared by hardware.
5. Wait until TXEMPTYF (UART_FIFOSTS[28]) set to 1 by hardware.

Note1: The default setting of break field is 12 dominant bits (break field) and 1 recessive bit break/sync delimiter. Setting BRKFL (UART_LINCTL [19:16]) and BSL (UART_LINCTL[21:20]) to change the LIN break field length and break/sync delimiter length.

Note2: The default setting of break/sync delimiter length is 1-bit time and the inter-byte spaces default setting is also 1-bit time. Setting BSL (UART_LINCTL[21:20]) and DLY(UART_TOUT[15:8]) can change break/sync delimiter length and inter-byte spaces.

Note3: If the header includes the “break field, sync field and frame ID field”, software must fill frame ID to PID (UART_LINCTL[31:24]) before trigger header transmission (setting the SENDH (UART_LINCTL[8]). The frame ID parity can be generated by software or hardware depending on IDPEN (UART_LINCTL[9]) setting. If the parity generated by software with IDPEN (UART_LINCTL[9]) is set to ‘0’, software must fill 8 bit data (include 2 bit parity) in this field. If the parity generated by hardware with IDPEN (UART_LINCTL[9]) is set to ‘1’ , software fills ID0~ID5 and hardware calculates P0 and P1.

The procedure with software error monitoring in Master mode:

1. Choose the hardware transmission header field to only include “break field” by setting HSEL (UART_LINCTL [23:22])] to ‘00’.
2. Enable break detection function by setting BRKDETEN (UART_LINCTL[10]).
3. Request break + break/sync delimiter transmission by setting the SENDH (UART_LINCTL[8]).
4. Wait until the BRKDETF (UART_LINSTS[8]) flag is set to “1” by hardware.
5. Request sync field transmission by writing 0x55 into UART_DAT register.
6. Wait until the RDAIF (UART_INTSTS[0]) is set to “1” by hardware and then read back the UART_DAT register.
7. Request header frame ID transmission by writing the protected identifier value to UART_DAT register.
8. Wait until the RDAIF (UART_INTSTS[0]) is set to “1” by hardware and then read back the UART_DAT register.

LIN Break and Delimiter Detection

When software enables the break detection function by setting BRKDETEN (UART_LINCTL[10]), the break detection circuit is activated. The break detection circuit is totally independent from the UART receiver.

When the break detection function is enabled, the circuit looks at the input UART_RX pin for a start signal. If UART LIN controller detects consecutive dominant is greater than 11 bits dominant followed by a recessive bit (delimiter), the BRKDETF (UART_LINSTS[8]) flag is set at the end of break field. If the LINIEN (UART_INTEN[8]) bit is set to 1, an interrupt LININT (UART_INTSTS[15]) will be generated. The behavior of the break detection and break flag are shown in Figure 6.16-19.

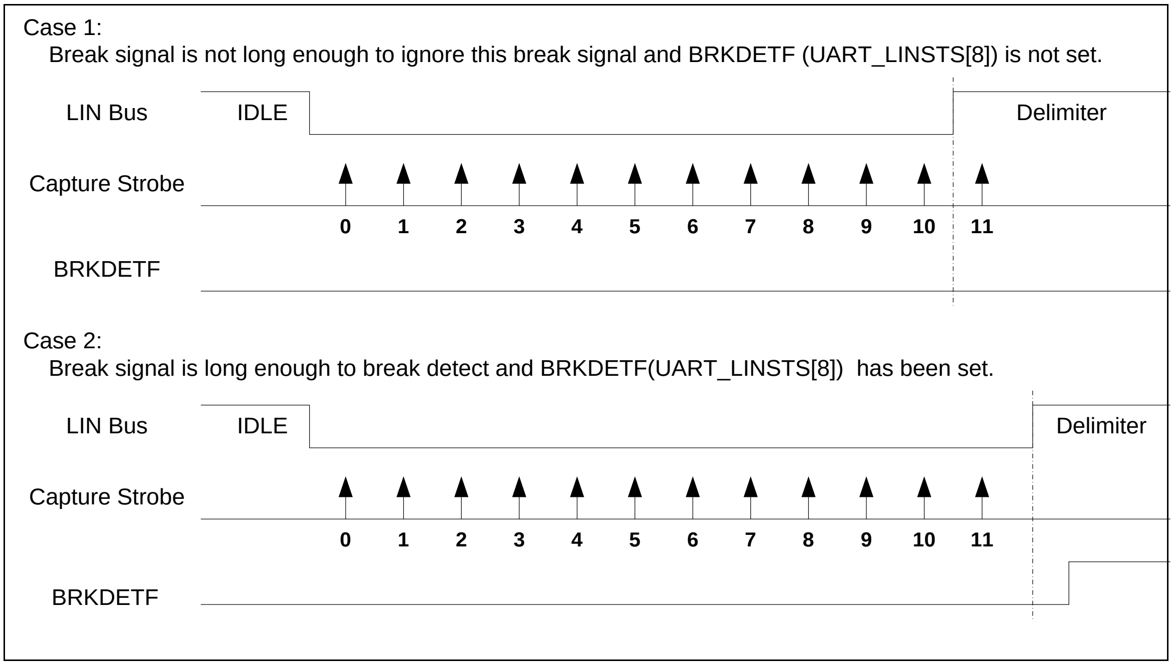

Case 1:
Break signal is not long enough to ignore this break signal and BRKDETF (UART_LINSTS[8]) is not set.
LIN Bus
IDLE
Delimiter
Capture Strobe
0
1
2
3
4
5
6
7
8
9
10
11
BRKDETF
Case 2:
Break signal is long enough to break detect and BRKDETF(UART_LINSTS[8]) has been set.
LIN Bus
IDLE
Delimiter
Capture Strobe
0
1
2
3
4
5
6
7
8
9
10
11
BRKDETF

Figure 6.16-19 Break Detection in LIN Mode

LIN Frame ID and Parity Format

The LIN frame ID value in LIN function mode is shown, the frame ID parity can be generated by software or hardware depends on IDPEN (UART_LINCTL[9]).

If the parity generated by hardware (IDPEN (UART_LINCTL[9])=1), user fill ID0~ID5 (UART_LINCTL [29:24]) hardware will calculate P0 (UART_LINCTL[30]) and P1 (UART_LINCTL[31]) otherwise user must filled frame ID and parity in this field.

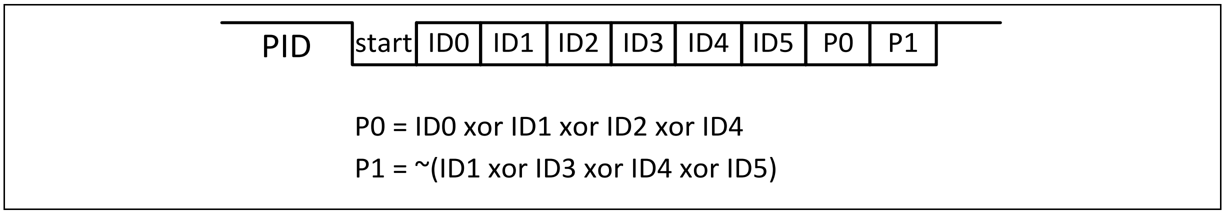

PID
start
ID0
ID1
ID2
ID3
ID4
ID5
P0
P1
P0 = ID0 xor ID1 xor ID2 xor ID4
P1 = ~(ID1 xor ID3 xor ID4 xor ID5)

Figure 6.16-20 LIN Frame ID and Parity Format

LIN Slave Mode

The UART Controller supports LIN Slave mode. To enable and initialize the LIN Slave mode, the following steps are necessary:

1. Set the UART_BAUD register to select the desired baud rate.
2. Configure the data length to 8 bits by setting WLS (UART_LINE[1:0]) to ‘11’ and disable parity check by clearing PBE (UART_LINE[3]) bit and configure with one stop bit by clearing NSB (UART_LINE[2]) bit.
3. Select LIN function mode by setting FUNCSEL (UART_FUNCSEL[1:0]) to ‘01’.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) to 1.

LIN Header Reception

According to the LIN protocol, a slave node must wait for a valid header which comes from the master node. Next the slave task will take one of following actions (depend on the master header frame ID value).

- Receive the response.
- Transmit the response.
- Ignore the response and wait for next header.

In LIN Slave mode, user can enable the slave header detection function by setting the SLVHDEN (UART_LINCTL[1]) to detect complete frame header (receive “break field”, “sync field” and “frame ID field”). When a LIN header is received, the SLVHDETF (UART_LINSTS[0]) flag will be set. If the LINIEN (UART_INTEN[8]) bit is set to ‘1’, an interrupt will be generated. User can enable the frame ID parity check function by setting IDPEN (UART_LINCTL[9]). If only received frame ID parity is not correct (break and sync filed are correct), the SLVIDPEF (UART_LINSTS[2]) flag is set to ‘1’. If the LINIEN(UART_INTEN[8]) is set to ‘1’, an interrupt will be generated and SLVHDETF (UART_LINSTS[0]) is set to ‘1’. User can also put LIN in mute mode by setting MUTE (UART_LINCTL[4]) to ‘1’. This mode allows detection of headers only (break + sync + frame ID) and prevents the reception of any other characters. In order to avoid bit rate tolerance, the controller supports automatic resynchronization function to avoid clock deviation error, user can enable this feature by setting SLVAREN (UART_LINCTL[2]).

LIN Response Transmission

The LIN slave node can transmit response and receive response. When slave node is the publisher of the response, the slave node sends response by filling data to the UART_DAT register. If the slave node is the subscriber of the response, the slave node receives data from LIN bus.

LIN Header Time-out Error

The LIN slave controller contains a header time-out counter. If the entire header is not received within the maximum time limit of 57 bit times, the header error flag SLVHEF (UART_LINSTS [1]) will be set. The time-out counter is enabled at each break detect edge and stopped in the following conditions.

- A LIN frame ID field has been received.
- The header error flag asserts.
- Writing 1 to the SLVSYNCF (UART_LINSTS[3]) to re-search a new frame header.

Mute Mode and LIN Exit from Mute Mode Condition

In Mute mode, a LIN slave node will not receive any data until specified condition occurred. It allows header detection only and prevents the reception of any other characters. User can enable Mute mode by setting the MUTE (UART_LINCTL[4]) and exiting from Mute mode condition can be selected by HSEL (UART_LINCTL[23:22]).

Note: It is recommended to set LIN slave node to Mute mode after checksum transmission.

The LIN slave controller exiting from Mute mode is described as follows: If HSEL (UART_LINCTL[23:22]) is set to “break field”, when LIN slave controller detects a valid LIN break + delimiter, the controller will enable the receiver (exit from Mute mode) and subsequent data (sync data, frame ID data, response data) are received in RX FIFO.

If HSEL (UART_LINCTL[23:22]) is set to “break field and sync field”, when the LIN slave controller detects a valid LIN break + delimiter followed by a valid sync field without frame error, the controller will enable the receiver (exit from mute mode) and subsequent data(ID data, response data) are received in RX FIFO. If HSEL (UART_LINCTL[23:22]) is set to “break field, sync field and ID field”, when the LIN slave controller detects a valid LIN break + delimiter and valid sync field without frame error followed by ID data without frame error and received ID data matched PID (UART_LINCTL[31:24]) value. The controller will enable the receiver (exit from mute mode) and subsequent data (response data) are received in RX FIFO.

Slave Mode Non-automatic Resynchronization (NAR)

User can disable the automatic resynchronization function to fix the communication baud rate. When operating in Non-Automatic Resynchronization mode, software needs some initial process, and the initialization process flow of Non-Automatic Resynchronization mode is shown as follows:

1. Select the desired baud rate by setting the UART_BAUD register.
2. Select LIN function mode by setting FUNCSEL (UART_FUNCSEL[1:0]) to ‘01’.
3. Disable automatic resynchronization function by setting SLVAREN (UART_LINCTL[2]) is set to ‘0’.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) is set to ‘1’.

Slave Mode with Automatic Resynchronization (AR)

In Automatic Resynchronization (AR) mode, the controller will adjust the baud rate generator after each sync field reception. The initialization process flow of Automatic Resynchronization mode is shown as follows:

1. Select the desired baud rate by setting the UART_BAUD register.
2. Select LIN function mode by setting UART_FUNCSEL (UART_FUNCSEL[1:0]) to ‘01’.
3. Enable automatic resynchronization function by setting SLVAREN (UART_LINCTL[2]) to ‘1’.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) is set to ‘1’.

When the automatic resynchronization function is enabled, after each LIN break field, the time duration between five falling edges is sampled on peripheral clock and the result of this measurement is stored in an internal 13-bit register and the UART_BAUD register value will be automatically updated at the end of the fifth falling edge. If the measure timer (13-bit) overflows before five falling edges, then the header error flag SLVHEF (UART_LINSTS [1]) will be set.

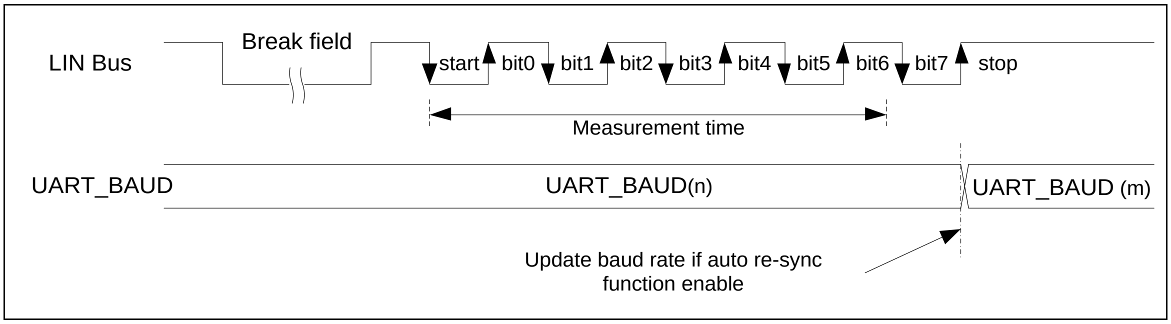


Figure 6.16-21 LIN Sync Field Measurement

When operating in Automatic Resynchronization (AR) mode, software must select the desired baud rate by setting the UART_BAUD register and hardware will store it at internal TEMP_REG register, after each LIN break field, the time duration between five falling edges is sampled on peripheral clock and the result of this measurement is stored in an internal 13-bit register BAUD_LIN and the result will be updated to UART_BAUD register automatically.

To guarantee the transmission baud rate, the baud rate generator must reload the initial value before each new break reception. The initial value is programmed by the application during initialization (TEMP_REG). User can set SLVDUEN (UART_LINCTL [3]) to enable auto reload initial baud rate value function. If the SLVDUEN (UART_LINCTL [3]) is set, when received the next character, hardware will auto reload the initial value to UART_BAUD, and when the UART_BAUD be updated, the SLVDUEN (UART_LINCTL [3]) will be cleared automatically. The behavior of LIN updated method as shown in Figure 6.16-22.

Note1: It is recommended to set the SLVDUEN bit before every checksum reception.

Note2: When a header error is detected, user must write ‘1’ to SLVSYNCF (UART_LINSTS[3]) to re-search new frame header. When writing ‘1’ to it, hardware will reload the initial baud rate TEMP_REG and re-search new frame header.

Note3: When operating in Automatic Resynchronization mode, the baud rate setting must be operated at mode2 (BAUDM1 (UART_BAUD [29]) and BAUDM0 (UART_BAUD[28]) must be ‘1’).

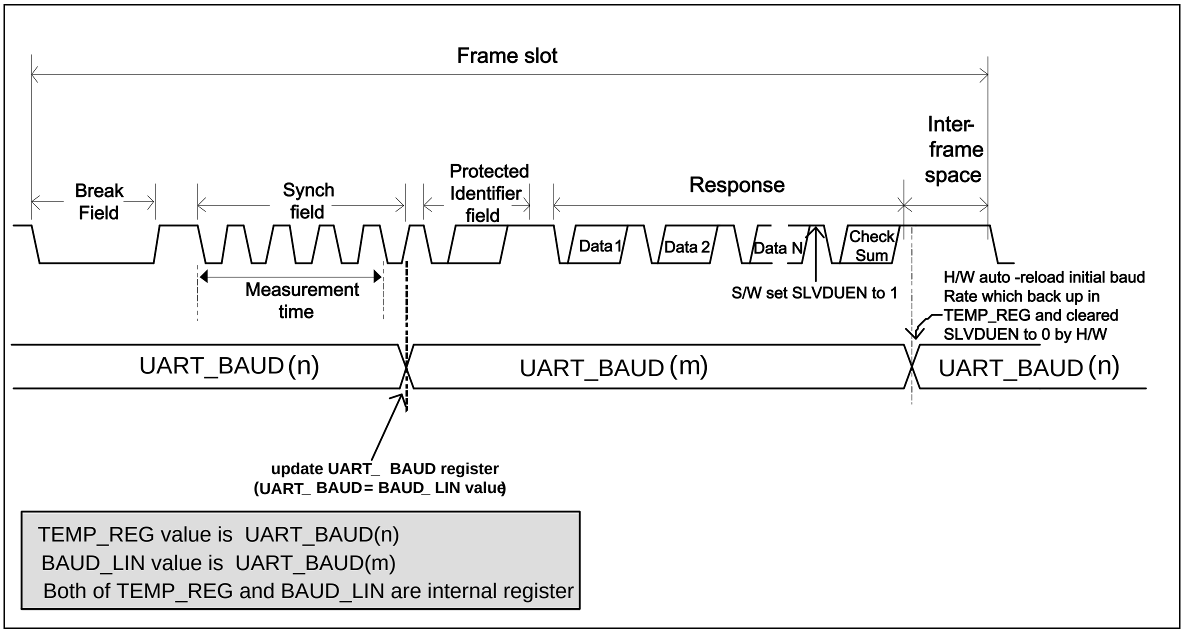

Frame slot
Break Field
Synch field
Protected Identifier field
Response
Inter-frame space
Data 1
Data 2
Data N
Check Sum
Measurement time
S/W set SLVDUEN to 1
H/W auto -reload initial baud Rate which back up in TEMP_REG and cleared SLVDUEN to 0 by H/W
UART_BAUD (n)
UART_BAUD (m)
UART_BAUD (n)
update UART_ BAUD register (UART_ BAUD= BAUD_ LIN value)
TEMP_REG value is UART_BAUD(n)
BAUD_LIN value is UART_BAUD(m)
Both of TEMP_REG and BAUD_LIN are internal register

Figure 6.16-22 UART_BAUD Update Sequence in AR Mode if SLVDUEN is 1

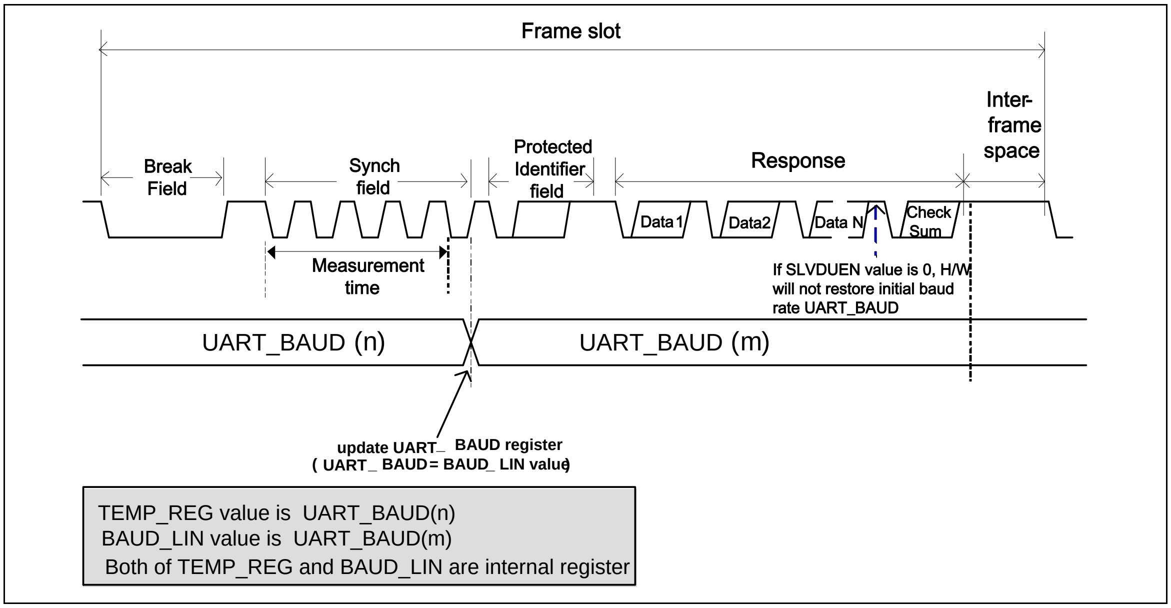

Frame slot
Break Field
Synch field
Protected Identifier field
Response
Inter-frame space
Data 1
Data2
Data N
Check Sum
Measurement time
If SLVDUEN value is 0, H/W will not restore initial baud rate UART_BAUD
UART_BAUD (n)
UART_BAUD (m)
update UART_ BAUD register
(UART_ BAUD= BAUD_ LIN value)
TEMP_REG value is UART_BAUD(n)
BAUD_LIN value is UART_BAUD(m)
Both of TEMP_REG and BAUD_LIN are internal register

Figure 6.16-23 UART_BAUD Update Sequence in AR Mode if SLVDUEN is 0

Deviation Error on the Sync Field

When operating in Automatic Resynchronization mode, the controller will check the deviation error on the sync field. The deviation error is checked by comparing the current baud rate with the received sync field. Two checks are performed in parallel.

Check1: Based on measurement between the first falling edge and the last falling edge of the sync field.

- If the difference is more than 14.84%, the header error flag SLVHEF (UART_LINSTS[1]) will be set.
- If the difference is less than 14.06%, the header error flag SLVHEF (UART_LINSTS[1]) will not be set.
- If the difference is between 14.84% and 14.06%, the header error flag SLVHEF (UART_LINSTS[1]) may either set or not.

Check2: Based on measurement of time between each falling edge of the sync field.

- If the difference is more than 18.75%, the header error flag SLVHEF (UART_LINSTS[1]) will be set.
- If the difference is less than 15.62%, the header error flag SLVHEF (UART_LINSTS[1]) will not be set.
- If the difference is between 18.75% and 15.62%, the header error flag SLVHEF (UART_LINSTS[1]) may either set or not.

Note: The deviation check is based on the current baud rate clock. Therefore, in order to guarantee correct deviation checking, the baud rate must reload the nominal value before each new break reception by setting SLVDUEN (UART_LINCTL[3]) register (It is recommend setting the SLVDUEN (UART_LINCTL[3]) bit before every checksum reception).

LIN Header Error Detection

In LIN Slave function mode, when user enables the header detection function by setting the SLVHDEN (UART_LINCTL[1]), hardware will handle the header detect flow. If the header has an error, the LIN header error flag SLVHEF (UART_LINSTS[1]) will be set and an interrupt is generated if the LINIEN (UART_INTEN[8]) bit is set. When header error is detected, user must reset the detect circuit to re-search a new frame header by writing ‘1’ to SLVSYNCF (UART_LINSTS[3]) to re-search a new frame header.

The LIN header error flag SLVHEF (UART_LINSTS[1]) is set if one of the following conditions occurs:

- Break Delimiter is too short (less than 0.5-bit time).
- Frame error in sync field or Identifier field.
- The sync field data is not 0x55 (Non-Automatic Resynchronization mode).
- The sync field deviation error (With Automatic Resynchronization mode).
- The sync field measure time-out (With Automatic Resynchronization mode).
- LIN header reception time-out.

6.16.5.11 RS-485 Function Mode

Another alternate function of UART controller is RS-485 function (user must set UART_FUNCSEL [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 UART_LINE register to control the 9-th bit (When the PBE, EPE and SPE are set, the 9-th bit is transmitted 0 and when PBE and SPE are set and EPE is cleared, the 9-th bit is transmitted 1).

The controller supports three operation modes: RS-485 Normal Multidrop Operation Mode (NMM), RS-485 Auto Address Detection Operation Mode (AAD) and RS-485 Auto Direction Control Operation Mode (AUD). Software can choose any operation mode by programming the UART_ALTCTL register, and drive the transfer delay time between the last stop bit leaving the TX FIFO and the de-assertion of by setting DLY (UART_TOUT [15:8]) register.

6.16.5.12 RS-485 Normal Multidrop Operation Mode (NMM)

In RS-485 Normal Multidrop Operation Mode (RS485NMM (UART_ALTCTL[8]) = ‘1’), in first, software must decide the data which before the address byte be detected will be stored in RX FIFO or not. If software wants to ignore any data before address byte detected, the flow is set RXOFF (UART_FIFO [8]) then enable RS485NMM (UART_ALTCTL [8]) and the receiver will ignore any data until an address byte is detected (bit 9 = ‘1’) and the address byte data will be stored in the RX FIFO. If software wants to receive any data before address byte detected, the flow is disables RXOFF (UART_FIFO [8]) then enable RS485NMM (UART_ALTCTL [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 RXOFF (UART_FIFO [8]) can decide whether accepting the following data bytes are stored in the RX FIFO. If software disables receiver by setting RXOFF (UART_FIFO [8]) register, when a next address byte is detected, the controller will clear the RXOFF (UART_FIFO [8]) bit and the address byte data will be stored in the RX FIFO.

6.16.5.13 RS-485 Auto Address Detection Operation Mode (AAD)

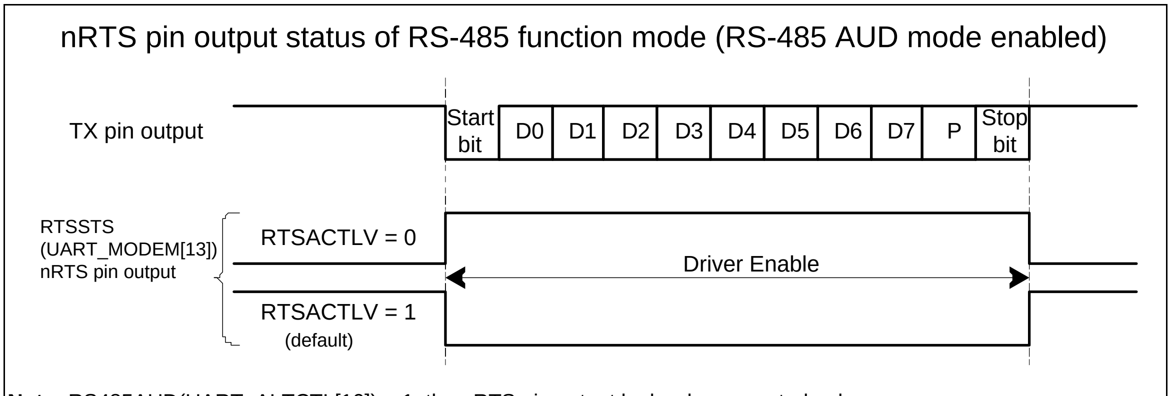
In RS-485 Auto Address Detection Operation Mode (RS485AAD (UART_ALTCTL[9]) = ‘1’), the receiver will ignore any data until an address byte is detected (bit 9 = ‘1’) and the address byte data matches the ADDRMV (UART_ALTCTL[31:24]) value. The address byte data will be stored in the RX FIFO. The all received byte data will be accepted and stored in the RX FIFO until an address byte data not match the ADDRMV (UART_ALTCTL[31:24]) value.

6.16.5.14 RS-485 Auto Direction Function (AUD)

Another option function of RS-485 controllers is RS-485 auto direction control function (RS485AUD (UART_ALTCTL[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 set RTSACTLV in UART_MODEM register to change the nRTS driving level.

Figure 6.16-24 demonstrates the RS-485 nRTS driving level in AUD mode. The nRTS pin will be automatically driven during TX data transmission.

Setting RTSACTLV(UART_MODEM[9]) can control nRTS pin output driving level. User can read the RTSSTS(UART_MODEM[13]) bit to get real nRTS pin output voltage logic status.

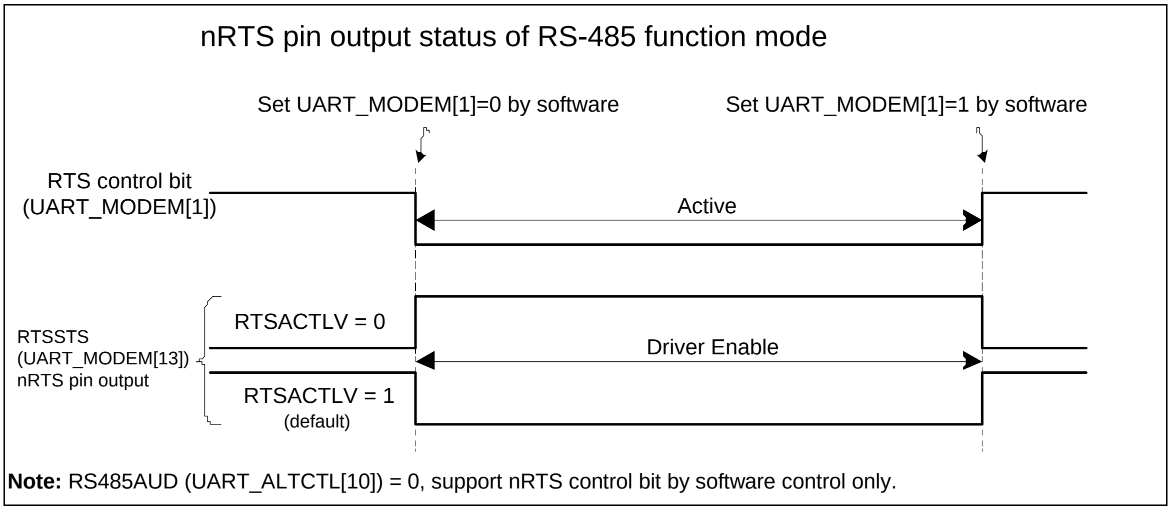


Note: RS485AUD(UART_ALTCTL[10]) = 1, the nRTS pin output by hardware control only.

Figure 6.16-24 RS-485 nRTS Driving Level in Auto Direction Mode

Figure 6.16-25 demonstrates the RS-485 nRTS driving level in software control (RS485AUD (UART_ALTCTL[10])=’0’). The nRTS driving level is controlled by programing the RTS(UART_MODEM[1]) control bit.

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

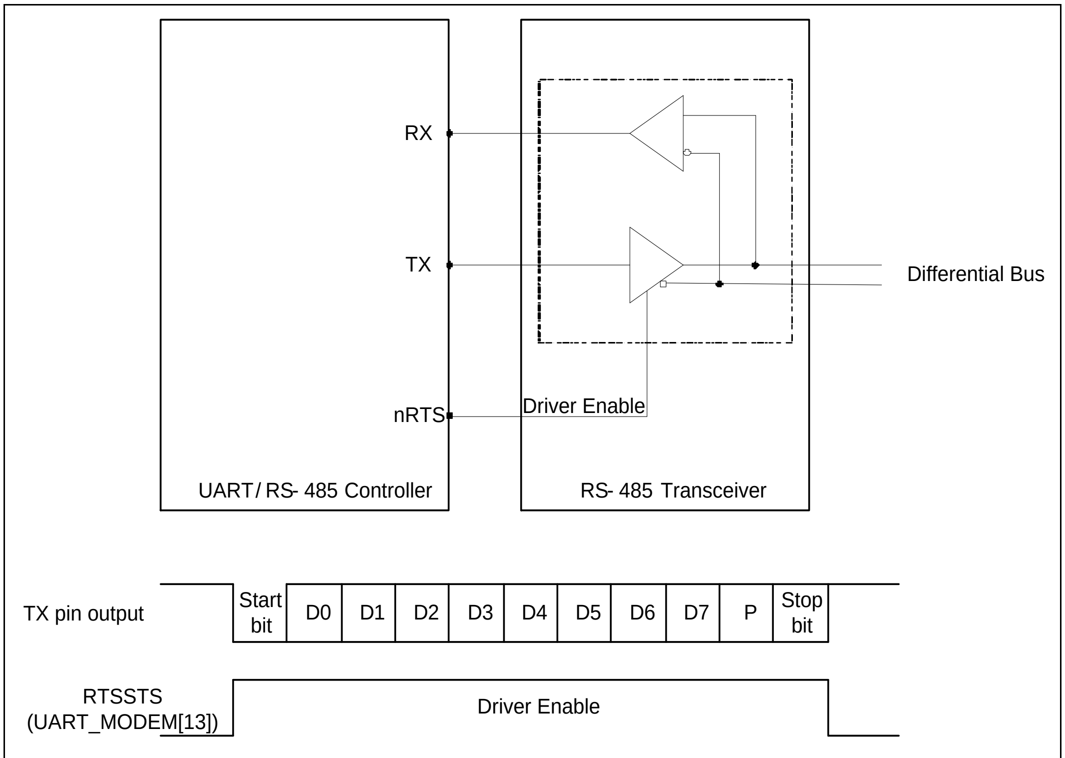


Note: RS485AUD (UART_ALTCTL[10]) = 0, support nRTS control bit by software control only.

Figure 6.16-25 RS-485 nRTS Driving Level with Software Control

Programming Sequence Example:

1. Program FUNCSEL in UART_FUNCSEL to select RS-485 function.
2. Program the RXOFF (UART_FIFO[8]) to determine enable or disable the receiver RS-485 receiver.
3. Program the RS485NMM (UART_ALTCTL[8]) or RS485AAD (UART_ALTCTL[9]) mode.
4. If the RS485AAD (UART_ALTCTL[9]) mode is selected, the ADDRMV (UART_ALTCTL[31:24]) is programmed for auto address match value.
5. Determine auto direction control by programming RS485AUD (UART_ALTCTL[10]).



Note: RS485AUD(UART_ALTCTL[10]) must be set to 1, and RTSACTLV(UART_MODEM[9]) must be set to 0.

Figure 6.16-26 Structure of RS-485 Frame

6.16.5.15 PDMA Transfer Function

UART controller supports PDMA transfer function.

By configuring PDMA parameter and set UART_DAT as the PDMA destination address. When TXPDMAEN (UART_INTEN[14]) is set to ‘1’, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

By configuring PDMA parameter and set UART_DAT as the PDMA source address. When RXPDMAEN (UART_INTEN[15]) is set to ‘1’, the controller will start the PDMA reception process. UART controller will issue request to PDMA controller automatically when there is data in the RX FIFO buffer.

Note: If STOPn (PDMA_STOP[n]) is set to stop UART RXPDMA task and the UART receive is not finish. UART controller will complete the transfer and stored current receive data in receive buffer. By reading RXEMPTY (UART_FIFOSTS[14]) to check there is valid data in receive buffer or not.

6.16.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UART Base Address: **UART0_BA = 0x4005_0000**				
UART_DAT **x=0**	UARTx_BA+0x00	R/W	UART Receive/Transmit Buffer Register	Undefined
UART_INTEN **x=0**	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000
UART_FIFO **x=0**	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101
UART_LINE **x=0**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000
UART_MODEM **x=0**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200
UART_MODEMSTS **x=0**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110
UART_FIFOSTS **x=0**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000
UART_INTSTS **x=0**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002
UART_TOUT **x=0**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000
UART_BAUD **x=0**	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000
UART_IRDA **x=0**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040
UART_ALTCTL **x=0**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C
UART_FUNCSEL **x=0**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000
UART_LINCTL **x=0**	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000
UART_LINSTS **x=0**	UARTx_BA+0x38	R/W	UART LIN Status Register	0x0000_0000

UART_BRCOMP **x=0**	UARTx_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000
UART_WKCTL **x=0**	UARTx_BA+0x40	R/W	UART Wake-up Control Register	0x0000_0000
UART_WKSTS **x=0**	UARTx_BA+0x44	R/W	UART Wake-up Status Register	0x0000_0000
UART_DWKCOMP **x=0**	UARTx_BA+0x48	R/W	UART Imcoming Data Wake-up Compensation Register	0x0000_0000

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

6.16.7 Register Description

UART Receive/Transmit Buffer Register (UART_DAT)

Register	Offset	R/W	Description	Reset Value
UART_DAT **x=0**	UARTx_BA+0x00	R/W	UART Receive/Transmit 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							**PARITY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAT							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**PARITY**	**Parity Bit Receive/Transmit Buffer** Write Operation: By writing to this bit, the parity bit will be stored in transmitter FIFO. If PBE (UART_LINE[3]) and PSS (UART_LINE[7]) are set, the UART controller will send out this bit follow the DAT (UART_DAT[7:0]) through the UART_TXD. Read Operation: If PBE (UART_LINE[3]) and PSS (UART_LINE[7]) are enabled, the parity bit can be read by this bit. **Note:** This bit has effect only when PBE (UART_LINE[3]) and PSS (UART_LINE[7]) are set.
[7:0]	**DAT**	**Data Receive/Transmit Buffer** Write Operation: 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. Read Operation: By reading this register, the UART controller will return an 8-bit data received from receiver FIFO.

UART Interrupt Enable Register (UART_INTEN)

Register	Offset	R/W	Description	Reset Value
UART_INTEN **x=0**	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	TXENDIEN	Reserved			ABRIEN	Reserved	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXPDMAEN	TXPDMAEN	ATOCTSEN	ATORTSEN	TOCNTEN	Reserved		LINIEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	WKIEN	BUFERRIEN	RXTOIEN	MODEMIEN	RLSIEN	THREIEN	RDAIEN

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22]	**TXENDIEN**	**Transmitter Empty Interrupt Enable Bit** If TXENDIEN (UART_INTEN[22]) is enabled, the Transmitter Empty interrupt TXENDINT (UART_INTSTS[30]) will be generated when TXENDIF (UART_INTSTS[22]) is set (TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted). 0 = Transmitter empty interrupt Disabled. 1 = Transmitter empty interrupt Enabled.
[21:19]	**Reserved**	Reserved.
[18]	**ABRIEN**	**Auto-baud Rate Interrupt Enable Bit** 0 = Auto-baud rate interrupt Disabled. 1 = Auto-baud rate interrupt Enabled.
[17:16]	**Reserved**	Reserved.
[15]	**RXPDMAEN**	**RX PDMA Enable Bit** This bit can enable or disable RX PDMA service. 0 = RX PDMA Disabled. 1 = RX PDMA Enabled. **Note:** If RLSIEN (UART_INTEN[2]) is enabled and HWRLSINT (UART_INTSTS[26]) is set to '1', the RLS (Receive Line Status) Interrupt is caused. If RLS interrupt is caused by Break Error Flag BIF(UART_FIFOSTS[6]), Frame Error Flag FEF(UART_FIFO[5]) or Parity Error Flag PEF(UART_FIFOSTS[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]	**TXPDMAEN**	**TX PDMA Enable Bit** This bit can enable or disable TX PDMA service. 0 = TX PDMA Disabled. 1 = TX PDMA Enabled.**Note:** If RLSIEN (UART_INTEN[2]) is enabled and HWRLSINT (UART_INTSTS[26]) is set to '1', the RLS (Receive Line Status) Interrupt is caused. If RLS interrupt is caused by Break Error Flag BIF(UART_FIFOSTS[6]), Frame Error Flag

		FEF(UART_FIFO[5]) or Parity Error Flag PEF(UART_FIFOSTS[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.
[13]	**ATOCTSEN**	**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 if nCTS input assert (UART will not send data to device until nCTS is asserted).
[12]	**ATORTSEN**	**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 equals the RTSTRGLV (UART_FIFO[19:16]), the UART will de-assert nRTS signal.
[11]	**TOCNTEN**	**Receive Buffer Time-out Counter Enable Bit** 0 = Receive Buffer Time-out counter Disabled. 1 = Receive Buffer Time-out counter Enabled.
[10:9]	**Reserved**	Reserved.
[8]	**LINIEN**	**LIN Bus Interrupt Enable Bit** 0 = LIN bus interrupt Disabled. 1 = LIN bus interrupt Enabled. **Note:** This bit is used for LIN function mode.
[7]	**Reserved**	Reserved.
[6]	**WKIEN**	**Wake-up Interrupt Enable Bit** 0 = Wake-up Interrupt Disabled. 1 = Wake-up Interrupt Enabled.
[5]	**BUFERRIEN**	**Buffer Error Interrupt Enable Bit** 0 = Buffer error interrupt Disabled. 1 = Buffer error interrupt Enabled.
[4]	**RXTOIEN**	**RX Time-out Interrupt Enable Bit** 0 = RX time-out interrupt Disabled. 1 = RX time-out interrupt Enabled.
[3]	**MODEMIEN**	**Modem Status Interrupt Enable Bit** 0 = Modem status interrupt Disabled. 1 = Modem status interrupt Enabled.
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable Bit** 0 = Receive Line Status interrupt Disabled. 1 = Receive Line Status interrupt Enabled.
[1]	**THREIEN**	**Transmit Holding Register Empty Interrupt Enable Bit** 0 = Transmit holding register empty interrupt Disabled. 1 = Transmit holding register empty interrupt Enabled.
[0]	**RDAIEN**	**Receive Data Available Interrupt Enable Bit** 0 = Receive data available interrupt Disabled. 1 = Receive data available interrupt Enabled.

UART FIFO Control Register (UART_FIFO)

Register	Offset	R/W	Description	Reset Value
UART_FIFO **x=0**	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				**RTSTRGLV**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**RXOFF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RFITL				**Reserved**	**TXRST**	**RXRST**	**Reserved**

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**RTSTRGLV**	**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]	**RXOFF**	**Receiver Disable Bit** The receiver is disabled or not (set 1 to disable receiver). 0 = Receiver Enabled. 1 = Receiver Disabled. **Note:** This bit is used for RS-485 Normal Multi-drop mode. It should be programmed before RS485NMM (UART_ALTCTL [8]) is programmed.
[7:4]	**RFITL**	**RX FIFO Interrupt Trigger Level** When the number of bytes in the receive FIFO equals the RFITL, the RDAIF (UART_INTSTS[0]) will be set (if RDAIEN (UART_INTEN [0]) enabled, and 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]	**TXRST**	**TX Field Software Reset**

		When TXRST (UART_FIFO[2]) 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. **Note1:** This bit will automatically clear at least 3 UART peripheral clock cycles. **Note2:** Before setting this bit, it should wait for the TXEMPTYF (UART_FIFOSTS[28]) be set.
[1]	**RXRST**	**RX Field Software Reset** When RXRST (UART_FIFO[1]) 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. **Note1:** This bit will automatically clear at least 3 UART peripheral clock cycles. **Note2:** Before setting this bit, it should wait for the RXIDLE (UART_FIFOSTS[29]) be set.
[0]	**Reserved**	Reserved.

UART Line Control Register (UART_LINE)

Register	Offset	R/W	Description	Reset Value
UART_LINE **x=0**	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						**RXDINV**	**TXDINV**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSS	**BCB**	**SPE**	**EPE**	**PBE**	**NSB**	**WLS**	

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**RXDINV**	**RX Data Inverted** 0 = Received data signal inverted Disabled. 1 = Received data signal inverted Enabled. **Note1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select UART, LIN or RS485 function.
[8]	**TXDINV**	**TX Data Inverted** 0 = Transmitted data signal inverted Disabled. 1 = Transmitted data signal inverted Enabled. **Note1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select UART, LIN or RS485 function.
[7]	**PSS**	**Parity Bit Source Selection** The parity bit can be selected to be generated and checked automatically or by software. 0 = Parity bit is generated by EPE (UART_LINE[4]) and SPE (UART_LINE[5]) setting and checked automatically. 1 = Parity bit generated and checked by software. **Note1:** This bit has effect only when PBE (UART_LINE[3]) is set. **Note2:** If PSS is 0, the parity bit is transmitted and checked automatically. If PSS is 1, the transmitted parity bit value can be determined by writing PARITY (UART_DAT[8]) and the parity bit can be read by reading PARITY (UART_DAT[8]).
[6]	**BCB**	**Break Control Bit** 0 = Break Control Disabled. 1 = Break Control Enabled.

		Note: When this bit is set to logic 1, the transmitted serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX line 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 (UART_LINE[3]) and EPE (UART_LINE[4]) are logic 1, the parity bit is transmitted and checked as logic 0. If PBE (UART_LINE[3]) is 1 and EPE (UART_LINE[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 (UART_LINE[3]) is set.
[3]	**PBE**	**Parity Bit Enable Bit** 0 = Parity bit generated Disabled. 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 (UART_MODEM)

Register	Offset	R/W	Description	Reset Value
UART_MODEM x=0	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		RTSSTS	Reserved			RTSACTLV	Reserved
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						RTS	Reserved

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**RTSSTS**	**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]	**RTSACTLV**	**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 6.16-13 and Figure 6.16-14 for UART function mode. **Note2:** Refer to Figure 6.16-24 and Figure 6.16-25 for RS-485 function mode. **Note3:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller.
[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 RTSACTLV 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 (UART_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
UART_MODEMSTS **x=0**	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							**CTSACTLV**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**CTSSTS**	**Reserved**			**CTSDETF**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CTSACTLV**	**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) **Note:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller.
[7:5]	**Reserved**	Reserved.
[4]	**CTSSTS**	**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]	**CTSDETF**	**Detect nCTS State Change Flag** This bit is set whenever nCTS input has change state, and it will generate Modem interrupt to CPU when MODEMIEN (UART_INTEN [3]) is set to 1. 0 = nCTS input has not change state. 1 = nCTS input has change state. **Note:** This bit can be cleared by writing “1” to it.

UART FIFO Status Register (UART_FIFOSTS)

Register	Offset	R/W	Description	Reset Value
UART_FIFOSTS x=0	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000

31	30	29	28	27	26	25	24
TXRXACT	Reserved	RXIDLE	TXEMPTYF	Reserved			TXOVIF
23	22	21	20	19	18	17	16
TXFULL	TXEMPTY	TXPTR					
15	14	13	12	11	10	9	8
RXFULL	RXEMPTY	RXPTR					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	ADDRDETF	ABRDTOIF	ABRDIF	RXOVIF

Bits	Description	
[31]	TXRXACT	**TX and RX Active Status (Read Only)** This bit indicates TX and RX are active or inactive. 0 = TX and RX are inactive. 1 = TX and RX are active. (Default) **Note:** When TXRXDIS (UART_FUNCSEL[3]) is set and both TX and RX are in idle state, this bit is cleared. The UART controller can not transmit or receive data at this moment. Otherwise this bit is set.
[30]	Reserved	Reserved.
[29]	RXIDLE	**RX Idle Status (Read Only)** This bit is set by hardware when RX is idle. 0 = RX is busy. 1 = RX is idle. (Default)
[28]	TXEMPTYF	**Transmitter Empty Flag (Read Only)** This bit is set by hardware when TX FIFO (UART_DAT) 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 been not 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]	TXOVIF	**TX Overflow Error Interrupt Flag** If TX FIFO (UART_DAT) is full, an additional write to UART_DAT 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 writing "1" to it.
[23]	TXFULL	**Transmitter FIFO Full (Read Only)**

		This bit indicates TX FIFO full or not. 0 = TX FIFO is not full. 1 = TX FIFO is full. **Note:** This bit is set when the number of usage in TX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[22]	**TXEMPTY**	**Transmitter FIFO Empty (Read Only)** This bit indicates TX FIFO empty or not. 0 = TX FIFO is not empty. 1 = TX FIFO is empty. **Note:** When the last byte of TX FIFO has been transferred to Transmitter Shift Register, hardware sets this bit high. It will be cleared when writing data into UART_DAT (TX FIFO not empty).
[21:16]	**TXPTR**	**TX FIFO Pointer (Read Only)** This field indicates the TX FIFO Buffer Pointer. When CPU writes one byte into UART_DAT, TXPTR increases one. When one byte of TX FIFO is transferred to Transmitter Shift Register, TXPTR decreases one. The Maximum value shown in TXPTR is 15. When the using level of TX FIFO Buffer equal to 16, the TXFULL bit is set to 1 and TXPTR will show 0. As one byte of TX FIFO is transferred to Transmitter Shift Register, the TXFULL bit is cleared to 0 and TXPTR will show 15.
[15]	**RXFULL**	**Receiver FIFO Full (Read Only)** This bit initiates RX FIFO full or not. 0 = RX FIFO is not full. 1 = RX FIFO is full. **Note:** This bit is set when the number of usage in RX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[14]	**RXEMPTY**	**Receiver FIFO Empty (Read Only)** This bit initiate RX FIFO empty or not. 0 = RX FIFO is not empty. 1 = RX FIFO is empty. **Note:** When the last byte of RX FIFO has been read by CPU, hardware sets this bit high. It will be cleared when UART receives any new data.
[13:8]	**RXPTR**	**RX FIFO Pointer (Read Only)** This field indicates the RX FIFO Buffer Pointer. When UART receives one byte from external device, RXPTR increases one. When one byte of RX FIFO is read by CPU, RXPTR decreases one. The Maximum value shown in RXPTR is 15. When the using level of RX FIFO Buffer equal to 16, the RXFULL bit is set to 1 and RXPTR will show 0. As one byte of RX FIFO is read by CPU, the RXFULL bit is cleared to 0 and RXPTR 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 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]	**ADDRDETF**	**RS-485 Address Byte Detect 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 ADDRDEN (UART_ALTCTL[15]) is set to 1 to enable Address detection mode. **Note2:** This bit can be cleared by writing “1” to it.
[2]	**ABRDTOIF**	**Auto-baud Rate Detect Time-out Interrupt Flag** This bit is set to logic “1” in Auto-baud Rate Detect mode when the baud rate counter is overflow. 0 = Auto-baud rate counter is underflow. 1 = Auto-baud rate counter is overflow. **Note:** This bit can be cleared by writing “1” to it.
[1]	**ABRDIF**	**Auto-baud Rate Detect Interrupt Flag** This bit is set to logic “1” when auto-baud rate detect function is finished. 0 = Auto-baud rate detect function is not finished. 1 = Auto-baud rate detect function is finished. **Note:** This bit can be cleared by writing “1” to it.
[0]	**RXOVIF**	**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 (UART_DAT) size 16 bytes, 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 Register (UART_INTSTS)

Register	Offset	R/W	Description	Reset Value
UART_INTSTS **x=0**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002

31	30	29	28	27	26	25	24
ABRINT	TXENDINT	HWBUFEINT	HWTOINT	HWMODINT	HWRLSINT	Reserved	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	TXENDIF	HWBUFEIF	HWTOIF	HWMODIF	HWRLSIF	Reserved	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
LININT	WKINT	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THREINT	RDAINT
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LINIF	WKIF	BUFERRIF	RXTOIF	MODEMIF	RLSIF	THREIF	RDAIF

Bits	Description	
[31]	**ABRINT**	**Auto-baud Rate Interrupt Indicator (Read Only)** This bit is set if ABRIEN (UART_INTEN[18]) and ABRIF (UART_ALTCTL[17]) are both set to 1. 0 = No Auto-baud Rate interrupt is generated. 1 = The Auto-baud Rate interrupt is generated.
[30]	**TXENDINT**	**Transmitter Empty Interrupt Indicator (Read Only)** This bit is set if TXENDIEN (UART_INTEN[22]) and TXENDIF(UART_INTSTS[22]) are both set to 1. 0 = No Transmitter Empty interrupt is generated. 1 = Transmitter Empty interrupt is generated.
[29]	**HWBUFEINT**	**PDMA Mode Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN (UART_INTEN[5]) and HWBUFEIF (UART_INTSTS[21]) are both set to 1. 0 = No buffer error interrupt is generated in PDMA mode. 1 = Buffer error interrupt is generated in PDMA mode.
[28]	**HWTOINT**	**PDMA Mode RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (UART_INTEN[4]) and HWTOIF(UART_INTSTS[20]) are both set to 1. 0 = No RX time-out interrupt is generated in PDMA mode. 1 = RX time-out interrupt is generated in PDMA mode.
[27]	**HWMODINT**	**PDMA Mode MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN (UART_INTEN[3]) and HWMODIF(UART_INTSTS[19]) are both set to 1. 0 = No Modem interrupt is generated inP DMA mode. 1 = Modem interrupt is generated in PDMA mode.
[26]	**HWRLSINT**	**PDMA Mode Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (UART_INTEN[2]) and HWRLSIF(UART_INTSTS[18]) are both set to 1.

		0 = No RLS interrupt is generated in PDMA mode. 1 = RLS interrupt is generated in PDMA mode.
[25:23]	**Reserved**	Reserved.
[22]	**TXENDIF**	**Transmitter Empty Interrupt Flag** This bit is set when TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted (TXEMPTYF (UART_FIFOSTS[28]) is set). If TXENDIEN (UART_INTEN[22]) is enabled, the Transmitter Empty interrupt will be generated. 0 = No transmitter empty interrupt flag is generated. 1 = Transmitter empty interrupt flag is generated. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[21]	**HWBUFEIF**	**PDMA Mode Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX or RX FIFO overflows (TXOVIF (UART_FIFOSTS [24]) or RXOVIF (UART_FIFOSTS[0]) is set). When BUFERRIF (UART_INTSTS[5]) is set, the transfer maybe is not correct. If BUFERRIEN (UART_INTEN [5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated in PDMA mode. 1 = Buffer error interrupt flag is generated in PDMA mode. **Note:** This bit is cleared when both TXOVIF (UART_FIFOSTS[24]]) and RXOVIF (UART_FIFOSTS[0]) are cleared.
[20]	**HWTOIF**	**PDMA Mode RX 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 (UART_TOUT[7:0]). If RXTOIEN (UART_INTEN [4]) is enabled, the RX time-out interrupt will be generated . 0 = No RX time-out interrupt flag is generated in PDMA mode. 1 = RX time-out interrupt flag is generated in PDMA mode. **Note:** This bit is read only and user can read UART_DAT (RX is in active) to clear it.
[19]	**HWMODIF**	**PDMA Mode MODEM Interrupt Flag (Read Only)** This bit is set when the nCTS pin has state change (CTSDETF (UART_MODEMSTS [0] =1)). If MODEMIEN (UART_INTEN [3]) is enabled, the Modem interrupt will be generated. 0 = No Modem interrupt flag is generated in PDMA mode. 1 = Modem interrupt flag is generated in PDMA mode. **Note:** This bit is read only and reset to 0 when the bit CTSDETF (UART_MODEMSTS[0]) is cleared by writing 1 on CTSDETF (UART_MODEMSTS [0]).
[18]	**HWRLSIF**	**PDMA Mode Receive Line Status Flag (Read Only)** This bit is set when the RX receive data have parity error, frame error or break error (at least one of 3 bits, BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]) and PEF (UART_FIFOSTS[4]) is set). If RLSIEN (UART_INTEN [2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated in PDMA mode. 1 = RLS interrupt flag is generated in PDMA mode. **Note1:** In RS-485 function mode, this field include “receiver detect any address byte received address byte character (bit9 = ‘1’) bit". **Note2:** In UART function mode, this bit is read only and reset to 0 when all bits of BIF(UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]) and PEF(UART_FIFOSTS[4]) are cleared. **Note3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF(UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]), PEF(UART_FIFOSTS[4]) and ADDRDETF (UART_FIFOSTS[3]) are cleared.
[17:16]	**Reserved**	Reserved.
[15]	**LININT**	**LIN Bus Interrupt Indicator (Read Only)**

		This bit is set if LINIEN (UART_INTEN[8]) and LINIF(UART_INTSTS[7]) are both set to 1. 0 = No LIN Bus interrupt is generated. 1 = The LIN Bus interrupt is generated.
[14]	**WKINT**	**UART Wake-up Interrupt Indicator (Read Only)** This bit is set if WKIEN (UART_INTEN[6]) and WKIF (UART_INTSTS[6]) are both set to 1. 0 = No UART wake-up interrupt is generated. 1 = UART wake-up interrupt is generated.
[13]	**BUFERRINT**	**Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN(UART_INTEN[5]) and BUFERRIF(UART_ INTSTS[5]) are both set to 1. 0 = No buffer error interrupt is generated. 1 = Buffer error interrupt is generated.
[12]	**RXTOINT**	**RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (UART_INTEN[4]) and RXTOIF(UART_INTSTS[4]) are both set to 1. 0 = No RX time-out interrupt is generated. 1 = RX time-out interrupt is generated.
[11]	**MODEMINT**	**MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN(UART_INTEN[3]) and MODEMIF(UART_INTSTS[3]) are both set to 1 0 = No Modem interrupt is generated. 1 = Modem interrupt is generated..
[10]	**RLSINT**	**Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (UART_INTEN[2]) and RLSIF(UART_INTSTS[2]) are both set to 1. 0 = No RLS interrupt is generated. 1 = RLS interrupt is generated.
[9]	**THREINT**	**Transmit Holding Register Empty Interrupt Indicator (Read Only)** This bit is set if THREIEN (UART_INTEN[1]) and THREIF(UART_INTSTS[1]) are both set to 1. 0 = No THRE interrupt is generated. 1 = THRE interrupt is generated.
[8]	**RDAINT**	**Receive Data Available Interrupt Indicator (Read Only)** This bit is set if RDAIEN (UART_INTEN[0]) and RDAIF (UART_INTSTS[0]) are both set to 1. 0 = No RDA interrupt is generated. 1 = RDA interrupt is generated.
[7]	**LINIF**	**LIN Bus Interrupt Flag** This bit is set when LIN slave header detect (SLVHDETF (UART_LINSTS[0] =1)), LIN break detect (BRKDETF(UART_LINSTS[8]=1)), bit error detect (BITEF(UART_LINSTS[9]=1)), LIN slave ID parity error (SLVIDPEF(UART_LINSTS[2] = 1)) or LIN slave header error detect (SLVHEF (UART_LINSTS[1])). If LINIEN (UART_INTEN [8]) is enabled the LIN interrupt will be generated. 0 = None of SLVHDETF, BRKDETF, BITEF, SLVIDPEF and SLVHEF is generated. 1 = At least one of SLVHDETF, BRKDETF, BITEF, SLVIDPEF and SLVHEF is generated. **Note:** This bit is cleared when SLVHDETF(UART_LINSTS[0]), BRKDETF(UART_LINSTS[8]), BITEF(UART_LINSTS[9]), SLVIDPEF (UART_LINSTS[2]) and SLVHEF(UART_LINSTS[1]) all are cleared and software writing ‘1’ to

		LINIF(UART_INTSTS[7]).
[6]	**WKIF**	**UART Wake-up Interrupt Flag (Read Only)** This bit is set when TOUTWKF (UART_WKSTS[4]), RS485WKF (UART_WKSTS[3]), RFRTWKF (UART_WKSTS[2]), DATWKF (UART_WKSTS[1]) or CTSWKF(UART_WKSTS[0]) is set to 1. 0 = No UART wake-up interrupt flag is generated. 1 = UART wake-up interrupt flag is generated. **Note:** This bit is cleared if all of TOUTWKF, RS485WKF, RFRTWKF, DATWKF and CTSWKF are cleared to 0 by writing 1 to the corresponding interrupt flag.
[5]	**BUFERRIF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX FIFO or RX FIFO overflows (TXOVIF (UART_FIFOSTS[24]) or RXOVIF (UART_FIFOSTS[0]) is set). When BUFERRIF (UART_INTSTS[5]) is set, the transfer is not correct. If BUFERRIEN (UART_INTEN [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 RXOVIF(UART_FIFOSTS[0]) and TXOVIF(UART_FIFOSTS[24]) are cleared to 0 by writing 1 to RXOVIF(UART_FIFOSTS[0]) and TXOVIF(UART_FIFOSTS[24]).
[4]	**RXTOIF**	**RX Time-out Interrupt Flag (Read Only)** This bit is set when the RX FIFO is not empty, RX does not receive data and the time-out counter equal to TOIC (UART_TOUT[7:0]). If RXTOIEN (UART_INTEN [4]) is enabled, the RX time-out interrupt will be generated. 0 = No RX time-out interrupt flag is generated. 1 = RX time-out interrupt flag is generated. **Note1:** This bit is read only and user can read UART_DAT (RX is in active) to clear it. **Note2:** This bit is set once if RX does not receive data after this bis is clear.
[3]	**MODEMIF**	**MODEM Interrupt Flag (Read Only)** This bit is set when the nCTS pin has state change (CTSDETF (UART_MODEMSTS[0]) = 1). If MODEMIEN (UART_INTEN [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 CTSDETF is cleared by a write 1 on CTSDETF(UART_MODEMSTS[0]).
[2]	**RLSIF**	**Receive Line Interrupt Flag (Read Only)** This bit is set when the RX receive data have parity error, frame error or break error (at least one of 3 bits, BIF(UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]) and PEF(UART_FIFOSTS[4]), is set). If RLSIEN (UART_INTEN [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 ADDRDETF (UART_FIFOSTS[3]) is also set. **Note2:** This bit is read only and reset to 0 when all bits of BIF (UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]) and PEF(UART_FIFOSTS[4]) are cleared. **Note3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF (UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]), PEF(UART_FIFOSTS[4]) and ADDRDETF (UART_FIFOSTS[3]) are cleared.
[1]	**THREIF**	**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 THREIEN (UART_INTEN[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 UART_DAT (TX FIFO not empty).
[0]	**RDAIF**	**Receive Data Available Interrupt Flag (Read Only)** When the number of bytes in the RX FIFO equals the RFITL then the RDAIF(UART_INTSTS[0]) will be set. If RDAIEN (UART_INTEN [0]) is enabled, the RDA interrupt will be generated. 0 = No RDA interrupt flag is generated. 1 = RDA interrupt flag is generated. **Note:** This bit is read only and it will be cleared when the number of unread bytes of RX FIFO drops below the threshold level (RFITL(UART_FIFO[7:4]).

UART Time-out Register (UART_TOUT)

Register	Offset	R/W	Description	Reset Value
UART_TOUT **x=0**	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 (the counting clock = baud rate) whenever the RX FIFO receives a new data word if time out counter is enabled by setting TOCNTEN (UART_INTEN[11]). Once the content of time-out counter is equal to that of time-out interrupt comparator (TOIC (UART_TOUT[7:0])), a receiver time-out interrupt (RXTOINT(UART_INTSTS[12])) is generated if RXTOIEN (UART_INTEN [4]) enabled. A new incoming data word or RX FIFO empty will clear RXTOIF (UART_INTSTS[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 with 40, the time-out interrupt is generated after four characters are not received when 1 stop bit and no parity check is set for UART transfer.

UART Baud Rate Divider Register (UART_BAUD)

Register	Offset	R/W	Description	Reset Value
UART_BAUD **x=0**	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000

31	30	29	28	27	26	25	24
Reserved		BAUDM1	BAUDM0	EDIVM1			
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]	**BAUDM1**	**BAUD Rate Mode Selection Bit 1** This bit is baud rate mode selection bit 1. UART provides three baud rate calculation modes. This bit combines with BAUDM0 (UART_BAUD[28]) to select baud rate calculation mode. The detail description is shown in Table 6.13-3. **Note:** In IrDA mode must be operated in mode 0.
[28]	**BAUDM0**	**BAUD Rate Mode Selection Bit 0** This bit is baud rate mode selection bit 0. UART provides three baud rate calculation modes. This bit combines with BAUDM1 (UART_BAUD[29]) to select baud rate calculation mode. The detail description is shown in Table 6.13-3.
[27:24]	**EDIVM1**	**Extra Divider for BAUD Rate Mode 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. The detail description is shown in Table 6.13-3.
[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. The detail description is shown in Table 6.13-3.

UART IrDA Control Register (UART_IRDA)

Register	Offset	R/W	Description	Reset Value
UART_IRDA **x=0**	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	RXINV	TXINV	Reserved			TXEN	Reserved

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**RXINV**	**IrDA Inverse Receive Input Signal** 0 = None inverse receiving input signal. 1 = Inverse receiving input signal. (Default) **Note1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select IrDA function.
[5]	**TXINV**	**IrDA Inverse Transmitting Output Signal** 0 = None inverse transmitting signal. (Default). 1 = Inverse transmitting output signal. **Note1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select IrDA function.
[4:2]	**Reserved**	Reserved.
[1]	**TXEN**	**IrDA Receiver/Transmitter Selection Enable Bit** 0 = IrDA Transmitter Disabled and Receiver Enabled. (Default) 1 = IrDA Transmitter Enabled and Receiver Disabled. **Note:** In IrDA mode, the BAUDM1 (UART_BAUD [29]) register must be disabled, the baud equation must be Clock / (16 * (BRD + 2)).
[0]	**Reserved**	Reserved.

Note: In IrDA mode, the BAUDM1 (UART_BAUD [29]) register must be disabled, the baud equation must be Clock / (16 * (BRD + 2)).

UART Alternate Control/Status Register (UART_ALTCTL)

Register	Offset	R/W	Description	Reset Value
UART_ALTCTL **x=0**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C

31	30	29	28	27	26	25	24
ADDRMV							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			ABRDBITS		ABRDEN	ABRIF	Reserved
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ADDRDEN	Reserved				RS485AUD	RS485AAD	RS485NMM
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LINTXEN	LINRXEN	Reserved		BRKFL			

Bits	Description	
[31:24]	**ADDRMV**	**Address Match Value** This field contains the RS-485 address match values. **Note:** This field is used for RS-485 auto address detection mode.
[23:21]	**Reserved**	Reserved.
[20:19]	**ABRDBITS**	**Auto-baud Rate Detect Bit Length** 00 = 1-bit time from Start bit to the 1st rising edge. The input pattern shall be 0x01. 01 = 2-bit time from Start bit to the 1st rising edge. The input pattern shall be 0x02. 10 = 4-bit time from Start bit to the 1st rising edge. The input pattern shall be 0x08. 11 = 8-bit time from Start bit to the 1st rising edge. The input pattern shall be 0x80. **Note :** The calculation of bit number includes the START bit.
[18]	**ABRDEN**	**Auto-baud Rate Detect Enable Bit** 0 = Auto-baud rate detect function Disabled. 1 = Auto-baud rate detect function Enabled. **Note :** This bit is cleared automatically after auto-baud detection is finished.
[17]	**ABRIF**	**Auto-baud Rate Interrupt Flag (Read Only)** This bit is set when auto-baud rate detection function finished or the auto-baud rate counter was overflow and if ABRIEN(UART_INTEN [18]) is set then the auto-baud rate interrupt will be generated. 0 = No auto-baud rate interrupt flag is generated. 1 = Auto-baud rate interrupt flag is generated. **Note:** This bit is read only, but it can be cleared by writing “1” to ABRDTOIF (UART_FIFOSTS[2]) and ABRDIF(UART_FIFOSTS[1]).
[16]	**Reserved**	Reserved.
[15]	**ADDRDEN**	**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 bit is used for RS-485 any operation mode.
[14:11]	**Reserved**	Reserved.
[10]	**RS485AUD**	**RS-485 Auto Direction Function (AUD)** 0 = RS-485 Auto Direction Operation function (AUD) Disabled. 1 = RS-485 Auto Direction Operation function (AUD) Enabled. **Note:** It can be active with RS-485_AAD or RS-485_NMM operation mode.
[9]	**RS485AAD**	**RS-485 Auto Address Detection Operation Mode (AAD)** 0 = RS-485 Auto Address Detection Operation mode (AAD) Disabled. 1 = RS-485 Auto Address Detection Operation mode (AAD) Enabled. **Note:** It cannot be active with RS-485_NMM operation mode.
[8]	**RS485NMM**	**RS-485 Normal Multi-drop Operation Mode (NMM)** 0 = RS-485 Normal Multi-drop Operation mode (NMM) Disabled. 1 = RS-485 Normal Multi-drop Operation mode (NMM) Enabled. **Note:** It cannot be active with RS-485_AAD operation mode.
[7]	**LINTXEN**	**LIN TX Break Mode Enable Bit** 0 = LIN TX Break mode Disabled. 1 = LIN TX Break mode Enabled. **Note:** When TX break field transfer operation finished, this bit will be cleared automatically.
[6]	**LINRXEN**	**LIN RX Enable Bit** 0 = LIN RX mode Disabled. 1 = LIN RX mode Enabled.
[5:4]	**Reserved**	Reserved.
[3:0]	**BRKFL**	**UART LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note1:** This break field length is BRKFL + 1. **Note2:** According to LIN spec, the reset value is 0xC (break field length = 13).

UART Function Select Register (UART_FUNCSEL)

Register	Offset	R/W	Description	Reset Value
UART_FUNCSEL x=0	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				TXRXDIS	Reserved	FUNCSEL	

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**TXRXDIS**	**TX and RX Disable Bit** Setting this bit can disable TX and RX. 0 = TX and RX Enabled. 1 = TX and RX Disabled. **Note:** The TX and RX will not disable immediately when this bit is set. The TX and RX compelet current task before disable TX and RX. When TX and RX disable, the TXRXACT (UART_FIFOSTS[31]) is cleared.
[2]	**Reserved**	Reserved.
[1:0]	**FUNCSEL**	**Function Select** 00 = UART function. 01 = LIN function. 10 = IrDA function. 11 = RS-485 function.

UART LIN Control Register (UART_LINCTL)

Register	Offset	R/W	Description	Reset Value
UART_LINCTL **x=0**	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000

31	30	29	28	27	26	25	24
PID							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HSEL		BSL		BRKFL			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			BITERREN	LINRXOFF	BRKDETEN	IDPEN	SENDH
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			MUTE	SLVDUEN	SLVAREN	SLVHDEN	SLVEN

Bits	Description	
[31:24]	**PID**	**LIN PID Bits** This field contains the LIN frame ID value when in LIN function mode, the frame ID parity can be generated by software or hardware depends on IDPEN (UART_LINCTL[9]) = 1. If the parity generated by hardware, user fill ID0~ID5 (PID [29:24]), hardware will calculate P0 (PID[30]) and P1 (PID[31]), otherwise user must filled frame ID and parity in this field. **Note1:** User can fill any 8-bit value to this field and the bit 24 indicates ID0 (LSB first). **Note2:** This field can be used for LIN master mode or slave mode.
[23:22]	**HSEL**	**LIN Header Select** 00 = The LIN header includes “break field”. 01 = The LIN header includes “break field” and “sync field”. 10 = The LIN header includes “break field”, “sync field” and “frame ID field”. 11 = Reserved. **Note:** This bit is used to master mode for LIN to send header field (SENDH (UART_LINCTL [8]) = 1) or used to slave to indicates exit from mute mode condition (MUTE (UART_LINCTL[4] = 1).
[21:20]	**BSL**	**LIN Break/Sync Delimiter Length** 00 = The LIN break/sync delimiter length is 1-bit time. 01 = The LIN break/sync delimiter length is 2-bit time. 10 = The LIN break/sync delimiter length is 3-bit time. 11 = The LIN break/sync delimiter length is 4-bit time. **Note:** This bit used for LIN master to sending header field.
[19:16]	**BRKFL**	**LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note1:** These registers are shadow registers of BRKFL (UART_ALTCTL[3:0]), User can read/write it by setting BRKFL (UART_ALTCTL[3:0]) or BRKFL (UART_LINCTL[19:16]). **Note2:** This break field length is BRKFL + 1. **Note3:** According to LIN spec, the reset value is 12 (break field length = 13).
[15:13]	**Reserved**	Reserved.

[12]	**BITERREN**	**Bit Error Detect Enable Bit** 0 = Bit error detection function Disabled. 1 = Bit error detection function Enabled. **Note:** In LIN function mode, when occur bit error, the BITEF (UART_LINSTS[9]) flag will be asserted. If the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated.
[11]	**LINRXOFF**	**LIN Receiver Disable Bit** If the receiver is enabled (LINRXOFF (UART_LINCTL[11]) = 0), all received byte data will be accepted and stored in the RX FIFO, and if the receiver is disabled (LINRXOFF (UART_LINCTL[11] = 1), all received byte data will be ignore. 0 = LIN receiver Enabled. 1 = LIN receiver Disabled. **Note:** This bit is only valid when operating in LIN function mode (FUNCSEL (UART_FUNCSEL[1:0]) = 01).
[10]	**BRKDETEN**	**LIN Break Detection Enable Bit** When detect consecutive dominant greater than 11 bits, and are followed by a delimiter character, the BRKDETF (UART_LINSTS[8]) flag is set at the end of break field. If the LINIEN (UART_INTEN [8])=1, an interrupt will be generated. 0 = LIN break detection Disabled . 1 = LIN break detection Enabled.
[9]	**IDPEN**	**LIN ID Parity Enable Bit** 0 = LIN frame ID parity Disabled. 1 = LIN frame ID parity Enabled. **Note1:** This bit can be used for LIN master to sending header field (SENDH (UART_LINCTL[8])) = 1 and HSEL (UART_LINCTL[23:22]) = 10 or be used for enable LIN slave received frame ID parity checked. **Note2:** This bit is only used when the operation header transmitter is in HSEL (UART_LINCTL[23:22]) = 10.
[8]	**SENDH**	**LIN TX Send Header Enable Bit** The LIN TX header can be “break field” or “break and sync field” or “break, sync and frame ID field”, it is depend on setting HSEL (UART_LINCTL[23:22]). 0 = Send LIN TX header Disabled. 1 = Send LIN TX header Enabled. **Note1:** This bit is shadow bit of LINTXEN (UART_ALTCTL [7]); user can read/write it by setting LINTXEN (UART_ALTCTL [7]) or SENDH (UART_LINCTL [8]). **Note2:** When transmitter header field (it may be “break” or “break + sync” or “break + sync + frame ID” selected by HSEL (UART_LINCTL[23:22]) field) transfer operation finished, this bit will be cleared automatically.
[7:5]	**Reserved**	Reserved.
[4]	**MUTE**	**LIN Mute Mode Enable Bit** 0 = LIN mute mode Disabled. 1 = LIN mute mode Enabled. **Note:** The exit from mute mode condition and each control and interactions of this field are explained in 6.16.5.10 (LIN slave mode).
[3]	**SLVDUEN**	**LIN Slave Divider Update Method Enable Bit** 0 = UART_BAUD updated is written by software (if no automatic resynchronization update occurs at the same time). 1 = UART_BAUD is updated at the next received character. User must set the bit before checksum reception. **Note1:** This bit only valid when in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note2:** This bit used for LIN Slave Automatic Resynchronization mode. (for Non-Automatic Resynchronization mode, this bit should be kept cleared)

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

		Note3: The control and interactions of this field are explained in 6.16.5.10 (Slave mode with automatic resynchronization).
[2]	**SLVAREN**	**LIN Slave Automatic Resynchronization Mode Enable Bit** 0 = LIN automatic resynchronization Disabled. 1 = LIN automatic resynchronization Enabled. **Note1:** This bit only valid when in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note2:** When operation in Automatic Resynchronization mode, the baud rate setting must be mode2 (BAUDM1 (UART_BAUD [29]) and BAUDM0 (UART_BAUD [28]) must be 1). **Note3:** The control and interactions of this field are explained in 6.16.5.10 (Slave mode with automatic resynchronization).
[1]	**SLVHDEN**	**LIN Slave Header Detection Enable Bit** 0 = LIN slave header detection Disabled. 1 = LIN slave header detection Enabled. **Note1:** This bit only valid when in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note2:** In LIN function mode, when detect header field (break + sync + frame ID), SLVHDETF (UART_LINSTS [0]) flag will be asserted. If the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated.
[0]	**SLVEN**	**LIN Slave Mode Enable Bit** 0 = LIN slave mode Disabled. 1 = LIN slave mode Enabled.

UART LIN Status Register (UART_LINSTS)

Register	Offset	R/W	Description	Reset Value
UART_LINSTS **x=0**	UARTx_BA+0x38	R/W	UART LIN Status Register	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**BITEF**	**Bit Error Detect Status Flag** At TX transfer state, hardware will monitor the bus state, if the input pin (UART_RXD) state not equals to the output pin (UART_TXD) state, BITEF (UART_LINSTS[9]) will be set. When occur bit error, if the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated. 0 = Bit error not detected. 1 = Bit error detected. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid when enable bit error detection function (BITERREN (UART_LINCTL [12]) = 1).
[8]	**BRKDETF**	**LIN Break Detection Flag** This bit is set by hardware when a break is detected and be cleared by writing 1 to it through software. 0 = LIN break not detected. 1 = LIN break detected. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid when LIN break detection function is enabled (BRKDETEN (UART_LINCTL[10]) =1).
[7:4]	**Reserved**	Reserved.
[3]	**SLVSYNCF**	**LIN Slave Sync Field** This bit indicates that the LIN sync field is being analyzed in Automatic Resynchronization mode. When the receiver header have some error been detect, user must reset the internal circuit to re-search new frame header by writing 1 to this bit. 0 = The current character is not at LIN sync state. 1 = The current character is at LIN sync state. **Note1:** This bit is only valid when in LIN Slave mode (SLVEN(UART_LINCTL[0]) = 1). **Note2:** This bit can be cleared by writing 1 to it. **Note3:** When writing 1 to it, hardware will reload the initial baud rate and re-search a new

		frame header.
[2]	**SLVIDPEF**	**LIN Slave ID Parity Error Flag** This bit is set by hardware when receipted frame ID parity is not correct. 0 = No active. 1 = Receipted frame ID parity is not correct. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid when in LIN slave mode (SLVEN (UART_LINCTL [0])= 1) and enable LIN frame ID parity check function IDPEN (UART_LINCTL [9]).
[1]	**SLVHEF**	**LIN Slave Header Error Flag** This bit is set by hardware when a LIN header error is detected in LIN slave mode and be cleared by writing 1 to it. The header errors include “break delimiter is too short (less than 0.5 bit time)”, “frame error in sync field or Identifier field”, “sync field data is not 0x55 in Non-Automatic Resynchronization mode”, “sync field deviation error with Automatic Resynchronization mode”, “sync field measure time-out with “Automatic Resynchronization mode” and “LIN header reception time-out”. 0 = LIN header error not detected. 1 = LIN header error detected. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid when UART is operated in LIN slave mode (SLVEN (UART_LINCTL [0]) = 1) and enables LIN slave header detection function (SLVHDEN (UART_LINCTL [1])).
[0]	**SLVHDETF**	**LIN Slave Header Detection Flag** This bit is set by hardware when a LIN header is detected in LIN slave mode and be cleared by writing 1 to it. 0 = LIN header not detected. 1 = LIN header detected (break + sync + frame ID). **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid when in LIN slave mode (SLVEN (UART_LINCTL [0]) = 1) and enable LIN slave header detection function (SLVHDEN (UART_LINCTL [1])). **Note3:** When enable ID parity check IDPEN (UART_LINCTL [9]), if hardware detect complete header (“break + sync + frame ID”), the SLVHDETF will be set whether the frame ID correct or not.

UART Baud Rate Compensation Register (UART_BRCOMP)

Register	Offset	R/W	Description	Reset Value
UART_BRCOMP **x=0**	UARTx_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000

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

Bits	Description	
[31]	**BRCOMPDEC**	**Baud Rate Compensation Decrease** 0 = Positive (increase one module clock) compensation for each compensated bit. 1 = Negative (decrease one module clock) compensation for each compensated bit.
[30:9]	**Reserved**	Reserved.
[8:0]	**BRCOMP**	**Baud Rate Compensation Patten** These 9-bits are used to define the relative bit is compensated or not. BRCOMP[7:0] is used to define the compensation of UART_DAT[7:0] and BRCOM[8] is used to define the parity bit.

UART Wake-up Control Register (UART_WKCTL)

Register	Offset	R/W	Description	Reset Value
UART_WKCTL **x=0**	UARTx_BA+0x40	R/W	UART 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			**WKTOUTEN**	**WKRS485EN**	**WKRFRTEN**	**WKDATEN**	**WKCTSEN**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**WKTOUTEN**	**Received Data FIFO Reached Threshold Time-out Wake-up Enable Bit** 0 = Received Data FIFO reached threshold time-out wake-up system function Disabled. 1 = Received Data FIFO reached threshold time-out wake-up system function Enabled, when the system is in Power-down mode, Received Data FIFO reached threshold time-out will wake-up system from Power-down mode. **Note:** It is suggest the function is enabled when the WKRFRTEN (UART_WKCTL[2]) is set to 1.
[3]	**WKRS485EN**	**RS-485 Address Match (AAD Mode) Wake-up Enable Bit** 0 = RS-485 Address Match (AAD mode) wake-up system function Disabled. 1 = RS-485 Address Match (AAD mode) wake-up system function Enabled, when the system is in. Power-down mode, RS-485 Address Match will wake-up system from Power-down mode. **Note:** This bit is used for RS-485 Auto Address Detection (AAD) mode in RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1.
[2]	**WKRFRTEN**	**Received Data FIFO Reached Threshold Wake-up Enable Bit** 0 = Received Data FIFO reached threshold wake-up system function Disabled. 1 = Received Data FIFO reached threshold wake-up system function Enabled, when the system is in Power-down mode, Received Data FIFO reached threshold will wake-up system from. Power-down mode.
[1]	**WKDATEN**	**Incoming Data Wake-up Enable Bit** 0 = Incoming data wake-up system function Disabled. 1 = Incoming data wake-up system function Enabled, when the system is in Power-down mode, incoming data will wake-up system from Power-down mode.
[0]	**WKCTSEN**	**nCTS Wake-up Enable Bit** 0 = nCTS Wake-up system function Disabled. 1 = nCTS Wake-up system function Enabled, when the system is in Power-down mode, an

		external nCTS change will wake-up system from Power-down mode.

UART Wake-up Status Register (UART_WKSTS)

Register	Offset	R/W	Description	Reset Value
UART_WKSTS **x=0**	UARTx_BA+0x44	R/W	UART 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			**TOUTWKF**	**RS485WKF**	**RFRTWKF**	**DATWKF**	**CTSWKF**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**TOUTWKF**	**Received Data FIFO Threshold Time-out Wake-up Flag** This bit is set if chip wake-up from power-down state by Received Data FIFO Threshold Time-out wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Received Data FIFO reached threshold time-out wake-up. **Note1:** If WKTOUTEN (UART_WKCTL[4]) is enabled, the Received Data FIFO reached threshold time-out wake-up cause this bit is set to '1'. **Note2:** This bit can be cleared by writing '1' to it.
[3]	**RS485WKF**	**RS-485 Address Match (AAD Mode) Wake-up Flag** This bit is set if chip wake-up from power-down state by RS-485 Address Match (AAD mode). 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by RS-485 Address Match (AAD mode) wake-up. **Note1:** If WKRS485EN (UART_WKCTL[3]) is enabled, the RS-485 Address Match (AAD mode) wake-up cause this bit is set to '1'. **Note2:** This bit can be cleared by writing '1' to it.
[2]	**RFRTWKF**	**Received Data FIFO Reached Threshold Wake-up Flag** This bit is set if chip wake-up from power-down state by Received Data FIFO reached threshold wake-up . 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Received Data FIFO Reached Threshold wake-up. **Note1:** If WKRFRTEN (UART_WKCTL[2]) is enabled, the Received Data FIFO Reached Threshold wake-up cause this bit is set to '1'.

		Note2: This bit can be cleared by writing '1' to it.
[1]	**DATWKF**	**Incoming Data Wake-up Flag** This bit is set if chip wake-up from power-down state by data wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Incoming Data wake-up. **Note1:** If WKDATEN (UART_WKCTL[1]) is enabled, the Incoming Data wake-up cause this bit is set to '1'. **Note2:** This bit can be cleared by writing '1' to it.
[0]	**CTSWKF**	**nCTS Wake-up Flag** This bit is set if chip wake-up from power-down state by nCTS wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by nCTS wake-up. **Note1:** If WKCTSEN (UART_WKCTL[0]) is enabled, the nCTS wake-up cause this bit is set to '1'. **Note2:** This bit can be cleared by writing '1' to it.

UART Imcoming Data Wake-up Compensation Register (UART_DWKCOMP)

Register	Offset	R/W	Description	Reset Value
UART_DWKCOMP **x=0**	UARTx_BA+0x48	R/W	UART Imcoming Data Wake-up Compensation 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**
STCOMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STCOMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**STCOMP**	**Start Bit Compensation Value** These bits field indicate how many clock cycle selected by UART_CLK do the UART controller can get the 1st bit (start bit) when the device is wake-up from power-down mode. **Note:** It is valid only when WKDATEN (UART_WKCTL[1]) is set.

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

6.17.1 Overview

I^{2}C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. The I^{2}C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously.

There are two sets of I^{2}C controllers which support Power-down wake-up function.

6.17.2 Features

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

- Supports up to two I^{2}C ports
- Supports speed up to 1Mbps
- 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
- Serial clock synchronization used as a handshake mechanism to suspend and resume serial transfer
- Built-in 14-bit time-out counter requesting the I^{2}C interrupt if the I^{2}C bus hangs up and timer-out counter overflows
- Programmable clocks 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
- Supports PDMA with one buffer capability
- Supports two-level buffer function
- Supports setup/hold time programmable

6.17.3 Basic Configuration

The basic configurations of I2C0 are as follows:

- I2C0 pins are configured on SYS_GPA_MFPL or SYS_GPD_MFPL or SYS_GPA_MFPH registers.
- Enable I2C0 clock I2C0CKEN (CLK_APBCLK0 [8]) register.
- Reset I2C0 controller I2C0RST (SYS_IPRST1 [8]) register.

The basic configurations of I2C1 are as follows:

- I2C1 pins are configured on SYS_GPC_MFPL or SYS_GPE_MFPL or SYS_GPF_MFPL registers.
- Enable I2C1 clock I2C1CKEN (CLK_APBCLK0[9]) register.
- Reset I2C1 controller I2C1RST (SYS_IPRST1[9]) register.

6.17.4 Block Diagram

The I^2C controller block diagram is shown as Figure 6.17-1.

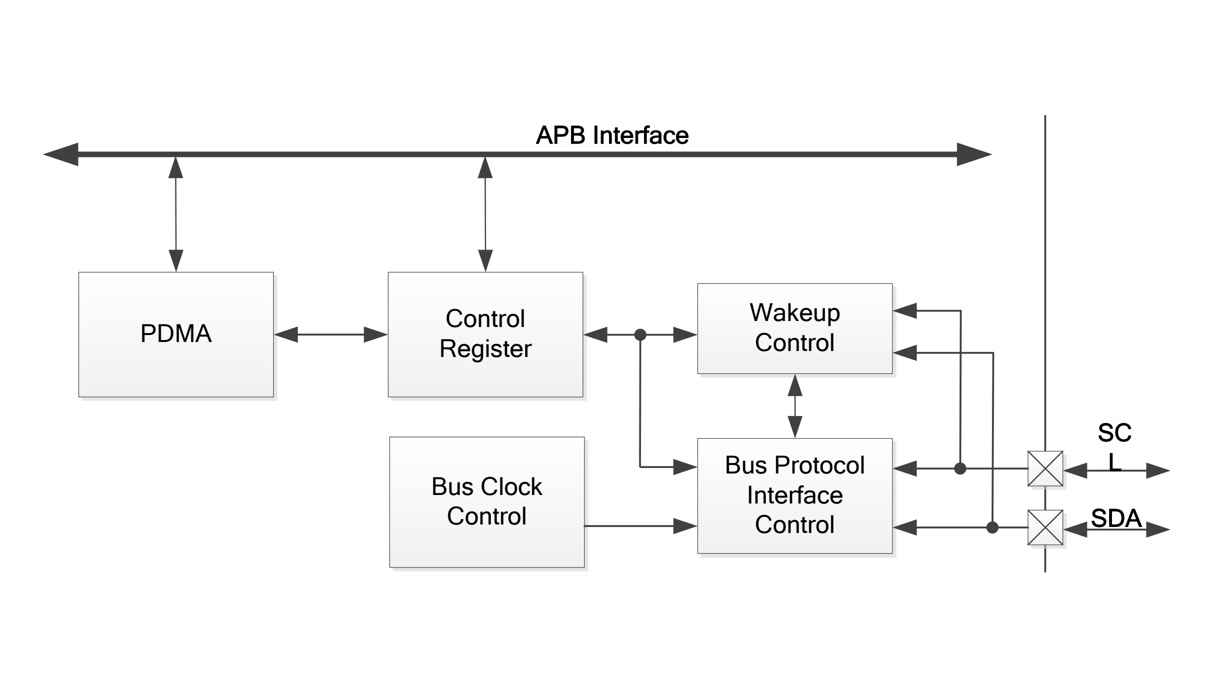


Figure 6.17-1 I^2C Controller Block Diagram

6.17.5 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 Figure 6.17-2 for more detailed I^2C BUS Timing.

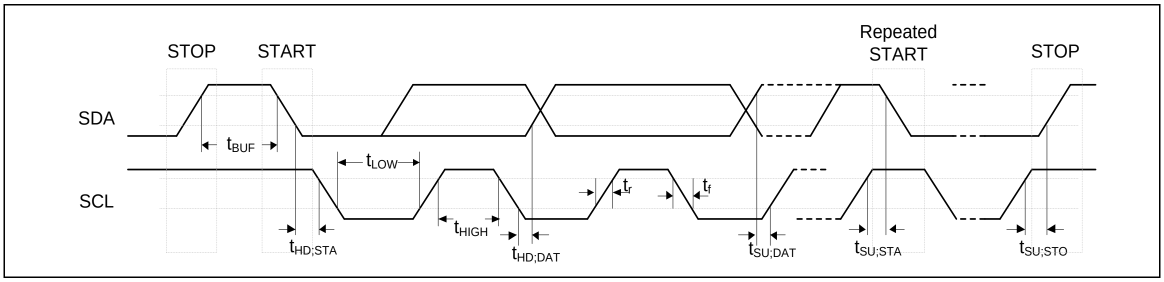


Figure 6.17-2 I^{2}C Bus Timing

The device's on-chip I^{2}C provides the serial interface that meets the I^{2}C bus standard mode specification. The I^{2}C port handles byte transfers autonomously. To enable this port, the bit I2CEN (I2C_CTL [6]) should be set to '1'. The I^{2}C hardware interfaces to the I^{2}C bus via two pins: SDA and SCL. When I/O pins are used as I^{2}C ports, user must set the pins function to I^{2}C in advance.

There is two-level buffer to improve the performance of I^{2}C bus. In two-level buffer mode, the next transmitted or the last received data can be active even if the current data is transmitted or the last received isn't read back yet.

There are under run or over run interrupt when the two-level buffer mode is enabled and the interrupt event enable is set. When two-level buffer under run, user will capture the last data be read. When two-level buffer over run, user will capture the last data be written.

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

6.17.5.1 I^{2}C Protocol

Figure 6.17-3 shows the typical I^{2}C protocol. Normally, a standard communication consists of four parts:

- START or Repeated START signal generation
- Slave address and R/W bit transfer
- Data transfer
- STOP signal generation

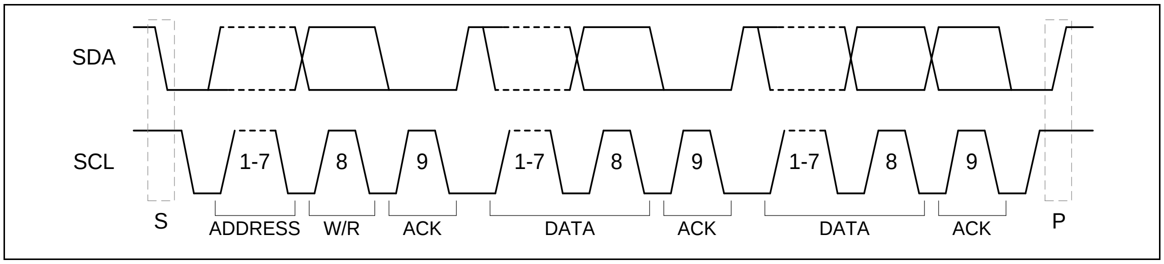


Figure 6.17-3 I^{2}C Protocol

START or Repeated START Signal

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

SCL is HIGH. The START signal denotes the beginning of a new data transmission.

After having sent the address byte (address and read/write bit), the master may send any number of bytes followed by a stop condition. Instead of sending the stop condition it is also allowed to send another start condition again followed by an address (and of course including a read/write bit) and more data. The start condition is called as Repeat START (Sr). This is defined recursively allowing any number of start conditions to be sent. The purpose of this is to allow combined write/read operations to one or more devices without releasing the bus and thus with the guarantee that the operation is not interrupted. The controller uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

STOP Signal

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

Figure 6.17-4 shows the waveform of START, Repeat START and STOP.

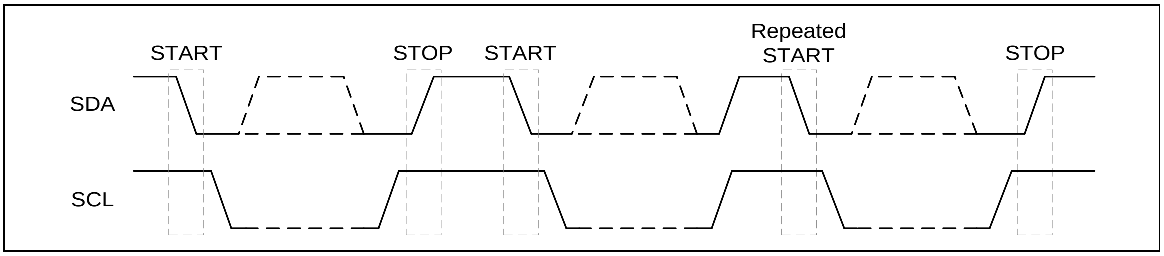


Figure 6.17-4 START and STOP Conditions

Slave Address Transfer

After a (repeated) start condition, the master sends a slave address to identify the target device of the communication. The start address can comprise one byte. After an address byte, a slave sensitive to the transmitted address has to acknowledge the reception.

Therefore, the slave's address can be programmed in the device, where it is compared to the received address. In case of a match, the slave answers with an acknowledge (SDA = 0). Slaves that are not targeted answer with a non-acknowledge (SDA = 1). In addition to the match of the programmed address, another address byte value has to be answered with an acknowledge if the slave is capable to handle the corresponding requests.

Data Transfer

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

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

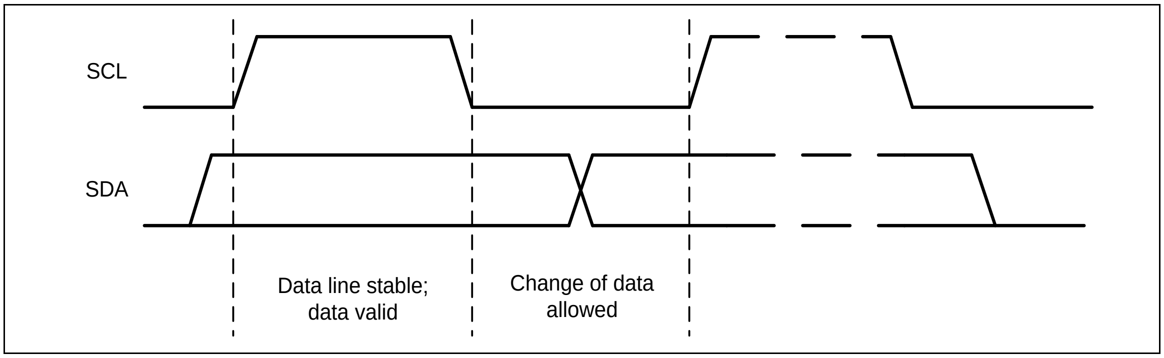


Figure 6.17-5 Bit Transfer on the I^{2}C Bus

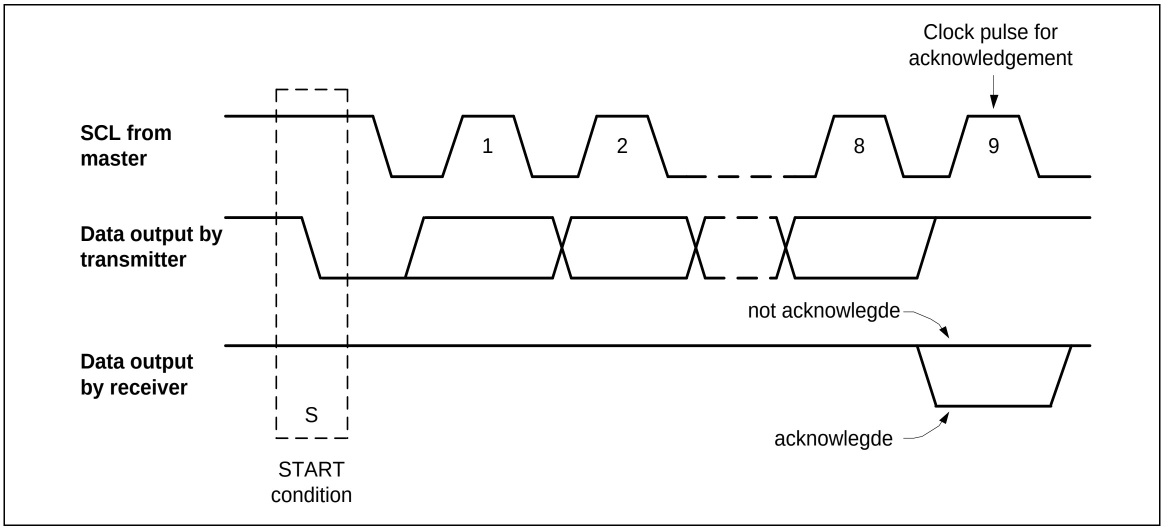


Figure 6.17-6 Acknowledge on the I^{2}C Bus

Data Transfer on I^{2}C Bus

Figure 6.17-7 shows a master transmits data to slave by 7-bit. 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.

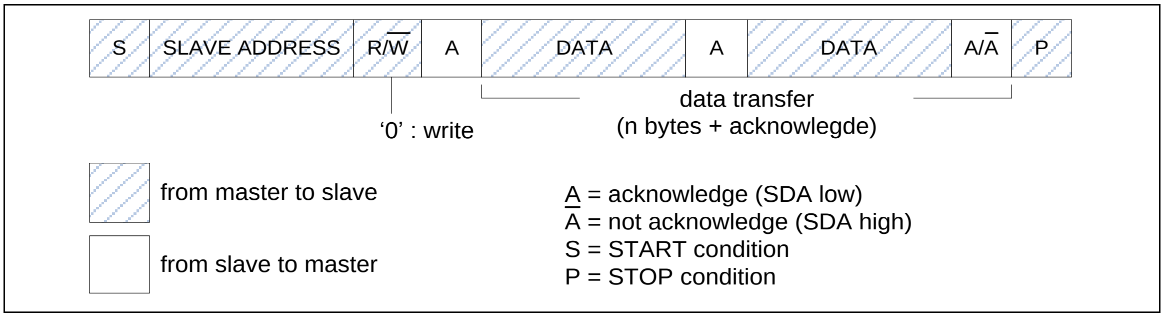


Figure 6.17-7 Master Transmits Data to Slave by 7-bit

Figure 6.17-8 shows a master read data from slave by 7-bit. 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.

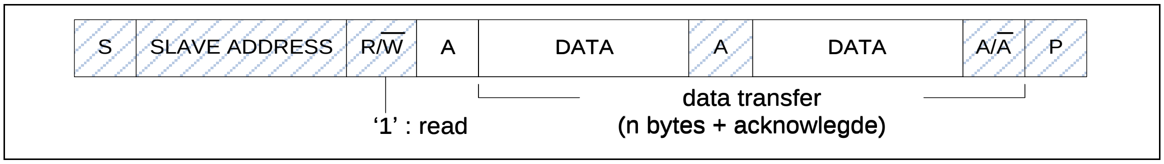


Figure 6.17-8 Master Reads Data from Slave by 7-bit

6.17.5.2 Operation Modes

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

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

To control the I^{2}C bus transfer in each mode, user needs to set I2C_CTL, I2C_DAT registers according to current status code of I2C_STATUS register. In other words, for each I^{2}C bus action, user needs to check current status by I2C_STATUS register, and then set I2C_CTL, I2C_DAT registers to take bus action. Finally, check the response status by I2C_STATUS.

The bits, STA, STO and AA in I2C_CTL register are used to control the next state of the I^{2}C hardware after SI (I2C_CTL [3]) is cleared. Upon completion of the new action, a new status code will be updated in I2C_STATUS register and the SI flag of I2C_CTL register will be set. But the SI flag will not be set when I^{2}C STOP. If the I^{2}C interrupt control bit INTEN (I2C_CTL [7]) is set, appropriate action or software branch of the new status code can be performed in the Interrupt service routine.

Figure 6.17-8 shows the current I^{2}C status code is 0x08, and then set I2C_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 ACK, the I2C_STATUS will be updated by status code 0x18.

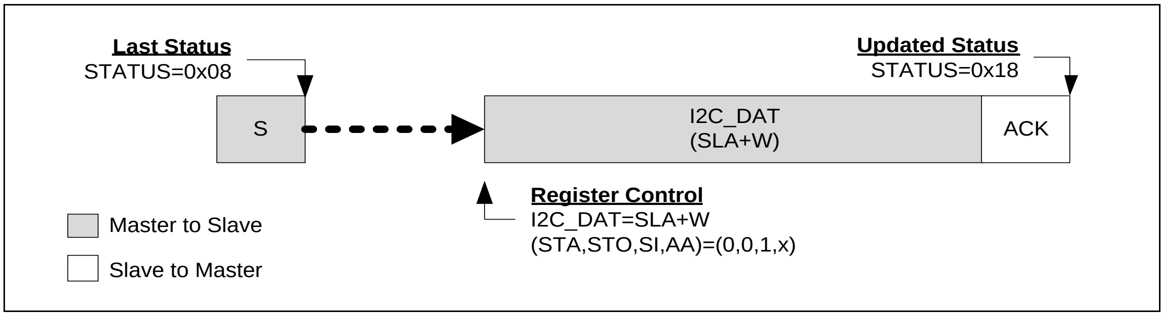


Figure 6.17-9 Control I^{2}C Bus according to the current I^{2}C Status

Master Mode

In Figure 6.17-10 and Figure 6.17-11, 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 (MT) mode (Figure 6.17-10) or Master receiver (MR) mode (Figure 6.17-11) after START signal has been sent successfully and new status code would be 0x08. Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^{2}C protocol.

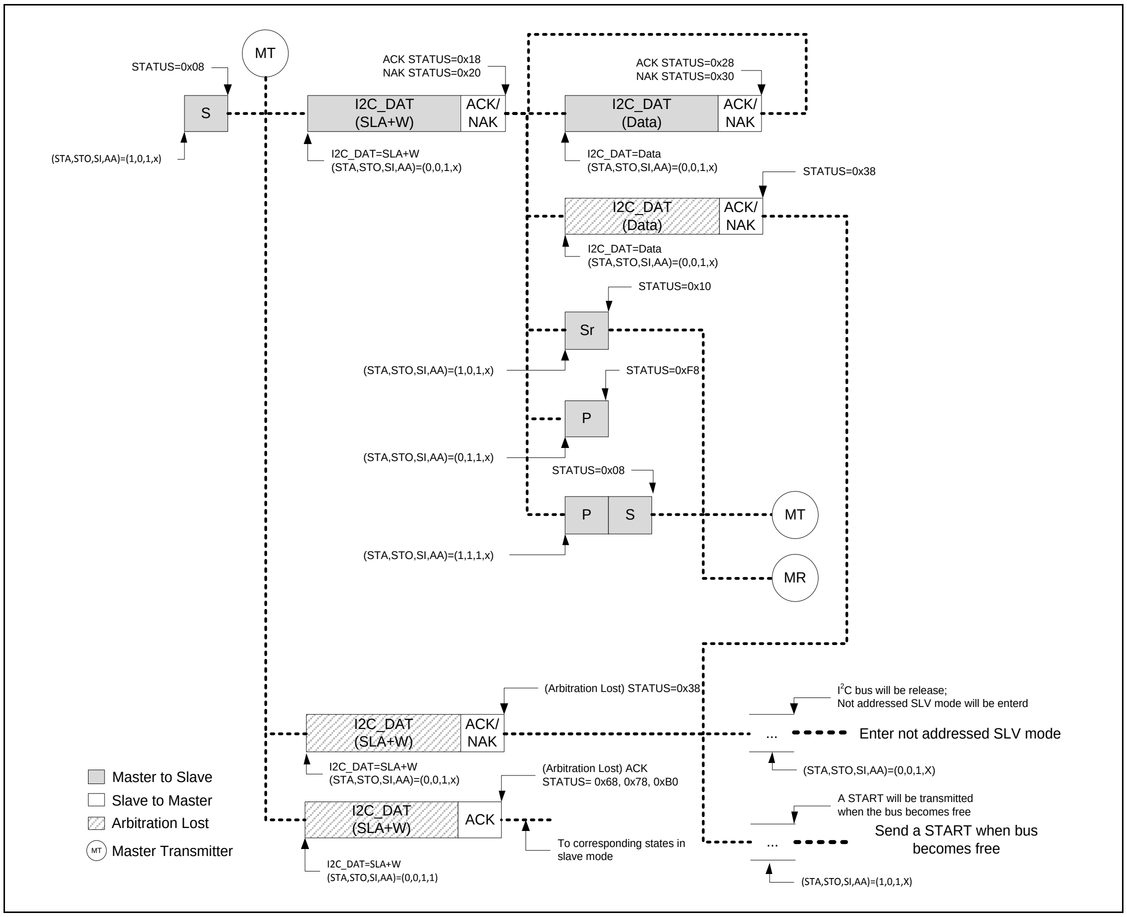

STATUS=0x08
MT
S
(STA,STO,SI,AA)=(1,0,1,x)
ACK STATUS=0x18
NAK STATUS=0x20
I2C_DAT (SLA+W)
ACK/ NAK
I2C_DAT=SLA+W
(STA,STO,SI,AA)=(0,0,1,x)
ACK STATUS=0x28
NAK STATUS=0x30
I2C_DAT (Data)
ACK/ NAK
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,x)
STATUS=0x38
I2C_DAT (Data)
ACK/ NAK
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,x)
STATUS=0x10
Sr
(STA,STO,SI,AA)=(1,0,1,x)
STATUS=0xF8
P
(STA,STO,SI,AA)=(0,1,1,x)
STATUS=0x08
P
S
MT
(STA,STO,SI,AA)=(1,1,1,x)
MR
(Arbitration Lost) STATUS=0x38
I2C_DAT (SLA+W)
ACK/ NAK
I2C_DAT=SLA+W
(STA,STO,SI,AA)=(0,0,1,x)
I^{2}C bus will be release;
Not addressed SLV mode will be enterd
...
Enter not addressed SLV mode
(STA,STO,SI,AA)=(0,0,1,X)
(Arbitration Lost) ACK
STATUS= 0x68, 0x78, 0xB0
I2C_DAT (SLA+W)
ACK
To corresponding states in slave mode
I2C_DAT=SLA+W
(STA,STO,SI,AA)=(0,0,1,1)
A START will be transmitted
when the bus becomes free
...
Send a START when bus becomes free
(STA,STO,SI,AA)=(1,0,1,X)
Master to Slave
Slave to Master
Arbitration Lost
MT
Master Transmitter

Figure 6.17-10 Master Transmitter Mode Control Flow

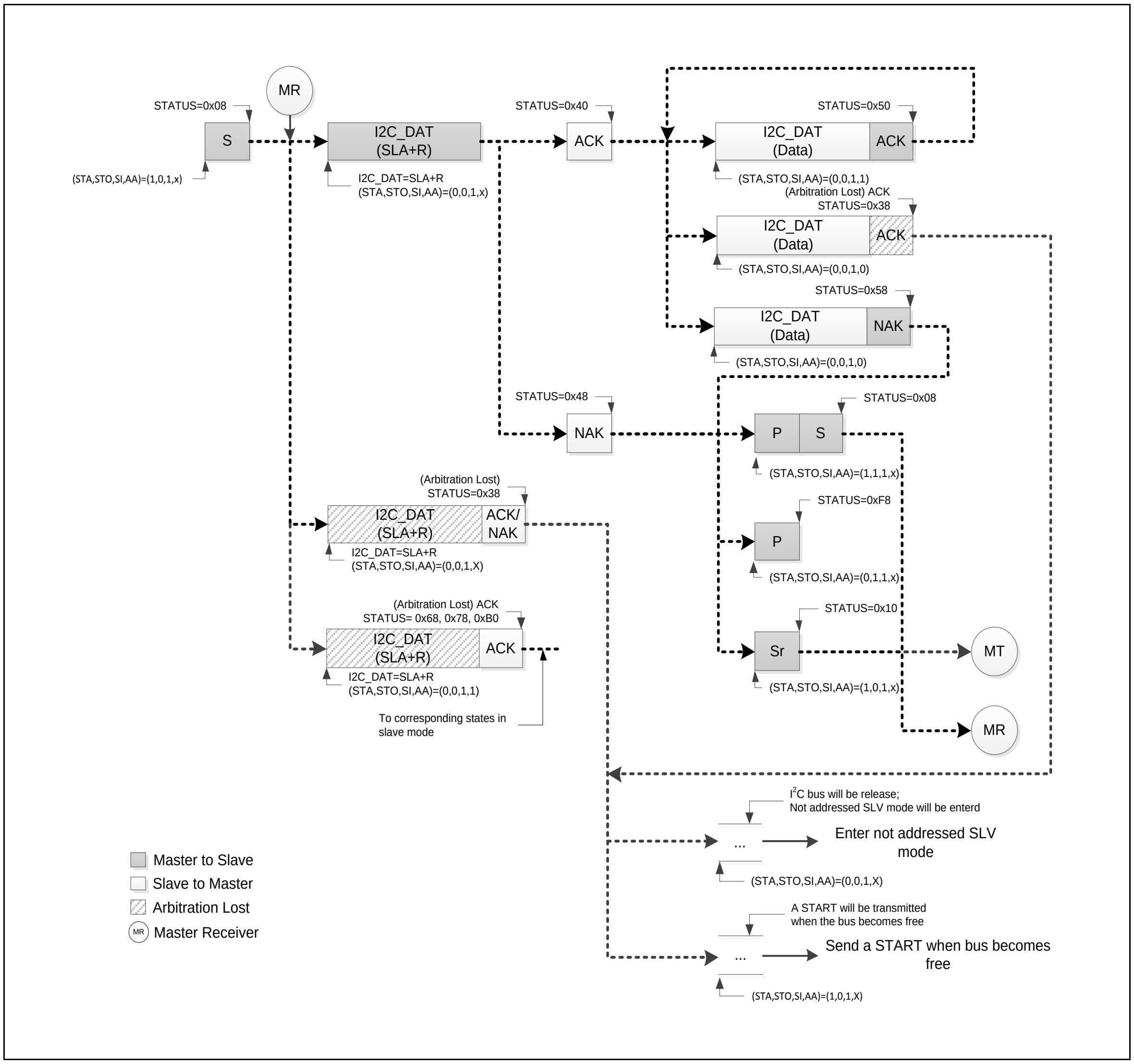

MR
STATUS=0x08
S
(STA,STO,SI,AA)=(1,0,1,x)
I2C_DAT (SLA+R)
I2C_DAT=SLA+R
(STA,STO,SI,AA)=(0,0,1,x)
STATUS=0x40
ACK
STATUS=0x50
I2C_DAT (Data)
ACK
(STA,STO,SI,AA)=(0,0,1,1)
(Arbitration Lost) ACK
STATUS=0x38
I2C_DAT (Data)
ACK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0x58
I2C_DAT (Data)
NAK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0x48
NAK
STATUS=0x08
P
S
(STA,STO,SI,AA)=(1,1,1,x)
(Arbitration Lost)
STATUS=0x38
I2C_DAT (SLA+R)
ACK/ NAK
I2C_DAT=SLA+R
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xF8
P
(STA,STO,SI,AA)=(0,1,1,x)
(Arbitration Lost) ACK
STATUS= 0x68, 0x78, 0xB0
I2C_DAT (SLA+R)
ACK
I2C_DAT=SLA+R
(STA,STO,SI,AA)=(0,0,1,1)
To corresponding states in slave mode
STATUS=0x10
Sr
MT
(STA,STO,SI,AA)=(1,0,1,x)
MR
I²C bus will be release;
Not addressed SLV mode will be enterd
...
Enter not addressed SLV mode
(STA,STO,SI,AA)=(0,0,1,X)
A START will be transmitted when the bus becomes free
...
Send a START when bus becomes free
(STA,STO,SI,AA)=(1,0,1,X)
Master to Slave
Slave to Master
Arbitration Lost
MR Master Receiver

Figure 6.17-11 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.

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 I2C_ADDRn (n=0~3) and set (STA, STO, SI, AA) = (0, 0, 1, 1) to let I^{2}C recognize the address sent by master. Figure 6.17-12 shows all the possible flow for I^{2}C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.17-12 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.

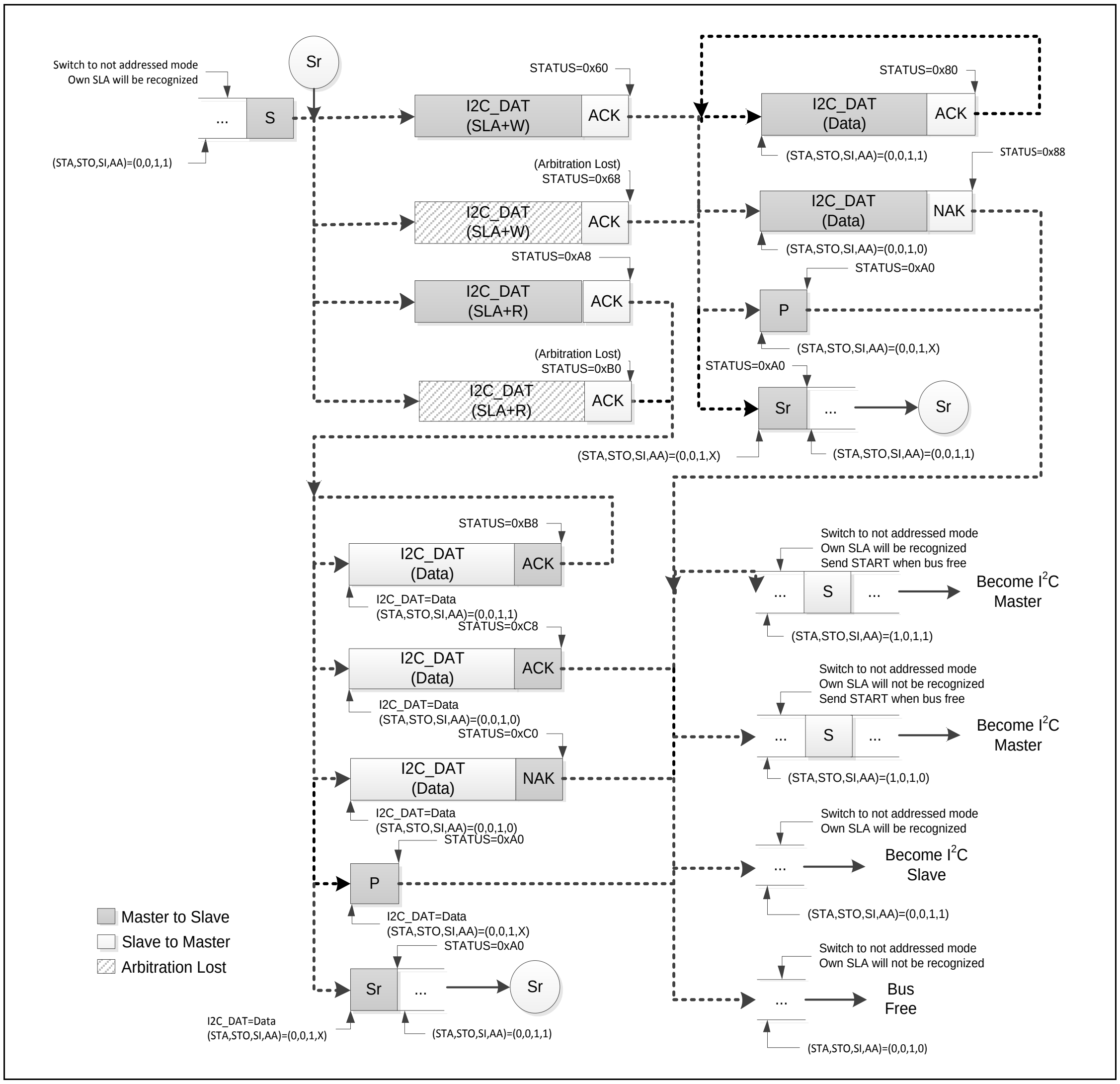

Switch to not addressed mode
Own SLA will be recognized
...
S
Sr
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0x60
I2C_DAT
(SLA+W)
ACK
(Arbitration Lost)
STATUS=0x68
I2C_DAT
(SLA+W)
ACK
STATUS=0xA8
I2C_DAT
(SLA+R)
ACK
(Arbitration Lost)
STATUS=0xB0
I2C_DAT
(SLA+R)
ACK
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0x80
I2C_DAT
(Data)
ACK
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0x88
I2C_DAT
(Data)
NAK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xA0
P
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xA0
Sr
...
Sr
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0xB8
I2C_DAT
(Data)
ACK
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0xC8
I2C_DAT
(Data)
ACK
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xC0
I2C_DAT
(Data)
NAK
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xA0
P
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xA0
Sr
...
Sr
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,X)
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will be recognized
Send START when bus free
... S ...
Become I2C Master
(STA,STO,SI,AA)=(1,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
Send START when bus free
... S ...
Become I2C Master
(STA,STO,SI,AA)=(1,0,1,0)
Switch to not addressed mode
Own SLA will be recognized
...
Become I2C Slave
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
...
Bus Free
(STA,STO,SI,AA)=(0,0,1,0)
Master to Slave
Slave to Master
Arbitration Lost

Figure 6.17-12 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 Figure 6.17-12 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 Figure 6.17-12 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 enter into idle mode.

General Call (GC) Mode

If the GC bit (I2C_ADDRn [0]) is set, the I^{2}C port hardware will respond to General Call address (00H). User can clear GC bit to disable general call function. When the GC bit is set and the I^{2}C 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.

The GC mode can wake up when address matched.

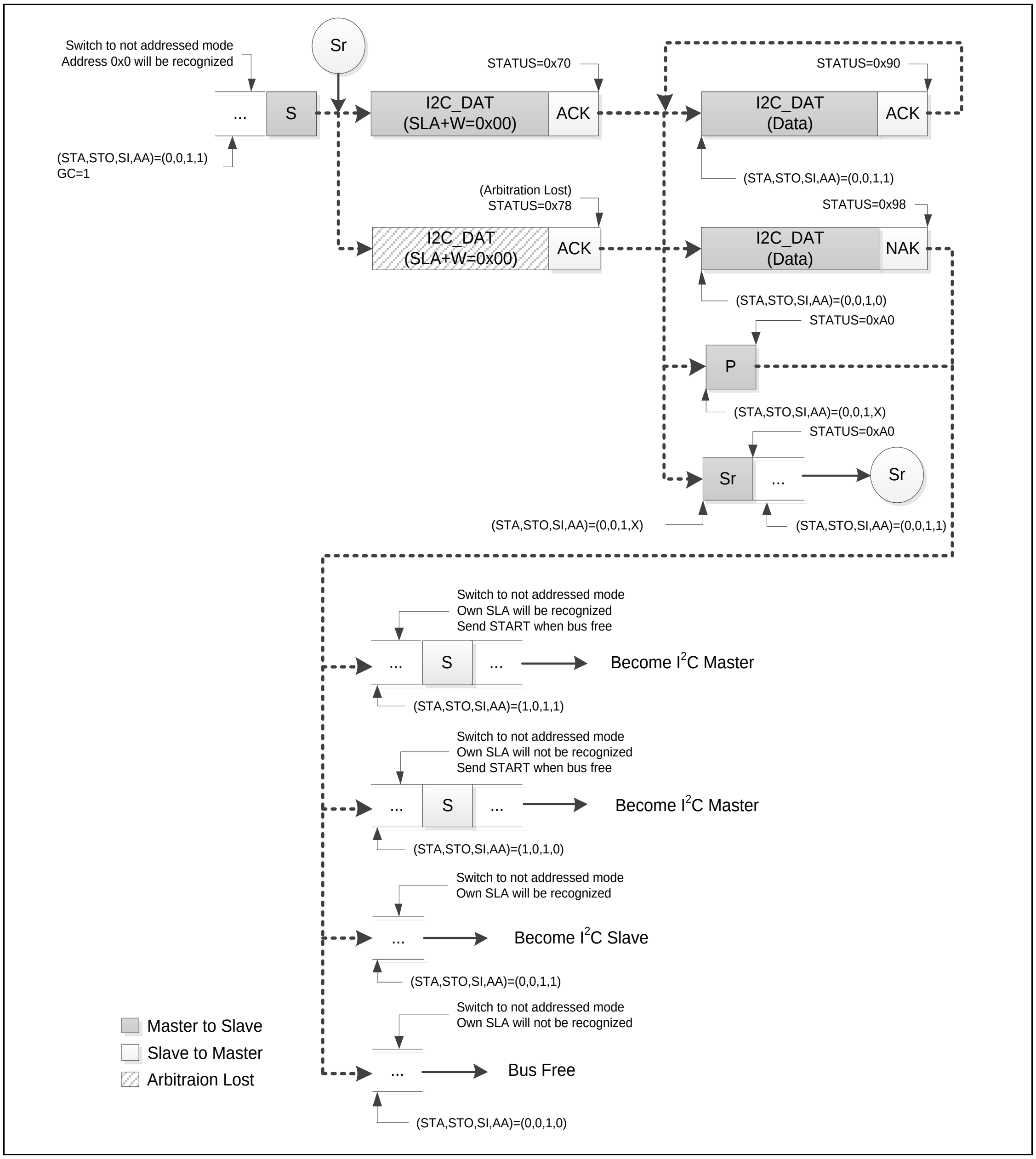

Switch to not addressed mode
Address 0x0 will be recognized
Sr
STATUS=0x70
STATUS=0x90
...
S
I2C_DAT
(SLA+W=0x00)
ACK
I2C_DAT
(Data)
ACK
(STA,STO,SI,AA)=(0,0,1,1)
GC=1
(STA,STO,SI,AA)=(0,0,1,1)
(Arbitration Lost)
STATUS=0x78
STATUS=0x98
I2C_DAT
(SLA+W=0x00)
ACK
I2C_DAT
(Data)
NAK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xA0
P
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xA0
Sr
...
Sr
(STA,STO,SI,AA)=(0,0,1,X)
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will be recognized
Send START when bus free
...
S
...
Become I²C Master
(STA,STO,SI,AA)=(1,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
Send START when bus free
...
S
...
Become I²C Master
(STA,STO,SI,AA)=(1,0,1,0)
Switch to not addressed mode
Own SLA will be recognized
...
Become I²C Slave
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
...
Bus Free
(STA,STO,SI,AA)=(0,0,1,0)
Master to Slave
Slave to Master
Arbitraion Lost

Figure 6.17-13 GC Mode

If I^2C 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 Figure 6.17-13 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^2C signal or address from master. At this time, I^2C controller should enter into idle mode.

Multi-Master

In some applications, there are two or more masters on the same I^2C bus to access slaves, and the masters may transmit data simultaneously. The I^2C 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.

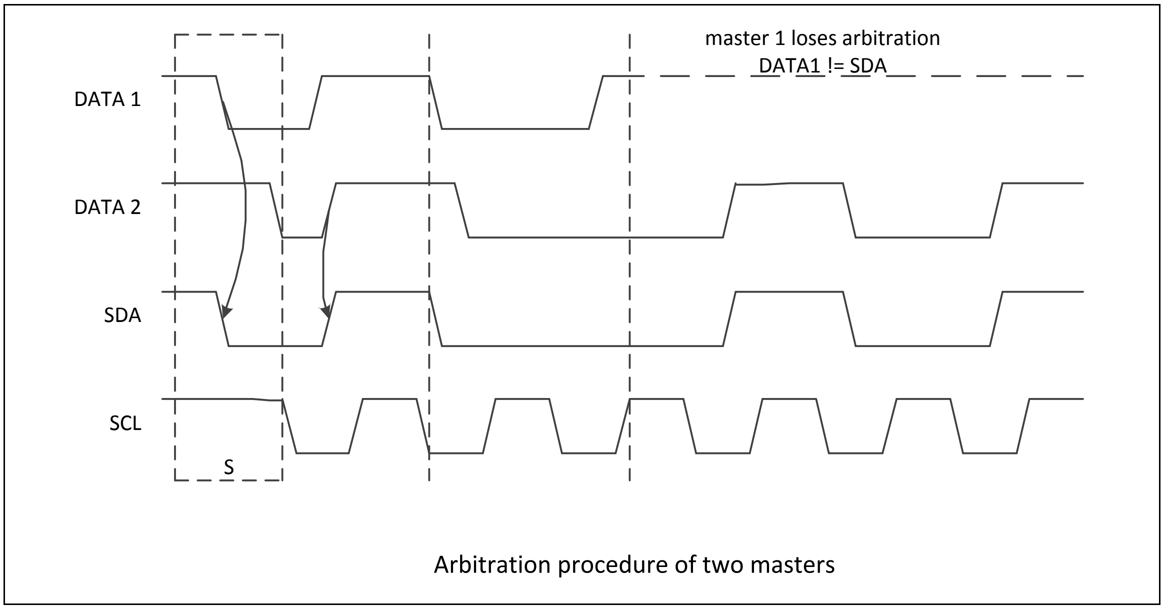

master 1 loses arbitration
DATA1 != SDA
DATA 1
DATA 2
SDA
SCL
S
Arbitration procedure of two masters

Figure 6.17-14 Arbitration Lost

- When I2C_STATUS = 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 not addressed Slave mode. User can detect bus free by ONBUSY (I2C_STATUS1 [8]).

- When I2C_STATUS = 0x00, a “Bus Error” is received. To recover I^2C bus from a bus

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

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

6.17.5.3 PDMA Transfer Function

I^2C controller supports PDMA transfer function.

When TXPDMAEN (I2C_CTL1 [0]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically. When RXPDMAEN (I2C_CTL1 [1]) is set to 1, the I^2C controller will start the receive PDMA process. I^2C controller will issue the request to PDMA controller automatically when there is data written into the received BUFFER or the status of RXEMPTY (I2C_STATUS1 [5]) is set to 0.

When I^2C enter PDMA mode, the mostly status interrupt will be masked. Let the interrupt not occur besides the bus error or NACK or STOP interrupt (0x20, 0x30, 0x38, 0x48, 0x58, 0x00, 0xA0, 0xC0, 0x88 and 0x98).

Set the PDMASTR (I2C_CTL1 [8]) only the I^2C controller in master TX mode. If PDMASTR is cleared to 0, I^2C will send STOP automatically after PDMA transfer done and buffer empty. If PDMASTR is set to 1, SI will be set to 1 and I^2C bus will be stretched by hardware after PDMA transfer done and buffer empty.

6.17.5.4 Two-level Buffer Mode on I^2C bus

Set to enable the two-level buffer for I^2C transmitted or received buffer. It is used to improve the performance of the I^2C bus. If this TWOBUFEN bit is set = 1, the control bit of STA for repeat start or STO bit should be set as normal operation.

For example: if there are 4 data shall be transmitted and then stop it. The STO bit shall be set after the 4rd data's SI event being clear. In this time, the 4th data can be transmitted and the I^2C stop after the 4th data transmission done. The STOP signal will occur after the 4th data's SI event being clear.

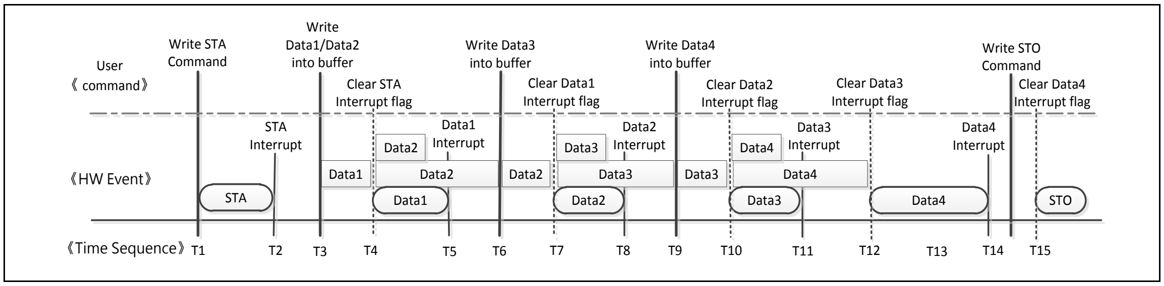


Figure 6.17-15 Timing of Two-level Buffer Transmit in Master Write

For example: if there are 4 data shall be received in Slave mode. The controller can receives the 2nd data in the I^2C bus after the 1st data had been loaded into the received buffer and the user can read the 1st data after the 1st interrupt status be cleared.

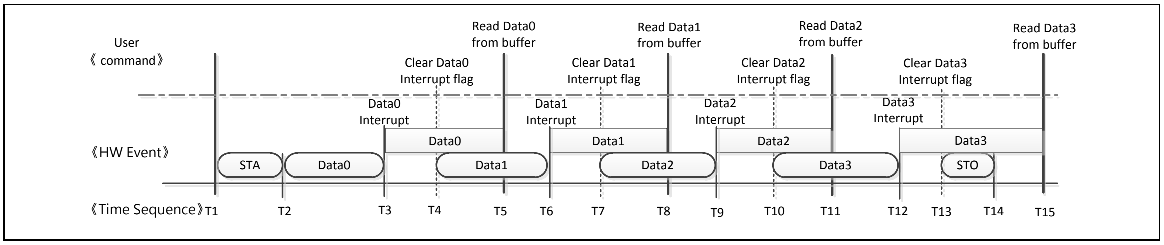


Figure 6.17-16 Timing of Two-level Buffer Transmit in Slave Read

If the buffer is not empty after the bus stop (STO), the user can set the TWOBUFEN bit or TWOBUFRST to ‘0’ to make the buffer control state machine enter idle state and the buffer status will be set to the default value.

6.17.5.5 Programmable setup and hold times

In order to guarantee a correct data setup and hold time, the timing must be configured. By programming HTCTL [5:0] (I2C_TMCTL[11:6]) to configure hold time and STCTL [5:0] (I2C_TMCTL[5:0]) to configure setup time.

The delay timing refer peripheral clock (PCLK). When device stretch master clock, the setup and hold time configuration value will not affected by stretched.

User should focus the limitation of setup and hold time configuration, the timing setting must follow I^2C protocol. Once setup time configuration greater than design limitation, that means if setup time setting make SCL output less than three PCLKs, I^2C controller can’t work normally due to SCL must sample three times. Once hold time configuration is greater than I^2C clock limitation, I^2C will occur bus error. Suggest that user calculate suitable timing with baud rate and protocol before setting timing. Table 6.17-1 shows the relationship between I^2C baud rate and PCLK, the number of table represent one clock duty contain how many PCLKs. Setup and hold time configuration even can program some extreme values in the design, but user should follow I^2C protocol standard.

I^2C Baud Rate **PCLK**	**100k**	**200k**	**400k**
12 MHz	120	60	30
24 MHz	240	120	60
48 MHz	480	240	120
72 MHz	720	360	180

Table 6.17-1 Relationship between I^2C Baud Rate and PCLK

For setup time wrong adjustment example, we assume one SCL cycle contains 5 PCLKs and set STCTL [5:0] (I2C_TMCTL[5:0]) to 3 that stretch three PCLKs for setup time setting. The setup time setting limitation: ST_{limit} = (I2C_CLKDIV[7:0]+1) X 2 - 6.

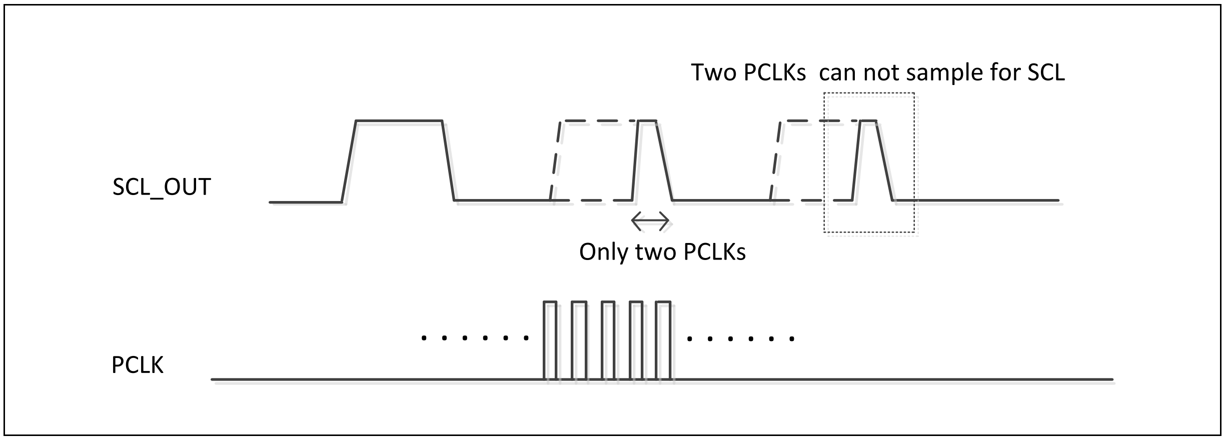


Figure 6.17-17 Setup Time Wrong Adjustment

For hold time wrong adjustment example, we use I^{2}C Baud Rate = 400k and PCLK = 24MHz, the SCL high/low duty = 60 PCLK. When we set HTCTL [5:0] (I2C_TMCTL[11:6]) to 61 and STCTL [5:0] (I2C_TMCTL[5:0]) to 0, then SDA output delay will over SCL high duty and cause bus error. The hold time setting limitation: HT_{limit} = (I2C_CLKDIV[7:0]+1) X 2 - 9.

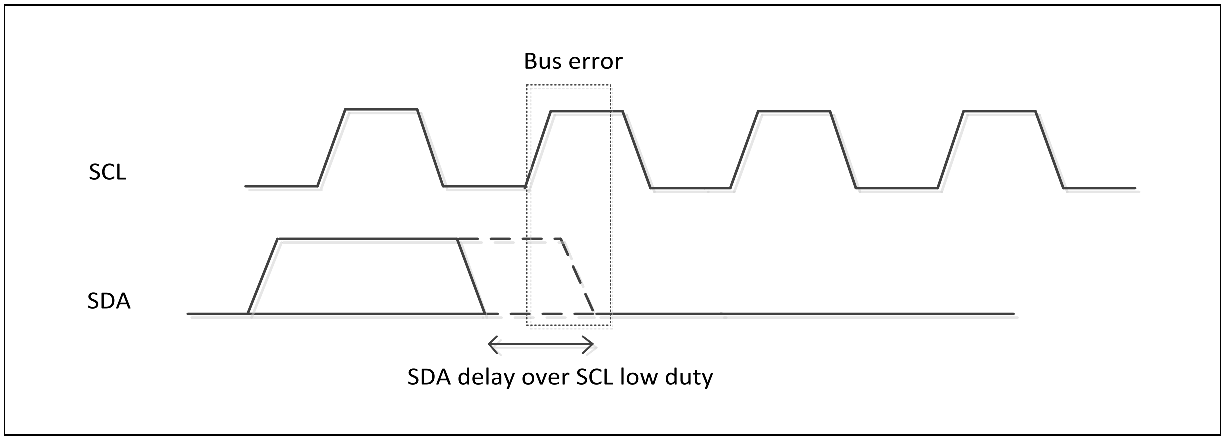


Figure 6.17-18 Hold Time Wrong Adjustment

6.17.5.6 I^{2}C Protocol Registers

To control I^{2}C port through the following fifteen special function registers: I2C_CTL (control register), I2C_STATUS (status register), I2C_DAT (data register), I2C_ADDRn (address registers, n=0~3), I2C_ADDRMSKn (address mask registers, n=0~3), I2C_CLKDIV (clock civided register), I2C_TOCTL (Time-out control register), I2C_WKCTL(wake up control register) and I2C_WKSTS(wake up status register).

Address Registers (I2C_ADDR)

The I^{2}C port is equipped with four slave address registers, I2C_ADDRn (n=0~3). The contents of the register are irrelevant when I^{2}C is in Master mode. In Slave mode, the bit field ADDR(I2C_ADDRn[7:1]) must be loaded with the chip's own slave address. The I^{2}C hardware will react if the contents of I2C_ADDRn are matched with the received slave address.

The I^{2}C ports support the “General Call” function. If the GC bit (I2C_ADDRn [0]) is set the I^{2}C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.

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

Slave Address Mask Registers (I2C_ADDRMSK)

The I^{2}C bus controller supports multiple address recognition with four address mask registers I2C_ADDRMSKn (n=0~3). When the bit in the address mask register is set to 1, it means the received corresponding address bit is "Don’t care". If the bit is set to 0, it means the received corresponding register bit should be exactly the same as address register.

Data Register (I2C_DAT)

This register contains a byte of serial data to be transmitted or a byte which just has been received. The CPU can be read from or written to the 8-bit (I2C_DAT [7:0]) directly while it is not in the process of shifting a byte. When I^{2}C is in a defined state and the serial interrupt flag (SI) is set, data in I2C_DAT [7:0] remains stable. While data is being shifted out, data on the bus is simultaneously being shifted in; I2C_DAT [7:0] always contains the last data byte presented on the bus.

The acknowledge bit is controlled by the I^{2}C hardware and cannot be accessed by the CPU. Serial data is shifted into I2C_DAT [7:0] on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into I2C_DAT [7:0], the serial data is available in I2C_DAT [7:0], and the acknowledge bit (ACK or NACK) is returned by the control logic during the ninth clock pulse. In order to monitor bus status while sending data, the bus data will be shifted to I2C_DAT[7:0] when sending I2C_DAT[7:0] to bus. In the case of sending data, serial data bits are shifted out from I2C_DAT [7:0] on the falling edge of SCL clocks, and is shifted to I2C_DAT [7:0] on the rising edge of SCL clocks.

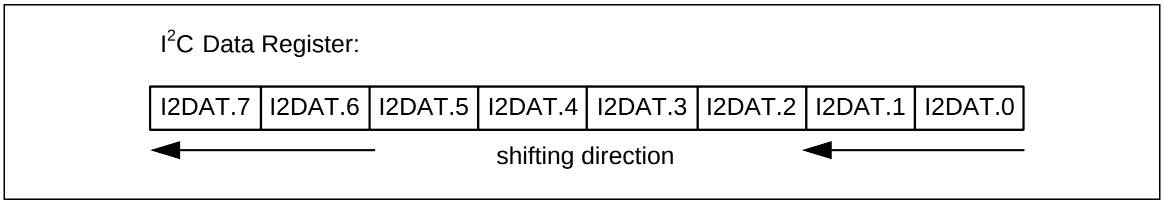


Figure 6.17-19 I^{2}C Data Shifting Direction

Control Register (I2C_CTL)

The CPU can be read from and written to I2C_CTL [7:0] directly. When the I^{2}C port is enabled by setting I2CEN (I2C_CTL [6]) to high, the internal states will be controlled by I2C_CTL and I^{2}C logic hardware.

There are two bits are affected by hardware: the SI bit is set when the I^{2}C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when I2CEN = 0.

Once a new status code is generated and stored in I2C_STATUS, the I^{2}C Interrupt Flag bit SI

(I2C_CTL [3]) will be set automatically. If the Enable Interrupt bit INTEN (I2C_CTL [7]) is set at this time, the I^{2}C interrupt will be generated. The bit field I2C_STATUS[7:0] stores the internal state code, the content keeps stable until SI is cleared by software.

Status Register (I2C_STATUS)

I2C_STATUS [7:0] is an 8-bit read-only register. The bit field I2C_STATUS [7:0] contains the status code and there are 26 possible status codes. All states are listed in Table 6.17-2. When I2C_STATUS [7:0] is F8H, no serial interrupt is requested. All other I2C_STATUS [7:0] values correspond to the defined I^{2}C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2C_STATUS[7:0] one cycle PCLK after SI set by hardware and is still present one cycle PCLK after SI reset by software.

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

Master Mode		**Slave Mode**	
STATUS	**Description**	**STATUS**	**Description**
0x08[1]	Start	0xA0	Slave Transmit Repeat Start or Stop
0x10[1]	Master Repeat Start	0xA8[1]	Slave Transmit Address ACK
0x18[1]	Master Transmit Address ACK	0xB8[1]	Slave Transmit Data ACK
0x20	Master Transmit Address NACK	0xC0	Slave Transmit Data NACK
0x28[1]	Master Transmit Data ACK	0xC8[1]	Slave Transmit Last Data ACK
0x30	Master Transmit Data NACK	0x60[1]	Slave Receive Address ACK
0x38	Master Arbitration Lost	0x68[1]	Slave Receive Arbitration Lost
0x40[1]	Master Receive Address ACK	0x80[1]	Slave Receive Data ACK
0x48	Master Receive Address NACK	0x88	Slave Receive Data NACK
0x50[1]	Master Receive Data ACK	0x70[1]	GC mode Address ACK
0x58	Master Receive Data NACK	0x78[1]	GC mode Arbitration Lost
0x00	Bus error	0x90[1]	GC mode Data ACK
		0x98	GC mode Data NACK
		0xB0[1]	Address Transmit Arbitration Lost
0xF8	Bus Released **Note:** Status "0xF8" exists in both master/slave modes, and it won't raise interrupt. **Note [1]:** No interrupt in PDMA mode.		

Table 6.17-2 I^{2}C Status Code Description

Clock Divided Register (I2C_CLKDIV)

The data baud rate of I^{2}C is determines by DIVIDER(I2C_CLKDIV [7:0])register when I^{2}C is in

Master Mode, and it is not necessary in a Slave mode. In the Slave mode, I^2C will automatically synchronize it with any clock frequency from master I^2C device. In the slave mode, system clock frequency should greater than I^2C bus maximum clock 20 times.

The data baud rate of I^2C setting is Data Baud Rate of I^2C = (system clock) / (4x (I2C_CLKDIV [7:0] +1)). If system clock = 16 MHz, the I2C_CLKDIV [7:0] = 40 (28H), the data baud rate of I^2C = 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec.

Time-out Control Register (I2C_TOCTL)

There is a 14-bit time-out counter which can be used to deal with the I^2C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows (TOIF=1) and generates I^2C interrupt to CPU or stops counting by clearing TOCEN to 0. When time-out counter is enabled, writing 1 to the SI flag will reset counter and re-start up counting after SI is cleared. If I^2C bus hangs up, it causes the I2C_STATUS and flag SI are not updated for a period, the 14-bit time-out counter may overflow and acknowledge CPU the I^2C interrupt. Set TOCDIV4 (I2C_TOCTL[1]) that time-out period is extend 4 times. Refer to Figure 6.17-20 for the 14-bit time-out counter. User may write 1 to clear TOIF to 0.

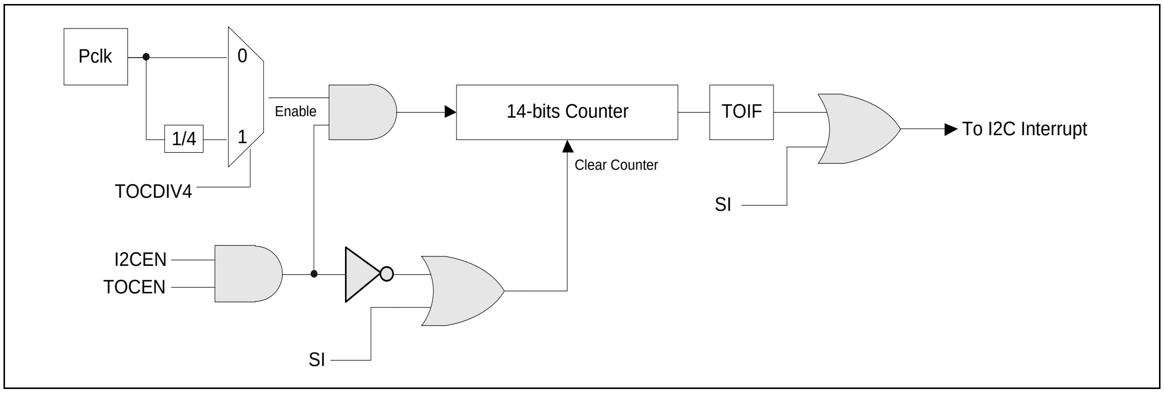


Figure 6.17-20 I^2C Time-out Count Block Diagram

Wake-up Control Register (I2C_WKCTL)

When chip enters Power-down mode and sets WKEN (WKCON [0]) to 1, other I^2C master can wake up the chip by addressing the I^2C device, user must configure the related setting before entering sleep mode. The ACK bit cycle of address match frame is done in power-down. The controller will stretch the SCL to low when the address is matched the device's address and the ACK cycle done, then I^2C controller will go ahead. If NHDBUSEN (WKCON [7]) is set, the controller will don't stretch the SCL to low. Note that when the controller does not stretch the SCL to low, transmit or receive data will perform immediately. If data transmitted or received when SI event is not clear, user must reset I^2C controller and execute the original operation again.

Wake-up Status Register (I2C_WKSTS)

When system is woken up by other I^2C master device, WKIF (I2C_WKSTS [0]) is set to indicate this event. User needs write "1" to clear this bit.

When the chip is woken-up by address match with one of the device address register (I2C_ADDRn), the user shall check the WKAKDONE (I2C_WKSTS [1]) bit is set to 1 to confirm the address byte has done. The WKAKDONE bit indicates that the ACK bit cycle of address byte is done in power-down. The controller will stretch the SCL to low when the address is matched the device's slave address and the ACK cycle done. The SCL is stretched until WKAKDONE is clear by user. If the frequency of SCL is low speed and the system has wakeup from address match frame, the user shall check WKAKDONE to confirm this frame has transaction done and then to do the wakeup procedure. Note that user can not release WKIF through clearing the WKAKDONE bit to 0.

The WRSTSWK (I2C_WKSTS [2]) bit records the Read/Write command before the I^{2}C controller send address. The user can read this bit's status to prepare the next transmitted data (WRSTSWK = 0) or to wait the incoming data (WRSTSWK = 1) can be stored in time after the system is wake-up by the address match frame. Note that the WRSTSWK (I2C_WKSTS [2]) bit is cleared when write one to the WKAKDONE (I2C_WKSTS [1]) bit.

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

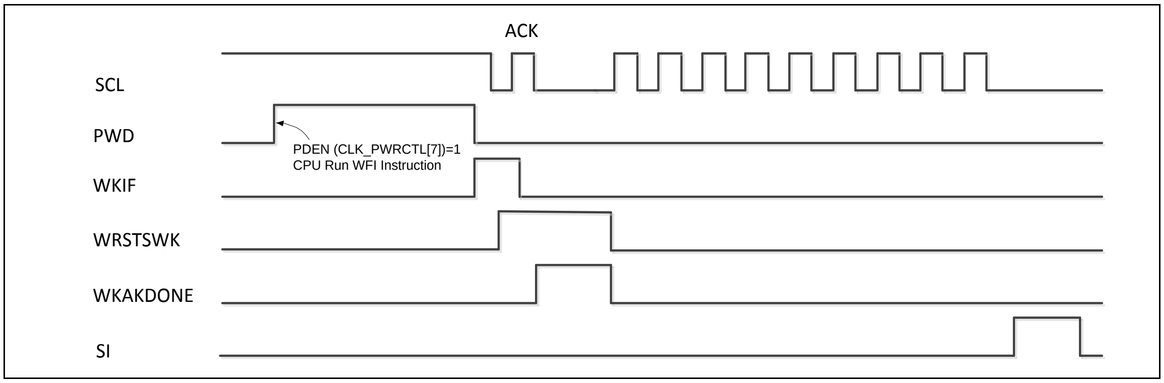


Figure 6.17-21 I^{2}C Wake-Up Related Signals Waveform

I^{2}C Control Register 1 (I2C_CTL1)

For the TWOBUFEN (I2C_CTL1[5]) bit, it is used to enable the two-level buffer for I^{2}C transmitted or received buffer. It is used to improve the performance of the I^{2}C bus.

Set UDRIEN (I2C_CTL1[4]) or OVRIEN (I2C_CTL1[3]) that interrupt will occur when two-level buffer generate the under run or over run event. It is worth noting when buffer empty, it will pull the bus. When I^{2}C transmit or receive STOP, the data in buffer will be cleared. If user want to reset two-level buffer, then set the TWOBUFRST (I2C_CTL1 [6]) to clear buffer. Set NSTRETCH (I2C_CTL1[7]) that I^{2}C SCL bus is not stretched by hardware if the SI is not cleared in master mode.

For PDMA function, set TXPDMAEN (I2C_CTL1 [0]) and RXPDMAEN (I2C_CTL1 [1]) can be set to operate. Also, set PDMARST (I2C_CTL1 [2]) to reset the PDMA control logic.

I^{2}C Status Register 1 (I2C_STATUS1)

The two-level buffer status, busy free information, EMPTY(I2C_STATUS1 [5]), FULL(I2C_STATUS1 [4]) and UDR(I2C_STATUS1 [7]) or OVR(I2C_STATUS1 [6]) are also list in the this register.

I²C Timing Configure Control Register (I2C_TMCTL)

In order to configure setup/hold time, the HTCTL [5:0] (I2C_TMCTL[11:6]) and STCTL [5:0] (I2C_TMCTL[5:0]) are set based on actual demand.

6.17.5.7 Example for Random Read on EEPROM

The following steps are used to configure the I²C0 related registers when using I²C to read data from EEPROM.

1. Set I²C0 the multi-function pin in the SYS_GPA_MFPL or SYS_GPD_MFPL or SYS_GPE_MFPL or SYS_GPA_MFPH registers as SCL and SDA pins.
2. Enable I²C0 APB clock, I2C0CKEN=1 in the “CLK_APBCLK0” register.
3. Set I2C0RST=1 to reset I²C0 controller then set I²C0 controller to normal operation, I2C0RST=0 in the “SYS_IPRST1” register.
4. Set I2CEN=1 to enable I²C0 controller in the “I2C_CTL” register.
5. Give I²C0 clock a divided register value for I²C clock rate in the “I2C_CLKDIV”.
6. Set SETENA=0x00000040 in the “NVIC_ISER2” register to set I²C0 IRQ.
7. Set INTEN=1 to enable I²C0 Interrupt in the “I2C_CTL” register.
8. Set I²C0 address registers “I2C_ADDR0 ~ I2C_ADDR3”.

Random read operation is one of the methods of access EEPROM. The method allows the master to access any address of EEPROM space. Figure 6.17-22 shows the EEPROM random read operation.

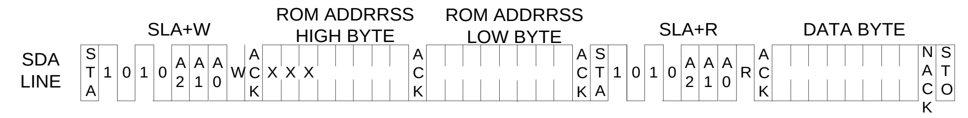


Figure 6.17-22 EEPROM Random Read

Figure 6.17-23 shows how to use I²C controller to implement the protocol of EEPROM random read.

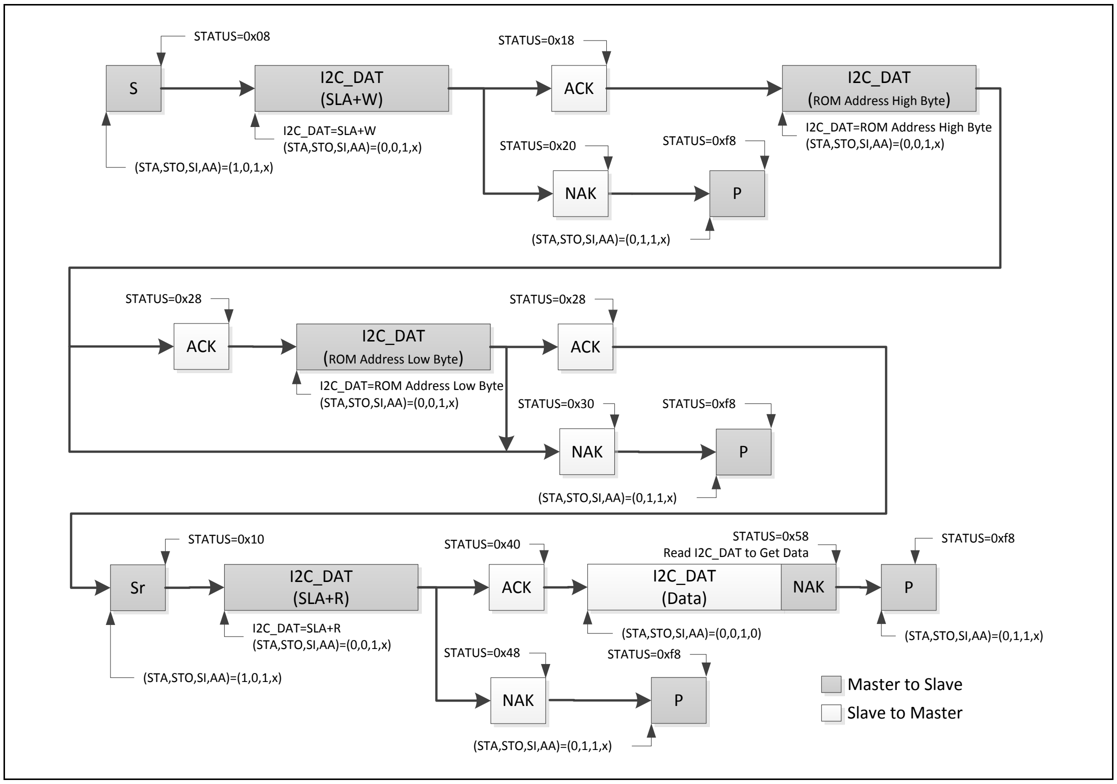


Figure 6.17-23 Protocol of EEPROM Random Read

The I^{2}C controller, which is a master, sends START to bus. Then, it sends a SLA+W (Slave address + Write bit) to EERPOM followed by two bytes data address to set the EEPROM address to read. Finally, a Repeat START followed by SLA+R is sent to read the data from EEPROM.

6.17.6 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 + (0x10_0000 *n)** **n= 0,1**				
I2C_CTL	I2Cn_BA+0x00	R/W	I^{2}C Control Register 0	0x0000_0000
I2C_ADDR0	I2Cn_BA+0x04	R/W	I^{2}C Slave Address Register0	0x0000_0000
I2C_DAT	I2Cn_BA+0x08	R/W	I^{2}C Data Register	0x0000_0000
I2C_STATUS	I2Cn_BA+0x0C	R	I^{2}C Status Register 0	0x0000_00F8
I2C_CLKDIV	I2Cn_BA+0x10	R/W	I^{2}C Clock Divided Register	0x0000_0000
I2C_TOCTL	I2Cn_BA+0x14	R/W	I^{2}C Time-out Control Register	0x0000_0000
I2C_ADDR1	I2Cn_BA+0x18	R/W	I^{2}C Slave Address Register1	0x0000_0000
I2C_ADDR2	I2Cn_BA+0x1C	R/W	I^{2}C Slave Address Register2	0x0000_0000
I2C_ADDR3	I2Cn_BA+0x20	R/W	I^{2}C Slave Address Register3	0x0000_0000
I2C_ADDRMSK0	I2Cn_BA+0x24	R/W	I^{2}C Slave Address Mask Register0	0x0000_0000
I2C_ADDRMSK1	I2Cn_BA+0x28	R/W	I^{2}C Slave Address Mask Register1	0x0000_0000
I2C_ADDRMSK2	I2Cn_BA+0x2C	R/W	I^{2}C Slave Address Mask Register2	0x0000_0000
I2C_ADDRMSK3	I2Cn_BA+0x30	R/W	I^{2}C Slave Address Mask Register3	0x0000_0000
I2C_WKCTL	I2Cn_BA+0x3C	R/W	I^{2}C Wake-up Control Register	0x0000_0000
I2C_WKSTS	I2Cn_BA+0x40	R/W	I^{2}C Wake-up Status Register	0x0000_0000
I2C_CTL1	I2Cn_BA+0x44	R/W	I^{2}C Control Register 1	0x0000_0000
I2C_STATUS1	I2Cn_BA+0x48	R	I^{2}C Status Register 1	0x0000_0000
I2C_TMCTL	I2Cn_BA+0x4C	R/W	I2C Timing Configure Control Register	0x0000_0000

6.17.7 Register Description

I^{2}C Control Register (I2C_CTL)

Register	Offset	R/W	Description	Reset Value
I2C_CTL	I2Cn_BA+0x00	R/W	I^{2}C Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTEN	**I2CEN**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**INTEN**	**Enable Interrupt** 0 = I^{2}C interrupt Disabled. 1 = I^{2}C interrupt Enabled.
[6]	**I2CEN**	**I^{2}C Controller Enable Bit** Set to enable I^{2}C serial function controller. When I2CEN=1 the I^{2}C serial function enable. The multi-function pin function must set to SDA, and SCL of I^{2}C function first. 0 = I^{2}C controller Disabled. 1 = I^{2}C controller Enabled.
[5]	**STA**	**I^{2}C START Control** Setting STA to logic 1 to enter Master mode, the I^{2}C hardware sends a START or repeat START condition to bus when the bus is free.
[4]	**STO**	**I^{2}C STOP Control** In Master mode, setting STO to transmit a STOP condition to bus then I^{2}C controller will check the bus condition if a STOP condition is detected. This bit will be cleared by hardware automatically.
[3]	**SI**	**I^{2}C Interrupt Flag** When a new I^{2}C state is present in the I2C_STATUS register, the SI flag is set by hardware. If bit INTEN (I2C_CTL [7]) is set, the I^{2}C interrupt is requested. SI must be cleared by software. Clear SI by writing 1 to this bit.
[2]	**AA**	**Assert Acknowledge Control** 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.

I^{2}C Data Register (I2C_DAT)

Register	Offset	R/W	Description	Reset Value
I2C_DAT	I2Cn_BA+0x08	R/W	I^{2}C Data Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DAT**	**I^{2}C Data** Bit [7:0] is located with the 8-bit transferred/received data of I^{2}C serial port.

I^{2}C Status Register (I2C_STATUS)

Register	Offset	R/W	Description	Reset Value
I2C_STATUS	I2Cn_BA+0x0C	R	I^{2}C Status Register 0	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**
STATUS							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**STATUS**	**I^{2}C Status** 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 I2C_STATUS is F8H, no serial interrupt is requested. Others I2C_STATUS values correspond to defined I^{2}C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2C_STATUS 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 (I2C_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2C_CLKDIV	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**
DIVIDER							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DIVIDER**	**I²C Clock Divided** Indicates the I²C clock rate: Data Baud Rate of I²C = (system clock) / (4x (I2C_CLKDIV+1)). **Note:** The minimum value of I2C_CLKDIV is 4.

I^{2}C Time-out Control Register (I2C_TOCTL)

Register	Offset	R/W	Description	Reset Value
I2C_TOCTL	I2Cn_BA+0x14	R/W	I^{2}C Time-out 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					**TOCEN**	**TOCDIV4**	**TOIF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**TOCEN**	**Time-out Counter Enable Bit** When Enabled, the 14-bit time-out counter will start counting when SI is clear. Setting 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]	**TOCDIV4**	**Time-out Counter Input Clock Divided by 4** When it Enabled, The time-out period is extend 4 times. 0 = Time-out period is extend 4 times Disabled. 1 = Time-out period is extend 4 times Enabled.
[0]	**TOIF**	**Time-out Flag** This bit is set by hardware when I^{2}C time-out happened and it can interrupt CPU if I^{2}C interrupt enable bit (INTEN) is set to 1. **Note:** Software can write 1 to clear this bit.

I^{2}C Slave Address Register (ADDRx)

Register	Offset	R/W	Description	Reset Value
I2C_ADDR0	I2Cn_BA+0x04	R/W	I^{2}C Slave Address Register0	0x0000_0000
I2C_ADDR1	I2Cn_BA+0x18	R/W	I^{2}C Slave Address Register1	0x0000_0000
I2C_ADDR2	I2Cn_BA+0x1C	R/W	I^{2}C Slave Address Register2	0x0000_0000
I2C_ADDR3	I2Cn_BA+0x20	R/W	I^{2}C Slave Address Register3	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**ADDR**	**I^{2}C Address** The content of this register is irrelevant when I^{2}C is in Master mode. In the slave mode, the seven most significant bits must be loaded with the chip's own address. The I^{2}C hardware will react if either of the address is matched.
[0]	**GC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled.

I^{2}C Slave Address Mask Register (ADDRMSKx)

Register	Offset	R/W	Description	Reset Value
I2C_ADDRMSK0	I2Cn_BA+0x24	R/W	I^{2}C Slave Address Mask Register0	0x0000_0000
I2C_ADDRMSK1	I2Cn_BA+0x28	R/W	I^{2}C Slave Address Mask Register1	0x0000_0000
I2C_ADDRMSK2	I2Cn_BA+0x2C	R/W	I^{2}C Slave Address Mask Register2	0x0000_0000
I2C_ADDRMSK3	I2Cn_BA+0x30	R/W	I^{2}C Slave Address Mask Register3	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**ADDRMSK**	**I^{2}C Address Mask** 0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don't care.). I^{2}C 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.

I^{2}C Wake-up Control Register (I2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
I2C_WKCTL	I2Cn_BA+0x3C	R/W	I^{2}C Wake-up Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**NHDBUSEN**	**I^{2}C No Hold BUS Enable Bit** 0 = I^{2}C don’t hold bus after wake-up disable. 1= I^{2}C don’t hold bus after wake-up enable. **Note:** I^{2}C controller could response when WKIF event is not clear, it may cause error data transmitted or received. If data transmitted or received when WKIF event is not clear, user must reset I^{2}C controller and execute the original operation again.
[6:1]	**Reserved**	Reserved.
[0]	**WKEN**	**I^{2}C Wake-up Enable Bit** 0 = I^{2}C wake-up function Disabled. 1= I^{2}C wake-up function Enabled.

I^{2}C Wake-up Status Register (I2C_WKSTS)

Register	Offset	R/W	Description	Reset Value
I2C_WKSTS	I2Cn_BA+0x40	R/W	I^{2}C 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					**WRSTSWK**	**WKAKDONE**	**WKIF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**WRSTSWK**	**Read/Write Status Bit in Address Wakeup Frame** 0 = Write command be record on the address match wakeup frame. 1 = Read command be record on the address match wakeup frame. **Note:** This bit will be cleared when software can write 1 to WKAKDONE bit.
[1]	**WKAKDONE**	**Wakeup Address Frame Acknowledge Bit Done** 0 = The ACK bit cycle of address match frame isn't done. 1 = The ACK bit cycle of address match frame is done in power-down. **Note:** This bit can't release WKIF. Software can write 1 to clear this bit.
[0]	**WKIF**	**I^{2}C Wake-up Flag** When chip is woken up from Power-down mode by I^{2}C, this bit is set to 1. Software can write 1 to clear this bit.

I^2C Control Register 1 (I2C_CTL1)

Register	Offset	R/W	Description	Reset Value
I2C_CTL1	I2Cn_BA+0x44	R/W	I^2C 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							**PDMASTR**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NSTRETCH	**TWOBUFRST**	**TWOBUFEN**	**UDRIEN**	**OVRIEN**	**PDMARST**	**RXPDMAEN**	**TXPDMAEN**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**PDMASTR**	**PDMA Stretch Bit** 0 = I^2C sends STOP automatically after PDMA transfer done. (only master TX) 1 = I^2C SCL bus is stretched by hardware after PDMA transfer done if the SI is not cleared. (only master TX)
[7]	**NSTRETCH**	**No Stretch on the I^2C Bus** 0 = The I^2C SCL bus is stretched by hardware if the SI is not cleared in master mode. 1 = The I^2C SCL bus is not stretched by hardware if the SI is not cleared in master mode.
[6]	**TWOBUFRST**	**Two-level Buffer Reset** 0 = No effect. 1 = Reset the related counters, two-level buffer state machine, and the content of data buffer.
[5]	**TWOBUFEN**	**Two-level Buffer Enable Bit** 0 = Two-level buffer Disabled. 1 = Two-level buffer Enabled. Set to enable the two-level buffer for I^2C transmitted or received buffer. It is used to improve the performance of the I^2C bus. If this bit is set = 1, the control bit of STA for repeat start or STO bit should be set after the current SI is clear. For example: if there are 4 data shall be transmitted and then stop it. The STO bit shall be set after the 3rd data's SI event being clear. In this time, the 4th data can be transmitted and the I^2C stop after the 4th data transmission done.
[4]	**UDRIEN**	**I^2C Under Run Interrupt Control Bit** Setting UDRIEN to logic 1 will send a interrupt to system when the TWOBUFEN bit is enabled and there is under run event happened in transmitted buffer.
[3]	**OVRIEN**	**I^2C over Run Interrupt Control Bit** Setting OVRIEN to logic 1 will send a interrupt to system when the TWOBUFEN bit is enabled and there is over run event in received buffer.

[2]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the PDMA control logic. This bit will be cleared to 0 automatically.
[1]	**RXPDMAEN**	**PDMA Receive Channel Available** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[0]	**TXPDMAEN**	**PDMA Transmit Channel Available** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled.

I^{2}C Status Register 1 (I2C_STATUS1)

Register	Offset	R/W	Description	Reset Value
I2C_STATUS1	I2Cn_BA+0x48	R	I^{2}C Status 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							**ONBUSY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UDR	**OVR**	**EMPTY**	**FULL**	**Reserved**			

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**ONBUSY**	**Bus Busy** Indicates that a communication is in progress on the bus. It is set by hardware when a START condition is detected. It is cleared by hardware when a STOP condition is detected. 0 = The bus is IDLE (both SCLK and SDA High). 1 = The bus is busy.
[7]	**UDR**	**I^{2}C Under Run Status Bit** This bit indicates the transmitted two-level buffer TX or RX is under run when the TWOBUFEN = 1.
[6]	**OVR**	**I^{2}C over Run Status Bit** This bit indicates the received two-level buffer TX or RX is over run when the TWOBUFEN = 1.
[5]	**EMPTY**	**Two-level Buffer Empty** This bit indicates two-level buffer TX or RX empty or not when the TWOBUFEN = 1. This bit is set when buffer empty.
[4]	**FULL**	**Two-level Buffer Full** This bit indicates two-level buffer TX or RX full or not when the TWOBUFEN = 1. This bit is set when buffer full.
[3:0]	**Reserved**	Reserved.

I^{2}C Timing Configure Control Register (I2C_TMCTL)

Register	Offset	R/W	Description	Reset Value
I2C_TMCTL	I2Cn_BA+0x4C	R/W	I^{2}C Timing Configure 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				**HTCTL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HTCTL		**STCTL**					

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:6]	**HTCTL**	**Hold Time Configure Control Register** This field is used to generate the delay timing between SCL falling edge and SDA rising edge in transmission mode. The delay hold time is numbers of peripheral clock = HTCTL x PCLK.
[5:0]	**STCTL**	**Setup Time Configure Control Register** This field is used to generate a delay timing between SDA falling edge and SCL rising edge in transmission mode. The delay setup time is numbers of peripheral clock = STCTL x PCLK. **Note:** Setup time setting should not make SCL output less than three PCLKs.

6.18 Serial Peripheral Interface (SPI)

6.18.1 Overview

The Serial Peripheral Interface (SPI) applies to synchronous serial data communication and allows full duplex transfer. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The NUC121/125 series contains one SPI controller performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. The SPI controller can be configured as a master or a slave device.

This controller also supports the PDMA function to access the data buffer. The SPI controller also supports I^2S mode to connect external audio CODEC.

6.18.2 Features

- SPI Mode
  - One set of SPI controller
  - Supports Master or Slave mode operation
  - Configurable bit length of a transaction word from 8 to 32-bit
  - Provides separate 4-level depth transmit and receive FIFO buffers
  - Supports MSB first or LSB first transfer sequence
  - Supports Byte Reorder function
  - Supports PDMA transfer
  - Supports one data channel half-duplex transfer
  - Support receive-only mode
- I^2S Mode
  - Supports Master or Slave
  - Capable of handling 8-, 16-, 24- and 32-bit word sizes
  - Provides separate 4-level depth transmit and receive FIFO buffers
  - Supports monaural and stereo audio data
  - Supports PCM mode A, PCM mode B, I^2S and MSB justified data format
  - Supports PDMA transfer

6.18.3 Block Diagram

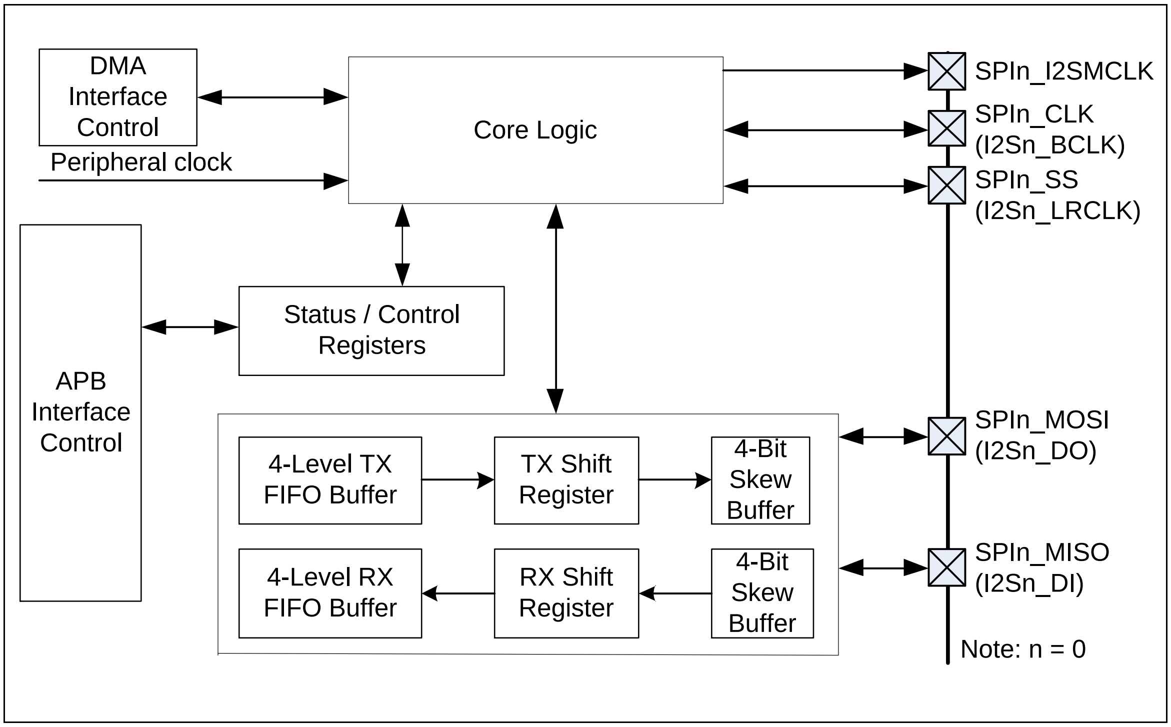


Figure 6.18-1 SPI Block Diagram

TX FIFO Buffer:

The transmit FIFO buffer is an 4-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 a 4-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 shift 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-level 1-bit buffer. For transmitting, it is written from shift register by peripheral clock and read out by SPI bus clock. Three bit data is loaded into this buffer first and the 4th bit data is written into the buffer after 1-bit data is read out by the SPI bus clock. For receiving, the serial data in the bus of SPI_MOSI (in slave mode) is written into the skew buffer based on the SPI bus clock and it is read out and written into the RX shift register by SPI peripheral clock after there is no empty in the RX skew buffer.

6.18.4 Basic Configuration

The basic configurations of SPI0 are as follows:

- SPI0 pins are configured in SYS_GPA_MFPH**,** SYS_GPB_MFPH**,** SYS_GPC_MFPL, SYS_GPC_MFPH, or SYS_GPD_MFPL registers.
- Select the source of SPI0 peripheral clock on SPI0SEL (CLK_CLKSEL2[25:24]).
- Enable SPI0 peripheral clock in SPI0CKEN (CLK_APBCLK0[12]).
- Reset SPI0 controller in SPI0RST (SYS_IPRST1[12]).

SPI/I^{2}S Interface Controller Pin description is shown as follows:

Pin	SPI Mode	I^{2}S Mode
SPI_SS	SPI slave selection pin	I^{2}S left/right channel synchronization clock pin (I2Sn_LRCLK)
SPI_CLK	SPI clock pin	I^{2}S bit clock pin (I2Sn_BCLK)
SPI_MISO	SPI master input or slave output pin	I^{2}S data input pin (I2Sn_DI)
SPI_MOSI	SPI master output or slave input pin	I^{2}S data output pin (I2Sn_DO)
SPI_I2SMCLK	Not available	I^{2}S Master clock output pin

Table 6.18-1 SPI/I^{2}S Interface Controller Pin

6.18.5 Functional Description

6.18.5.1 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 divisor (SPI_CLKDIV) and the clock source which can be HXT, HIRC, PLL out or the PCLK. SPI0SEL of CLK_CLKSEL2 register determines the clock source of the peripheral clock. The DIVIDER (SPI_CLKDIV[7:0]) setting determines the divisor of the clock rate calculation.

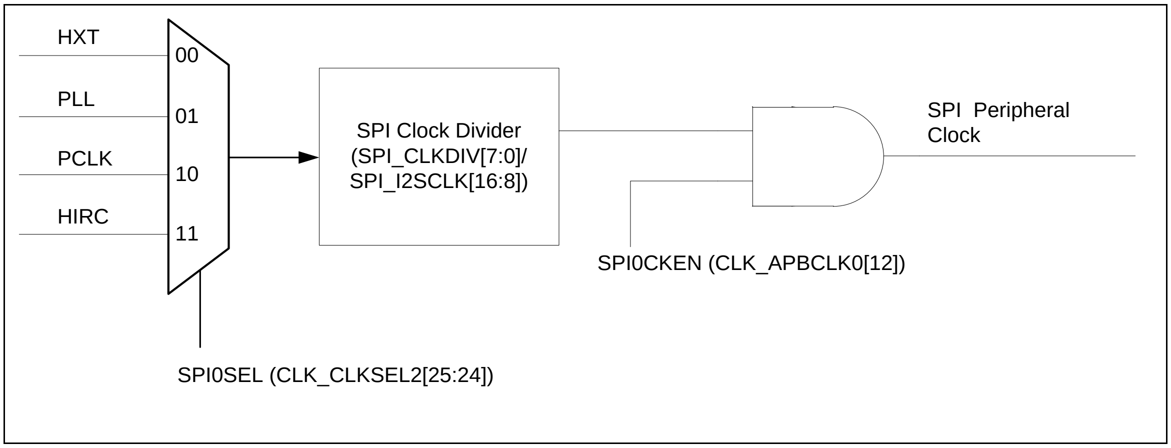


Figure 6.18-2 SPI Peripheral Clock

In Master mode, the frequency of the SPI bus clock is equal to the peripheral clock rate. In general, the SPI bus clock is denoted as SPI clock. In Slave mode, the SPI bus clock is provided by a master device. The frequency of SPI peripheral clock cannot be faster than the system clock rate regardless of Master or Slave mode. If the clock source of peripheral clock is not system clock, the frequency of SPI peripheral clock shall be slower than the system clock frequency regardless of Master or Slave mode.

In I^{2}S mode, the peripheral clock rate is equal to I^{2}S bit clock rate determined by SPI_I2SCLK register.

Master/Slave mode

This SPI controller can be set as Master or Slave mode by setting the SLAVE (SPI_CTL[18]) to communicate with the off-chip SPI slave or master device. The HALFDPX (SPI_CTL[14]) can be used to select the full-duplex or half-duplex in SPI transmission. The application block diagrams in Master and Slave mode are shown below.

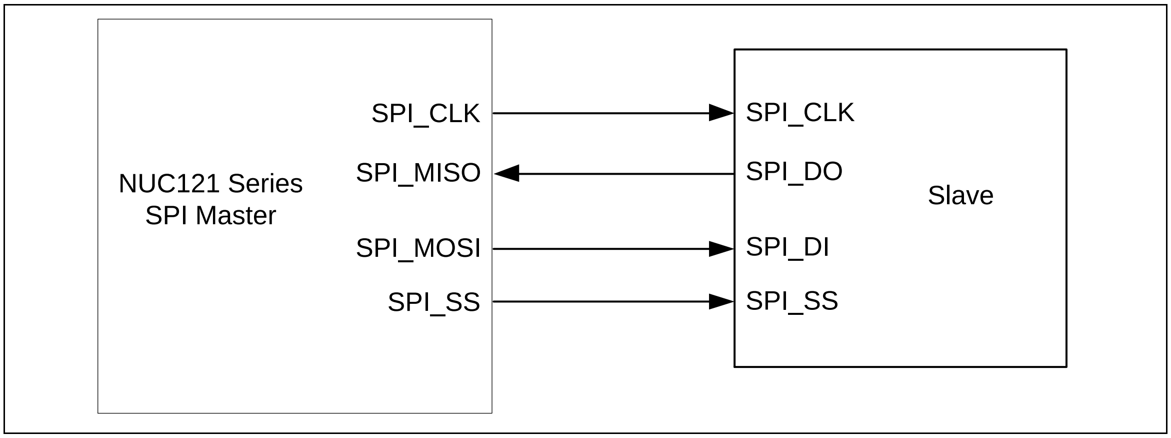


Figure 6.18-3 SPI Full-Duplex Master Mode Application Block Diagram

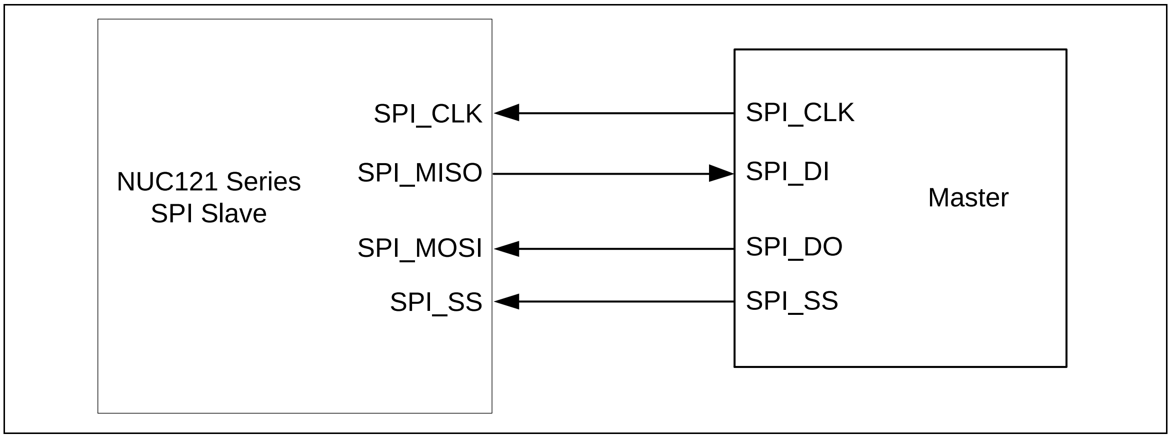


Figure 6.18-4 SPI Full-Duplex Slave Mode Application Block Diagram

Slave Selection

In Master mode, the SPI controller can drive off-chip slave device through the slave select output pin SPI_SS. In Slave mode, the off-chip master device drives the slave selection signal from the SPI_SS input port 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 selection signal can be programmed to low or high active in SSACTPOL (SPI_SSCTL[2]). The selection of slave select conditions depends on what type of device is connected. In Slave mode, 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.

Timing Condition

The CLKPOL (SPI_CTL[3]) defines the SPI clock idle state. If CLKPOL = 1, the output SPI clock is idle at high state; if CLKPOL = 0, it is idle at low state.

TXNEG (SPI_CTL[2]) defines the data transmitted out either on negative edge or on positive edge of SPI clock. RXNEG (SPI_CTL[1]) defines the data received either on negative edge or on positive edge of SPI clock.

Note: The settings of TXNEG and RXNEG 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 DWIDTH (SPI_CTL[12:8]) 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 specific count of bits defined in DWIDTH (SPI_CTL[12:8]), the unit transfer interrupt flag will be set to 1.

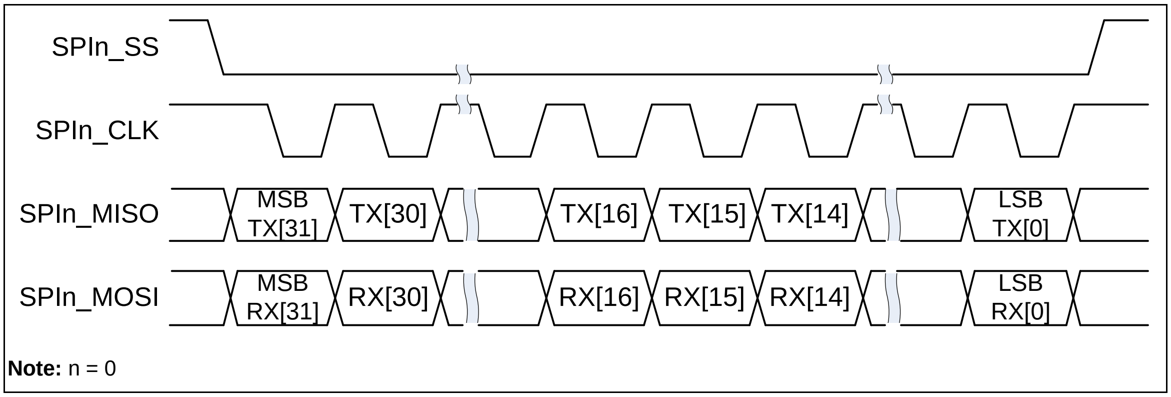


Figure 6.18-532-Bit in One Transaction

LSB/MSB First

LSB (SPI_CTL[13]) defines the bit transfer sequence in a transaction. If the LSB (SPI_CTL[13]) is set to 1, the transfer sequence is LSB first. The bit 0 will be transferred firstly. If the LSB (SPI_CTL[13]) is cleared to 0, the transfer sequence is MSB first.

Suspend Interval

SUSPITV (SPI_CTL[7:4]) provides a configurable suspend interval, 0.5 ~ 15.5 SPI clock periods, between two successive transaction words in Master mode. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SUSPITV is 0x3 (3.5 SPI clock cycles).

6.18.5.2 Automatic Slave Selection

In Master mode, if AUTOSS (SPI_SSCTL[3]) is set, the slave selection signal will be generated automatically and output to the SPI_SS pin according to whether SS (SPI_SSCTL[0]) is enabled or not. The slave selection signal will be set to active state by the SPI controller when the SPI data transfer is started by writing to FIFO. 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 SUSPITV (SPI_CTL[7:4]) is greater than or equal to 3.

In Master mode, if the value of SUSPITV is less than 3 and the AUTOSS is set as 1, the slave selection signal will be kept at active state between two successive transactions.

If the AUTOSS bit is cleared, the slave selection output signal will be determined by the SS setting. The active state of the slave selection output signal is specified in SSACTPOL (SPI_SSCTL[2]).

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.

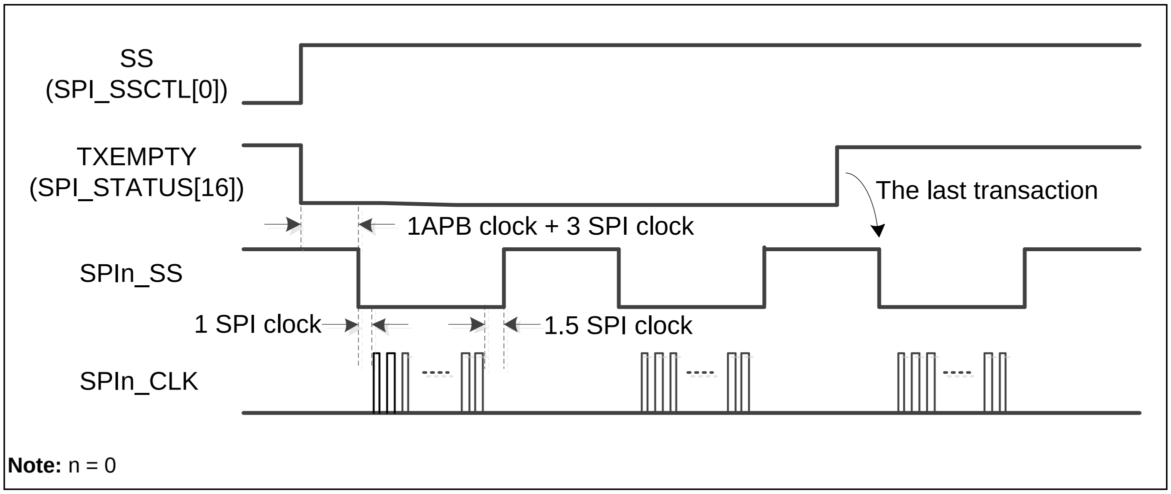


Figure 6.18-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2)

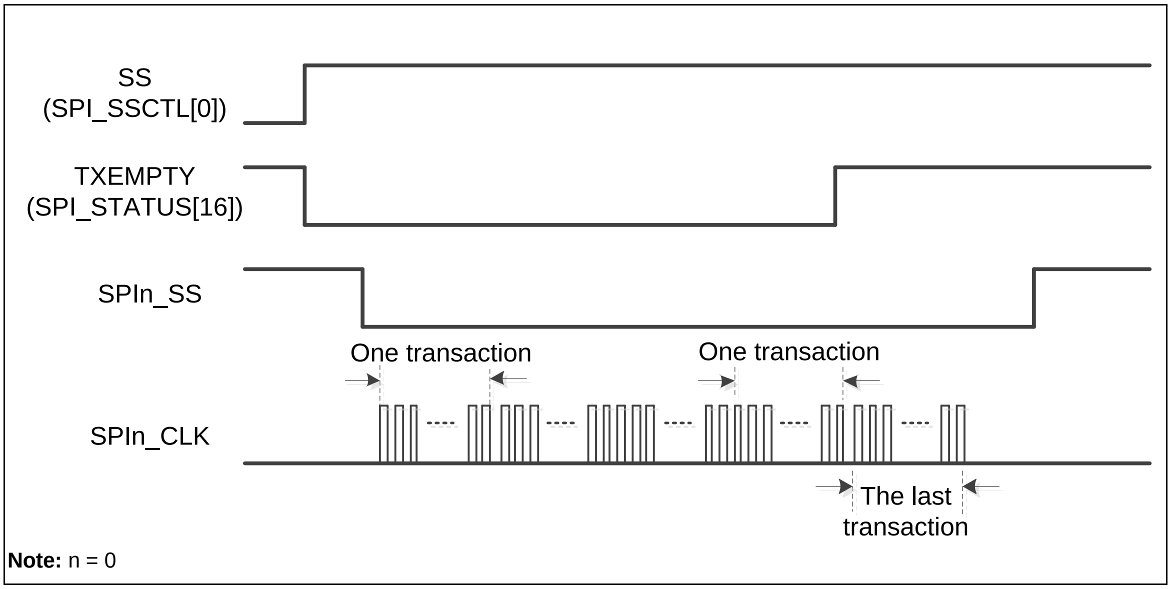


Figure 6.18-7 Automatic Selection (SSACTPOL = 0, SUSPITV < 0x3)

6.18.5.3 *Byte Reorder and Suspend Function*

When the transfer is set as MSB first (LSB = 0) and the REORDER (SPI_CTL[19]) 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 (DWIDTH = 0). The sequence of transmitted/received data will be Byte0, Byte1, Byte2, and then Byte3. If the DWIDTH 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 DWIDTH is configured as 16, 24, and 32 bits.

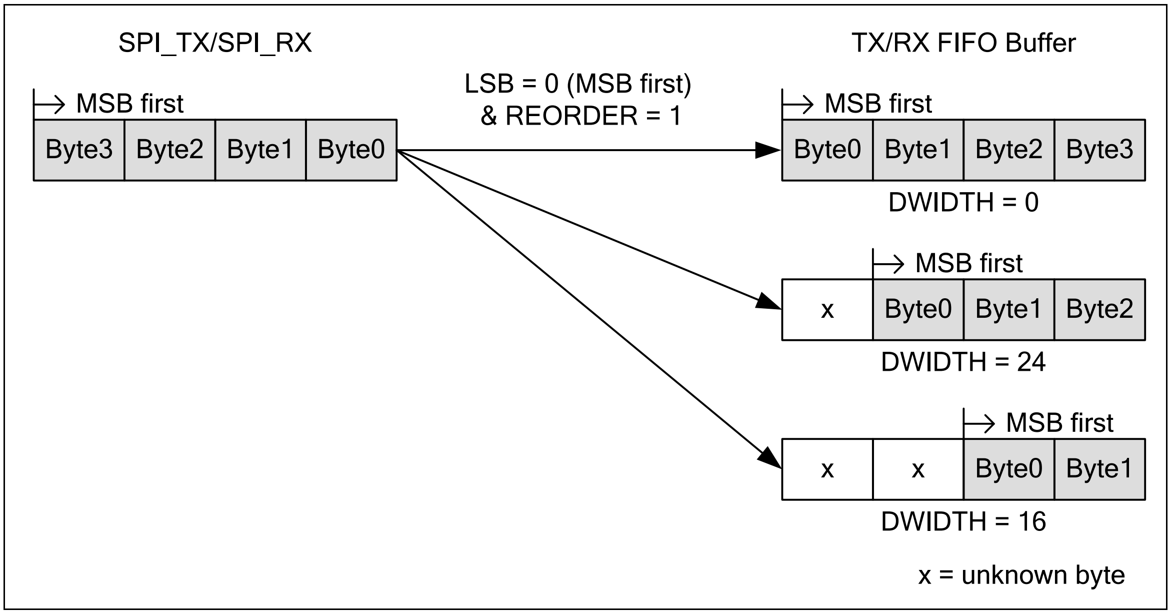
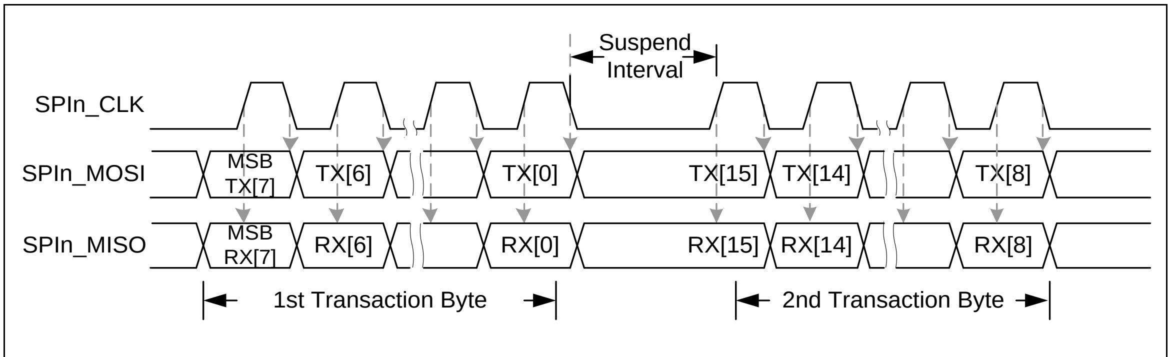


Figure 6.18-8 Byte Reorder Function

In Master mode, if REORDER (SPI_CTL[19]) is set to 1, a suspend interval of 0.5 ~ 15.5 SPI clock periods will be inserted by hardware between two successive bytes in a transaction word. The suspend interval is configured in SUSPITV (SPI_CTL[7:4]).



Note 1: n = 0

Note 2: Timing Condition is CLKPOL = 0, LSB = 0, TXNEG = 1, DWIDTH = 0, REORDER = 1 and SLAVE = 0

Figure 6.18-9 Timing Waveform for Byte Suspend

6.18.5.4 *Half-Duplex Communication*

The SPI controller can communicate in half-duplex mode by setting HALFDPX (SPI_CTL[14]) bit. In half-duplex mode, there is only one data line for receiving or transmitting data direction which is defined by DATDIR (SPI_CTL[20]). In half-duplex configuration, the SPI_MISO pin is free for other applications and it can be configured as GPIO. Enabling or disabling the control bit HALFDPX (SPI_CTL[14]) will produce TXRST (SPI_FIFOCTL[1]) and RXRST (SPI_FIFOCTL[0]) at the same time automatically.

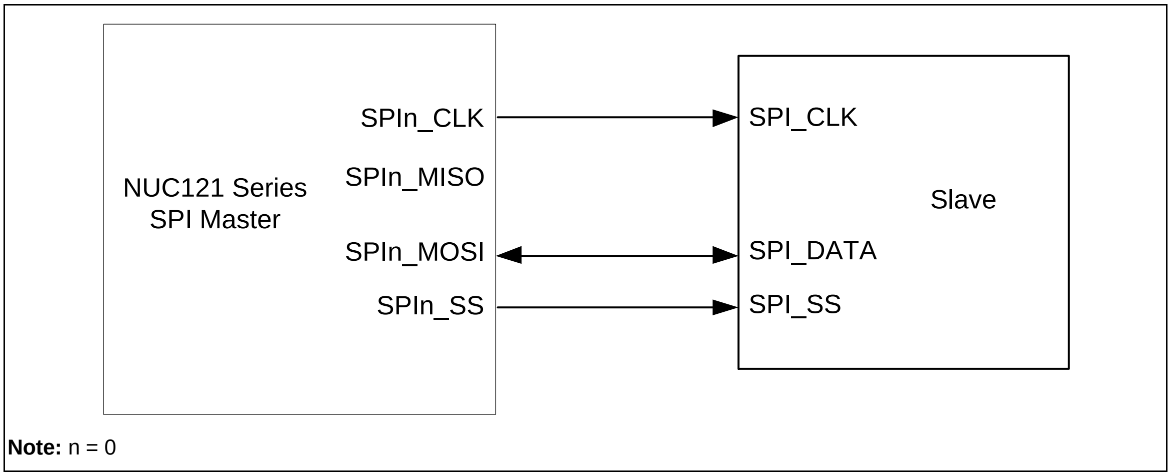


Figure 6.18-10 SPI Half-Duplex Master Mode Application Block Diagram

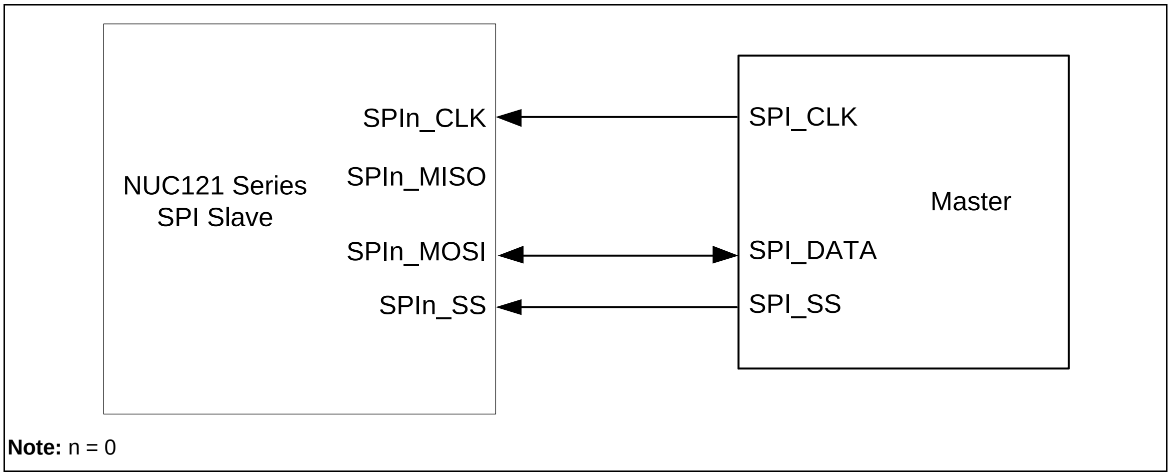


Figure 6.18-11 SPI Half-Duplex Slave Mode Application Block Diagram

6.18.5.5 Receive-Only Mode

In SPI Master device, it can communicate in receive-only mode by setting RXONLY (SPI_CTL[15]). In this configuration, the SPI Master device will generate SPI bus clock continuously as long as the receive-only mode is enabled for receiving data bit from SPI slave device. If AUTOSS (SPI_SSCTL[3]) is enabled in receive-only mode, SPI Master will keep activating the slave select signal.

The remaining SPI_MOSI pin of SPI Master device is not used for communication and can be configured as GPIO. The status BUSY (SPI_STATUS[0]) will be asserted in receive-only mode due to the generation of SPI bus clock. Entering this mode will produce the TXRST (SPI_FIFOCTL[1]) and RXRST (SPI_FIFOCTL[0]) at the same time automatically. After enabling this mode, the output SPI bus clock will be sent out in 6 peripheral clock cycles. In this mode, the data which has been written into transmit FIFO will be loaded into transmit shift register and sent out.

6.18.5.6 PDMA Transfer Function

SPI controller supports PDMA transfer function.

When TXPDMAEN (SPI_PDMACTL[0]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (SPI_PDMACTL[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.

Note: SPI supports single request PDMA (Read/Write) only, burst request PDMA is not supported.

6.18.5.7 FIFO Buffer Operation

The SPI controllers equip with four 32-bit wide transmit and receive FIFO buffers. The data stored in the transmit FIFO buffer will be read and sent out by the transmission control logic. If the transmit FIFO buffer is full, the TXFULL (SPI_STATUS[17]) will be set to 1. When the SPI transmission logic unit draws out the last datum of the transmit FIFO buffer, so that the transmit FIFO buffer is empty, the TXEMPTY (SPI_STATUS[16]) will be set to 1. Note that the TXEMPTY (SPI_STATUS[16]) flag is set to 1 while the last transaction is still in progress. In Master mode, the BUSY (SPI_STATUS[0]) is set to 1 when the FIFO buffer is written any data or there is any transaction on the SPI bus. (e.g. the slave selection signal is active and the SPI controller is receiving data in Slave mode). It will set to 0 when the transmit FIFO is empty and the current transaction has done. Thus, the status of BUSY (SPI_STATUS[0]) should be checked by software to make sure whether the SPI is in idle or not.

The receive control logic will store the SPI input data into the receive FIFO buffer. There are FIFO related status bits, like RXEMPTY (SPI_STATUS[8]) and RXFULL (SPI_STATUS[9]), to indicate the current status of RX FIFO buffer.

The transmitting and receiving threshold can be configured by setting TXTH (SPI_FIFOCTL[29:28]) and RXTH (SPI_FIFOCTL[25:24]). When the count of valid data stored in transmit FIFO buffer is less than or equal to TXTH (SPI_FIFOCTL[29:28]) setting, TXTHIF (SPI_STATUS[18]) will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RXTH (SPI_FIFOCTL[25:24]) setting, RXTHIF (SPI_STATUS[10]) will be set to 1.

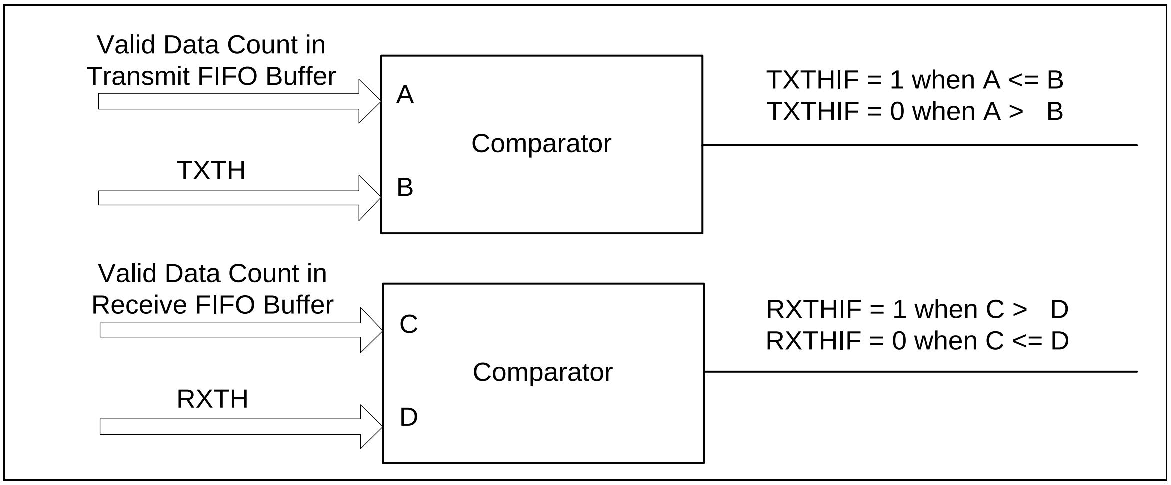


Figure 6.18-12 FIFO Threshold Comparator

In Master mode, when the first datum is written to the SPI_TX register, the flag TXEMPTY (SPI_STATUS[16]) will be cleared to 0. The transmission will start after 1 APB clock cycles 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 SUSPITV (SPI_CTL[7:4]). If the SUSPITV (SPI_CTL[7:4]) equals 0, SPI controller can perform continuous transfer. User can write data into SPI_TX register as long as the TXFULL (SPI_STATUS[17]) is 0.

The Example 1 of Figure 6.18-13 indicates the updated condition of TXEMPTY (SPI_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer. The TXEMPTY (SPI_STATUS[16]) is set to 0 when the Data0 is written into the FIFO buffer. The Data0 will be loaded into the shift register by core logical and the TXEMPTY (SPI_STATUS[16]) will be 1. The Data0 in shift register will be shift into skew buffer by bit for transmission until the transfer is done.

The Example 2 of Figure 6.18-13 indicates the updated condition of TXFULL (SPI_STATUS[17]) when there are 4 data in the FIFO buffer and the next data of Data5 does not be written into the FIFO buffer when the TXFULL = 1.

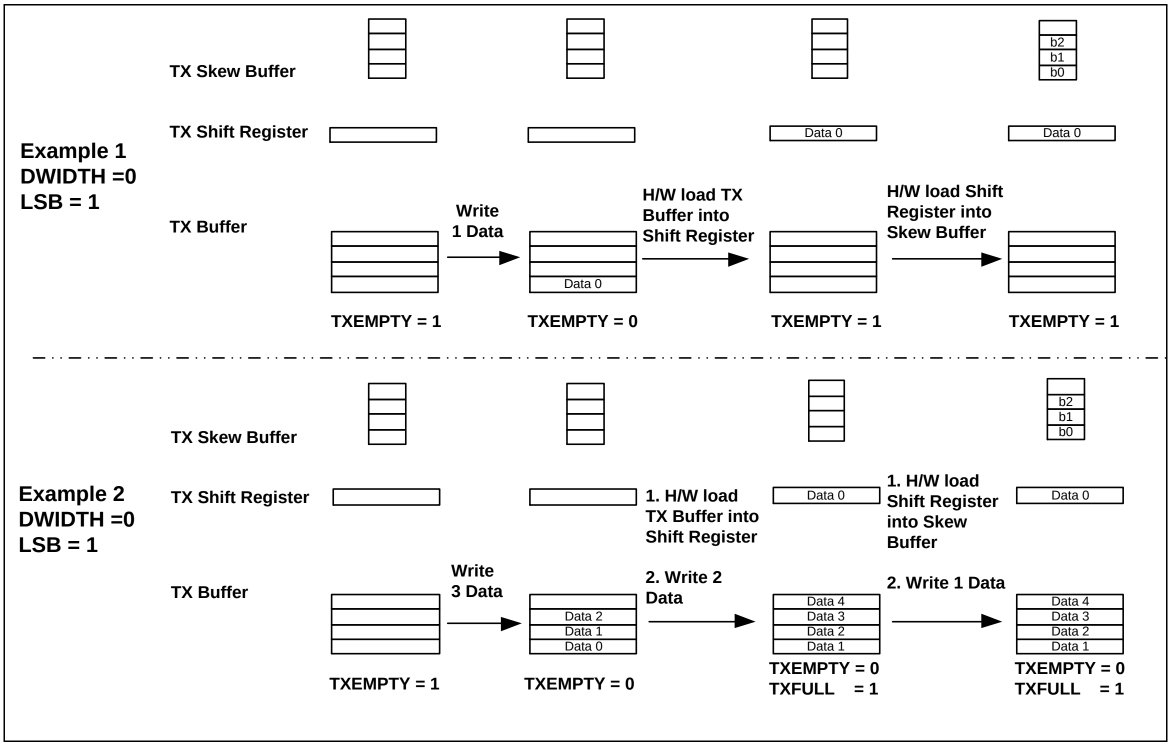


Figure 6.18-13 Transmit FIFO Buffer Example

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the SPI_TX register does not be updated after all data transfer are done, the transfer will stop.

In Master mode, during receiving operation, the serial data are received from SPI_MISO pin and stored to receive FIFO buffer.

In the Example 1 of Figure 6.18-14, the receive data (Data0's b0, b1, …b31) is stored into skew buffer first according the serial clock (SPI_CLK) and then it is shift 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 count reach the value of DWIDTH (SPI_CTL[12:8]). The RXEMPTY (SPI_STATUS[8]) will be cleared to 0 while the receive FIFO buffer contains unread data. The received data can be read by software from SPI_RX register as long as the RXEMPTY (SPI_STATUS[8]) is 0.

In the Example 2 of Figure 6.18-14, if the receive FIFO buffer contains 4 unread data, the RXFULL (SPI_STATUS[9]) will be set to 1.

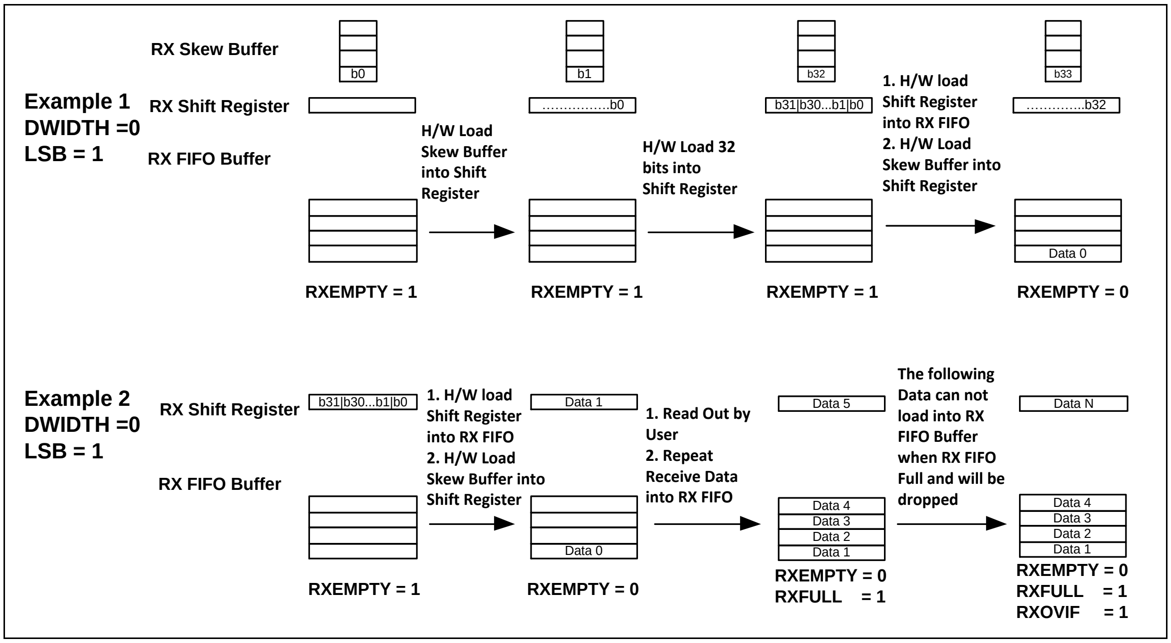


Figure 6.18-14 Receive FIFO Buffer Example

In Slave mode, during transmission operation, when data is written to the SPI_TX register by software, the data will be loaded into transmit FIFO buffer and the TXEMPTY (SPI_STATUS[16]) will be set to 0. The transmission will start when the slave device receives clock signal from master. Data can be written to SPI_TX register as long as the TXFULL (SPI_STATUS[17]) is 0. After all data have been drawn out by the SPI transmission logic unit and the SPI_TX register is not updated by software, the TXEMPTY (SPI_STATUS[16]) will be set to 1.

If there is no any data written to the SPI_TX register, the transmit underflow interrupt flag, TXUFIF (SPI_STATUS[19]) will be set to 1 when the slave selection signal is active. The output data will be held by TXUFPOL (SPI_FIFOCTL[6]) setting during this transfer until the slave selection signal goes to inactive state. When the transmit underflow event occurs, the slave under run flag, SLVURIF (SPI_STATUS[7]), will be set to 1 as SPI_SS goes to inactive state.

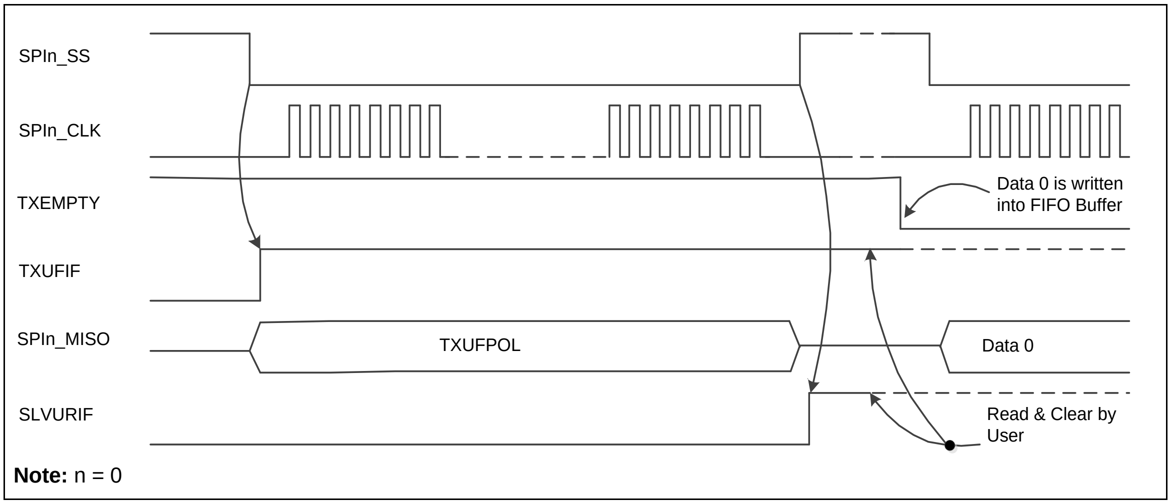


Figure 6.18-15 TX Underflow Event and Slave Under Run Event

In Slave mode, during receiving operation, the serial data is received from SPI_MOSI pin and stored to SPI_RX register. The reception mechanism is similar to Master mode reception operation. If the receive FIFO buffer contains 4 unread data, the RXFULL (SPI_STATUS[9]) will be set to 1 and the RXOVIF (SPI_STATUS[11]) will be set 1 if there is more serial data received from SPI_MOSI and follow-up data will be dropped (refer to the Example 2 of Figure 6.18-14). If the receive bit count mismatch with the DWIDTH (SPI_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (SPI_STATUS[6]) will be set to 1.

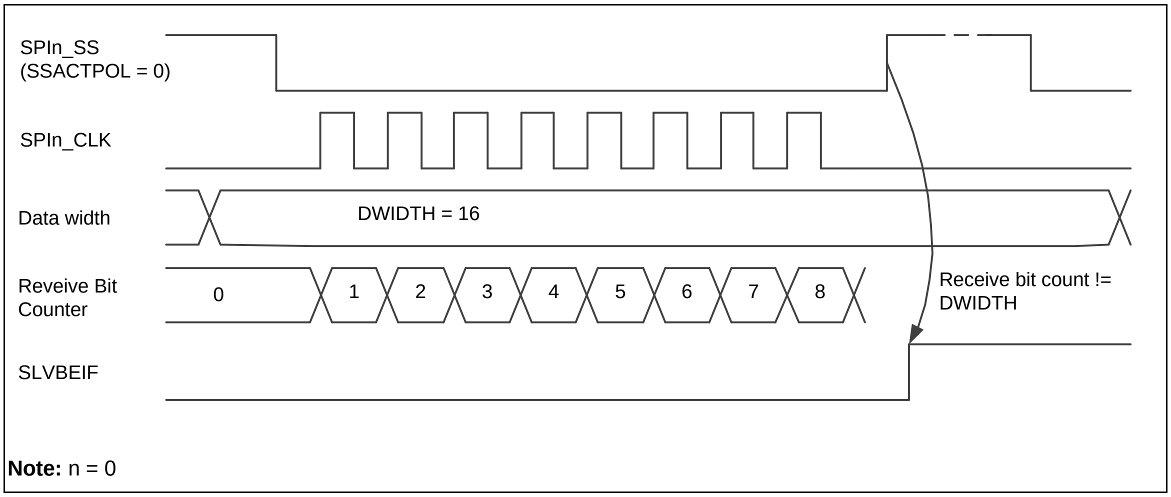


Figure 6.18-16 Slave Mode Bit Count Error

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 periods in Master mode or over 576 SPI peripheral clock periods in Slave mode, the receive time-out occurs and the RXTOIF (SPI_STATUS[12]) be set to 1. When the receive FIFO is read by user, the time-out status will be cleared automatically.

6.18.5.8 Interrupt

- SPI unit transfer interrupt

As the SPI controller finishes a unit transfer, the unit transfer interrupt flag UNITIF (SPI_STATUS[1]) will be set to 1. The unit transfer interrupt event will generate an interrupt to CPU if the unit transfer interrupt enable bit UNITIEN (SPI_CTL[17]) is set. The unit transfer interrupt flag can be cleared only by writing 1 to it.

- SPI slave selection active/inactive interrupt

In Slave mode, the slave selection active/inactive interrupt flag, SSACTIF (SPI_STATUS[2]) and SSINAIF (SPI_STATUS[3]), will be set to 1 when the SPIEN (SPI_CTL[0]) and SLAVE (SPI_CTL[18]) are set to 1 and the slave selection signal goes to active/inactive state. The SPI controller will issue an interrupt if the SSINAIEN (SPI_SSCTL[13]) or SSACTIEN (SPI_SSCTL[12]), are set to 1.

- Slave bit count error interrupt

In Slave mode, if the transmit receive bit count mismatch with the DWIDTH (SPI_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (SPI_STATUS[6]) will be set to 1. The uncompleted transaction will be dropped from TX and RX shift registers. The SPI controller will issue an interrupt if the SLVBEIEN (SPI_SSCTL[8]) is set to 1.

Note: If the slave selection signal is active but there is no any serial clock input, the SLVBEIF (SPI_STATUS[6]) will be set to 1 when the slave selection signal goes to inactive state.

- TX underflow interrupt

In SPI Slave mode, if there is no any data written to the SPI_TX register, the TXUFIF (SPI_STATUS[19]) will be set to 1 when the slave selection signal is active. The SPI controller will issue a TX underflow interrupt if the TXUFIEN (SPI_FIFOCTL[7]) is set to 1.

Note: If underflow event occurs in SPI Slave mode, there are two conditions which make SPI Slave mode return to idle state and then goes for next transfer: (1) set TXRST to 1 (2) slave select signal is changed to inactive state.

- Slave TX under run interrupt

If the TX underflow event occurs, the SLVURIF (SPI_STATUS[7]) will be set to 1 when SPI_SS goes to inactive state. The SPI controller will issue a TX under run interrupt if the SLVURIEN (SPI_SSCTL[9]) is set to 1.

- Receive Overrun interrupt

In Slave mode, if the receive FIFO buffer contains 4 unread data, the RXFULL (SPI_STATUS[9]) flag will be set to 1 and the RXOVIF (SPI_STATUS[11]) will be set 1 if there is more serial data received from SPI bus and follow-up data will be dropped. The SPI controller will issue an interrupt if the RXOVIEN (SPI_FIFOCTL[5]) is set to 1.

- Receive FIFO time-out interrupt

If there is a received data in the FIFO buffer and it is not read by software over 64 SPI peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode, it will send a RX time-out interrupt to the system if the RX time-out interrupt enable bit, RXTOIEN (SPI_FIFOCTL[4]), 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 TXTH (SPI_FIFOCTL[29:28]), the transmit FIFO interrupt flag TXTHIF (SPI_STATUS[18]) will be set to 1. The SPI controller will generate a transmit FIFO interrupt to the system if the transmit FIFO interrupt enable bit, TXTHIEN (SPI_FIFOCTL[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 RXTH (SPI_FIFOCTL[25:24]), the receive FIFO interrupt flag RXTHIF (SPI_STATUS[10]) will be set to 1. The SPI controller will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, RXTHIEN (SPI_FIFOCTL[2]), is set to 1.

6.18.5.9 I^{2}S Mode

The SPI0 controllers support I^{2}S mode with PCM mode A, PCM mode B, MSB justified and I^{2}S data format. The bit count of an audio channel is determined by WDWIDTH (SPI_I2SCTL[5:4]). The transfer sequence is always first from the most significance bit, MSB. Data are read on rising clock edge and are driven on falling clock edge.

In I^{2}S data format, the MSB is sent and latched on the second clock of an audio channel. The I2S_LRCLK signal indicates which audio channel is in transferring.

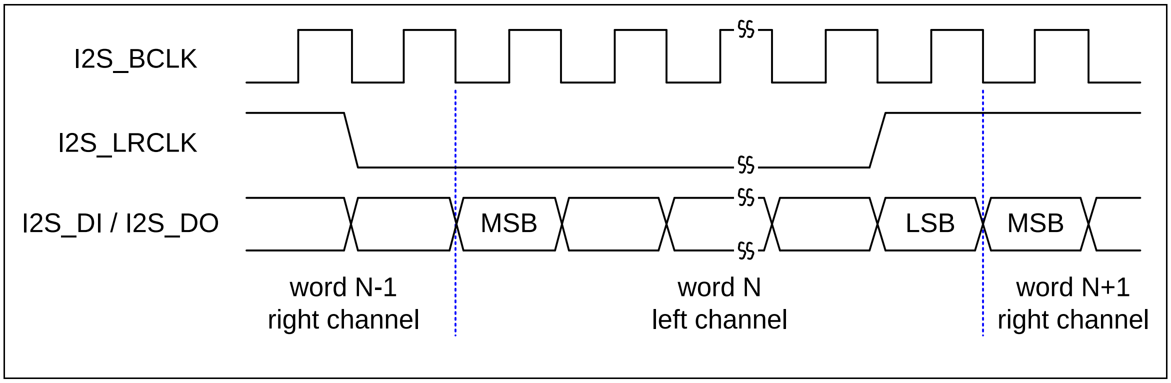


Figure 6.18-17 I^{2}S Data Format Timing Diagram

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

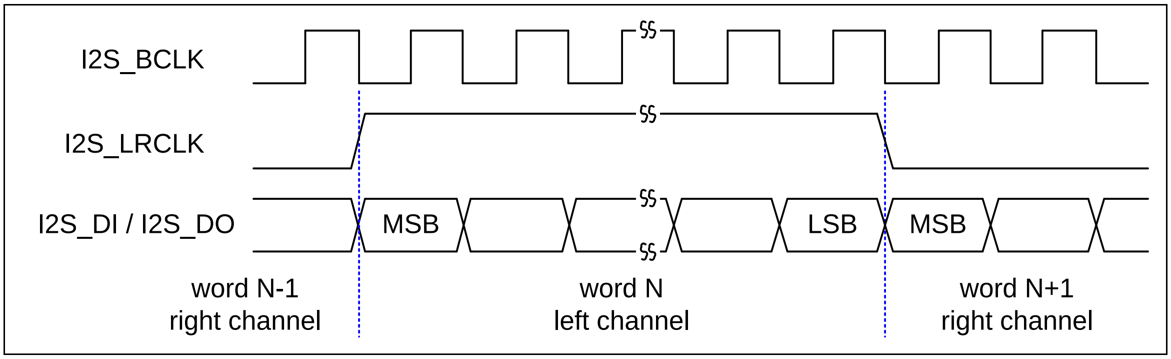


Figure 6.18-18 MSB Justified Data Format Timing Diagram

The I2S_LRCLK signal also supports PCM mode A and PCM mode B. The I2S_LRCLK signal in PCM mode indicates the beginning of an audio frame.

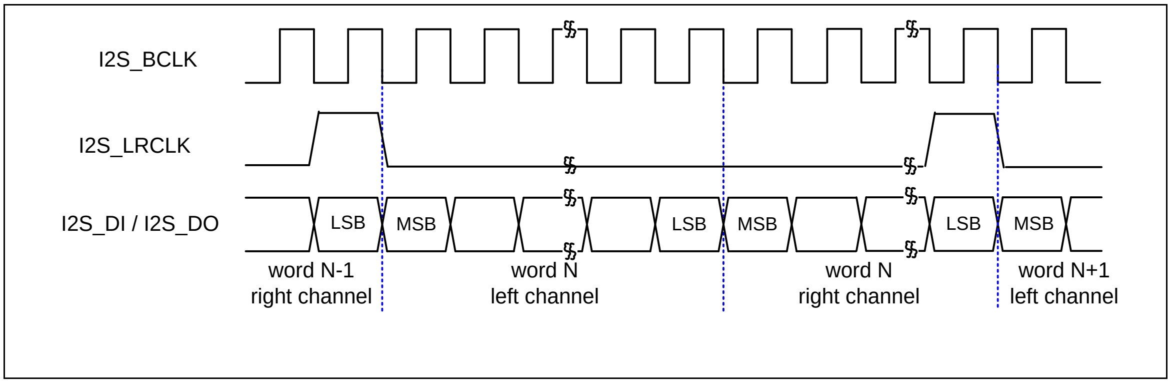


Figure 6.18-19 PCM Mode A Timing Diagram

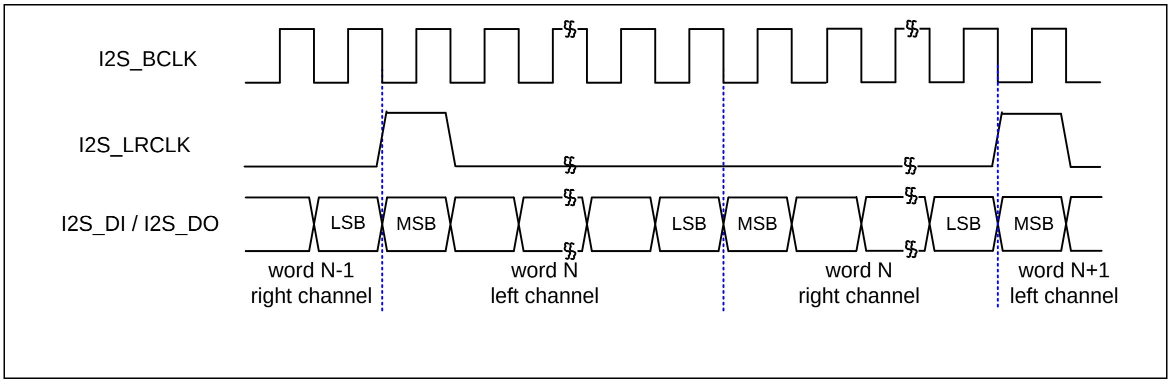


Figure 6.18-20 PCM Mode B Timing Diagram

6.18.5.10 I^{2}S Mode FIFO operation

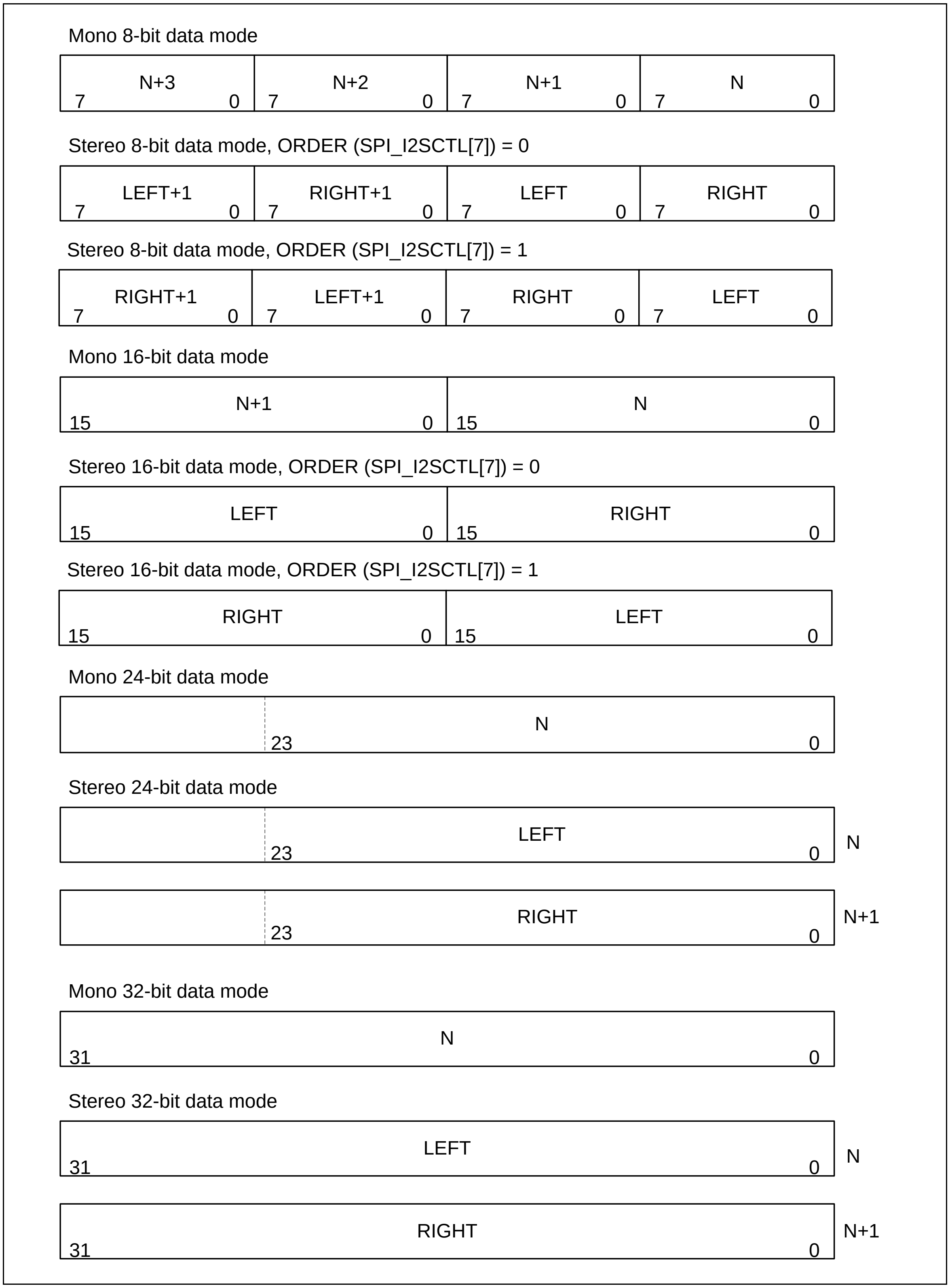
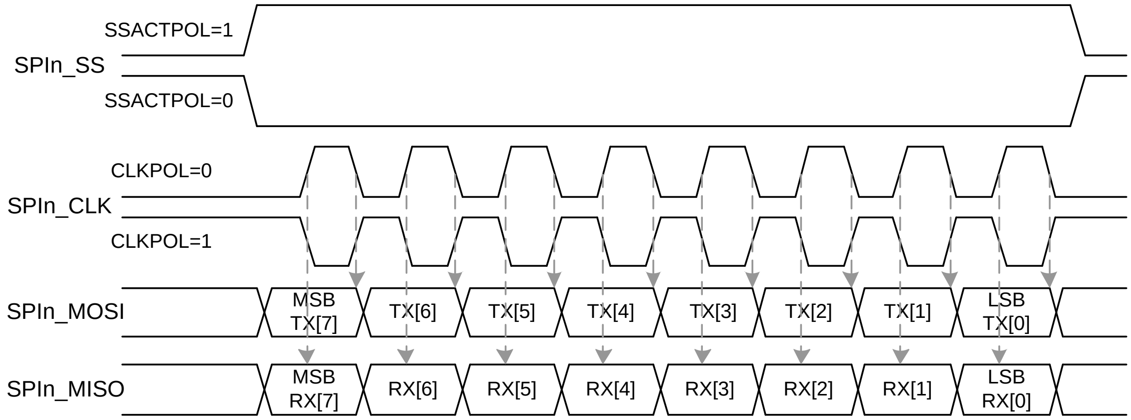


Figure 6.18-21 FIFO Contents for Various I^{2}S Modes

6.18.6 Timing Diagram

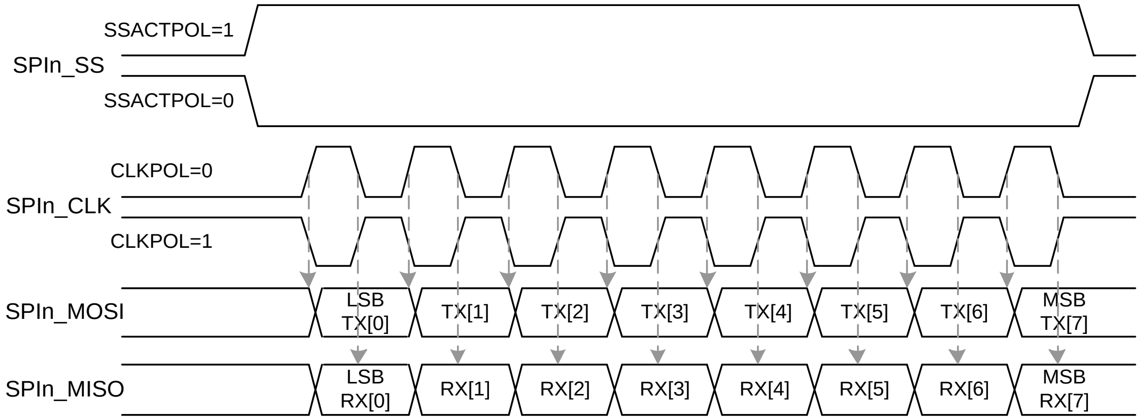
The active state of slave selection signal can be defined by setting the SSACTPOL (SPI_SSCTL[2]). The SPI clock which is in idle state can be configured as high or low state by setting the CLKPOL (SPI_CTL[3]). It also provides the bit length of a transaction word in DWIDTH (SPI_CTL[12:8]), and transmitting/receiving data from MSB or LSB first in LSB bit (SPI_CTL[13]). User can also select which edge of SPI clock to transmit/receive data in TXNEG/RXNEG (SPI_CTL[2:1]). Four SPI timing diagrams for master/slave operations and the related settings are shown below.



Master Mode: SLVAE=0, LSB=0, DWIDTH=0x08
1. CLKPOL=0, TXNEG=1, RXNEG=0 or
2. CLKPOL=1, TXNEG=0, RXNEG=1

Note: n = 0

Figure 6.18-22 SPI Timing in Master Mode



Master Mode: SLVAE=0, LSB=1, DWIDTH=0x08
1. CLKPOL=0, TXNEG=0, RXNEG=1 or
2. CLKPOL=1, TXNEG=1, RXNEG=0

Note: n = 0

Figure 6.18-23 SPI Timing in Master Mode (Alternate Phase of SPI_CLK)

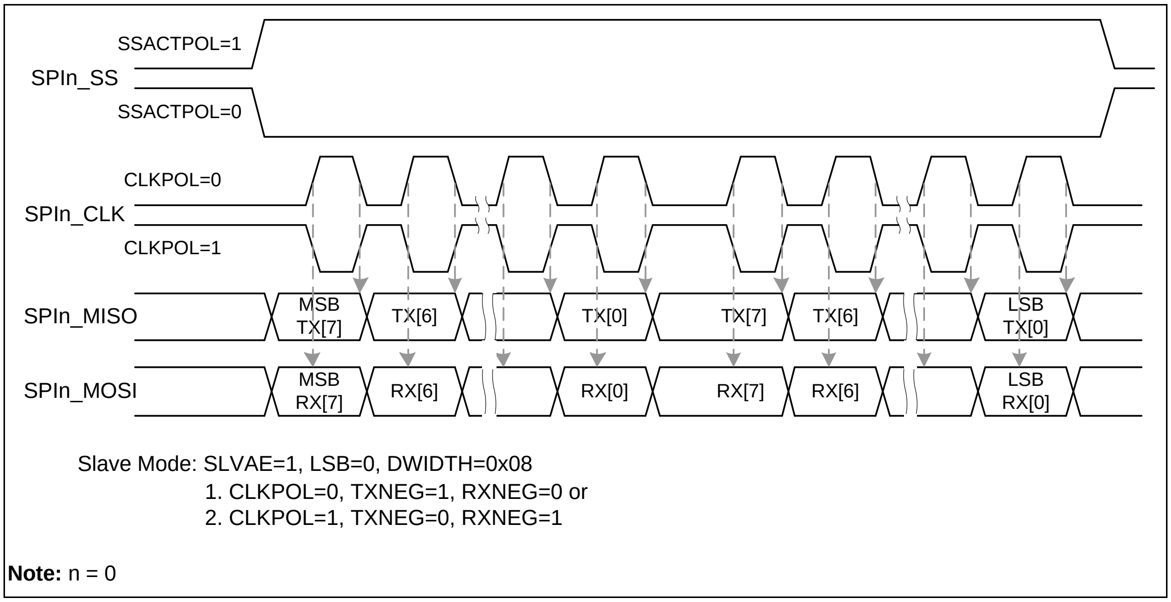


Figure 6.18-24 SPI Timing in Slave Mode

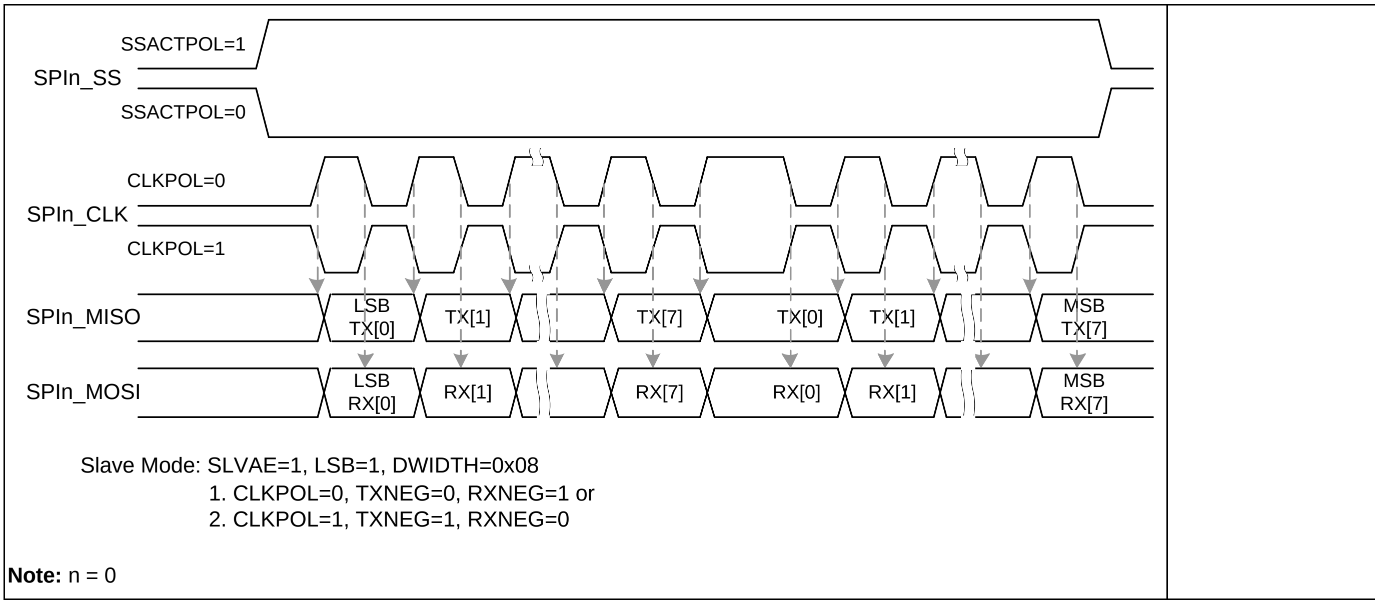


Figure 6.18-25 SPI Timing in Slave Mode (Alternate Phase of SPI_CLK)

6.18.7 Programming Examples

Example 1: The SPI controller is set as a full-duplex master to access an off-chip slave device 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 MSB first.
- SPI bus clock is idle at low state.
- Only one byte of data to be transmitted/received in a transaction.
- Uses the first SPI slave select pin to connect with an off-chip slave device. The slave selection signal is active low.

The operation flow is as follows:

1) Set DIVIDER (SPI_CLKDIV [7:0]) to determine the output frequency of SPI clock.
2) Write the SPI_SSCTL register a proper value for the related settings of Master mode:
   1. Clear AUTOSS (SPI_SSCTL[3]) to 0 to disable the Automatic Slave Selection function.
   2. Configure slave selection signal as active low by clearing SSACTPOL (SPI_SSCTL[2]) to 0.
   3. Enable slave selection signal by setting SS (SPI_SSCTL[0]) to 1 to active the off-chip slave device.
3) Write the related settings into the SPI_CTL register to control the SPI master actions.
   1. Configure this SPI controller as master device by setting SLAVE (SPI_CTL[18]) to 0.
   2. Force the SPI clock idle state at low by clearing CLKPOL (SPI_CTL[3]) to 0.
   3. Select data transmitted on negative edge of SPI bus clock by setting TXNEG (SPI_CTL[2]) to 1.
   4. Select data latched on positive edge of SPI bus clock by clearing RXNEG (SPI_CTL[1]) to 0.
   5. Set the bit length of a transaction as 8-bit in DWIDTH bit field (SPI_CTL[12:8] = 0x08).
   6. Set MSB transfer first by clearing LSB (SPI_CTL[13]) to 0.
4) Set SPIEN (SPI_CTL[0]) to 1 to enable the data transfer with the SPI interface.
5) If this SPI master attempts to transmit (write) one byte data to the off-chip slave device, write the byte data that will be transmitted into the SPI_TX register.
6) Waiting for SPI interrupt if the UNITIEN (SPI_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (SPI_STATUS[1]).
7) Read out the received one byte data from SPI_RX register.
8) Go to 5) to continue another data transfer or set SS (SPI_SSCTL[0]) to 0 to inactivate the off-chip slave device.

Example 2: The SPI controller is set as a full-duplex 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 is idle at high state.
- Only one byte of data to be transmitted/received in a transaction.
- Slave selection signal is active high.

The operation flow is as follows:

1) Write the SPI_SSCTL register a proper value for the related settings of Slave mode.

   Select high level for the input of slave selection signal by setting SSACTPOL (SPI_SSCTL[2]) to 1.

2) Write the related settings into the SPI_CTL register to control this SPI slave actions

   1. Set the SPI controller as slave device by setting SLAVE (SPI_CTL[18]) to 1.
   2. Select the SPI clock idle state at high by setting CLKPOL (SPI_CTL[3]) to 1.
   3. Select data transmitted on negative edge of SPI bus clock by setting TXNEG (SPI_CTL[2]) to 1.
   4. Select data latched on positive edge of SPI bus clock by clearing RXNEG (SPI_CTL[1]) to 0.
   5. Set the bit length of a transaction as 8-bit in DWIDTH bit field (SPI_CTL[12:8] = 0x08).
   6. Set LSB transfer first by setting LSB (SPI_CTL[13]) to 1.

3) Set the SPIEN (SPI_CTL[0]) to 1. Wait for the slave select trigger input and SPI clock input from the off-chip master device to start the data transfer.

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

5) If this SPI slave just only attempts to receive (be written) one byte data from the off-chip master device and does not care what data will be transmitted, the SPI_TX register does not need to be updated by software.

6) Waiting for SPI interrupt if the UNITIEN (SPI_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (SPI_STATUS[1]).

7) Read out the received one byte data from SPI_RX register.

8) Go to 4) to continue another data transfer or stop data transfer.

6.18.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI Base Address: **SPI_BA = 0x4003_0000**				
SPI_CTL	SPI_BA+0x00	R/W	SPI Control Register	0x0000_0034
SPI_CLKDIV	SPI_BA+0x04	R/W	SPI Clock Divider Register	0x0000_0000
SPI_SSCTL	SPI_BA+0x08	R/W	SPI Slave Select Control Register	0x0000_0000
SPI_PDMACTL	SPI_BA+0x0C	R/W	SPI PDMA Control Register	0x0000_0000
SPI_FIFOCTL	SPI_BA+0x10	R/W	SPI FIFO Control Register	0x2200_0000
SPI_STATUS	SPI_BA+0x14	R/W	SPI Status Register	0x0005_0110
SPI_TX	SPI_BA+0x20	W	SPI Data Transmit Register	0x0000_0000
SPI_RX	SPI_BA+0x30	R	SPI Data Receive Register	0x0000_0000
SPI_I2SCTL	SPI_BA+0x60	R/W	I^{2}S Control Register	0x0000_0000
SPI_I2SCLK	SPI_BA+0x64	R/W	I^{2}S Clock Divider Control Register	0x0000_0000
SPI_I2SSTS	SPI_BA+0x68	R/W	I^{2}S Status Register	0x0005_0100

6.18.9 Register Description

SPI Control Register (SPI_CTL)

Register	Offset	R/W	Description	Reset Value
SPI_CTL	SPI_BA+0x00	R/W	SPI Control Register	0x0000_0034

Note: Not supported in I^{2}S mode.

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**DATDIR**	**REORDER**	**SLAVE**	**UNITIEN**	**Reserved**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXONLY	**HALFDPX**	**LSB**	**DWIDTH**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SUSPITV				**CLKPOL**	**TXNEG**	**RXNEG**	**SPIEN**

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**DATDIR**	**Data Port Direction Control** This bit is used to select the data input/output direction in half-duplex transfer. 0 = SPI data is input direction. 1 = SPI data is output direction.
[19]	**REORDER**	**Byte Reorder Function Enable Bit** 0 = Byte Reorder function Disabled. 1 = Byte Reorder function Enabled. A byte suspend interval will be inserted among each byte. The period of the byte suspend interval depends on the setting of SUSPITV. **Note:** Byte Reorder function is only available if DWIDTH is defined as 16, 24, and 32 bits.
[18]	**SLAVE**	**Slave Mode Control** 0 = Master mode. 1 = Slave mode.
[17]	**UNITIEN**	**Unit Transfer Interrupt Enable Bit** 0 = SPI unit transfer interrupt Disabled. 1 = SPI unit transfer interrupt Enabled.
[16]	**Reserved**	Reserved.
[15]	**RXONLY**	**Receive-only Mode Enable Bit (Master Only)** This bit field is only available in Master mode. In receive-only mode, SPI Master will generate SPI bus clock continuously for receiving data bit from SPI slave device and assert the BUSY status. 0 = Receive-only mode Disabled. 1 = Receive-only mode Enabled.
[14]	**HALFDPX**	**SPI Half-duplex Transfer Enable Bit** This bit is used to select full-duplex or half-duplex for SPI transfer. The bit field DATDIR

		(SPI_CTL[20]) can be used to set the data direction in half-duplex transfer. 0 = SPI operates in full-duplex transfer. 1 = SPI operates in half-duplex transfer.
[13]	**LSB**	**Send LSB First** 0 = The MSB, which bit of transmit/receive register depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, bit 0 of the SPI TX 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_RX).

[12:8]	**DWIDTH**	**Data Width**
		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.
		DWIDTH = 0x08 … 8 bits.
		DWIDTH = 0x09 … 9 bits.
		……
		DWIDTH = 0x1F … 31 bits.
		DWIDTH = 0x00 … 32 bits.

[7:4]	**SUSPITV**	**Suspend Interval (Master Only)** The four bits provide configurable suspend interval between two successive transmit/receive transaction in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. (SUSPITV[3:0] + 0.5) * period of SPICLK clock cycle Example: SUSPITV = 0x0 … 0.5 SPICLK clock cycle. SUSPITV = 0x1 … 1.5 SPICLK clock cycle. …… SUSPITV = 0xE … 14.5 SPICLK clock cycle. SUSPITV = 0xF … 15.5 SPICLK clock cycle.
[3]	**CLKPOL**	**Clock Polarity** 0 = SPI bus clock is idle low. 1 = SPI bus clock is idle high.
[2]	**TXNEG**	**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. **Note:** The setting of TXNEG and RXNEG are mutual exclusive
[1]	**RXNEG**	**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. **Note:** The setting of TXNEG and RXNEG are mutual exclusive
[0]	**SPIEN**	**SPI Transfer Control Enable Bit** In Master mode, the transfer will start when there is data in the FIFO buffer after this is set to 1. In Slave mode, this device is ready to receive data when this bit is set to 1. 0 = Transfer control Disabled. 1 = Transfer control Enabled. **Note:** Before changing the configurations of SPI_CTL, SPI_CLKDIV, SPI_SSCTL and SPI_FIFOCTL registers, user shall clear the SPIEN (SPI_CTL[0]) and confirm the SPIENSTS (SPI_STATUS[15]) is 0.

SPI Divider Register (SPI_CLKDIV)

Register	Offset	R/W	Description	Reset Value
SPI_CLKDIV	SPI_BA+0x04	R/W	SPI Clock Divider Register	0x0000_0000

Note: Not supported in I^{2}S mode

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DIVIDER**	**Clock Divider** The value in this field is the frequency divider for generating the peripheral clock, f_{spi_eclk}, and the SPI bus clock of SPI Master. The frequency is obtained according to the following equation. $f_{spi\_eclk} = \frac{f_{spi\_clock\_src}}{(DIVIDER + 1)}$ where $f_{spi\_clock\_src}$ is the peripheral clock source, which is defined in the clock control register, CLK_CLKSEL2. **Note:** Not supported in I^{2}S mode. **Note:** User should set DIVIDER carefully because the peripheral clock frequency must be slower than or equal to system frequency

SPI Slave Select Register (SPI_SSCTL)

Register	Offset	R/W	Description	Reset Value
SPI_SSCTL	SPI_BA+0x08	R/W	SPI Slave Select Control Register	0x0000_0000

Note: Not supported in I^{2}S mode.

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable Bit** 0 = Slave select inactive interrupt Disabled. 1 = Slave select inactive interrupt Enabled.
[12]	**SSACTIEN**	**Slave Select Active Interrupt Enable Bit** 0 = Slave select active interrupt Disabled. 1 = Slave select active interrupt Enabled.
[11:10]	**Reserved**	Reserved.
[9]	**SLVURIEN**	**Slave Mode TX Under Run Interrupt Enable Bit** 0 = Slave mode TX under run interrupt Disabled. 1 = Slave mode TX under run interrupt Enabled.
[8]	**SLVBEIEN**	**Slave Mode Bit Count Error Interrupt Enable Bit** 0 = Slave mode bit count error interrupt Disabled. 1 = Slave mode bit count error interrupt Enabled.
[7:4]	**Reserved**	Reserved.
[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_SSCTL[0]). 1 = Automatic slave selection function Enabled.
[2]	**SSACTPOL**	**Slave Selection Active Polarity** This bit defines the active polarity of slave selection signal (SPI_SS). 0 = The slave selection signal SPI_SS is active low. 1 = The slave selection signal SPI_SS is active high.
[1]	**Reserved**	Reserved.

[0]	**SS**	**Slave Selection Control (Master Only)** If AUTOSS bit is cleared to 0, 0 = set the SPI_SS line to inactive state. 1 = set the SPI_SS line to active state. If the AUTOSS bit is set to 1, 0 = Keep the SPI_SS line at inactive state. 1 = SPI_SS 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 SPI_SS is specified in SSACTPOL (SPI_SSCTL[2]).

SPI PDMA Control Register (SPI_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SPI_PDMACTL	SPI_BA+0x0C	R/W	SPI 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					PDMARST	RXPDMAEN	TXPDMAEN

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the PDMA control logic of the SPI controller. This bit will be automatically cleared to 0.
[1]	**RXPDMAEN**	**Receive PDMA Enable Bit** 0 = Receiver PDMA function Disabled. 1 = Receiver PDMA function Enabled.
[0]	**TXPDMAEN**	**Transmit PDMA Enable Bit** 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 FIFO Control Register (SPI_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
SPI_FIFOCTL	SPI_BA+0x10	R/W	SPI FIFO Control Register	0x2200_0000

31	30	29	28	27	26	25	24
Reserved		TXTH		Reserved		RXTH	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						TXFBCLR	RXFBCLR
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXUFIEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:28]	**TXTH**	**Transmit FIFO Threshold** If the valid data count of the transmit FIFO buffer is less than or equal to the TXTH setting, the TXTHIF bit will be set to 1, else the TXTHIF bit will be cleared to 0.
[27:26]	**Reserved**	Reserved.
[25:24]	**RXTH**	**Receive FIFO Threshold** If the valid data count of the receive FIFO buffer is larger than the RXTH setting, the RXTHIF bit will be set to 1, else the RXTHIF bit will be cleared to 0.
[23:8]	**Reserved**	Reserved.
[9]	**TXFBCLR**	**Transmit FIFO Buffer Clear** 0 = No effect. 1 = Clear transmit FIFO pointer. The TXFULL bit will be cleared to 0 and the TXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 1 system clock after it is set to 1. **Note:** The TX shift register will not be cleared.
[8]	**RXFBCLR**	**Receive FIFO Buffer Clear** 0 = No effect. 1 = Clear receive FIFO pointer. The RXFULL bit will be cleared to 0 and the RXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 1 system clock after it is set to 1. **Note:** The RX shift register will not be cleared.
[7]	**TXUFIEN**	**TX Underflow Interrupt Enable Bit** When TX underflow event occurs in Slave mode, TXUFIF (SPI_STATUS[19]) will be set to 1. This bit is used to enable the TX underflow interrupt. 0 = Slave TX underflow interrupt Disabled. 1 = Slave TX underflow interrupt Enabled.
[6]	**TXUFPOL**	**TX Underflow Data Polarity** 0 = The SPI data out is keep 0 if there is TX underflow event in Slave mode.

		1 = The SPI data out is keep 1 if there is TX underflow event in Slave mode. **Note:** 1. The TX underflow event occurs if there is not any data in TX FIFO when the slave selection signal is active. 2. This bit should be set as 0 in I^{2}S mode. 3. When TX underflow event occurs, SPI_MISO pin state will be determined by this setting even though TX FIFO is not empty afterward. Data stored in TX FIFO will be sent through SPI_MISO pin in the next transfer frame.
[5]	**RXOVIEN**	**Receive FIFO Overrun Interrupt Enable Bit** 0 = Receive FIFO overrun interrupt Disabled. 1 = Receive FIFO overrun interrupt Enabled.
[4]	**RXTOIEN**	**Slave Receive Time-out Interrupt Enable Bit** 0 = Receive time-out interrupt Disabled. 1 = Receive time-out interrupt Enabled.
[3]	**TXTHIEN**	**Transmit FIFO Threshold Interrupt Enable Bit** 0 = TX FIFO threshold interrupt Disabled. 1 = TX FIFO threshold interrupt Enabled.
[2]	**RXTHIEN**	**Receive FIFO Threshold Interrupt Enable Bit** 0 = RX FIFO threshold interrupt Disabled. 1 = RX FIFO threshold interrupt Enabled.
[1]	**TXRST**	**Transmit Reset (Only for SPI)** 0 = No effect. 1 = Reset transmit FIFO pointer and transmit circuit. The TXFULL (SPI_STATUS[17]) bit will be cleared to 0 and the TXEMPTY (SPI_STATUS[16]) bit will be set to 1. This bit will be cleared to 0 by hardware about 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (SPI_STATUS[23]) to check if reset is accomplished or not. **Note:** If TX under-run event occurs in SPI Slave mode, this bit can be used to make SPI return to idle state.
[0]	**RXRST**	**Receive Reset (Only for SPI)** 0 = No effect. 1 = Reset receive FIFO pointer and receive circuit. The RXFULL (SPI_STATUS[9]) bit will be cleared to 0 and the RXEMPTY (SPI_STATUS[8]) bit will be set to 1. This bit will be cleared to 0 by hardware about 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (SPI_STATUS[23]) to check if reset is accomplished or not.

SPI Status Register(SPI_STATUS)

Register	Offset	R/W	Description	Reset Value
SPI_STATUS	SPI_BA+0x14	R/W	SPI Status Register	0x0005_0110

Note: Not supported in I^{2}S mode.

31	30	29	28	27	26	25	24
TXCNT				**RXCNT**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TXRXRST	**Reserved**			**TXUFIF**	**TXTHIF**	**TXFULL**	**TXEMPTY**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SPIENSTS	**Reserved**		**RXTOIF**	**RXOVIF**	**RXTHIF**	**RXFULL**	**RXEMPTY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SLVURIF	**SLVBEIF**	**Reserved**	**SSLINE**	**SSINAIF**	**SSACTIF**	**UNITIF**	**BUSY**

Bits	Description	
[31:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**TX or RX Reset Status (Read Only)** 0 = The reset function of TXRST or RXRST is done. 1 = Doing the reset function of TXRST or RXRST. **Note:** Both the reset operations of TXRST and RXRST need 3 system clock cycles + 2 peripheral clock cycles. User can check the status of this bit to monitor the reset function is doing or done.
[22:20]	**Reserved**	Reserved.
[19]	**TXUFIF**	**TX Underflow Interrupt Flag** When the TX underflow event occurs, this bit will be set to 1, the state of data output pin depends on the setting of TXUFPOL. 0 = No effect. 1 = No data in Transmit FIFO and TX shift register when the slave selection signal is active. **Note 1:** This bit will be cleared by writing 1 to it. **Note 2:** If reset slave's transmission circuit when slave selection signal is active, this flag will be set to 1 after 2 peripheral clock cycles + 3 system clock cycles since the reset operation is done.
[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TXTH. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TXTH.
[17]	**TXFULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** 0 = Transmit FIFO buffer is not full.

		1 = Transmit FIFO buffer is full.
[16]	**TXEMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[15]	**SPIENSTS**	**SPI Enable Status (Read Only)** 0 = The SPI controller is disabled. 1 = The SPI controller is enabled. **Note:** The SPI peripheral clock is asynchronous with the system clock. In order to make sure the SPI control logic is disabled, this bit indicates the real status of SPI controller.
[14:13]	**Reserved**	Reserved.
[12]	**RXTOIF**	**Receive Time-out Interrupt Flag** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 SPI clock periods in Master mode or over 576 peripheral clock periods in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. **Note:** This bit will be cleared by writing 1 to it.
[11]	**RXOVIF**	**Receive FIFO Overrun Interrupt Flag** When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1. 0 = No FIFO is over run. 1 = Receive FIFO over run. **Note:** This bit will be cleared by writing 1 to it.
[10]	**RXTHIF**	**Receive FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the RX FIFO buffer is smaller than or equal to the setting value of RXTH. 1 = The valid data count within the receive FIFO buffer is larger than the setting value of RXTH.
[9]	**RXFULL**	**Receive FIFO Buffer Full Indicator (Read Only)** 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[8]	**RXEMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[7]	**SLVURIF**	**Slave Mode TX Under Run Interrupt Flag** In Slave mode, if TX underflow event occurs and the slave select line goes to inactive state, this interrupt flag will be set to 1. 0 = No Slave TX under run event. 1 = Slave TX under run occurs. **Note:** This bit will be cleared by writing 1 to it.
[6]	**SLVBEIF**	**Slave Mode Bit Count Error Interrupt Flag** In Slave mode, when the slave select line goes to inactive state, if bit counter is mismatch with DWIDTH, this interrupt flag will be set to 1. 0 = No Slave mode bit count error event. 1 = Slave mode bit count error event occurs. **Note:** If the slave select is active but there is no any bus clock input, the SLVBCEIF is also active when the slave select goes to inactive state. This bit will be cleared by writing 1 to it.
[5]	**Reserved**	Reserved.

[4]	**SSLINE**	**Slave Select Line Bus Status (Read Only)** 0 = The slave select line status is 0. 1 = The slave select line status is 1. **Note:** This bit is only available in Slave mode. If SSACTPOL (SPI_SSCTL[2]) is set to 0, and the SSLINE is 1, the SPI slave select is in inactive status.
[3]	**SSINAIF**	**Slave Select Inactive Interrupt Flag** 0 = Slave select inactive interrupt was cleared or not occurred. 1 = Slave select inactive interrupt event occurred. **Note:** Only available in Slave mode. This bit will be cleared by writing 1 to it.
[2]	**SSACTIF**	**Slave Select Active Interrupt Flag** 0 = Slave select active interrupt was cleared or not occurred. 1 = Slave select active interrupt event occurred. **Note:** Only available in Slave mode. This bit will be cleared by writing 1 to it.
[1]	**UNITIF**	**Unit Transfer Interrupt Flag** 0 = No transaction has been finished since this bit was cleared to 0. 1 = SPI controller has finished one unit transfer. **Note:** This bit will be cleared by writing 1 to it.
[0]	**BUSY**	**Busy Status (Read Only)** 0 = SPI controller is in idle state. 1 = SPI controller is in busy state. The following listing are the bus busy conditions: a. SPIEN (SPI_CTL[0]) = 1 and TXEMPTY = 0. b. For SPI Master mode, SPIEN (SPI_CTL[0]) = 1 and TXEMPTY = 1 but the current transaction is not finished yet. c. For SPI Master mode, SPIEN (SPI_CTL[0]) = 1 and RXONLY = 1. d. For SPI Slave mode, the SPIEN (SPI_CTL[0]) = 1 and there is serial clock input into the SPI core logic when slave select is active. e. For SPI Slave mode, the SPIEN (SPI_CTL[0]) = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive.

SPI Data Transmit Register (SPI_TX)

Register	Offset	R/W	Description	Reset Value
SPI_TX	SPI_BA+0x20	W	SPI Data Transmit Register	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 pass through the transmitted data into the 4-level transmit FIFO buffer. The number of valid bits depends on the setting of DWIDTH (SPI_CTL[12:8]) in SPI mode or WDWIDTH (SPI_I2SCTL[5:4]) in I^{2}S mode. In SPI mode, if DWIDTH is set to 0x08, the bits TX[7:0] will be transmitted. If DWIDTH is set to 0x00, the SPI controller will perform a 32-bit transfer. In I^{2}S mode, if WDWIDTH (SPI_I2SCTL[5:4]) is set to 0x2, the data width of audio channel is 24-bit and corresponding to TX[24:0]. If WDWIDTH is set as 0x0, 0x1, or 0x3, all bits of this field are valid and referred to the data arrangement in I^{2}S mode FIFO operation section **Note:** In Master mode, SPI controller will start to transfer the SPI bus clock after 1 APB clock and 6 peripheral clock cycles after user writes to this register.

SPI Data Receive Register (SPI_RX)

Register	Offset	R/W	Description	Reset Value
SPI_RX	SPI_BA+0x30	R	SPI Data Receive Register	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** There are 4-level FIFO buffers in this controller. The data receive register holds the data received from SPI data input pin. If the RXEMPTY (SPI_STATUS[8] or SPI_I2SSTS[8]) is not set to 1, the receive FIFO buffers can be accessed through software by reading this register. This is a read only register.

I^{2}S Control Register (SPI_I2SCTL)

Register	Offset	R/W	Description	Reset Value
SPI_I2SCTL	SPI_BA+0x60	R/W	I^{2}S Control Register	0x0000_0000

Note: Not supported in SPI mode.

31	30	29	28	27	26	25	24
Reserved		**FORMAT**		**Reserved**		**LZCIEN**	**RZCIEN**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXLCH	**Reserved**					**LZCEN**	**RZCEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MCLKEN	**Reserved**						**SLAVE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ORDER	**MONO**	**WDWIDTH**		**MUTE**	**RXEN**	**TXEN**	**I2SEN**

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:28]	**FORMAT**	**Data Format Selection** 00 = I^{2}S data format. 01 = MSB justified data format. 10 = PCM mode A. 11 = PCM mode B.
[27:26]	**Reserved**	Reserved.
[25]	**LZCIEN**	**Left Channel Zero-cross Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and left channel zero-cross event occurs. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[24]	**RZCIEN**	**Right Channel Zero-cross Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and right channel zero-cross event occurs. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[23]	**RXLCH**	**Receive Left Channel Enable Bit** When monaural format is selected (MONO = 1), I^{2}S controller will receive right channel data if RXLCH is set to 0, and receive left channel data if RXLCH is set to 1. 0 = Receive right channel data in Mono mode. 1 = Receive left channel data in Mono mode.
[22:18]	**Reserved**	Reserved.
[17]	**LZCEN**	**Left Channel Zero Cross Detection Enable Bit** If this bit is set to 1, when left channel data sign bit changes or next shift data bits are all 0 then LZCIF flag in SPI_I2SSTS register is set to 1. This function is only available in transmit operation. 0 = Left channel zero cross detection Disabled. 1 = Left channel zero cross detection Enabled.

[16]	**RZCEN**	**Right Channel Zero Cross Detection Enable Bit** If this bit is set to 1, when right channel data sign bit change or next shift data bits are all 0 then RZCIF flag in SPI_I2SSTS register is set to 1. This function is only available in transmit operation. 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^{2}S controller will generate master clock on I2Sn_MCLK pin for external audio devices. 0 = Master clock Disabled. 1 = Master clock Enabled. **Note:** n = 0.
[8]	**SLAVE**	**Slave Mode** I^{2}S can operate as master or slave. For Master mode, I2Sn_BCLK and I2Sn_LRCLK pins are output mode and send bit clock from the NUC121/125 series to audio CODEC chip. In Slave mode, I2Sn_BCLK and I2Sn_LRCLK pins are input mode and I2Sn_BCLK and I2Sn_LRCLK signals are received from outer audio CODEC chip. 0 = Master mode. 1 = Slave mode. **Note:** n = 0.
[7]	**ORDER**	**Stereo Data Order in FIFO** 0 = Left channel data at high byte. 1 = Left channel data at low byte.
[6]	**MONO**	**Monaural Data** 0 = Data is stereo format. 1 = Data is monaural format.
[5:4]	**WDWIDTH**	**Word Width** 00 = data size is 8-bit. 01 = data size is 16-bit. 10 = data size is 24-bit. 11 = data size 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 receive Disabled. 1 = Data receive Enabled.
[1]	**TXEN**	**Transmit Enable Bit** 0 = Data transmit Disabled. 1 = Data transmit Enabled.
[0]	**I2SEN**	**I^{2}S Controller Enable Bit** 0 = Disabled I^{2}S mode. 1 = Enabled I^{2}S mode. **Note 1:** If enable this bit, I2Sn_BCLK will start to output in Master mode. n = 0. **Note 2:** Before changing the configurations of SPI_I2SCTL, SPI_I2SCLK, and SPI_FIFOCTL registers, user shall clear the I2SEN (SPI_I2SCTL[0]) and confirm the I2SENSTS (SPI_I2SSTS[15]) is 0.

I^{2}S Clock Divider (SPI_I2SCLK)

Register	Offset	R/W	Description	Reset Value
SPI_I2SCLK	SPI_BA+0x64	R/W	I^{2}S Clock Divider Control Register	0x0000_0000

Note: Not supported in SPI mode.

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16:8]	**BCLKDIV**	**Bit Clock Divider** The I^{2}S controller will generate bit clock in Master mode. The clock frequency of bit clock, f_{BCLK}, is determined by the following expression: $$f_{BCLK} = \frac{f_{i2s\_clock\_src}}{2 \times (\mathrm{BCLKDIV}+1)}$$ where $f_{i2s\_clock\_src}$ is the frequency of I^{2}S peripheral clock source, which is defined in the clock control register CLK_CLKSEL2. In I^{2}S Slave mode, this field is used to define the frequency of peripheral clock and it's determined by $f_{i2s\_clock\_src} \div \left(\frac{\mathrm{BCLKDIV}}{2}+1\right)$. The peripheral clock frequency in I^{2}S Slave mode must be equal to or faster than 6 times of input bit clock. **Note:** User should set BCLKDIV carefully because the peripheral clock frequency must be slower than or equal to system frequency
[7:6]	**Reserved**	Reserved.

[5:0]	**MCLKDIV**	**Master Clock Divider** If MCLKEN is set to 1, I^{2}S controller will generate master clock for external audio devices. The frequency of master clock, f_{MCLK}, is determined by the following expressions: If MCLKDIV >= 1,. $f_{MCLK} = \frac{f_{i2s\_clock\_src}}{2 \times \mathrm{MCLKDIV}}$ If MCLKDIV = 0,. $f_{MCLK} = f_{i2s\_clock\_src}$ where $f_{i2s\_clock\_src}$ is the frequency of I^{2}S peripheral clock source, which is defined in the clock control register CLK_CLKSEL2. In general, the master clock rate is 256 times sampling clock rate.

I^{2}S Status Register (SPI_I2SSTS)

Register	Offset	R/W	Description	Reset Value
SPI_I2SSTS	SPI_BA+0x68	R/W	I^{2}S Status Register	0x0005_0100

Note: Not supported in SPI mode.

31	30	29	28	27	26	25	24
Reserved	TXCNT			Reserved	RXCNT		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TXRXRST	Reserved	LZCIF	RZCIF	TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
I2SENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			RIGHT	Reserved			

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27]	**Reserved**	Reserved.
[26:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**TX or RX Reset Status (Read Only)** 0 = The reset function of TXRST or RXRST is done. 1 = Doing the reset function of TXRST or RXRST. **Note:** Both the reset operations of TXRST and RXRST need 3 system clock cycles + 2 peripheral clock cycles. User can check the status of this bit to monitor the reset function is doing or done.
[22]	**Reserved**	Reserved.
[21]	**LZCIF**	**Left Channel Zero Cross Interrupt Flag** 0 = No zero cross event occurred on left channel. 1 = Zero cross event occurred on left channel.
[20]	**RZCIF**	**Right Channel Zero Cross Interrupt Flag** 0 = No zero cross event occurred on right channel. 1 = Zero cross event occurred on right channel.
[19]	**TXUFIF**	**Transmit FIFO Underflow Interrupt Flag** When the transmit FIFO buffer is empty and there is no datum written into the FIFO buffer, if there is more bus clock input, this bit will be set to 1. **Note:** This bit will be cleared by writing 1 to it.
[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TXTH.

		1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TXTH. **Note:** If TXTHIEN = 1 and TXTHIF = 1, the SPI controller will generate a SPI interrupt request.
[17]	**TXFULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.
[16]	**TXEMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** 0 = Transmit FIFO buffer is not empty. 1 = Transmit FIFO buffer is empty.
[15]	**I2SENSTS**	**I^{2}S Enable Status (Read Only)** 0 = The SPI/I^{2}S control logic is disabled. 1 = The SPI/I^{2}S control logic is enabled. **Note:** The SPI peripheral clock is asynchronous with the system clock. In order to make sure the SPI/I^{2}S controller logic is disabled, this bit indicates the real status of SPI/I^{2}S controller logic for user.
[14:13]	**Reserved**	Reserved.
[12]	**RXTOIF**	**Receive Time-out Interrupt Flag** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 SPI clock period in Master mode or over 576 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 it.
[11]	**RXOVIF**	**Receive FIFO Overrun Interrupt Flag** When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1. **Note:** This bit will be cleared by writing 1 to it.
[10]	**RXTHIF**	**Receive FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the Rx FIFO buffer is smaller than or equal to the setting value of RXTH. 1 = The valid data count within the receive FIFO buffer is larger than the setting value of RXTH. **Note:** If RXTHIEN = 1 and RXTHIF = 1, the SPI/I^{2}S controller will generate a SPI interrupt request.
[9]	**RXFULL**	**Receive FIFO Buffer Full Indicator (Read Only)** 0 = Receive FIFO buffer is not full. 1 = Receive FIFO buffer is full.
[8]	**RXEMPTY**	**Receive FIFO Buffer Empty Indicator (Read Only)** 0 = Receive FIFO buffer is not empty. 1 = Receive FIFO buffer is empty.
[7:5]	**Reserved**	Reserved.
[4]	**RIGHT**	**Right Channel (Read Only)** This bit indicates the current transmit data is belong to which channel. 0 = Left channel. 1 = Right channel.
[3:0]	**Reserved**	Reserved.

6.19 USB Device Controller (USBD)

6.19.1 Overview

There is one set of USB 2.0 full-speed device controller and transceiver in this device. It is compliant with USB 2.0 full-speed device specification and supports control/bulk/interrupt/isochronous transfer types. It implements a full-speed (12 Mbit/s) function interface with added support for USB 2.0 Link Power Management.

In this device controller, there are two main interfaces: the APB bus and USB bus which comes from the USB PHY transceiver. For the APB bus, the CPU can program control registers through it. There are 768 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 (USBD_BUFSEGn, n=0~7).

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 states, current start address, transaction status, and data buffer status for each endpoint.

There are five different interrupt events in this controller. They are the SOF event, no-event-wake-up, device plug-in or plug-out event, USB events, like IN ACK, OUT ACK etc, and BUS events, like suspend and resume, etc. Any event will cause an interrupt, and users just need to check the related event flags in interrupt event status register (USBD_INTSTS) to acknowledge what kind of interrupt occurring, and then check the related USB Endpoint Status Register (USBD_EPSTS) to acknowledge what kind of event occurring in this endpoint.

A software-disconnect function is also supported for this USB controller. It is used to simulate the disconnection of this device from the host. If user enables SE0 bit (USBD_SE0), the USB controller will force the output of USB_D+ and USB_D- to level low and its function is disabled. After disable the SE0 bit, host will enumerate the USB device again.

For more information on the Universal Serial Bus, please refer to *Universal Serial Bus Specification Revision 1.1.*

6.19.2 Features

- Compliant with USB 2.0 Full-Speed specification
- Provides 1 interrupt vector with 5 different interrupt events (NEVWK, VBUSDET, USB, BUS and SOF)
- Supports Control/Bulk/Interrupt/Isochronous transfer type
- Supports suspend function when no bus activity existing for 3 ms
- Supports 8 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 768 bytes buffer size
- Provides remote wake-up capability
- Supports USB 2.0 Link Power Management

6.19.3 Block Diagram

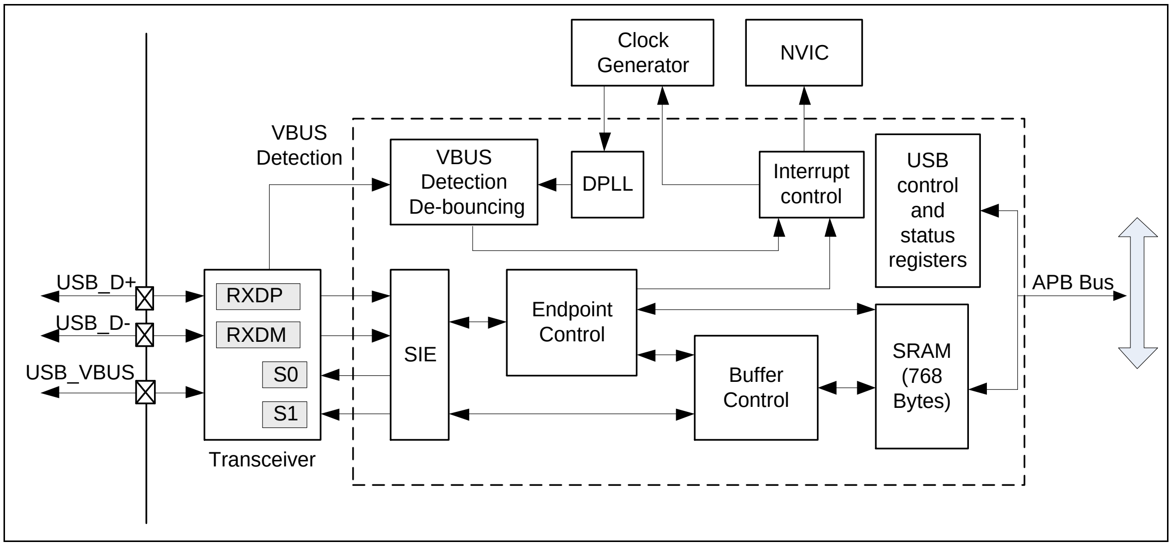


Figure 6.19-1 USB Block Diagram

6.19.4 Basic Configuration

The USBD clock source could be derived from internal 48MHz RC oscillator or PLL. User has to enable the 48MHz RC oscillator or set the PLL related configurations before USB device controller is enabled. When USB device clock source stable,user can set the USBDCKEN(CLK_APBCLK0[27])) bit to enable USBD clock and 4-bit pre-scaler USBDIV (CLK_CLKDIV0[7:4]) to generate the proper USBD clock rate if PLL is used as USBD clock source. USBDRST (SYS_IPRST1[27]) is used to reset USBD. Set USBDRST=1 to force USBD reset and clear reset by set USBDRST = 0.

6.19.5 Functional Description

6.19.5.1 Serial Interface Engine (SIE)

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

6.19.5.2 Endpoint Control

This controller supports 8 endpoints. Each of the endpoint can be configured as Control, Bulk, Interrupt, or Isochronous transfer type. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. It is also used to manage the data sequential synchronization, endpoint state control, current endpoint start address, current transaction status, and data buffer status in each endpoint.

6.19.5.3 Digital Phase Lock Loop (DPLL)

The bit rate of USB data is 12 MHz. The DPLL uses the 48 MHz which comes from the clock controller to lock the input data RXDP and RXDM. The 12 MHz bit rate clock is also converted from DPLL.

6.19.5.4 VBUS Detection De-bouncing

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-bouncing for USB VBUS detection interrupt to avoid bounce problems on USB plug-in or unplug. VBUS detection interrupt appears about 10 ms later than USB plug-in or plug-out. User can acknowledge USB plug-in/plug-out by reading USBD_VBUSDET register. The VBUSDET flag represents the current state on the bus without de-bouncing. If VBUSDET is 1, it means the USB cable is plugged-in. If user polls the flag to check USB state, software de-bouncing must be added if needed.

6.19.5.5 Interrupt control

This USB provides 1 interrupt vector with 5 interrupt events (NEVWK, VBUSDET, USB, BUS and SOF). The NEVWK event occurs after waking up the system from Power-down mode (The power mode function is defined in system power-down control register, CLK_PWRCTL). The VBUSDET event is used for USB plug-in or unplug. The USB event notifies users of some USB requests, such as IN ACK, OUT ACK, and the BUS event notifies users of some bus events, such as suspend and resume. The number of SOF can be configured by SOFITH(USBD_ATTR[18:16]) bits to occur one SOF interrupt event. The related bits must be set in the interrupt enable register (USBD_INTEN) of USB Device Controller to enable USB interrupts.

NEVWK interrupt is only presented when no the other USB interrupt events happened more than 20ms after the chip is waked up from Power-down mode. After the chip enters Power-down mode, any change on USB_VBUS, USB_D+ and USB_D- can wake up this chip if USB wake-up function is enabled. If this change is not intentionally, no interrupt but NEVWK interrupt will occur. After waking up by USB, this interrupt will occur when no the other USB interrupt events are presented for more than 20ms. Figure 6.19-2 is the control flow of wake-up interrupt.

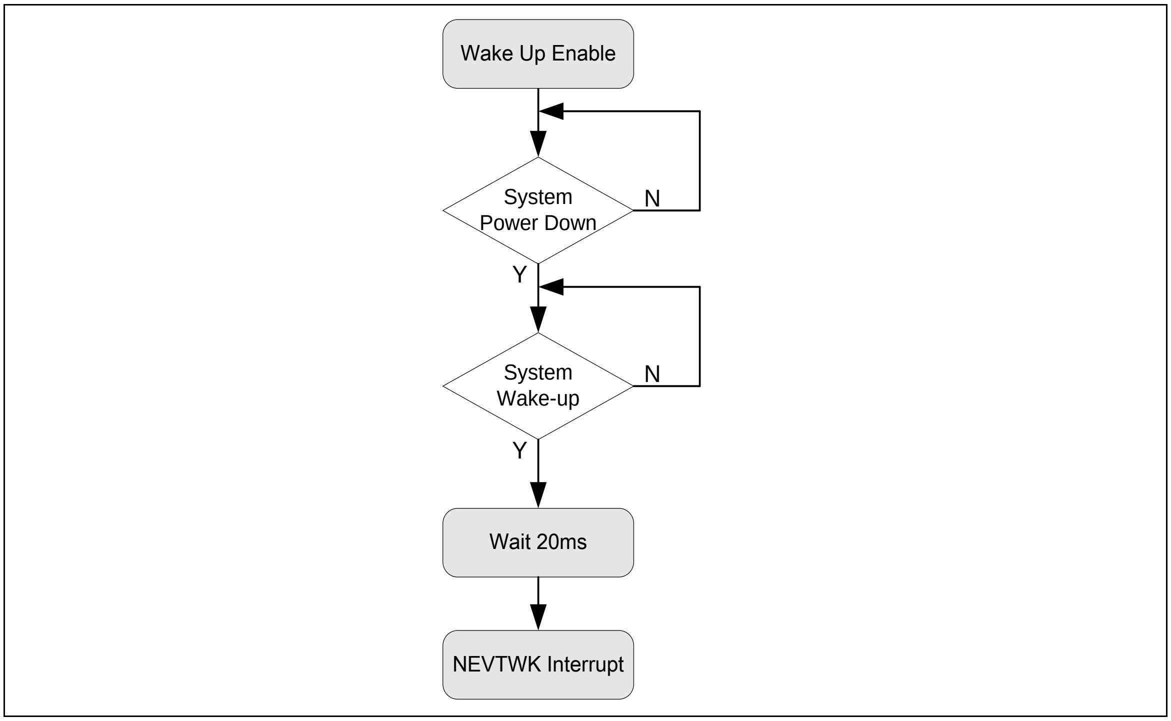


Figure 6.19-2 NEVWK Interrupt Operation Flow

The USB interrupt is used to notify users of any USB event on the bus, and user can read EPSTS (USBD_EPSTS[31:8]) and EPEVT7~0 (USBD_INTSTS[23:16]) to take necessary responses.

Same as USB interrupt, BUS interrupt notifies users of some bus events, like USB reset, suspend, time-out, and resume. User can read USBD_ATTR to acknowledge bus events.

6.19.5.6 Power Saving

User can write 0 to USBD_ATTR[4] to disable PHY under special circumstances, like suspend, to conserve power.

6.19.5.7 Buffer Control

There is 768 bytes SRAM in the controller and the 8 endpoints share this buffer. User shall configure each endpoint's effective starting address in the buffer segmentation register before the USB function active. The "Buffer Control" block is used to control each endpoint's effective starting address and its SRAM size is defined in the USBD_MXPLDn (n=0~7) register.

Figure 6.19-3 depicts the starting address for each endpoint according the content of USBD_BUFSEGn (n=0~7) and USBD_MXPLDn (n=0~7) registers. If the USBD_BUFSEG0 is programmed as 0x08 and USBD_MXPLD0 is set as 0x40, the SRAM size of endpoint 0 is start from USBD_BA+0x108 and end in USBD_BA+0x147. (**Note:** The USBD SRAM base is USBD_BA+0x100).

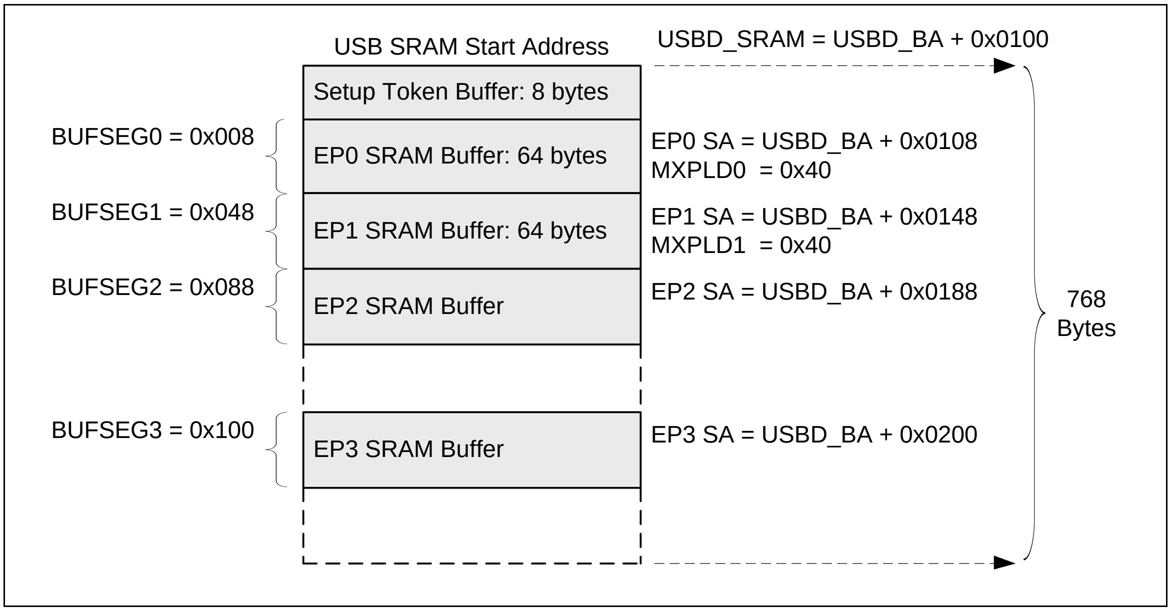

USB SRAM Start Address
USBD_SRAM = USBD_BA + 0x0100
Setup Token Buffer: 8 bytes
BUFSEG0 = 0x008
EP0 SRAM Buffer: 64 bytes
EP0 SA = USBD_BA + 0x0108
MXPLD0 = 0x40
BUFSEG1 = 0x048
EP1 SRAM Buffer: 64 bytes
EP1 SA = USBD_BA + 0x0148
MXPLD1 = 0x40
BUFSEG2 = 0x088
EP2 SRAM Buffer
EP2 SA = USBD_BA + 0x0188
768
Bytes
BUFSEG3 = 0x100
EP3 SRAM Buffer
EP3 SA = USBD_BA + 0x0200

Figure 6.19-3 Endpoint SRAM Structure

6.19.5.8 Handling Transactions with USB Device Peripheral

User can use interrupt or polling USBD_INTSTS to monitor the USB transactions. When transactions occur, USBD_INTSTS will be set by hardware and send an interrupt request to CPU (if related interrupt enabled), or user can polling USBD_INTSTS to get these events without interrupt. The following is the control flow with interrupt enabled.

When USB host has requested data from a device controller, user needs to prepare related data in the specified endpoint buffer in advance. After buffering the required data, user needs to write the actual data length in the specified USBD_MXPLDn (n=0~7) 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.

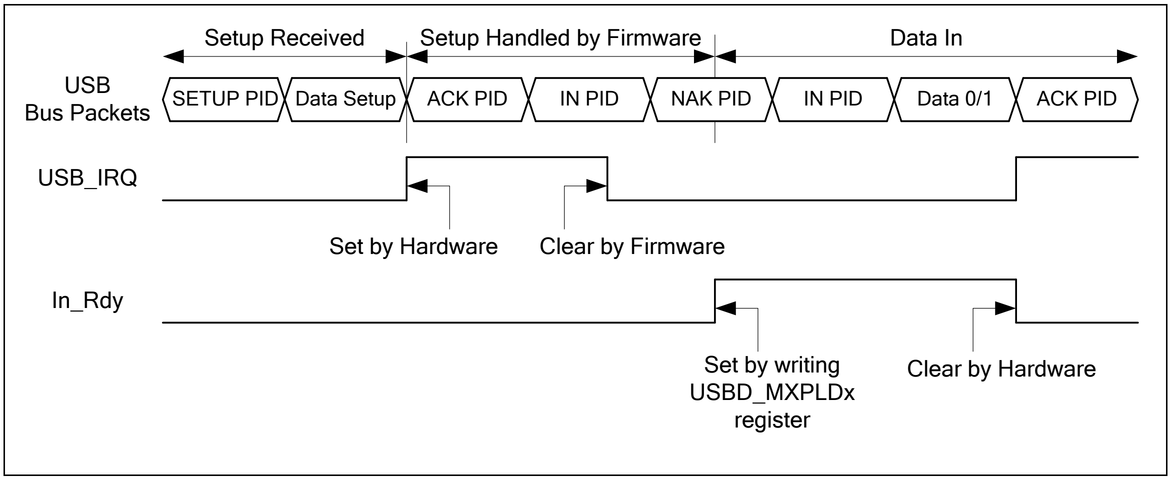


Figure 6.19-4 Setup Transaction Followed by Data IN Transaction

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 specified USBD_MXPLDn (n=0~7) register and de-assert the internal signal “Out_Rdy”. This will avoid hardware accepting next transaction until user moves out the current data in the related endpoint buffer. Once users have processed this transaction, the specified USBD_MXPLDn (n=0~7) register needs to be written by firmware to assert the signal “Out_Rdy” again to accept the next transaction.

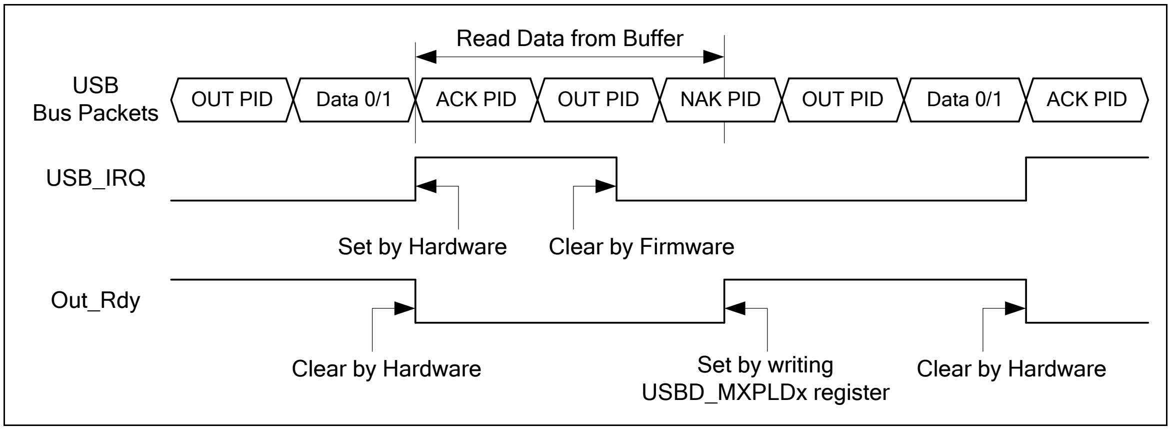


Figure 6.19-5 Data Out Transfer

6.19.5.9 Link Power Management(LPM)

Power Management(LPM) which is similar to the suspend/resume function, but has transitional latencies of tens of microseconds between power states (instead of three to greater than 20 millisecond latencies of the USB2.0 suspend/resume)

New fast mechanism for transitioning the bus on a root port from an enable state (called L0), to a new Sleep state(called L1), detail define for L0 and L1 state see table 6.19-1, the register USBD_ATTR & USBD_LPMATTR can let user know current power state for LPM mechanism.

LPM State	**Description**
L0(On)	In this state, the port is enabled for propagation of transaction signaling traffic. A port in L0 is either actively transmitting or receiving data (L0-Active) or able to do so but not currently transmitting or receiving information (L0-Idle). While in this state Start-of-Frame (SOF) packets are issued by the host at a rate corresponding to the speed of the client device
L1(Sleep)	L1 is similar to L2 (below) but supports finer granularity in use. When in L1, the line state is identical to L2. Entry to L1 is started by a request to a hub or host port to transition to L1. A LPM transaction is sent to the downstream device. The requested transition can only occur if the device response with an ACK handshake. Exit from L1 is via remote wake, resume signaling, reset signaling or disconnect. L1 does not impose any specific power draw requirements (from VBUS) on the attached device as L2 does. Either the host or device can initiate resume signaling when in L1. Although the signaling levels of resume are the same as L2, the duration of the signaling and transitional latencies associated with the L1 to L0 transition are much shorter
L2(Suspend)	This is the formalized name for USB 2.0 Suspend, Entry to L2 is nominally triggered by a command to a hub or host port to transition to suspend. The device discovers the suspend condition via observing 3ms of inactivity. The resultant line state is either Low or Full-speed idle. L2 also imposes power draw requirements (from VBUS) on the attached device. Exit from this state is via remote wake, resume signaling, reset signaling or disconnect.
L3(Off)	In this state, the port is not capable of performing any data signaling. It corresponds to the powered-off, disconnected, and disabled states

Table 6.19-1 USB Link Power Manager (Lx) States

The state transaction process please refer to Figure 6.19-6, and for more information on the USB Link Power Manager(LPM), please refer to USB2.0 Link Power Mangement ECN.

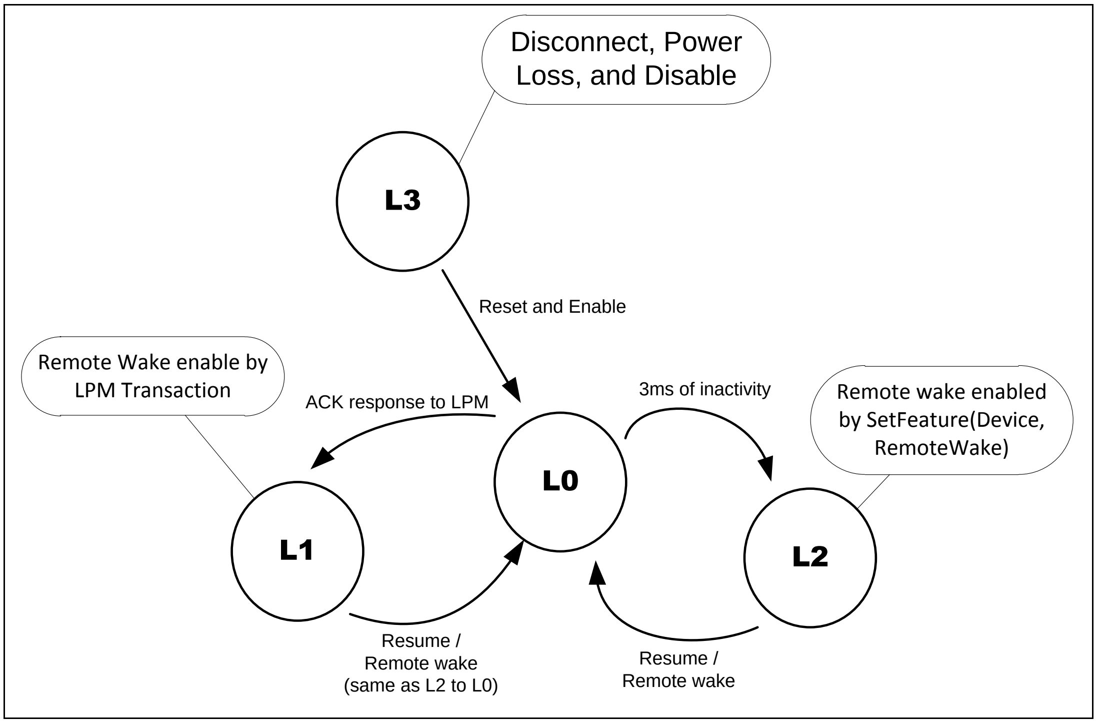

Disconnect, Power Loss, and Disable
L3
Reset and Enable
Remote Wake enable by LPM Transaction
ACK response to LPM
3ms of inactivity
Remote wake enabled by SetFeature(Device, RemoteWake)
L0
L1
L2
Resume / Remote wake (same as L2 to L0)
Resume / Remote wake

Figure 6.20-6 LPM State Transition Diagram

6.19.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
USBD Base Address: **USBD_BA = 0x4006_0000**				
USBD_INTEN	USBD_BA+0x000	R/W	USB Device Interrupt Enable Register	0x0000_0000
USBD_INTSTS	USBD_BA+0x004	R/W	USB Device Interrupt Event Status Register	0x0000_0000
USBD_FADDR	USBD_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000
USBD_EPSTS	USBD_BA+0x00C	R	USB Device Endpoint Status Register	0x0000_0000
USBD_ATTR	USBD_BA+0x010	R/W	USB Device Bus Status and Attribution Register	0x0000_0040
USBD_VBUSDET	USBD_BA+0x014	R	USB Device VBUS Detection Register	0x0000_0000
USBD_STBUFSEG	USBD_BA+0x018	R/W	Setup Token Buffer Segmentation Register	0x0000_0000
USBD_LPMATTR	USBD_BA+0x088	R	USB LPM Attribution Register	0x0000_0000
USBD_FN	USBD_BA+0x08C	R	USB Frame number Register	0x0000_0XXX
USBD_SE0	USBD_BA+0x090	R/W	USB Device Drive SE0 Control Register	0x0000_0001
USBD_BUFSEG0	USBD_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD0	USBD_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USBD_CFG0	USBD_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USBD_CFGP0	USBD_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG1	USBD_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD1	USBD_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USBD_CFG1	USBD_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USBD_CFGP1	USBD_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG2	USBD_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD2	USBD_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USBD_CFG2	USBD_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USBD_CFGP2	USBD_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG3	USBD_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD3	USBD_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USBD_CFG3	USBD_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USBD_CFGP3	USBD_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

USBD_BUFSEG4	USBD_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD4	USBD_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USBD_CFG4	USBD_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USBD_CFGP4	USBD_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG5	USBD_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD5	USBD_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USBD_CFG5	USBD_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USBD_CFGP5	USBD_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG6	USBD_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD6	USBD_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USBD_CFG6	USBD_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USBD_CFGP6	USBD_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG7	USBD_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD7	USBD_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000
USBD_CFG7	USBD_BA+0x578	R/W	Endpoint 7 Configuration Register	0x0000_0000
USBD_CFGP7	USBD_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

Memory Type	**Address**	**Size**	**Description**
USBD_BA = 0x4006_0000			
USBD_SRAM	USBD_BA+0x100 ~ USBD_BA+0x3FF	768 Bytes	The SRAM is used for the entire endpoints buffer. Refer to section 6.19.5.7 for the endpoint SRAM structure and its description.

6.19.7 Register Description

USB Interrupt Enable Register (USBD_INTEN)

Register	Offset	R/W	Description	Reset Value
USBD_INTEN	USBD_BA+0x000	R/W	USB Device 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
INNAKEN	Reserved						WKEN
7	6	5	4	3	2	1	0
Reserved			SOFIEN	NEVWKIEN	VBDETIEN	USBIEN	BUSIEN

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**INNAKEN**	**Active NAK Function and Its Status in IN Token** 0 = When device responds NAK after receiving IN token, IN NAK status will not be updated to USBD_EPSTS register, 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]	**WKEN**	**Wake-up Function Enable Bit** 0 = USB wake-up function Disabled. 1 = USB wake-up function Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**SOFIEN**	**Start of Frame Interrupt Enable Bit** 0 = SOF Interrupt Disabled. 1 = SOF Interrupt Enabled.
[3]	**NEVWKIEN**	**USB No-event-wake-up Interrupt Enable Bit** 0 = No-event-wake-up Interrupt Disabled. 1 = No-event-wake-up Interrupt Enabled.
[2]	**VBDETIEN**	**VBUS Detection Interrupt Enable Bit** 0 = VBUS detection Interrupt Disabled. 1 = VBUS detection Interrupt Enabled.
[1]	**USBIEN**	**USB Event Interrupt Enable Bit** 0 = USB event interrupt Disabled. 1 = USB event interrupt Enabled.

[0]	**BUSIEN**	**Bus Event Interrupt Enable Bit** 0 = BUS event interrupt Disabled. 1 = BUS event interrupt Enabled.

USB Interrupt Event Status Register (USBD_INTSTS)

Register	Offset	R/W	Description	Reset Value
USBD_INTSTS	USBD_BA+0x004	R/W	USB Device 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			SOFIF	NEVWKIF	VBDETIF	USBIF	BUSIF

Bits	Description	
[31]	**SETUP**	**Setup Event Status** 0 = No Setup event. 1 = Setup event occurred, cleared by write 1 to USBD_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 USBD_EPSTS[31:29] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[23] or USBD_INTSTS[1].
[22]	**EPEVT6**	**Endpoint 6's USB Event Status** 0 = No event occurred in endpoint 6. 1 = USB event occurred on Endpoint 6, check USBD_EPSTS[28:26] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[22] or USBD_INTSTS[1].
[21]	**EPEVT5**	**Endpoint 5's USB Event Status** 0 = No event occurred in endpoint 5. 1 = USB event occurred on Endpoint 5, check USBD_EPSTS[25:23] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[21] or USBD_INTSTS[1].
[20]	**EPEVT4**	**Endpoint 4's USB Event Status** 0 = No event occurred in endpoint 4. 1 = USB event occurred on Endpoint 4, check USBD_EPSTS[22:20] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[20] or USBD_INTSTS[1].
[19]	**EPEVT3**	**Endpoint 3's USB Event Status** 0 = No event occurred in endpoint 3. 1 = USB event occurred on Endpoint 3, check USBD_EPSTS[19:17] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[19] or USBD_INTSTS[1].

[18]	**EPEVT2**	**Endpoint 2's USB Event Status** 0 = No event occurred in endpoint 2. 1 = USB event occurred on Endpoint 2, check USBD_EPSTS[16:14] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[18] or USBD_INTSTS[1].
[17]	**EPEVT1**	**Endpoint 1's USB Event Status** 0 = No event occurred in endpoint 1. 1 = USB event occurred on Endpoint 1, check USBD_EPSTS[13:11] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[17] or USBD_INTSTS[1].
[16]	**EPEVT0**	**Endpoint 0's USB Event Status** 0 = No event occurred in endpoint 0. 1 = USB event occurred on Endpoint 0, check USBD_EPSTS[10:8] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[16] or USBD_INTSTS[1].
[15:5]	**Reserved**	Reserved.
[4]	**SOFIF**	**Start of Frame Interrupt Status** 0 = SOF event does not occur. 1 = SOF event occurred, cleared by write 1 to USBD_INTSTS[4].
[3]	**NEVWKIF**	**No-event-wake-up Interrupt Status** 0 = NEVWK event does not occur. 1 = No-event-wake-up event occurred, cleared by write 1 to USBD_INTSTS[3].
[2]	**VBDETIF**	**VBUS Detection Interrupt Status** 0 = There is not attached/detached event in the USB. 1 = There is attached/detached event in the USB bus and it is cleared by write 1 to USBD_INTSTS[2].
[1]	**USBIF**	**USB Event Interrupt Status** The USB event includes the SETUP Token, IN Token, OUT ACK, ISO IN, or ISO OUT events in the bus. 0 = No USB event occurred. 1 = USB event occurred, check EPSTS0~5[2:0] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[1] or EPSTS0~7 and SETUP (USBD_INTSTS[31]).
[0]	**BUSIF**	**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 USBD_ATTR[3:0] and USBD_ATTR[13:12] to know which kind of bus event was occurred, cleared by write 1 to USBD_INTSTS[0].

USB Device Function Address Register (USBD_FADDR)

A 7-bit value is used as the address of a device on the USB BUS.

Register	Offset	R/W	Description	Reset Value
USBD_FADDR	USBD_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:0]	**FADDR**	USB Device Function Address

USB Endpoint Status Register (USBD_EPSTS)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS	USBD_BA+0x00C	R	USB Device 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**
OV	Reserved						

Bits	Description	
[31:29]	**EPSTS7**	**Endpoint 7 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 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 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 Status** These bits are used to indicate the current status of this endpoint 000 = In ACK.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

		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 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 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 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 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]	**OV**	**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.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

USB Bus Status and Attribution Register (USBD_ATTR)

Register	Offset	R/W	Description	Reset Value
USBD_ATTR	USBD_BA+0x010	R/W	USB Device 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					**SOFITH**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**L1RESUME**	**L1SUSPEND**	**LPMACK**	**BYTEM**	**PWRDN**	**DPPUEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBEN	**Reserved**	**RWAKEUP**	**PHYEN**	**TOUT**	**RESUME**	**SUSPEND**	**USBRST**

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18:16]	**SOFITH**	**Start of Frame Interrupt Threshold Control Bits** Configure the number of SOF to trigger SOF interrupt if SOF interrupt is enabled (USBD_INTEN[4]=1). 000 = every SOF trigger one SOF interrupt. 001 = every 2 SOFs trigger one SOF interrupt. 010 = every 4 SOFs trigger one SOF interrupt. 011 = every 8 SOFs trigger one SOF interrupt. 100 = every 16 SOFs trigger one SOF interrupt. 101 = every 32 SOFs trigger one SOF interrupt. 110 = every 64 SOFs trigger one SOF interrupt. 111 = every 128 SOFs trigger one SOF interrupt.
[15:14]	**Reserved**	Reserved.
[13]	**L1RESUME**	**LPM L1 Resume** 0 = Bus no LPM L1 state resume. 1 = LPM L1 state Resume from LPM L1 state suspend. **Note:** This bit is read only.
[12]	**L1SUSPEND**	**LPM L1 Suspend** 0 = Bus no L1 state suspend. 1 = This bit is set by the hardware when LPM command to enter the L1 state is successfully received and acknowledged. **Note:** This bit is read only.
[11]	**LPMACK**	**LPM Token Acknowledge Enable Bit** The NYET/ACK will be returned only on a successful LPM transaction if no errors in both the EXT token and the LPM token and a valid bLinkState = 0001 (L1) is received, else ERROR and STALL will be returned automatically, respectively. 0= the valid LPM Token will be NYET. 1= the valid LPM Token will be ACK.

[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** 1 = Turn on related circuit of PHY transceiver. 0 = Power donw related circuit of PHY transceiver.
[8]	**DPPUEN**	**Pull-up Resistor on USB_DP Enable Bit** 0 = Pull-up resistor in USB_D+ bus Disabled. 1 = Pull-up resistor in USB_D+ bus Active.
[7]	**USBEN**	**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, USB_D-: high) state, used for remote wake-up.
[4]	**PHYEN**	**PHY Transceiver Function Enable Bit** 0 = PHY transceiver function Disabled. 1 = PHY transceiver function Enabled.
[3]	**TOUT**	**Time-out Status** 0 = No time-out. 1 = No Bus response more than 18 bits time. **Note:** This bit is read only.
[2]	**RESUME**	**Resume Status** 0 = No bus resume. 1 = Resume from suspend. **Note:** This bit is read only.
[1]	**SUSPEND**	**Suspend Status** 0 = Bus no suspend. 1 = Bus idle more than 3ms, either cable is plugged off or host is sleeping. **Note:** This bit is read only.
[0]	**USBRST**	**USB Reset Status** 0 = Bus no reset. 1 = Bus reset when SE0 (single-ended 0) more than 2.5us. **Note:** This bit is read only.

USB Device VBUS Detection Register (USBD_VBUSDET)

Register	Offset	R/W	Description	Reset Value
USBD_VBUSDET	USBD_BA+0x014	R	USB Device VBUS Detection 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							**VBUSDET**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**VBUSDET**	**Device VBUS Detection** 0 = Controller is not attached to the USB host. 1 = Controller is attached to the USB host.

USB Setup Token Buffer Segmentation Register (USBD_STBUFSEG)

Register	Offset	R/W	Description	Reset Value
USBD_STBUFSEG	USBD_BA+0x018	R/W	Setup Token Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:3]	**STBUFSEG**	**SETUP Token Buffer Segmentation** It is used to indicate the offset address for the SETUP token with the USB Device SRAM starting address. The effective starting address is "USBD_SRAM address + offset " {offset = BUFSEG[9:3] * 8} . Where the USBD_SRAM address = USBD_BA+0x100h. **Note:** It is used for SETUP token only.
[2:0]	**Reserved**	Reserved.

NUC121/125 SERIES TECHNICAL REFERENCE MANUAL

USB LPM Attribution Register (USBD_LPMATTR)

Register	Offset	R/W	Description	Reset Value
USBD_LPMATTR	USBD_BA+0x088	R	USB LPM Attribution 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							**LPMRWAKUP**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LPMBESL				**LPMLINKSTS**			

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**LPMRWAKUP**	**LPM Remote Wakeup** This bit contains the bRemoteWake value received with last ACK LPM Token
[7:4]	**LPMBESL**	**LPM Best Effort Service Latency** These bits contain the BESL value received with last ACK LPM Token 0000 = 125us. 0001 = 150us. 0010 = 200us. 0011 = 300us. 0100 = 400us. 0101 = 500us. 0110 = 1000us. 0111 = 2000us. 1000 = 3000us. 1001 = 4000us. 1010 = 5000us. 1011 = 6000us. 1100 = 7000us. 1101 = 8000us. 1110 = 9000us. 1111 = 10000us.
[3:0]	**LPMLINKSTS**	**LPM Link State** These bits contain the bLinkState received with last ACK LPM Token 0000 = Reserve. 0001 = L1 (Sleep). 0010 – 1111 = Reserve.

USB Frame Number Register (USBD_FN)

Register	Offset	R/W	Description	Reset Value
USBD_FN	USBD_BA+0x08C	R	USB Frame number Register	0x0000_0XXX

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**FN**	**Frame Number** These bits contain the 11-bits frame number in the last received SOF packet.

USB Drive SE0 Register (USBD_SE0)

Register	Offset	R/W	Description	Reset Value
USBD_SE0	USBD_BA+0x090	R/W	USB Device Drive SE0 Control Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SE0**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SE0**	**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 = Normal operation. 1 = Force USB PHY transceiver to drive SE0.

USB Buffer Segmentation Register (USB_BUFSEGn, n=0~7)

Register	Offset	R/W	Description	Reset Value
USBD_BUFSEG0	USBD_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG1	USBD_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG2	USBD_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG3	USBD_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG4	USBD_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG5	USBD_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG6	USBD_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG7	USBD_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	
[31:10]	**Reserved**	Reserved.
[9:3]	**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 USBD_SRAM address + { BUFSEG[9:3], 3'b000} Where the USBD_SRAM address = USBD_BA+0x100h. Refer to the section 6.19.5.7 for the endpoint SRAM structure and its description.
[2:0]	**Reserved**	Reserved.

USB Maximal Payload Register (USB_MXPLDn, n=0~7)

Register	Offset	R/W	Description	Reset Value
USBD_MXPLD0	USBD_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USBD_MXPLD1	USBD_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USBD_MXPLD2	USBD_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USBD_MXPLD3	USBD_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USBD_MXPLD4	USBD_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USBD_MXPLD5	USBD_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USBD_MXPLD6	USBD_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USBD_MXPLD7	USBD_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	
[31:10]	**Reserved**	Reserved.
[9:0]	**MXPLD**	**Maximal Payload** Define the data length which is transmitted to host (IN token) or the actual data length which is received from the host (OUT token). It also used to indicate that the endpoint is ready to be transmitted in IN token or received in OUT token. (1) When the register is written by CPU, For IN token, the value of MXPLD is used to define the data length to be transmitted and indicate the data buffer is ready. For OUT token, it means that the controller is ready to receive data from the host and the value of MXPLD is the maximal data length comes from host. (2) When the register is read by CPU, For IN token, the value of MXPLD is indicated by the data length be transmitted to host For OUT token, the value of MXPLD is indicated the actual data length receiving from host. **Note:** Once MXPLD is written, the data packets will be transmitted/received immediately after IN/OUT token arrived.

USB Configuration Register (USB_CFGn, n=0~7)

Register	Offset	R/W	Description	Reset Value
USBD_CFG0	USBD_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USBD_CFG1	USBD_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USBD_CFG2	USBD_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USBD_CFG3	USBD_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USBD_CFG4	USBD_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USBD_CFG5	USBD_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USBD_CFG6	USBD_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USBD_CFG7	USBD_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**
DSQSYNC	STATE		ISOCH	EPNUM			

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**CSTALL**	**Clear STALL Response** 0 = Disable the device to clear the STALL handshake in setup stage. 1 = Clear the device to response STALL handshake in setup stage.
[8]	**Reserved**	Reserved.
[7]	**DSQSYNC**	**Data Sequence Synchronization** 0 = DATA0 PID. 1 = DATA1 PID. **Note:** It is used to specify the DATA0 or DATA1 PID in the following IN token transaction. Hardware will toggle automatically in IN token base on the bit.
[6:5]	**STATE**	**Endpoint STATE** 00 = Endpoint is Disabled. 01 = Out endpoint. 10 = IN endpoint. 11 = Undefined.
[4]	**ISOCH**	**Isochronous Endpoint**

		This bit is used to set the endpoint as Isochronous endpoint, no handshake. 0 = No Isochronous endpoint. 1 = Isochronous endpoint.
[3:0]	**EPNUM**	**Endpoint Number** These bits are used to define the endpoint number of the current endpoint

USB Extra Configuration Register (USB_CFGPn, n=0~7)

Register	Offset	R/W	Description	Reset Value
USBD_CFGP0	USBD_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP1	USBD_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP2	USBD_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP3	USBD_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP4	USBD_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP5	USBD_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP6	USBD_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP7	USBD_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 = Disable the device to response STALL. 1 = Set the device to respond STALL automatically.
[0]	**CLRRDY**	**Clear Ready** When the USBD_MXPLDn (n=0~7) register is set by user, it means that the endpoint is ready to transmit or receive data. If the user wants to disable this transaction before the transaction start, users can set this bit to 1 to disable it and it is auto clear to 0. For IN token, write '1' to clear the IN token had ready to transmit the data to USB. For OUT token, write '1' to clear the OUT token had ready to receive the data from USB. This bit is write 1 only and is always 0 when it is read back.

6.20 Analog-to-Digital Converter (ADC)

6.20.1 Overview

The NUC121/125 series contains one 12-bit successive approximation analog-to-digital converter (SAR A/D converter) with 14 input channels. The A/D converter supports four operation modes: Single, Burst, Single-cycle Scan and Continuous Scan mode. The A/D converter can be started by software, external pin (STADC/PC.8), timer0~3 overflow pulse trigger and PWM trigger.

6.20.2 Features

- Analog input voltage range: 0 ~ AV_{DD}.
- 12-bit resolution and 10-bit accuracy is guaranteed
- Up to 12 single-end analog input channels or 6 differential analog input channels
- Maximum ADC peripheral clock frequency is 16 MHz
- Up to 800k SPS sampling rate
- Configurable ADC internal sampling time
- Four operation modes:
  - Single mode: A/D conversion is performed one time on a specified channel.
  - Burst mode: A/D converter samples and converts the specified single channel and sequentially stores the result in FIFO.
  - Single-cycle Scan mode: A/D conversion is performed only one cycle on all specified channels with the sequence from the smallest numbered channel to the largest numbered channel.
  - Continuous Scan mode: A/D converter continuously performs Single-cycle Scan mode until software stops A/D conversion.
- An A/D conversion can be started by:
  - Software Write 1 to ADST bit
  - External pin (STADC)
  - Timer 0~3 overflow pulse trigger
  - PWM trigger with optional start delay period
- Each conversion result is held in data register of each channel with valid and overrun indicators.
- Conversion result can be compared with specified value and user can select whether to generate an interrupt when conversion result matches the compare register setting.
- Two internal channels which are band-gap voltage (VBG) and temperature sensor (VTEMP).
- Supports PDMA transfer mode.

Note 1: ADC sampling rate = (ADC peripheral clock frequency) / (total ADC conversion cycle)

Note 2: If the internal channel (VTEMP) is selected to convert, the sampling rate needs to be less than 300k SPS for accurate result.

6.20.3 Block Diagram

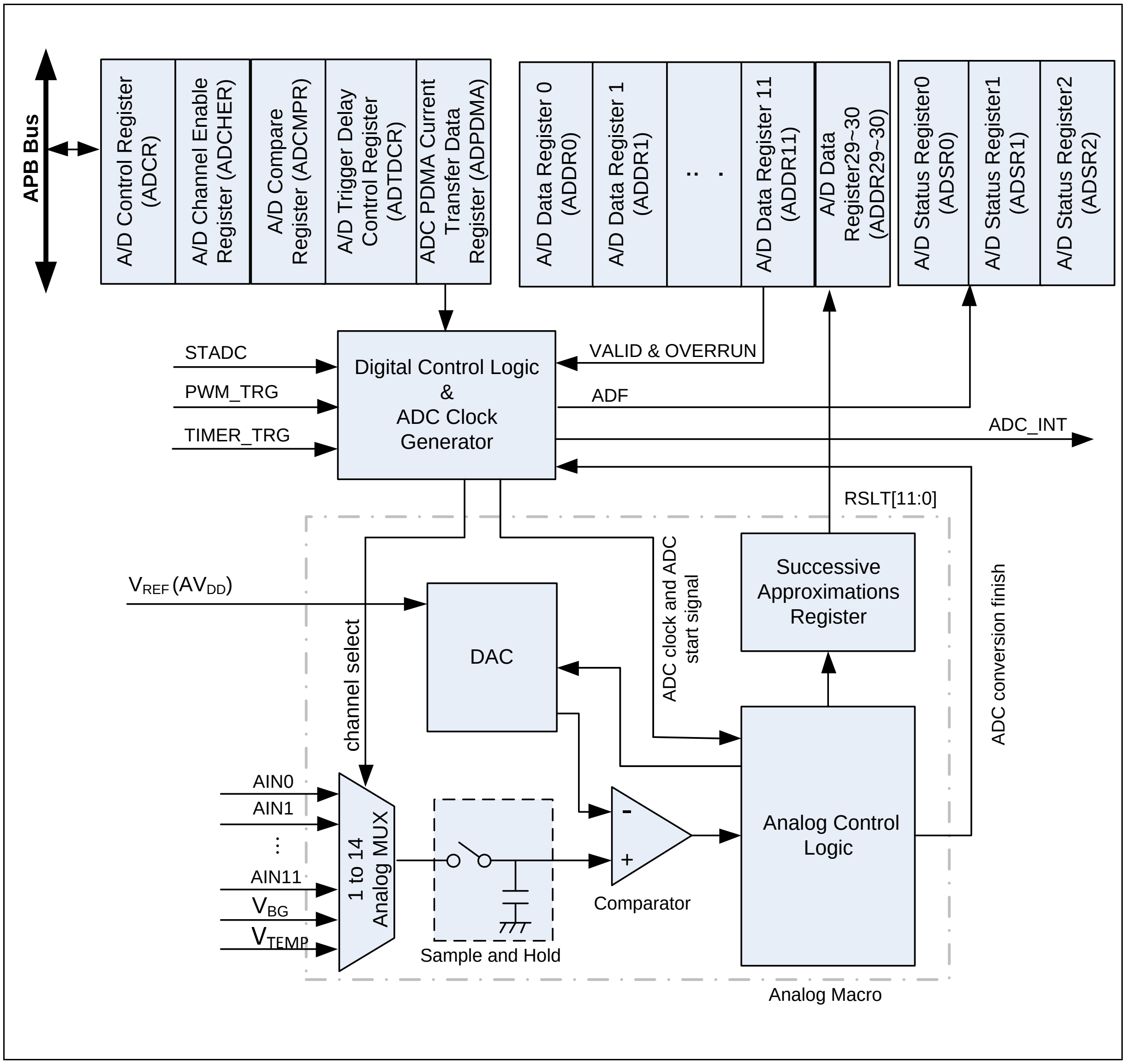

APB Bus
A/D Control Register (ADCR)
A/D Channel Enable Register (ADCHER)
A/D Compare Register (ADCMPR)
A/D Trigger Delay Control Register (ADTDCR)
ADC PDMA Current Transfer Data Register (ADPDMA)
A/D Data Register 0 (ADDR0)
A/D Data Register 1 (ADDR1)
. . .
A/D Data Register 11 (ADDR11)
A/D Data Register29~30 (ADDR29~30)
A/D Status Register0 (ADSR0)
A/D Status Register1 (ADSR1)
A/D Status Register2 (ADSR2)
STADC
PWM_TRG
TIMER_TRG
Digital Control Logic & ADC Clock Generator
VALID & OVERRUN
ADF
ADC_INT
RSLT[11:0]
V_REF (AV_DD)
DAC
channel select
ADC clock and ADC start signal
Successive Approximations Register
ADC conversion finish
AIN0
AIN1
AIN11
V_BG
V_TEMP
1 to 14 Analog MUX
Sample and Hold
-
+
Comparator
Analog Control Logic
Analog Macro

Figure 6.20-1 ADC Controller Block Diagram

6.20.4 Basic Configuration

The ADC pin functions are configured in SYS_GPB_MFPH, SYS_GPC_MFPH, SYS_GPD_MFPL, SYS_GPE_MFPL and SYS_GPF_MFPL registers. It is recommended to disable the digital input path of the analog input pins to avoid the leakage current. User can disable the digital input path by configuring PB_DINOFF, PC_DINOFF, PD_DINOFF, PE_DINOFF and PF_DINOFF registers.

The ADC peripheral clock can be enabled in APBCLK0[28]. The ADC peripheral clock source is selected by CLKSEL1[3:2]. The clock pre-scalar is determined by CLKDIV0[23:16].

6.20.5 Functional Description

The A/D converter operates by successive approximation with 12-bit resolution. User needs to set ADEN (ADC_ADCR[0]) as 1 to enable ADC first. The ADC has four operation modes: Single, Burst, Single-cycle Scan mode and Continuous Scan mode. User may set ADMD (ADC_ADCR[3:2]) to choose which operation mode is used. When user wants to change the operation mode or analog input channel, in order to prevent incorrect operation, software must clear ADST bit to 0 in advance.

6.20.5.1 ADC peripheral Clock Generator

The maximum sampling rate is up to 800k SPS. The ADC has four clock sources selected by ADCSEL (CLKSEL1[3:2]), the ADC peripheral clock frequency is divided by an 8-bit pre-scalar with the following formula:

ADC peripheral clock frequency = (ADC peripheral clock source frequency) / (ADCDIV+1); where the 8-bit ADCDIV is located in register CLKDIV0[23:16].

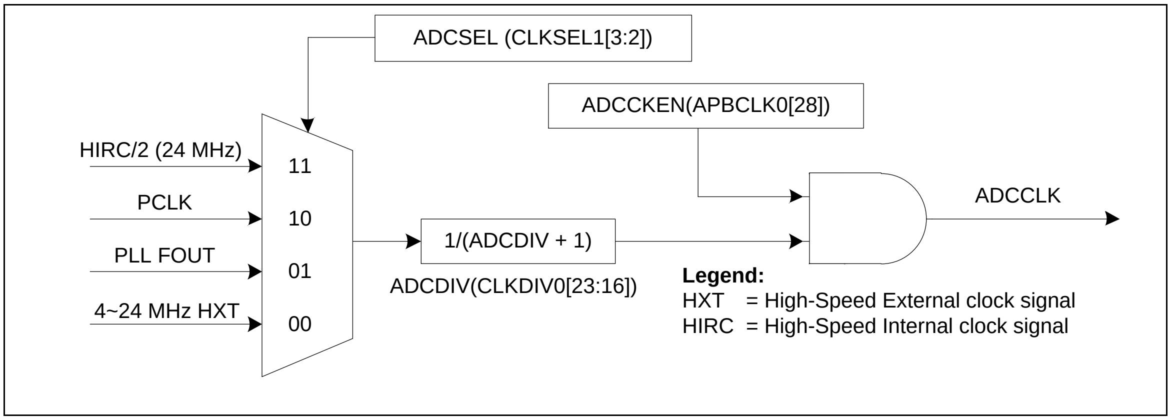


Figure 6.20-2 ADC Peripheral Clock Control

6.20.5.2 Single Mode

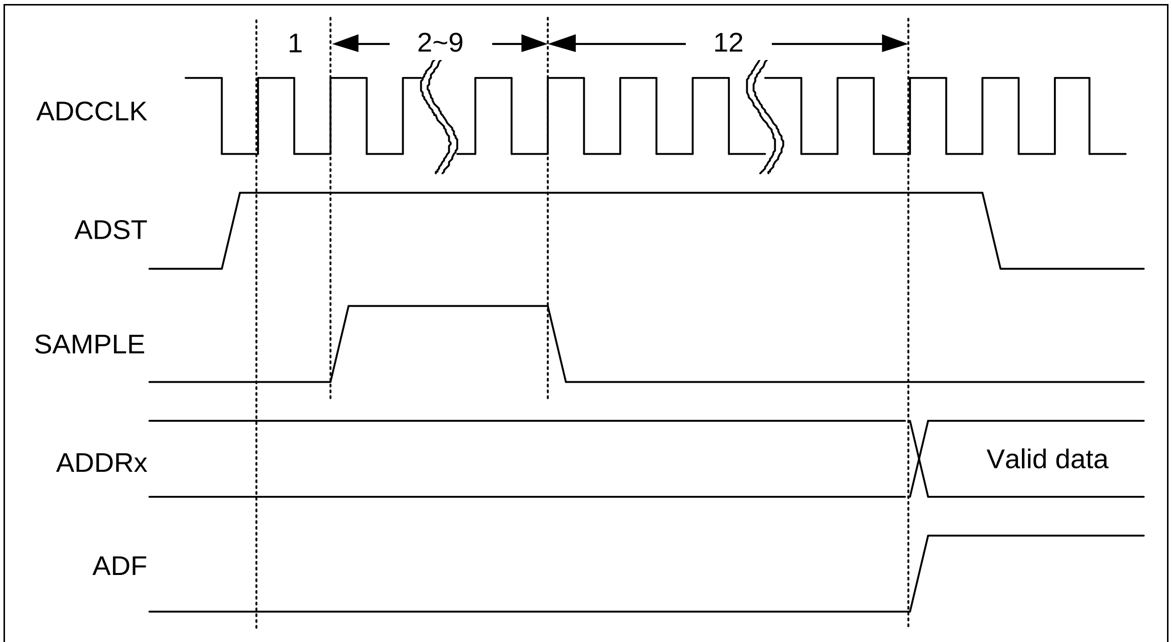
In Single mode, A/D conversion is performed only once on the specified single channel. The operations are as follows:

1. A/D conversion will be started when the ADST bit of ADCR is set to 1 by software or external trigger input.
2. When A/D conversion is finished, the result is stored in the A/D data register corresponding to the channel. For example, the conversion result of AIN1 is stored in RSLT (ADC_ADDR1[11:0]).
3. The ADF bit of ADSR register will be set to 1. If the ADIE bit of ADCR register is set to 1, the ADC interrupt will be asserted.
4. The ADST bit remains 1 during A/D conversion. When A/D conversion ends, the ADST bit is automatically cleared to 0 and the A/D converter enters idle state.

Note 1: If software enables more than one channel in Single mode, only the channel with the smallest number will be selected and the other enabled channels will be ignored.

Note 2: If ADST bit is cleared to 0 when ADC is in converting, the BUSY bit will be cleared to 0 immediately, ADC cannot finish the current conversion and A/D converter enters idle state directly.

An example timing diagram for Single mode is shown below.



Note: SAMPLE is an internal signal indicates sample stage
x = the selected channel

Figure 6.20-3 Single Mode Conversion Timing Diagram

6.20.5.3 Burst Mode

In Burst mode, A/D converter samples and converts the specified single channel and sequentially stores the result into FIFO (up to 16 samples). The operations are as follows:

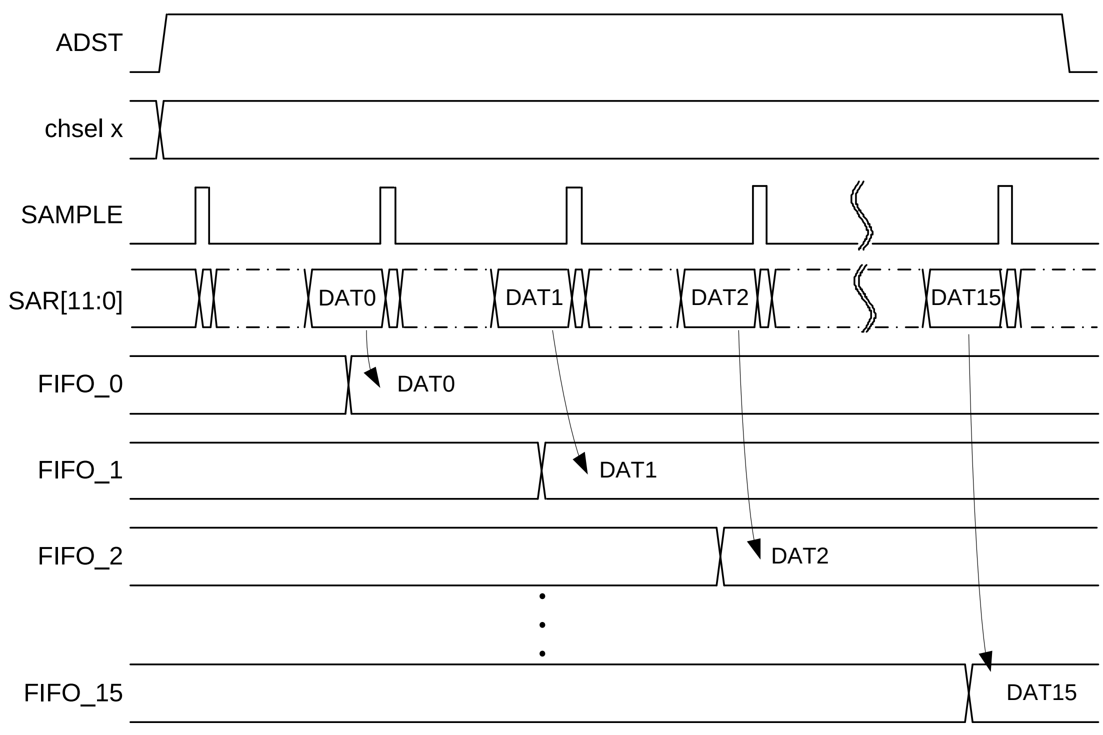
1. When the ADST bit in ADCR is set to 1 by software or external trigger input, A/D conversion is started on the enabled channel with the smallest number.

2. When A/D conversion for the specified channel is completed, the result is sequentially transferred to FIFO and can be accessed only from the A/D data register 0 (ADDR0).

3. When more than 8 samples in FIFO, the ADF bit in ADSR is set to 1. If the ADIE bit is set to 1 at this time, an ADC interrupt is requested after finishing the A/D conversion.

4. Steps 2 to 3 are repeated as long as the ADST bit remains 1. When the ADST bit is cleared to 0, ADC cannot finish the current conversion and A/D converter enters idle state directly

An example timing diagram for Burst mode is shown below.



Note: x = the smallest channel selection
SAMPLE is an internal signal indicates sample stage
SAR[11:0] is ADC analog macro output

Figure 6.20-4 Burst Mode Conversion Timing Diagram

Note 1: If software enables more than one channel in Burst mode, only the channel with the smallest number is converted and other enabled channels will be ignored.

Note 2: User can obtain the conversion results by reading the A/D data register 0 (ADDR0) repeatedly until VALIDF (ADSR[8]) turns to 0. For example, if there are 8 conversion results at FIFO, it needs to read A/D data register 0 (ADDR0) 8 times to get all of result.

6.20.5.4 Single-Cycle Scan Mode

In Single-cycle Scan mode, A/D converter samples and converts all of the specified channels once in the sequence from the smallest number enabled channel to the largest number enabled channel. Operations are as follows:

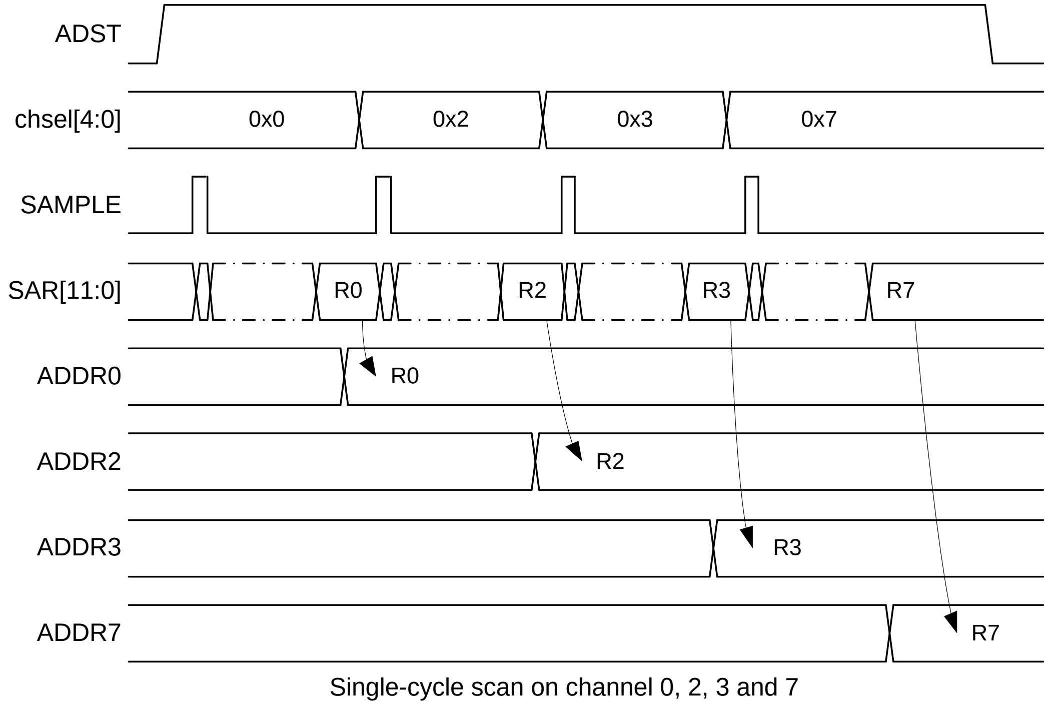
1. When the ADST bit in ADCR is set to 1 by software or external trigger input, A/D conversion is started on the enabled channel with the smallest number.

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

3. When the conversions of all the enabled channels are completed, the ADF bit in ADSR is set to 1. If the ADC interrupt function is enabled, the ADC interrupt occurs.

4. After ADC finishes one cycle conversion, the ADST bit is automatically cleared to 0 and the A/D converter enters idle state. If ADST is cleared to 0 before all enabled ADC channels conversion done, ADC cannot finish the current conversion and A/D converter enters idle state directly.

An example timing diagram for Single-cycle Scan mode on enabled channels (0, 2, 3 and 7) is shown below.



Single-cycle scan on channel 0, 2, 3 and 7

Note: SAMPLE is an internal signal indicates sample stage
SAR[11:0] is ADC analog macro output

Figure 6.20-5 Single-Cycle Scan Mode on Enabled Channels Timing Diagram

6.20.5.5 *Continuous Scan Mode*

In Continuous Scan mode, A/D conversion is performed sequentially on the specified channels that enabled by CHEN bits in ADCHER register. The operations are as follows:

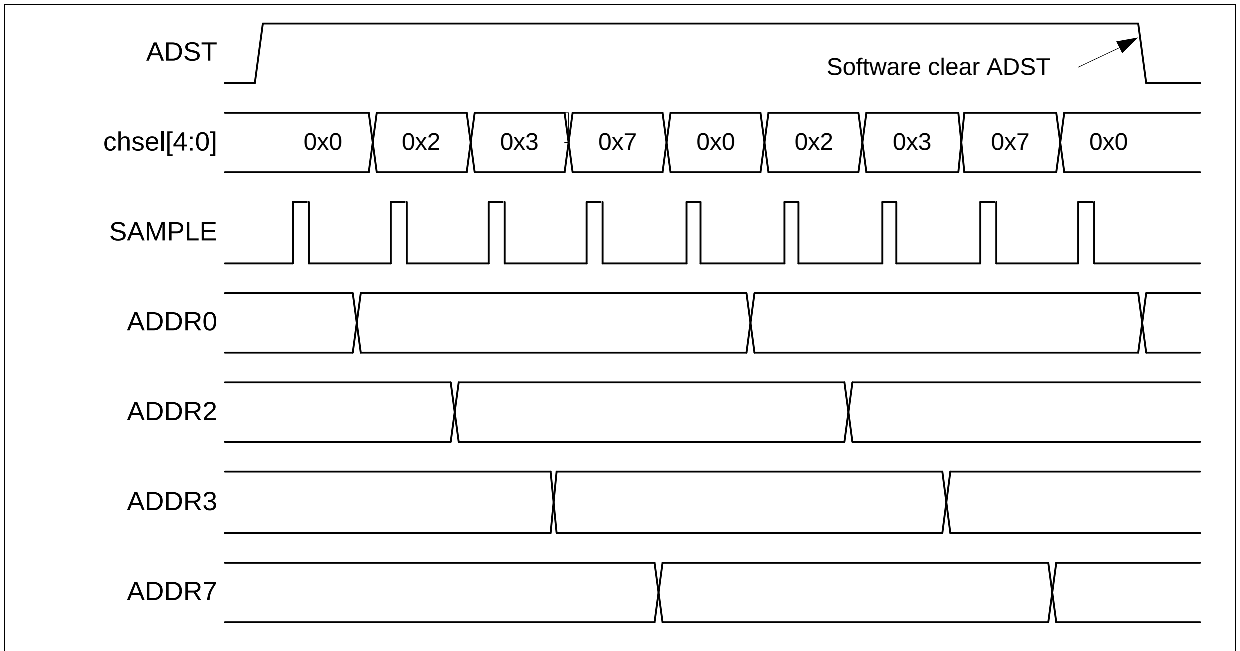
1. When the ADST bit in ADCR is set to 1 by software or external trigger input, A/D conversion is started on the enabled channel with the smallest number.

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

3. When A/D converter completes the conversions of all enabled channels sequentially, the ADF bit (ADSR0[0]) will be set to 1. If the ADC interrupt function is enabled, the ADC interrupt occurs. The conversion of the enabled channel with the smallest number will start again if software does not clear the ADST bit.

4. As long as the ADST bit remains at 1, the step 2 ~ 3 will be repeated. When ADST is cleared to 0, ADC cannot finish the current conversion and A/D converter enters idle state directly.

An example timing diagram for Continuous Scan mode on enabled channels (0, 2, 3 and 7) is shown below.



Continuous scan on channel 0, 2, 3 and 7

Note: SAMPLE is an internal signal indicates sample stage

Figure 6.20-6 Continuous Scan Mode on Enabled Channels Timing Diagram

6.20.5.6 *Differential Input Mode*

User may set DIFFEN (ADC_ADCR[10]) as 1 to enable differential input mode. There are 6 pairs differential input as Table 6.20-1. The differential input voltage $V_{diff} = V_{plus} - V_{minus}$. In differential input mode, only the even number of the two corresponding channels needs to be enabled in ADCHER register. The conversion result will be placed to the corresponding RSLT (ADC_ADDRx[15:0]) of the enabled channel. The transfer function between differential input voltage and conversion result is as follows :

$$V = \frac{[V_{ref} - LSB - (-V_{ref})] * \mathcal{X}}{4096} - V_{ref} = (\frac{8191}{4096^2} \cdot \mathcal{X} - 1) * V_{ref}$$

$$\mathcal{X}\ is\ the\ decimal\ number\ of\ ADC\_ADDRx[15:0], \qquad \mathcal{X}\ \in\ [0, 4095]$$

$$V = \frac{[V_{ref} - LSB - (-V_{ref})] * (x - (-2048))}{4096} - V_{ref} = [\frac{8191}{4096^2} * (x + 2048) - 1] * V_{ref}$$

$$\mathcal{X}\ is\ the\ decimal\ number\ of\ ADC\_ADDRx[15:0], \qquad \mathcal{X}\ \in\ [-2048, 2047]$$

For example, if the conversion result of 2's complement is 0xF801 and Vref = 3.3V, the value of hex will be 0x7FF and the decimal number will be -2047. Therefore the differential input voltage will be Vin = { [8191 * (-2047+2048) / 4096^2] – 1 } * 3.3 = - 3.298 V

Differential Input Paired Channel	**ADC Analog Input**	
	V_{plus}	V_{minus}
0	AIN0	AIN1
1	AIN2	AIN3
2	AIN4	AIN5
3	AIN6	AIN7
4	AIN8	AIN9
5	AIN10	AIN11

Table 6.20-1 Differential Input Pairs

6.20.5.7 *External Trigger Input*

In Single-cycle Scan mode, A/D conversion can be triggered by external pin request. When the TRGEN (ADCR[8]) bit of ADCR register is set to 1 to enable ADC external trigger function, setting the TRGS (ADCR[5:4]) bits to 2'b00 is to select external trigger input from the STADC pin. Software can set TRGCOND (ADCR[7:6]) to select trigger condition is falling/rising edge or low/high level. If level trigger condition is selected, the STADC pin must be kept at specified state at least 8 PCLKs. The ADST bit will be set to 1 at the 9th PCLK and start to convert. Conversion will keep going if external trigger input is kept at active state in level trigger mode. It is stopped only when external condition trigger condition disappears. If edge trigger condition is selected, the high and low state must be kept at least 4 PLCKs. Pulse that is shorter than this specification will be ignored.

Note: User enables the external trigger function or enables ADC must be at least 4 PCLKs after enabling ADC peripheral clock.

6.20.5.8 *Timer Trigger*

There are 4 Timer trigger sources for ADC. When the TRGEN (ADCR[8]) is set to high to enable ADC external hardware trigger function, setting the TRGS (ADCR[5:4]) bits to 2'b01 is to select external hardware trigger input source from Timer trigger. The detail trigger conditions of Timer are described at TIMER0_CTL ~ TIMER3_CTL register.

6.20.5.9 PWM Trigger

In Single-cycle Scan mode, A/D conversion can be triggered by PWM request. When the TRGEN (ADCR[8]) is set to high to enable ADC external hardware trigger function, setting the TRGS (ADCR[5:4]) bits to 2'b11 is to select external hardware trigger input source from PWM trigger. When PWM trigger is enabled, setting PTDT (ADTDCR[7:0]) bits can insert a delay time between PWM trigger condition and ADC start conversion. The detail trigger conditions of PWM are described at the section of BPWM.

6.20.5.10 Conversion Result Monitor by Compare Mode Function

The ADC controller provides two compare registers, ADCMPR0 and ADCMPR1, to monitor maximum two specified channels. User may set CMPEN (ADC_ADCMPRx[0]) as 1 to enable compare mode function. Software can select which channel to be monitored by setting CMPCH (ADCMPRx[7:3]). CMPCOND (ADCMPRx[2]) bit is used to determine the compare condition. If CMPCOND bit is cleared to 0, the internal match counter will increase one when the conversion result is less than the value specified in CMPD (ADCMPRx[27:16]); 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 (ADCMPRx[27:16]). 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 the match counter reaches the setting of (CMPMATCNT+1) then CMPF0/1 (ADSR0[1]/[2]) bit will be set to 1, if CMPIE (ADCMPRx[1]) bit is set then 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. The detailed logic diagram is shown below.

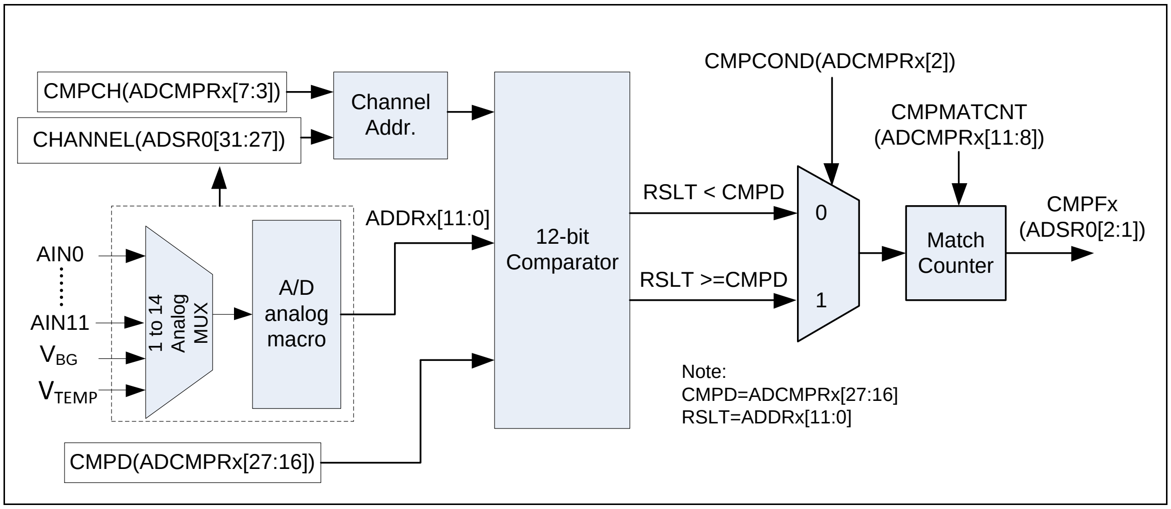


Figure 6.20-7 A/D Conversion Result Monitor Logic Diagram

6.20.5.11 Compare Window Mode

The ADC controller supports a compare window mode. User can set CMPWEN (ADCMPR0[15]) to enable this function. If user enables this function, CMPF0 (ADSR0[1]) will be set only when compared conditions of two conversion result monitor logic are matched. But CMPF1 (ADSR0[2]) will keep 0. The range of compare window is between CMPD0 (ADCMPR0[27:16]) and CMPD1

(ADCMPR1[27:16]).

6.20.5.12 PDMA Transfer Mode

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 (ADCR[9]) 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 sample or next selected channel if the operation mode of ADC is burst mode, 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 (ADDRx[16]) bit of the corresponding channel will be set and the last ADC conversion result will be overwritten by the new ADC conversion result. If any one of OVERRUN (ADDRx[16]) is set, OVERRUNF (ADC_ADSR0[16]) will be set to 1. PDMA will transfer the latest data of selected channels to the user-specified destination address. User may observe the data which is under transfering by reading CURDAT (ADC_ADPDMA[17:0]) simultaneously.

6.20.5.13 Interrupt Sources

There are three interrupt sources of ADC interrupt. When an ADC operation mode finishes its conversion, the A/D conversion end flag, ADF, will be set to 1. The CMPF0 and CMPF1 are the compare flags of compare function. When the conversion result meets the settings of ADCMPR0/1, the corresponding flag will be set to 1. When one of the flags, ADF, CMPF0 and CMPF1, is set to 1 and the corresponding interrupt enable bit, ADIE of ADCR and CMPIE of ADCMPR0/1, is set to 1, the ADC interrupt will be asserted. Software can clear these flags to revoke the interrupt request.

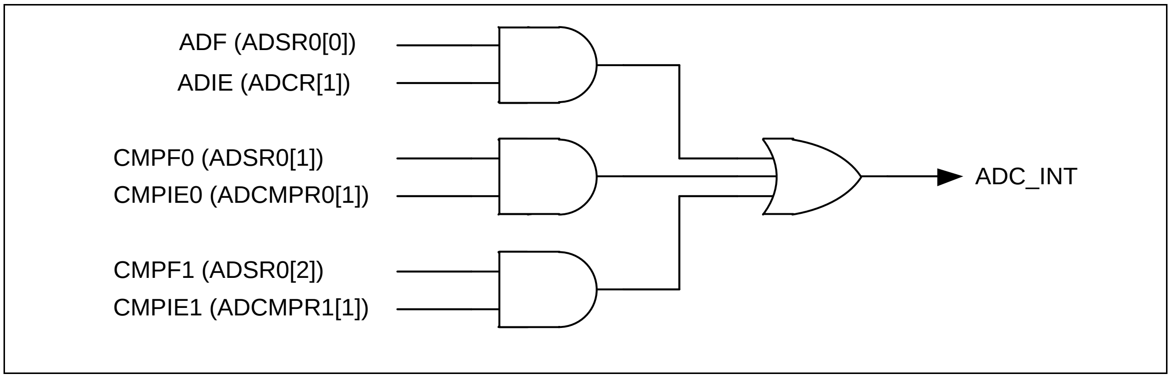


Figure 6.20-8 A/D Controller Interrupt

6.20.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
ADC Base Address: **ADC_BA = 0x400E_0000**				
ADC_ADDR0	ADC_BA+0x00	R	ADC Data Register 0	0x0000_0000
ADC_ADDR1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000
ADC_ADDR2	ADC_BA+0x08	R	ADC Data Register 2	0x0000_0000
ADC_ADDR3	ADC_BA+0x0C	R	ADC Data Register 3	0x0000_0000
ADC_ADDR4	ADC_BA+0x10	R	ADC Data Register 4	0x0000_0000
ADC_ADDR5	ADC_BA+0x14	R	ADC Data Register 5	0x0000_0000
ADC_ADDR6	ADC_BA+0x18	R	ADC Data Register 6	0x0000_0000
ADC_ADDR7	ADC_BA+0x1C	R	ADC Data Register 7	0x0000_0000
ADC_ADDR8	ADC_BA+0x20	R	ADC Data Register 8	0x0000_0000
ADC_ADDR9	ADC_BA+0x24	R	ADC Data Register 9	0x0000_0000
ADC_ADDR10	ADC_BA+0x28	R	ADC Data Register 10	0x0000_0000
ADC_ADDR11	ADC_BA+0x2C	R	ADC Data Register 11	0x0000_0000
ADC_ADDR29	ADC_BA+0x74	R	ADC Data Register 29	0x0000_0000
ADC_ADDR30	ADC_BA+0x78	R	ADC Data Register 30	0x0000_0000
ADC_ADCR	ADC_BA+0x80	R/W	ADC Control Register	0x0005_0000
ADC_ADCHER	ADC_BA+0x84	R/W	ADC Channel Enable Register	0x0000_0000
ADC_ADCMPR0	ADC_BA+0x88	R/W	ADC Compare Register 0	0x0000_0000
ADC_ADCMPR1	ADC_BA+0x8C	R/W	ADC Compare Register 1	0x0000_0000
ADC_ADSR0	ADC_BA+0x90	R/W	ADC Status Register0	0x0000_0000
ADC_ADSR1	ADC_BA+0x94	R	ADC Status Register1	0x0000_0000
ADC_ADSR2	ADC_BA+0x98	R	ADC Status Register2	0x0000_0000
ADC_ADTDCR	ADC_BA+0x9C	R/W	ADC Trigger Delay Control Register	0x0000_0000
ADC_ADPDMA	ADC_BA+0x100	R	ADC PDMA Current Transfer Data Register	0x0000_0000

6.20.7 Register Description

ADC Data Registers (ADC_ADDRx x = 0~11, 29~30)

Register	Offset	R/W	Description	Reset Value
ADC_ADDR0	ADC_BA+0x00	R	ADC Data Register 0	0x0000_0000
ADC_ADDR1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000
ADC_ADDR2	ADC_BA+0x08	R	ADC Data Register 2	0x0000_0000
ADC_ADDR3	ADC_BA+0x0C	R	ADC Data Register 3	0x0000_0000
ADC_ADDR4	ADC_BA+0x10	R	ADC Data Register 4	0x0000_0000
ADC_ADDR5	ADC_BA+0x14	R	ADC Data Register 5	0x0000_0000
ADC_ADDR6	ADC_BA+0x18	R	ADC Data Register 6	0x0000_0000
ADC_ADDR7	ADC_BA+0x1C	R	ADC Data Register 7	0x0000_0000
ADC_ADDR8	ADC_BA+0x20	R	ADC Data Register 8	0x0000_0000
ADC_ADDR9	ADC_BA+0x24	R	ADC Data Register 9	0x0000_0000
ADC_ADDR10	ADC_BA+0x28	R	ADC Data Register 10	0x0000_0000
ADC_ADDR11	ADC_BA+0x2C	R	ADC Data Register 11	0x0000_0000
ADC_ADDR29	ADC_BA+0x74	R	ADC Data Register 29	0x0000_0000
ADC_ADDR30	ADC_BA+0x78	R	ADC Data Register 30	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**VALID**	**Valid Flag** This bit will be set to 1 when the conversion of the corresponding channel is completed. This bit will be cleared to 0 by hardware after ADDR register is read 0 = Data in RSLT bits is not valid. 1 = Data in RSLT bits is valid.
[16]	**OVERRUN**	**Overrun Flag** If converted data in RSLT has not been read before new conversion result is loaded to this register, OVERRUN is set to 1. It is cleared by hardware after ADDR register is read 0 = Data in RSLT is not overwrote. 1 = Data in RSLT is overwrote..
[15:0]	**RSLT**	**A/D Conversion Result** This field contains conversion result of ADC.

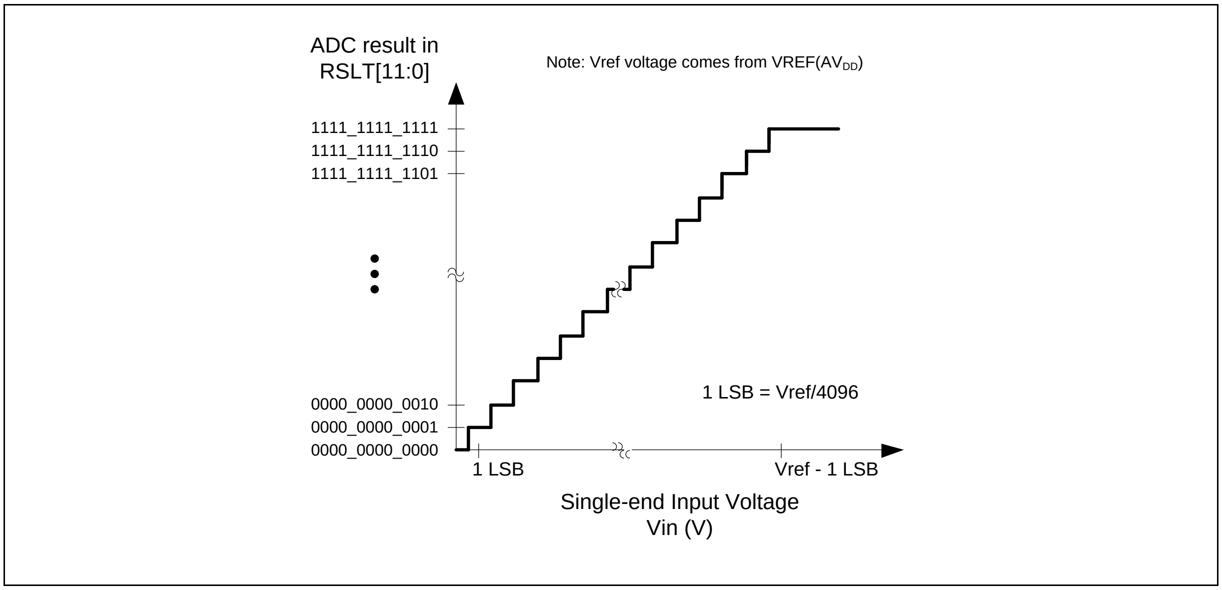


Figure 6.20-9 Conversion Result Mapping Diagram of ADC Single-end Input

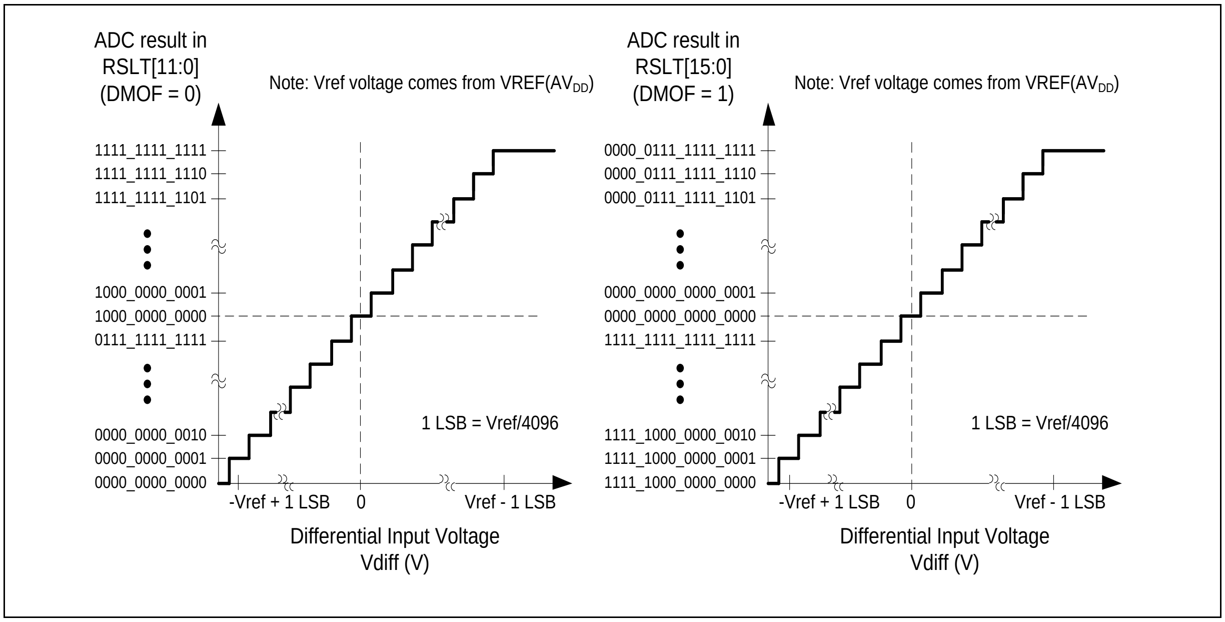


Figure 6.20-10 Conversion Result Mapping Diagram of ADC Differential Input

ADC Control Register (ADC_ADCR)

Register	Offset	R/W	Description	Reset Value
ADC_ADCR	ADC_BA+0x80	R/W	ADC Control Register	0x0005_0000

31	30	29	28	27	26	25	24
DMOF	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**SMPTSEL**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ADST**	**DIFFEN**	**PTEN**	**TRGEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRGCOND		**TRGS**		**ADMD**		**ADIE**	**ADEN**

Bits	Description	
[31]	**DMOF**	**Differential Input Mode Output Format** If user enables differential input mode, the conversion result can be expressed with binary straight format (unsigned format) or 2's complement format (signed format). 0 = A/D Conversion result will be filled in RSLT at ADDRx registers with unsigned format (straight binary format). 1 = A/D Conversion result will be filled in RSLT at ADDRx registers with 2's complement format.
[30:19]	**Reserved**	Reserved.
[18:16]	**SMPTSEL**	**ADC Internal Sampling Time Selection** Total ADC conversion cycle = sampling cycle + 12. 000 = 2 ADC clock for sampling ; 14 ADC clock for complete conversion. 001 = 3 ADC clock for sampling ; 15 ADC clock for complete conversion. 010 = 4 ADC clock for sampling ; 16 ADC clock for complete conversion. 011 = 5 ADC clock for sampling ; 17 ADC clock for complete conversion. 100 = 6 ADC clock for sampling ; 18 ADC clock for complete conversion. 101 = 7 ADC clock for sampling ; 19 ADC clock for complete conversion. 110 = 8 ADC clock for sampling ; 20 ADC clock for complete conversion. 111 = 9 ADC clock for sampling ; 21 ADC clock for complete conversion.
[15:12]	**Reserved**	Reserved.
[11]	**ADST**	**A/D Conversion Start** ADST bit can be set to 1 from four sources: software, external pin STADC, PWM trigger and Timer trigger. ADST will be cleared to 0 by hardware automatically at the ends of Single mode and Single-cycle Scan mode. In Continuous Scan mode and Burst mode, A/D conversion is continuously performed until software writes 0 to this bit or chip is reset. 0 = Conversion stops and A/D converter enters idle state. 1 = Conversion starts.
[10]	**DIFFEN**	**Differential Input Mode Control** Differential input voltage (V_{diff}) = V_{plus} - V_{minus},.

		where V_{plus} is the analog input; V_{minus} is the inverted analog input. The V_{plus} of differential input paired channel x is from ADC0_CHy pin; V_{minus} is from ADC0_CHz pin, x=0,1..5, y=2*x, z=y+1. 0 = Single-end analog input mode. 1 = Differential analog input mode. **Note:** In Differential Input mode, only the even number of the two corresponding channels needs to be enabled in ADCHER register. The conversion result will be placed to the corresponding data register of the enabled channel.
[9]	**PTEN**	**PDMA Transfer Enable Bit** When A/D conversion is completed, the converted data is loaded into ADDR0~11, ADDR29~ADDR30. Software can enable this bit to generate a PDMA data transfer request. 0 = PDMA data transfer Disabled. 1 = PDMA data transfer in ADDR0~11, ADDR29~ADDR30 Enabled. **Note:** When PTEN=1, software must set ADIE=0 to disable interrupt.
[8]	**TRGEN**	**External Trigger Enable Control** Enable or disable triggering of A/D conversion by external STADC pin, PWM trigger and Timer trigger. If external trigger is enabled, the ADST bit can be set to 1 by the selected hardware trigger source. 0= External trigger Disabled. 1= External trigger Enabled. **Note:** The ADC external trigger function is only supported in Single-cycle Scan mode.
[7:6]	**TRGCOND**	**External Trigger Condition** These two bits decide external pin STADC trigger event is level or edge. The signal must be kept at stable state at least 8 PCLKs for level trigger and at least 4 PCLKs for edge trigger. 00 = Low level. 01 = High level. 10 = Falling edge. 11 = Rising edge.
[5:4]	**TRGS**	**Hardware Trigger Source** 00 = A/D conversion is started by external STADC pin. 01 = Timer0 ~ Timer3 overflow pulse trigger. 10 = Reserved. 11 = A/D conversion is started by PWM trigger. **Note:** Software should clear TRGEN bit and ADST bit to 0 before changing TRGS.
[3:2]	**ADMD**	**A/D Converter Operation Mode Control** 00 = Single conversion. 01 = Burst conversion. 10 = Single-cycle Scan. 11 = Continuous Scan. **Note1:** When changing the operation mode, software should clear ADST bit first. **Note2:** In Burst mode, the A/D result data always at Data Register 0.
[1]	**ADIE**	**A/D Interrupt Enable Control** A/D conversion end interrupt request is generated if ADIE bit is set to 1. 0 = A/D interrupt function Disabled. 1 = A/D interrupt function Enabled.
[0]	**ADEN**	**A/D Converter Enable Bit**

		0 = A/D converter Disabled. 1 = A/D converter Enabled. **Note:** Before starting A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit to save power consumption.

ADC Channel Enable Register (ADC_ADCHER)

Register	Offset	R/W	Description	Reset Value
ADC_ADCHER	ADC_BA+0x84	R/W	ADC Channel Enable Register	0x0000_0000

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

Bits	Description	
[31:0]	CHEN	**Analog Input Channel Enable Control** Set CHEN[11:0] to enable the corresponding analog input channel 11 ~ 0. If DIFFEN bit is set to 1, only the even number channel needs to be enabled. Besides, set CHEN[29] to CHEN[30] will enable internal channel for band-gap voltage and temperature sensor respectively. Other bits are reserved. 0 = Channel Disabled. 1 = Channel Enabled. **Note 1 :** If the internal channel for band-gap voltage (CHEN[29]) is active, the maximum sampling rate will be TBD SPS **Note 2 :** If the internal channel for temperature sensor (CHEN[30]) is active, the maximum sampling rate will be 300k SPS

A/D Compare Register 0/1 (ADC_ADCMPR0/1)

Register	Offset	R/W	Description	Reset Value
ADC_ADCMPR0	ADC_BA+0x88	R/W	ADC Compare Register 0	0x0000_0000
ADC_ADCMPR1	ADC_BA+0x8C	R/W	ADC Compare Register 1	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**CMPD**	**Comparison Data** The 12-bit data is used to compare with conversion result of specified channel. **Note:** CMPD should be filled in unsigned format (straight binary format).
[15]	**CMPWEN**	**Compare Window Mode Enable Bit** 0 = Compare Window Mode Disabled. 1 = Compare Window Mode Enabled. **Note:** This bit is only presented in ADCMPR0 register.
[14:12]	**Reserved**	Reserved.
[11:8]	**CMPMATCNT**	**Compare Match Count** When the specified A/D channel analog conversion result matches the compare condition defined by CMPCOND, the internal match counter will increase 1, otherwise, the internal match counter will be cleared to 0. When the internal counter reaches the value to (CMPMATCNT +1), the CMPFx bit will be set.
[7:3]	**CMPCH**	**Compare Channel Selection** 00000 = Channel 0 conversion result is selected to be compared. 00001 = Channel 1 conversion result is selected to be compared. 00010 = Channel 2 conversion result is selected to be compared. 00011 = Channel 3 conversion result is selected to be compared. 00100 = Channel 4 conversion result is selected to be compared. 00101 = Channel 5 conversion result is selected to be compared. 00110 = Channel 6 conversion result is selected to be compared. 00111 = Channel 7 conversion result is selected to be compared. 01000 = Channel 8 conversion result is selected to be compared. 01001 = Channel 9 conversion result is selected to be compared.

		01010 = Channel 10 conversion result is selected to be compared. 01011 = Channel 11 conversion result is selected to be compared. 11101 = Band-gap voltage conversion result is selected to be compared. 11110 = Temperature sensor conversion result is selected to be compared. Others = Reserved.
[2]	**CMPCOND**	**Compare Condition** 0= Set the compare condition as that when a 12-bit A/D conversion result is less than the 12-bit CMPD (ADCMPRx[27:16]), the internal match counter will increase one. 1= Set the compare condition as that when a 12-bit A/D conversion result is greater than or equal to the 12-bit CMPD (ADCMPRx[27:16]), the internal match counter will increase one. **Note:** When the compare result meets the condition setting, the internal match counter will increase 1, otherwise, the internal match counter will be cleared to 0**.** **Note:** When the internal counter reaches to (CMPMATCNT +1), the CMPFx bit will be set.
[1]	**CMPIE**	**Compare Interrupt Enable Control Bit** If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMATCNT, CMPFx bit will be asserted, in the meanwhile, if CMPIE is set to 1, a compare interrupt request is generated. 0 = Compare function interrupt Disabled. 1 = Compare function interrupt Enabled.
[0]	**CMPEN**	**Compare Enable Control Bit** Set this bit to 1 to enable ADC controller to compare CMPD[11:0] with specified channel conversion result when converted data is loaded into ADDR register. 0 = Compare function Disabled. 1 = Compare function Enabled.

A/D Status Register0 (ADC_ADSR0)

Register	Offset	R/W	Description	Reset Value
ADC_ADSR0	ADC_BA+0x90	R/W	ADC Status Register0	0x0000_0000

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

Bits	Description	
[31:27]	**CHANNEL**	**Current Conversion Channel** When BUSY=1, this filed reflects current conversion channel. When BUSY=0, it shows the number of the next converted channel. It is read only.
[26:17]	**Reserved**	Reserved.
[16]	**OVERRUNF**	**Overrun Flag** If any one of OVERRUN (ADDRx[16]) is set, this flag will be set to 1. This is a read only bit. **Note:** When ADC is in burst mode and the FIFO is overrun, this flag will be set to 1.
[15:9]	**Reserved**	Reserved.
[8]	**VALIDF**	**Data Valid Flag** If any one of VALID (ADDRx[17]) is set, this flag will be set to 1. This is a read only bit. **Note:** When ADC is in burst mode and any conversion result is valid, this flag will be set to 1
[7]	**BUSY**	**BUSY/IDLE** This bit is a mirror of ADST bit in ADCR register. It is read only. 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion.
[6:3]	**Reserved**	Reserved.
[2]	**CMPF1**	**Compare Flag 1** When the A/D conversion result of the selected channel meets setting condition in ADCMPR1 then this bit is set to 1; it is cleared by writing 1 to it 0 = Conversion result in ADDR does not meet ADCMPR1 setting. 1 = Conversion result in ADDR meets ADCMPR1 setting.
[1]	**CMPF0**	**Compare Flag 0** When the A/D conversion result of the selected channel meets setting condition in ADCMPR0 then this bit is set to 1. This bit is cleared by writing 1 to it.

		0 = Conversion result in ADDR does not meet ADCMPR0 setting. 1 = Conversion result in ADDR meets ADCMPR0 setting.
[0]	**ADF**	**A/D Conversion End Flag** A status flag that indicates the end of A/D conversion. Software can write 1 to clear this bit. ADF is set to 1 at the following three conditions: 1. When A/D conversion ends in Single mode. 2. When A/D conversion ends on all specified channels in Single-cycle Scan mode and Continuous Scan mode. 3. When more than 8 samples in FIFO in Burst mode.

A/D Status Register1 (ADC_ADSR1)

Register	Offset	R/W	Description	Reset Value
ADC_ADSR1	ADC_BA+0x94	R	ADC Status Register1	0x0000_0000

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

Bits	Description	
[31:0]	**VALID**	**Data Valid Flag** VALID[30:29, 11:0] are the mirror of the VALID bits in ADDR30[17] ~ ADDR29[17], ADDR11[17]~ ADDR0[17]. The other bits are reserved. **Note:** When ADC is in burst mode and any conversion result is valid, VALID[30:29, 11:0] will be set to 1.

A/D Status Register2 (ADC_ADSR2)

Register	Offset	R/W	Description	Reset Value
ADC_ADSR2	ADC_BA+0x98	R	ADC Status Register2	0x0000_0000

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

Bits	Description	
[31:0]	**OVERRUN**	**Overrun Flag** OVERRUN[30:29, 11:0] are the mirror of the OVERRUN bit in ADDR30[16] ~ADDR29[16], ADDR11[16] ~ ADDR0[16]. The other bits are reserved. **Note:** When ADC is in burst mode and the FIFO is overrun, OVERRUN[30:29, 11:0] will be set to 1.

ADC Trigger Delay Control Register (ADC_ADTDCR)

Register	Offset	R/W	Description	Reset Value
ADC_ADTDCR	ADC_BA+0x9C	R/W	ADC Trigger Delay Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**PTDT**	**PWM Trigger Delay Time** Set this field will delay ADC start conversion time after PWM trigger. PWM trigger delay time is (4 * PTDT) * system clock

ADC PDMA Current Transfer Data Register (ADC_ADPDMA)

Register	Offset	R/W	Description	Reset Value
ADC_ADPDMA	ADC_BA+0x100	R	ADC PDMA Current Transfer Data Register	0x0000_0000

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

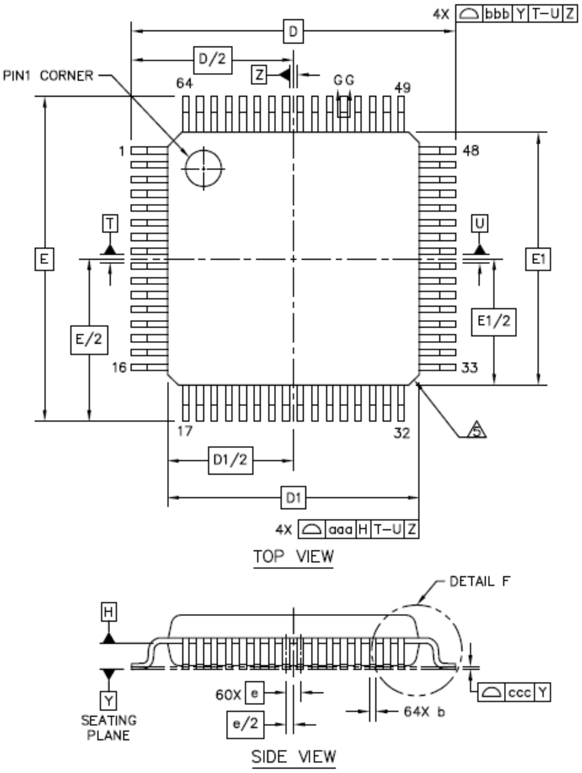
Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:0]	**CURDAT**	**ADC PDMA Current Transfer Data Register** When PDMA transferring, read this register can monitor current PDMA transfer data. Current PDMA transfer data could be the content of ADDR0 ~ ADDR11 and ADDR29 ~ ADDR30. This is a read only register.

7 ELECTRICAL CHARACTERISTICS

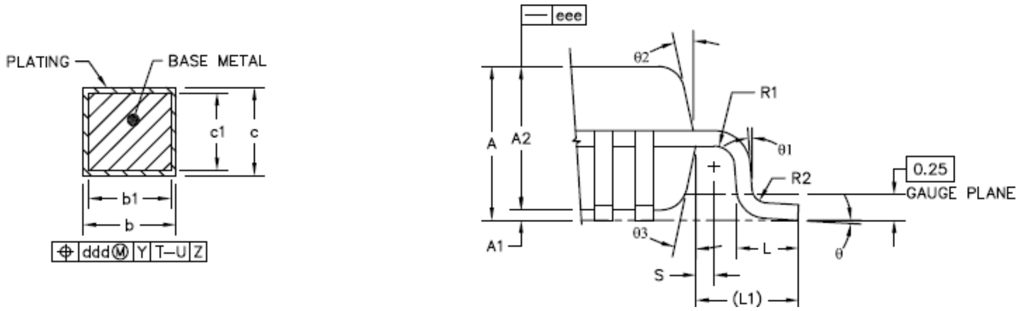
For information on the NUC121/125 series electrical characteristics, please refer to NuMicro® NUC121/125 Series Datasheet.

8 PACKAGE DIMENSIONS

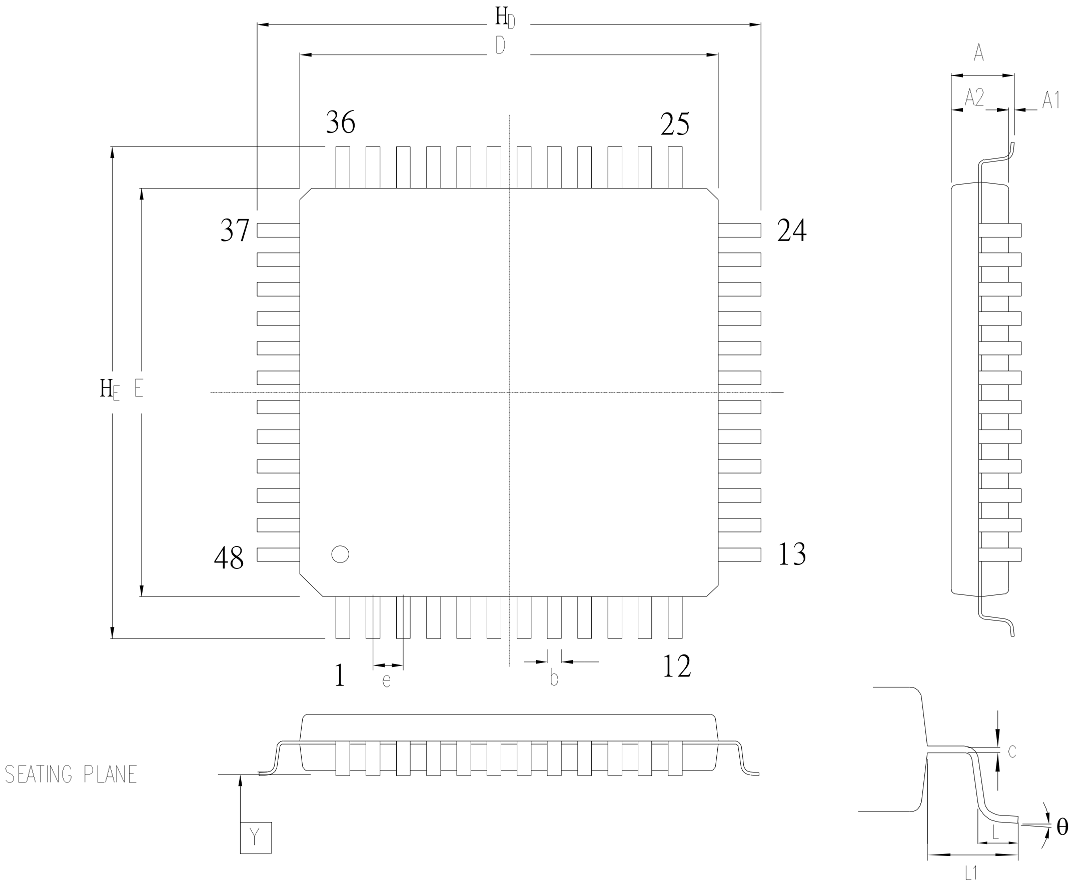
8.1 LQFP 64L (7x7x1.4 mm)



		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
	X	D		9 BSC	
	Y	E		9 BSC	
BODY SIZE	X	D1		7 BSC	
	Y	E1		7 BSC	
LEAD PITCH		e		0.4 BSC	
		L	0.45	0.6	0.75
FOOTPRINT		L1		1 REF	
		θ	0°	3.5°	7°
		θ1	0°	---	---
		θ2	11°	12°	13°
		θ3	11°	12°	13°
		R1	0.08	---	---
		R2	0.08	---	0.2
		S	0.2	---	---
PACKAGE EDGE TOLERANCE		aaa		0.2	
LEAD EDGE TOLERANCE		bbb		0.2	
COPLANARITY		ccc		0.08	
LEAD OFFSET		ddd		0.07	
MOLD FLATNESS		eee		0.05	



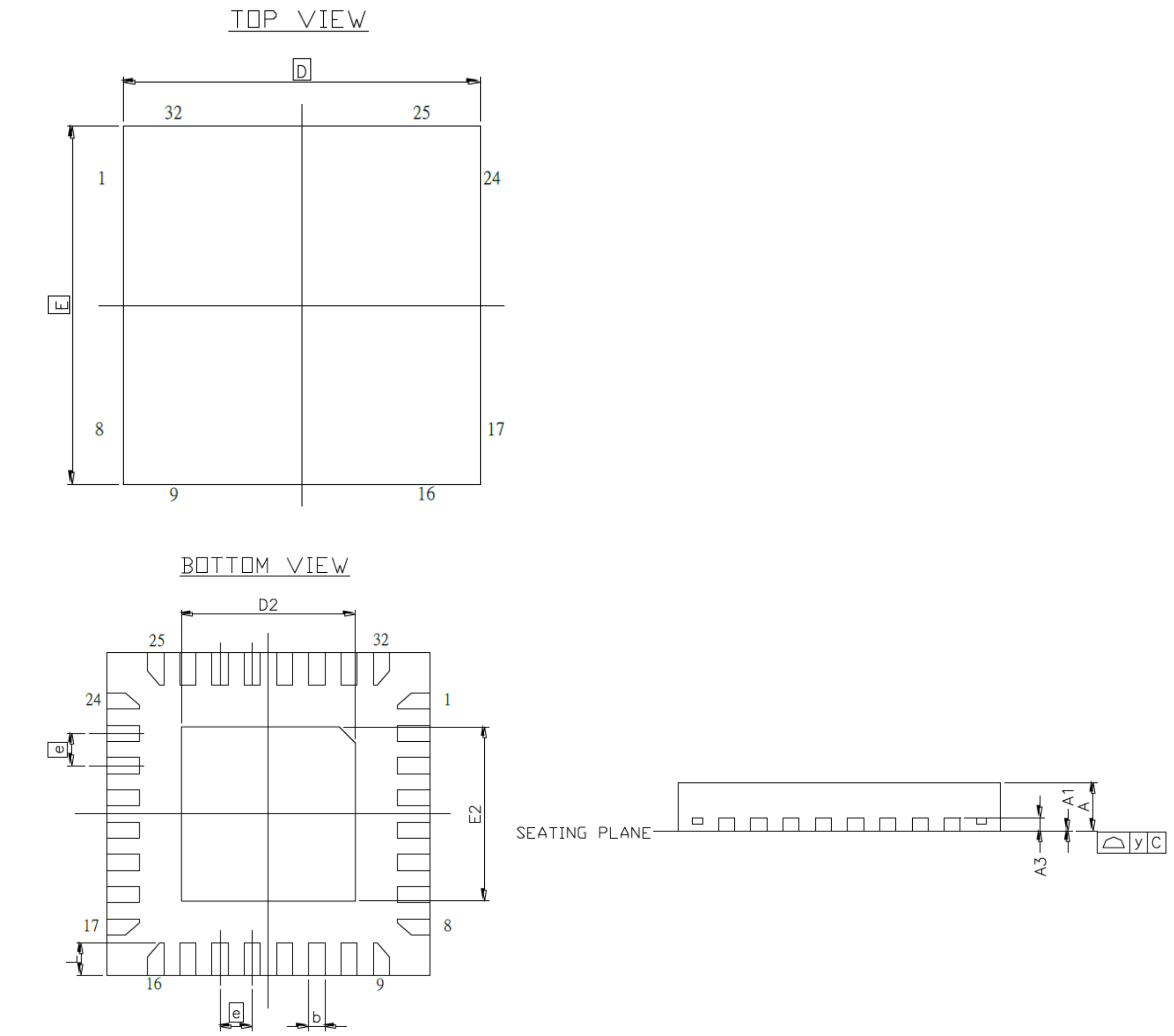
8.2 LQFP 48L (7x7x1.4 mm)



Controlling dimension : Millimeters

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

8.3 QFN 33 (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

9 REVISION HISTORY

Revision	Date	Description
1.00	2017.02.15	Preliminary version
1.01	2017.12.14	1. Fixed the number of ADC channel in Table 4.1-2. 2. Fixed Internal Reference Voltage to Internal Voltage (Band-gap) in Figure 6.2-7. 3. Removed Note 2 in description of HXTGAIN (CLK_PWRCTL [11:10]) in section 6.3.8. 4. Removed MBS content in description of CBS (Config0 [7:6]) in section 6.4.4.3.
1.02	2018.08.17	1. Removed the ICE_DAT and ICE_CLK description from PF.2 and PF.3 in section 4.2 and 4.3. 2. Fixed the register name in Table 6.8-1 and Table 6.9-1. 3. Fixed the clock source in Figure 6.16-1. 4. Fixed the baud rate setting of Mode 1 in Table 6.16-4 and Table 6.16-5.

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.

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